UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MATEMÁTICA

ÁRBOLES DE DECISIÓN E IDENTIFICACIÓN DE GENES EN BACTERIAS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL MATEMÁTICO

ALONSO TOMÁS GUZMÁN TORO

PROFESOR GUÍA:
SERVET MARTÍNEZ AGUILERA

MIEMBROS DE LA COMISIÓN:
ANDREW HART
ALEJANDRO MAASS SEPÚLVEDA
FELIPE TOBAR HENRÍQUEZ

Este trabajo ha sido parcialmente financiado por CMM-Conicyt PIA AFB170001

SANTIAGO DE CHILE
2018

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO
POR: ALONSO TOMÁS GUZMÁN TORO
FECHA: 2018
PROF. GUÍA: SERVET MARTÍNEZ AGUILERA

ÁRBOLES DE DECISIÓN E IDENTIFICACIÓN DE GENES EN BACTERIAS

El presente trabajo muestra la implementación de técnicas de clasificación basadas en árboles de decisión para resolver y entender el problema de identificación de genes anotados en el ADN de la bacteria *Escherichia Coli*. Junto a lo anterior, se pretenden entender algunos principios biológicos subyacentes tras el mecanismo celular de identificación genética.

Los métodos de clasificación que se implementan en este trabajo intentan simular la manera en que los complejos procesos celulares de transcripción y traducción genética identifican o encuentran las posiciones de inicio de los genes responsables de la posterior síntesis proteica. Se respeta la forma en que esta información es adquirida sin caer en el error de alejarse del marco biológico en cuestión. Para resolver el problema se crearon tres estrategias de clasificación basadas en la combinación de modelos de árboles de decisión y de un algoritmo de optimización sobre el área ocupada en el ADN por zonas génicas.

La primera estrategia consiste en utilizar el algoritmo de optimización sobre candidatos a genes, obtenidos de una lectura secuencial en la doble hebra, para reducir la cantidad de potenciales genes. La solución obtenida es clasificada por los árboles de decisión. La segunda estrategia consiste en realizar el mismo proceso pero usando candidatos obtenidos desde una lectura en ambos sentidos de la doble hebra de ADN. La tercera estrategia consiste en iterar sucesivamente la optimización junto a los árboles utilizando la información incorrectamente clasificada por estos.

Los resultados obtenidos se resumen como un conjunto de candidatos clasificados positivamente por los árboles de decisión y que cumplen con las restricciones impuestas por el algoritmo de optimización.

En la memoria de mi madre Patricia y en el futuro de mi padre Hugo.
La felicidad más grande es haberlos conocido.

Agradecimientos

Es difícil comenzar a escribir los agradecimientos cuando hay tanto que recordar en una etapa de altos y bajos pero que, de manera subliminal puedo decir agradecido, muchos más son los buenos momentos.

De manera no estándar comienzo por agradecer a todo el cuerpo docente que tuve a lo largo de la carrera, desde plan común a la especialidad. Y, aunque fue una etapa muy dificil, inicialmente plagada de incertidumbres, desmotivación y cuestionamientos, valoro todo lo aprendido, incluidos fracasos y éxitos. Al final del camino, sacando cuentas minuciosas, todo tiene un sentido más claro, y eso fue siempre lo que busqué.

Agradezco a mis compañeros de carrera y generación, en especial a aquellos por los que guardo un inmenso cariño y respeto: Mauricio, Enrique, Rodrigo, Francisco, Camilo, Cristóbal (ocho), Ricardo y Roberto. Lamento no haber estado presente en tantos momentos, pero se que mi admiración por sus cualidades como persona nunca mermaron. Son buenos cabros. Agradezco de sobremanera a mis profesores guía de la tesis, profesor Servet Martinez y Andrew Hart. Su paciencia, motivación y amor por lo que hacen produjeron en mi un sentimiento irrefrenable de superación, que creí perder hace eones. El tiempo dedicado, la manera sabia en que me condujeron y la simpatía compartida no me hacen más que ratificar mi admiración por ustedes como personas y profesionales.

Agradezo a los profesores restantes de la comisión por el tiempo dedicado a revisar y corregir la memoria. El tiempo no es algo que se deba subvalorar.

Agradezco de mi etapa universitaria, en escalas astronómicas, a las preciosas personas que conocí a lo largo de un camino lleno de dificultades, pero también con lo mejor que uno puede recibir de la vida como lo son el acompañamiento, amor y amistad. El orden de los factores no altera mi amor fraternal hacia ustedes: Esteban, Álvaro, Pablo y Luis Felipe Osorno, sin duda nada de todo este camino hubiese valido la pena si no nos hubiéramos topado en aquél destello exacto que poca veces brilla en la vida. Los amo cabros, siempre estarán en mi corazón.

Agradezo a la gente que conocí inesperadamente en mi vida pero que terminaron formando parte de ella de una manera muy especial: tía Susana, Tita, Ángela y todos los muchachos de Puente Alto, es una alegría tenerlos aún a mi lado, a pesar que ya no nos veamos tanto. En especial a la primera, su cariño, rica comida y amor me mantuvieron motivado durante muchas noches de estudio y, de seguro que a algunos más también. La quiero mucho.

Agradezco a todas las personas que conocí en distintos grupos universitarios: sección 7 mechona (y sus derivados) y el roble. Guti, Felipe, Javier, Negro, Renex y muchos otros me permitieron reir a carcajadas en incontables ocasiones. Axel, Dani, Potito, Papa, Pin Pón, Alday, Moritas, Bicho y Kevin, el roble fue algo notable y lindo que surgió espontáneamente en medio de un ambiente muchas veces hostil. Aquellas buenas convsersaciones nunca se me olvidarán, ni tampooco toda su alegría y cariño. Los quiero mucho.

Agradezco a los que siempre han estado conmigo desde tiempos ancestrales y que, sin lugar a dudas, conforman una de las cuatro patas de la silla que es mi vida. Mis amigos del colegio: Plutón, Jaña, Rodrigo, Ricardo, Job, Mario, Enzo, Maximiliano y Sebastián. Hemos pasado por tantas cosas juntos que se me hace difícil sintetizar todo en tan poco espacio. Son las personas que me mantuvieron en pie cuando más lo necesité y estoy profundamente agradecido de tenerlos a mi lado. Cada uno de ustedes es un mundo único por explorar pero, que en conjunto, conforman un sistema planetario pleno y lleno de vida, probablemente único en su tipo. Los amo demasiado y sin ustedes no estoy completo.

Agradezco de manera especial a la persona que cada día de mi vida me hace sentir inmensamente feliz desde que la tengo a mi lado: Javi, wawita, sin usted no se que haría. Es el sol que brilla en todo momento para darle el mayor de los sentidos a mi vida. Su apoyo incondicional, amor, tolerancia, empatía y calidez son fundamentales para mi. El sendero caminado junto a usted me hizo mirar al costado para reconocer las infinitas posibilidades que existen allá afuera. Es única y sublime. La amo con todo mi ser.

Agradezco a mi familia. A mi mamá, Patricia, por todas sus enseñanzas, dedicación y amor incondicional, que hasta sus últimos días siempre se mantuvieron intactos. Te extraño inmensamente y se que estarías contenta con lo logrado: te amo infinito. A mi papá, Hugo, un hombre increiblemente cariñoso y que sin el cual no podría respirar. El dolor pasado se mitigó gracias a tu incondicionalidad e inmenso amor. Eres mi ejemplo, el mejor papá del mundo. Te amo infinito. A mis hermanos, por todo el amor que me tienen pero el cual no he sabido corresponder de igual manera. Los amo, en especial a ti Álvaro, ser mágico. A mis tíos, Juan, Alejandro y Elia, por todo su cariño y apoyo en todos estos años: son los mejores.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Introducción

Cada organismo del planeta tiene cualidades únicas que lo diferencian del resto de seres vivos e incluso entre sujetos dentro de su misma especie. Conocer los procesos biológicos que expresan dichas cualidades y que controlan en homeostasis el correcto funcionamiento del ser con su entorno son fundamentales para entender la vida misma, transformándose en una quimera para el ser humano desde que el avance tecnológico ha permitido su estudio.

El campo de la genética es justamente quien toma esta bandera, desentrañando los complejos procesos que acontecen en el núcleo celular, específicamente en su estructura fundamental: el ADN. El estudio del ADN tuvo un gran auge cuando Watson y Crick propusieron su modelo de doble hélice en 1953 y que con posteriores estudios experimentales tuvo su validación. De ahí en adelante desde la creación de la computación se logró avanzar sustancialmente en el entendimiento de estos procesos y en particular del mecanismo de codificación genética, a través de la simulación de modelos matemáticos que lo intentan explicar.

En particular, la identificación de genes en una molécula de ADN ha sido un problema de suma dificultad para los científicos. A través de décadas de trabajo colaborativo e interdisciplinario se han estudiado y logrado clasificar regiones codificadoras correspondientes a genes en organismos en específico, dando lugar a una gran variedad de bases de datos disponibles para su posterior comprensión. Debido a la alta laboriosidad que implica secuenciar el genoma de un organismo de alguna especie (moléculas de ADN con millones de nucleótidos), se hace necesario recurrir a modelos matemáticos y herramientas computacionales que ayuden en el proceso de caracterizar los genes que expresan todas las funciones y comportamientos que definen a un individuo.

Actualmente técnicas basadas en modelos de cadenas de Markov a estados ocultos, como las usadas por el sistema R'HOM (ver [9],[10]) o aquellas basadas en cadenas de Markov interpoladas empleadas por el sistema GLIMMER (ver[12]) han contribuido enormemente en esta tarea de identificación cuando los organismos de estudio son bacterias, virus o arqueas. En particular, esta última técnica encuentra entre un 98 % a 99 % de todas las zonas codificantes en proteínas de gran tamaño. Sin embargo, no todo es identificación por medio de modelos matemáticos. Los resultados arrojados por estos se deben verificar experimentalmente para corroborar que efectivamente constituyen zonas codificantes en cierto organismo de estudio. Este paso es fundamental para la completitud del modelo mismo.

Dada la alta tasa de identificación que consiguen sistemas como los mencionados anteriormente, es natural preguntarse que otro tipo de método podría conseguir mejores resultados. La respuesta es que, si bien es interesante medir el poder de identificación de genes por me-

dio de otros métodos, no siempre es el objetivo principal. Muchos principios biológicos que actúan por detrás de la expresión genética aún resultan ser muy misteriosos para el mundo científico, por lo que resulta fundamental tener una comprensión más profunda de ellos. Así, sin el ánimo de reinventar la rueda, el trabajo presente intenta enfocarse más en el estudio de esta última tarea.

En estos días, los campos del **data mining** y **machine learning** ofrecen una nueva forma de enfrentar problemas en los que se necesitan encontrar patrones muchas veces ocultos en grandes volúmenes de datos. Estos patrones podrían ser cruciales en la tarea de automatizar el proceso de identificación genética. Es precisamente el tamaño del genoma lo que hacía complicado extraer resultados necesarios y valiosos para su posterior estudio y/o aplicación en alguna área de la industria.

Capítulo 1

Preliminares en genómica

En este capítulo se sentan los conceptos fundamentales sobre genómica que serán de utilidad en el entendimiento del problema sobre identificación de genes en un organismo. Esta memoria intenta aportar en dicho problema por medio del estudio de las principales decisiones que lleva a cabo la maquinaria celular para reconocer el correcto inicio de un gen. Dichas decisiones se concretizan específicamente en el proceso de traducción genética donde una secuencia de ARN se utiliza para sintetizar cierta proteina con una funcionalidad particular sobre el individuo. Aquí, la secuencia de ARN es leída linealmente por enzimas que interpretan la información contenida en un gen como una instrucción en la síntesis proteica, por lo que conocer el inicio de un gen es sumamente importante para comprender procesos aún más complejos que se originan en etapas posteriores. Estos mecanismos abarcan todo un campo del conocimiento sobre biología molecular - específicamente en genética - que no se pretende detallar por lo que la información presentada solo aporta en la contextualización y generalidades sobre los procesos que inciden en la síntesis proteica. La información sobre genómica presentada se basa el libro de *Griffiths et al* sobre genética moderna (ver[4]).

1.1. Estructura y organización del ADN / ARN

Todo organismo vivo está conformado por un conjunto de unidades morfológicas y funcionales básicas llamadas células que regulan las funciones vitales de su existencia. Dependiendo de la complejidad del individuo estos pueden ser unicelulares si es que solo tienen una, o pluricelulares en caso contrario. Ejemplos del primer caso son las bacterias y protozoos mientras que del segundo grupo son las plantas y animales tales como el ser humano. La célula en sí misma es la unidad fundamental de la vida por lo que todos los procesos que la definen se llevan a cabo en su interior, constituyendo en si misma un ser vivo. Fenómenos cruciales como la respiración, la ingesta de nutrientes o la reproducción son controlados por ella por medio de mecanismos celulares complejos.

Robert Hooke durante el siglo XVII describió su existencia por primera vez al observar estructuras organizadas en los vegetales. Con el paso del tiempo el avance en la tecnología permitió a científicos como Theodor Schwann y Matthias Schleiden definir los principales

postulados de la teoría celular moderna: todo ser vivo está formado por células siendo la unidad morfológica por excelencia; toda célula deriva de otra precedente a ella; en el interior de la célula ocurren todas las funciones vitales para el funcionamiento del individuo intercambiando energía y materia con su entorno; la célula tiene la información hereditaria necesaria para controlar su propio funcionamiento como para la transmisión a subsiguientes generaciones celulares. Respecto a este último punto, las instrucciones que dirigen el desarrollo y funcionamiento de todo organismo vivo se encuentran almacenadas en la molécula de ADN que además es la responsable de su transmisión hereditaria. El ADN es uno de los dos ácidos nucleicos junto al ARN y es un polímero formado por la repetición lineal de miles a millones de monómeros denominados nucleótidos. Los nucleótidos son moléculas orgánicas formadas por tres subunidades moleculares constituyentes: un monosacárido denominado pentosa, una base nitrogenada y un grupo fosfato. A continuación, se detallan con más profundidad las características de cada uno de estos componentes:

- **Pentosa:** son monosacáridos formados por una cadena de cinco átomos de carbono C y cumple una función estructural. Su fórmula general es $C_5H_{10}O_5$. Dependiendo su función existen muchos tipos de pentosa, pero en los ácidos nucleicos como el ARN y el ADN se diferencian solo en dos tipos respectivamente, la ribosa y la desoxirribosa. Al visualizar una pentosa como una cadena de carbonos numeradas $C1-C2-C3-C4-C5$, la ribosa tiene un grupo hidroxilo (OH) en el carbono $C2$ mientras que la desoxirribosa en la misma posición posee un átomo de hidrógeno (H).

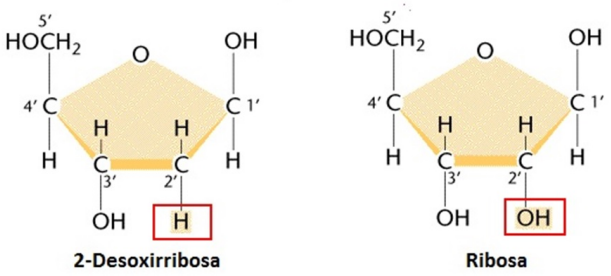


Figura 1.1: Visualización de las pentosas ribosa y desoxirribosa (ver [15]).

- **Grupo fosfato:** es una molécula de fórmula química H_3PO_4 y es fundamental como pieza estructural en una molécula de ADN o ARN pues mediante enlaces químicos con la pentosa de otro nucleótido mantienen la unión del ácido nucleico.
- **Bases nitrogenadas:** son compuestos orgánicos cíclicos que poseen dos o más átomos de nitrógeno (N) en su estructura, de ahí su nombre. Se agrupan en tres grupos: bases isoaloxazínicas, bases púricas y bases pirimidínicas. En el contexto de esta tesis los dos últimos grupos son los más importantes ya que son aquellos presentes en el ADN y ARN. Dentro de las bases púricas están la *adenina* (A) y la *guanina* (G) mientras que en las pirimidínicas están la *timina* (T), la *citosina* (C) y el *uracilo* (U). La representación por las letras antes indicadas es la convención más utilizada para trabajar con ellas y será la adoptada a lo largo de este trabajo. Una característica entres las bases púricas y pirimidínicas es que son complementarias entre sí, es decir, forman enlaces que las mantienes unidas de forma estable. La *adenina* es complementaria con la *timina* a través de la formación de dos puentes de hidrógeno entre ellas mientras que la *guanina* lo es con la *citosina* a través de tres puentes de hidrógeno. Las bases A,G,T,C están presenten

en el ADN mientras que A,G,U,C están en el ARN, es decir, en lugar de *timina* una molécula de ARN presenta *uracilo* lo que se traduce en una complementariedad de la *adenina* con el *uracilo* a través de dos puentes de hidrógeno.

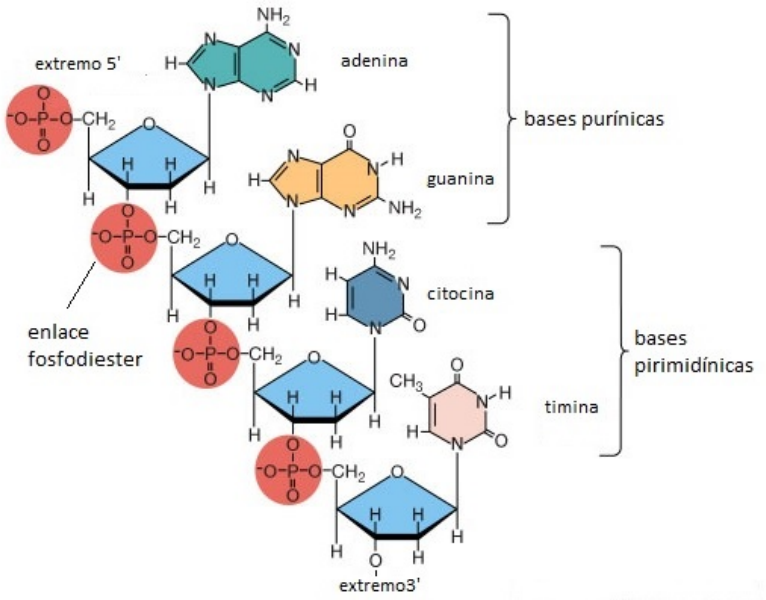


Figura 1.2: Visualización de los grupos de bases nitrogenadas púricas y pirimidínicas (ver[3]).

El ADN es una molécula bicatenaria de dos cadenas de multinucleótidos unidas entre sí por medio de enlaces puente de hidrógeno entre sus bases nitrogenadas de manera tal que se cumple la complementariedad entre los pares de bases. Estructuralmente esta doble cadena se dispone en forma de doble hélice enrollada helicoidalmente en torno a un eje imaginario. Este plegamiento le da a la molécula nuevas funcionalidades respecto a la estructura más básica de una cadena poli nucleotídica que, salvo en algunos virus, no juega ningún papel de control en la célula.

A diferencia del ADN el ARN es una molécula monocatenaria de nucleótidos que tiene por función expresar la información contenida en el ADN para sintetizar las proteínas, macromoléculas formadas por subunidades denominadas aminoácidos que son esenciales en cualquier proceso biológico que involucra la vida de una célula. En el proceso de síntesis proteica participan tres tipos de ARN: el ARN mensajero (ARNm) es quien transfiere la información contenida en el ADN hacia los centros de producción proteica en la célula conocidos como ribosomas; el ARN de transferencia (ARNt) se encarga de llevar los aminoácidos hacia el ribosoma de acuerdo con la información proporcionada por el ARNm; y el ARN ribosómico (ARNr) que es el constituyente principal de los ribosomas y actúa como catalizador en las reacciones que dan lugar a las proteinas.

Una de las preguntas cruciales en lo que tiene que ver con la síntesis proteica es como se organiza la información contenida en una molécula de ADN para poder ser leída y descifrada correctamente en etapas posteriores. La información se organiza dependiendo la complejidad evolutiva de la célula siendo procarionte como las bacterias o eucarionte como animales y

vegetales. La diferencia más fundamental entre ambas es la compartimentalización de las funciones que llevan a cabo en su interior, es decir, la existencia de subestructuras llamadas organelos delimitados por membranas que permiten diferenciar las tareas de la célula, y que están dispuestos sobre un medio filamentoso llamado citoplasma. Las células procariotas no constan de tales compartimentos de forma tal que todas las funciones celulares se desarrollan en el citoplasma mismo y, tal que la información desplegada en el ADN está flotando libremente por él. Por el contrario, las células eucariotas poseen un organelo llamado núcleo celular donde las moléculas de ADN se sitúan actuando como centro de operaciones en el transcurso de la vida celular.

En una molécula de ADN no toda la información secuenciada como nucleótidos se expresa posteriormente en la creación de una proteína en particular. A aquellas subregiones del ADN que reportan efectivamente información codificante de proteinas se les denomina zonas codificantes y a las que no como zonas no codificantes. Es importante señalar que, el que las zonas no codificantes no sean utilizadas posteriormente en forma directa durante la síntesis proteica no significa que no posean algún valor sobre este proceso: se ha visto en investigaciones que dichas regiones son trascendentales en mecanismos regulatorios sobre muchos procesos intracelulares incluidos la formación de proteinas. Las unidades básicas codificantes de información funcional se denominan *genes* y corresponden a subsecuencias de nucleótidos distribuidas a lo largo de diferentes posiciones definidas - o *locus* - en las hebras de una molécula de ADN. Para referirse al conjunto de genes de un organismo en particular se habla del *genoma* del organismo.

1.2. Síntesis Proteica

Para la síntesis proteica la información contenida en los genes debe ser leída por medio de mecanismos celulares que aseguren la correcta expresión de las instrucciones conducentes al ensamblaje de los aminoácidos que darán lugar a la nueva proteina. Los procesos celulares que conforman la expresión genética son la *transcripción* y *traducción* genética.

1.2.1. Transcripción genética

La transcripción es la etapa inicial de la expresión genética donde la información contenida en una molécula de ADN es transformada en ARNm para que posteriormente viaje a los ribosomas donde se iniciará la síntesis proteica. El mecanismo es regulado por una enzima de nombre ARN-polimerasa la cual se encarga de leer una de las dos hebras que componen el ADN con el fin de producir una nueva hebra, copia complementaria de la que es usada como plantilla. Dada la complementariedad entre las hebras no es trascendente cual se usará en la transcripción, pero al momento de seleccionar una cualquiera se hace la siguiente distinción entre ambas: aquella utilizada como plantilla se denomina hebra no codificante mientras que aquella no usada en la transcripción se llama hebra codificante y la razón de esto es que la hebra de ARN producida por la ARN-polimerasa será exactamente igual a la que no fue leída salvo que las *timinas* son reemplazadas por *uracilos*. Específicamente, la ARN-polimerasa se

enlaza a la doble hebra de ADN separando los puentes de hidrógenos entre pares de bases complementarios que las mantenían unidas para empezar a generar una hebra de ARN, complementaria a la usada como molde por medio de la lectura de sus nucleótidos en forma secuencial respetando la complementariedad de las bases.

Una cuestión de suma importancia en la lectura de una hebra de ADN es la dirección escogida para hacerlo. Cada hebra de ADN o ARN tiene dos extremos en que cada uno está asociado a dos posibles posiciones del carbono en la molécula de azúcar presente en los nucleótidos terminales. Si un extremo termina en el tercer carbono se denomina extremo $3'$ y si termina en el quinto se llama extremo $5'$. Dada la complementariedad de las hebras en el ADN, el extremo $3'$ de la hebra de molde coincide con el extremo $5'$ de la hebra codificante y también viceversa, es decir, el extremo $3'$ de esta última coincide con el extremo $5'$ de la hebra de molde. En el proceso posterior de traducción genética, la síntesis proteica ocurre desde el extremo $5'$ al $3'$ adoptándose por convención que en la etapa de transcripción la lectura de la hebra de molde se realiza desde el extremo $3'$ al $5'$ formándose una hebra de ARN copia de la hebra codificante que, al momento de la transcripción, se recorre desde $5'$ a $3'$. El término de la transcripción del ADN a ARNm se da cuando a través de la acción de enzimas específicas se desestabiliza la interacción entre la hebra de molde y la molécula de ARNm en formación, liberándose está última de la ARN-polimerasa. A modo de ejemplo la Figura 1.3 muestra la transcripción de una molécula de ADN en ARNm:

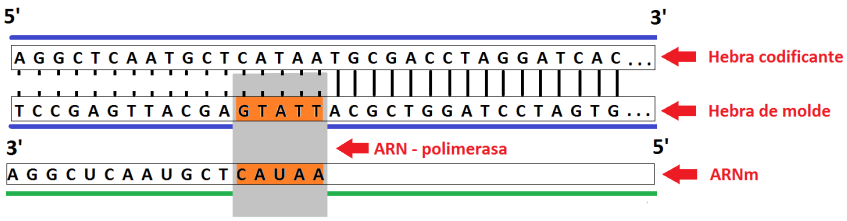


Figura 1.3: Transcripción genética. La ARN-polimerasa va creando una hebra de ARN, copia de la hebra codificante usando su hebra complementaria como molde.

Una vez liberada la nueva molécula de ARNm esta viajará a través de la célula hasta los ribosomas en donde se dará paso a la siguiente etapa de traducción genética. Para células procariotas el ARNm viaja libremente a través del citoplasma, mientras que para las células eucariotas el ARNm debe primero dejar el núcleo celular pasando a través de subestructuras llamadas poros situados en la membrana nuclear, y que se visualizan como verdaderos agujeros que permiten el intercambio de materia y energía con el demás medio intracelular.

1.2.2. Traducción genética

La traducción genética es el conjunto de mecanismos que se encargan de decodificar la información almacenada en el ARNm para producir las cadenas de aminoácidos - también llamados *polipéptidos* - que luego de procesos de plegamiento se transforman en proteínas activas, listas para realizar alguna función específica dentro de la célula. La manera en que

este proceso opera se basa en la forma en que el ribosoma secuencia la información en el ARNm para decidir cuando comienza un gen codificante de alguna proteina y, así discriminar esto del resto de la información que no interviene directamente en la síntesis proteica. Este fenómeno es crucial para entender el problema estudiado en el presente trabajo pues se relacionan directamente, por lo que su entendimiento es fundamental en lo que sigue.

En la naturaleza existen veinte aminoácidos codificados en la forma de una o más subunidades llamadas codones. Los codones son tripletas de nucleótidos y constituyen la unidad básica de información utilizada en el proceso de traducción, por lo que la lectura del ARNm se realiza de codón en codón. Dado que existen cuatro posibles bases nitrogenadas $\{A, G, T, C\}$ para configurar un nucleótido se pueden formar $4^3 = 64$ posibles tripletas de ellas o codones. Los aminoácidos pueden estar codificados por un único codón como es el caso de la *metionina* (ATG) y el *triptófano* (TGG), o por 2, 3, 4 o 6 codones diferentes como es el caso de la *leucina*, *serina* y *arginina*, todos aminoácidos codificados por 6 codones. La abundancia de codones da lugar a una degeneración del código genético pues existen más de los necesarios para codificar toda la información. Del total de codones, $\{TAG, TGA, TAA\}$ no están asociados a ningún aminoácido y su función es meramente informar al ribosoma cuando un gen termina, por lo que naturalmente se les denomina codón de término o *stop codon* por su denominación en inglés (ver[4]).

La traducción del ARNm comienza por la fijación del ribosoma en una posición inicial para comenzar la lectura en la dirección $5'$ a $3'$. En general, tanto para células procariotas como eucariotas, la partida corresponde al primer codón de metionina (ATG) encontrado, razón por la que se le denomina en la literatura comúnmente como codón de inicio o *start codon* (ver[4]). Una vez fijada la partida el ribosoma comienza a traducir codón a codón hasta encontrar en algún momento alguno perteneciente a la familia $\{TAG, TGA, TAA\}$, representando el término de un gen. Se continúa con la lectura por el resto de la hebra de ARNm hasta encontrar otro codón de inicio indicando la partida de un nuevo gen.

La naturaleza del término traducción se debe específicamente a que cuando se está leyendo un gen, el ribosoma situado sobre alguno de los codones envía una señal al resto de su estructura para que un ARN de transferencia (ARNt) se una al aminoácido respectivo transportándolo hasta el sitio activo. Dado que el ARNt es una secuencia corta de nucleótidos, en uno de sus extremos tiene el codón complementario (o anticodón) al codón sobre el que está ocurriendo la traducción, de manera que sabe la posición en el ARNm sobre la cual situarse. Una vez fijado el complejo ARNt-aminoácido sobre el codón traducido, el ribosoma se traslada hacia el próximo repitiendo el proceso antes descrito. En el momento en que llega el nuevo complejo ARNt - aminoácido, el aminoácido de la posición anterior se une a través de enlaces peptídicos al actual, liberándose el ARNt que lo fijaba a su sitio con el propósito de ser usado durante la traducción de un posterior codón. Al encontrar el codón de término al final de la traducción del gen, el ribosoma por medio de factores enzimáticos inhibe la posible traducción de otro codón, cortando la secuencia de aminoácidos formada transportándola hacia otra región donde se complejizará su estructura espacial para añadirle la funcionalidad que la transformará en proteína.

La Figura 1.4 ejemplifica la etapa de traducción del ARNm mostrando como actúa el ribosoma sobre cada codón a través del complejo ARNt - aminoácido:

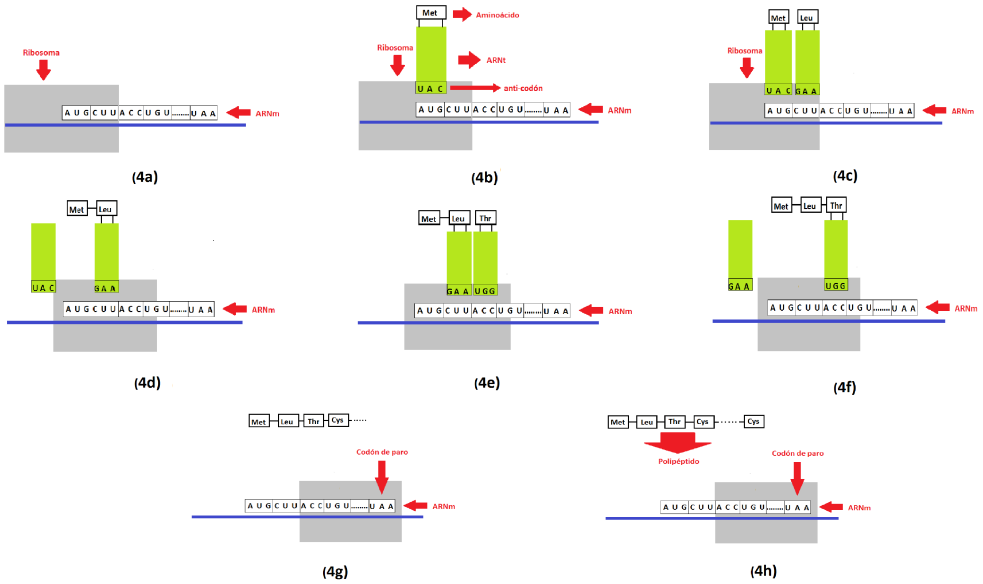


Figura 1.4: Traducción genética. En (4a) el ribosoma se sitúa sobre el codón de inicio. De (4b) a (4f) se describe la traducción del codón en un aminoácido utilizando ARNm para transportarlo. En (4g) y (4h) el ribosoma termina la traducción al encontrarse con el codón de término liberando el polipéptido formado.

Cuando el ribosoma comienza la traducción sobre el ARNm tiene tres posibles maneras de hacerlo de acuerdo con la posición del primer nucleótido dentro del codón que emplea como inicio para empezar la lectura. Al pensar la hebra de ARNm como una secuencia numerada de nucleótidos desde el primero en el extremo $5'$ hasta el n-ésimo en el extremo $3'$ en caso de que esta posea una cantidad total de $n \in \mathbb{N}$ nucleótidos, el ribosoma se situaría inicialmente en el codón cuyos nucleótidos, digamos, ocupan las posiciones dadas por el triplete $(j, j+1, j+2)$ respectivamente. Entonces, la lectura de codones se puede realizar en cualquiera de las siguientes formas: se comienzan a leer desde la posición j-ésima por lo que los codones traducidos ocuparán las posiciones dadas por $(j, j+1, j+2)$ con $j \geq 1$; se comienza a leer desde la posición $(j+1)$-ésima de modo que los codones ocuparán las posiciones $(j+1, j+2, j+3)$; se comienza a leer desde la posición $(j+2)$-ésima con lo que los codones ocuparán las posiciones $(j+2, j+3, j+4)$ dentro de la hebra (ver[6]). A cada una de estas formas de lectura se les denomina marcos de lectura o *Reading Frames* por su nombre en inglés y cabe notar que son los únicos posibles pues si se comienza la lectura en la posición $j+3$ se está utilizando el mismo marco que para la j-ésima. Como el ADN es una molécula de doble hebra existen tres marcos de lectura por cada una dado que cualquiera se escoge para el proceso de traducción genética. Si la complementaria es elegida la lectura durante la traducción en su dirección $5'$ a $3'$ corresponde a la dirección $3'$ a $5'$ en la primaria debido a que son antiparalelas la una con la otra. Una vez iniciada la traducción de un gen en un marco de lectura, este se mantiene hasta el final del proceso evitando la superposición de codones por lo que generalmente el largo de los genes es un múltiplo de tres. Al marco de lectura que da origen a la posterior síntesis de una proteina se le denomina marco de lectura abierto (ORF en inglés). Si bien la superposición de marcos de lectura en un mismo ORF no es la regla, si existen casos donde la superposición de genes tiene lugar. Tal es el caso de virus, células procariotas y genomas mitocondriales en células eucariotas.

1.3. Regiones Promotoras

Durante la etapa de transcripción genética la proteina ARN - polimerasa va codificando la información almacenada en los nucleótidos para producir una copia de la hebra codificante de la molécula de ADN. Una pregunta trascendental que entender es: ¿cómo la ARN - polimerasa sabe en qué momento comenzar a transcribir un gen? y aún más específicamente, ¿cómo reconoce el codón de inicio de un gen?. La respuesta está en cómo se lleva a cabo el proceso de transcripción genética. La ARN - polimerasa lee de manera secuencial la hebra de molde en el ADN en cualquiera de los marcos de lectura en dirección $5' \rightarrow 3'$. Al momento de transcribir un gen la enzima sabe exactamente en qué momento hacerlo y, por tanto, la información alerta sobre la presencia de dicha zona génica se debe encontrar en las regiones previas en sentido $5'$ a este. Las regiones que almacenan dicha información se conocen en la literatura como promotores (promoters en inglés) o zonas promotoras y se configuran como pequeños trozos o secuencias de nucleótidos en diferentes posiciones en las regiones intergénicas del ARNm.

Las secuencias que conforman las zonas promotoras y sus posiciones relativas en las zonas intergénicas previas al inicio de los genes se han conservado a lo largo de la evolución y, aunque no tienden a presentarse de igual manera en todos los organismos celulares, los estudios en el área han mostrado que ciertas variaciones se manifiestan en un gran número de especímenes. Dada las diferencias evolutivas entre células procariontes y eucariontes, sus respectivas zonas promotoras son muy diferentes y por tanto su estructura y análisis conlleva el uso de herramientas totalmente diferentes. Por el alcance de esta memoria y dado lo complejo que resultan ser los procesos que intervienen en la traducción y transcripción genética en los organismos eucariontes, se describen únicamente los promotores en células procariontes.

En organismos como las bacterias, las zonas promotoras se presentan principalmente como dos secuencias de nucleótidos situadas en las zonas intergénicas previas a la etapa de transcripción de genes. Dichas secuencias se encuentran aproximadamente 10 y 35 nucleótidos previos a los sitios que dan comienzo a la transcripción de los genes (ver[5]). La identificación de dichas secuencias se realiza por medio del estudio de las secuencias de consenso de las zonas intergénicas en las distintas especies de bacterias. Las secuencias de consenso dan cuenta de cuáles son los nucleótidos más frecuentes de encontrar en ciertas posiciones de interés en una hebra de ADN y se obtienen al alinear muchas secuencias de estudio en torno a dichas posiciones, por lo que corresponden a secuencias ideales difíciles de encontrar de manera natural en todos los organismos.

- **Zona 10:** También conocida como Caja de Pribnow corresponde a la secuencia de seis nucleótidos que en su forma idealizada es $5' - TATAAT - 3'$. Se ubica idealmente en torno a la posición 10 previa al inicio de un gen. A continuación, se muestra la probabilidad posición a posición de encontrar cada una de las bases que representa la secuencia de consenso del promotor Zona 10 (ver[5]).
- **Zona 35:** En torno a la posición 35 previa al inicio de un gen la secuencia de consenso resulta ser $5' - TTGACA - 3'$ cuyas frecuencias por nucleótidos se muestran a continuación (ver[5]).

Además de las secuencias descritas anteriormente algunos promotores contienen las secuencias

T	**A**	**T**	**A**	**A**	**T**
82 %	89 %	52 %	59 %	49 %	89 %

Tabla 1.1: Frecuencias de ocurrencia para las bases encontradas en la secuencia de consenso del promotor Zona 10.

T	**T**	**G**	**A**	**C**	**A**
69 %	79 %	61 %	56 %	54 %	54 %

Tabla 1.2: Frecuencias de ocurrencia para las bases encontradas en la secuencia de consenso del promotor Zona 35.

de consenso $5' - AAAAAARNR - 3'$ y $5' - AWWWWWTTTTT - 3'$ ubicadas alrededor de las posiciones 42 (ver[11]) y 52 (ver[2]) previas al sitio de inicio de transcripción genética respectivamente. En este caso $W = A/T$, $R = A/G$ y N puede ser cualquier base.

Capítulo 2

Modelamiento Matemático

Clasificar objetos a través de la observación de sus características ha sido una actividad que ha acompañado al ser humano a lo largo de su historia. En el afán de simplificar la vida se ha transformado en un acto casi reflejo segmentar todo lo que nos rodea en paquetes específicos de información de acuerdo con cierto criterio escogido. Con esta idea en mente se han propuesto desde la matemática una serie de herramientas que ayuden en esta tarea de clasificación. A mediados de la década de los años ochenta, Breiman y Olshen (ver[1]) sentaron la formulación matemática del algoritmo que hoy se conoce como CART, *Classification and Regression Trees*. La idea tras su creación era entender y simular el modo en que las personas toman decisiones para llevar a cabo una clasificación exitosa.

En este capítulo se introducen las principales ideas tras el funcionamiento de los árboles de decisión o clasificación. En la primera sección se dan las definiciones básicas tras el concepto de clasificador y de cómo estos se utilizan en la tarea de segmentar conjuntos de observaciones. En la segunda sección se describe el concepto de impureza de una partición que será importante en la construcción de los árboles de decisión. En la tercera sección se formaliza la etapa de construcción de los árboles de clasificación y cómo estos son generalizables en la tarea de predecir observaciones. En el último capítulo se sentan las bases teóricas del modelo matemático de una molécula de ADN. Se presenta también un algoritmo de optimización sobre la organización de candidatos a genes en una molécula de ADN bacteriano desarrollado por Hart, Martínez y Videla (ver[6]), que será de suma relevancia a lo largo de este trabajo. La mayor parte de la notación referente a árboles de clasificación corresponde a aquella utilizada por Breiman en su libro sobre CART (ver[1]). La notación e ideas tras el concepto de impureza de una partición son tomadas de los apuntes preliminares desarrollados por Martínez y Hart (ver [7]). Finalmente, la notación tras el modelo matemático de la doble hebra de ADN como la respectiva al algoritmo de optimización son tomadas del artículo de Hart, Martínez y Videla (ver[6]).

2.1. Preliminares sobre Clasificadores

Sean $\mathfrak{X}$ un conjunto finito arbitrario y $\mathfrak{J} = \{1, ..., J\}$ un conjunto finito tal que $J \in \mathbb{N}$. Se considera el espacio de probabilidad $(\mathfrak{X} \times \mathfrak{J}, \sigma(\mathfrak{X} \times \mathfrak{J}), \mathbb{P})$ donde la medida de probabilidad $\mathbb{P}$ se toma sobre la sigma-álgebra $\sigma(\mathfrak{X} \times \mathfrak{J})$. Consideremos además la distribución empírica p construida a partir de $\mathfrak{X}$. Al conjunto $\mathfrak{X}$ se le denomina espacio de estados y a $\mathfrak{J}$ se le denomina conjunto de clases. El espacio de estados contiene todas las posibles mediciones que se pueden realizar a través de una observación del objeto de estudio y por tanto se pretende establecer un método sistemático por el cual asociar una clase de $\mathfrak{J}$ a cada elemento en $\mathfrak{X}$.

Definición 2.1 *Un clasificador o regla de clasificación es una función* $\mathrm{d} : \mathfrak{X} \to \mathfrak{J}$*, es decir, a cada* $x \in \mathfrak{X}$ *el clasificador le asigna una de las* J *clases existentes en* $\mathfrak{J}$*.*

El clasificar objetos no es un acto sin lógica sino por el contrario se basa en un conocimiento previo de elementos ya observados y estudiados. En la construcción de un clasificador d dicha experiencia previa se condensa en un conjunto $\mathfrak{L}$ denominado conjunto de aprendizaje o entrenamiento (ver [1]).

Definición 2.2 *Un conjunto de aprendizaje o entrenamiento consiste en un conjunto finito de la forma* $\mathfrak{L} = \{(x_{\mathrm{i}}, j_{\mathrm{i}})\}_{\mathrm{i}=1}^{N}$ *donde* $(x_{\mathrm{i}}, j_{\mathrm{i}}) \in \mathfrak{X} \times \mathfrak{J} \quad \forall \mathrm{i} \in [N]$.

Una pregunta relevante respecto a los clasificadores es "¿qué tan buen clasificador es?", o en otras palabras, cuan preciso es su poder predictivo. Una manera de cuantificar lo anterior consiste en probar el clasificador sobre algún conjunto de observaciones de las cuales se conoce a priori la clase $j \in \mathfrak{J}$ a la que pertenece cada una. Conociendo la clase correcta y la clase predicha se puede estimar la precisión de un clasificador como la proporción de los casos mal clasificados sobre el total. A esta cantidad se le denomina tasa de clasificación errónea para el clasificador d, y se denota por $R^*(\mathrm{d})$ (ver [1]).

Para $A \subseteq \mathfrak{X}$ y $j \in \mathfrak{J}$ interpretamos $p(A, j)$ como la probabilidad que al escoger una observación $x \in \mathfrak{X}$, entonces que $x \in A$ y que su clase sea j. Luego, desde el conjunto de aprendizaje $\mathfrak{L}$ (extraído desde la distribución $p(A, j)$), se construye el clasificador $\mathrm{d}(x)$ y se define la tasa de clasificación errónea R^*_{d} como:

Definición 2.3 *Sea* $(X, y) \in \mathfrak{X} \times \mathfrak{J}$ *una muestra extraída desde la distribución de probabilidad* $p(A, j)$*, es decir,* $p(X \in A, Y = j) = p(A, j)$ *y además* (X, y) *es independiente a* $\mathfrak{L}$*. Entonces se define:*

$$R^*(\mathrm{d}) = p(\mathrm{d}(X) \neq Y \mid \mathfrak{L})$$

Observación Es importante notar que la tasa de clasificación errónea $R^*(\mathrm{d})$ se define condicional al conjunto de aprendizaje puesto que el clasificador d se construyó a partir del conjunto $\mathfrak{L}$.

Tras definir $R^*(\mathrm{d})$, un problema que surge es como estimar su valor utilizando las observaciones en el conjunto de estados. La manera natural de de hacer esto es construir un conjunto de aprendizaje $\mathfrak{L}$ a través de un muestreo desde una distribución conocida. Luego de construir un clasificador d en base al conocimiento obtenido de $\mathfrak{L}$, se extrae un nuevo sampleo (y desde la misma distribución) independiente de $\mathfrak{L}$ y suficientemente grande para que d los clasifique. Al calcular la razón de casos erróneamente clasificados se tiene una estimación de $R^*(\mathrm{d})$.

El problema del método anterior es que en general, la cantidad de datos disponibles solo permite construir el conjunto de aprendizaje $\mathfrak{L}$ por razones que se traducen en la complejidad de contar con nuevas observaciones. Dado esto, el conjunto $\mathfrak{L}$ debe ser utilizado no solo en la construcción del clasificador d sino también en la estimación de la tasa de clasificación errónea $R^*(\mathrm{d})$. En vista de esta limitante, a continuación se presentan tres métodos de estimación para el factor $R^*(\mathrm{d})$ (ver [1]):

- **Estimador de resustitución:** Luego que el clasificador d es construido mediante el conjunto de aprendizaje, los mismos casos en $\mathfrak{L}$ son utilizados para estimar $R^*(\mathrm{d})$. La tasa de casos mal clasificados se denomina estimador de resustitución y denotado por $R(\mathrm{d})$, el cual tiene la siguiente forma:

$$R(\mathrm{d}) = \frac{1}{N}\sum_{\mathrm{i}=1}^{N} 1_{\{\mathrm{d}(x_\mathrm{i})\neq j_\mathrm{i}\}} \tag{2.1}$$

- **Estimador de conjunto de prueba:** El conjunto de aprendizaje $\mathfrak{L}$ se divide en dos subconjuntos denotados por $\mathfrak{L}_1$ y $\mathfrak{L}_2$. El primero se usa esclusivamente para construir la regla de clasificación d, por lo que se le suele llamar conjunto de entrenamiento, mientras que el segundo se utiliza para estimar el valor de $R^*(\mathrm{d})$. A este último subconjunto se le conoce como conjunto de test o prueba. Suponiendo que $|\mathfrak{L}_2| = N_2$, entonces el estimador de conjunto de prueba, $R^{cp}(\mathrm{d})$, está dado por:

$$R^{cp}(\mathrm{d}) = \frac{1}{N_2}\sum_{(x_\mathrm{i},j_\mathrm{i})\in\mathfrak{L}_2} 1_{\{\mathrm{d}(x_\mathrm{i})\neq j_\mathrm{i}\}} \tag{2.2}$$

- **Estimador de validación cruzada de K-conjuntos**: Del conjunto $\mathfrak{L}$ se extrae aleatoriamente una partición de K subconjuntos $\mathfrak{L}_m$ con $m \in \{1, ..., K\}$ con aproximadamente el mismo número de observaciones. Para cada $m = 1, ..., K$ se ejecuta el procedimiento explicado en el estimador de conjuntos de prueba, usando como conjunto de entrenamiento $\mathfrak{L} \setminus \mathfrak{L}_m$ y como conjunto de prueba $\mathfrak{L}_m$. Los clasificadores construidos sobre cada conjunto de entrenamiento se denotan por d^m y el estimador para la tasa de clasificación errónea de cada clasificador sobre el respectivo conjunto de prueba $\mathfrak{L}_m$, corresponde al estimador $R^{cp}(\mathrm{d}^m)$ dado por:

$$R^{cp}(\mathrm{d}^m) = \frac{1}{N_m} \sum_{(x_\mathrm{i}, j_\mathrm{i}) \in \mathfrak{L}_m} 1_{\{\mathrm{d}^m(x_\mathrm{i}) \neq j_\mathrm{i}\}} \tag{2.3}$$

donde $N_m = |\mathfrak{L}_m|$ talque $N_m \approx N/K \ \forall m = 1, ..., K$. Al construir un clasificador d utilizando todo $\mathfrak{L}$ se asume que a mayor valor de K cada tasa de clasificación errónea $R^*(\mathrm{d}^m)$ se aproxima a $R^*(\mathrm{d})$, puesto que d^m se construye usando el conjunto $\mathfrak{L} \setminus \mathfrak{L}_\mathfrak{m}$ con un tamaño cercano a $\mathfrak{L}$ y de valor aproximado $N - N_m \approx N - N/K = N(1 - 1/K)$. Bajo la consideración previa, se define el estimador de validación cruzada de d, $R^{cv}(\mathrm{d})$, como:

$$R^{cv}(\mathrm{d}) = \frac{1}{K} \sum_{m=1}^{K} R^{cp}(\mathrm{d}^m) \tag{2.4}$$

2.2. Impureza sobre una partición

Para cada $\mathrm{i} \in \mathfrak{J}$ se define el conjunto $\mathfrak{X}_\mathrm{i}$ como el subconjunto de $\mathfrak{X}$ tal que sus elementos pertenecen a la clase i. Lo anterior define la partición $\mathfrak{F} = \{\mathfrak{X}_\mathrm{i} : \mathrm{i} \in \mathfrak{J}\}$ sobre $\mathfrak{X}$. La idea tras los árboles de decisión como método de clasificación consiste en reconstruir la partición $\mathfrak{F}$ a través de una secuencia de preguntas realizadas sobre las variables que definen una observación en $\mathfrak{X}$. Estas preguntas dividen consecutivamente el espacio de estados generando una partición sobre él, a través de la cual quedará determinado el clasificador tras una cantidad finita de pasos. La elección de preguntas se basa en el concepto de impureza de una partición.

Sea $\mathrm{S}_J = \left\{\vec{p} = (p_\mathrm{i} : \mathrm{i} = 1, ..., J) : \vec{p} \geq 0 \ , \ \sum_{\mathrm{i}=1}^{J} p_\mathrm{i} = 1\right\}$ el simplex de vectores de probabilidad sobre $\mathfrak{J}$. Se llama estado puro al vector $\vec{\delta^\mathrm{i}} \in \mathrm{S}_J$ tal que tiene un 1 en la coordenada i-ésima y 0 en el resto de coordenadas (ver [7]).

Definición 2.4 *Una función de impureza es una función $\phi : \mathrm{S}_J \to \mathbb{R}_+$ no negativa con las siguientes propiedades:*

1. $\phi(\vec{\delta^\mathrm{i}}) = 0 \ \ \forall \mathrm{i} \in \mathfrak{J}$
2. *ϕ alcanza su máximo en el punto $(\frac{1}{J}, \frac{1}{J}, ..., \frac{1}{J})$.*
3. *ϕ es simétrica en $p_1, ..., p_J$.*

Definición 2.5 *Sea $\mathrm{e} \subseteq \mathfrak{X}$ tal que $p(\mathrm{e}) > 0$. Se define $\vec{\rho}(\mathrm{e}) \in \mathrm{S}_J$ por $\rho(\mathrm{e})_\mathrm{i} = p(\mathfrak{X}_\mathrm{i} \mid \mathrm{e}) \ \ \forall \mathrm{i} \in \mathfrak{J}$, es decir, la probabilidad de que una observación esté en la clase* i *dado que pertenece al*

conjunto e. *Entonces de define la impureza de* e *por (ver [7]):*

$$\mathrm{I}(\mathrm{e}) := \begin{cases} \phi(\vec{\rho}(\mathrm{e})) & si \quad p(\mathrm{e}) > 0 \\ \\ 0 & si \quad p(\mathrm{e}) = 0 \end{cases} \tag{2.5}$$

Consideremos ahora $\mathrm{U} = (u_{\mathrm{i}} : \mathrm{i} = 1, ..., m)$ una partición cualquiera de $\mathfrak{X}$. Entonces se define su impureza como:

$$\mathrm{I}(\mathrm{U}) = \sum_{\mathrm{i}=1}^{m} p(u_{\mathrm{i}})\mathrm{I}(u_{\mathrm{i}}) = \sum_{\mathrm{i}=1}^{m} p(u_{\mathrm{i}})\phi(\vec{\rho}(u_{\mathrm{i}})) \tag{2.6}$$

Para $\mathrm{e} \subseteq \mathfrak{X}$, consideremos la partición de e vía U dada por $\mathrm{U} \cap \mathrm{e} = \{u_{\mathrm{i}} \cap \mathrm{e} : \mathrm{i} = 1, ..., m\}$, entonces para todo $j \in \mathfrak{J}$:

$$\rho(\mathrm{e})_j = p(\mathfrak{X}_j \mid \mathrm{e}) = \sum_{\mathrm{i}=1}^{m} p(\mathfrak{X}_j \cap u_{\mathrm{i}} \mid \mathrm{e}) \tag{2.7}$$

$$= \sum_{\mathrm{i}=1}^{m} \frac{p(\mathfrak{X}_j \cap u_{\mathrm{i}} \cap \mathrm{e})}{p(u_{\mathrm{i}} \cap \mathrm{e})} \; \frac{p(u_{\mathrm{i}} \cap \mathrm{e})}{p(\mathrm{e})} \tag{2.8}$$

$$= \sum_{\mathrm{i}=1}^{m} p(\mathfrak{X}_j \mid u_{\mathrm{i}} \cap \mathrm{e}) p(u_{\mathrm{i}} \mid \mathrm{e}) \tag{2.9}$$

$$= \sum_{\mathrm{i}=1}^{m} p(u_{\mathrm{i}} \mid \mathrm{e}) \rho(u_{\mathrm{i}} \cap \mathrm{e})_j \tag{2.10}$$

Cuando la función de impureza $\phi : \mathrm{S}_J \to \mathbb{R}_+$ es cóncava y además al notar que $\sum_{\mathrm{i}=1}^{m} p(u_{\mathrm{i}} \mid \mathrm{e}) = 1$ se obtiene:

Observación La interpretación de la sentencia anterior es que cualquier tipo de partición siempre genera menor impureza respecto del conjunto original no particionado.

Ejemplos de funciones de impureza cóncavas son:

1. **Función de entropía:** $h(\vec{p}) = -\sum_{\mathrm{i}=1}^{J} p_{\mathrm{i}} \log(p_{\mathrm{i}})$
2. **Función de Gini:** $g(\vec{p}) = \sum_{\mathrm{i} \neq j} p_{\mathrm{i}} p_j$
3. **Probabilidad de clasificación errónea:** $r(\vec{p}) = 1 - \mathrm{máx}_{\mathrm{i}=1,...J}\, p_{\mathrm{i}}$

Una vez definida la noción de impureza tanto de subconjuntos del espacio de estado como de particiones sobre este, se introduce el concepto de cadenas de partición, esencial en la construcción de un árbol de clasificación.

El reconstruir la partición original dada por $\mathfrak{F}$ se basa en la idea de generar una serie de particiones sobre $\mathfrak{X}$ de forma que al generar una nueva partición se disminuya lo máximo posible la impureza con respecto a la anterior. Para realizar una partición se debe hacer una pregunta sobre alguna variable en un nodo de la partición. El conjunto de posibles preguntas se denota por Q y tienen una forma estándar de acuerdo con Breiman (ver [1]):

1. Cada partición sobre algún átomo depende del valor de una única variable.
2. Sobre cada variable numérica x_{i}, Q incluye todas las preguntas de la forma "¿es $x_{\mathrm{i}} \leq c$?" $\forall c \in \mathbb{R}$
3. Si x_{i} es una variable categórica tomando valores en $\{b_1, b_2, ..., b_L\}$, entonces Q incluye todas las preguntas de la forma "¿es $x_{\mathrm{i}} \in S$?" $\forall S \in 2^{\{b_1,...,b_L\}}$

Formalmente, cada pregunta $q \in Q$ tiene m posibles respuestas, las que generan una partición sobre $\mathfrak{X}$ de la forma $\mathfrak{F}^q = \{X_{\mathrm{i}}^q : \mathrm{i} = 1, ..., m\}$. De lo anterior se desprende que para un subconjunto $\mathrm{e} \subseteq \mathfrak{X}$ se genera una partición $\mathfrak{F} \cap \mathrm{e} = \{\mathrm{e}_{\mathrm{i}}^q := \mathfrak{X}_{\mathrm{i}}^q \cap \mathrm{e} : \mathrm{i} = 1, ..., m\}$. Si E representa una partición cualquiera de $\mathfrak{X}$ y $\mathrm{e} \in \mathrm{E}$ un átomo cualquiera de esta, se define la partición $\mathfrak{G}(\mathrm{E}, \mathrm{e}, q)$ mediante la pregunta q por los átomos (ver [7]):

$$\mathfrak{G}(\mathrm{E}, \mathrm{e}, q) = \left\{\mathrm{e}^{'} \in \mathrm{E} : \mathrm{e}^{'} \neq \mathrm{e}\right\} \cup \{\mathrm{e}_{\mathrm{i}}^q = \mathfrak{X}_{\mathrm{i}}^q \cap \mathrm{e} : 1, ..., m\} \tag{2.11}$$

es decir, son los átomos en E menos $\mathrm{e} \in \mathrm{E}$, el cual fue particionado por la pregunta q en $\mathfrak{F}^q \cap \mathrm{e}$. Al calcular la diferencia de impureza entre ambas particiones se obtiene que:

$$\begin{aligned}
\mathrm{I}(\mathrm{E}) - \mathrm{I}(\mathfrak{G}(\mathrm{E}, \mathrm{e}, q)) &= \sum_{\mathrm{e}^{'} \in \mathrm{E}} p(\mathrm{e}^{'})\mathrm{I}(\mathrm{e}^{'}) - \sum_{\mathrm{e}^{'} \in \mathfrak{G}(\mathrm{E},\mathrm{e},q)} p(\mathrm{e}^{'})\mathrm{I}(\mathrm{e}^{'}) && (2.12)\\
&= p(\mathrm{e})\mathrm{I}(\mathrm{e}) - \sum_{\mathrm{i}=1}^{m} p(\mathrm{e}_{\mathrm{i}}^q)\mathrm{I}(\mathrm{e}_{\mathrm{i}}^q) && (2.13)\\
&= \sum_{\mathrm{i}=1}^{m} p(\mathrm{e}_{\mathrm{i}}^q)\mathrm{I}(\mathrm{e}) - \sum_{\mathrm{i}=1}^{m} p(\mathrm{e}_{\mathrm{i}})\mathrm{I}(\mathrm{e}_{\mathrm{i}}^q) && (2.14)\\
&= \sum_{\mathrm{i}=1}^{m} p(\mathrm{e}_{\mathrm{i}}^q)(\mathrm{I}(\mathrm{e}) - \mathrm{I}(\mathrm{e}_{\mathrm{i}}^q)) && (2.15)
\end{aligned}$$

De lo anterior, usando la expresión 2.14 se concluye que $\mathrm{I}(\mathrm{E}) - \mathrm{I}(\mathfrak{G}(\mathrm{E}, \mathrm{e}, q)) \geq 0$. Es importante notar que la diferencia de impureza anterior solo depende de las impurezas del nodo e y de su partición $\{\mathrm{e}_{\mathrm{i}}^q\}_{\mathrm{i}=1}^m$.

Sea $(\mathrm{E}(k) : k \geq 0)$ una cadena de particiones tal que $\mathrm{E}(0) = \{\mathfrak{X}\}$, es decir, $\mathfrak{X}$ es el único átomo de E(0). Sea $\mathrm{E}(k)$ la partición en un nivel k-ésimo. Para generar una nueva partición que reduzca la mayor impureza posible desde $\mathrm{E}(k)$ se deben escoger adecuadamente tanto el átomo de $\mathrm{E}(k)$ como la pregunta $q \in Q$ que lo logren. De lo anterior, se deduce que $\mathrm{E}(k+1) = \mathfrak{G}(\mathrm{E}(k), \mathrm{e}(k), q(k))$ donde $(\mathrm{e}(k), q(k))$ se escogen tal que se maximiza la diferencia $\mathrm{I}(\mathrm{E}(k)) - \mathrm{I}(\mathrm{E}(k+1))$, es decir, de acuerdo con el siguiente principio de optimización:

$$\Delta \mathrm{I}(\mathrm{e}(k), q(k)) = \max_{(\mathrm{e},q)\in \mathrm{E}(k)\times Q} \mathrm{I}(\mathrm{E}(k)) - \mathrm{I}(\mathrm{E}(k+1)) \tag{2.16}$$

$$= \max_{(\mathrm{e},q)\in \mathrm{E}(k)\times Q} \sum_{\mathrm{i}=1}^{m} p(\mathrm{e}_{\mathrm{i}}^{q})(\mathrm{I}(\mathrm{e}) - \mathrm{I}(\mathrm{e}_{\mathrm{i}}^{q})) \tag{2.17}$$

$$= \sum_{\mathrm{i}=1}^{m} p(\mathrm{e}(k)_{\mathrm{i}}^{q(k)})(\mathrm{I}(\mathrm{e}(k)) - \mathrm{I}(\mathrm{e}(k)_{\mathrm{i}}^{q(k)})) \tag{2.18}$$

2.3. Árboles de Decisión

En la sección anterior se establecieron los conceptos de impureza y de cadena de partición, fundamentales en la formulación de un árbol de decisión. Tal como se menciona al inicio de este capítulo, la idea tras un árbol consiste en ir particionando sucesivamente el conjunto $\mathfrak{X}$ a través de las preguntas pertinentes que hagan disminuir la impureza en un átomo tanto como se pueda. Tras una cantidad arbitraria de pasos se obtendrá la partición final que permitirá asignar a cada átomo una clase de acuerdo con un criterio aún por definir. Toda esta construcción conforma lo que se denomina árbol de decisión y constituye en sí mismo una regla de clasificación, puesto que siguiendo el camino definido por la cadena de particiones desde la raíz $\mathfrak{X}$ podemos asociar cada observación en $\mathfrak{X}$ a uno de los átomos de la partición final y así a una clase final.

En un m-árbol $T = (V, F, r)$ con m respuestas por pregunta, V representa el conjunto de vértices, $F \subseteq V^2$ el conjunto de aristas y r la ráiz (ver [7]). Cada vértice $a \neq r$ tiene un único vértice padre y m vértices hijos denotados por $a^{\mathrm{i}}, \mathrm{i} = 1, ..., m$. Un vértice $a \in V$ que no tiene sucesores se denomina hoja y el conjunto de hojas se denota por H. Claramente los vértices interiores o no hojas son aquellos en $V \setminus \mathrm{H}$.

El vínculo con el concepto de impureza de una partición se observa al notar que los vértices $a^{\mathrm{i}}, \mathrm{i} = 1, ..., m$ corresponden a una partición del vértice padre a. Es decir, a se ve como un átomo $\mathrm{e}(k)$ perteneciente a cierta partición $\mathrm{E}(k)$ tal que por medio de la pregunta $q(k)$ se particiona en $\left\{\mathrm{e}(k)_{\mathrm{i}}^{q(k)}\right\}_{\mathrm{i}=1}^{m}$ formando así la nueva partición $\mathrm{E}(k+1)$ (ver [7]). El árbol se interpreta entonces como una familia de particiones $(\mathrm{E}(k) : k \geq 0)$, cuya construcción se detiene cuando se satisface el siguiente criterio de parada:

$$\mathrm{I}(\mathrm{E}(k+1)) - \mathrm{I}(\mathrm{E}(k)) \leq \varepsilon \tag{2.19}$$

para cierto $\varepsilon > 0$ definido desde un comienzo. Lo anterior dice que un árbol para de dividirse cuando no es posible seguir disminuyendo la impureza de sus vértices en una cantidad significativa. Esto forma una cadena de particiones $(\mathrm{E}(k) : 0 \leq k \leq K)$, donde la partición final $\mathrm{E}(K)$ representa a los vértices o nodos terminales del árbol, definidos anteriormente por H. La regla de parada antes definida es una de las tantas que se pueden imponer. Otra muy

usada, sobretodo en las implementaciones numéricas, corresponde a definir un nodo como terminal cuando el número de observaciones en él no supera cierto mínimo preestablecido, dado por cierto $N^* \in \mathbb{N}$ (ver [1]).

Se define la clase de conjuntos $\mathrm{E}^* = \{\mathrm{e}(k) : 0 \leq k \leq K-1\} \cup \{\mathrm{e} \in \mathrm{E}(K)\}$ (ver [7]). E^* está formada por todos los átomos o vértices $\mathrm{e}(k)$ que se dividen por medio de la pregunta $q(k) \in Q^*$ y por todos aquellos vértices que no lo hace, es decir, las hojas o nodos terminales del árbol. A cualquier nodo terminal - denotado por $\mathrm{e}(K)$ - se le asocia una pregunta vacía $q(K) = q^* \notin Q$. Se define así $Q^* = Q \cup \{q^*\}$.

Entonces, en el contexto de árboles de decisión un vértice $a \in V$ es un par $a = (\mathrm{e}_a, q_a)$ con $q_a \in \mathrm{E}^*$ y $q_a \in Q^*$. La raíz $r = (\mathrm{e}_r, q_r)$ está dada por $\mathrm{e}_r = \mathrm{e}(0) = \mathfrak{X}$, es decir, el único átomo de la partición inicial $\mathrm{E}(0)$ en la cadena y $q_r = q(0)$ es la pregunta que divide $\mathrm{e}(0)$. Para cada uno de los m sucesores r^{i} de r con $\mathrm{i} = 1, ..., m$ definimos $\mathrm{e}_{r^{\mathrm{i}}} = \mathrm{e}(0)_{\mathrm{i}}^{q(0)}$ para $\mathrm{i} = 1, ..., m$ (ver [7]). Por inducción se concluye que $\mathrm{e}_a \in \mathrm{E}^*$ para todo $a \in V$. Si $a \notin \mathrm{E}(K)$, entonces $\mathrm{e}_a = \mathrm{e}(k)$ para cierto $k = 0, ..., K-1$ y se define $q_a = q(k)$, es decir, a es nodo interior en $V \setminus \mathrm{H}$. Los sucesores de a son a^{i} con $\mathrm{i} = 1, ..., m$ y definimos $\mathrm{e}_{a^{\mathrm{i}}} = \mathrm{e}(k)_{\mathrm{i}}^{q(k)}$. Si $\mathrm{e}_a \in \mathrm{E}(K)$, entonces es una hoja, es decir, $a \in \mathrm{H}$ y colocamos $q_a = q^*$. Entonces, $q_a \in Q$ si y solo si a es un vértice interior (ver [7]).

2.3.1. Podamiento

Cuando en la construcción de un árbol se utilizan una gran cantidad de cortes sobre sus nodos, al punto de que cada hoja o nodo terminal consta exactamente de una observación de $\mathfrak{X}$, el estimador del error de resustitución R posiblemente está sesgado. La razón de esto es que, si cada hoja presenta una única observación, el árbol procede a clasificarlo en su respectiva clase de modo que $R(T) = 0$, concluyendo apresuradamente que a mayor cantidad de cortes el clasificador funciona mejor. Sin embargo, el resultado es todo lo contrario: la tasa de clasificación errónea tiende a aumentar a medida que la complejidad (cantidad de nodos terminales) aumenta, sobrea justando el modelo (ver [1]). Por otro lado, el utilizar árboles con muy pocos cortes evita emplear toda la información disponible en el conjunto de entrenamiento $\mathfrak{L}$ resultando en tasas de clasificación erróneas muy altas.

El enfoque que mejor enfrenta la situación antes descrita se sustenta en dos ideas principales:

- Para árboles suficientemente grandes se realiza una poda de aquellas ramas que no aportan información trascendente, en dirección ascendente desde las hojas a la raíz.
- Dado que el estimador R tiende a producir resultados muy optimistas (recordar que se calcula del conjunto $\mathfrak{L}$)- cuando en realidad se tiene lo contrario - es necesario utilizar estimadores más precisos para $R^*(T)$.

Para un árbol T cualquiera, donde el conjunto $\widetilde{T} \doteq H$ denota sus hojas, el estimador de resus-

titución $R(T)$ para la tasa de clasificación errónea $R^*(T)$ se define por $R(T) = \sum_{t\in\widetilde{T}} R(t) = \sum_{t\in\widetilde{T}} r(t)p(t)$, donde cada término $r(t)p(t)$ corresponde a una ponderación entre el error de clasificación en el nodo terminal t y la proporción de observaciones totales presentes en él (ver [1]). Inicialmente se hace crecer el árbol hasta que tenga un tamaño lo suficientemente grande pero no necesariamente de tamaño máximo, es decir, evitando tener una observación por hoja puesto que conlleva mayor tiempo de cómputo. Un criterio razonable para realizar tal crecimiento consiste en especificar un número de mínimo $N_{\min}$ de observaciones para las hojas tal que si para un nodo t, $N(t) \leq N_{\min}$ entonces no se continúa particionando dicho nodo pasando a constituir una hoja. El árbol construido bajo dicho procedimiento se denota por $T_{\max}$ y como se mencionó antes está sobre ajustado en el conjunto de entrenamiento $\mathfrak{L}$.

Para el proceso de podamiento se precisa de las siguientes definiciones:

Definición 2.6 *Una rama T_t de T con raíz en $t \in T$ se define como el conjunto $T_t = \{t' \in T : \exists$ un camino de aristas que conecta t' con $t\}$.*

Definición 2.7 *Podar una rama T_t de ún árbol T consiste en eliminar todo $t' \in T_t \setminus \{t\}$. El árbol podado de esta forma se denota por $T - T_t$.*

Si T' se obtiene desde T mediante la poda sucesiva de ramas, entonces T' se denomina subárbol podado desde T y se denota por $T' \prec T$. Claramente el proceso de podamiento en un árbol se puede realizar de múltiples maneras aún más si este posee una gran cantidad de cortes. El primer paso consiste en podar el árbol $T_{\max}$ en dirección ascendente tal que en cada paso el estimador de resustitución general $R(T)$ sea lo más pequeño posible. Al realizar este procedimiento se genera una secuencia de árboles cada vez más pequeños $T_{\max}, T_1, T_2, ..., r$ hasta alcanzar la raíz. Suponiendo que la cantidad de nodos terminales de $T_{\max}$ está dado por L, entonces para cada $M = 1, ..., L$ se considera la clase $\mathcal{T}_M$ de todos los subárboles de $T_{\max}$ con $L - M$ nodos terminales. Para cada $M = 1, ..., L$ se escoge el árbol T_M como aquel en la clase $\mathcal{T}_M$ que minimiza el estimador $R(T)$, es decir, $R(T_M) = \min_{T\in\mathcal{T}_M} R(T)$. El problema de este primer enfoque es que la secuencia de de subárboles $\{T_M\}_{M=1}^{L}$ no está necesariamente anidada por lo que T_{M+1} no resulta ser un subárbol de T_M (ver [1]). Este se traduce en que la poda no se realiza progresivamente en forma ascendente hacia la raíz, por tanto, es posible que nodos que fueron podados en cierta etapa vuelvan a aparecer en una etapa posterior.

La solución a este problema consiste en definir un nuevo estimador para la tasa de clasificación errónea R^* (ver [1]).

Definición 2.8 *Para cualquier subárbol $T \prec T_{max}$, se define su complejidad como la cantidad de hojas que posee, es decir, $|\widetilde{T}|$. Sea $\alpha \geq 0$ un número real llamado el parámetro de complejidad. Se define la medida de complejidad del árbol T como $R_\alpha(T) = R(T) + \alpha|\widetilde{T}|$.*

La medida de complejidad es una combinación lineal entre el estimador de resustitución del árbol y su complejidad, y se interpreta como una penalización al costo según el tamaño del árbol. Al fijar $\alpha \in \mathbb{R}_+$, se busca el árbol $T(\alpha)$ que minimice la medida de complejidad $R_\alpha(T)$, es decir, $R(T(\alpha)) = \min_{T\prec T_{\max}} R_\alpha(T)$. Si α es pequeño la penalización por tener un gran número de hojas es baja y por lo tanto $T(\alpha)$ tiende a ser un árbol de gran

tamaño, mientras que si α tiende a valores más pequeños, la penalización por complejidad es alta y así el árbol construido es de menor tamaño. Los siguientes dos resultados son de carácter fundamental pues establecen la existencia y unicidad de los subárboles óptimos a nivel α.

Proposición 2.9 *El menor árbol que minimiza R_α para cierto parámetro de complejidad α se define por las condiciones: i) $R_\alpha(T(\alpha)) = \min_{T \prec T_{max}} R_\alpha(T)$ y ii) si $R_\alpha(T) = R_\alpha(T(\alpha))$ entonces $T(\alpha) \prec T$.*

Proposición 2.10 *Para cada $\alpha \in \mathbb{R}_+$ existe el subárbol de menor tamaño que minimiza R_α.*

Se define $T_1 := T(0)$ como el subárbol de $T_{\max}$ de menor tamaño tal que $R(T_1) = R(T_{\max})$. Sea $(\mathrm{E}(k) : \ k = 1, ..., K)$ la partición que define a $T_{\max}$ notando que K se relaciona directamente con su complejidad. Para $t \in \mathrm{E}(K-1)$ suponemos que existe una pregunta $q \in Q$ que lo particiona en la familia de hojas $\{t^{\mathrm{i}} : \ \mathrm{i} = 1, ..., m\} \subseteq \mathrm{E}(K)$. Dado que $R(T)$ disminuye al ir aumentando la complejidad, se tiene que $R(\{t\}) \geq \ \sum_{\mathrm{i}}^{m} R(\{t^{\mathrm{i}}\})$. Si se tiene la igualdad entonces particionar t no representa ninguna mejora en $R(T)$ por lo podando sus sucesores se obtiene un subárbol de menor complejidad. Este proceso se continúa realizando hasta no poder continuar con la poda y el subárbol resultante corresponde a T_1. Para cualquier rama T_t de T_1 se define $R(T_t) = \sum_{t' \in \widetilde{T}_t} R(\{t'\}) = \sum_{t' \in \widetilde{T}_t} r(t')$ como el costo sobre todos los nodos terminales de T_t. Un hecho a destacar es que para todo nodo $t \in T_1 \setminus \widetilde{T_1}$, $R(\{t\}) > R(T_t)$. Se fija $R_\alpha(\{t\}) = R(\{t\}) + \alpha$ y $R_\alpha(T_t) = R(T_t) + \alpha|\widetilde{T}_t|$. Para continuar con el proceso de poda ahora sobre T_1 sería conveniente tener $R_\alpha(\{t\}) = R_\alpha(T_t)$ ya que implica preferir t sobre T_t dado su menor nivel de complejidad. Resolviendo la ecuación para α se tiene lo siguiente:

- Si $\alpha < \frac{R(\{t\}) - R(T_t)}{|\widetilde{T}_t| - 1}$ entonces $R_\alpha(T_t) < R_\alpha(\{t\})$ por lo que no conviene podar la rama T_t
- Si $\alpha = \frac{R(\{t\}) - R(T_t)}{|\widetilde{T}_t| - 1}$ entonces es preferible cortar la rama T_t para disminuir la complejidad de T_1.

Pero, ¿qué rama se debe podar cuando es posible que exista más de un nodo para el que se alcance la igualdad entre $R_\alpha(\{t\})$ y $R_\alpha(T_t)$?. La respuesta es elegir aquel nodo que presenta el vínculo más débil con el árbol T_1, es decir, aquel donde primero se alcanza el valor crítico cuando se incrementa el valor del parámetro de complejidad α. Más precisamente, si se define la función:

$$g_1(t) = \begin{cases} \frac{R(\{t\}) - R(T_t)}{|\widetilde{T}_t| - 1} & , \ t \notin \widetilde{T}_t \\ +\infty & , \ t \in \widetilde{T}_t \end{cases}$$

El vínculo más débil $\bar{t}_1$ en T_1 es aquel nodo tal que $g_1(\bar{t}_1) = \min_{t \in T_1} g_1(t)$. De la definición de g_1 se concluye que el mínimo nunca se alcanza sobre un nodo terminal debido a que sobre ellos se toma valor $+\infty$, con lo que efectivamente el vínculo más débil correspon-

de a nodos interiores de T_1. Luego, se observa que cuando α crece hasta alcanzar el valor $\alpha_2 := g_1(\bar{t}_1)$ se tiene que $R_{\alpha_2}(\{\bar{t}_1\}) = R_{\alpha_2}(T_{\bar{t}_1})$ y por tanto se poda $T_{\bar{t}_1}$ de T_1. Al subárbol resultante de esta elección se le denota T_2 y corresponde a $T_2 = T_1 - T_{\bar{t}_1}$. En caso de existir más de un vínculo más débil se procede a podar todas las ramas asociadas a ellos. Al repetir sucesivamente este procedimiento se construye una sucesión decreciente de subárboles anidados $T_1 \succ T_2 \succ ... \succ \{r\}$.

2.4. Modelo para una molécula de ADN

En la presente sección se modelerá matemáticamente la estructura y relación de los genes en una molécula de ADN, la que se compone de dos hebras. Luego se presenta un algoritmo de optimización desarrollado por Hart, Martínez y Videla (ver[6]), el cual maximiza la tasa de ocupación de las zonas génicas en ambas hebras de acuerdo con las observaciones experimentales que se han realizado en múltiples organismos procariotas como las bacterias. La distribución de los genes obtenida de acuerdo con este criterio permite tener un conjunto de potenciales candidatos a genes que constituirán la base para el problema de identificación de genes a través de la etapa de clasificación por medio de árboles de decisión.

En una molécula de ADN bacteriano, las hebras que la constituyen presentan zonas que codifican posibles genes y zonas que no codifican genes. A estas segundas se les suele llamar en la literatura zonas intergénicas o gaps pues están entre candidatos a genes. Como ya se ha mencionado en capítulos anteriores, el proceso de modificación post-transcripcional splicing no ocurre en organismos como las bacterias por lo que los intrones o gaps no son eliminados y así se mantienen fijos a lo largo de los procesos de replicación del ADN dentro de una célula. En el proceso de transcripción de una molécula de ADN, cada hebra puede ser leída en cualquiera de los posibles tres marcos de lectura.

Se asocia la hebra primaria y complementaria por 1 y -1 respectivamente. Los tres marcos de lectura relativos a la hebra primaria se denotan por 1, 2 y 3, mientras que para la complementaria se hace con -1, -2 y -3. Ambas hebras se modelan como semirrectas continuas, es decir, como $\mathbb{R}_+$ y las seis maneras de leer el ADN secuencialmente son enumerados por el índice i donde $\mathrm{i} \in \mathcal{R} = \{-3, -2, -1, 1, 2, 3\}$. La unidad básica de información en una hebra se denomina codón, el cual está formado por tres nucleótidos. Al ir moviéndose por una hebra en cierta dirección la lectura se realiza de codón en codón como si un cabezal apuntara a este para luego moverse tres posiciones hacia el siguiente. Por lo tanto, las zonas génicas como los gaps son grupos de codones. Una vez leída una hebra de acuerdo con uno de sus marcos de lectura esta se particiona en posibles zonas génicas y en gaps. En $\mathbb{R}_+$ cada posible gen y gap se modela como un intervalo por lo que, en un marco de lectura particular, una hebra se concibe como una unión disjunta de intervalos.

Para cada marco de lectura $\mathrm{i} \in \mathcal{R}$ se define el proceso estocástico $\mathrm{E}^{\mathrm{i}} = (\mathrm{E}^{\mathrm{i}}_t : \ t \in \mathbb{R}_+)$ a valores en $\{0, 1\}$ el cual produce una partición sobre la hebra respectiva (ver[6]). Aquellas zonas que toman valor 1 se asocian a posibles genes y aquellas con valor 0 se asocian a los gaps. Al considerar cada candidato a gen como un intervalo, se define $\mathcal{I}^{\mathrm{i}}$ como la familia de intervalos que representan a un gen, o de otra manera $\{t \in \mathbb{R}_+ : \ E^{\mathrm{i}}_t = 1\} = \cup_{I \in \mathcal{I}^{\mathrm{i}}} I$. El conjunto de todos los candidatos a genes en ambas hebras para todos los marcos de lectura se denota por $\mathcal{I} = \cup_{\mathrm{i} \in \mathcal{R}} \mathcal{I}^{\mathrm{i}}$. Para $I \in \mathcal{I}$ sea $s(I)$ y $f(I)$ su coordenada de inicio y fin respectivamente, entonces $I = [s(I), f(I))$ y además se define por $\mathrm{i}(I)$ al marco de lectura al cual pertenece: $\mathrm{i}(I) = \mathrm{i}$ si $I \in \mathcal{I}^{\mathrm{i}}$. Si bien se ha observado empíricamente que dos genes en diferentes hebras cada uno pueden solaparse en cierta zona, estos representan una pequeña fracción del total por lo que el presente modelo no considera este fenómeno: se busca aquella distribución de candidatos que maximice la tasa de ocupación de zonas codificantes sobre el largo total del ADN y en caso de existir un solapamiento siempre se debe escoger aquel

candidato que haga cumplir dicho principio (ver[6]).

Sea K una componente conexa del conjunto $\{t \geq 0 : \sum_{\mathrm{i}\in\mathcal{R}} E_t^{\mathrm{i}} \geq 1\}$. K está cubierto por candidatos a genes que posiblemente se solapan entre ellos y además no contiene gaps, pues si los tuviera no sería componente conexa. Es decir, cada componente conexa K constituye un intervalo en $\mathbb{R}_+$ con inicio y fin dados por $s(K)$ y $f(K)$ respectivamente. A cada componente K se le llama isla y al conjunto de estas se le denota por $\mathcal{K}$. De la condición $\mathcal{I}^* \subseteq \mathcal{I}$ se desprende que si $I \in \mathcal{I}^*$ entonces representa una zona con valor 1 para cierto marco de lectura y por tanto está contenido en alguna isla. Se fija cierta isla K y se considera el conjunto $\mathcal{I}^{\mathrm{i}}(K) = \{I \in \mathcal{I}^{\mathrm{i}} : \ I \subseteq K\}$ de todos los candidatos a genes del marco de lectura $\mathrm{i} \in \mathcal{R}$ que están en la isla K. De la misma forma se define $\mathcal{I}(K) = \cup_{\mathrm{i}\in\mathcal{R}}\mathcal{I}^{\mathrm{i}}(K)$ como todos los candidatos a genes contenidos en K sin importar el marco de lectura utilizado. Dado que se pretende maximizar la tasa de ocupación de los candidatos sobre el largo total es necesaria la noción de largo de un gen I que naturalmente corresponde a la cantidad $|I| = f(I) - s(I)$. De igual manera, para una isla K se define el largo de una familia de candidatos $\widetilde{\mathcal{I}} \subseteq \mathcal{I}(K)$ por $\Sigma(\widetilde{\mathcal{I}}) = \sum_{I\in\widetilde{\mathcal{I}}} |I|$.

Con las nociones introducidas anteriormente el principio de maximización sobre la tasa de ocupación de los candidatos a genes con respecto al largo de la molécula de ADN, dice que el conjunto final de genes escogidos, $\mathcal{I}^* = \cup_{K\in\mathcal{K}}\mathcal{I}^*(K)$, es tal que para cada isla $K \in \mathcal{K}$, la clase $\mathcal{I}^*(K)$ resuelve el siguiente problema de optimización (ver[6]):

$$\Sigma(\mathcal{I}^*(K)) := \text{máx}\left\{\Sigma(\widetilde{\mathcal{I}}) : \ \widetilde{\mathcal{I}} \subseteq \mathcal{I}(K) \text{ es una familia disjunta}\right\} \tag{2.20}$$

es decir, sobre cada isla se busca aquella familia de candidatos que no solapen entre sí y tal que además maximicen su largo.

Para resolver este problema de maximización se fijará una isla $K \in \mathcal{K}$ arbitraria sobre la cual se trabaja. Sea $\mathcal{J}^{\mathrm{i}} := \mathcal{I}^{\mathrm{i}}(K)$ para $\mathrm{i} \in \mathcal{R}$ y $\mathcal{J} = \cup_{\mathrm{i}\in\mathcal{R}}\mathcal{J}^{\mathrm{i}}$ ambas familias de intervalos en K. Se define la vecindad de un candidato a gen $I \in \mathcal{J}$ como $\mathcal{N}(I) := \{J \in \mathcal{J} : \ I \cap J \neq \emptyset\}$. Se supone además que $n := |\mathcal{J}|$ es el número total de candidatos. Se procede a establecer una relación de orden en $\mathcal{J}$ dada por (ver[6]):

$$\begin{aligned} I \preceq_f J \ \text{ ssi } \quad & f(I) < f(J), \ \text{ ó} \\ & f(I) = f(J) \ \text{ y } \ s(I) > f(J), \ \text{ ó} \\ & f(I) = f(J) \ \text{ y } \ s(I) = s(J) \ \text{ y } \ \mathrm{i}(I) \leq \mathrm{i}(J) \end{aligned}$$

Proposición 2.11 *La relación $\preceq_f$ sobre $\mathcal{J}$ es una relación de orden total.*

De esta manera se asume que $\mathcal{J} = \{I_1, I_2, ..., I_n\}$ donde $I_{\mathrm{i}} \preceq_f J$, $\forall\ 1 \leq \mathrm{i} \leq j \leq n$. Esta relación ordena los candidatos a genes primero de acuerdo su extremo final, luego de acuerdo con su extremo inicial y finalmente según el marco de lectura en que se encuentran. Se define ahora $t_j := s(I_j)$ y $T_j := f(I_j)$, $\forall\ j = 1, ..., n$ donde $I_j = [t_j, T_j]$. Se debe notar que:

$$I_{\mathrm{i}} \cap I_k \neq \emptyset \implies I_j \cap I_k \neq \emptyset \quad \forall\ 1 \leq \mathrm{i} \leq j \leq k \leq n$$

En efecto, si $I_j \cap I_k = \emptyset$ entonces $T_{\text{i}} \leq T_j < t_k$ y así $I_{\text{i}} \cap I_k = \emptyset$ lo que es una contradicción (ver[6]). Este resultado quiere decir que si dos candidatos en $\mathcal{J}$ solapan entre si, entonces todos aquellos candidatos situados entre ambos según la relación $\preceq_f$ también solapan con aquel que es mayor.

Se definen las siguientes familias en $\mathcal{J}$: $\bar{J}_0 := \emptyset$ y $\bar{J}_m := \{I_1, I_2, ..., I_m\}$ para $m = 1, ..., n$. A continuación, se presenta un lema fundamental que será posteriormente usado en el algoritmo para resolver el problema de optimización (ver[6]):

Lema 2.12 *Sea $m \geq 1$. Entonces existe l_m con $0 \leq l_m \leq m$, tal que $\bar{J}_m \cap \mathcal{N}(I_m) = \bar{J}_m \setminus \bar{J}_{l_m}$.*

Demostración. Sea $l = \text{mín}\left\{l' : \ I_{l'} \in \bar{J}_m \cap \mathcal{N}(I_m)\right\}$ entonces I_l es el menor vecino anterior a I_m. De la definición de vecino se tiene que $I_l \cap I_m \neq \emptyset$ y de la definición de l se tiene aquellos candidatos menores a I_l no son vecinos de I_m, es decir, $\bar{J}_{l-1} \cap \mathcal{N}(I_m) = \emptyset$. Resulta natural entonces colocar $l_m = l-1$ y por la observación anterior se concluye que para $j = l, l+1, ..., m$, $I_j \cap I_m \neq \emptyset$. □

El lema se interpreta de la siguiente forma: todos los vecinos anteriores (menores según $\preceq_f$) al candidato a gen I_m se encuentran a partir de cierta posición l_m, es decir son $I_{l_m+1}, I_{l_m+2}, ..., I_m$. Esto se concluye directamente de la observación anterior, pues si el primer candidato al cual intersecta I_m es I_{l_m+1} entonces también intersecta a todos aquellos cuyos índices estén entre l_{m+1} y m.

Para cada $m = 1, ..., n$ se plantea el siguiente problema de optimización sobre la familia $\bar{J}_m$, cuyo valor está dado por la cantidad w_m:

$$w_m = \text{máx}\left\{\Sigma(\widetilde{\mathcal{I}}) : \ \widetilde{\mathcal{I}} \subseteq \bar{J}_m \text{ es una familia disjunta}\right\} \tag{2.21}$$

El siguiente algoritmo resuelve el problema anterior (ver[6]):

Algoritmo:

1. Sea $\hat{w}_0 = 0$ y $\hat{w}_1 = |I_1|$. Sea además $\hat{J}_0 = \emptyset$ y $\hat{J}_1 = \{I_1\}$.
2. Para cada $m = 2, 3, ..., n$, sea l_m el índice tal que $\mathcal{N}(I_m) \cap \bar{J}_m = \bar{J}_m \setminus \bar{J}_{l_m}$ dado por el lema.
3. Recursivamente se calculan las cantidades $\hat{w}_m = \text{máx}\left\{\hat{w}_{m-1}, \hat{w}_{l_m} + |I_m|\right\}$.
4. Recursivamente se calcula la solución óptima correspondiente a $\hat{w}_m$ de acuerdo con el siguiente esquema de decisión:
   (a) Si $\hat{w}_{m-1} > \hat{w}_{l_m} + |I_m|$, entonces se coloca $\hat{J}_m = \hat{J}_{m-1}$.
   (b) Si $\hat{w}_{m-1} < \hat{w}_{l_m} + |I_m|$, entonces se coloca $\hat{J}_m = \hat{J}_{l_m} \cup \{I_m\}$.

(c) Si $\hat{w}_{m-1} = \hat{w}_{l_m} + |I_m|$, entonces se escoge arbitrariamente $\hat{J}_m$ como $\hat{J}_{m-1}$ o $\hat{J}_{l_m} \cup \{I_m\}$.

5. La elección final de genes es $\hat{J} := \hat{J}_n$ y el área ocupada por esta elección es $\hat{w} := \hat{w}_n$.

Proposición 2.13 *Para todo $m \geq 1$, $\hat{J}_m$ es una solución óptima sobre el subconjunto $\bar{J}_m$ y $\hat{w}_m$ es el área ocupada por dicha solución. Es decir, $\hat{w}_m = \Sigma(\hat{J}_m) = \Sigma^{*M}(\bar{J}_m) = w_m$. Así, el algoritmo produce una solución general sobre $\mathcal{J}$ dada por $\hat{J} = \hat{J}_n$ y el espacio ocupado en la molécula de ADN está dado por $\hat{w} = \hat{w}_n$.*

Demostración. Se prueba la correctitud del algoritmo por inducción sobre $m = 1, ..., n$, (ver[6]):

- **Caso Base:** $\hat{w}_0 = \Sigma(\hat{J}_0) = \Sigma(\emptyset) = 0 = \Sigma^{*M}(\bar{J}_0) = w_0$ y $\hat{w}_1 = \Sigma(\hat{J}_1) = \Sigma(\{I_1\}) = |I_1| = \Sigma^{*M}(\bar{J}_1) = w_1$

- **Paso Inductivo:** Suponer que para $m < n - 1$ se tiene que $\hat{w}_{m-1} = \Sigma(\hat{J}_{m-1}) = \Sigma^{*M}(\bar{J}_{m-1}) = w_{m-1}$. Se calcula $w_m = \text{máx}\{\hat{w}_{m-1}, \hat{w}_{l_m} + |I_m|\}$:

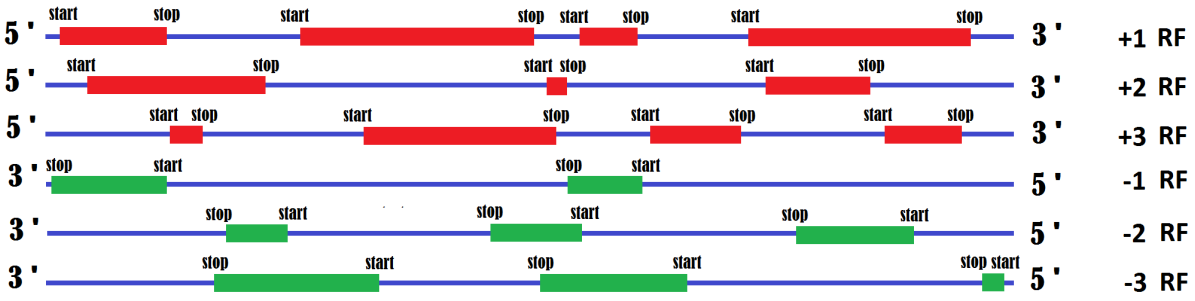
  - Si $\hat{w}_{m-1} > \hat{w}_{l_m} + |I_m|$ entonces al agregar I_m a la solución óptima $\hat{J}_{l_m}$ no es posible mejorar la solución dada en el paso anterior, por lo que se resuelve el mismo problema, es decir, $\hat{J}_m = \hat{J}_{m-1}$ cuya ocupación es $\hat{w}_m = \hat{w}_{m-1}$.
  - Si $\hat{w}_{m-1} < \hat{w}_{l_m} + |I_m|$ basta notar que la familia $\bar{J}_{l_m}$ no intersecta a I_m por lo que al agregar a la solución óptima $\hat{J}_{l_m}$ el candidato I_m, se obtiene una familia disjunta que mejora la solución con respecto a $\hat{w}_{m-1}$. Se fija así $\hat{J}_m = \hat{J}_{l_{m-1}} \cup \{I_m\}$ cuya ocupación es $\hat{w}_m = \hat{l_m} + |I_m|$.

  La solución para el problema general sobre $\mathcal{I}(K)$ se obtiene notando que $\bar{J}_n = \mathcal{J} = \mathcal{I}(K)$, por lo que para $m = n$ la familia óptima es $\hat{J}_n$ con una ocupación en el ADN de $\hat{w} = \hat{w}_n$.

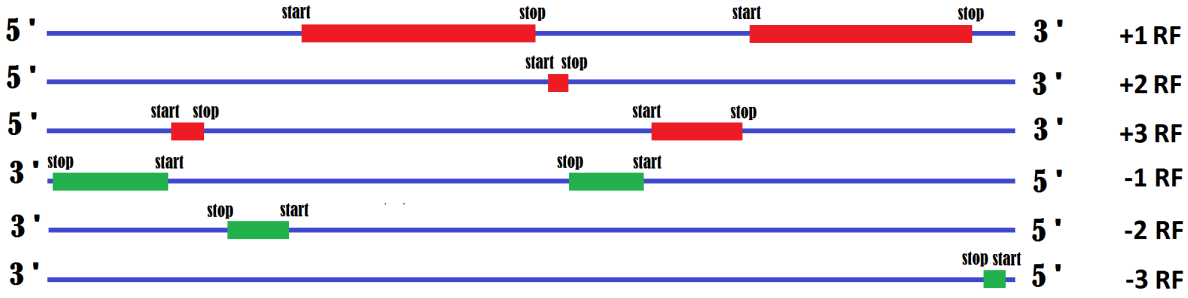
□

El algoritmo anterior recursivamente va mejorando la solución siempre y cuando sea posible combinar el candidato I_m junto a una solución óptima previa $\hat{J}_{l_m}$, de tal forma que esta nueva familia siga siendo disjunta. En caso contrario se mantiene la solución óptima resultante del paso anterior.

La Figura 2.1 ejemplifica el uso del algoritmo de optimización sobre una distribución inicial de candidatos a genes sobre los seis marcos de lectura. Se observa que tras su uso, se produce una nueva distribución tal que ningún candidato seleccionado se solapa con algún otro.



(a) Antes de maximización



(b) Después de maximización

Figura 2.1: Ejemplo del uso del algoritmo de optimización. En (a) la configuración de los candidatos en los marcos de lectura previa a su uso. En (b) la configuración posterior a su uso.

Una pregunta interesante por realizar acerca de este algoritmo de optimización es si es factible o no relajar la condición sobre que los candidatos de un marco de lectura en particular no deban traslaparse y que la única posibilidad de presentarse dicha situación sea sobre candidatos en marcos diferentes. Para tales efectos consideremos el problema de optimización restringido a una isla $K = [s(K), f(K))$ en particular, y supongamos sin pérdida de generalidad que ella está cubierta por los candidatos $I^{\mathrm{i}} = [s^{\mathrm{i}}, f^{\mathrm{i}})$ con $\mathrm{i} \in \mathcal{R}$, es decir, existe un candidato por cada posible marco de lectura. Supongamos también que algunos de los candidatos en marcos de lectura diferentes traslapan entre sí y, que además existe otro candidato $I^* = [s^*, f^*)$ con $I^* \subseteq K$ tal que este pertenece al marco de lectura $\hat{\mathrm{i}} \in \mathcal{R}$ con $I^* \cap I^{\hat{\mathrm{i}}} \neq \emptyset$. Lo último dice que I^* e $I^{\hat{\mathrm{i}}}$ son candidatos traslapando en el mismo marco de lectura. Entonces se tiene la siguiente proposición.

Proposición 2.14 *El problema de optimización definido por* $\Sigma(\mathcal{I}^*(K))$ *tiene solución óptima.*

Demostración. El algoritmo de optimización antes presentado es generalizable a un número arbitrario de hebras, pues se basa únicamente en la relación de orden $\preceq_f$ antes definida.

Esto permite mirar cada marco de lectura $\mathrm{i} \in \mathcal{R}$ como una hebra diferente del resto, y lo mismo ocurre con el candidato I^*. Basta entonces considerar a I^* como parte de una hebra independiente a la representada por su marco de lectura original $\hat{\mathrm{i}}$, digamos $\bar{\mathrm{i}} \notin \mathcal{R}$.Entonces, como al algoritmo de optimización no le importa la manera en que fueron construidos los candidatos por cada marco de lectura sino, por el contrario, se fija solamente en la relación de orden $\preceq_f$ entre los extremos que los definen, se puede considerar $\bar{\mathrm{i}} \neq \hat{\mathrm{i}}$. Tomando entonces $\bar{\mathcal{R}} = \mathcal{R} \cup \left\{\bar{\mathrm{i}}\right\}$ se resuelve el problema de optimización $\Sigma(\mathcal{I}^*(K))$ de la manera usual sobre los marcos de lectura $\bar{\mathcal{R}}$. □

Capítulo 3

Presentación y Análisis de Datos

Los datos utilizados en este trabajo se obtuvieron del *National Center for Biotechnology Information* (*NCBI*), una rama del *National Institutes of Health* (*NIH*) de los Estados Unidos. Este centro de investigación posee bases de datos muy relevantes para la biotecnología, biomedicina y bioinformática. Específicamente, su sistema *Genbank* proporciona bases de acceso abierto con las secuencias de nucleótidos del genoma y su respectiva traducción proteica para más de cien mil organismos diferentes y hasta febrero del 2013 contenía más de 150 mil millones de bases de nucleótidos. GenBank funciona gracias a la colaboración de muchos laboratorios a lo largo del mundo que aportan resultados originales sobre secuenciación. Genbank se encarga de verificar la calidez de los datos entregados y a cada secuencia le asigna un número de acceso con el que después se consulta libremente.

En este capítulo se introducen los datos a utilizar durante la etapa de simulaciones y se realizan análisis a partir de ellos. En la primera sección se muestran las distintas bases de datos que contienen la información trascendental a considerar en este trabajo. Se describen sus características y los vínculos con las demás bases. La segunda sección se dedica a analizar los datos presentados bajo criterios de relevancia en el área de la genómica. Además, se evalúa el rendimiento del algoritmo de optimización presentado en el capítulo anterior sobre distintas distribuciones de candidatos a genes.

3.1. Presentación de Datos

Los datos que se usarán en este trabajo corresponden a aquellos que describen el ADN genómico de la bacteria *Eschericha Coli* de cepa *K-12* y subcepa *MG1655*. Este genoma de estructura circular está secuenciado completamente con un total de 4641652 nucleótidos y, para acceder a él a través de *GenBank* se debe usar el acceso NC-000913 en su tercera versión con última fecha de actualización el 15 de mayo del 2014. La *Escherichia Coli* es un bacilo de la familia de las enterobacterias que forma parte de la flora microbiana del tracto gastrointestinal de humanos y otros animales. Son necesarias para el correcto funcionamiento

de la digestión de alimentos ayudando en la absorción de nutrientes, además de participar en otras funciones como la síntesis de vitaminas B y K. A pesar de realizar un gran aporte en el organismo que la hospeda bajo condiciones de equilibrio existen cepas patógenas originadas desde procesos de mutación genética que pueden incluso asociarse a la aparición de ciertas enfermedades de cuidado como es el caso de la enfermedad de Crhon y otros tipos de infecciones intestinales. Se estima que al ser una de las especies de bacterias más diversas dentro del reino, solo el 20 % de los genes presentes en una *E. Coli* común son compartidos entre todas las variedades de cepas.

El genoma de la *E. Coli* fue uno de los primeros en ser secuenciado completamente por lo que se ha usado de manera transversal en muchos campos de la biología y la industria para estudiar modelos de síntesis en proteínas, el desarrollo de vacunas y el uso en procesos de fermentación industrial . Particularmente el uso de la cepa K-12 es muy común en los laboratorios dada su buena adaptación a dichos ambientes lo que facilita su manipulación.

Las bases de datos de *GenBank* contienen una gran cantidad de información sobre los genomas de los individuos a estudiar. Cualquiera de ellas sigue una estructura predeterminada de acuerdo a una pauta que establece como almacenar la información dependiendo de las características de la secuenciación y si esta proviene de un organismo procariota o eucariota. Los datos que se almacenan en una típica base de GenBank son por ejemplo las características generales de cada gen, las zonas codificantes dentro de un gen para el caso de eucariotas, las proteínas que genera cada gen con su respectiva función y las características de las secuencias de ARNm y ARNt utilizados en el proceso de transcripción y traducción solo por nombrar algunos.

La enorme cantidad de información contenida en una típica base de datos de GenBank necesita ser procesada para extraer la información considerada relevante en el siguiente trabajo. Es importante mencionar que dicha labor fue llevada a cabo previamente por Hart, Martínez y Videla en un trabajo previo sobre métodos de organización de candidatos a genes en una secuencia de ADN bacteriano (ver [6]). Por tanto, las bases efectivas a utilizar son las obtenidas posterior a dicho procesamiento. Dado que interesa estudiar las decisiones tomadas en el proceso de identificación de genes, la información relevante para este propósito se obtendrá inicialmente conociendo las posiciones de inicio y fin de cada gen en la bacteria, sus codones de inicio y término, el marco de lectura por el cual se obtiene cada uno y las respectivas secuencias de bases que representan las hebras primaria y complementaria.

Las bases de datos a utilizar se explican a continuación:

- **Bases primaria y complementaria**

  Estas bases de datos son dos vectores que representan las secuencias de nucleótidos que conforman las hebras primaria y complementaria. Naturalmente los vectores se denominan **primaria** y **complementaria** respectivamente y son obtenidos desde el acceso NC-000913 para la *E. coli* en GenBank. La información relevante de los nucleótidos está presente en las bases nitrogenadas que los conforman por lo que ambos vectores se visualizan como secuencias de letras en el alfabeto $\{A, G, T, C\}$. Dado que

ambas hebras son antiparalelas, los vectores asociados tienen el mismo largo siendo este n, donde $n = 4641652$ para el caso de la *E. Coli*. El sentido en que se leen las hebras en los procesos de transcripción y traducción también incide en cómo se almacena la información: ambas hebras se leen en sentido $5^{'}$ a $3^{'}$, sin embargo al ser antiparalelas la lectura se realiza en sentido contrario por lo que el extremo inicial en la lectura de una corresponde al extremo final en la lectura de la otra. Los vectores primaria y complementaria se visualizan en el sentido $5^{'}$ a $3^{'}$ de modo que si la base en la j-ésima coordenada de primaria es una A, su base complementaria T aparece en la $(n-j+1)$-ésima posición del vector complementaria y viciversa. La misma situación ocurre para los pares de bases C y G. La Tabla 3.1 muestra las primeras 15 posiciones para los vectores primaria y complementaria respectivamente:

Base/Posición	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	...
Primaria	a	g	c	t	t	t	t	c	a	t	t	c	t	g	a	...
Complementaria	g	a	a	a	a	a	t	a	c	t	t	a	c	t	a	...

Tabla 3.1: Visualización de las bases primaria y complementaria.

- **Base de Características**

  Esta base contiene información relevante sobre los genes anotados de la *E. Coli* presentes en la base respectiva de GenBank para la fecha de actualización señalada anteriormente. La información se estructura como una matriz de dimensión 4260×6 en que cada representa un gen en particular y donde las columnas son los atributos o características que se utilizan para describir los genes. Dichos atributos son descritos a continuación:

  - **inicio:** representa la posición en los respectivos vectores "primaria" o "complementaria" de la primera base nitrogenada perteneciente al codón de inicio de un gen. Toma valores en $[n] = \{1, ..., n\}$ donde n es la cantidad de bases en cualquiera de ambas hebras.
  - **término:** representa la posición en los respectivos vectores "primaria" o "complementaria" de la última base nitrogenada perteneciente al codón de término de un gen. Al igual que para ïnicio", este toma valores en $[n]$.
  - **codón de inicio:** representa la tripleta de bases nitrogenadas codificadas como texto que componen el codón de inicio de un gen. Para la *E. Coli* existen 41 posibles codones de inicio distintos para sus genes.
  - **codón de término:** representa la tripleta de bases nitrogenadas codificadas como texto que componen el codón de término de un gen. Para la *E. Coli* existen 32 posibles codones de término distintos para sus genes.
  - **hebra:** representa la hebra de la molécula de ADN a la cual pertenece un gen. Toma valores en el conjunto $\bar{\mathcal{H}} = \{1, 2\}$ por lo que la notación se cambia respecto a la usada en el marco teórico en que se emplea $\mathcal{H} = \{-1, 1\}$.
  - **Marco de Lectura:** representa el marco de lectura o reading frame desde el cual se lee un gen. Si bien en el marco teórico estos se representan como elementos

en el conjunto $\mathcal{R} = \{-3, -2, -1, 1, 2, 3\}$, en este apartado se utiliza el conjunto $\bar{\mathcal{R}} = \{1, 2, 3\}$ indistintamente de la hebra contenedora del gen. La razón de esta ambigüedad se fundamenta en que basta conocer la hebra recipiente para diferenciar el marco de lectura quedando entonces cada uno únicamente identificado por el par ordenado $(s, \mathrm{i}) \in \bar{\mathcal{H}} \times \bar{\mathcal{R}}$.

A continuación se muestran 3 filas de **Características** a modo de ejemplo:

inicio	**término**	**codón de inicio**	**codón de término**	**hebra**	**Marco de Lectura**
190	255	atg	tga	1	1
337	2799	atg	tga	1	1
2801	3733	atg	taa	1	2

Tabla 3.2: Visualización de la base Características.

- **Base de Marcos de Lectura (RF)**

  La base de Marcos de Lectura o por su sigla "RF" (Reading Frame en inglés) contiene todos los posibles candidatos a genes presentes en la doble hebra de ADN para cualquiera de los marcos de lectura definidos por $\mathcal{R}$, por lo que reproduce la información contenida en $\mathcal{I}$. Se estructura como una lista de seis elementos donde cada uno es una matriz que representa los candidatos a genes extraídos de cada uno de los diferentes marcos de lectura, es decir, están asociadas a $\mathcal{I}^{\mathrm{i}}$ con $\mathrm{i} \in \mathcal{R}$. Más precisamente, para $\mathrm{i} \in \mathcal{R}$ sea n_{i} el número de candidatos en $\mathcal{I}^{\mathrm{i}}$, entonces las matrices antes señaladas se denotan por M_{i} tales que $M_{\mathrm{i}} \in \mathbb{R}^{n_{\mathrm{i}} \times 2} \ \ \forall \mathrm{i} \in \mathcal{R}$. Para todo $\mathrm{i} \in \mathcal{R}$, cada fila en M_{i} representa un candidato a gen en dicho marco de lectura y las columnas corresponden a las posiciones de la primera y última base nitrogenada de los codones de inicio y término respectivamente.

  La construcción de cada matriz M_{i} se realiza utilizando únicamente los vectores "primaria" y "complementaria". La cantidad y el tipo de candidatos en M_{i} dependen de los codones de inicio y término seleccionados para extraerlos, por lo que llamando INICIO y TÉRMINO a los subconjuntos en $\{A, T, G, C\}^3$ que representan los codones de inicio y término respectivamente que se usarán en el proceso de extracción, se puede concluir que a diferentes configuraciones de ambos, las matrices M_{i} cambian. La elección de las familias INICIO y TÉRMINO se hace en base a la información contenida en las columnas "codón de inicio" y "codón de término" de la base "Características" del organismo que se pretende estudiar. En general TÉRMINO $= \{TAA, TAG, TGA\}$ y se considera inmutable para cualquier tipo de bacteria mientras que INICIO se elige como aquellos codones con mayor frecuencia de aparición en "codón de inicio". Si bien, para *Escherichia Coli* hay 41 codones de inicio distintos tal que todos están presentes en al menos un gen, el 96,7 % comienzan con ATG y GTG, por lo que en lo que sigue se considera INICIO $= \{ATG, GTG\}$. En la Figura 3.1 se muestran los codones de inicio

más frecuentes para la *E. Coli*:

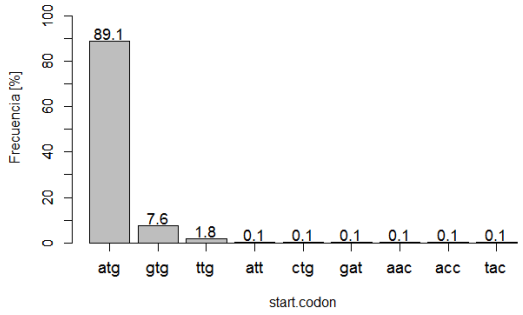


Figura 3.1: Codones de inicio más frecuentes en E. Coli. El codón de inicio ATG ocurre en el 89.1 % de los genes anotados y GTG en el 7.6 %.

Para encontrar los candidatos a genes en una de las matrices M_{i} de la hebra primaria bajo el marco de lectura i-ésimo, se deben consultar las posiciones $\{\mathrm{i}+3k\}_{k\in\mathbb{N}}$ en el vector "primaria" - denotado por P - de la siguiente manera: se busca el menor $k^* \in \mathbb{N}$ tal que $(P_{\mathrm{i}+3k^*}, P_{\mathrm{i}+3k^*+1}, P_{\mathrm{i}+3k^*+2}) \in$ INICIO, momento en que se guarda la posición $t = \mathrm{i}+3k^*$. Luego se continúa con la lectura hasta encontrar el menor $k^{**} \in \mathbb{N}$ tal que $k^{**} > t$ y $(P_{\mathrm{i}+3k^{**}}, P_{\mathrm{i}+3k^{**}+1}, P_{\mathrm{i}+3k^{**}+2}) \in$ TÉRMINO momento en que se registra $T = \mathrm{i}+3k^{**}+2$. Se ha encontrado así el candidato $I = [t, T)$ y se guardan sus posiciones de inicio y término en M_{i}. Se continúa este proceso de búsqueda desde la posición $T+1$ hasta recorrer todo P. Para generar los candidatos a genes en M_{i} de la hebra complementaria se debe tomar la precaución que el marco de lectura de referencia respectivo en la hebra primaria sea respetado. Esto quiere que si se extraen los candidatos usando el marco de lectura i en el vector "complementaria" - denotado por C -, sus posiciones de inicio y término deben mantenerse en el marco de lectura $|\,\mathrm{i}\,|$ de la hebra primaria. De esta manera para $\mathrm{i} \in \{-3,-2,-1\}$ la búsqueda de candidatos comienza en las posiciones $(4+\mathrm{i})$-ésima de la hebra complementaria para que exista la correspondencia entre el marco i de C y e $|\,\mathrm{i}\,|$ de P.

- **Bases Muestra1 y Muestra2**

Guardan información sobre las bases en dirección $5'$ o upstream al codón de inicio de todos los candidatos a genes de las matrices M_{i} con $\mathrm{i} \in \mathcal{R}$. Cada una se estructura como una lista de 2 tablas dependiendo del codón de inicio de los candidatos y de la hebra usada. Así, la lista "Muestra1" hace mención a la hebra primaria y contiene las tablas "Muestra1a" y "Muestra1g", las cuales guardan las regiones upstream a los candidatos comenzando con *ATG* y *GTG* respectivamente. Análogamente "Muestra2" contiene la misma información para la hebra complementaria. Estas bases de datos son fundamentales para identificar y entender la expresión de los genes reales pues codifican las regiones precursoras de los mismos, siendo el medio por el cual el mecanismo celular de traducción genética sabe que un gen está por comenzar una cantidad de bases más

adelante en la dirección de lectura. El número de bases usadas para codificar dichas regiones será fijo para todos los candidatos y puede variar dependiendo el criterio que se desea emplear en el estudio. En este trabajo se utilizarán regiones de 30 y 70 bases, cada una utilizadas bajo condiciones específicas de simulación. Las tablas contienen por cada fila la información relativa a cada candidato a gen, y usan como características: los codones de inicio que están codificados por sus bases en las variables ”d0”, ”d1”, ”d2”; las regiones upstream codificadas como secuencias de X cantidad de bases en las variables ”u1” hasta ”uX”, donde X representa la posición X-ésima previa a ”d0”; una variable de nombre ”inicio” que indica si el candidato corresponde a un gen anotado o no, y toma valores "s"(si) y "n"(no) respectivamente; una columna que indica la posición de ”d0” dentro de la hebra respectiva y corresponde a la posición de inicio del candidato. Se añade que el número de observaciones para ”Muestra1a”, ”Muestra1g”, ”Muestra2a” y ”Muestra2g” son 76281, 56971, 77040 y 57518 respectivamente.

A continuación se muestran 4 filas de la tabla **Muestra1a** usando 30 bases upstream:

	inicio	**u30**	**u29**	**u28**	**u27**	**.....**	**u5**	**u4**	**u3**	**u2**	**u1**	**d0**	**d1**	**d2**
190	si	c	a	g	a	.....	c	a	t	c	c	a	t	g
337	si	t	t	t	t	.....	c	a	a	c	c	a	t	g
8890	no	g	t	a	c	.....	c	t	g	a	c	a	t	g
70261	no	a	c	g	g	.....	a	c	t	a	a	a	t	g

Tabla 3.3: Visualización de la base Muestra1a usando 30 bases upstream. La primera columna indica la posición que ocupa el codón ATG en la hebra primaria.

3.2. Análisis de los Datos

Ya definidas las distintas bases que se utilizarán en este trabajo, se procede a realizar un análisis descriptivo y exploratorio con los datos que ellas poseen en orden para entender y dimensionar el problema. Un primer análisis sobre la base ”características” de genes anotados se muestra en la Tabla 3.4. En ella se resumen varios atributos para los seis marcos de lectura: el número de genes anotados, el largo mínimo, máximo y promedio de estos. Todas las características de largo se miden en función de la cantidad de pares de bases nitrogenadas relativa a cada candidato a gen, y que se abreviará en lo que sigue de esta memoria como *pb*.

Se observa que de 4260 genes anotados para la bacteria *E. Coli*, 2076 están presentes en la hebra primaria y 2184 en la hebra complementaria. El número promedio de genes por marco de lectura en la hebra primaria es de 692 mientras que para la complementaria es de 728 y la misma tendencia se observa con el largo promedio de los genes donde para la hebra primaria es de 940 *pb*. y para la complementaria de 947 *pb*. Lo anterior se sintetiza diciendo que la hebra complementaria contiene en promedio más cantidad de genes y en promedio dicho genes son más largos. Otro aspecto a destacar es la presencia del gen de mayor largo en el marco de lectura +3 con un total de 7076 *pb*.

Hebra	Marco de Lectura	genes anotados	Largo mín	Largo máx	Largo prom
Primaria	+1	673	50	4616	941.5
	+2	716	50	4460	948.7
	+3	687	47	7076	932.8
Complementaria	-1	700	44	5322	956.9
	-2	730	50	4604	947.7
	-3	754	44	4961	938.1

Tabla 3.4: Estadísticos básicos de los genes anotados en los seis marcos de lectura en base "características".

Como se mencionó previamente en la descripción de la base "RF", solo se considerará aquellos candidatos a genes con codón de inicio en el conjunto $\{ATG, GTG\}$, por lo que resulta importante conocer como se diferencian los genes al segmentarlos en aquellos que tienen los codones de inicio en el grupo de estudio y aquellos que no. Un criterio simple para hacer dicha comparación resulta ser el largo de los genes asique se procede a comparar sus distribuciones respecto a la del total de genes. En la Figura 3.2 se visualizan dichas distribuciones de los largos diferenciándolas también por el tipo de hebra con el fin de buscar algún patrón más específico.

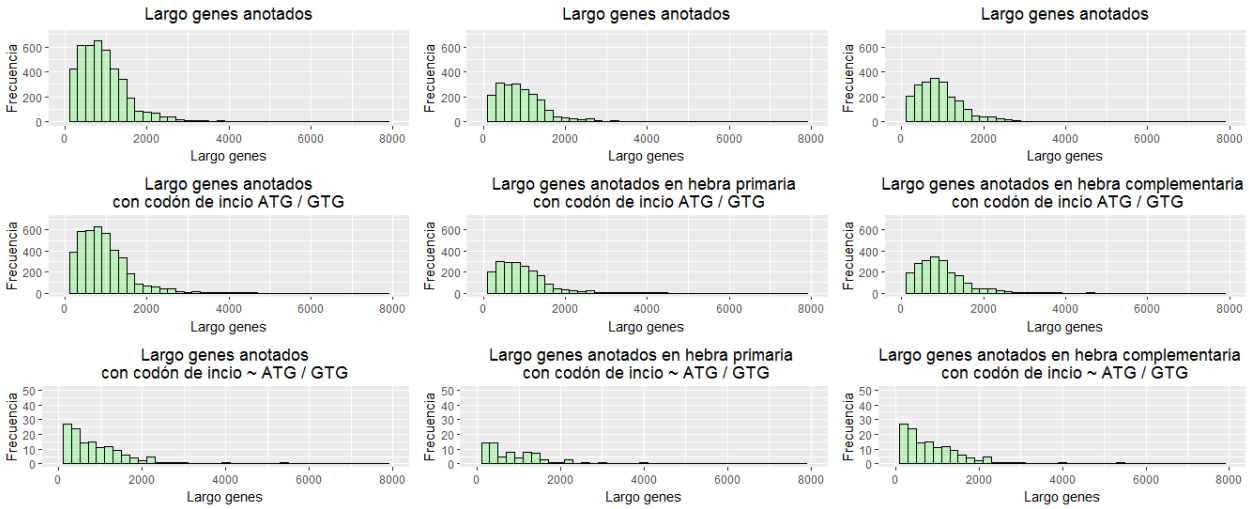


Figura 3.2: Distribución de los largos de genes anotados segmentados por tipo de codón de inicio y tipo de hebra.

Se observa que no hay una diferencia significativa entre las distribuciones del largo de todos los genes y de aquellos comenzando con un codón de inicio en el conjunto $\{ATG, GTG\}$ debido al gran peso que tiene este subconjunto en el total. Tampoco se observa mayor diferencia según el tipo de hebra. Si es importante destacar que más del 50 % del peso de la distribución

corresponde a genes con un largo inferior a 850 *pb.* y que aproximadamente el 93 % tiene un largo menor a 2000 *pb.* e independiente de la hebra que se escoja. Sin embargo, dentro de aquellos genes con menos de 850 *pb.*, un 1,1 % tiene un largo menor a 100 *pb.* y un 4,1 % menor a 200 *pb.* Por tanto, las distribuciones del largo de los genes están inclinadas hacia la izquierda sugiriendo que hay más genes de largo mediano que genes excesivamente cortos y largos. Bajo este mismo análisis se ve que los genes que no comienzan con un codón *ATG* o *GTG* tienen una distribución con mucha más masa sobre aquellos con un largo pequeño, donde por ejemplo el porcentaje de genes con largo menor a 200 *pb.* sube a un 12 % y donde la mediana además se alcanza en un largo de 700 *pb.*

Un segundo análisis, esta vez sobre la base "RF", se muestra en la Tabla 3.5. En ella se resumen varios atributos para los seis marcos de lectura: número de candidatos a genes (n_{i}), el número de genes anotados y el largo mínimo, máximo y promedio de los candidatos.

Hebra	**Marco de Lectura**	**n_i**	**n_i anotados**	**Largo mín**	**Largo máx**	**Largo prom**
Primaria	+1	20277	471	5	4616	107.6
	+2	20150	501	5	4568	109.2
	+3	20506	483	5	7103	107.2
Complementaria	-1	20077	542	5	4961	109.0
	-2	20062	521	5	4562	110.1
	-3	20277	496	5	4709	108.0

Tabla 3.5: Estadísticos básicos en base "RF" de marcos de lectura.

Se observa que en todos los marcos de lectura los candidatos a genes superan los veinte mil y que de aquellos solo el 2.5 % en promedio son genes anotados en "Características", es decir, 3014 candidatos de 121299 son genes anotados en GenBank. Además, el largo promedio de los genes es muy bajo en comparación con el largo promedio de los anotados, lo que se interpreta como una abundancia de candidatos pequeños. En el histograma mostrado en la Figura 3.3 se observa la distribución de largos de los candidatos combinando todos los marcos de lectura. Cada barra representa un aumento de 100 bases en el largo a medida que se avanza en el eje de las abscisas, por lo que el 89.7 % de los candidatos a genes tiene un largo inferior a 200 bases. Sin embargo, al consultar "Características" se observa que la cantidad de genes anotados con un largo menor a 100 bases son solo 49 (1.15 % del total) y solo 128 (3 %) tienen un largo entre 101 y 200 bases. Esto último es alarmante cuando se quiera hablar de la sensibilidad en la identificación de genes por tanto es necesario inicializar la búsqueda de genes verdaderos desde bases de datos conmenores cantidades de candidatos. Es por ello que se usan solo los candidatos en los marcos de "RF" con un largo mayor a 100 bases y está constituirá el "RF" que se usará en adelante.

Tras la condición de tomar solos los candidatos en "RF" con más de 100 bases, los estadísticos se actualizan tomando los valores mostrados en la Tabla 3.6. Se aprecia que aunque la cantidad de genes anotados en la base bajó de 3014 a 2985, no es significativo respecto a la disminución en el número de candidatos a genes (n_{i}) pues este pasó desde 20000 en promedio

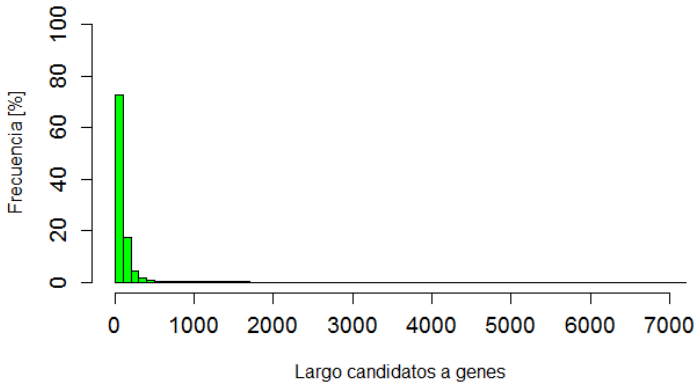


Figura 3.3: Distribución de largos de los candidatos en la base "RF" de marcos de lectura.

por marco de lectura a 5500, lo que aumenta también la sensibilidad de la base con respecto a los genes anotados desde un 1.15 % a un 9 % aproximadamente.

Hebra	**Marco de Lectura**	$\mathbf{n_i}$	$\mathbf{n_i}$ **anotados**	**Largo mín**	**Largo máx**	**Largo prom**
Primaria	+1	5621	468	101	4616	284.2
	+2	5522	497	101	4568	293.3
	+3	5628	476	101	7103	285.4
Complementaria	-1	5370	536	101	4961	298.9
	-2	5548	517	101	4562	292.8
	-3	5561	491	101	4709	288.4

Tabla 3.6: Estadísticos básicos para candidatos de largo mayor a 100 bases en base "RF" de marcos de lectura.

Al repetir el procedimiento anterior sobre candidatos a genes con un largo mínimo cada vez mayor es posible observar la sensibilidad de la base de candidatos con respecto a los genes anotados que poseen. En la Figura 3.4 se muestra la variación de la sensibilidad de genes anotados respecto al total de candidatos cuando el largo mínimo de estos aumenta de a 100 *pb*. En particular se toma una sucesión decreciente de subconjuntos de candidatos de la base "RF" original, donde el primero contiene candidatos con un largo mayor a 100 *pb*. y el último contiene aquellos con un largo mayor a 3500 *pb*. Por cada una de estas bases se calcula el número total de candidatos y la proporción de genes anotados que contiene cada una.

Se observa, de acuerdo a lo esperado, que el número de candidatos disminuye progresivamente en las bases cuando el largo mínimo de los candidatos a genes aumenta. Misma observación se tiene para los candidatos que son realmente genes anotados y además, cuando se segmentan por hebra se aprecia que la diferencia en el número de genes anotados entre la hebra primaria y complementaria suele ser estable a favor de esta última hasta la base con candidatos de un largo mayor a 2400 *pb*., momento en el que se revierte la situación a favor de la hebra

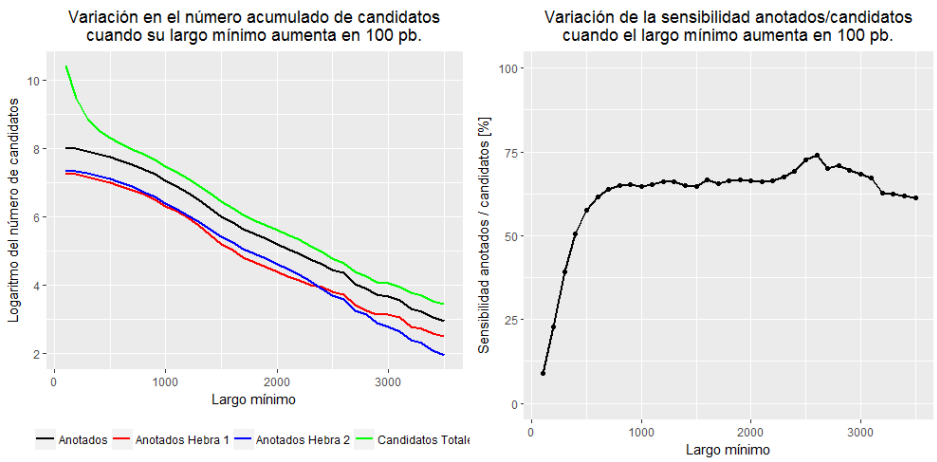


Figura 3.4: Variación del número de candidatos y de la sensibilidad respecto a genes anotados cuando el largo mínimo de los candidatos aumenta cada 100 *pb.* desde 100 *pb.* a 3500 *pb.*

primaria con cada vez mayor diferencia respecto a la hebra complementaria. Otro punto importante por notar es la clara diferencia entre la cantidad total de candidatos entre las bases que contienen a aquellos con un largo mayor a 100 *pb.* y 200 *pb.* respecto al a pesar de que el largo esté en una escala logarítmica, sin embargo, el número de genes anotados no presenta dicha diferencia tan significativa. Esto da cuenta del comportamiento de la sensibilidad de las bases respecto a los genes anotados mostrado en el gráfico de la derecha: se ve un aumento sostenido en la sensibilidad hasta la base con candidatos de un largo mayor o igual a 600 *pb.* (2081 genes anotados en 3380 candidatos), donde posteriormente se estabiliza entorno a un 60 % aunque alcanza su máximo de un 74 % cuando los candidatos tienen un largo mayor a 2600 *pb.* (77 genes anotados en 104 candidatos).

Por otro lado, para tener una idea de cómo se distribuyen los largos de los candidatos a genes a medida que se aumenta el largo mínimo de estos desde los 100 *pb.* a los 3500 *pb.*, se utiliza un gráfico de caja y bigotes por cada una de estas bases. De acuerdo a la Figura 3.5 es claro que al ir aumentando el largo mínimo exigido en los candidatos a genes la mediana también lo hace, por lo que es normal que las cajas vayan subiendo de posición a medida que nos movemos por el eje de largos mínimos. Sin embargo, lo importante a considerar es lo estrecha que es la caja que representa la base con candidatos a genes de un largo mayor a 100 *pb.* y lo poco que se extienden sus bigotes desde esta. Dado el bajo valor de su mediana quiere decir que gran parte de los candidatos son de un largo pequeño, por lo que no es raro la presencia de una gran cantidad de **outliers**, región donde se encuentran también la mayor parte de genes anotados. A medida que aumenta el largo mínimo, las distribuciones empiezan a variar más entorno a la mediana y la cantidad de datos atípicos también empiezan a disminuir dada la mayor extensión de los bigotes. Este comportamiento nos permite concluir que mientras mayor sea el largo mínimo requerido de los candidatos más genes largos son incorporados en la distribución natural de los datos. Un hecho curioso por lo demás es que en todas las bases se presenta un dato atípico representando a un candidato de largo 7103 *pb.*, el que contiene al gen anotado más largo (7076 *pb.*) presente en la base de genes anotados comenzando con ATG o GTG. Este candidato es bastante largo comparado al que le sigue, el cual tiene 4961 *pb.* por lo que es natural que se escape bastante de la distribución inducida por los datos.

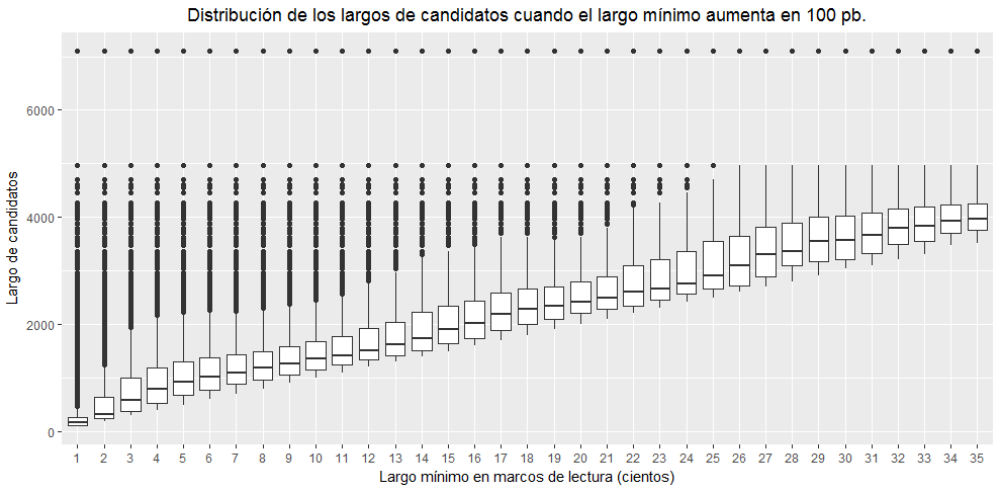


Figura 3.5: Distribución del largo de los candidatos en función del largo mínimo de *pb.* exigida en las bases.

3.2.1. Análisis de la secuencia de consenso

Como se describió en el capítulo de preliminares sobre genómica, la información relevante que necesita la proteína ARN-polimerasa durante la transcripción genética para reconocer el sitio de inicio de un gen se encuentra en la región previa a este desde la dirección de lectura de la hebra (dirección upstream tomando como referencia el codón de inicio o dirección de lectura desde el extremo 5' de la hebra). Esta región de unos cuantos pares de bases se conoce como región promotora pues una vez decodificada la información que en ella se encuentra la maquinaria celular sabe que prontamente está por presentare un gen. Se mencionó también que estas regiones se diferencian según el estado evolutivo de la célula teniendo diferentes secuencias de bases nitrogenadas y puntos de aparición previo al inicio de un gen. En el caso de estudio que nos concierne (bacterias), se ha logrado determinar que son dos las principales secuencias de nucleótidos que juegan un rol fundamental como regiones promotoras: la secuencia denominada **Caja de Pribnow 5'-TATAAT-3'** encontrada aproximadamente 10 nucleótidos previos al comienzo de un gen (ver[5]) y otra secuencia (sin denominación aún) **5'-TTGACA-3'** encontrada 35 nucleótidos previos al gen (ver[5]). Ambas secuencias son de consenso, pues se calculan de manera tal que los nucleótidos en cada una de sus posiciones son los que aparecen con más frecuencia al alinear muchas secuencias en un proceso de alineamiento usado típicamente para encontrar patrones (ver[13]). Por tal motivo, no es una regla que ambas secuencias se encuentren intactas en todos los organismos y genes sino más bien, se suelen presentar solo unos cuantos nucleótidos de ellas. Por otro lado, la cantidad de nucleótidos que aparecen previos al gen también es un número de consenso por lo que pueden sufrir variaciones caso a caso, aunque se ha estudiado que el número óptimo de bases entre ambas secuencias es de 17 *pb.* Dependiendo el organismo existen otras

secuencias de consenso que cumplen el rol de regiones promotoras y que están ubicadas cerca de las posiciones 42 y 52 previas al inicio de un gen siendo ellas **5'-AAAAAARNR-3'** y **5'-AWWWWWTTTTT-3'** respectivamente donde W = A/T, R = A/G y N = A/T/G/C (ver[11], [2]).

Teniendo en mente lo explicado en el párrafo anterior es importante conocer cuál es la secuencia de consenso para los genes anotados encontrados en la base "características" (comenzando con *ATG* o *GTG*) y que se denominará por secuencia de consenso anotada. Posteriormente, en base a dicha secuencia, se pretende establecer una medida de comparación con las secuencias de consenso obtenidas desde las bases de candidatos a genes a medida que se varía el largo mínimo de estos. Para dicho propósito y sabiendo que las posibles regiones promotoras se encuentran en las posiciones aproximadas de 10, 35, 42 y 52 nucleótidos previos al inicio de un gen, se calcularán dichas secuencias de consenso utilizando 70 posiciones previas a cada gen comenzando con un codón de inicio *ATG* o *GTG*.

Para extraer las secuencias de consenso se alinean todos los genes y candidatos a partir de sus codones de inicio y por cada posición previa a estos se calculan las frecuencias de aparición por cada una de las bases. Luego, por cada posición se extrae la base con mayor frecuencia y la secuencia resultante es precisamente la secuencia de consenso. Luego de aplicar este procedimiento para los genes anotados la secuencia de consenso resulta ser:

5' - **TTTTTTTTTTT**TTTTATATTTTAATA**TTAATA**ATTTATTTTATTTTAA**TATAAA**AAAAGGGG
AAAAATT - 3'

Las regiones marcadas con negro corresponden a las secuencias más cercanas a aquellas indicadas como promotoras según la literatura. Notemos que la secuencia **TATAAA** es muy similar a aquella definida por la Caja de Pribnow (ver[5]) difiriendo únicamente en la última base en que aparece una *A* en lugar de una *T* y además aparece en torno a la posición 18 desde el extremo 3'. También se observa la secuencia **TTAATA** la que es similar a la encontrada en la literatura (ver[5]) en torno a la posición 35 - **TTGACA** - aunque la primera presenta la base *A* en lugar de *G* y *C* y además está ubicada en torno a la posición 40 desde el extremo 3'. Es importante notar también que el espacio entre ambas secuencias es de 16 *pb*. lo que es muy cercano al óptimo de 17 *pb* indicado previamente. Por último se observa la secuencia **TTTTTTTTTTT** ubicada en torno a la posición 64 desde el extremo 3' la cual podría estar relacionada con **AWWWWWTTTTTT** descrita en torno a la posición 52 donde las bases *A* y *W* son reemplazadas por *T* (ver[2]). Esta información da cuenta entonces de regiones que posiblemente tienen una gran importancia a la hora de identificar genes.

Como se mencionó antes, la idea es comparar la secuencia de consenso anotada con las secuencias de consenso obtenidas de la secuencia de bases de candidatos a genes usada anteriormente en el análisis de sensibilidad de genes anotados. Es una secuencia decreciente de bases de datos donde la primera corresponde a aquellos candidatos con un largo superior a 100 *pb*, la última corresponde a aquellos con un largo superior a 3500 *pb* y donde además el largo mínimo se va aumentando de a 100 *pb* entre una y otra.

Para comparar las secuencias de consenso, en la literatura se suelen usar algoritmos de alineamiento local y global con el propósito de encontrar patrones de semejanza inaltera-

bles entre las secuencias. Los métodos actuales se basan primordialmente en el algoritmo de alineamiento global de secuencias de Needleman-Wunsch (ver [8]) y en el algoritmo de alineamiento local de Smith-Waterman (ver [16]). En esta memoria, con el fin de establecer una noción simple de comparación entre secuencias de nucleótidos, se utiliza una simplificación de los alineamientos globales en que únicamente se comparan las secuencias posición por posición. Se introduce un poco de notación al respecto: sean $S = s_1 s_2 s_3 ....s_k$ y $S^{'} = s_1^{'} s_2^{'} s_3^{'} ....s_k^{'}$ dos secuencias cualesquiera de largo k donde $s_{\mathrm{i}}, s_{\mathrm{i}}^{'} \in \{A, T, G, C\}$ $\forall \mathrm{i} = 1, ..., k$. Definimos entonces la función $D_k : \mathfrak{S}_k \times \mathfrak{S}_k \to [0, 1]$ por:

$$D_k(S, S^{'}) = \frac{1}{k} \sum_{\mathrm{i}=1}^{k} 1_{\{s_{\mathrm{i}} = s_{\mathrm{i}}^{'}\}}$$

la cual toma dos secuencias en el espacio $\mathfrak{S}_k$ de secuencias de tamaño k y calcula su similitud comparando posición a posición las bases en ambas. Dos secuencias exactamente iguales tienen una similitud igual a 1 y una similitud igual a 0 en caso contrario. En la Figura 3.6 se visualiza la variación de la similitud entre las secuencias de consenso obtenidas de las bases descritas y la secuencia de consenso anotada, en función del largo mínimo exigido para los candidatos.

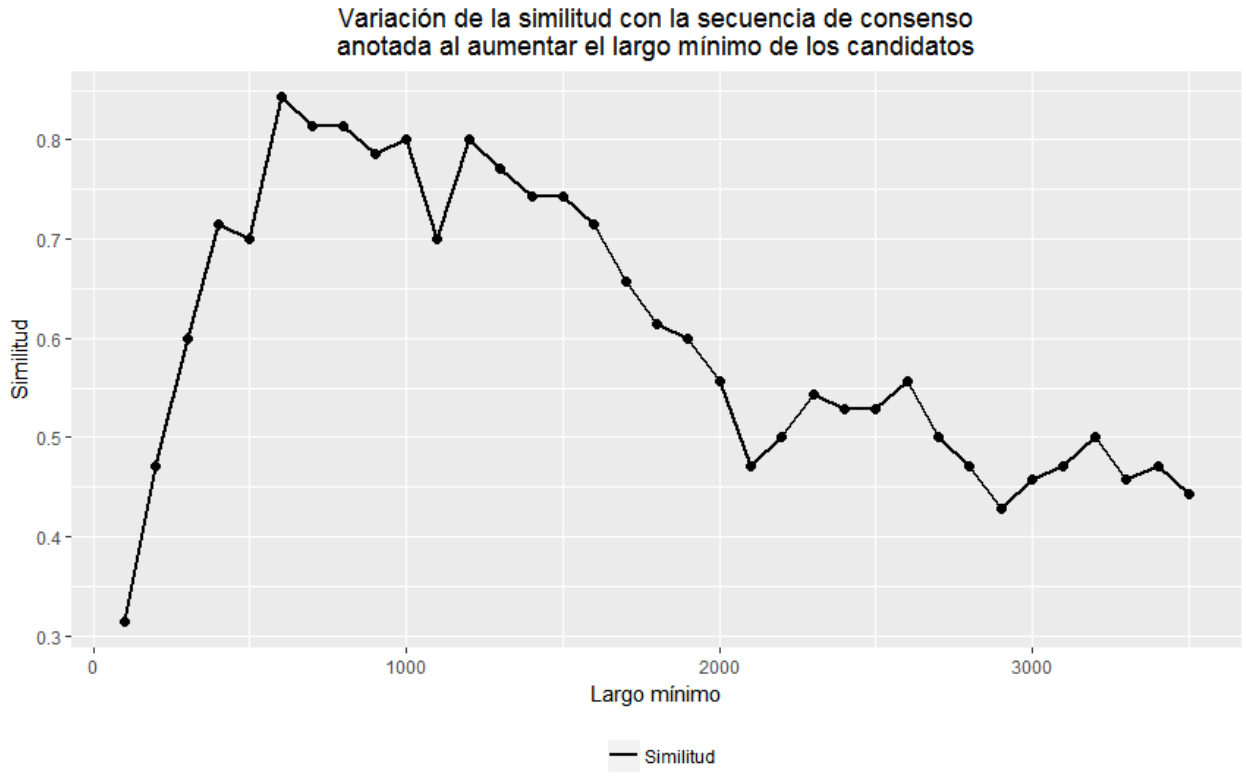


Figura 3.6: Variación de la similitud entre la secuencia de consenso anotada y las secuencias de consenso de los candidatos al aumentar su largo mínimo.

Se observa que la similitud entre la secuencia de consenso anotada y las secuencias de consenso para distintos largos mínimos alcanza su valor más bajo con un 31 % en aquellos candidatos con un largo mayor a 100 *pb.*, y aumenta paulatinamente hasta llegar a su máximo con un 84 % cuando se consideran los candidatos con un largo mayor a 600 *pb.* Luego, la tendencia es a la disminución a medida que aumenta el largo mínimo, llegando finalmente a un 44 % cuando se toma un largo superior a 3500 *pb.* Una observación clave a notar es la similitud entre esta curva con la de sensibilidad de genes anotados mostrada en la Figura 3.4 cuando se restringen los valores del largo mínimo entre 100 y 1000 *pb*: cuando la sensibilidad es muy baja y predominan candidatos espurios (por ejemplo cuando el largo mínimo es de 100 *pb.*), la similitud entre las secuencias de consenso tiende a ser baja debido posiblemente a que aquellos candidatos presentan secuencias muy disímiles a las codificadas por los genes anotados lo que manifiesta su carácter de no gen. Por otro lado, cuando la sensibilidad es alta, el número de genes anotados presentes en la base aumenta y por tanto la secuencia de consenso tenderá a agrupar mayor información relativa a estos lo que se traduce en una mayor similitud con la secuencia de consenso anotada.

Sin embargo, al superar el largo mínimo de 1000 *pb.* los comportamientos son muy diferentes: la sensibilidad se mantiene estable en torno a un 60 % mientras que la similitud de las secuencias de consenso tiende a la baja. Una justificación plausible a este fenómeno es que posiblemente el tipo de información situada en las regiones previas a los genes anotados varía según el tamaño que posean estos.

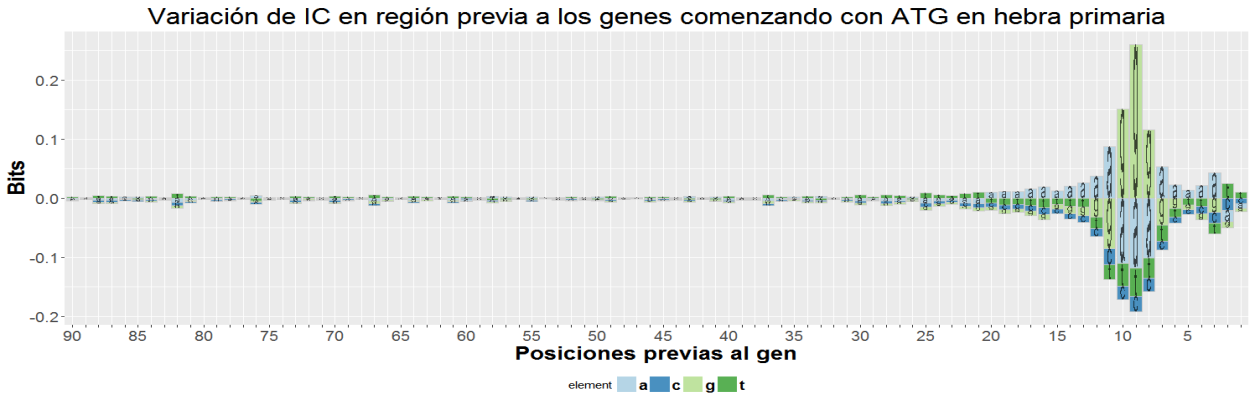
La información expuesta por las secuencias de consenso constituye un primer paso en la búsqueda de patrones que se asemejen a los presentes en los genes anotados, sin embargo, brinda un esquema muy general sobre como debiesen ser los verdaderos genes puesto que solo se consideran las bases más probables de encontrar posición en cada posición. Una alternativa en la búsqueda de patrones a la secuencia de consenso resulta ser la información o entropía contenida en cada una de las posiciones que definen un alineamiento de secuencias de nucleótidos. Los alineamientos de secuencias de ADN se representan usualmente como una matriz de posiciones ponderadas (PWM en inglés) de dimensión $J \times W$ donde J denota el número de letras en el alfabeto a estudiar siendo este en el presente trabajo $\{A, T, G, C\}$, y W denota la posición relativa en las secuencias alineadas. Se define entonces el contenido de información en bits de la posición w del alineamiento (ver[14]) como:

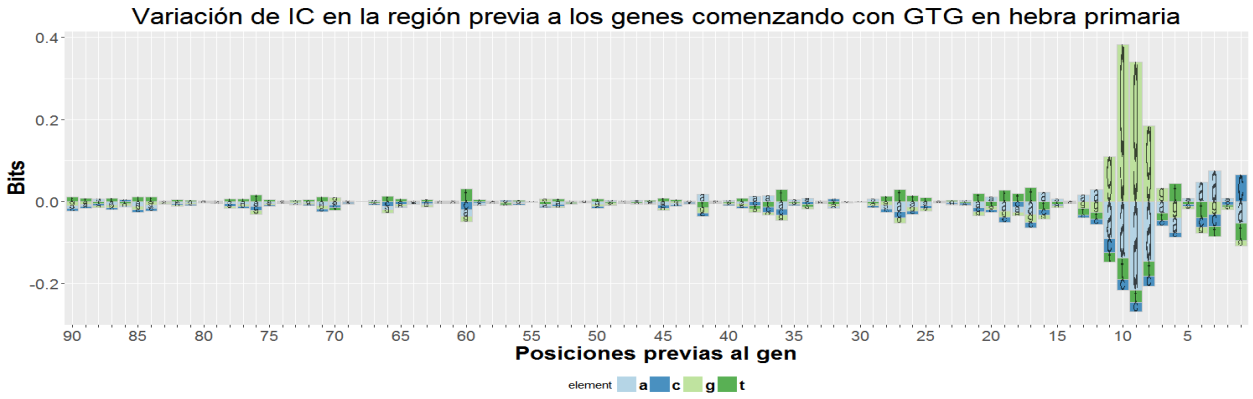
$$IC(w) = \log_2(J) + \sum_{j=1}^{4} p_{jw} \log_2(p_{jw}) = \log_2(J) - entropía(w)$$

Para estudiar el contenido de información (IC) de los genes anotados se considera la región de 90 nucleótidos previa a su inicio por lo que $W = 90$, y así la matriz de posiciones ponderadas resulta ser de dimensión 4×90 donde cada una de las filas da cuenta de la frecuencia relativa dentro del alineamiento de cada una de las cuatro bases nitrogenadas. Por lo dicho anteriormente es claro que la suma por columnas debe ser igual a 1 y que además dado que hay 4 letras en el alfabeto, entonces el término $\log_2(J)$ es igual 2. Así, una posición en el alineamiento en que todas las bases ocurren con igual probabilidad tiene $IC = 0$, y en caso en que solo una se manifiesta en dicha posición da cuenta de $IC = 2$. Lo anterior se

interpreta de la siguiente forma: las posiciones que son altamente conservadas y por tanto presentan menor resistencia al cambio tienen un IC alto mientras que aquellas que son menos conservadas y por tanto varían más tienen un IC bajo.

Una forma de graficar el IC sin perder la información de las frecuencias relativas en la matriz de posiciones ponderada es realizar un gráfico de logotipo de las secuencias alineadas. Schneider y Stephens (ver[14]) diseñaron este método el cual consiste en graficar el IC versus las posiciones en el alineamiento, con la sutileza que posición a posición se grafica una columna con las cuatro letras representando las bases nitrogenadas. La altura de la columna es proporcional al valor de IC y el tamaño de cada letra dentro de esta informa sobre la frecuencia relativa de cada base en dicha posición. A continuación se muestran los gráficos de logotipo para los genes anotados segmentados por hebra y por tipo de codón de inicio.





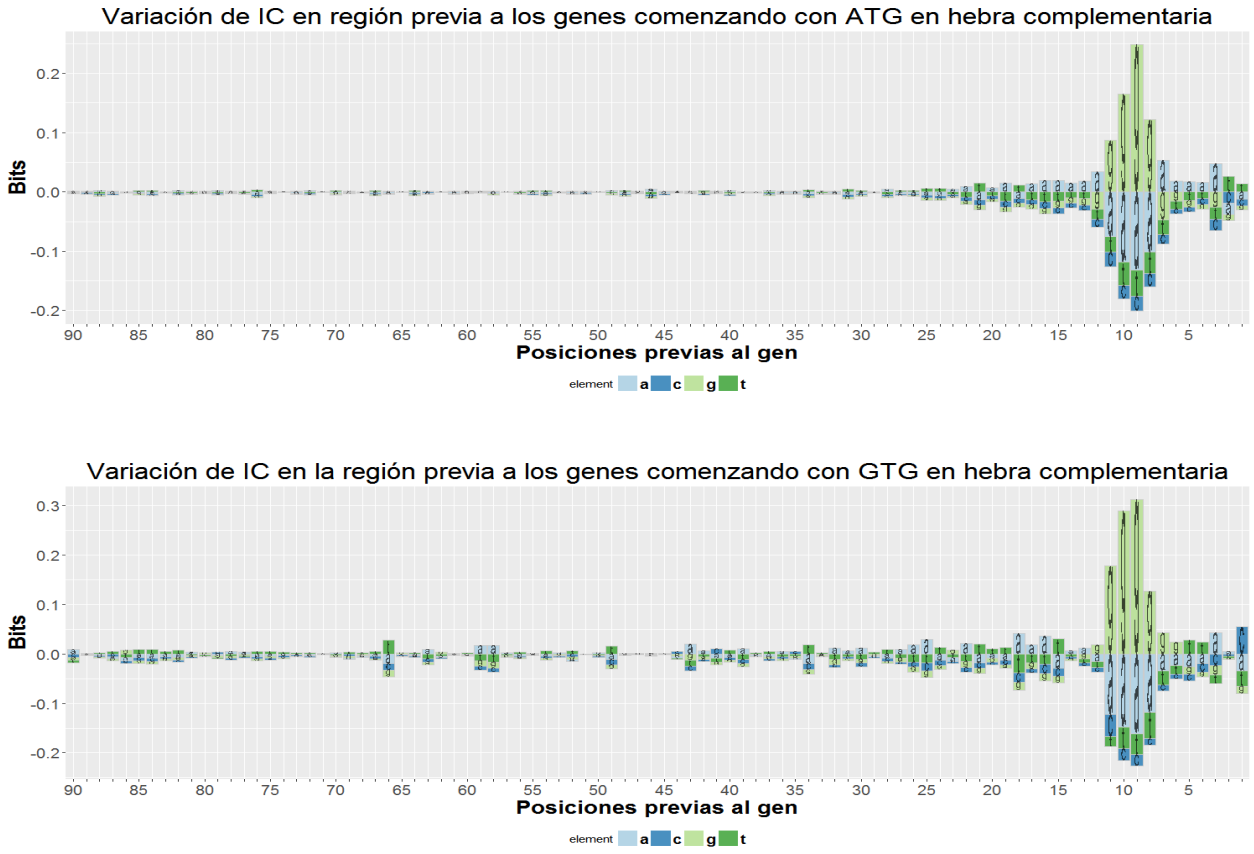


Figura 3.7: Variación de IC en regiones previas a los genes según tipo de hebra y codón de inicio.

El eje que mide el contenido de información (IC) alcanza valores negativos debido a que las columnas de letras se normalizaron de tal manera que aquella que representa la base más frecuente en cada posición esté sobre el valor 0 y el resto bajo este, por lo que el valor de IC se calcula como el largo de la columna o pila. Se debe agregar además que, por motivos de visualización, solo se muestra la secuencia de logotipos hasta 90 posiciones anteriores al inicio de cada gen.

La secuencia de logotipos establece que los genes de la hebra primaria como complementaria tienen un mayor contenido de información (IC) en torno a las primeras posiciones previas a su comienzo: el máximo IC se alcanza en la posición 9 y en su entorno se observa un patrón similar de comportamiento. Esto quiere decir que las primeras posiciones son las más conservadas entre los genes anotados. Cuando el análisis se segmenta según el tipo de codón de inicio se observan diferencias claras en el comportamiento general del IC. En los genes de ambas hebras comenzando con ATG se ve que el IC máximo alcanza un valor aproximado de 0.45 bits, mientras que en aquellos con codón de inicio GTG el valor aumenta por sobre los 0.6 bits y por tanto en estos últimos la conservación de las bases es mayor. Otra diferencia clara es el tipo de base más conservada según el codón de inicio: para ATG la secuencia más conservada en las primeras 13 posiciones es **5'-AAXGGGAAAAATT-3'** donde **X=A** si la hebra es primaria y **X=G** si es complementaria y, para GTG la secuencia para ambas hebras es **5'-AAGGGGGTAAAAC-3'**. Por otro lado se aprecia que el comportamiento del con-

tenido de información tiende a ser más variable para los genes comenzando con GTG después de cierto punto dentro de las 90 posiciones mostradas. Por el contrario, el valor de IC para los genes con comienzo ATG tiende a disminuir progresivamente al ir alejándose del inicio.

3.2.2. Análisis de rendimiento del algoritmo de optimización sobre distintos candidatos a genes

El algoritmo de optimización constituye un esquema sencillo para modelar la organización de los genes basado en el criterio de maximizar la ocupación génica sobre una molécula de ADN seleccionando candidatos que no traslapen entre sí. No toma en cuenta la información codificada a través de los nucleótidos ya que solo usa como inputs las posiciones de inicio y término de los candidatos para lograr una solución óptima. Sin embargo constituye una forma simple de obtener muestras más acotadas de candidatos lo que será clave al momento de clasificar por medio de árboles de decisión.

Es importante saber cómo se distribuyen los genes anotados encontrados en las soluciones del algoritmo de optimización para estimar la sensibilidad de estas muestras respecto a genes reales. Una alta proporción de genes verdaderos en la solución encontrada se traduce en una mayor probabilidad de identificarlos sin cometer muchos errores en las etapas de clasificación.

Para tales efectos se simulan diferentes etapas de optimización según el largo mínimo de los candidatos considerados en los marcos de lectura de la base "RF". Es decir, se genera nuevamente una secuencia anidada decreciente de 35 submuestras desde la base "RF" de tal modo que en cada una se va incrementando el largo mínimo exigido para los candidatos comenzando con todos aquellos de un largo mayor a 100 *pb*. hasta llegar a la última submuestra compuesta de candidatos con un largo mayor a 3500 *pb*. Por cada una de ellas se ejecuta el algoritmo de optimización y se mide la proporción de genes anotados en cada solución segmentando por tipo de hebra y por codón de inicio. En la Figura 3.8 se muestra como varía dicha sensibilidad en función del largo mínimo.

Se observa que para la submuestra con candidatos de largo mayor a 100 *pb*, la solución a la optimización consta con un 37 % de genes anotados. De ese porcentaje sobre un 90 % corresponden a genes comenzando con un codón *ATG* con proporciones similares en ambas hebras. A medida que se incrementa el largo mínimo por submuestra la sensibilidad de las soluciones respecto a los genes anotados comienza a aumentar superando el 68 % una vez pasado el largo mínimo de los 500 *pb*. La proporción según tipo de hebra y codón de inicio sigue manteniéndose pareja en este punto. La sensibilidad total se mantiene relativamente estable en torno al 68 % a medida que aumenta el largo mínimo exigido, aunque con una ligera preponderancia de genes anotados en la hebra complementaria. La sensibilidad máxima se alcanza cuando se consideran candidatos con un largo mayor a los 2600 *pb*. llegando a valores totales por sobre el 75 % de la solución reportada por la optimización respectiva. Después de este punto la sensibilidad tiende a disminuir llegando a un 61 % en la última muestra de candidatos de un largo superior a las 3500 %.

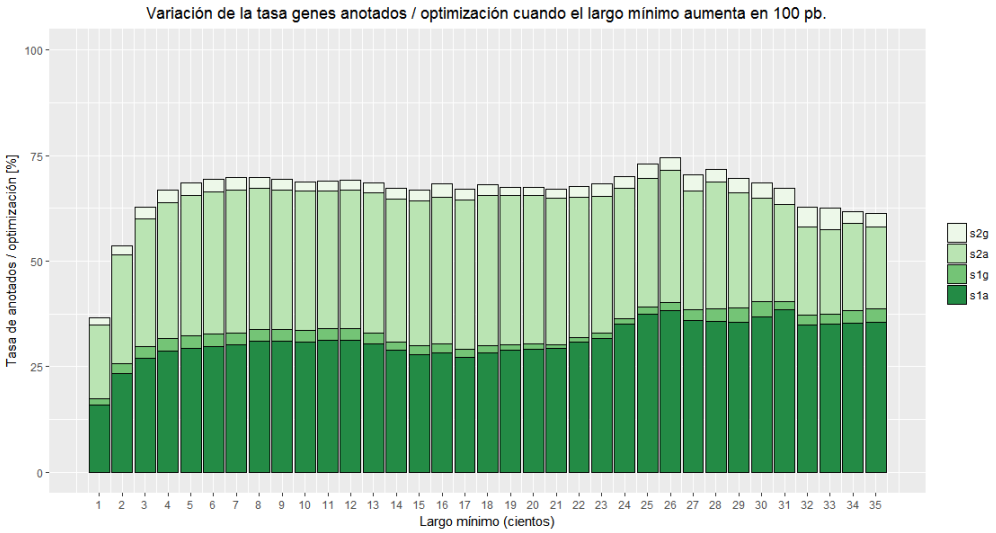


Figura 3.8: Variación en porcentaje de la cantidad de genes anotados en la solución por optimización de acuerdo al tipo de hebra y codón de inicio. Las hebras se codifican como 1/2 para prim./compl. y los codones de inicio posibles como A/G en las etiquetas s1a/s1g/s2a/s2g.

Este resultado sugiere sorprendentemente que a medida que se restringe la búsqueda de genes verdaderos entre candidatos de un largo cada vez mayor entonces más alta será la probabilidad de encontrar un gen efectivamente anotado, es decir, la cantidad de falsos positivos arrojados por la optimización disminuye. Dicho de otra forma, mientras mayor sea el largo de un candidato entonces mayor es la probabilidad de que corresponda a un gen anotado. Esta afirmación solo vale entre cierto rango de valores para el largo mínimo exigido pues, como ya se mencionó, luego de cierta tolerancia la sensibilidad comienza a decaer. Es importante añadir que este comportamiento es similar al observado en las submuestras de candidatos extraídos de "RF" sin utilizar el algoritmo de optimización.

Capítulo 4

Simulaciones y Resultados

En este capítulo se expondrá la metodología de las simulaciones llevadas a cabo junto a los resultados de las mismas. Los algoritmos de clasificación base en todas ellas son los árboles de clasificación estudiados en el capítulo 2. Se mezclan distintos enfoques en la utilización de árboles con el objetivo de imitar - dentro de lo posible - el procedimiento por el cual se expresan los genes en el ADN al interior de la célula. El algoritmo que maximiza la tasa de ocupación génica sobre las hebras de ADN juega un papel crucial en el desempeño que obtendrán los árboles de decisión puesto que permite obtener una submuestra de candidatos más balanceada entre las clases "si" y "no" presentes en las bases Muestra1 y Muestra2. Se inicia el capítulo con la descripción de las principales métricas empleadas para medir el rendimiento de los árboles. Luego, se presentan las técnicas en base a árboles junto a su rendimiento a través de las diversas simulaciones realizadas.

4.1. Medidas de Rendimiento

El objetivo principal de este trabajo consiste en identificar genes usando la información disponible en la secuencia de nucleótidos del ADN de la bacteria *Escherichia Coli*. En el camino para alcanzar dicha tarea se desea entender los mecanismos celulares de transcripción y traducción involucrados en el reconocimiento de genes por lo que, siempre se realizarán experimentos que busquen simular dichos procesos sin caer en el error de usar información no disponible para tales efectos. De este modo solo se emplearán como datos las regiones previas al inicio de los potenciales candidatos a gen pues así se concibe la lectura secuencial $5' \rightarrow 3'$ de la información llevada a cabo por la proteina ARN - polimerasa en una secuencia de nucleótidos en el ADN. En otras palabras, las potenciales zonas codificantes (candidatos a genes) situadas inmediatamente después de los codones de inicio serán intocables por las técnicas descritas posteriormente.

Antes de describir la metodología de las simulaciones es necesario mencionar cual será la

principal medida de éxito que se busca optimizar al momento de comparar los distintos modelos utilizados. La identificación de genes corresponde a un problema de clasificación binaria donde la variable a predecir es aquella que da cuenta de la presencia o no de un gen y que está codificada en las bases **Muestra1a**, **Muestra1g**, **Muestra2a** y **Muestra2g** por la característica **inicio** con posibles valores "si" (gen) y "no" (no gen). La naturaleza de este problema - y en general de todos los referentes a clasificación - permite medir el rendimiento de algún algoritmo a través de una sencilla matriz conocida como matriz de confusión en la cual se registran los aciertos y no aciertos efectuados por el algoritmo. En el caso de clasificación binaria la matriz de confusión tiene dimensión 2×2 y de ella se pueden computar variados indicadores que dan cuenta del rendimiento del algoritmo a través de diferentes criterios. En la siguiente tabla se muestra una típica matriz de confusión para un problema de clasificación binario:

		Valores Reales	
		si	**no**
Valores Predichos	**si**	Verdaderos Positivos (TP)	Falsos Positivos (FP)
	no	Falsos Negativos (FN)	Verdaderos Negativos (TN)

Los valores TP y TN dan cuenta del total de casos que fueron clasificados correctamente de la clase "si" y "no" por un algoritmo de clasificación genérico. El valor FP corresponde al total de casos que siendo clasificados como "si" en realidad pertenecen a la clase "no" y FN es análogo para la situación reversa. Estos dos valores situados fuera de la diagonal de acierto se interpretan como los casos mal clasificados por el algoritmo. Por tanto siempre se debe intentar disminuir dichos valores con el ánimo de conseguir buenos resultados. Los siguientes indicadores son construidos a través de los valores arrojados por la matriz de confusión y son útiles para medir el rendimiento de los diferentes métodos que se usan en esta memoria:

- **Sensibilidad (TPR):** corresponde a la proporción de casos en la clase positiva "si" correctamente predichos y se calcula como $TPR = \frac{TP}{TP+FN}$.
- **Precisión (PPV):** corresponde a la proporción de casos que siendo clasificados en la clase positiva "si" efectivamente son "si" y se calcula como $PPV = \frac{TP}{TP+FP}$.
- **Accuracy (ACC):** corresponde a la proporción de casos correctamente predichos en ambas clases respecto al total y se calcula como $ACC = \frac{TP+TN}{TP+TN+FP+FN}$.
- **Mededida F1:** corresponde a media armónica entre la precisión y la sensibilidad y se calcula como $F1 = 2\frac{PPV \times TPR}{PPV+TPR}$.

La **sensibilidad** nos informa sobre que tan bien un algoritmo de clasificación puede identificar los elementos en la clase de interés o positiva. Si esta es baja, entonces no se capturan los elementos correctos y por tanto la clasificación es paupérrima. Una sensibilidad máxima informa que no hay falsos negativos y por tanto todos los elementos en la clase "si" fueron correctamente clasificados, sin embargo, la ausencia de falsos positivos no está asegurada ya que el algoritmo puede clasificar positivamente más casos de los estrictamente pertenecientes a la clase "si". La **precisión** es la medida que cuantifica la presencia de falsos positivos por lo que si su valor es bajo, entonces muy pocos de los elementos predichos como "si" efectivamente están en dicha clase. La **accuracy** mide la cantidad de aciertos totales para ambas clases. En problemas de clasificación donde el desbalance entre clases es significativo, la accuracy representa una mala de medida de la performance de algún algoritmo pues tiende a ser alta. La razón es que las decisiones de clasificación privilegian a la clase mayoritaria (que no es la interesante en este caso) y por tanto se ocultan los errores sobre la minoritaria. La medida **F1** corresponde a la media armónica entre sensibilidad y precisión y se usa para balancear los resultados obtenidos de estas.

De lo anterior se desprende que la situación ideal de clasificación donde todos los casos son correctamente asignados en sus clases respectivas ocurre cuando la sensibilidad y precisión son máximas. En lo que concierne al problema de clasificación de genes se debe hacer un balance entre estos indicadores buscando maximizar la cantidad de genes encontrados sin caer en la abundancia de falsos positivos. Particularmente la precisión es de suma relevancia ya que si bien es importante reconocer la mayor cantidad de genes, lo es más el estar seguro de que aquellos candidatos predichos positivamente por el algoritmo son efectivamente genes.

4.2. Implementación Numérica

Las simulaciones numéricas fueron implementadas en el lenguaje de programación R el cuál consta con una amplia variedad de paquetes de funciones especializados en algoritmos de aprendizaje supervisado en el área del aprendizaje de máquinas. En particular, para árboles de clasificación y regresión, el paquete **rpart** implementa de muy buena manera el algoritmo original CART desarrollado a mediados de la década de los años ochenta por Breiman (ver[1]). Este paquete tiene la ventaja de contar con funciones que permiten seguir el rastro cabal de todo el proceso de construcción de un árbol como también de su posterior poda y, por tal motivo es el elegido para llevar a cabo las pruebas numéricas.

A continuación se presentan los principales parámetros de la función **rpart** que deben ser ajustados para calibrar el crecimiento de los árboles de clasificación:

- **formula:** corresponde al modelo de respuesta y características usado. En lo que sigue esté será **inicio** $\sim$ **u1** $+ .. +$ **u30** o **inicio** $\sim$ **u1** $+ .. +$ **u70** según el caso a estudiar.
- **data:** corresponde a los datos usados para entrenar un árbol. Estas serán Mestra1a, Muestra1g, Muestra2a, Muestra2g o submuestras de ellas.

- **method:** especifica el tipo de árbol a usar: clasificación (çlass") o regresión (.ªnova"). En todo este trabajo se usarán los primeros.
- **parms:** es una lista de parámetros involucrados en las divisiones de nodos. Las probabilidades a priori **prior** de las clases **"si"** y **"no"** se eligieron como la proporción de ocurrencia en los datos. La función de impureza usada se ajusta por el parámetro **split** y en todo este trabajo será la **entropía**. El parámetro **loss** ajusta la matriz de costos usada para penalizar los errores de clasificación y que en este trabajo será siempre c("si" | "no")=3c("n|o" | "si") y c("si" | "si"), c("no"| "no")=0.
- **control:** es una lista de parámetros que controla el crecimiento de un árbol. El parámetro **minsplit** es el mínimo número de observaciones por nodo necesarias para hacer un corte. El parámetro **minbucket** es el mínimo de observaciones en un nodo para declararlo terminal. El parámetro **xval** es el número de validaciones cruzadas hechas para calcular el mejor subárbol podado, el cual se fija en **xval**=10. El parámetro **cp** controla el crecimiento de un árbol. A mayor valor se penaliza más el tamaño del árbol en función del número de nodos terminales.

4.3. Simulaciones

Como incursión exploratoria inicial sobre los datos se realiza una simulación mediante árboles de clasificación sobre las bases de datos **Muestra1a**, **Muestra1g**, **Muestra2a** y **Muestra2g** usando solamente 30 nucleótidos previos al codón de inicio de los candidatos a genes. Cada una de estas bases se divide en dos conjuntos de observaciones: uno para entrenar y ajustar los parámetros de crecimiento y división del árbol llamado conjunto de entrenamiento y otro para evaluar el poder predictivo sobre datos no usados en el proceso de entrenamiento llamado conjunto de prueba. La división se hace de manera aleatoria pidiendo que la proporción de muestras se mantenga en un 70 % - 30 % para el par entrenamiento-prueba asegurando que la proporción entre las clases se mantenga relativamente constante en ambos conjuntos. La razón de las proporciones 70 % - 30 % es que mientras más datos posea el árbol durante la etapa de entrenamiento más información sobre la relación de las variables podrá utilizar en su construcción.

La función de impureza elegida para la selección de los cortes óptimos en la etapa de crecimiento es la entropía. Dado que las clases "si" y "no" están muy desbalanceadas con unas proporciones del 2,8 % - 97,2 % para **Muestra1a** y **Muestra2a** y del 0,3 % - 99,7 % para **Muestra1g** y **Muestra2g**, se emplea una función de costo para las clasificaciones erróneas tal que clasificar mal un elemento de la clase positiva ("si") es tres veces más costoso que clasificar mal uno de la negativa ("no"). Con lo anterior se pretende que el árbol asigne más nodos terminales a la clase "si". Por otro lado, se hace crecer el árbol lo máximo posible por lo que el parámetro de complejidad en la función rpart será cp= 0,0001, la cantidad mínima de observaciones por nodo necesarias para efectuar un corte será minsplit=3 y la cantidad mínima para considerar un nodo como terminal será minbucket= 1.

Luego del crecimiento inicial, los árboles son podados con el fin de evitar el sobreajuste del modelo en los datos de entrenamiento y así poder generalizar los resultados a datos no

vistos en el conjunto de prueba. Para elegir el mejor subárbol podado se realiza una validación cruzada interna de 10 partes durante el proceso de entrenamiento. El tamaño óptimo de dicho subárbol estará dado por aquel que reporte el menor error de clasificación durante la validación cruzada. A continuación se muestra una tabla que contiene información relevante sobre el rendimiento de los cuatro árboles sobre cada uno de los conjuntos de prueba, tales como la matriz de confusión y las métricas descritas anteriormente. Se observa que el rendimiento en los cuatro casos es muy bajo en términos de sensibilidad (TPR) y precisión (PPV) especialmente en las bases de candidatos con codón de inicio GTG. Por el contrario, la accuracy (ACC) es casi perfecta dada la enorme cantidad de candidatos pertenecientes a la clase "no" lo que refleja la paradoja de esta métrica inclinándose prioritariamente hacia la clase mayoritaria cuando estas están en proporciones desbalanceadas.

Base de Datos	**TP**	**FP**	**TN**	**FN**	**TPR**	**PPV**	**ACC**	**F1**
Muestra1a	172	491	21840	380	0.31	0.25	0.96	0.28
Muestra1g	7	51	16992	41	0.14	0.12	0.99	0.13
Muestra2a	170	509	22016	416	0.29	0.25	0.96	0.27
Muestra2g	2	69	17137	47	0.04	0.03	0.99	0.03

Tabla 4.1: Rendimiento de los árboles sobre los conjuntos de prueba asociados a cada base Muestra usando 30 posiciones previas al inicio de los candidatos. Las métricas usadas son la sensibilidad (TPR), precisión (PPV), accuracy (ACC) y el puntaje F1.

Una segunda aproximación consiste en realizar las mismas simulaciones tomando esta vez 70 nucleótidos previos al inicio de los candidatos. A priori este cambio podría ser significativo considerando que alrededor de las posiciones 35, 42 y 52 existen en general zonas que actúan como promotores en la expresión de los genes de acuerdo a la literatura aunque, según el análisis de las secuencias de consenso realizado para los genes anotados en un capítulo previo, dichos promotores se encuentran en posiciones un tanto diferentes a las mencionadas. En el proceso de entrenamiento se utilizaron los mismos parámetros empleados en la simulación anterior como también los mismos conjuntos de entrenamiento. A continuación se muestra el rendimiento de dichos árboles sobre los conjuntos de prueba.

Base de Datos	**TP**	**FP**	**TN**	**FN**	**TPR**	**PPV**	**ACC**	**F1**
Muestra1a	164	529	21802	388	0.29	0.24	0.96	0.26
Muestra1g	9	59	16984	29	0.19	0.13	0.99	0.16
Muestra2a	166	507	22018	420	0.28	0.25	0.96	0.26
Muestra2g	5	77	17129	44	0.1	0.06	0.99	0.08

Tabla 4.2: Resultados de la predicción de los árboles sobre los conjuntos de prueba asociados a cada base Muestra usando 70 posiciones previas al inicio de los candidatos.

Se observa que no hay diferencias significativas entre estos resultados y los anteriores en términos de sensibilidad y precisión por lo que los resultados siguen siendo insatisfactorios. Además, al observar el número de nodos terminales en ambas simulaciones se comprueba que los árboles son excesivamente grandes con valores por sobre las 100 hojas para aquellos relativos a candidatos con inicio GTG y por sobre 1000 para los que comienzan con ATG lo

que indica un clara sobreajuste de los modelos a los datos de entrenamiento.

Dado que las clases en las bases de datos están altamente desbalanceadas en desmedro de la clase de interés, resulta necesario elaborar estrategias que permitan compensar los problemas de clasificación reportados anteriormente.

4.3.1. Estrategia 1: Optimización y Clasificación

Como primera estrategia se usa una combinación del algoritmo de optimización sobre la tasa de ocupación génica junto a los árboles de decisión. El motivo de realizar esto es que mediante la optimización se extrae una submuestra desde las bases Muestra1a, Muestra1g, Muestra2a y Muestra2g que balancea mejor las clases "si" y "no". Inicialmente se ejecuta el algoritmo de optimización sobre los candidatos a genes situados en los seis marcos de lectura de la base "RF". En particular se usan solo aquellos candidatos con un largo mayor a 100 nucleótidos.

El resultado de la optimización arroja los siguientes resultados mostrados en la Tabla 3.3: el algoritmo seleccionó 6449 candidatos a genes como solución óptima al problema de maximizar el área ocupada en una hebra de ADN sin incurrir en solapamientos entre candidatos pertenecientes a distintos marcos de lectura. El porcentaje del área total ocupada llega a un 87,9 % y de los 6449 candidatos 2354 corresponden a genes anotados lo que representa un 36,5 % del total. De dichos genes anotados el 91,6 % comienzan con ATG por lo que solo 198 genes tienen codón de inicio GTG. Se observa que la tasa de genes anotados en las respectivas bases "Muestra" alcanza un valor promedio del 49 % para "Muestra1a" y Muestra2a mientras en las relativas al inicio GTG dicha tasa llega a un 9,5 % promedio para ambas hebras.

Lo esencial a rescatar es que de los 2985 genes presentes a lo largo de los seis marcos de lectura en la base "RF" de candidatos con un largo mayor a 100 *pb.*, el algoritmo de optimización logró seleccionar 2354 (79 %) teniendo como única restricción que estos no traslapen. Por otro lado, al separar los candidatos de esta submuestra en función de su hebra contenedora y del tipo de codón de inicio la proporción entre las clases positiva y negativa cambia radicalmente a lo que ocurría en las simulaciones exploratorias: el balance si/no en los candidatos de ambas hebras con inicio ATG pasó de un pobre 2,8 % a casi un 50 % y en aquellos con comienzo GTG este pasó de un 0,3 % a un 9,5 % lo que sin duda representa una mejora pero que debe ser balanceada con la factible pérdida de genes anotados tras la optimización.

Una vez separadas las muestras se procede a ejecutar la clasificación sobre estos nuevos conjuntos por medio de árboles de decisión. Nuevamente se usa una proporción 70-30 en los conjuntos de entrenamiento y prueba. Estos son muestreados aleatoriamente tal que la proporción de las clases sea aproximadamente la misma en ambos.

Elección de parámetros : La matriz de costos asociada al error de clasificación man-

Base de Datos	Candidatos encontrados	Anotados Encontrados	Tasa de Anotados en Base Muestra [%]
Muestra1a	2114	1031	48.7
Muestra1g	1030	98	9.5
Muestra2a	2274	1125	49.4
Muestra2g	1031	100	9.7

Tabla 4.3: Resultados de la optimización sobre los candidatos a genes en los marcos de lectura con un largo mayor a 100 *pb*.

tiene sus valores penalizando tres veces más clasificar erróneamente una observación como gen cuando en realidad no lo es versus el caso opuesto. El resto de parámetros a ajustar son el parámetro de complejidad (cp), el número mínimo de observaciones por nodos necesario para llevar a cabo un corte (minsplit) y el número mínimo de ellas para considerar un nodo como terminal u hoja. Se buscarán los parámetros que reporten el mejor rendimiento en precisión (PPV) por medio de una validación cruzada de 10 partes. La manera para efectuar dicha optimización es realizar una búsqueda en grilla, es decir, se fijan intervalos de valores para cada parámetro y se simula un árbol por cada tupla (*cp*, *minsplit*, *minbucket*) posible de valores. Los valores en cada intervalo se restringen a una cantidad finita de modo que la búsqueda se logre en una cantidad finita de pasos. La tupla escogida es aquella que reporta la mayor precisión en la etapa de validación cruzada. Es importante aclarar que la tupla óptima puede cambiar en función de la base "Muestra" por lo que es pertinenete aclarar cuál fue seleccionada caso a caso.

En la siguiente tabla se muestra la tabla con las predicciones y rendimientos de los árboles podados en cada conjunto de prueba además de las tuplas de los parámetros óptimos encontrados caso a caso. Se observa que el rendimiento ha mejorado ostensiblemente en función de la sensibilidad (TPR) y la precisión (PPV) observadas en las simulaciones de exploración iniciales. La sensibilidad alcanza su valor más alto en **Muestra1a** con un 68 % y la precisión hace lo mismo en **Muestra2a** con un 92 %. En particular las métricas son más altas en aquellas bases con candidatos que comienzan con *ATG*. Esta diferencia con aquellos de comienzo *GTG* se produce probablemente al desbalanceo entre las clases positiva y negativa en este último caso pero, aún así el desempeño es mejor que aquel observado en las etapas exploratorias.

Base de Datos	TP	FP	TN	FN	TPR	PPV	ACC	F1	Parámetros Óptimos (cp, minsplit, minbucket)
Muestra1a	210	39	285	99	0.68	0.84	0.78	0.75	(0.01, 30, 10)
Muestra1g	8	3	276	21	0.28	0.73	0.92	0.4	(0.01, 20, 7)
Muestra2a	172	15	329	165	0.51	0.92	0.74	0.66	(0.01, 30, 10)
Muestra2g	7	5	274	23	0.23	0.58	0.91	0.33	(0.01, 20, 7)

Tabla 4.4: Resultados de la predicción de árboles sobre las muestras obtenidas post algoritmo de optimización para candidatos de un largo mayor 100 *pb*.

Si bien es cierto que las mejoras en las métricas de desempeño son evidentes, estas dependen fuertemente de la división aleatoria hecha al comienzo en la conformación del conjunto de

entrenamiento y prueba. Los resultados podrían tener una gran varianza en función de la división establecida por lo que es necesario disminuirla con el fin de obtener desempeños verídicos y no inflados. La alternativa para llevarlo a cabo consiste en repetir el proceso anterior una cantidad de N veces con el fin de someter el modelo a diferentes particiones de entrenamiento y prueba. Las métricas resultantes son luego promediadas para obtener una noción más clara del desempeño efectivo de los árboles de clasificación.

Se realizan 10 particiones diferentes entre conjuntos de entrenamiento y prueba tal que sobre cada una se entrena el modelo de árbol utilizando los mismos parámetros optimizados previamente. Este procedimiento se realiza sobre cada una de las bases "Muestra" y los resultados de sus desempeños en los conjuntos de prueba respectivos son promediados. En la siguiente tabla se muestran las métricas asociadas a cada base utilizada. Se observa que los resultados en las distintas métricas variaron respecto a la simulación anterior. Para las bases comenzando con *ATG* la sensibilidad y precisión se estabilizaron en torno al 56 % y 75 % promedio respectivamente mientras que en aquellas comenzando con *GTG* estos alcanzan valores aproximados de un 22 % y 91 % en promedio. Si bien los resultados son diferentes y en algunos casos empeoran, en general siguen siendo bastante mejor que los reportados en la etapa de exploración.

Base de Datos	**TPR**	**PPV**	**ACC**	**F1**	**Parámetros Óptimos (cp, minsplit, minbucket)**
Muestra1a	0.56	0.88	0.74	0.68	(0.01, 30, 10)
Muestra1g	0.21	0.66	0.91	0.31	(0.01, 20, 7)
Muestra2a	0.56	0.89	0.75	0.68	(0.01, 30, 10)
Muestra2g	0.23	0.62	0.91	0.33	(0.01, 20, 7)

Tabla 4.5: Resultados de la predicción de árboles post optimización usando 10 particiones diferentes del par entrenamiento/prueba.

4.3.2. Estrategia 2: Optimización y Clasificación sobre candidatos con codón de inicio degenerado

Las bases de candidatos a genes por marco de lectura con un largo mayor a 100 *pb.* situadas en "RF" tienen 2985 genes anotados lo que representa un 72,4 % de los 4121 genes con inicio *ATG*/*GTG*. Al momento de ejecutar el algoritmo de optimización, la muestra de candidatos obtenidas tendrá una cantidad de genes anotados acotada superiormente por la sensibilidad de la base "RF" respecto a los genes verdaderos, es decir, en una situación ideal no podrá obtener más de 2985 genes verdaderos en su solución. Este hecho también limitará el rendimiento de los árboles de clasificación pues las bases del tipo "Muestra" usadas para su ajuste están elaboradas directamente de la solución a la optimización, por lo que no cuenta con la información de todos los genes anotados. Si el caso fuera el contrario, los árboles

podrían utilizar cabalmente todos los genes para encontrar las relaciones pertinentes en las regiones previas a los codones de inicio.

La limitante en la cantidad de genes anotados que reconoce la optimización es la forma en que se extraen los genes en los distintos marcos de lectura. La lectura de forma secuencial a lo largo de las hebras para encontrar codones de inicio en el conjunto INICIO y codones de parada en TÉRMINO no captura todos los genes anotados. El problema es el siguiente: los genes anotados no encontrados en "RF" están ocultos en otros candidatos presentes en "RF", es decir, si $I = [s(I), f(I)]$ representa un candidato a gen con $s(I) < f(I)$ y $s(I), f(I) \in \{1, ..., n\}$ con n el largo de la cadena de ADN, entonces existe $s^* \in \mathbb{N}$ con $s^* < f(I)$ y $s^* = s(I) + 3k^*$ para cierto $k^* \in \mathbb{N}$ tal que $I^* = [s^*, f(I)]$ es un gen anotado, es decir, $I^* \subset I$. Dicho de otra manera, existen genes verdaderos en el interior de candidatos a genes bajo el mismo marco de lectura. Se habla entonces de candidatos con codón de inicio degenerado pues enmascaran la presencia de un codón de inicio que si reporta un gen con el mismo codón de término. Para extraer dichos candidatos se debe cambiar la manera en que se leen las hebras en cualquiera de sus marcos de lectura: se mantiene la lectura secuencial de codones de inicio y parada explicadas en el capítulo de presentación de los datos con la sutiliza que una vez reportado un codón de término en cierto marco de lectura se recorre nuevamente dicho candidato a gen desde su codón de inicio y en el mismo marco de lectura hasta encontrar un nuevo codón de inicio en el conjunto INICIO. Dicha subregión se registra como un nuevo candidato en el marco de lectura al que pertenece el candidato que la contiene y en caso de no existir se procede de la manera secuencial habitual.

En la siguiente tabla se resumen las principales estadísticas de esta nueva base de candidatos denominada **RFD** (Reading Frame Degeneracy), de la misma forma que se hizo para la base **RF** e imponiendo un largo mínimo por candidato mayor a 100 *pb*. La observación más importante es que a través de este nuevo proceso de extracción se obtuvieron un total de 124719 (n_{i}) candidatos a genes de los cuales 4028 corresponden a genes anotados o verdaderos. Los 4028 genes encontrados corresponden al (97,7 %) del total que comienzan con ATG/GTG lo que corresponde casi a la totalidad de ellos. Sin embargo, también se extrajeron una alta cantidad de candidatos espurios lo que disminuye considerablemente la sensibilidad de la base respecto a los anotados a un 3,2 %. Otro hecho importante a destacar es el aumento en el largo promedio de los candidatos llegando a las 500 bases aproximadamente para los seis marcos de lectura. Este hecho es llamativo puesto que si bien aumentó considerablemente la cantidad total de candidatos estos contribuyen al aumento del largo promedio y la razón a esto es que los candidatos degenerados difieren en muy pocas bases de aquellos que los contienen además de existir en aquellos candidatos de un largo mediano en adelante.

En este punto es interesante saber cómo será el rendimiento del algoritmo de optimización sobre la base "RFD". Para ello basta simular la optimización con cada marco de lectura visto como una hebra diferente tal como se hizo en la Estrategia 1 y sin preocuparse de la degeneración de los codones de inicio pues, según la *Proposición 2.14* del capítulo 2, cada candidato degenerado puede ser concebido como un nuevo marco de lectura y por tanto la optimización funciona de la manera usual. En el siguiente recuadro se muestran los resultados tras la optimización usando los candidatos en "RFD" con un largo mayor a 100 nucleótidos. La solución del algoritmo de optimización arroja un total de 6787 candidatos con una

Hebra	Marco de Lectura	n_i	n_i anotados	Largo mín	Largo máx	Largo prom
Primaria	+1	20214	638	101	4616	490.5
	+2	21031	681	101	4568	505.3
	+3	20439	642	101	7103	510.5
Complementaria	-1	21134	709	101	4961	492.3
	-2	21285	690	101	4562	497.2
	-3	20616	668	101	4709	496.9

Tabla 4.6: Estadísticos básicos para candidatos de largo mayor a 100 nucleótidos en base "RFD" de marcos de lectura con codón de inicio degenerado.

ocupación en la doble hebra de ADN del 91,7 % ambas cantidades superiores en comparación a la solución presentada en la Estrategia 1. Contrario a lo que se piensa, la cantidad total de genes anotados encontrada es de 1894 observaciones lo que está muy por debajo de los 2354 encontrados en la optimización anterior. Del mismo modo la tasa de anotados respecto al total de candidatos en cada base del tipo "Muestra" disminuye aproximadamente un 10 % para aquellas que comienzan con ATG y un 3 % para las que comienzan con GTG.

Los resultados sobre la optimización parecieran ser peores que los encontrados en la estrategia previa, pero de todos modos es importante simular los árboles respectivos para entender si hay información que incida de manera diferente en sus resultados.

Base de Datos	Candidatos encontrados	Anotados Encontrados	Tasa de Anotados en Base Muestra [%]
Muestra1a	2127	828	38.9
Muestra1g	1219	77	6.3
Muestra2a	2226	909	40.8
Muestra2g	1215	80	6.6

Tabla 4.7: Resultados de la optimización sobre los candidatos a genes en los marcos de lectura de "RFD" con un largo mayor a 100 *pb*.

El procedimiento de entrenamiento y prueba es repetido de forma exacta a lo hecho en la Estrategia 1. Se separan los candidatos post optimización de acuerdo a su hebra y codón de inicio respectivos. Posteriormente se extraen las características de las regiones previas a su comienzo considerando 70 posiciones o nucleótidos. La proporción entrenamiento/test es nuevamente $70 - 30$ y para evitar la varianza de la elección de una partición en particular se llevan a cabo 10 remuestreos entrenamiento/test por cada base "Muestra" cuya performance será calculada como el promedio de las métricas ya establecidas anteriormente.

Elección de parámetros: La matriz de costo será idéntica a la usada durante la Estrategia 1 penalizando tres veces un falso positivo que un falso negativo. Para el resto de los parámetros se vuelve a utilizar una búsqueda en grilla para encontrar la tupla (*cp*, *minsplit*, *minbucket*) que reporte la máxima precisión con el fin de cometer pocos falsos positivos. Esta

búsqueda se realiza a través de una validación cruzada de 10 partes sobre el conjunto de entrenamiento y se elige el trío de valores que reporte la mayor precisión.

En la siguiente tabla se resume la performance de los árboles de clasificación en los conjuntos de prueba para cada base del tipo "Muestra" indicando las diferentes métricas y parámetros óptimos utilizados.

Base de Datos	**TPR**	**PPV**	**ACC**	**F1**	**Parámetros Óptimos (cp, minsplit, minbucket)**
Muestra1a	0.45	0.87	0.76	0.60	(0.01, 30, 10)
Muestra1g	0.20	0.48	0.93	0.27	(0.001, 20, 7)
Muestra2a	0.49	0.86	0.76	0.62	(0.01, 30, 10)
Muestra2g	0.16	0.49	0.93	0.22	(0.001, 20, 7)

Tabla 4.8: Resultados de la predicción de árboles post optimización en candidatos degenerados de "RFD". Se usan 10 particiones diferentes del par entrenamiento/prueba.

Se observa que el desempeño de los árboles merma respecto a lo obtenido en la Estrategia 1. La sensibilidad (TPR) disminuye para todas las bases de datos llegando en el mejor de los casos a un 49 % en "Muestra2g" y en el peor a un 16 % en "Muestra2g". Por otro lado se ve que la precisión (PPV) se ha mantenido relativamente igual para las muestras de candidatos con comienzo ATG en ambas hebras y cuyo valor se alza sobre el 85 %. Sin embargo, misma métrica se ve disminuida respecto a Estrategia 1 para aquellas bases cuyos candidatos comienzan con GTG llegando a un escuálido 48 % aproximadamente en ambas hebras. La razón del empeoramiento en la perforfmance de estos árboles se debe posiblemente al mayor desbalance entre las clases "si" y "no" post optimización, producto de la menor cantidad de genes anotados encontrados en dicha etapa como también por el aumento de falsos positivos. En efecto, si observamos la solución tras la optimización de las bases "Muestra1g" y "Muestra2g" en el cuadro 15, estas son las únicas que han aumentado en número de candidatos respecto a la Estrategia 1, con un valor cercano a 200. Al combinar lo anterior con el hecho de que también disminuyen los genes anotados resulta convincente la merma tanto en sensibilidad como precisión. Por otro lado, en las bases "Muestra1a" y "Muestra2a" el número de candidatos post optimización se ha mantenido relativamente constante a lo observado en Estrategia 1 y lo único que disminuyó fue la cantidad de anotados. Esto se traduce en una menor sensibilidad pero en una preservación de la precisión dado que no aumentaron los candidatos espurios y por tanto los resultados de los árboles se condicen con los de la optimización.

A continuación se visualizan los árboles obtenidos por tipo de base "Muestra". La información importante por rescatar en la Figura 4.1 es la relativa a las decisiones tomadas por los árboles para efectuar la clasificación. Los cortes disminuyen en importancia a medida que se baja por el árbol puesto que cada vez se logra una menor disminución de la impureza respecto a la partición previa del espacio de características. Se observa en general para los cuatro árboles, que las variables de mayor importancia son aquellas situadas en torno a la posición 10 en dirección $5'$ o upstream. Variables tales como u9, u8, u10, u11, u12 tienden a aparecer con mucha frecuencia en los primeros cortes. Además, siempre se pregunta por si dichas variables toman como valor A o G. Esto se relaciona con el análisis sobre secuencias

de consenso realizado anteriormente, en el que las posiciones más conservadas son efectivamente las que toman mayor importancia en estos árboles y donde las bases A y G son las establecidas en la secuencia de consenso.

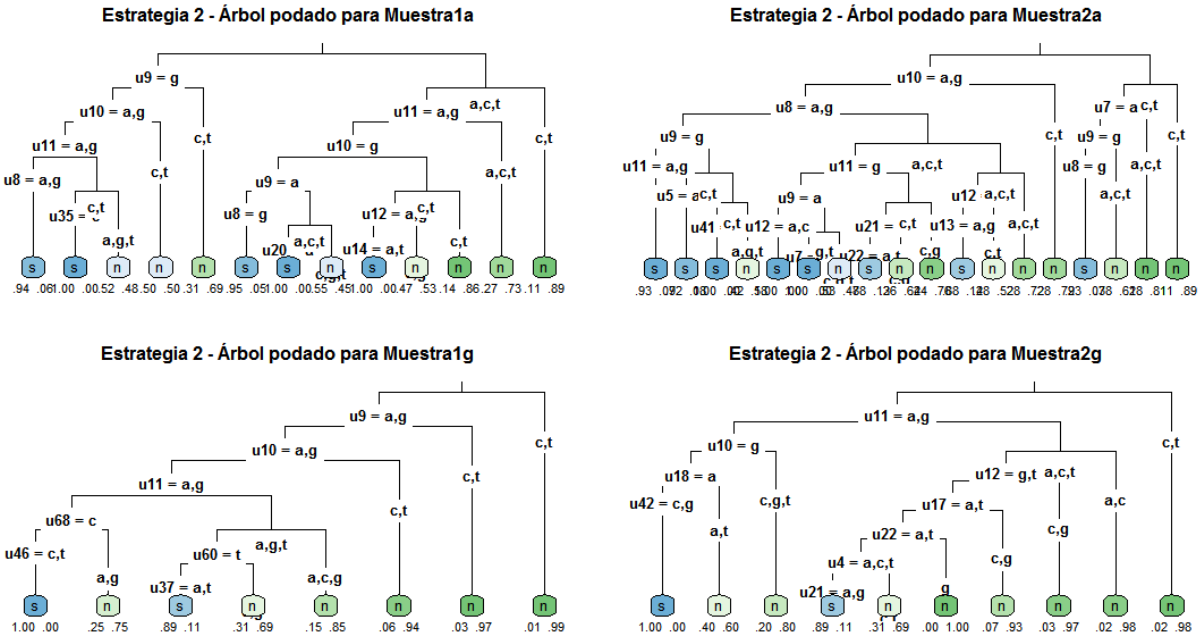


Figura 4.1: Árboles obtenidos durante el entrenamiento de las bases "Muestra". Estas se obtienen post optimización usando las bases de candidatos degenerados "RFD"

Otras variables que aparecen en los árboles son u20-u35 y u21-u41 para las bases "Muestra1a" y "Muestra2a" respectivamente. En particular, u35 y u41 pueden estar relacionadas al promotor ubicado 35 posiciones antes del inicio de la transcripción genética. Para el árbol de "Muestra1g" aparecen las variables u37, u46, u60 y u68 lo que podría indicar la presencia de promotores en dichas posiciones según los valores del contenido de información (IC) reportados en el gráfico logotipo de los genes de la hebra primaria comenzando con GTG.

4.3.3. Estrategia 3: Secuencias anidadas de Optimización y Clasificación

Los niveles de sensibilidad reportados en las Estrategias anteriores son muy bajos si lo que se quiere es identificar genes anotados. Es necesario cambiar la metodología con el objetivo de conseguir mejores resultados. En capítulos anteriores se ha estudiado como varía la sensibilidad de las bases de candidatos respecto a los genes anotados cuando se extraen submuestras de un largo mínimo cada vez mayor. Los resultados expuestos en la Figura 3.4

dan cuenta de una sensibilidad promedio por sobre el 62 % en las submuestras con un largo mínimo mayor a 600 *pb*. llegando incluso a un máximo del 75 % cuando la submuestra tiene genes por sobre los 2600 *pb*. Un resultado de la misma naturaleza se observa si sobre cada una de estas submuestras se simula el algoritmo de optimización. Las soluciones reportadas por estos algoritmos también dan cuenta de una mayor sensibilidad respecto a genes anotados cuando se va aumentando el largo mínimo de las submuestras llegando a un valor promedio del 68 % por sobre los 500 *pb* exigidos.

Como los resultados de la clasificación por medio de árboles depende fuertemente del balance entre las clases positiva y negativa es interesante saber cómo varía el desempeño de estos en las distintas métricas a medida que varía el largo mínimo de los candidatos en la en las soluciones post optimización. En la Figura 4.2 se muestra la variación de la sensibilidad y precisión arrojadas por los árboles sobre los conjuntos de prueba, a medida que aumenta el largo de los candidatos en las soluciones post optimización. Las sucesivas optimizaciones se llevaron a cabo sobre las submuestras de candidatos extraídas desde de la base "RF", por lo que no se usaron genes degenerados. La razón de esto es el bajo rendimiento observado en los árboles sobre candidatos de este tipo en la base "RFD" durante la Estrategia 2.

Para entrenar los árboles por cada base tipo "Muestra" provenientes de las soluciones post optimización, se ejecuta el mismo procedimiento llevado a cabo en las Estrategias anteriores. Cada muestra se particiona en entrenamiento/prueba en una proporción $70 - 30$. Para disminuir la varianza en los resultados al escoger una determinada partición se realizan 10 divisiones aleatorias diferentes del espacio de características en entrenamiento/prueba. Se debe tener el cuidado que las proporciones entre las clases "si" y "no" se mantengan similares en cada una de dichas divisiones. Posteriormente se ajusta un árbol en cada conjunto de entrenamiento y se prueba el modelo en el respectivo conjunto de prueba. Los resultados sobre estos últimos son entonces promediados para disminuir la varianza en las distintas métricas empleadas. Los parámetros utilizados para el ajuste de los árboles son los reportados en la Estrategia 1. La razón de dicha elección es que reportan buenos niveles de precisión.

Se observa que, la sensibilidad inicialmente aumenta progresivamente desde un 55 % en las bases "Muestra1a" y "Muestra2a" cuando el largo mínimo es de 100 *pb*., hasta estabilizarse en torno al 80 % cuando el largo mínimo es de 300 *pb*. Dicha estabilidad se mantiene hasta un largo de 900 *pb*, momento en que disminuye en torno a un 75 %. Luego de los 1200 *pb*. los comportamientos se separan tal que la sensibilidad en "Muestra2a" vuelve a llegar al 80 % mientras que en "Muestra1a" esta disminuye a un 70 %. En las bases relativas a un comienzo con GTG, las sensibilidades alcanzadas son ostensiblemente menores oscilando en torno a un 35 % llegando "Muestra2g" a un máximo de 50 % aproximadamente cuando el largo mínimo es de 900 *pb*. y "Muestra1a" aun 55 % cuando el largo es de 1100 *pb*. El comportamiento de la clasificación para estas bases es más errático y no es posible asegurar que el aumentar el largo mínimo en los candidatos asegura una mayor sensibilidad.

Por otro lado, la precisión es alta y relativamente constante en las muestras comenzando con ATG a medida que aumenta el largo mínimo. Esta varía en torno a un 80 % y 90 %. La razón de estos valores se atribuye a los costos de clasficación errónea utilizados, donde se penaliza tres veces más un falso positivo que un falso negativo. Para el caso de "Muestra1g"

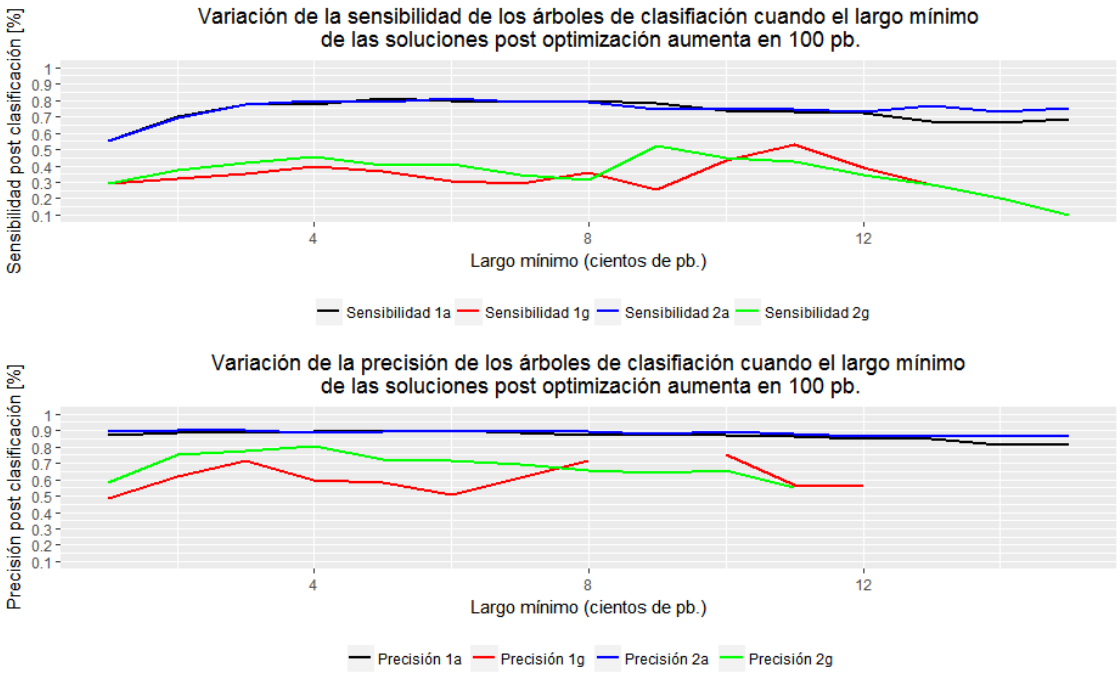


Figura 4.2: Desempeño de los árboles en los conjuntos de prueba post optimización al aumentar el largo mínimo de los candidatos en 100 *pb*.

la variación es más errática y no se puede establecer una mejora luego de 300 *pb*. Por otro lado se aprecia una observación faltante en esta curva cuando el largo es de 900 *pb*. La razón es que el árbol respectivo no clasificó correctamente ninguna observación en la clase "si". En "Muestra2g" la precisión aumenta progresivamente hasta alcanzar un 80 % cuando el largo mínimo es de 400 *pb*., aunque luego la tendencia es a la disminución.

Los resultados anteriores serán claves para estructurar la nueva estrategia en el reconocimiento de genes y una pregunta que surge en este punto es: **¿existe una manera de utilizar los resultados de clasificación de los árboles en el proceso de búsqueda de genes?**. La situación ideal de clasificación es aquella en que no existen falsos positivos ni falsos negativos. En el mundo real las cosas cambian, los clasificadores producen errores debido a que es imposible reproducir las reglas de clasificación o asignación perfectas, dada la enorme complejidad en las relaciones que las definen. Sin embargo es de aquellos errores de los que se puede recuperar información crucial para mejorar los métodos de búsqueda de los elementos de interés, que en este caso son genes.

En un problema de clasificación binaria, cuando un algoritmo predice erróneamente la clase de cierta observación, los valores fuera de la diagonal en su matriz de confusión ya no son cero. Se cometen falsos positivos y falsos negativos. En el problema de identificación de genes la clasificación por medio de árboles particiona el espacio de observaciones en aquellos considerados genes ("si") y en aquellos que no ("no"). En esta última se encuentran los falsos negativos que corresponden a observaciones pertenecientes a la clase positiva. Resulta primordial encontrar un método de búsqueda que rescate dichas observaciones para poder asignarlas a la clase de genes anotados. La clave para ello vuelve a ser el algoritmo de optimización. A este solo le preocupa como se organizan los candidatos en la doble hebra de

ADN por medio de las posiciones que ocupan en la molécula.

Las observaciones predichas como "si" originan una partición en las hebras de la molécula de ADN. Las predicciones provienen desde la solución post optimización por lo que, si se define $\mathcal{P}$ como aquella región en el ADN que ocupan los candidatos a genes predichos en la clase "si" por los árboles de decisión, entonces $\mathcal{P}$ corresponde a una unión disjunta de intervalos de la forma $[s, f)$ donde cada uno representa un candidato predicho como "si" por los árboles. En esta región claramente existen candidatos que corresponden a falsos postivos y que lamentablemente no se pueden discriminar o dejar de lado puesto que corresponden a predicciones positivas de los algoritmos de clasificación. Es por ello que buscar una alta precisión durante la clasificación es muy importante ya que permite tener una alta confianza en los resultados. Se define $\mathcal{N}$ como la región en el ADN complementaria a $\mathcal{P}$ y por tanto contiene a todos los candidatos predichos en la clase "no" por los árboles. Además de ellos, también agrupa a la región no cubierta por la solución del algoritmo de optimización. La idea principal es entonces extraer todos los candidatos posibles en cualquiera de los marcos de lectura de "RF" que se encuentren en la región $\mathcal{N}$. Luego, se busca la organización óptima de ellos sobre la región $\mathcal{N}$ a través del algoritmo de optimización para finalmente volver a clasificar las nuevas soluciones usando árboles de decisión. Este proceso puede ser repetido sucesivamente de acuerdo a la siguiente metodología.

Optimización y Clasificación anidadas Sea $RF(\mathrm{i})_0 \subseteq RF$ una muestra de candidatos en todos los marcos de lectura tal que su largo mínimo es de i pb. Entonces:

1. Sea S_0 la solución del algoritmo de optimización usando los marcos de lectura en $RF(\mathrm{i})$. S_0 es particionado en S_0^{1A}, S_0^{1G}, S_0^{2A} y S_0^{2G} donde el superíndice indica la hebra y codón de inicio de los candidatos. Sobre cada elemento de la partición, se entrena y prueba un árbol de clasificación. Los candidatos predichos positivamente para toda la partición definen la región $\mathcal{P}_0$ sobre la molécula de ADN y su complemento $\mathcal{N}_0$ sobre esta.
2. En la región $\mathcal{N}_0$ se buscan todos los candidatos de $RF(\mathrm{i})_0$ que estén totalmente incluidos en ella. Esta intersección se denota como $RF(\mathrm{i})_1$.
3. Sea S_1 la solución al algoritmo de optimización usando los marcos de lectura en $RF(\mathrm{i})_1$. Se repite el paso 1. nuevamente.

La metodología anterior produce sucesiones de regiones en el ADN, $\mathcal{P}_0, \mathcal{P}_1, \mathcal{P}_2, ...$ y $\mathcal{N}_0, \mathcal{N}_1, \mathcal{N}_2, ...$ tales que $\forall k \geq 0$ se tienen las relaciones $\mathcal{P}_{k+1} \subseteq \mathcal{N}_k$, $\mathcal{P}_k \cap \mathcal{P}_{k+1} = \emptyset$, $\mathcal{P}_k \cap \mathcal{N}_k = \emptyset$ y $\mathcal{N}_{k+1} \subseteq \mathcal{N}_k$. En otras palabras, las regiones definidas por las predicciones positivas en el paso $k+1$ no intersectan con las definidas en el paso k asegurando que las predicciones positivas que se vayan haciendo no hayan sido vistas previamente. Cabe agregar que no es estrictamente necesario usar solo una submuestra de largo mínimo i de RF ($RF(\mathrm{i})$) sino que se pueden combinar varias de ellas hasta incluso el punto de usar una diferente por cada iteración.

Es importante notar que por cada iteración crecen 4 árboles - uno por cada hebra y codón de inicio - los cuales podrían ser bastantes diferentes a los observados en iteraciones anteriores. Esto es por cómo se va particionando la hebra de ADN a medida que se avanza en el algoritmo: los árboles de iteraciones mayores que actúan, por ejemplo, en función de

la solución post optimización S_k no disponen de las observaciones situadas en las regiones P_l con $l < k$. Esto se traduce en que tienen una menor cantidad de observaciones verdaderas positivas ("si") de las cuales aprender a reconocer información asociada a genes anotados.

Para simular el método anterior se cambia la forma en que se efectúa el entrenamiento y prueba del modelo. En las estrategias anteriores se particionaba el espacio de características en conjuntos de entrenamiento y prueba para ajustar y medir el modelo respectivamente. Para evitar la varianza en los resultados provenientes de la elección de una partición en particular, la división entrenamiento/prueba se realizaba diez veces de manera aleatoria. El rendimiento final se calculaba como el promedio de los rendimientos sobre cada uno de los conjuntos de prueba tras ajustar los árboles en los conjuntos de entrenamientos respectivos. Se vuelve a mencionar que cada árbol es podado para evitar el sobreajuste de los modelos sobre los conjuntos de entrenamiento, con el ánimo de lograr una mayor generalidad en datos no vistos en dicha etapa. En la estrategia actual se pretende simular el método de optimización y clasificación anidadas. En su naturaleza este método produce una gran cantidad de árboles que en su conjunto son difíciles de rastrear ya que, por ejemplo, con 10 iteraciones se generan 40 de ellos. Si se imitara la forma de entrenamiento/prueba ya descrita, el número de observaciones en la clase positiva "si" se reduciría importantemente en unas primeras iteraciones del método. La razón es que al avanzar por él cada vez son menos los genes anotados disponibles y si además se restringe el entrenamiento a un 70 % de los datos totales, aún menor será la cantidad de anotados. Por tal motivo, en lugar del par entrenamiento/prueba se realiza una validación cruzada. En ausencia de suficientes datos para realizar simulaciones, el error de clasificación proveniente de un proceso de validación cruzada suele usarse como estimador para medir el desempeño de un modelo, en reemplazo del error medido en un conjunto de prueba aunque tiende a ser optimista. Se realiza una validación cruzada de 10 partes sobre cada árbol de decisión construido en el método de optimización y clasificación anidados.

Elección de parámetros: Los parámetros utilizados se fijan con un único valor durante toda la simulación del método anidado ya que efectuar una búsqueda en grilla sobre cada árbol entrenado tiene un costo computacional muy alto. Se utiliza como tupla de parámetros (cp=0.01, minsplit=30, minbucket=10) la que fue elegida por presentar el mejor desempeño frente a otros parámetros evitando caer en el sobreajuste de datos.

Simulación 1

Se simula el método anidado por cada base del tipo "Muestra" usando los candidatos en $RF(100)$, es decir, aquellos con un largo mayor a 100 pb. La simulación se lleva a cabo por cada base del tipo "Muestra" cuyos resultados se resumen en las siguientes tablas. Por iteración se incluye información como el número observaciones (N), coeficientes de la matriz de confusión (TP, FP, TN y FN) y medidas de rendimiento (TPR y PPV) tras cada validación cruzada.

Iteración	N	TP	FP	FN	TN	TPR	PPV
1	2114	578	89	453	1000	56.1	86.7
2	1453	86	49	367	945	19.0	63.7
3	1312	62	71	305	874	16.9	46.6
4	1179	64	54	241	820	21.0	54.2
5	1061	47	63	194	757	19.5	42.7
6	951	43	68	151	689	22.2	38.7
7	840	20	58	131	631	13.2	25.6
8	762	11	30	120	601	8.4	26.8
9	721	31	49	89	552	25.8	38.8
10	641	8	23	81	529	9.0	25.8

Tabla 4.9: Resultados Simulación 1 para Muestra1a. Se usaron 10 simulaciones.

Iteración	N	TP	FP	FN	TN	TPR	PPV
1	889	38	27	60	764	38.8	58.5
2	824	3	11	57	753	5.0	21.4
3	810	4	11	53	742	7.0	26.7
4	795	3	9	50	733	5.7	25.0
5	783	4	10	46	723	8.0	28.6
6	769	2	9	44	714	4.3	18.2
7	758	3	10	41	704	6.8	23.1
8	745	2	3	39	701	4.9	40.0
9	740	0	5	39	696	0	0
10	735	2	5	37	691	5.1	28.6

Tabla 4.10: Resultados Simulación 1 para Muestra1g. Se usaron 10 simulaciones.

Iteración	N	TP	FP	FN	TN	TPR	PPV
1	2274	614	63	511	1086	54.6	90.7
2	1597	104	39	407	1047	20.4	72.7
3	1454	75	44	332	1003	18.4	63.0
4	1335	90	37	242	1000	27.1	70.9
5	1242	23	45	219	921	9.5	33.8
6	1140	14	46	205	875	6.4	23.3
7	1080	25	52	180	823	12.2	32.5
8	1003	16	30	164	793	8.9	34.8
9	957	12	32	152	761	7.3	27.3
10	913	15	32	137	729	9.9	31.9

Tabla 4.11: Resultados Simulación 1 para Muestra2a. Se usaron 10 simulaciones.

Iteración	N	TP	FP	FN	TN	TPR	PPV
1	870	32	23	68	747	32.0	58.2
2	815	13	26	55	721	19.1	33.3
3	776	4	9	51	712	7.3	30.8
4	763	3	8	48	704	5.9	27.3
5	752	4	10	44	694	8.3	28.6
6	738	2	5	42	689	4.5	28.6
7	731	2	4	40	685	4.8	33.3
8	725	0	1	40	684	0	0
9	724	0	2	40	682	0	0
10	722	0	0	40	682	0	NA

Tabla 4.12: Resultados Simulación 1 para Muestra2g. Se usaron 10 simulaciones.

Se observa que el número de genes anotados totales encontrados en las cuatro bases "Muestra" (columnas TPR) asciende a un total de 2059, de los cuales 950, 61, 988 y 60 están presentes en "Muestra1a", "Muestra1g", "Muestra2a" y "Muestra2g" respectivamente. La ganancia obtenida en las bases con comienzo *ATG* representa una mejora sustantiva respecto a las estrategias anteriores aunque el desempeño en aquellos de inicio *GTG* no aumenta demasiado. En total se encontró un 68 % de los genes presentes en RF(100) y un 87 % de aquellos reportados tras la aplicación del algoritmo de optimización en la primera iteración. Es importante señalar que al sumar los verdaderos positivos (TP) con los falsos negativos (FN) de una iteración se obtienen los falsos negativos de la iteración anterior. Esto quiere decir que a medida que se avanza en el método, los nuevos casos clasificados como genes verdaderos vienen directamente desde los errores cometidos por los árboles de pasos más atrás. En términos más generales solo se encontró el 50 % del total de genes anotados comenzando con *ATG* o *GTG* lo que es bajo si comparamos los resultados con aquellos provenientes desde el algoritmo GLIMMER (ver[12]), los cuales alcanzan valores sobre el 90 % de genes encontrados.

Simulación 2

En esta simulación se procede de la misma forma que la anterior pero combinando dos bases de marcos de lectura con distinto largo mínimo y usando candidatos degenerados y no. La idea tras esto es aumentar la sensibilidad de las soluciones post optimización tal como se observó en el capítulo de análisis de datos de forma que los árboles puedan identificar más genes. Se realiza una simulación por cada par de bases candidatos elegidos, usando 10 iteraciones y los resultados son mostrados en términos de candidatos totales encontrados entre todas las bases "Muestra", es decir, sumando las columnas TP de las cuatro tablas mostradas en la Simulación 1. La forma en que se combinan las bases puede diferir en los resultados de forma que usar primero RF(100) y luego RFD(100) puede variar a usarlas en orden inverso. Los resultados son los siguientes.

Se observa que a diferentes combinaciones de candidatos diferentes son los resultados obtenidos. El peor resultado, e incluso peor que el encontrado en Simulación 1, se obtiene desde

Base 1	Base 2	Genes Encontrados
RF(100)	RFD(100)	1761
RFD(100)	RF(100)	1975
RF(200)	RF(200)	2207
RF(200)	RFD(200)	2210
RFD(200)	RF(200)	2269
RFD(200)	RFD(200)	1906
RFD(200)	RF(100)	2292
RFD(200)	RF(300)	2326
RFD(200)	RF(500)	2233
RFD(100)	RF(500)	2206
RFD(500)	RF(200)	2287
RF(200)	RFD(300)	2321

Tabla 4.13: Resultados Simulación 2. Se muestra el número total de genes encontrados luego de 10 iteraciones por cada combinación de bases de marcos de lectura del tipo RF(i)-RF(j), RFD(i)-RF(j), RFD(i)-RFD(j) o RF(i)-RFD(j).

la combinación de RF(100) y RFD(100) con 1761 anotados en dicho orden. Esto se obtiene luego de usar primero RF(100) en la primera iteración y RFD(100) a partir de la segunda hasta la décima. Para las demás simulaciones se obtienen mejoras aunque no se logra superar la barrera de los 2321 genes anotados en el par RF(200)-RFD(300). Este caso reporta una identificación del 56,4 % de los genes totales anotados con inicio ATG y GTG. La potencia de los resultados es muy dependiente de los genes anotados que se obtienen desde las soluciones a los sucesivos algoritmos de optimización usados y de las bases de candidatos con largo mínimo desde las que se extraen las submuestras de dichas soluciones. Por tanto, la cantidad final de anotados está acotada superiormente por los anotados encontrados en dichos pasos.

Conclusión

A lo largo de este trabajo se han desarrollado una serie de estrategias que pretenden ayudar en la tarea de identificar genes a lo largo de una molécula de ADN en la bacteria *Escherichia Coli*. La idea de usar árboles de decisión permite entender de mejor manera cuales son las decisiones importantes en dicho proceso ya que muestran cuales son las regiones de mayor importancia biológica que ayudan a prever el comienzo de los genes. El uso del contenido de información (IC) en estas regiones, previas a los genes, indica que las primeras posiciones en torno al décimo nucleótido son las más importantes y que se condicen con aquellas reportadas por los árboles de decisión en su etapa de crecimiento y ajuste a los datos. Misma conclusión se verifica también con el uso de secuencias de consenso en torno a dichas regiones. Los nucleótidos que ocupan las posiciones ocho, nueve, diez, once y doce en dichas regiones son las que determinan la construcción de los árboles y de acuerdo a la literatura corresponden también a laz zonas identificadas como promotoras de genes.

En las primeras simulaciones se ha visto que el poder predictivo de los árboles está sujeto al balance de clases entre anotados y candidatos espurios, por lo que el uso del algoritmo de optimización sobre la distribución de candidatos se vuelve fundamental a la hora de obtener una submuestra más equilibrada. La sensibilidad encontrada ronda el 55 % sobre candidatos que comienzan con codón de inicio ATG mientras que para aquellos con inicio GTG esta disminuye en torno al 20 %. Por otro lado, la precisión de la clasificación ronda el 85 % entre los genes ATG, lo que constituye un elemento positivo pues asegura que los errores cometidos entre los predichos como genes son pocos.

Las siguientes estrategias implementadas intentan llevar la clasificación un paso más allá. El uso anidado entre optimización y árboles aprovecha los errores cometidos en la clasificación anterior para obtener nuevos genes anotados ocultos en estas regiones definidas por las predicciones sobre la clase mayoritaria o negativa. Se usaron, además, candidatos de diferentes largos mínimos para aprovechar el hecho que es más probable encontrar un gen de gran largo. Los resultados obtenidos arrojan un rango entre los 1700 y 2300 genes encontrados aproximadamente, lo que representa un éxito entre un 41 % y 57 % en la identificación de todos los anotados con inicio ATG y GTG.

Los resultados dependen mucho de la cantidad de genes anotados encontrados en la etapa de construcción de los marcos de lectura. No se logra superar esta cota pues los algoritmos obtienen submuestras desde dichos candidatos. La extracción de candidatos con codón de inicio degenerado permite conseguir casi la totalidad de anotados en desmedro de una menor sensibilidad de las bases para reconocerlos. Por lo anterior, se usaron combinaciones entre

estos candidatos y los extraidos de manera secuencial aunque el rendimiento no cambia cuantiosamente.

Este trabajo podría mejorarse entendiendo de mejor manera como recuperar los candidatos en la construcción inicial de los marcos de lectura. Además, muchos de los candidatos degenerados contenían genes anotados que probablemente no eran extraidos en las etapas de optimización y que, por tanto, no eran vistos por los árboles de clasificación. Dicha estrategia se podría mejorar cambiando la etiqueta usada durante la clasificación. En lugar de ser tan drástica en categorizar un candidato como gen o no gen, se podría preguntar si los candidatos contiene zonas génicas. De este modo, los candidatos degenerados que contienen genes si participarían positivamente en la etapa de clasificación.

Bibliografía

[1] Leo Breiman. *Classification and regression trees*. Routledge, 1984.

[2] Shawn T Estrem, Wilma Ross, Tamas Gaal, ZW Susan Chen, Wei Niu, Richard H Ebright, and Richard L Gourse. Bacterial promoter architecture: subsite structure of up elements and interactions with the carboxy-terminal domain of the rna polymerase α subunit. *Genes & development*, 13(16):2134–2147, 1999.

[3] National Center for Biotechnology Information. Pubchem compound database, 2017, Abril 22.

[4] AJF Griffiths, WM Gelbart, JH Miller, and RC Lewontin. *Genética moderna (Modern Genetic Analysis)*. McGraw-Hill (Interamericana de EspanÄa), Madrid, 2000.

[5] Calvin B Harley and Robert P Reynolds. Analysis of e. coli pormoter sequences. *Nucleic acids research*, 15(5):2343–2361, 1987.

[6] Andrew G. Hart, Servet Martínez, and Leonardo Videla. A simple maximization model inspired by algorithms for the organization of genetic candidates in bacterial dna. *Advances in applied probability*, 38(4):1071–1097, 2006.

[7] Servet Martínez and Andrew G. Hart. A theorical view on some problems of decision trees: a work in progress. 2017.

[8] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

[9] Pierre Nicolas. *Mise au point et utilisation de modèles de chaînes de Markov cachées pour l'étude des séquences d'ADN*. PhD thesis, Evry-Val d'Essonne, 2003.

[10] Pierre Nicolas and Florence Muri-Majoube. R'hom–programs to segment dna sequences into homogeneous regions. Technical report, Tech. Rep., Université d'Evry. Available at http://genome. jouy. inra. fr/ssb/rhom/rhom_doc/rhom_doc. html, 2001.

[11] Wilma Ross, Khoosheh K Gosink, Julia Salomon, Kazuhiko Igarashi, Chao Zou, Akira Ishihama, Konstantin Severinov, and Richard L Gourse. A third recognition element in bacterial promoters: Dna binding by the alpha subunit of rna polymerase. *Science*, 262(5138):1407–1413, 1993.

[12] Steven L Salzberg, Arthur L Delcher, Simon Kasif, and Owen White. Microbial gene identification using interpolated markov models. *Nucleic acids research*, 26(2):544–548, 1998.

[13] Thomas D Schneider. Consensus sequence zen. *Applied bioinformatics*, 1(3):111, 2002.

[14] Thomas D Schneider and R Michael Stephens. Sequence logos: a new way to display consensus sequences. *Nucleic acids research*, 18(20):6097–6100, 1990.

[15] Carr S.M. Deoxyribose versus ribose sugars, 2014.

[16] T Smith and M Waterman. Identification of common molecular subsequences. *Molecular Biology*, 147:195–197, 1981.