Universidad del Bío-Bío, Chile

Facultad de Ciencias Empresariales

Departamento de Sistemas de Información

Identificación de factores discriminantes basados en distancia máxima y mínima entre átomos de aminoácidos de proteínas pertenecientes al mismo complejo proteico.

Tesis presentada por Rodrigo Bustos Flores
para obtener el grado de Ingeniero Civil Informático
Dirigida por Tatiana Gutiérrez-Bunster, Claudio Gutiérrez

2018

Resumen

El análisis del comportamiento de las proteínas es muy importante para los seres vivos, predecir su comportamiento puede ayudar a fabricar mejores vacunas o prevenir enfermedades. Para complementar los estudios actuales acerca de las proteínas, este documento se basa en encontrar nueva información a partir del cálculo de la distancia entre las proteínas de un mismo complejo, para ello se utilizarán bases de datos de proteínas, se confeccionará un software que permita obtener los datos de esas bases de datos, y finalmente confeccionar un archivo compatible con un software de análisis estadístico, para descubrir si existe alguna relación importante entre dichas distancias y características de sus interacciones.

Palabras Clave — Proteínas, Aninoácidos, Complejo proteico, data mining, weka, fast-contact, algoritmos, python.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introduccion

1.1. introducción

En nuestro entorno existen muchos procesos de distinto tipo, unos en cadena, otros independientes, los vemos a diario y estamos tan acostumbrados a ellos que muchas veces nos olvidamos que están ahí, como el funcionamiento de una ampolleta, que recibe electricidad y mediante cierto proceso, esta electricidad se transforma principalmente en luz y calor, un vendedor en una cafetería que transforma sus insumos en desayunos, un vehículo transportando personas, estos son ejemplos de procesos o parte de procesos a simple vista, quizás puedan considerarse malos ejemplos pero lo que debemos comprender es que hay interacción entre distintas actividades, pero ¿Qué pasa con aquellos procesos que no vemos?, bueno, estos procesos ocurren en todas las escalas, algunos gigantes como lo que ocurre en el cosmos y otros tan pequeños como lo que ocurre en los átomos.

Para esta investigación, hay que acercarse exageradamente a lo que podemos ver a simple vista, al punto atómico. A esta escala se podrá encontrar estructuras orgánicas y no orgánicas, pero nos enfocaremos en lo que pasa con las proteínas, que son las estructuras orgánicas mas abundantes en los seres vivos, (adelantando un poco en la investigación, quizás sea una característica de comparación, la diferencia entre la proteína animal y la vegetal, con los resultados obtenidos).

1.2. Objetivos

Este estudio tiene como objetivo general, analizar las distancias entre los átomos de aminoácidos de las proteínas en la zona de interacción, para ver si su tipo de interacción (permanente o transitoria) o energía producida, depende de la distancia entre los átomos de la zona de interacción.

Para este estudio se deben realizar los siguientes pasos:

1. Estudiar contenidos de biología, para comprender que son las proteínas y su importancia en los seres vivos.

2. Estudiar algoritmos y métodos de medición de distancias masivas en espacios tridimensionales de complejos proteicos (transitorios y permanentes).
3. Confeccionar un software en lenguaje Python que genere las distancias correspondientes junto a la información relevante para su posterior análisis.
4. Generar matrices de características con los resultados obtenidos, para ingresarlas en un software de aprendizaje automático llamado WEKA, que nos permitirá abordar patrones de comportamiento de los complejos proteicos ya clasificados.
5. Analizar los resultados obtenidos del software WEKA mediante el uso de métodos estadísticos.

1.3. Organización

- Capítulo 2: Estado del arte. Éste capítulo contiene información acerca de la biología referente a las proteínas, algo más profundo de lo mencionado en la introducción, observaciones propias entre el mundo a escala celular y el mundo a escala humana para una mejor comprensión del estudio.
- Capítulo 3: Estudio del problema. Aquí explicaremos el método escogido para calcular estas distancias, resultados que se obtienen de los complejos proteicos, datos importantes que debemos considerar de los complejos proteicos.
- Capítulo 4: Implementación de algoritmos. Implementación del cálculo de distancias entre los átomos y confección de matriz resumen para el análisis de características.
- Capítulo 5: Análisis de datos. Mediante el software WEKA para ingresar los datos de la matriz mencionada en el capítulo anterior y obtener las características de las distancias, posteriormente analizaremos estadísticamente la relación entre distancias entre átomos y el tipo de interacción entre proteínas, tipo de energía producida, o cualquier conclusión importante que se pueda apreciar.
- Capítulo 6: Conclusiones. Después de realizar el estudio, evaluaremos si lo que pensamos en la introducción realmente tiene alguna relación con los datos obtenidos. También se indicarán posibles trabajos futuros.

Capítulo 2

Estado del Arte

2.1. Conocimientos Biológicos necesarios

Antes de abarcar actividades específicas, se definirán ciertos términos biológicos necesarios, desde este punto de vista, se reunió información de distintas fuentes para entregar una visión más completa, especial para quienes no saben mucho de biología. Por conocimiento general casi todos han escuchado del átomo, pero quizás no sepan qué es realmente, bueno, el átomo es la estructura más pequeña de la materia, están compuestos por protones, neutrones y electrones, visualmente son como una esfera con un centro, entre centro contiene protones y neutrones y exteriormente electrones, se clasifican en distintos tipos de átomos dependiendo la cantidad de cada uno de estos elementos, acá esta disponible este video muy explicativo de un artista clásico [11].

2.1.1. Aminoácido

Definidos los átomos, es necesario conocer que estos átomos se relacionan con otros átomos para formar estructuras más grandes [21], pero antes de hablar de esas estructuras, hay que pensar en la unión de un átomo con otro.

Existe un grupo de átomos de ciertas características que se unen, debido a sus propiedades que facilitan compartir electrones entre ellos (quizas podría ser objeto de una investigación específica en opinión personal.), formando un grupo amino y un grupo ácido que corresponde a un carboxilo, ambos subgrupos de átomos, son la causa de la formación de un aminoácido, éstos están unidos a un mismo carbono de la estructura, que es denominado Carbono Alfa, el carbono alfa forma enlaces covalentes (octeto estable) además con un átomo de hidrógeno y con un grupo R, este grupo R varía entre los distintos aminoácidos y es el que le da la identidad y propiedades características a cada uno de ellos.

En la figura 2.1 podemos ver una estructura tridimensional de Aminoácido Cisteína (Cys, C) con identificación de sus grupos amino, carboxilo y R (cadena lateral). Éste aminoácido es uno de los más simples estructuralmente. Los aminoácidos son clasificados por su grupo R en polares y no polares, excepto uno, todos los aminoácidos tienen dos grupos cargados el amino y el carboxilo,

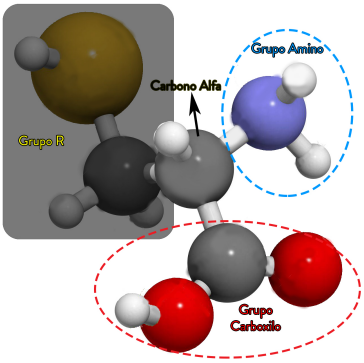

Figura 2.1: Imagen tridimensional de Aminoácido Cisteína (Cys, C).

estos son usados para formar los enlaces peptídicos de las proteínas y como componentes de ellas, es el grupo R el que determina si un residuo de aminoácido es polar o no [9].

Cuando los aminoácidos se unen entre sí forman enlaces péptidos (unión de varios aminoácidos) o polipéptidos (decenas o cientos de aminoácidos). Esta unión es lineal y al superar cierta longitud, o su masa molecular supera cierto valor, y cuando su estructura tridimensional es estable, se les llama proteínas.

2.1.2. Enlace peptídico

La unión de dos o más aminoácidos mediante enlaces amida origina los enlaces péptidos. En los péptidos y en las proteínas, estos enlaces amida reciben el nombre de enlaces peptídicos y son el resultado de la reacción del grupo carboxilo de un aminoácido con el grupo amino de otro, con eliminación de una molécula de agua.

En la figura 2.2 se puede ver el proceso de unión de aminoácidos, formación de enlace peptídico, que une de manera covalente el carbono carbonílico del primer aminoácido con el nitrógeno amínico del segundo (de mayor tamaño para mejor comprensión) y liberación de molécula de agua, acá se puede encontrar una animación paso a paso esquemáticamente con el detalle de éste proceso [24].

El enlace peptídico (-CO-NH-) se representa normalmente como un enlace sencillo. Sin embargo, posee una serie de características que lo aproximan más a un doble enlace. Como el nitrógeno es menos electronegativo que el oxígeno, el enlace C-O tiene un 60 % de carácter de doble enlace mientras que el enlace C-N tiene un 40 %. Por tanto, los enlaces C-O y N-C del enlace peptídico tienen características intermedias entre el enlace sencillo y el enlace doble. De hecho, las distancias interatómicas medidas en los enlaces C-O y C-N son intermedias entre las del enlace sencillo y el doble enlace [13].

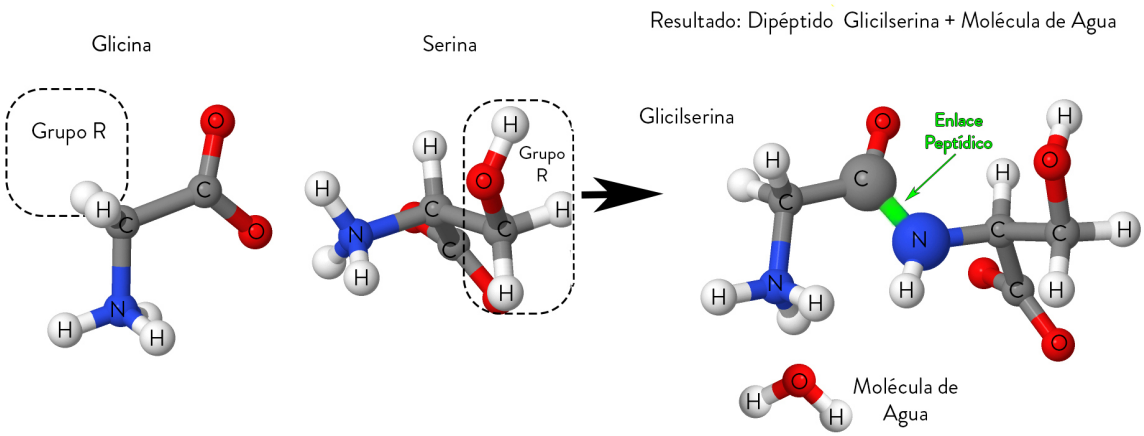

Figura 2.2: Proceso de unión de aminoácidos.

2.1.3. Péptido

Es una estructura de varios aminoácidos que se unen a través del enlace peptídico. Estas estructuras péptidas se clasifican según la cantidad de aminoácidos que lo forman, pocos aminoácidos forman un péptido, muchos forman una proteína, pero no hay una cantidad exacta para clasificarlos[13]. Lo habitual es:

- Oligopéptido: de 2 a 10 aminoácidos.
- Polipéptido: entre 10 y 100 aminoácidos[34].
- Proteína: más de 100 aminoácidos. Las proteínas con una sola cadena polipeptídica se denominan proteínas monoméricas, mientras que las compuestas de más de una cadena polipeptídica se conocen como proteínas multiméricas [35].

2.1.4. Proteínas

Se explicó acerca de cada una de las estructuras de la materia relevantes en este estudio, desde la más pequeña, tenemos que ir quitando el acercamiento poco a poco para visualizar las formaciones de conjuntos de muchos de los elementos mencionados. Resumiendo, el componente más pequeño de una proteína es un aminoácido, éste se une a otros mediante un enlace peptídico, y la agrupación de cierta cantidad de estos elementos se pueden clasificar en oligopéptido, polipéptido y proteína tal como se explicó en 2.1.3.

Las proteínas forman distintas estructuras, éstas se clasifican [28] en los siguientes tipos:

- **Estructura primaria:** Los carbonos alfa de cada aminoácido se van alternando con los enlaces peptídicos para formar el esqueleto del péptido. La unión similar de un gran número de aminoácidos forma polipéptidos, que se llaman proteínas cuando son suficientemente grandes y tienen una estructura tridimensional definida. Aquí vemos el acercamiento de una estructura primaria 2.3, estas estructuras son la composición de las que mencionamos anteriormente en 2.2. En la figura es importante comprender como se va formando el patrón

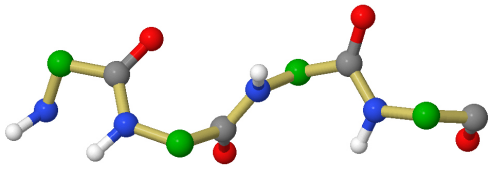

Figura 2.3: Estructura proteína primaria a: Modelo de un tetrapéptido.

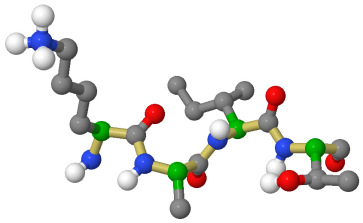

Figura 2.4: Estructura proteína primaria b: Tal como la figura anterior, modelo de un tetrapéptido., pero con más elementos.

entre los enlaces péptidicos, ya que a medida que se mostrarán figuras mas grandes, solo se considerarán estas uniones, ya que el resto solo aporta información menos importante, que puede llevar a la confusión, esto esta reflejado al comparar la figura anterior con esta 2.4.

En este modelo de un tetrapéptido sólo está representado lo que llamamos el esqueleto peptídico: los carbonos alfa(resaltados en verde), los átomos que intervienen en los enlaces peptídicos (—CO—NH—) y los grupos amino y carboxilo terminales. No se representan ni las cadenas laterales ni el resto de hidrógenos, y del carboxilo terminal sólo se representa uno de los oxígenos. En amarillo se resaltan los enlaces peptídicos para mejor comprensión de la formación conjunta.
Finalmente en la figura 2.4 se muestra una estructura primaria con mas elementos.

- **Estructura secundaria:** Es la organización regular y periódica en el espacio de las cadenas polipeptídicas en una dirección.El plegamiento característico de este tipo de organización está determinado por la secuencia de aminoácidos y la rigidez del enlace peptídico, que sólo posibilita giros en torno a los enlaces sencillos. La estabilidad de esta estructura es posible gracias a los puentes de hidrógeno que se establecen entre los grupos amino y carboxilo.
  - **Estructura helicoidal o hélice alfa:** Es la estructura secundaria más común. En ella, la cadena polipeptídica se va enrollando en espiral sobre sí misma debido a los giros que se producen en torno al carbono Alfa de cada aminoácido como veremos en la figura 2.5 que esta representada mediante átomos y enlaces (es común encontrar representaciones de cintas, flechas y otras más, que sirven para entender mejor sus patrones, pero se escogieron bolas y varas para no complicar al lector al variar entre un esquema y otro).

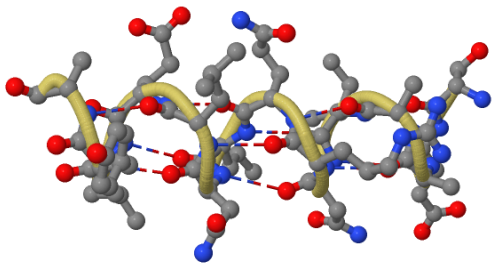

Figura 2.5: Estructura helicoidal: Esqueleto peptídico.

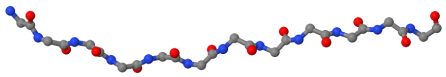

Figura 2.6: Hoja Beta.

En este 2.5 tipo de estructura secundaria el esqueleto peptídico se enrolla sobre sí mismo describiendo un helicoide compacto alrededor del eje longitudinal de la molécula. La cuerda amarilla en la figura nos sirve para ver mejor la hélice formada, no es parte de la estructura pero sigue ese patrón. Cada residuo se desplaza 0,15 nm a lo largo del eje con respecto al residuo anterior, y cada vuelta completa de la hélice supone una elevación de 0,54 nm (paso de hélice). La hélice está estabilizada por (—-) entre los nitrógenos y los grupos carbonilo de los enlaces peptídicos.

- **Estructura hoja beta:** el esqueleto polipeptídico se encuentra extendido, en lugar de retorcido sobre sí mismo en forma de hélice. Observa su disposición en zig-zag. ver en 2.6.
- **Estructura de hoja plegada:** Generalmente están junto a las hpelices alfa. Es común encontrar hojas betas en varios segmentos de la cadena polipeptídica, que se alinean paralelamente, formando así una hoja plegada u hoja beta. Las hojas beta se estabilizan mediante puentes de hidrógeno (—-) entre N y O pertenecientes a los enlaces peptídicos, como en la hélice alfa, pero en este caso son enlaces intercatenarios, entre las cadenas polipeptídicas adyacentes 2.7.

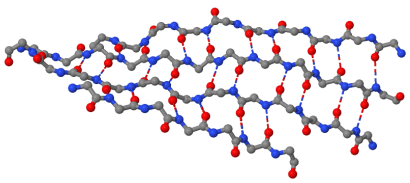

Figura 2.7: Hoja Beta plegada.

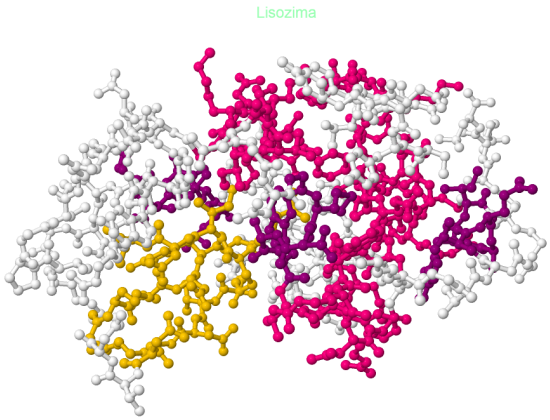

Figura 2.8: Proteína Lisozima.

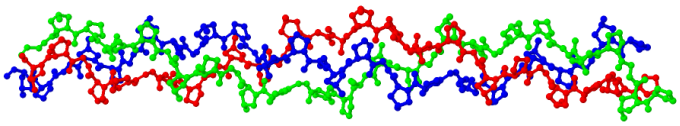

Figura 2.9: Estructura cuaternaria: Colágeno

- **Estructura Terciaria:** La estructura terciaria de una proteína (su estructura tridimensional completa) suele estar formada por varios tramos con estructuras secundarias diferentes. Así ocurre, por ejemplo, con la lisozima, una proteína pequeña (129 residuos aminoácidos) En el ejemplo existen distintos colores que representan a cada estructura secundaria 2.8. Tal como se aprecia en los colores, cada uno corresponde a una proteínas nonomérica, por lo que la Lisozima esta compuesta por distintas cadenas de polipétidos. Hélice alfa (Magenta), hélices mas pequeñas(Lila), hoja beta (Amarillo), Zonas sin estructura secundaria regular (Blanco).
- **Estructura Cuaternaria:** Cuando una proteína consta de más de una cadena polipeptídica, es decir, cuando se trata de una proteína oligomérica, decimos que tiene estructura cuaternaria.2.9. En la figura encontramos al colágeno, el colágeno es una proteína fibrosa, formada por la asociación de varias cadenas polipeptídicas; por eso es un ejemplo de estructura cuaternaria. La unidad básica de una fibra de colágeno es la molécula de tropocolágeno, una hélice triple de tres cadenas polipeptídicas iguales (A, B y C en el modelo), cada una de ellas con aproximadamente 1000 residuos (aquí se muestran tan solo 30) y la estructura secundaria característica, la hélice del colágeno.

Reiterando, las figuras mostradas están representadas a través del mismo tipo de diagramas, bolas y varillas, esto se realizó para que la información sea fácilmente comprensible por personas

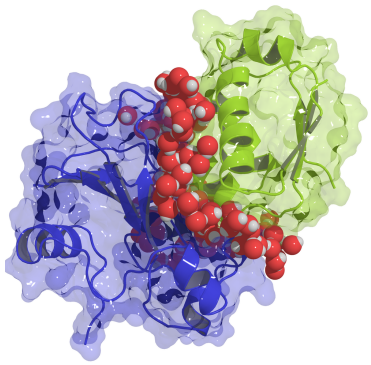

Figura 2.10: Estructura compuesta de hélices, hojas, átomos principalmente.

que desconocen acerca de la biología, existen otras representaciones con cintas, lineas, varillas, esferas y muchos mas, su uso esta dado porque se puede representar mejor las hélices o planos que forman estas estructuras, no se mezclaron estos conceptos pero es de gran utilidad comprender los patrones básicos de las estructuras mostradas.

Los conceptos anteriores son fundamentales para los que siguen, a continuación se explicarán conceptos más cercanos a nuestra área. Respecto a las representaciones gráficas de los elementos mencionados, podemos profundizar el tema en estas fuentes [14],[5],[17]. Un ejemplo con todos los conceptos anteriores es 2.10.

Químicamente, las proteínas contienen los mismos átomos que los hidratos de carbono o las grasas, como carbono, hidrógeno y oxígeno. Pero, a diferencia de éstas, las proteínas se caracterizan porque contienen nitrógeno y algunas veces azufre (aminoácidos azufrados como la metionina).

Las proteínas forman parte de la estructura básica de los tejidos, desempeñan funciones metabólicas y reguladoras, definen la identidad de cada ser vivo, por ser la base de la estructura del código genético, para entender más éste punto, pensemos en las personas, todas las personas tienen un "manual de instrucciones"de cómo hacer sus propias proteínas, llamado código genético o ADN[1]. El ADN de cada persona se hereda a partir de la unión de las células sexuales de sus padres (ADN espermatozoide y ADN del óvulo). Las células se reproducen una y otra vez hasta formar una persona con millones de células, las cuales contienen en su núcleo ese ADN.Por ejemplo, las células de los ojos reproducen la información del ADN que les indica cómo fabricar una proteína para ojos azules. Otra persona, tendrá la información para ojos verdes. Para poder fabricar proteínas, el cuerpo necesita, además de la información del ADN, un sustrato o "ladrillos": los aminoácidos, que puede obtener de los alimentos, o fabricar él mismo (el cuerpo sólo puede fabricar los aminoácidos no-esenciales) a partir de sustancias.

Aunque todas las células tienen el mismo ADN, éste no se lee de igual forma: Cada célula del organismo, según sus funciones, produce un tipo u otro de proteínas. Por ejemplo, sólo las

[1] ácido desoxirribonucleico la cual contiene las instrucciones biológicas que hacen de cada especie algo único

células del páncreas producen insulina, que es una proteína con efecto hormonal. Ningún otro órgano produce insulina. Cada célula del organismo produce sus propias proteínas, a partir de la información que contiene su código genético (ADN). Estas proteínas son las que formarán sus orgánulos celulares, sus enzimas, la pared de la célula, hormonas, etc.

Todas las proteínas del cuerpo (colágeno, insulina, enzimas, etc.) se originan a partir del ADN del núcleo de una célula.

El ADN es solo una parte donde existen las proteínas, existen muchos más ejemplos, pero es uno de los ejemplos más destacados acerca del papel que desempeñan las proteínas. Para entender un poco más acerca de la actividad entre proteínas, es esencial conocer lo que sucede al momento que se unen entre ellas, la interacción que ocurre en ese instante, ya que esta actividad es la que determina las características de las estructuras más grandes, procesos enzimáticos y regulación y trasducción de señales. Estas interacciones pueden clasificarse en transitorias (duran poco tiempo) y permanentes (permanecen por mucho tiempo). La comprensión de estas interacciones , otorga información útil acerca del mecanismo molecular de funcionamiento de la célula, ayudando así en el diseño de fármacos más específicos, como también en la ingeniería de procesos celulares.

Existen distintos métodos para determinar el tipo de interacción de proteínas, entre los principales estan:

Coinmunoprecipitación: es considerado el mejor ensayo para detectar las interacciones proteína-proteína, especialmente cuando se realiza con proteínas endógenas (ni sobreexpresadas ni marcadas). La proteína de interés se aisla con un anticuerpo específico. Las moléculas que interaccionan con la proteína se identifican posteriormente mediante un Western blot.

Ensayo de doble híbrido en levadura: investiga la interacción entre proteínas de fusión artificiales en el interior del núcleo de una levadura. Este método puede identificar moléculas que se unen a una proteína dada de forma inequívoca. No obstante, este método tiene una considerable tasa de falsos-positivos, lo que hace necesario verificar las interacciones identificadas mediante un ensayo de coinmunoprecipitación.

Tandem affinity purification (TAP): detecta interacciones en el ambiente celular real (ej. en el citosol de una célula de mamífero)(Rigaut et al., 1999). Esto supone una gran ventaja comparado con el ensayo de doble híbrido. Una desventaja importante es que la purificación de proteínas se realiza mediante dos lavados, de forma que no puede detectar interacciones proteína-proteína transitorias.

Inmunoprecipitación cuantitativa combinada con knok-out (QUICK - Quantitative Inmunoprecipitation Combined with Knock-out): se basa en la coinmunoprecipitación, la espectrometría de masa cuantitativa (SILAC) y la interferencia de ARN (RNA interference - RNAi). Este método detecta interacción entre proteínas endógenas sin marcar (Selbach and Mann, 2006), de modo que tiene la misma fiabilidad que la coinmunoprecipitación, aunque depende también de la disponibilidad de anticuerpos adecuados.

Dual Polarisation Interferometry (DPI): puede usarse para medir interacciones proteína-proteína. DPI proporciona medidas de tamaño molecular, densidad y masa, en tiempo real y con alta resolución.

Geles Nativos Blue Native (BN-PAGE): esta metodología se basa en la migración de los complejos proteicos en geles de poliacrilamida según su peso molecular. Dado que la migración también está definida por la carga, se utiliza como buffer catódico una solución que contiene azul de coomassie, el cual otorga carga negativa neta a las proteínas sin desnaturar ni romper sus interacciones con otras proteínas. Una segunda dimensión desnaturante en geles SDS-PAGE permite separar las manchas (spots) e identificar posteriormente la identidad de las subunidades componentes del complejo mediante espectrometría de masas.

La desventaja de estos métodos es que son lentos y costosos, por lo que el uso de la información obtenida es crucial para nuevos métodos de estudio de la interacción de proteínas, existen distintos sitios donde se almacenan los datos de las proteínas, para poder trabajar con esos datos y confeccionar algoritmos con distintos enfoques que nos permitan estudiar mediante los resultados de estos algoritmos, la dinámica de sus interacciones y así conocer anticipadamente que tipo de estructuras formarán, que energías producirán, cuáles son las interacciones más comunes, etc.

2.1.5. Interacciones Proteína-Proteína (IPP)

Tal como vimos en las moléculas más pequeñas, los átomos se unen a otros mediante enlaces, asi sucesivamente, creando estructuras cada vez mas grandes como las proteínas, y estas a su vez siguen interactuando con sus inmumerables átomos con otras proteínas, cual es la estructura que formarán y que efectos tiene esa nueva macro estructura, depende del tipo de interacción que la produjo. Las interacciones entre proteínas desempeñan un papel fundamental en varios aspectos de la organización estructural y funcional de la célula. Estas interacciones pueden ser clasificadas en tres tipos(que profundizaremos en estas clasificaciones más adelante):

(a) Las que ocurren entre dominios de una misma cadena polipeptídica.
(b) Las que ocurren entre dominios de diferentes cadenas polipeptídicas en proteínas multiméricas.
(c) Las que ocurren de manera transiente en complejos formados entre proteínas independientes.

Las interacciones proteína-proteína (IPP) operan en casi todos los niveles de las funciones celulares, y la mayoría de las proteínas forman parte de algún tipo de complejo proteico en algún momento particular de la vida de una célula. No sería posible mencionar acá todos los caso conocidos en los cuales dos o más proteínas interaccionan entre sí; y mucho menos los casos particulares caracterizados y reportados hasta la fecha. Solo para generar una idea de que tan magníficas y complejas pueden ser las IPP, se dirá que las proteínas forman redes y complejos de interacción en procesos celulares como replicación, transcripción, y traducción del DNA, metabolismo del RNA, control del ciclo celular, metabolismo energético; transducción de señales, transporte de metabolitos, macromoléculas, y otras sustancias tanto en el interior, como intercambio con el medio externo, detoxificación, supervivencia frente a condiciones del entorno desfavorales, respuesta inmune, funciones estructurales como andamiaje celular (citoesqueleto y estructura de organelas), contracción muscular, ensamblado de membranas, etc.

El estudio de las IPP requiere de la utilización de conocimientos y técnicas de diversas áreas

como biología, genética, bioquímica, biofísica. Comprender completamente la naturaleza de estos procesos es uno de los principales objetivos en la actualidad. El conocimiento completo de las redes de interacción proteica no solo es importante para comprender los procesos celulares y sus implicancias desde un punto de vista biológico, sino también por los fines aplicados que de él se desprenden, como son el tratamiento de muchas enfermedades y desórdenes que tienen base en defectos en IPP, diseño de drogas, tratamientos anti-virales y contra otros patógenos,etc.

Las interacciones que se producen entre dos o más proteínas se denominan complejos proteicos o complejos multiproteicos, estos son contactos físicos que se producen por un acoplamiento molecular entre proteínas que se lleva a cabo en una célula o en cualquier elemento organico, son esenciales para cualquier proceso celular, juegan un papel clave para la regulación celular,biosíntesis y degradación de vías, traducción de señales, iniciación de replicación de ADN, controlde expresión de genes, inhibición de enzimas, reconocimiento de anticuerpo-antígeno. [27].

Como fuente informativa adicional respecto a las proteínas, [26] en su publicación nos enseña información similar a la anteriormente mostrada, eso si, con la ventaja de ser un profesional experto en el área, explica más términos que se pudieron escapar en caso de necesitar información mas profunda respecto a las proteínas.

2.1.6. Protein Data Bank

Protein Data Bank (Banco de datos de proteínas) o comúnmente llamado PDB es el único repositorio mundial que alberga información de estructuras tridimensionales de macromoléculas como proteínas y ácidos nucleicos. Hoy en día la base de datos contiene más de cien mil entradas de datos de proteínas disponibles de manera gratuita para todo el público. La gran mayoría de los datos son conseguidos mediante a los métodos de difracción de rayos X y de resonancia magnética nuclear [3]. El PDB se estableció en el año 1971 por el doctor Walter Hamilton en el Brookhaven Nacional.

Laboratory (Laboratorio Nacional Brookhaven) por la sugerencia de la American Crystallographic Association (Asociación americana de cristalografía) y desde el año 1998 hasta la actualidad es gestionado por el Research Collaboratory for Structural Bionformatics (RCSB). En sus inicios el PDB contaba sólo con 7 estructuras, pero con el esfuerzo de biólogos y bioquímicos de todo el mundo, este número ha crecido anualmente de manera casi exponencial hasta llegar a lo que conocemos hoy en día.

Como las entradas de datos en el PDB se tratan de un esfuerzo en conjunto de científicos de todo el mundo, se creó un formato estándar para así facilitar la lectura de estos independiente de su creador, además de definir la forma para describir las estructuras tridimensionales y facilitarla creación de algoritmos para el análisis masivo de los datos. Este formato se creo en el año 1976 por miembros del Brookhaven Nacional Laboratory y se ha ido actualizando en base a nuevos requerimientos y necesidades.

2.1.7. Estructura de los registros del Protein Data Bank

Los registros del PDB contienen gran cantidad de información [31]. Esta información se distribuye en 12 secciones y cada sección comprende distintos apartados, tal y como se indica en la tabla inferior2.11.

Avanzando un poco, esta tabla contiene la sección de los datos tridimensionales de las proteínas del archivo pdb 2.12.

Columna 1: Describe el nombre del registro, el cual puede ser MODEL (modelo), ENDMDL, ATOM (átomo), ANISOU, HETATM y TER. Los registros con los que se trabajará son ATOM, ya que es el registro que posee el nombre y coordenadas de los aminoácidos y sus respectivos elementos químicos que lo conforman, y TER, ya que señala cuando finaliza una cadena de registros ATOM.

Columna 2: Número correlativo del átomo de la proteína.

Columna 3: Muestra el nombre del átomo. Por lo general en el nombre del átomo podemos observar que la primera letra es el símbolo químico de este, seguido por su identificación en cuanto a la cercanía del átomo al grupo funcional. Por ejemplo CA es una abreviación para carbono alfa, carbono central que enlaza el grupo carboxilo y el grupo amino en un aminoácido.

Columna 4: representa el nombre del residuo, tres caracteres que representan la abreviatura de los residuos de los aminoácidos en el registro. Las abreviaturas se pueden observar en la figura (aminoácidos).

Columna 5: Representa el identificador de la cadena, ya que la estructura puede estar formada por varias cadenas polipéptidas, en estos casos cada cadena posee un identificador distinto, ya que una proteína cuya estructura es del tipo cuaternario, tendra distintas "letrasçorrespondientes a las subproteínas que lo componen.

Columna 6: Representa el número de secuencia del aminoácido, este es el número que recibe el residuo descrito en la columna 4.

Columna 7-9: Representan las coordenadas X, Y y Z de todos los átomos de la proteína o complejo proteico registrado. Estas coordenadas describen la posición de cada átomo en un sistema de coordenadas ortogonal. La unidad de medida que utilizan las coordenadas es el Angstrom, que equivale a cien veces un picómetro o la diez mil millonésima parte de 1 metro de longitud, 10(-10) metros. Estas coordenadas serán usadas para calcular las distancias del complejo proteico.

Columna 10: Representa la ocupancia de los átomos. En algunos casos la cadena Lateral e incluso la cadena principal de un aminoácido puede tener dos o más conformaciones, esto debido a la flexibilidad local de esta, por este motivo se utiliza la columna de ocupancia, la cual representa la probabilidad de que el átomo en el registro ocupe la posición determinada. Por lo general ya ocupancia es 1.00, significando que el átomo efectivamente ocupa esa posición y no otro alternativa.

SECTION	DESCRIPTION	RECORD TYPE
Title	Summary descriptive remarks	HEADER, OBSLTE, TITLE, CAVEAT, COMPND, SOURCE, KEYWDS, EXPDTA, AUTHOR, REVDAT, SPRSDE, JRNL
Remark	Bibliography, refinement	REMARKs 1, 2, 3 & annotations
Primary structure	Peptide and/or nucleotide sequence and the relationship between the PDB sequence and that found in the sequence database(s)	DBREF, SEQADV, SEQRES MODRES
Heterogen	Description of non-standard groups	HET, HETNAM, HETSYN, FORMUL
Secondary structure	Description of secondary structure	HELIX, SHEET, TURN
Connectivity annotation	Chemical connectivity	SSBOND, LINK, CISPEP
Miscellaneous features	Features within the macromolecule	SITE
Crystallographic	Description of the crystallographic cell	CRYST1
Coordinate transformation	Coordinate transformation operators	ORIGXn, SCALEn, MTRIXn, TVECT
Coordinate	Atomic coordinate data	MODEL, ATOM, SIGATM, ANISOU, SIGUIJ, TER, HETATM, ENDMDL
Connectivity	Chemical connectivity	CONECT
Bookkeeping	Summary information, end-of-file marker	MASTER, END

Figura 2.11: Actualmente en Julio de 2018, el PDB contiene 148.102 registros. En la siguiente referencia hay enlaces interactivos acerca de esta tabla [20].

Columna 11: Representa el factor temperatura del átomo.

Columna 12: Indica el tipo de átomo del registro, representado por su símbolo químico.

2.1.8. Tipos de IPP

Las interacciones proteína-proteína (IPP) se producen entre cadenas idénticas o no idénticas (es decir, homo o hetero oligomeros, de las estructuras vistas en 2.1.4 representadas como cintas en la figura 2.13). Los oligómeros de unidades de proteínas idénticas u homólogas se pueden organizar de manera isóloga o heteróloga ([25]) con simetría estructural ([16]). Una asociación isóloga implica la misma superficie en ambos monómeros (por ejemplo, represor de arco y lisina, Figuras 1A y C), relacionados por un eje de simetría de 2 veces. En contraste con una asociación isóloga que solo puede oligomerizar más utilizando una interfaz diferente (por ejemplo, formar un dímero de dímeros con tres ejes de simetría de 2 veces), los ensamblados heterólogos utilizan diferentes interfaces que, sin una simetría cerrada (cíclica), pueden conducir a agregación infinita.

En la siguiente figura 2.13 existen diferentes tipos de interacciones proteína-proteína: (A) homodímero obligado, (B) heterodímero obligado, catepsina D humana que consiste en una cadena ligera (roja) no homóloga y pesada (verde), (C) homodímero no obligado, lisina de esperma, (D) complejo heterodímero no obligado, RhoA (verde) y RhoGAP (rojo), (E) heterodímero permanente no obligatorio, inhibidor de trombina (rojo) e inhibidor de rodniina (verde), (F) heterotrímero transitorio no obligado, proteína G bovina, es decir, la interacción entre G alfa (verde) y G Beta Gama (rojo, naranja) es transitoria.

1	2	3	4	5	6	7	8	9	10	11	12
ATOM	1	N	ALA	A	5	41.200	48.757	31.347	1.00	26.76	N
ATOM	2	CA	ALA	A	5	42.192	49.082	30.319	1.00	35.64	C
ATOM	3	C	ALA	A	5	41.919	48.441	28.949	1.00	33.20	C
ATOM	4	O	ALA	A	5	42.004	47.215	28.799	1.00	32.64	O
ATOM	5	CB	ALA	A	5	43.575	48.703	30.799	1.00	41.42	C
ATOM	6	N	GLY	A	6	41.603	49.280	27.958	1.00	23.34	N
ATOM	7	CA	GLY	A	6	41.339	48.800	26.605	1.00	23.33	C
ATOM	8	C	GLY	A	6	42.596	48.240	25.954	1.00	24.96	C
ATOM	9	O	GLY	A	6	43.706	48.708	26.231	1.00	18.32	O
ATOM	10	N	ALA	A	7	42.433	47.258	25.069	1.00	21.08	N
ATOM	11	CA	ALA	A	7	43.576	46.628	24.416	1.00	18.47	C
ATOM	12	C	ALA	A	7	44.234	47.483	23.335	1.00	16.64	C
ATOM	13	O	ALA	A	7	43.622	48.400	22.780	1.00	13.74	O
ATOM	14	CB	ALA	A	7	43.168	45.274	23.833	1.00	12.36	C
ATOM	15	N	VAL	A	8	45.505	47.192	23.077	1.00	11.34	N
ATOM	16	CA	VAL	A	8	46.268	47.859	22.033	1.00	5.64	C
ATOM	17	C	VAL	A	8	46.763	46.696	21.197	1.00	6.86	C
ATOM	18	O	VAL	A	8	47.570	45.885	21.654	1.00	13.08	O
ATOM	19	CB	VAL	A	8	47.436	48.657	22.603	1.00	8.12	C
ATOM	20	CG1	VAL	A	8	48.279	49.264	21.469	1.00	10.73	C
ATOM	21	CG2	VAL	A	8	46.900	49.755	23.498	1.00	7.38	C
ATOM	22	N	ILE	A	9	46.164	46.531	20.028	1.00	9.96	N
ATOM	23	CA	ILE	A	9	46.506	45.426	19.141	1.00	10.98	C
ATOM	24	C	ILE	A	9	47.229	45.903	17.898	1.00	13.08	C
ATOM	25	O	ILE	A	9	47.422	47.097	17.721	1.00	11.98	O
ATOM	26	CB	ILE	A	9	45.245	44.662	18.736	1.00	10.19	C
ATOM	27	CG1	ILE	A	9	44.335	45.551	17.880	1.00	6.94	C
ATOM	28	CG2	ILE	A	9	44.518	44.146	19.993	1.00	7.37	C
ATOM	29	CD1	ILE	A	9	43.141	44.811	17.344	1.00	5.00	C
ATOM	30	N	ASP	A	10	47.687	44.972	17.069	1.00	12.96	N
ATOM	31	CA	ASP	A	10	48.371	45.330	15.826	1.00	14.03	C
ATOM	32	C	ASP	A	10	47.348	45.992	14.916	1.00	6.63	C
ATOM	33	O	ASP	A	10	46.232	45.491	14.768	1.00	5.00	O
ATOM	34	CB	ASP	A	10	48.925	44.081	15.108	1.00	26.20	C
ATOM	35	CG	ASP	A	10	50.186	43.521	15.757	1.00	26.77	C
ATOM	36	OD1	ASP	A	10	50.790	44.196	16.615	1.00	32.80	O
ATOM	37	OD2	ASP	A	10	50.591	42.401	15.385	1.00	33.56	O
ATOM	38	N	GLY	A	11	47.743	47.101	14.293	1.00	11.77	N
ATOM	39	CA	GLY	A	11	46.848	47.813	13.401	1.00	10.78	C
ATOM	40	C	GLY	A	11	46.156	46.907	12.406	1.00	8.96	C

Figura 2.12: Sección coordenadas tridimensionales de archivo pdb.

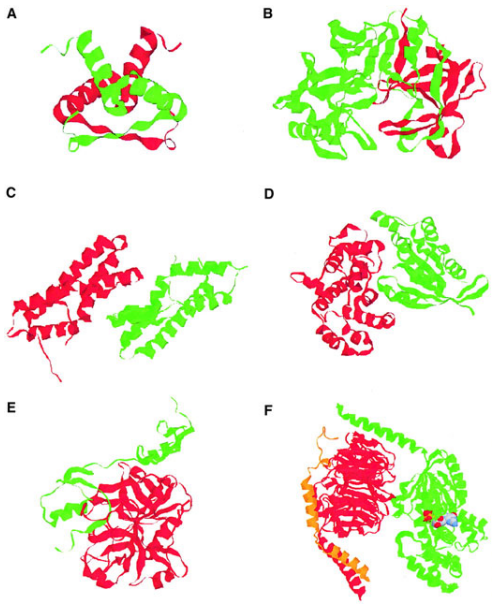

Figura 2.13: Ejemplos de diferentes tipos de interacciones proteína-proteína.

2.1.9. Complejos no obligados y obligados

Además de la composición, se pueden distinguir dos tipos diferentes de complejos en función de si un complejo es obligado o no. En una IPP obligado, los protómeros no se encuentran como estructuras estables por sí solos en vivo. Dichos complejos generalmente también son funcionalmente obligatorios; por ejemplo, el dímero represor de Arco ,Figura 2.13A, es esencial para la unión al ADN [32]. Muchas de las estructuras heterooligoméricas en Protein Data Bank implican interacciones no obligatorias de protómeros que existen de forma independiente, tales como complejos de señalización intracelular, por ejemplo, RhoA-RhoGAP, Figura 2.13D, y anticuerpo-antígeno, receptor-ligando e inhibidor de enzima (por ejemplo, trombina-rodniin, Figura 2.13E) complejos. Los componentes de tales complejos proteína-proteína a menudo inicialmente no están localizados y, por lo tanto, necesitan ser independientemente estables. Sin embargo, algunos homo-oligómeros, que por definición están co-localizados, también pueden formar conjuntos no obligados (por ejemplo, lisina de esperma, Figura 2.13C).

2.1.10. Complejos transitorios y permanentes

Las IPP también pueden distinguirse en función de la vida útil del complejo. A diferencia de una interacción permanente que suele ser muy estable y, por lo tanto, solo existe en su forma compleja, una interacción transitoria se asocia y disocia in vivo. Distinguimos interacciones transitorias débiles que presentan un equilibrio oligomérico dinámico en solución, donde la interacción se rompe y se forma continuamente, por ejemplo, lisina, Figura 2.13 C y fuertes asociaciones transitorias que requieren un disparador molecular para desplazar el equilibrio oligomérico. Por

ejemplo, la proteína G heterotrimérica, Figura 2.13 F se disocia en las subunidades G Alfa y G Beta Gama tras la unión del trifosfato de guanosina (GTP), pero forma un trímero estable con guanosina difosfato (GDP) unido. Las interacciones obligadas estructural o funcionalmente son generalmente permanentes, mientras que las interacciones no obligatorias pueden ser transitorias o permanentes [32].

Es importante tener en cuenta que muchas IPP no se clasifican en distintos tipos. Por el contrario, existe un continuo entre las interacciones obligadas y no obligatorias, y la estabilidad de todos los complejos depende en gran medida de las condiciones fisiológicas y el entorno (ver más abajo). Una interacción puede ser principalmente transitoria in vivo pero volverse permanente bajo ciertas condiciones celulares. Los datos plegables, así como los datos sobre la dinámica del ensamblaje en diferentes condiciones fisiológicas o entornos, a menudo no están disponibles. Sin embargo, la ubicación subcelular de las subunidades y la función de la proteína a menudo sugieren el tipo de interacción biológicamente relevante; por ejemplo, se espera que las interacciones en la señalización intracelular sean transitorias, ya que su función requiere una asociación y disociación listas. Complementar con [6].

2.1.11. Zona de interacción

Las proteínas involucradas en una interacción proteína-proteína no utilizan toda su estructura para formar la interacción entre cadenas, sino que lo hacen mediante una pequeña zona en la cual se produce el enlace entre proteínas. Esta zona se denomina zona de interacción (interface), la que consiste en el enlace de residuos de aminoácidos mediante puentes de hidrógeno, que pertenecen a dos cadenas diferentes, por lo que podemos decir que esta zona está formada por fragmentos de ambas cadenas polipéptidas [19].

La zona de interacción posee propiedades diferentes al resto de la estructura del complejo proteico, esto permite que la proteína interactúe específicamente con una o más proteínas.

En la figura 2.14, se puede ver un ejemplo de interacción entre dos proteínas del complejo ID:1JKG. La imagen esta representado con esferas aquellos aminoácidos de cada proteína que pertenecen a la zona de interacción. Nuevamente, en este tipo de representación, hay mezclas de representación de tipos de estructuras visto en 2.1.4, acá las estructuras hélice alfa y hojas beta se ven como simples cintas, pero están formadas por átomos como se explicó anteriormente, esto permite simplificar solo el esqueleto de la proteína, ya que ésta última contiene más elementos, como las cadenas laterales, grupos aminos y carboxilos, y estructuras cercanas a la proteína que no tienen clasificación por ser pequeños grupos, como los vistos en la figura 2.8, cuyas representaciones en blanco son muchos grupos mas pequeños. Imagen obtenida en [32].

2.1.12. Metodos de detección de IPP

- **Cristalografía de Rayos-X y espectrocopía de RMN:** La cristalografía de rayos X ha sido la técnica más importante para el análisis estructural de proteínas y complejos proteicos dada su precisión y resolución. Ejemplos de complejos resueltos mediante esta técnica son: La RNA polimerasa, el ribosoma completo, varias estructuras de cápside viral,

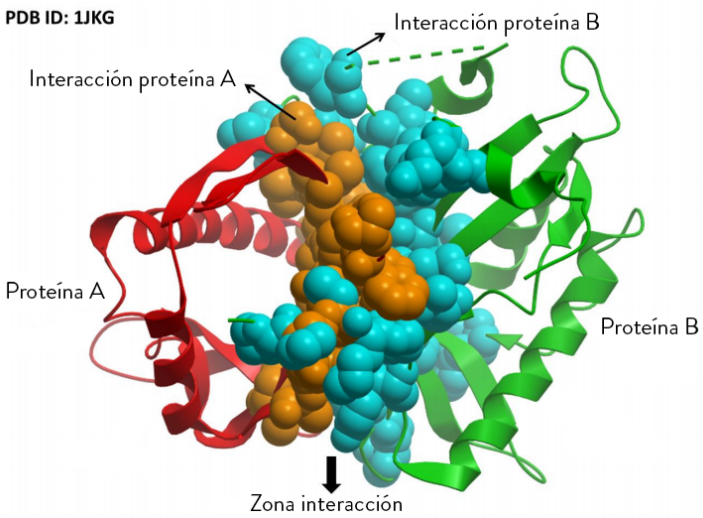

Figura 2.14: Ejemplo interacción entre dos proteínas en el complejo ID:1JKG.

el proteasoma, etc. Si bien no hay límites de tamaño para la estructura, la determinación de estructuras de complejos proteicos es más complicada que la de proteínas individuales, principalmente debido a la dificultad de producir suficientes cantidades de muestra y su cristalización. En líneas generales, la cristalografía de rayos X integra los patrones de difracción obtenidos después de bombardear el complejo proteico con rayos X para reconstruir su estructura tridimensional. Esta técnica provee estructuras de resolución atómica y por lo tanto los detalles moleculares de las interacciones.

- **Espectroscopia de RMN:** La espectroscopia de RMN extrae las distancias entre los átomos midiendo las transiciones entre diferentes estados de spin nuclear en un campo magnético. Estas distancias después se usan como restricciones para la construcción de estructuras 3D. La espectroscopía de RMN también provee resolución a escala atómica, pero generalmente se encuentra limitada a proteínas de aproximadamente 300 residuos. Esta técnica se ha venido utilizando para identificar los residuos involucrados en la interacción entre proteínas.
- **Microscopía electrónica y tomografía electrónica:** Existen múltiples variantes de microscopía electrónica (ME), incluidas microscopía electrónica de partícula única, tomografía electrónica y cristalografía electrónica de arreglos regulares bidimensionales de la muestra. La ME se basa en la visualización de imágenes de partículas teñidas. Muchas vistas y conformaciones deben ser capturadas para reconstruir la estructura tridimensional del complejo. La técnica de crio-ME de partícula única determina a densidad electrónica de un ensamblado con una resolución da aproximadamente 5 Å. La estructura tridimensional completa de una partícula se obtiene reconstruyendo múltiples proyecciones en dos dimensiones de la muestra, cada una mostrando un ángulo distinto. La obtención de imágenes por esta técnica no requiere de grandes cantidades de muestra, ni que ésta esté cristaliza-

da. Aunque la técnica de crio-ME no permite generar modelos a escala atómica provee un entendimiento de la estructura y simetría general de grandes complejos.

Los métodos de tomografía están basados en múltiples vistas de distintos ángulos del mismo objeto; la muestra bajo estudio se va inclinando progresivamente sobre un eje perpendicular al haz de electrones, y con el set de imágenes de proyecciones obtenido se reconstruye la estructura 3D. Si bien permite estudiar ensamblados aislados con una relativa baja resolución de unos 30 Å, su potencial radica en la visualización de complejos en un contexto celular, sin perturbar el ambiente fisiológico.

- **Métodos experimentales de baja resolución:** Existen numerosos métodos que proveen información estructural de baja resolución. Esta información puede ser utilizada para inferir la configuración de las proteínas en un complejo, y establecer las restricciones de contacto o proximidad útiles en el modelado de complejos de órdenes mayores. Algunos de estos métodos son nuevas variantes del sistema de dos híbridos, cromatografía de afinidad y bibliotecas de display en fagos. Además existen numerosos métodos biofísicos, bioquímicos y de biología molecular que pueden proveer información estructural: mutagénesis sitio dirigida, que permite identificar residuos que participan en la interacción, varias formas de footprinting, como el intercambio hidrógeno/deuterio, sobre el cual se hablará más adelante, que permite identificar superficies enterradas al formarse el complejo, cross-linking químico, para identificar proteínas y residuos interactuantes, transferencia de energía por resonancia de fluorescencia (FRET), también mencionado más adelante, que permite determinar la distancia entre grupos fluorescentes (o marcados con fluoróforos), y scattering de rayos X de ángulo chico (SAXS), entre otros.
- **Métodos computacionales:** Cuando las estructuras atómicas individuales de las proteínas que forman parte de un complejo se conocen, existen muchos métodos y algoritmos computacionales que permiten identificar potenciales superficies de unión; así como predecir o sugerir una estructura para la interacción. La mayoría de estos métodos, que se conocen como métodos de docking, tiene como objetivo predecir el modelo atómico del complejo, maximizando la forma y la complementariedad química entre un par dado de proteínas que interactúan. Estás metodologías generalmente se basan en un enfoque de dos estadíos: primero generan un set de posibles orientaciones para las proteínas acopladas, y luego se les asigna un score (similar a un puntaje), con la suposición de que la estructura nativa del complejo tendrá un rango alto.

  Los métodos de docking no son suficientemente precisos como para predecir si dos proteínas dadas interactúan entre sí, pero sí han permitido identificar con bastante éxito posibles superficies de interacción, y las superficies de contacto entre dos subunidades estructuralmente definidas.

2.1.13. Predicción de IPP

Gracias a los grandes avances en tecnología de alto rendimiento, una gran cantidad de datosde interacciones proteína-proteína se encuentran disponibles en numerosas bases de datos, varias de

estas bases de datos se encuentran en línea y disponibles para la comunidad científica para su análisis. Entre ellas se destacan BioGRID (General Repository of Interaction Database), una base de datos que contiene todas las interacciones genéticas o entre proteínas conocidas, DIP (Database of Interacting Proteins), una base de datos que combina datos provenientes de varias fuentes para formar un conjunto consistente de interacciones proteína-proteína y el Protein Data Bank. Estos datos son utilizados entre otras cosas, en el campo de la bioinformática sobre la predicción de interacciones de proteína-proteína que busca por medio de herramientas computacionales, identificar y categorizar el tipo de interacción que se produce en un complejo proteico dado.

Como la predicción de las IPP no tiene un mecanismo fijo o descubierto que sea exacto, podemos ver experimentos muy interesantes en Interacciones proteína-proteína: bases de datos y métodos teóricos de predicción [10]. Existen diversos métodos utilizados en el área de la predicción de interacciones, los cuales se pueden clasificar en 4 categorías:

1. métodos basados en el contenido genómico y función estructural
2. métodos que utilizan redes topológicas
3. métodos que utilizan minería de textos
4. métodos que utilizan aprendizaje automático para la predicción de interacciones

Lo que sigue más adelante estará basado en el item 4 anterior, como informáticos, nuestra tarea es trabajar con los datos, buscar formas de interpretar y dar sentido a estos datos para entregar información relevante de ellos. Como se ha mencionado previamente, existen múltiples lugares donde obtener datos, pero existe una gran cantidad de información oculta, de gran importancia estratégica, a la que no se puede acceder por las técnicas clásicas de recuperación de la información. El descubrimiento de esta información oculta es posible gracias a la Minería de Datos (Data Mining), que entre otras sofisticadas técnicas aplica la inteligencia artificial, para encontrar patrones y relaciones dentro de los datos permitiendo la creación de modelos, es decir, representaciones abstractas de la realidad, pero es el descubrimiento del conocimiento (KDD, por sus siglas en inglés) que se encarga de la preparación de los datos y la interpretación de los resultados obtenidos, los cuales dan un significado a estos patrones encontrados.

Así el valor real de los datos reside en la información que se puede extraer de ellos, información que ayude a tomar decisiones o mejorar nuestra comprensión de los fenómenos que nos rodean. Hoy, más que nunca, los métodos analíticos avanzados son el arma secreta de muchos negocios exitosos. Empleando métodos analíticos avanzados para la explotación de datos, los negocios incrementan sus ganancias, maximizan la eficiencia operativa, reducen costos y mejoran la satisfacción del cliente.

2.1.14. Descubrimiento del conocimiento KDD

Para el proceso del KDD se aplica inducción de reglas, los problemas de clasificación y clustering, el reconocimiento de patrones, el modelado predictivo, la detección de dependencias, etc.

Los datos recogen un conjunto de hechos (una base de datos) y los patrones son expresiones

que describen un subconjunto de los datos (un modelo aplicable a ese subconjunto). El KDD involucra un proceso iterativo e interactivo de búsqueda de modelos, patrones o parámetros, los cuales descubiertos han de ser válidos,novedosos para el sistema y potencialmente útiles.

2.1.15. Data Mining

La Data Mining es una etapa dentro del proceso completo del descubrimiento del conocimiento, este intenta obtener patrones o modelos a partir de los datos recopilados. Decidir si los modelos obtenidos son útiles o no suele requerir una valoración subjetiva por parte del usuario. Los algoritmos de data mining suelen tener tres componentes:

1. El modelo, que contiene parámetros que han de fijarse a partir de los datos de entrada.
2. El criterio de preferencia, que sirve para comparar modelos alternativos.
3. El algoritmo de búsqueda, que viene a ser como cualquier otro programa de inteligencia artificial (IA).

El criterio de preferencia suele ser algún tipo de heurística y los algoritmos de búsqueda empleados suelen ser los mismos que en otros programas de inteligencia artificial. Las principales diferencias entre los algoritmos de data mining se hallan en el modelo de representación escogido y la función del mismo, es decir según el objetivo perseguido.

2.1.16. Herramientas de Data Mining

Las herramientas de data mining empleados en el proceso de KDD se pueden clasificar en dos grandes grupos:

- Técnicas de verificación, en las que el sistema se limita a comprobar hipótesis suministradas por el usuario.
- Métodos de descubrimiento, en los que se han de encontrar patrones potencialmente interesantes de forma automática, incluyendo en este grupo todas las técnicas de predicción.

El resultado obtenido con la aplicación de algoritmos de data mining pertenecientes al segundo grupo, el de técnicas de descubrimiento, pueden ser de carácter descriptivo o predictivo. Las predicciones sirven para prever el comportamiento futuro de algún tipo de entidad mientras que una descripción puede ayudar a su comprensión. La aplicación de técnicas de data mining en grandes bases de datos persiguen los siguientes resultados:

1- **Clasificación:** Se trata de obtener un modelo que permita asignar un caso de clase desconocida a una clase concreta (seleccionada de un conjunto redefinido de clases), como son los árboles de clasificación (CART), cuyos resultados pueden expresarse mediante reglas ejecutables directamente del SQL o el método de Bayesiano.

2- **Regresión:** Se persigue la obtención de un modelo que permita predecir el valor numérico de alguna variable (modelos de regresión logística).

3- **Agrupamiento (clustering):** Hace corresponder cada caso a una clase, con la peculiaridad de que las clases se obtienen directamente de los datos de entrada utilizando medidas

de similaridad. Es decir, agrupan a los datos bajo diferentes métodos y criterios. Las técnicas más usadas son las clásicas (distancia mínima) y las redes neuronales (método de Kohonen o método de Neural-Gas).

4- **Resumen:** Se obtienen representaciones compactas para subconjuntos de los datos de entrada (análisis interactivo de datos, generación automática de informes, visualización de datos).

5- **Modelado de Dependencias:** Se obtienen descripciones de dependencias existentes entre variables. El análisis de relaciones (por ejemplo las reglas de asociación), en el que se determinan relaciones existentes entre elementos de una base de datos, podría considerarse un caso particular de modelado de dependencias.

6- **Análisis de Secuencias:** Se intenta modelar la evolución temporal de alguna variable, con fines descriptivos o predictivos (redes neuronales multicapas).

2.2. Aprendizaje Automático

Aprendizaje automático es la adquisición de nuevo conocimiento, el desarrollo de un motor y habilidades cognitivas a través de instrucciones o prácticas, la organización de nuevo conocimiento, representación efectiva y descubrimiento de nuevos hechos y teorías a través de la observación y experimentación.

Los tipos de conocimiento adquirido son parámetros en expresiones algebraicas, árboles de decisión, gramática formal, producción de reglas, lógica formal basada en expresiones, grafos y redes, marcos y esquemas y otras codificaciones procedimentales y programas de cómputo.

Este aprendizaje es aplicado a muchas áreas como la química, educación, programación computacional, sistemas expertos, videojuegos, matemáticas, música, procesamiento del lenguaje natural, robótica, reconocimiento del habla e imagen, y secuencias de predicción entre otras.

En la vida diaria vemos como las tecnologías de redes sociales cada vez siguen implementando nuevos y mejores algoritmos, que son de aprendizaje automático, por ejemplo en youtube al subir un video o música con derechos de autor, este gigante de la multimedia, automáticamente elimina los segmentos protegidos por estos derechos, asi mismo pasa no solo con el contenido existente con derechos de autor sino que se aplican a los nuevos contenidos, otro ejemplo es como facebook identifica fotos falsas de los perfiles, esto se realiza mediante un algoritmo que evalúa estrictamente a usuarios nuevos, analiza las conductas del usuario nuevo, y de antemano es exigente con el contenido cargado por este integrante de la red, este es otro ejemplo de algoritmos basados en aprendizaje automático.

Para este aprendizaje automático, se debe escoger aquellas características más relevantes de los datos, estas características difícilmente o nunca estarán explicitas en un dato en particular, sino en el conjunto de los datos. Para obtener estos datos debemos aplicar algoritmos que identifiquen los patrones más destacados del conjunto de datos. Esos algoritmos deben identificar problemas de clasificación, regresión y agrupamiento.

- **Problema de clasificación:** involucra predecir si un conjunto de datos pertenece a una cierta categoría predefinida. Cuando tratamos con dos posibles categorías o clases, hablamos de clasificación binario, si están definidas más de dos clases, hablamos de clasificación multiclase.
- **Problema de regresión:** es similar al problema de clasificación ya que ambos intentan relacionar las entradas y salidas de los conjuntos de datos de entrenamiento para predecir conjuntos de datos futuros, pero a diferencia de la clasificación, en el problema de regresión se intenta predecir valores numéricos reales.
- **Problema de agrupamiento:** En el problema de agrupamiento pretende formar agrupaciones de objetos que son similares, asegurándose que todas las agrupaciones formadas sean distintas entre sí. A diferencia del problema de clasificación, el problema de agrupamiento no necesita tener una categorización de clases.

Debido a lo amplio y complejo que es el campo del conocimiento, existe una gran cantidad de algoritmos de aprendizaje que lidian con diferentes tareas de aprendizajes. Estos algoritmos se clasifican por estilo de aprendizaje.

- **Aprendizaje supervisado:** El aprendizaje supervisado es bastante común en los problemas de clasificación porque el objetivo suele ser lograr que la computadora aprenda un sistema de clasificación que hemos creado. El reconocimiento de dígitos, una vez más, es un ejemplo común de aprendizaje de clasificación. De manera más general, el aprendizaje de la clasificación es apropiado para cualquier problema donde la deducción de una clasificación es útil y la clasificación es fácil de determinar. En algunos casos, tal vez ni siquiera sea necesario dar clasificaciones predeterminadas a cada instancia de un problema si el agente puede resolver las clasificaciones por sí mismo. Este sería un ejemplo de aprendizaje no supervisado en un contexto de clasificación[1].
- **Aprendizaje no supervisado:** El aprendizaje no supervisado parece mucho más difícil: el objetivo es que la computadora aprenda a hacer algo que no le decimos cómo hacer. En realidad, hay dos enfoques para el aprendizaje no supervisado. El primer enfoque es enseñar al agente no mediante categorizaciones explícitas, sino mediante el uso de algún tipo de sistema de recompensa para indicar el éxito. Tenga en cuenta que este tipo de capacitación generalmente encajará en el marco del problema de decisión porque el objetivo no es producir una clasificación, sino tomar decisiones que maximicen las recompensas. Este enfoque generaliza bien al mundo real, donde los agentes pueden ser recompensados por hacer ciertas acciones y castigados por hacer otras.

Capítulo 3

Estudio del problema

Se ha descrito como la parte mas pequeña de la materia (átomo) interactúa con otras pares, formando estructuras cada vez más grandes y compuestas, estos átomos forman aminoácidos que al unirse [1] a otros aminoácidos , producen energías y residuos en cadena hasta formar una proteína, las proteínas a su vez se unen a otras proteínas produciendo más energías y residuos. Estas interacciones son responsables de muchas estructuras mayores, pero en este estudio, se busca aportar más factores relevantes de las interacciones entre proteínas mediante el análisis de las distancias máximas y mínimas, con el fin de conocer más el rol que cumplen cada una de estas interacciones. Esto puede ser útil para personal científico que estudia las funciones de las proteínas, obtener datos más específicos de las interacciones, al relacionarlos con las distancias que existen en estas. Por ejemplo, existe cierto estudio respecto a un virus en las cuales se analizaron sus IPP, el resultado de este análisis entregó nuevas funciones de las proteínas, como la de romper la pared de una bacteria, si esas interacciones se estudiaran aún más en profundidad, midiendo sus distancias, esto podría aportar a este personal más datos relevantes para analizar el comportamiento de ciertas proteínas en otras condiciones.

En nuestro estudio, como en la actualidad los resultados de las IPP están almacenados en bases de datos públicas, es conveniente trabajar con estos datos mediante mecanismos informáticos, para obtener información que permita concluir estudios como el mencionado anteriormente, y así avanzar en la gran meta de predecir las IPP. Profundizaremos con esta investigación, esta [6] en el cual propone el estudio de las características energéticas más significativas, pertenecientes a la zona de interacción en un conjunto de complejos proteicos, por medio de la búsqueda de patrones mediante a aprendizaje automático. Los mejores resultados de su estudio estuvieron cercanos al 81 % de predicción de interacciones.

Para llevarlo a cabo, se obtendrán las bases de datos de los complejos proteicos que serán utilizados como conjunto de datos de entrenamiento para los algoritmos de aprendizaje. Seguidamente se buscará la sección donde identifica las proteínas pertenecientes al complejo. Enseguida se buscará la sección de los datos almacenados de la IPP, específicamente los aminoácidos de la

[1] Las uniones se realizan entre el grupo amino de un aminoácido y el grupo carboxilo de otro aminoácido, en mi opinión, probablemente se generan por alguna polaridad electromagnética positiva y negativa en estos grupos

IPP y la energía producida. El siguiente paso es analizar las coordenadas del aminoácido en la base de datos correspondiente a la proteína, una vez ahí, se buscará el aminoácido del complejo. Posteriormente se analizarán las coordenadas del segundo aminoácido en la base de datos de la segunda proteína, para finalmente calcular la distancia entre aminoácidos. Como el estudio esta basado en las distancias máximas y mínimas en los complejos proteicos, los resultados anteriores serán sometidos a análisis mediante un algoritmo que nos entregará del total de datos, los aminoácidos que pertenecen a la IPP cuya distancia es menor y mayor en el complejo proteico junto a los datos importantes de esas partículas.

3.1. Procedimientos a seguir

Para realizar el estudio, es necesario seguir los siguientes pasos. Al obtener los datos, se debe comprender la estructura de los archivos de las plataformas donde están almacenados los datos científicos de las proteínas, posteriormente se necesitará separar las secciones a medir de la estructura original, para que estas distancias puedan clasificarse posteriormente. Identificando las secciones se podrá confeccionar un algoritmo para recopilar las coordenadas y calcular las distancias. A continuación, con las distancias calculadas, se confeccionará un archivo con la estructura del software de aprendizaje automático, con el cual se pretende obtener resultados estadísticos de clasificación. Se estudiarán los resultados de la aplicación anterior finalmente se haran las conclusiones de todas las actividades realizadas.

Específicamente las actividades a realizar son las siguientes:

- Descargar archivos de complejos proteicos.
- Separar de cada complejo proteico el ligando y receptor y guardarlos en archivos separados.
- Ingresar ambos archivos anteriores a `fastcontact`, este software nos devolverá los mismos dos archivos pero con uno extra que corresponde a 20 energías que contribuyen más y menos al complejo proteico.
- Crear un directorio con los archivos anteriores e identificarlos por el nombre del complejo
- Crear directorios que contengan los subdirectorios que contienen a los complejos por las categorías Transientes y Obligados.
- Por cada directorio, crear un archivo a partir del generado por `fastcontact`, de la sección de aminoácidos relacionados por sus energías.
- De cada par de aminoácidos de cada energía, identificar el número y nombre de los átomos de cada uno con el cual la magnitud vectorial formada entre ellos sea máxima.
- De cada par de aminoácidos de cada energía, identificar el número y nombre de los átomos de cada uno con el cual la magnitud vectorial formada entre ellos sea mínima.
- Crear un archivo que identifique los directorios con los complejos transientes y obligados, leer cada archivo que contiene las distancias del ítem anterior, ordenar cada uno de los directorios de los complejos en una fila, y almacenarlos.
- Leer el archivo anterior para formatearlo, agregar una fila que asigne un título y tipo de dato por cada columna, introducir el símbolo coma entre cada columnas y guardar con extensión `.arff`

3.2. Obtención de los datos

Los datos previamente mencionados ya están preparados y existe una cantidad importante de ellos para ser ingresados al software de aprendizaje automático como conjunto de datos de entrenamiento.

Mintseris y Weng en 2005 realizaron un estudio sobre la estructura, función y evolución de la secuencia de interacciones proteína-proteína de complexos permanentes y transitorios para obtener una mejor clasificación de estos. En su estudio, compilaron un conjunto de datos no redundante de complejos proteicos pertenecientes al `Protein Data Bank` y manualmente separados en complejos proteicos permanentes y transitorios [22].

Este conjunto de datos, corresponde a 326 complejos, de los cuales 115 corresponden a interacciones proteína proteína permanentes y 211 a interacciones transitorias. El listado corresponde a las ID's de los 326 complejos proteicos, cuya identificación es accesible en el repositorio `Protein Data Bank`. En la investigación realizada por Gutiérrez-Bunster [6] se obtuvieron todos los archivos con extensión `pdb` de los complejos.

3.3. Propiedades energéticas e interacciones entre aminoácidos

En este trabajo de investigación se utilizarán las propiedades energéticas y las distancias entre los aminoácidos pertenecientes a la zona de interacción, para obtener esta información, se utilizará la aplicación llamada `fastcontact` [7].

3.3.1. Software fastcontact

`Fastcontact` es un software desarrollado por Carlos Camacho y Chao Zhang, en conjunto con su Departamento de Biología Computacional en la Universidad de Pittsburgh. Esta aplicación permite estimar las energías libres de interacción entre dos proteínas. Para lograr esto fastcontact utiliza una función de puntuación de energía libre, la cual asegura que las energías que aportan los residuos pertenecen a la zona de interacción.

Para poder utilizar el software, el archivo de extensión `pdb` de cada complejo se deben separar en dos archivos `pdb` que contienen la información de cada proteína por separado. Gutiérrez-Bunster realizó la separación de estos complejos para su posterior utilización en el software.

`Fastcontact` entrega tres archivos por cada complejo proteico ingresado. Dos de ellos sonarchivos de texto similares a los archivos `.pdb` ingresados, los cuales contienen sólo la sección de coordenadas del archivo `pdb` original, excluyendo en esta sección la ocupancia, factor temperatura y elemento químico. El tercer archivo de salida de la aplicación contiene las características energéticas. Estas características están organizadas en diferentes secciones. Cada sección contiene los 20 residuos que más energía contribuyen en la interacción y los 20 residuos que menos energía contribuyen.

En la Tabla 3.1 se ven las secciones de datos que contiene el archivo generado por `fastcontact` y los títulos de los datos que lo componen. Las últimas dos filas de color rojo corresponden a

Residuos que contribuyen a la energía libre de unión.	20 Mínimos 20 Máximos
Residuos Ligando que contribuyen a la energía libre de desolvatación.	20 Mínimos 20 Máximos
Residuos Ligando que contribuyen a la energía electrostática.	20 Mínimos 20 Máximos
Residuos Receptor que contribuyen a la energía libre de desolvatación.	20 Mínimos 20 Máximos
Residuos Receptor que contribuyen a la energía electrostática.	20 Mínimos 20 Máximos
Residuos electrostáticos contacto Receptor-Ligando.	20 Mínimos 20 Máximos
Residuos energía libre contacto Receptor-Ligando.	20 Mínimos 20 Máximos

Cuadro 3.1: Estructura de archivo `fastcontact`.

los pares de residuos que interactúan, de ellos se obtendrán los datos que necesitamos para los algoritmos posteriores.

Por lo tanto existen tres archivos por complejo proteico:

- Proteina A (fort.19)
- Proteina B (fort.20)
- Archivo generado por fastcontact con las propiedades energéticas (idcomplejo.txt)

En la figura 3.1 se puede ver un extracto del archivo que contiene a una de las dos proteínas a medir. Las secciones con recuadros coloreados corresponden a los aminoácidos y sus elementos, como el hidrógeno (H), carbono(C), carbono alfa (CA), estos por ejemplo son comunes entre ellos. Una proteína puede estar compuesta por otras, dependiendo de su estructura 2.1.4. En la columna color amarillo identifica la cadena de la proteína a la que pertenecen los aminoácidos en los recuadros coloreados. En este archivo pueden existir múltiples cadenas proteicas.

En la figura 3.2 se puede ver la sección que se usará del archivo `fastcontact`. Se puede observar la energía en la primera columna, el número del aminoácido de la proteína A en la segunda columna, el nombre del aminoácido de la proteína A en la tercera columna, el número del aminoácido de la proteína B en la cuarta columna, el nombre del aminoácido de la proteína B en la quinta columna. Más adelante veremos más acerca de esta tabla.

Figura 3.1: Datos que contiene la base de datos de una proteína.

3.4. Medida de distancia entre cadenas de átomos

Con el archivo de energías de los aminoácidos de las proteínas obtenido de `fastcontact` y las bases de datos que contienen información estructural completa de cada proteína del complejo, se podrá realizar el cálculo mediante un algoritmo, que seleccionará cada aminoácido de cada proteína de cada complejo, lo comparará mediante las distancias obtenidas entre cada uno de sus átomos y almacenará aquellas que son máximas y mínimas.

La distancia utilizada es la que se obtiene con el método distancia Euclidiana para calcular la longitud entre dos puntos, en nuestro modelo tridimensional, nos permitirá calcular la magnitud del los vectores formados entre los átomos relacionados en el complejo proteico, este método mide los puntos sin importar la ubicación que tengan.

En la figura 3.3 se aprecia como se realizan las mediciones, las lineas verdes muestran esquemáticamente como se miden las magnitudes de las distancias con el método de Euclidiano, estas magnitudes corresponden a las relaciones de aminoácidos relacionados del complejo proteico. Las imágenes no son una relación real sino un montaje para representación, ambas proteínas si corresponden al mismo complejo proteico tipo transiente.

```
Top 20 Min & Max receptor-ligand residue electrostatic contacts
 -0.313  248 GLY   35 THR
 -0.276  244 LEU   35 THR
 -0.177  216 PRO   32 TYR
 -0.094  180 LEU   64 LYS
 -0.056  216 PRO   39 GLU
 -0.031  245 HIS   68 LEU
 -0.028  244 LEU   36 LEU
 -0.027   92 PRO   35 THR
 -0.025  180 LEU   63 GLU
 -0.022  165 GLN   66 GLY
 -0.014  244 LEU   37 GLY
 -0.014  215 ASP   12 GLY
 -0.013  163 GLU    9 VAL
 -0.012   82 VAL   70 ASP
 -0.012  214 GLU   12 GLY
 -0.012  165 GLN    9 VAL
 -0.011  180 LEU   65 PHE
 -0.010  180 LEU   62 GLN
 -0.010  181 PRO   12 GLY
 -0.010  216 PRO   27 GLU
-------------------------
  0.644  246 ASN   71 GLY
  0.649   69 ASP   69 ARG
  0.658  165 GLN  100 ASP
  0.677  163 GLU   62 GLN
  0.705  217 ILE   32 TYR
  0.718   85 GLN   35 THR
  0.802  215 ASP   59 THR
  0.846  245 HIS   69 ARG
  0.856  247 PHE   35 THR
  0.876  213 ASP   64 LYS
```

Figura 3.2: Sección datos energéticos a utilizar de aminoácidos obtenido de `fastcontact`.

$$átomoA : (a_x, a_y, a_z)$$

$$átomoB : (b_x, b_y, b_z)$$

Cuadro 3.2: Representación de puntos en un espacio tridimensional.

$$Distancia_{AB} = |\vec{AB}| = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2 + (a_z - b_z)^2}$$

Cuadro 3.3: Fórmula para el cálculo de distancia de dos puntos en un vector tridimensional.

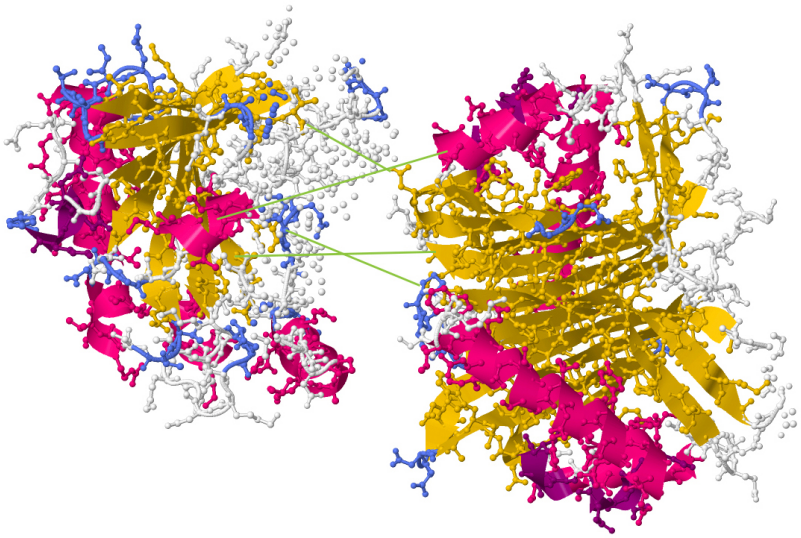

Figura 3.3: Ejemplo de las magnitudes vectoriales formadas por los átomos entre las proteínas en el complejo ID 1a2k.

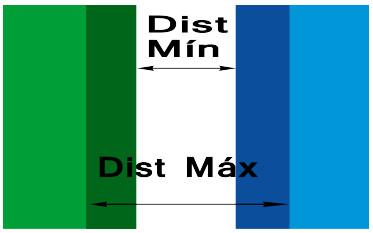

Figura 3.4: Ejemplo de las magnitudes vectoriales esquemático en complejo proteico.

Capítulo 4

Implementación de algoritmos

Para confeccionar el algoritmo de distancias, se escogió el lenguaje python, por su gran popularidad en el desarrollo de aplicaciones bioinformáticas y de su extensa documentación formal e informal, junto a una extensa colección de funciones que simplifican bastante el desarrollo de soluciones complejas, también ayuda enormemente al momento de querer implementar soluciones en ésta y muchas otras áreas. Otra característica importante de python, es que en la actualidad, con la diversificación de sistemas operativos, es amigable para muchos usuarios, ya que es compatible con los principales sistemas operativos, eliminando la barrera de tecnologías monopolizadas de cierto sistema en particular, con el fin de que el uso de este lenguaje no sea incorporado como un costo adicional a la investigación biológica, respecto a licenciamiento de software ya que es distribuido mediante BSD[1].

En la figura 4.1 existe un directorio raíz (root u origen) donde se ejecutará el software, este directorio contiene subdirectorios con los datos, el que contiene el total de los datos se llama PDB, que contiene los dos grandes grupos de complejos obligados y transientes, cada uno de estos últimos contiene directorios con nombres de cuatro caracteres alfa numéricos con los datos a trabajar, fort.19 y fort.20 son las estructuras de las proteínas del complejo, cuyo nombre es estático para todos los complejos, y el archivo de extensión "txt"que contiene el resultado de la ejecución del software `fastcontact` explicado previamente.

A continuación se pueden ver los pseudo-códigos de los algoritmos del estudio, en ellos están los ciclos mas importantes, detalles mas específicos como llamado de funciones están disponibles en el apéndice Algoritmos, en los comentarios. Para este cálculo, se crea un archivo que contiene las características de la relación entre aminoácidos y ademas por cada aminoácido se agregarán los datos de:

[1]La licencia BSD es la licencia de software otorgada principalmente para los sistemas BSD (Berkeley Software Distribution), un tipo del sistema operativo Unix-like. Es una licencia de software libre permisiva como la licencia de OpenSSL o la MIT License. Esto está en contraste con las licencias copyleft, que tienen de reciprocidad requisitos de compartir-igual. Esta licencia tiene menos restricciones en comparación con otras como la GPL estando muy cercana al dominio público. La licencia BSD al contrario que la GPL permite el uso del código fuente en software no libre. La versión original ya se ha revisado y sus variantes son denominadas licencias BSD modificadas. Es muy similar en efectos a la licencia MIT.

- Distancia máxima entre aminoácidos.
- Número de átomo de la primera proteína al que corresponde la medición.
- Nombre de identificación del átomo de la primera proteína.
- Número de átomo de la segunda proteína al que corresponde la medición.
- Nombre de identificación del átomo de la segunda proteína.
- Distancia mínima entre aminoácidos.
- Número de átomo de la primera proteína al que corresponde la medición.
- Nombre de identificación del átomo de la primera proteína.
- Número de átomo de la segunda proteína al que corresponde la medición.
- Nombre de identificación del átomo de la segunda proteína.

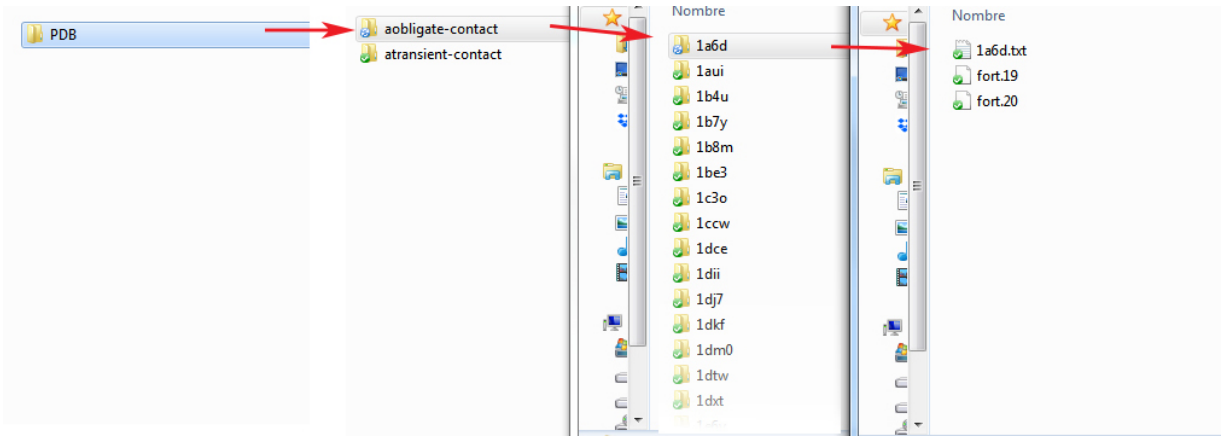

Figura 4.1: Directorio de trabajo de algoritmos.

Algoritmo 1: Cálculo de distancias entre átomos

```
1  Entrada: Elementos que contienen los datos de proteínas e información del complejo
     proteico. Figuras 4.2 y 4.3
2  Salida: Distancias calculadas del complejo proteico. ver figura 4.4
3  archivos=0 nos sirve para recorrer los directorios.
4  archivo=null almacena los archivos del directorio.
5  for cada directorio que contenga archivos de un complejo proteico do
6      abrir archivo fort.19, fort.20 y archivoComplejo.txt, guardar los resultados en el
         archivo distanciasidcomplejo.txt
7      for cada elemento de la fila del archivo archivoComplejo.txt do
8          if linea = "Top 20 Min & Max receptor-ligand residue electrostatic contacts"
             then
9              guardar la posicion que corresponde a la medicion actual
10         end
11         else
12             Guardar numero de linea siguiente
13             Break
14         end
15     end
16     for posicion valida de lineas de fastcontact do
17         if linea != ————————————-z linea es distinto de "Top 20 Min & Max
             receptor-ligand residue free energy contacts" then
18             Guardar aminoacidos de ambas proteinas
19             se puede calcula r= verdadero
20         end
21     end
22     for linea valida en archivo fort.19 do
23         if linea de fort.19 == linea archivoComplejo.txt then
24             for Linea valida en archivo fort.20 do
25                 calculamos y guardamos la distancia.
26             end
27         end
28     end
29     Calcular maximo y minimo
30     almacenar estructura de archivo final en variable
31 end
32 Guardar estructura final en archivo final
```

```
ATOM      1  N   LYS E   1      28.189   5.020  62.680
ATOM      2  H   LYS E   1     ************************
ATOM      3  CA  LYS E   1      27.705   5.368  64.017
ATOM      4  CB  LYS E   1      28.187   6.757  64.418
ATOM      5  CG  LYS E   1      29.706   6.868  64.478
ATOM      6  CD  LYS E   1      30.154   8.134  65.169
ATOM      7  CE  LYS E   1      30.309   9.320  64.241
ATOM      8  NZ  LYS E   1      28.989   9.913  63.876
```

```
ATOM      1  N   GLN I   1      69.129  20.057  76.586
ATOM      2  H   GLN I   1     ************************
ATOM      3  CA  GLN I   1      69.043  18.629  76.973
ATOM      4  CB  GLN I   1      68.724  18.435  78.439
ATOM      5  CG  GLN I   1      68.584  16.995  78.896
ATOM      6  CD  GLN I   1      67.731  16.888  80.150
ATOM      7  OE1 GLN I   1      66.511  16.744  80.116
ATOM      8  NE2 GLN I   1      68.413  16.943  81.301
```

Figura 4.2: Datos entrada de proteínas para calcular distancias.

```
   3.075  241 ARG
   3.546  213 ASP
   4.372  245 HIS
   4.638  165 GLN
   6.406  246 ASN
   9.836  215 ASP
  10.677  163 GLU
Top 20 Min & Max receptor-ligand residue electrostatic contacts
  -0.313  248 GLY   35 THR
  -0.276  244 LEU   35 THR
  -0.177  216 PRO   32 TYR
  -0.094  180 LEU   64 LYS
  -0.056  216 PRO   39 GLU
  -0.031  245 HIS   68 LEU
  -0.028  244 LEU   36 LEU
  -0.027   92 PRO   35 THR
  -0.025  180 LEU   63 GLU
```

Figura 4.3: Datos entrada de archivo generado en fastcontact.

Después de ejecutar el software con el algoritmo de medición de distancias, en cada directorio del complejo proteico se generara un archivo llamado `DistanciasMM_idcomplejo.txt` que contendrá la información detallada en la figura 4.4, la figura contiene un extracto de archivo generado llamado `DIST_1a2k.txt` que contiene en las columnas de izquierda a derecha: Energía: Contribución energética de la interacción de las siguientes 4 columnas, Número aminoácido proteína A, Nombre aminoácido proteína A, Número aminoácido proteína B, Nombre aminoácido proteína B, Distancia Máxima entre los átomos que genera la energía, Número átomo proteína A, Nombre átomo proteína A, Número átomo proteína B, Nombre átomo proteína B, Distancia Mínima entre los átomos que genera la energía, Número de átomo proteína A, Nombre átomo proteína A, Número átomo proteína B, Nombre átomo proteína B.

```
Top 20 Min & Max receptor-ligand residue electrostatic contacts
  -0.313  248 GLY   35 THR    3.297  2406  CA        336  OG1       8.394  2408  O         340  OG1
  -0.276  244 LEU   35 THR    3.338  2362  CA        340  O         9.447  2365  CD1       338  O
  -0.177  216 PRO   32 TYR    3.303  2086  CB        314  OH       12.795  2089  O         304  OH
  -0.094  180 LEU   64 LYS    3.633  1755  CD2       622  O        11.596  1754  CD1       617  O
  -0.056  216 PRO   39 GLU    6.360  2087  CG        370  OE2      13.660  2089  O         372  OE2
  -0.031  245 HIS   68 LEU    6.937  2380  O         652  C        15.608  2378  CE1       650  C
  -0.028  244 LEU   36 LEU    3.943  2366  CD2       348  C        10.336  2360  N         346  C
  -0.027   92 PRO   35 THR    9.548   884  C         338  CG2      16.116   880  CD        340  CG2
  -0.025  180 LEU   63 GLU    7.221  1755  CD2       609  O        13.727  1754  CD1       607  O
  -0.022  165 GLN   66 GLY    5.313  1613  NE2       637  CA       11.551  1617  O         639  CA
  -0.014  244 LEU   37 GLY    3.544  2366  CD2       350  N         9.536  2368  O         354  N
  -0.014  215 ASP   12 GLY    8.169  2080  OD2       108  CA       15.018  2082  O         110  CA
  -0.013  163 GLU    9 VAL   13.423  1597  OE1        88  CG1      19.348  1591  N          91  CG1
  -0.012   82 VAL   70 ASP   14.440   796  C         671  N        19.127   790  N         679  N
  -0.012  214 GLU   12 GLY   11.552  2064  N         108  CA       16.885  2070  OE1       106  CA
  -0.012  165 GLN    9 VAL   13.683  1613  NE2        87  CB       21.911  1617  O          84  CB
  -0.011  180 LEU   65 PHE    4.450  1755  CD2       634  O         9.466  1749  N         630  O
  -0.010  180 LEU   62 GLN    8.243  1755  CD2       599  O        16.781  1749  N         595  O
  -0.010  181 PRO   12 GLY    5.581  1761  CB        110  O         8.039  1758  N         109  O
  -0.010  216 PRO   27 GLU   11.833  2087  CG        253  OE2      19.478  2089  O         246  OE2
 -------------------------
   0.644  246 ASN   71 GLY    2.815  2386  OD1       680  N         7.700  2387  ND2       684  N
   0.649   69 ASP   69 ARG    7.680   681  OD2       666  NH2      18.764   683  O         670  NH2
   0.658  165 GLN  100 ASP    7.483  1613  NE2       987  OD2      17.168  1617  O         989  OD2
   0.677  163 GLU   62 GLN    6.096  1595  CG        599  O        13.538  1591  N         594  O
   0.705  217 ILE   32 TYR    4.474  2098  O         314  OH       15.361  2096  CD        304  OH
   0.718   85 GLN   35 THR    4.547   818  NE2       338  CG2      13.615   822  O         332  CG2
   0.802  215 ASP   59 THR    6.238  2079  OD1       576  O        13.019  2082  O         574  O
   0.846  245 HIS   69 ARG    3.469  2380  O         657  CB       12.127  2378  CE1       670  CB
   0.856  247 PHE   35 THR    3.354  2403  O         336  OG1      13.054  2401  CZ        332  OG1
   0.876  213 ASP   64 LYS    3.309  2061  OD2       617  NZ       13.299  2063  O         622  NZ
   0.906  215 ASP   62 GLN    5.815  2080  OD2       588  N        12.005  2082  O         599  N
   0.914  244 LEU   69 ARG    2.738  2368  O         660  NE       10.108  2360  N         670  NE
   0.977  245 HIS   35 THR    3.043  2369  N         340  O         8.433  2377  NE2       332  O
   1.128  215 ASP   64 LYS    2.757  2080  OD2       617  NZ       13.546  2082  O         622  NZ
```

Figura 4.4: Extracto de archivo generado llamado DIST_1a2k.txt

Algoritmo 2: Resumen distancias de complejos por tipo

1 **Entrada:** Elementos con las distancias de los complejos. Dos archivos similares al de la figura 4.4 correspondientes a transitorio y permanente.

2 **Salida:** resumen de las distancias por tipos de complejos. EN la figura 4.5 se muestra los datos resúmenes de datos.

3 archivos=0 nos sirve para recorrer los directorios.

4 archivo=null almacena los archivos del directorio.

5 **for** *cada directorio que contenga archivos de un complejo proteico* **do**

6 abrir archivo distanciaComplejo, guardar los resultados en el archivo resumendistanciasidcomplejo

7 **for** *cada elemento de la fila de distanciasComplejotransitorio* **do**

8 guardar linea en resumendistanciasidcomplejo

9 **end**

10 cerrar archivo transientes **for** *cada elemento de la fila de distanciasComplejotransitorio* **do**

11 guardar linea al final de resumendistanciasidcomplejo

12 **end**

13 **end**

14 guardar archivo distancias Complejo transitorio y permanentes en formato arff.

1	complex T : 1a14	-0.816	102 ARG	249 ASN	6.050	977	HH11	2420	OD1	13.690	968	H	2425	OD1	-0.711	147 SER	251 P
2	complex T : 1a2k	-0.313	248 GLY	35 THR	3.297	2406	CA	336	OG1	8.394	2408	O	340	OG1	-0.276	244 LEU	35 T
3	complex T : 1acb	-0.657	212 GLY	35 PRO	3.098	1903	O	343	CA	7.339	1899	N	342	CA	-0.315	211 TRP	36 V
4	complex T : 1agr	-0.277	179 GLY	33 GLU	3.125	1738	CA	319	O	8.341	1740	O	317	O	-0.044	205 LYS	73 A
5	complex T : 1ahw	-0.516	94 SER	155 LYS	3.774	905	OG	1555	C	7.599	907	C	1556	C	-0.272	92 GLY	157 A
6	complex T : 1ak4	-4.151	55 ARG	90 PRO	1.793	518	HH22	833	O	11.351	504	N	830	O	-0.830	71 ASN	87 H
7	complex T : 1akj	-0.073	225 THR	29 PRO	6.582	2278	O	286	C	12.469	2274	OG1	282	C	-0.067	134 THR	7 P
8	complex T : 1ao7	-0.110	377 PHE	30 GLN	4.256	3797	C	264	OE1	12.035	3795	CE2	258	OE1	-0.026	57 PRO	26 A
9	complex T : 1ar1	-0.250	694 PRO	100 GLU	6.085	6484	C	972	OE1	12.774	6480	CD	975	OE1	-0.188	769 GLU	27 P
10	complex T : 1aro	-0.364	4 PHE	266 ASP	3.476	40	CA	2576	O	9.567	49	O	2573	O	-0.330	3 GLN	267 V
11	complex O : 1a6d	-9.711	32 ASP	496 ARG	1.833	298	H	4600	O	8.628	305	O	4591	O	-9.588	253 GLU	228 L
12	complex O : 1au1	-0.756	327 PRO	126 GLN	3.604	3212	CA	1244	NE2	9.718	3211	CD	1237	NE2	-0.135	327 PRO	130 L
13	complex O : 1b4u	-0.414	6 TYR	157 PRO	3.414	55	OH	1438	CB	12.565	45	N	1441	CB	-0.336	119 GLY	202 G
14	complex O : 1b7y	-1.450	40 PRO	581 GLU	3.169	351	CA	5489	OE2	9.641	355	O	5491	OE2	-1.293	178 PRO	461 G
15	complex O : 1b8m	-0.353	64 ILE	65 GLY	3.571	585	CA	570	O	8.484	591	O	566	O	-0.291	63 GLY	66 V
16	complex O : 1be3	-0.508	624 ASP	170 PRO	4.347	5929	OD1	1609	CB	8.466	5930	OD2	1607	CB	-0.389	831 THR	241 I
17	complex O : 1c3o	-1.405	258 ASP	357 PRO	3.187	2370	OD2	3300	CA	9.022	2364	N	3304	CA	-0.461	265 ARG	359 P
18	complex O : 1ccw	-0.357	66 GLN	122 LEU	3.297	586	OE1	1214	CA	9.980	590	C	1217	CA	-0.168	97 GLY	333 G
19	complex O : 1dce	-0.654	225 GLN	231 PRO	3.580	2250	NE2	2162	CA	9.375	2254	O	2166	CA	-0.561	478 PRO	329 S
20	complex O : 1dii	-0.309	85 ARG	50 PRO	4.242	778	NH1	466	CB	10.868	785	O	469	CB	-0.245	38 TYR	53 P

Figura 4.5: Salida datos complejos permanentes y transitorios.

4.1. Software aprendizaje automático

Para estudiar las matrices de características, se utilizó el software WEKA (Waikato Environment for Knowledge Analysis), que significa entorno para análisis del conocimiento de la Universidad de Waikato, y es un una herramienta desarrollada en Java que contiene una colección de algoritmos de aprendizaje automático para el estudio de tareas de minería de datos [15]. Para concer más de WEKA con ejemplos prácticos se usaron materiales de [30],[12],[29],[8],[18] Para usar el software recientemente mencionado se recomienda empezar por estas lecturas [23] y [2], necesitamos confeccionar un archivo en formato arff (Attribute-Relation File Format), este archivo es un conjunto de datos con ciertas características, que se dividen en tres principales:

I.- **Encabezado:** Se define el nombre de la relación. Su formato es el siguiente:

`@relation <nombre-de-la-relación>`

Donde `<nombre-de-la-relación>` es de tipo String. Si dicho nombre contiene algún espacio será necesario expresarlo entrecomillado.

II.- **Declaraciones de atributos:** En esta sección se declaran los atributos que compondrán nuestro archivo junto a su tipo. La sintaxis es la siguiente:

`@attribute <nombre-del-atributo> <tipo>` Donde `<nombre-del-atributo>`es de tipo String. teniendo las mismas restricciones que el caso anterior. Weka acepta diversos tipos, estos son:

- **NUMERIC:** Expresa números reales.
- **INTEGER:** Expresa números enteros.
- **DATE:** Expresa fechas, para ello este tipo debe ir precedido de una etiqueta de formato entrecomillada. La etiqueta de formato está compuesta por caracteres separadores (guiones y/o espacios) y unidades de tiempo.
- **STRING:**Expresa cadenas de texto, con las restricciones del tipo String comentadas anteriormente.
- **ENUMERADO:** El identificador de este tipo consiste en expresar entre llaves y separados por comas los posibles valores (caracteres o cadenas de caracteres) que puede tomar el atributo. Por ejemplo, si tenemos un atributo que indica el tiempo podría definirse:

```
@attribute tiempo soleado,lluvioso,nublado
```

III.- **Sección de datos:** Declaramos los datos que componen la relación separando entre comas los atributos y con saltos de línea las relaciones.

```
@data

4,3.2
```

Aunque éste es el modo "completo.es posible definir los datos de una forma abreviada (sparse data). Si tenemos una muestra en la que hay muchos datos que sean 0 podemos expresar los datos prescindiendo de los elementos que son nulos, rodeando cada una de las filas entre llaves y situando delante de cada uno de los datos el número de atributo.

4.2. Carga de datos en formato arff

Se ha explicado como se estructuran estos archivos(la figura 4.6 muestra extracto del archivo con la estructura de 3 items descrita en 4.1), a continuación se confeccionará un archivo que transforme los resúmenes mostrados anteriormente para crear la estructura compatible para cargar el software WEKA.

A continuación se explica como se confeccionó la matriz arff:

Algoritmo 3: Crear matriz arff con complejos transientes y obligados por separado

```
1  Entrada: Resumen complejos transientes y obligados. Ver figura 4.5
2  Salida: matriz arff complejos y transitorios. Ver figura 4.6.
3  cabecera arff: Transientes
4  seccion atributos arff:
5  for para cada columna de atributos do
6  |  escribimos los identificadores de cada columna junto al número de ciclo para evitar
   |   duplicados (multiplos de 15)
7  end
8  seccion datos arff:
9  largoO= largo archivo resumen complejos obligados
10 largoT= largo archivo resumen complejos Transientes
11 for 0 hasta largoT do
12 |  for 0 a 80 relaciones de cada complejo proteico do
13 |  |  guardar los datos obtenidos de archivo resumen transientes, agregar una coma
   |  |   entre cada dato para que sea compatible con WEKA
14 |  end
15 end
16 cabecera arff: Obligados
17 for para cada columna de atributos do
18 |  Escribir los identificadores de cada columna junto al número de ciclo para evitar
   |   duplicados (multiplos de 15)
19 end
20 guardar archivo arff transitorios
21 guardar archivo arff obligados
```

Luego de ejecutar el algoritmo anterior, obtendremos dos matrices, una **ComplexObligate.arff** y **ComplexTransient.arff**. Estas matrices de datos contienen en cada una de sus filas la información del complejo, identificado en la primera columna, luego la cantidad de 80x15 columnas que corresponden a los datos específicos de las distancias calculadas. En total tendremos el número total de complejos (296) en filas, y 80x16 columnas que identifican cada columna, en este caso incluyendo la primera.

En el siguiente capítulo mostraremos el trabajo de estas matrices generadas en el software WEKA.

```
1202  @attribute nombreAtom_Dmax_A79 {C,CA,CB,CE,CE1,CD,CD1,CD2,CG,CG1,CZ,H,O,OD1,OD2,OE1,OE2,OG,OG1,
1203  @attribute numeroAtom_Dmax_B79 NUMERIC
1204  @attribute nombreAtom_Dmax_B79 {C,CA,CB,CE,CE1,CD,CD1,CD2,CG,CG1,CZ,H,O,OD1,OD2,OE1,OE2,OG,OG1,
1205  @data
1206  transien,1a14 , -0.816 , 102, ARG,  249, ASN,    6.050,   977,   HH11 ,    2420,   OD1 ,     13
1207  transien,1a2k , -0.313 , 248, GLY,   35, THR,    3.297,  2406,   CA   ,     336,   OG1 ,      8
1208  transien,1acb , -0.657 , 212, GLY,   35, PRO,    3.098,  1903,   O    ,     343,   CA  ,      7
1209  transien,1agr , -0.277 , 179, GLY,   33, GLU,    3.125,  1738,   CA   ,     319,   O   ,      8
```

Figura 4.6: Salida datos complejos permanentes y transitorios en formato arff.

Capítulo 5

Análisis de datos

5.1. Introducción a WEKA

WEKA (Waicato Environment for Knowledge Analysis) es un entorno para experimentación de análisis de datos que permite aplicar, analizar y evaluar las técnicas más relevantes de análisis de datos, principalmente las provenientes del aprendizaje automático, sobre cualquier conjunto de datos del usuario. Para que esto suceda sólo se requiere que los datos a analizar se encuentren almacenados en un formato conocido como ARFF (Attribute - Relation File Format). La herramienta permite cargar los datos en tres soportes: fichero de texto, acceso a una base de datos y acceso a través de Internet sobre una dirección URL de un servidor Web.

Es una conocida suite de software para el aprendizaje y la máquina que soporta varias tareas de minería de datos típicos, especialmente los datos del proceso previo, el agrupamiento, clasificación, regresión, visualización y selección de características. Sus técnicas se basan en la hipótesis de que los datos están disponibles en un único archivo plano o una relación, donde se etiqueta cada punto de datos por un número fijo de atributos. WEKA proporciona acceso a bases de datos SQL utilizando Java Database Connectivity y puede procesar el resultado devuelto por una consulta de base de datos.

Con el objeto de facilitar su uso por un mayor número de usuarios, WEKA además incluye una interfaz gráfica de usuario para acceder y configurar las diferentes herramientas integradas. Se distribuye como software de libre distribución desarrollado en Java. Está constituido por una serie de paquetes de código abierto. Estos paquetes pueden ser integrados en cualquier proyecto de análisis de datos e incluso pueden extenderse con contribuciones de los usuarios que desarrollen nuevos algoritmos, es por ello que WEKA está diseñado como una herramienta orientada a la extensibilidad por lo que añadir nuevas funcionalidades es una tarea sencilla.

La versión original de Weka fué un front-end para modelar algoritmos implementados en otros lenguajes de programación, más unas utilidades para pre procesamiento de datos desarrolladas en C para hacer experimentos de aprendizaje automático. Esta versión original se diseñó inicialmente como herramienta para analizar datos procedentes del dominio de la agricultura,pero la versión más reciente basada en Java (WEKA 3), que empezó a desarrollarse en 1997, se utiliza en muchas

y muy diferentes áreas, en particular con finalidades docentes y de investigación [33].

Ventajas de usar WEKA

- Se distribuye bajo GPL GNU.[1]
- Portable y multiplataforma.
- Contiene una extensa colección de técnicas para pre procesamiento de datos y modelado.

5.2. Datos en WEKA

Como se ha mencionado, WEKA es un software que permite realizar múltiples estudios con distintos tipos de datos, es un software que se emplea para la minería de datos y aprendizaje automático, ya que permite generar datos no explícitos de los que si lo están, esto se logra mediante un análisis del conjunto de datos, pués entregan nuevas características de estudio.

Lo primero a destacar al ingresar los datos generados a WEKA, almacenados previamente en el archivo `ComplexTO.arff`, es la cantidad de aminoácidos más comunes en los distintos complejos proteicos, y respecto a las distancias mínimas y máximas entre aminoácidos podemos ver lo mismo pero con los átomos, y en general, se observa que las mínimas y máximas distancias se encuentran entre los CA[2], CB[3], C [4], O[5]. Estos datos son obtenidos al cargar el archivo arff, y ver la sección de vista previa. Este es un patrón repetido en distintos pares de aminoácidos medidos, provenientes de los grupos "20 máximas y mínimas energías de contactos de residuos electrostáticosz "20 máximas y mínimas energías de contactos de energía libre".

En la figura 5.1 a continuación hay un histograma con las distancias que muestra WEKA en ciertos rangos, es como una especie de discretización de valores por defecto del software, donde los colores corresponden a los átomos participantes y la amplitud del color corresponde a las veces que se repite dentro de ese rango de medición.

El anterior es un ejemplo de los datos representados en el software, en primera instancia se describirán los datos cargados inicialmente, la información básica que se encontrará ahí, luego se aplicará un filtro de preprocesado que permitirá realizar ciertas clasificaciones, y finalmente se mostrarán los datos mas relevantes obtenidos de este conjunto de datos. Esto lo continuaremos después de definir algunas secciones del software que vienen a continuación:

5.2.1. Generalidades de WEKA

En esta sección se nombrarán algunos de los items importantes de WEKA. Bueno para trabajar con WEKA es necesario mencionar los dos grandes grupos con las principales tareas de minería de datos [4]:

(a) Técnicas Predictivas:

[1] La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License (o simplemente sus siglas en inglés GNU GPL) es una licencia de derecho de autor ampliamente usada en el mundo del software libre y código abierto.

[2] CA: Carbono Alfa.

[3] CB: Carbono Beta

[4] C:Carbono

[5] O:Oxígeno

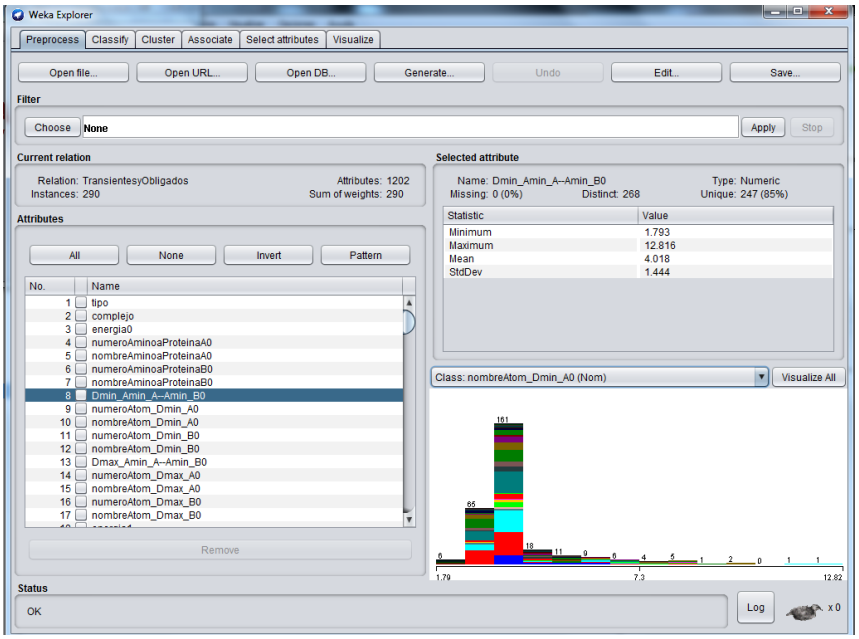

Figura 5.1: Captura de pantalla de software WEKA con los datos generados.

- Regresión y series temporales
- Análisis Discriminante
- Métodos Bayesianos
- Algoritmos Genéticos
- Árboles de decisión
- Redes neuronales

(b) Técnicas Descriptivas:

- Clustering y segmentación
- Escalamiento
- Reglas de asociación y dependencia
- Análisis exploratorio
- Reducción de la dimensión

Con los algoritmos confeccionados anteriormente, se obtuvieron los datos en el formato arff para ingresarlos al WEKA, con éste software de aprendizaje automático se podrá clasificar, agrupar (clustering), aplicar regresión, y reglas de asociación, entre muchas más. En el software existen distintas opciones para trabajar con estos datos.

- **Clasificación:** Esta tarea es muy importante, ya que principalmente con ella podremos encontrar alguna relación o patrón en los datos estudiados con el fin de predecirlos y comprenderlos más.
- **Agrupamiento:** El objetivo de esta tarea es buscar los datos con mayor similitud y que además son diferentes al resto de los datos.
- **Regresion:** El objetivo de esta tarea predictiva es identificar funciones para modelar lo

más perfectamente posible los datos originales, para que la predicción sea la óptima.
- **Asociación:** Esta tarea predictiva esta enfocada en encontrar los algoritmos que predigan las relaciones entre las variables o atributos.

5.2.2. Preprosesamiento de los datos

El primer paso para un análisis de datos con WEKA es el preprocesamiento de los mismos en la pestaña preprocess. Esta operación se lleva a cabo mediante el uso de filtros, que pueden ser aplicados a los atributos o a las instancias. En general, el tipo de filtro es no supervisado, esto es, el resultado obtenido es independiente del tipo de algoritmos que se utilice posteriormente.

5.2.3. Clasificación

Existe interés en encontrar patrones de comportamientos entre los datos, por lo que necesitaremos clasificarlos. Esta tarea es la más usada en la minería de datos. El objetivo será el de encontrar relaciones entre los atributos que permitan saber cuáles son las posibilidades de que el equipo seleccionado quede en un determinado lugar de la tabla clasificatoria. Estos son los métodos principales para clasificar los datos:
- **Bayes:** son métodos que intentan encontrar entre todas las hipótesis la más probable, a partir de un conjunto de entrenamiento. El algoritmo más utilizado en este apartado es el de NaiveBayes.
- **Funciones:** se corresponden con los métodos que están basados en modelos matemáticos, como por ejemplo: las redes neuronales, o los diferentes tipos de regresiones.
- **Lazy:** en este tipo de algoritmos, cada una de las instancias se compara con el resto del conjunto de datos, definiéndose una "medida de distancia", son métodos donde el objetivo es "encontrar al vecino más cercano"
- **Meta:** son los métodos que se obtienen al combinar distintos tipos de aprendizaje.
- **Trees:** métodos expresados a través de árboles de decisión. En este caso se construye un árbol desde la raíz hasta las hojas, de tal manera que las ramas se dividen en función de los valores que toman los atributos.
- **Rules:** son algoritmos que se expresan a través de reglas y que tienen la particularidad de ser autoaprendizajes.
- **ZeroR:** La línea de base para los problemas de clasificación y regresión se denomina algoritmo de regla cero.

Además de elegir el tipo de método a usar,también existe la posibilidad de elegir el tipo de validación del modelo, que puede ser:
- **Use training set:** con esta opción el programa utilizará el método elegido con todos los datos disponibles y luego realizará una evaluación sobre los mismos datos.
- **Supplied test set:** podemos realizar una evaluación sobre un conjunto de datos que hemos elegido previamente, que normalmente serán distintos a los datos del aprendizaje.
- **Cross-validation:** la evaluación se realizará mediante una técnica de validación cruzada, cuyo objetivo es asegurarse de que los análisis estadísticos realizados son independientes. De todas las posibilidades, esta opción es la que más tiempo. Con Folds se puede elegir

el número de evaluaciones que deseamos llevar a cabo, dividiendo el conjunto de datos en datos de prueba y datos de entrenamiento computacional consume.

- **Percentage split:** en esta última opción podemos definir un porcentaje con el que aprende el modelo, haciéndose la evaluación con los datos restantes.

5.3. Entorno de trabajo con WEKA

Ya mencionados estos conceptos se continuará con la carga de datos inicial de WEKA, se espera que con el detalle de estos pasos se facilite a cualquier lector el uso del software, aunque sea superficialmente, ya que se pueden hacer actividades muy complejas con el. Al cargar inicialmente nuestro archivo `ComplexTO.arff`, se puede observar al seleccionar la columna[6] tipo (transiente u obligado), la cantidad de elementos transientes y obligados en dos columnas5.2.

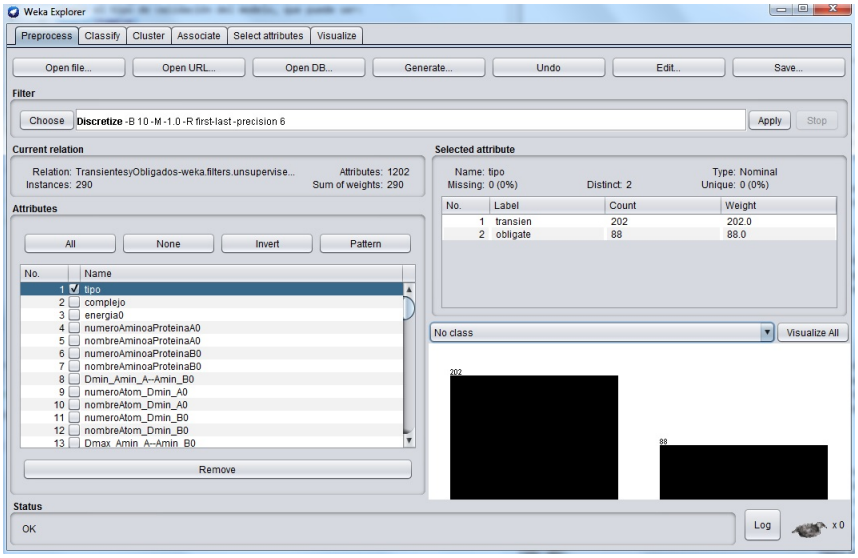

Figura 5.2: Captura de pantalla de software WEKA con elementos transientes y obligados.

Para poder clasificar mejor los datos, se discretizará las columnas en el menú

`preprocess->filter->choose`.

Aquí se verán las categorías mencionadas anteriormente acerca de los métodos supervisados y no supervisados, el método que se usará se encuentra en la sección no supervisados, en la sección attribute y luego en el filtro Discretize.

`unsupervised->attribute->discretize`.

Una vez realizado esto, nos vamos a la sección derecha de la interfaz, encontraremos un menú desplegable en el cual nos mostrará nuevamente los atributos de nuestro conjunto de datos.

[6]La columna aparece como fila en esta sección.

5.3.1. Items archivo arff

Para proceder y entender los datos que escogeremos describiremos sus ítem:

- **tipo:** es de tipo Nominal y tiene dos estados transien, obligate, para representar los complejos transientes y obligados.
- **complejo:** es de tipo `String` y contiene los nombres de los complejos, los complejos no se repiten.

  Los atributos siguientes se repiten 80 veces(0 a 79), dado que cada fila del complejo proteico se convirtió en una fila para estudiar mejor sus distancias basándose en la energía producida. Los primeros 40 conjuntos de atributos corresponden a `residuos de contacto electrostático` de izquierda a derecha estan desde los 20 que menos aportan energía al complejo hasta los 20 que más energía aportan al complejo. Los siguientes 40 conjuntos corresponden a lo mismo pero de `residuos de contactos de energía libre`
- **energia0:** Energía (0+1) de 20–
- **numeroAminoaProteinaA0:** Número de aminoácido de la proteina A.
- **nombreAminoaProteinaA0:** Nombre de aminoácido de la proteina A.
- **numeroAminoaProteinaB0:** Número de aminoácido de la proteina B.
- **numeroAminoaProteinaB0:** Nombre de aminoácido de la proteina B.
- **Dmin_Amin_A–Amin_B0:** Distancia mínima entre los aminoácidos anteriores.
- **numeroAtom_Dmin_A0:** Número del átomo del aminoácido de la proteína A anterior con el que se aplicó la distancia mínima.
- **nombreAtom_Dmin_A0:** Nombre del átomo del aminoácido de la proteína A anterior con el que se aplicó la distancia mínima.
- **numeroAtom_Dmin_B0:** Número del átomo del aminoácido de la proteína B anterior con el que se aplicó la distancia mínima.
- **nombreAtom_Dmin_B0:** Nombre del átomo del aminoácido de la proteína B anterior con el que se aplicó la distancia mínima.
- **Dmax_Amin_A–Amin_B0:** Distancia mínima entre los aminoácidos anteriores.
- **numeroAtom_Dmax_A0:** Número del átomo del aminoácido de la proteína A anterior con el que se aplicó la distancia máxima.
- **nombreAtom_Dmax_A0:** Nombre del átomo del aminoácido de la proteína A anterior con el que se aplicó la distancia máxima.
- **numeroAtom_Dmax_B0:** Número del átomo del aminoácido de la proteína B anterior con el que se aplicó la distancia máxima.
- **nombreAtom_Dmax_B0:** Nombre del átomo del aminoácido de la proteína B anterior con el que se aplicó la distancia máxima.

Ya explicado cada atributo, podemos continuar con nuestros datos.

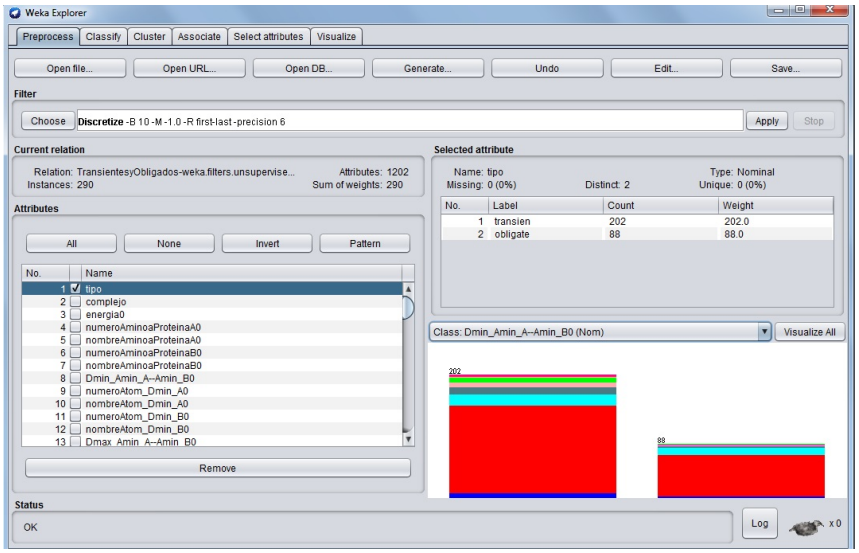

Figura 5.3: Captura de pantalla de software WEKA con elementos de distancia mínima

Al seleccionar el atributo `tipo` en la izquierda y en la parte derecha el atributo `Dmin_Amin_A-Amin_B0` podemos observar los grupos transientes y obligados divididos en secciones de colores5.3, cada uno de estos colores muestra donde están acumulados los intervalos de distancias mínimas más comunes. Si observamos estas columnas inclinando la cabeza a la izquierda, podemos comprender como estan representados en este grafico los que se muestran en la figura 5.4.

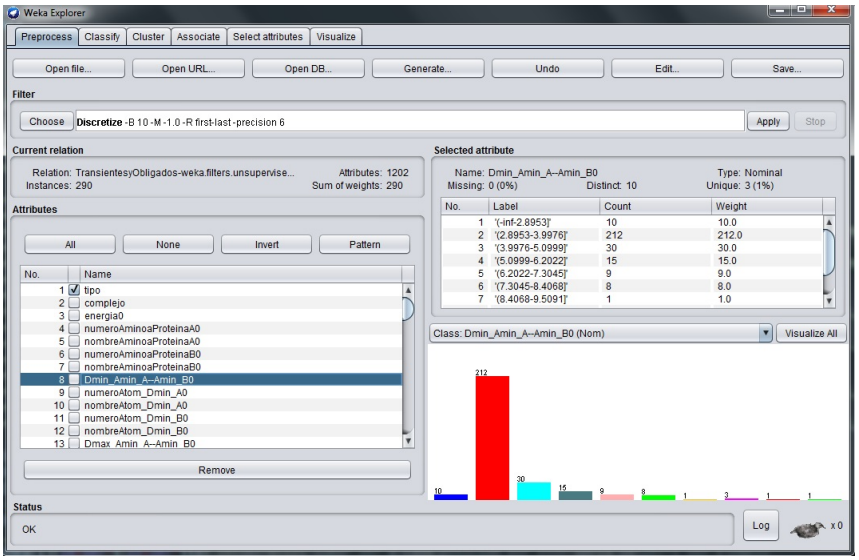

Figura 5.4: Captura de pantalla de software WEKA con elementos de distancia mínima individualmente.

De las tablas anteriores podemos concluir que las distancias mínimas aplicadas están en su mayoría entre los intervalos 2.8 y 3.9 aproximadamente, en la primera de "Top 20 Min receptor-ligand residue electrostatic contacts".

Si continuamos para observar los átomos correspondiente a la menor energia del grupo recientemente nombrado veremos lo siguiente 5.5:

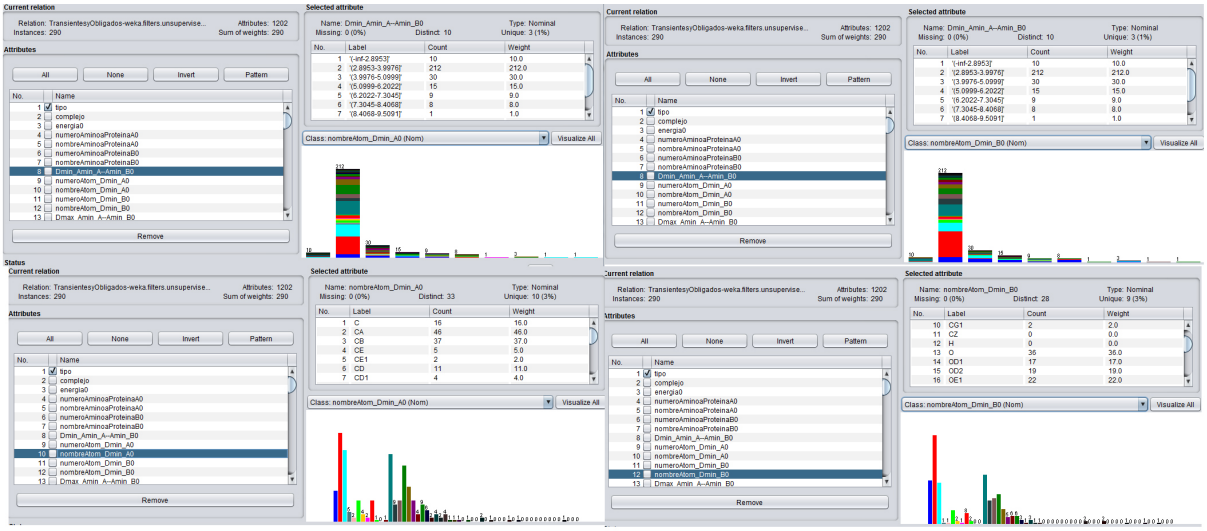

Figura 5.5: Captura de pantalla de software WEKA con átomos correspondientes a la distancia mínima.

De lo anterior se puede concluir que en esta distancia mínima está medida principalmente en los átomos CA, CB, O y OE1. todos ellos pertenecientes al aminoácido de la proteína A del complejo. Algo similar pasa en el segundo aminoácido, al tener casi los mismos átomos de la medición, C, CA,CB y OE1 entre los más cercanos en este aminoácido.

Ahora veremos que pasa con la distancia máxima basándonos en la misma estructura para reflejar alguna diferencia importante si es que la hubiera. En la figura 5.6 vemos inmediatamente que la distribución entre las distancias máximas obtenidas tiene una dispersión mas uniforme, lo que significa que las distancias máximas son más variadas que las distancias mínimas de esta energía.

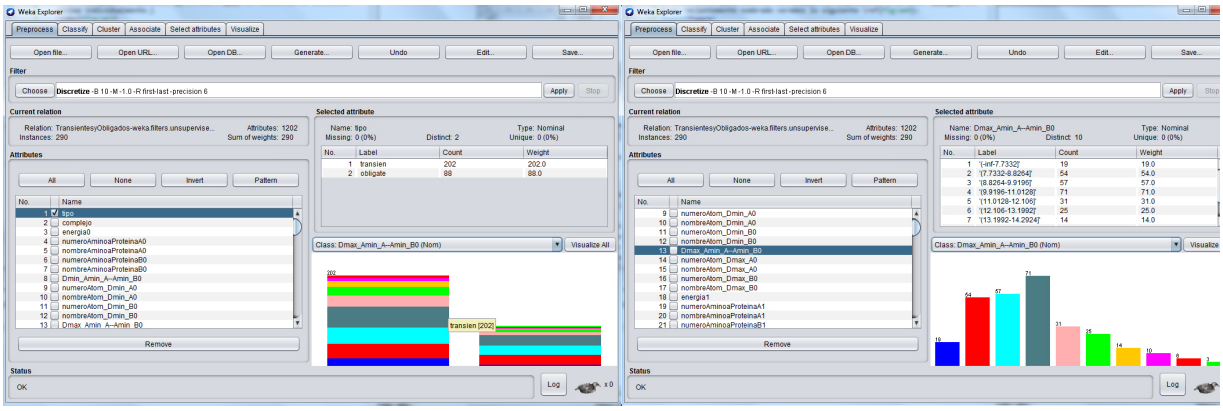

Figura 5.6: Captura de pantalla de software WEKA con elementos de distancia máxima

Queda ver que pasa con los átomos, ¿serán los mismos que en la distancia mínima? Para ello miraremos la figura 5.7, vemos que en la distancia máxima del átomo del aminoácido de la proteína A es diferente a lo que vimos en la distancia mínima, donde predominaban los átomos CA, CB, O y OE1, en este ejemplo vemos que predominan los átomos O, N, pero en los átomos del animoácido de la proteína B, predominan los átomos similares a la distancia mínima que son C, CA, CB y O.

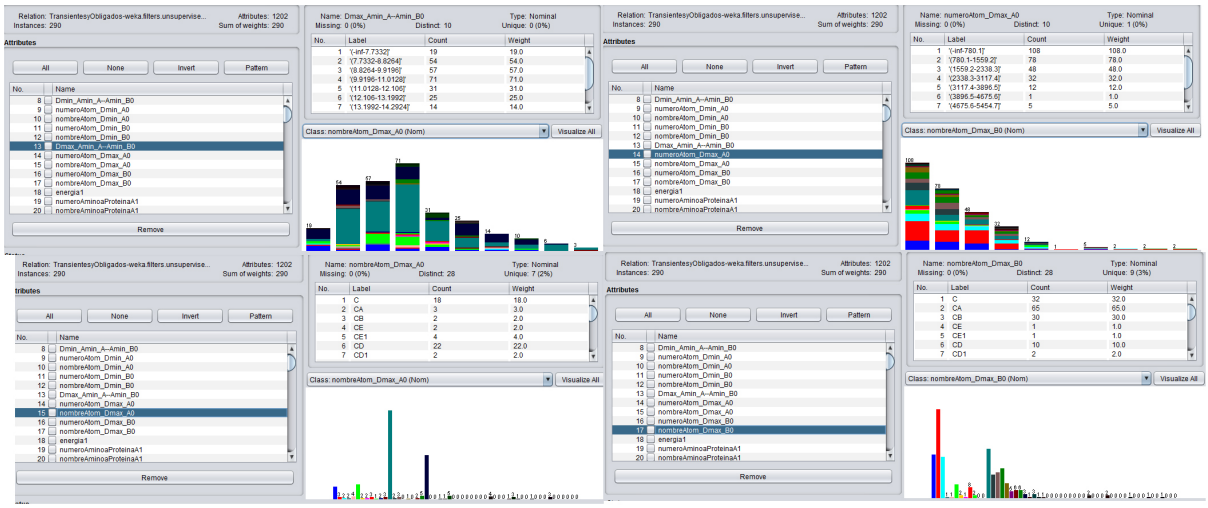

Figura 5.7: Captura de pantalla de software WEKA con los átomos de distancia máxima

Siguiendo con el análisis interno para no mostrar tantos gráficos, se comparó hasta el atributo terminado en 19, correspondiente a la energía máxima del grupo "Top 20 Min receptor-ligand residue electrostatic contactsz observamos que existe una distribución totalmente uniforme y balanceada de las distancias, átomos y rangos como veremos en la figura 5.8.

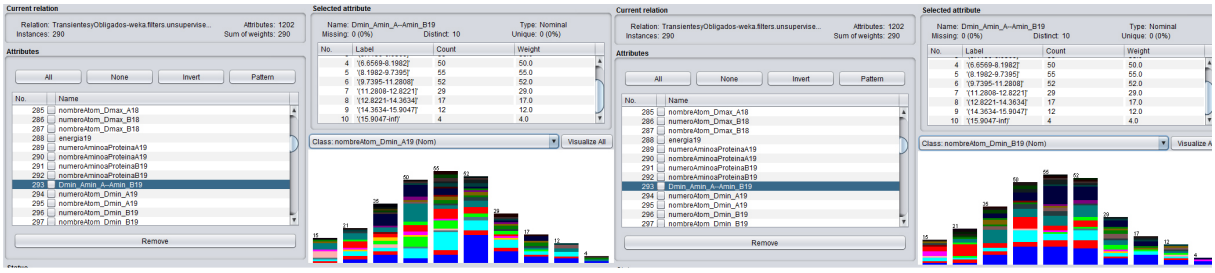

Figura 5.8: Captura de pantalla de software WEKA con los átomos de distancia mínima con más energía.

Y esto sucede con las distancias máximas de la máxima energia del mismo grupo 5.9.

Correctly Classified Instances	202	69.6552 %
Incorrectly Classified Instances	88	30.3448 %

Cuadro 5.1: Resultados de clasificación por Naive Bayes Multinomial Text, usando filtro tipo transientes, obligados

TP Rate	FP Rate	Precision	Recall	FMeasure	MCC	ROC Area	PRC Area	Class
1,000	1,000	0,697	1,000	0,821	?	0,500	0,697	transien
0,000	0,000	?	0,000	?	?	0,500	0,303	obligate
0,697	0,697	?	0,697	?	?	0,500	0,577	obligate

Cuadro 5.2: Resultados de clasificación por Naive Bayes Multinomial Text, usando filtro tipo transientes, obligados

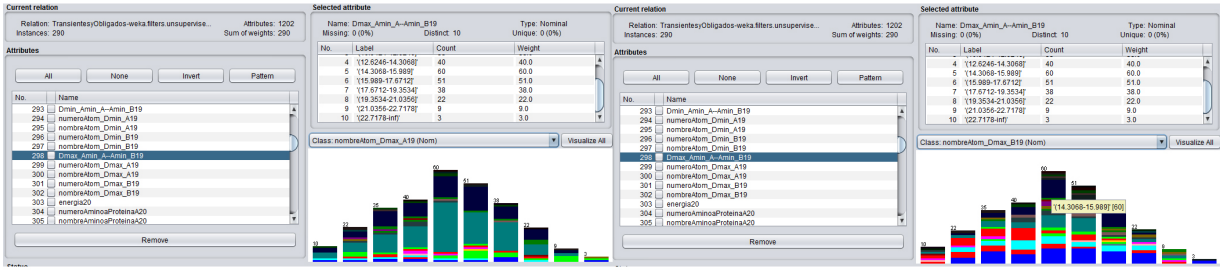

Figura 5.9: Captura de pantalla de software WEKA con los átomos de distancia máxima, con más energía de los 20 mínimos.

Cuando se sigue aplicando estas visualizaciones en los distintos atributos correspondientes a los 20 máximos, observamos variaciones interesantes. Para analizar todas estas variaciones necesitamos mucho tiempo y buena memoria para recordar el comportamiento de todas ellas, por lo cual se hace necesario estrictamente el aprendizaje automático, que permita concluir patrones que no somos capaces de analizar a simple vista.

5.3.2. Clasificadores

Para ello aplicaremos la sección de clasificadores del software WEKA. Para nuestros datos tenemos habilitados solo dos clasificadores: `NaiveBayesMultinomialText` y `ZeroR`

5.3.3. Naive Bayes Multinomial Text

Funciona directamente (y solo) en atributos de cadena. Se aceptan otros tipos de atributos de entrada, pero se ignoran durante el entrenamiento y la clasificación.

En la tabla 5.1 se aplica el filtro de clasificacion por tipo.

Correctly Classified Instances	212	73.1034 %
Incorrectly Classified Instances	78	26.8966 %

Cuadro 5.3: Resultados de clasificación por Naive Bayes Multinomial Text, complejos transientes correctamente clasificados y complejos obligados incorrectamente clasificados

Correctly Classified Instances	202	69.6552 %
Incorrectly Classified Instances	88	30.3448 %

Cuadro 5.4: Resultados de clasificación por ZeroR, usando filtro tipo transientes, obligados

En la figura 5.10 vemos el resultado de aplicar este filtro en distancias mínimas. En los datos 5.10 y en 5.3 vemos información muy similar a lo que analizamos gráficamente

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0,000	0,000	?	0,000	?	?	0,500	0,034	'(-inf-2.8953]'
1,000	1,000	0,731	1,000	0,845	?	0,500	0,731	'(2.8953-3.9976]'
0,000	0,000	?	0,000	?	?	0,500	0,103	'(3.9976-5.0999]'
0,000	0,000	?	0,000	?	?	0,433	0,046	'(5.0999-6.2022]'
0,000	0,000	?	0,000	?	?	0,500	0,031	'(6.2022-7.3045]'
0,000	0,000	?	0,000	?	?	0,500	0,028	'(7.3045-8.4068]'
0,000	0,000	?	0,000	?	?	0,500	0,003	'(8.4068-9.5091]'
0,000	0,000	?	0,000	?	?	0,500	0,010	'(9.5091-10.6114]'
0,000	0,000	?	0,000	?	?	0,500	0,003	'(10.6114-11.7137]'
0,000	0,000	?	0,000	?	?	0,500	0,003	'(11.7137-inf)'
0,731	0,731	?	0,731	?	?	0,497	0,551	

Figura 5.10: Clasificación de distancias mínimas de la energía que menos aporta en residuos de contacto electrostático.

5.3.4. ZeroR

Cuando se aplica esta clasificación disponible [8], se obtienen los mismos resultados que en 5.3.3, se puede ver en los resultados obtenidos en 5.4 y 5.5.

TP Rate	FP Rate	Precision	Recall	FMeasure	MCC	ROC Area	PRC Area	Class
1,000	1,000	0,697	1,000	0,821	?	0,500	0,697	transien
0,000	0,000	?	0,000	?	?	0,500	0,303	obligate
0,697	0,697	?	0,697	?	?	0,500	0,577	obligate

Cuadro 5.5: Resultados de clasificación por ZeroR, usando filtro tipo transientes, obligados

5.4. Implementación

Después de cargar los datos en WEKA, se observa que la mayoría de los clasificadores están deshabilitados, tal como se mencionó en la sección anterior, ante esto se mostrarán las

predicciones realizadas por el software con los clasificadores disponibles, estos son ZeroR y NaiveBayesMultinomialText.

Del total de datos, existen diferentes secciones que van desde 0 a 79, siendo las primeras 40 las distancias correspondientes a los residuos de contacto electrostático, de esos 40 los primeros 20 son los que menos energía aportan y los siguientes 20 son los que más energía aportan, las siguientes 40 secciones corresponden a los residuos de contactos de energía libre, al igual que el anterior, los 20 primeros son aquellos que menos energía aportan al complejo y los 20 restantes los que más energía aportan. Esto está detallado en 5.3.1, lo que prosigue es estudiar los resultados de aplicar el clasificador ZeroR.

5.4.1. ZeroR sin discretizar

Al utilizar la columna `tipo`, no obtenemos datos importantes, salvo la cantidad de registros que corresponden los complejos, sin obtener alguna característica propia de las distancias entre complejos transitorios o permanentes. en la figura 5.11 podemos ver resultado.

```
=== Summary ===

Correctly Classified Instances         202               69.6552 %
Incorrectly Classified Instances        88               30.3448 %
Kappa statistic                          0
Mean absolute error                      0.4233
Root mean squared error                  0.4597
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              290
```

Figura 5.11: ZeroR con respecto a los tipos de complejos.

Al utilizar la columna `Dmin_Amin_A-Amin_BO`, el resultado es mucho más interesante, ya que el algoritmo predice el valor promedio de la clase, y al escoger los atributos correspondientes al nombre del átomo, muestra el átomo que más aparece en la medición de la distancia mínima, en la parte derecha se aprecia lo mismo pero con las distancias máximas 5.6.

Línea	Átomo A	Dist. Mín	Átomo B		Átomo A	Dist. Máx	Átomo B
1	CA	4.017	CA		O	10.45	CA
20	C	8.89	C		O	15.33	C
21	O	4.76	O		O	12.8	O
40	OE1	3.03	OD2		O	11.7	OD2

Cuadro 5.6: Min & Max receptor-ligando residuo de contacto electrostático. Distancias y átomos después de aplicar el clasificador ZeroR.

En la tabla 5.7 se muestra lo mismo pero con la energía libre.

Línea	Átomo A	Dist. Mín	Átomo B		Átomo A	Dist. Máx	Átomo B
1	CB	3.56	CD1		O	9.94	CD1
20	O	5.54	O		O	12.37	O
21	O	5.06	O		O	12.97	O
40	OE2	3.05	OE2		O	11.62	OE2

Cuadro 5.7: Min & Max receptor-ligando residuo de contacto de energía libre. Distancias y átomos después de aplicar el clasificador ZeroR.

Es seguro que los números de líneas escogidos pueden generar dudas, pero la razón detrás de porqué esas líneas, es que contienen los indices energéticos extremos, esto viene dado desde la confección del archivo devuelto por `fastcontact`, cuyas líneas indican lo siguiente:

línea 1 : Corresponde a la unión que menos energía aporta.
línea 20 : Corresponde a la unión que más energía aporta del grupo que menos aporta.
línea 21 : Corresponde a la unión que menos energía aporta del grupo de las más positivas.
línea 40 : Corresponde a la unión que más energía aporta.

5.4.2. ZeroR discretizado

Para complementar los resultados anteriores, se aplicará el filtro discretizar, mencionado en 5.3.

Lo primero que se puede observar, es que al escoger el atributo tipo de complejo, no hay nuevos resultados respecto a los valores no discretizados, pero la diferencia esta al seleccionar una distancia, donde se aprecia un rango de distancias y la cantidad de mediciones que pertenecen a ese rango. En la figura 5.12 se puede ver esto. También se puede observar la predicción de la magnitud de distancia mínima, que esta situada entre los valores 2.8 y 3.9 Ångströms.

```
=== Classifier model (full training set) ===

ZeroR predicts class value: '(2.8953-3.9976]'

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.01 seconds

=== Summary ===

Correctly Classified Instances         212               73.1034 %
Incorrectly Classified Instances        78               26.8966 %
Kappa statistic                          0
Mean absolute error                      0.0928
Root mean squared error                  0.2121
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              290
```

Figura 5.12: ZeroR con distancia mínima 0 discretizada.

Línea	Átomo A	Dist. Mín	Átomo B		Átomo A	Dist. Máx	Átomo B
1	CA	2.8953-3.9976	CA		O	9.9196-11.0128	CA
20	C	8.1982-9.7395	C		O	14.3068-15.989	C
21	O	3.3266-4.3802	O		O	-inf-13.973	O
40	OE1	2.7694-3.3302	OD2		O	-inf-13.0968	OD2

Cuadro 5.8: Min & Max receptor-ligando residuo de contacto electrostático. Distancias y átomos después de aplicar el clasificador ZeroR.

A continuación se muestra una tabla 5.8, tal como en el ítem previo.

Línea	Átomo A	Dist. Mín	Átomo B		Átomo A	Dist. Máx	Átomo B
1	CB	3.387-3.7058	CD1		O	8.8844-9.8481	CD1
20	O	3.5159-4.8748	O		O	10.7986-12.6869	O
21	O	-inf-7.3358	O		O	-inf-15.4202	O
40	OE2	2.7694-3.3302	OE2		O	-inf-13.1611	OE2

Cuadro 5.9: Min & Max receptor-ligando residuo de contacto de energía libre. Distancias y átomos después de aplicar el clasificador ZeroR.

Tal como se puede observar, los valores obtenidos en la tabla son similares a los obtenidos con los datos sin discretizar, lo más importante que se puede deducir de estos valores, es que los átomos con los que se ha calculado la mínima y máxima distancia son los mismos en el rango de distancias discretizado, de esto tambien se puede concluir que los átomos ubicados en los extremos de la zona de interacción son similares.

A partir de la distribución de datos del archivo arff, se pueden realizar análisis de las energías, pero no sería un aporte para el estudio actual.

Los átomos mostrados por el algoritmo de clasificación ZeroR son aquellos que producen energías muy extremas, se pueden realizar predicciones para cada una de las energías generadas, solo bastaría con recorrer los atributos desde el 0 hasta el 79 luego de seleccionar el clasificador en WEKA.

5.4.3. NaiveBayesMultinomialText

El segundo clasificador disponible para usar con nuestros datos es NaiveBayesMultinomial-Text pero por su naturaleza, no es aplicable a nuestros datos, ya que este clasificador se utiliza para realizar text mining, en documentos de texto extensos, este no es el caso, lo que se necesita para clasificar las distancias no es solo texto

Respecto al análisis general, los resultados fueron buenos pero el enfoque de distancias máximas y mínimas reales quizás en la realidad no sea muy útil, esto solo lo podría afirmar o desmentir una persona que tenga más experiencia trabajando a este nivel biológico. Respecto a trabajos

anteriores, la tesis [30] provee de mayor información al utilizar promedios tridimensionales de las ubicaciones de los átomos.

Como trabajos futuros se puede seguir investigando con las tablas generadas, comparándolas con tablas de distintos complejos proteicos, es probable que ese estudio aporte más datos importantes.

Capítulo 6

Conclusiones

Con lo visto, entendimos la importancia del estudio de datos. En esta ocasión la minería de datos se aplicó al ámbito biológico pero puede ser aplicado a cualquier ciencia, a cualquier grupo de datos.

Respecto a la minería de datos, es una técnica usada en empresas para estudiar clientes, que después de un tiempo se ha ido expandiendo cada vez más. Aparentemente parece ser una técnica compleja de utilizar, pero con los datos técnicos necesarios es relativamente fácil aplicar estos algoritmos a cualquier oportunidad de estudio de datos que se presenten.

En nuestro caso pudimos ver toda la información escondida que estaba en grandes grupos de datos, estos volúmenes de datos actúan de forma muy parecida de como lo hace la mente humana, que basada en experiencias define los mecanismos para actuar, pero esto lleva años en una persona, sin embargo las tecnologías de la información pueden llegar a estos análisis, con los algoritmos correctos, en una pequeña fracción de tiempo.

En el estudio actual, al generar los datos con las estructuras actuales, no se logró aplicar un buen clasificador, el objetivo principal que se pretende lograr con este tipo de estudios es conocer más acerca de las interacciones entre las proteínas, la utopía en términos de aportes a este campo de estudio que se pretendía realizar, es haber encontrado algún patrón más evidente en las proteínas respecto a su distancia, lo cual no se logró, aunque si se logró un aporte en una medida menor, ya que existen átomos presentes frecuentemente a ciertas distancias y con ciertas energías, hubiese sido más concluyente haber encontrado otro clasificador que pudiera entregar resultados similares pero con otro algoritmo. Otro uso que se le puede dar a este algoritmo es como instrumento de medición de átomos para personal científico.

La nueva era de la información, en mi opinión, ya no es diversificar los softwares con los que trabajamos sino usar los datos que abundan en el mundo, grandes empresas lo hacen desde hace tiempo, nosotros poco a poco nos concientizarnos del valor de aquella información que regalamos al usar un smartphone, al usar nuestra tv, al conectarnos internet, al comprar en un supermercado con nuestro rut o conectarnos al wifi en el centro comercial . Esto me hace pensar que nadie se salva de la recolección estos datos, ni siquiera quien no los entrega, pues el circulo que roda a quienes no usan las tecnologías (muy pocos) entregan información indirectamente de

esas personas. El ejemplo anterior es para considerar lo que se nos viene a futuro como personas que trabajamos con las Tecnologías de la información.

Apéndice A

Anexo I: Algoritmos

A.1. Cálculo de distancias

Listing A.1: keywordstyle

```
# -*- coding: utf-8 -*-

from os import walk
import math
# Recorre los archivosde todas las carpetas que estan contenidas en la carpeta PDB.
for (path , ficheros , archivos ) in walk ( "./PDB") :
    for archivo in archivos:
        if archivo==(path [ -4:]+".txt" ) :
        # Dentro de la carpeta de un complejo, abre los archivos fort.19, fort.20, y "complejo".txt
        # con las estructuras de cada proteina del complejo y datos obtenidos en fastcontact, c
        # crea un archivo en el mismo directorio, un archivo con la informacion de fastcontact agregando
        # los datos de cada proteina
        FastContact= open ( path+"/"+archivo,"r+" )
        DistanciasMM=open ( path+"/Distancias_"+archivo,"w+",encoding="utf-8" )
        proteinaA = open ( path+"/fort.19", "r" )
        proteinaB = open ( path+"/fort.20" , "r" )
        archfastcont=FastContact.readlines()
        largoFC= len (archfastcont)-2
        proA=proteinaA.readlines()
```

```
20          largoProA=len (proA)-1
21          proB=proteinaB.readlines()
22          largoProB=len (proB)-1
23          proteinaA.seek( 0,0 )
24          proteinaB.seek( 0,0 )
25          # # ELIMINAR PRINT PARA EJECUTAR MUCHOS DATOS
26          print("Largo proA y proB:", largoProA,largoProB)
27          nEnergia=0
28          # Busca el comienzo de los datos de los pares de ↩
               aminoacidos que interactuan.
29          for i in range( largoFC ) :#largoFC es el largo total ↩
               del archivo fastcontact
30              if archfastcont[i]==" Top 20 Min & Max receptor-↩
                   ligand residue electrostatic contacts\n":
31                  #considerar el espacio al comienzo de la frase ↩
                       entre comillas
32                  nEnergia=i+1 #nEnergia es la linea donde estan ↩
                       las energias producidas
33                  break
34                  # Recorre el archivo FastContact desde el indice↩
                        encontrado en el bucle anterior, hasta el ↩
                       final del archivo.
35
36          for i in range ( nEnergia , largoFC ):
37          # Ignora informacion que no contienen datos de la ↩
               interaccion
38              if archfastcont [i]!= " -------------------------\n↩
                   " and
39                  archfastcont [i]!=" Top 20 Min & Max receptor-↩
                       ligand residue free energy contacts\n":
40                  # Guarda el identificador de cada aminoacido en ↩
                       dos variables.
41                  NaminoA=int ( archfastcont [i][9:13] )
42                  NaminoB=int ( archfastcont [i][19:22] )
43                  # variable booleana, si estadoA continua siendo ↩
                       True a la hora de calcular la distancia,
44                  # significa que tenemos las coordenadas de los ↩
                       dos aminoacidos para logralo, en caso ↩
                       contrario
45                  # no calculamos la distancia.
46                  estadoA=True
```

```
47                  # Recorre las lineas del archivo que contiene las coordenadas de cada aminoacido de la proteina A
48                  # hasta encontrar la ubicacion del aminoacido requerido.
49
50              #se definen valores iniciales a variables, valor minimo y maximo tienen valores aleatorios numericos sin importancia.
51              distancias = {}
52              minimo=1
53              maximo=2
54
55              for j, lineA in enumerate ( proA ):
56                  if  j<largoProA and NaminoA == int(lineA[23:26]) and lineA [30]!="*" :
57                      # FastContact maneja la falta de coordenadas con los caracteres ***, por lo tanto si nos encontramos con este caracter,
58                      # no podremos calcular la distancia, por lo tanto estadoA seria False.
59                      # mientras las coordenadas no sean ***
60                      #en este punto debemos buscar las coordenadas de b para asociarlas y medirlas
61                      for z, lineB in enumerate ( proB ):
62                          if z<largoProB and NaminoB== int(lineB[23:26]) and lineB [30]!="*" :
63                          #aqui coincide la relacion del arch fast contact
64                          #midamos la distancia entre coordenada J y coordenada Z
65                          distancia = round( math.sqrt(   (( float( lineA[30:38]) - float(lineB[30:38])) ** 2) + (( float ( lineA[38:46]) - float(lineB[38:46])) ** 2) + (( float ( lineA[46:54]) - float(lineB[46:54])) ** 2)),3 )
66                          # ELIMINAR PRINT PARA EJECUTAR MUCHOS DATOS
67                          print(distancia)
68                          distancias[distancia]=[lineA,lineB]
```

```
69                  else:
70                      pass
71                      #hemos guardado la distancia en el id de la lista, asi es mas facil aplicar funciones de max y min
72                      #por el indice y obtenemos coordenadas
73                  #aqui pondremos el valor distancia de mas abajo
74                  claves = list(distancias.keys())
75                  #minimo=0
76                  #maximo=0
77
78                  for s in range(len(distancias)):
79                      if s == 0:
80                          minimo = claves[s]
81                          maximo = claves[s]
82                      elif claves[s] < minimo:
83                          minimo = claves[s]
84                      elif claves[s] > maximo:
85                          maximo = claves[s]
86                      else:
87                          pass
88
89              # Si estadoA es igual a True, aplicamos la distancia euclidiana entre los dos aminoacidos
90              ## ELIMINAR PRINT PARA EJECUTAR MUCHOS DATOS
91              print(distancias[minimo][0][5:11])
92              temporal=archfastcont[i].rstrip('\n')+"{0:8.3f}{1:8}{2:8}{3:8}{4:8}{5:8.3f}{6:8}{7:8}{8:8}{9:8}". format(minimo, distancias[minimo][0][5:11], distancias[minimo][0][12:17],distancias[minimo][1][5:11],distancias[minimo][1][12:17],maximo,distancias[maximo][0][5:11],distancias[maximo][0][12:17], distancias[maximo][1][5:11],distancias[minimo][1][12:17])+ "\n"
93              if temporal[0:2]!=" -" and temporal[0:2]!=" T":
94                  archfastcont [i]= temporal
95              # BORRAR PARA EJECUTAR MUCHOS DATOS
96              print(archfastcont[i])
97              # BORRAR PARA EJECUTAR MUCHOS DATOS
98              print("i=",i,"------------------------------")
99          proteinaB.seek( 0,0 )
100         proteinaA.seek( 0,0 )
```

```
101         FastContact.close()
102         DistanciasMM.write( "Atom protein A - Atom protein B - ↩
               Min Dist - Atom protein A - Atom protein B - Max Dist↩
                 in    Complex :"+path[-4:]+"\n" )
103         # BORRAR PARA EJECUTAR MUCHOS DATOS
104         print("archfastcont [nEnergia -1:-1]",archfastcont [ ↩
               nEnergia-1: -1])
105         DistanciasMM.writelines( archfastcont [ nEnergia -1: ↩
               -1])
106         DistanciasMM.close()
107 proteinaA.close()
108 proteinaB.close()
```

A.2. Resúmenes Distancias de complejos

Listing A.2: keywordstyle

```
1  from os import walk
2  ## Crea dos archivos, uno para guardar todos los archivos ↩
      FastContact que
3  ## contienen los enfoques de distancia de los complejos ↩
      transitorios, y otro con los complejos permanentes.
4  fcTrans=open ( "ComplexTransient.txt","w+",encoding="utf-8" )
5  fcOblig=open ( "ComplexObligate.txt" ,"w+",encoding="utf-8" )
6
7  # Recorre los archivos de todas las carpetas que estan ↩
      contenidas en la carpeta atransient-contact.
8  for ( path , ficheros , archivos ) in walk ( "./PDB/atransient-↩
      contact" ) :
9
10     if len ( archivos ) >2:
11         nombreArch=path[-4:]+".txt"
12         archivoA = open ( path+"/DIST_"+nombreArch, "r",encoding↩
               ="utf-8" )
13         #print("Archivo A: ", archivoA)
14         lineasA=archivoA.readlines ()
15         #print("LINEAS ", lineasA)
16
17         # Inicia la fila con el nombre del complejo
18         #ttt="complexT:"+path [-4:]
19         print(ttt)
```

```
20          fcTrans.write ("complexT:"+path [-4:] )
21          #print(lineasA[2][0:9]+","+lineasA[2][9:14])#esta ↩
               instruccion me carga la primera linea de los archivos↩
                de distancia DE TODOS LOS COMPLEJOS
22          #22 43 64
23          # Aqui use un if con las condiciones pero por alguna ↩
               razon desconocida no #reconocio las condiciones e ↩
               incluia esas lineas que queria retirar para dejar #↩
               solo los datos
24          for i in range (2,22):
25              temp=lineasA[i][:-1]
26              #temp2=lineasA[i][0:9] +","+lineasA[i][9:14]
27              fcTrans.write ( temp )
28          for i in range (23,43):
29              temp=lineasA[i][:-1]
30              fcTrans.write ( temp )
31          for i in range (44,64):
32              temp=lineasA[i][:-1]
33              fcTrans.write ( temp )
34          for i in range (65,85):
35              temp=lineasA[i][:-1]
36              fcTrans.write ( temp )
37          fcTrans.write ("\n")
38          archivoA.close()
39
40
41  # Recorre los archivos de todas las carpetas que estan ↩
       contenidas en la carpeta aobligate-contact.
42  for ( path , ficheros , archivos ) in walk ( "./PDB/aobligate-↩
       contact" ) :
43       if len ( archivos ) >2:
44          nombreArch=path[-4:]+".txt"
45          archivoA = open ( path+"/DIST_"+nombreArch, "r",encoding↩
               ="utf-8" )
46          lineasA=archivoA.readlines()
47          # Concatena los archivos
48          # Inicia la fila con el nombre del complejo
49
50          fcOblig.write("complexO:"+path[-4:] )
51          for i in range (2,22):
52              temp=lineasA[i][:-1]
53              fcOblig.writelines ( temp )
```

```
54          for i in range (23,43):
55              temp=lineasA[i][:-1]
56              fcOblig.writelines ( temp )
57          for i in range (44,64):
58              temp=lineasA[i][:-1]
59              fcOblig.writelines ( temp )
60          for i in range (65,85):
61              temp=lineasA[i][:-1]
62              fcOblig.writelines ( temp )
63          fcOblig.write("\n")
64          archivoA.close()
65
66  fcTrans.close()
67  fcOblig.close()
```

A.3. Construcción archivo arff

Listing A.3: keywordstyle

```
1  from os import walk
2  ## Crea dos archivos, uno para guardar todos los archivos ↩
       FastContact que
3  ## contienen los enfoques de distancia de los complejos ↩
       transitorios, y otro con los complejos permanentes.
4  fcTrans=open ( "ComplexTransient.txt","r",encoding="utf-8" )
5  fcOblig=open ( "ComplexObligate.txt" ,"r",encoding="utf-8" )
6
7  arfftra=open ( "ComplexTO.arff" ,"w+",encoding="utf-8" )
8
9  linea=fcTrans.readlines()
10 linea2=fcOblig.readlines()
11 largo= len(linea)
12 largo2=len(linea2)
13
14 arfftra.write("% ComplejosTransientesyObligados\n")
15
16 arfftra.write("@relation TransientesyObligados\n")
17 arfftra.write("@attribute tipo {transien,obligate}\n")
18 arfftra.write("@attribute complejo STRING\n")
19 for k in range(0,80):
20     arfftra.write("@attribute energia"+str(k)+" NUMERIC\n")
```

```
21      arfftra.write("@attribute numeroAminoaProteinaA"+str(k)+" ↩
            NUMERIC\n")
22      arfftra.write("@attribute nombreAminoaProteinaA"+str(k)+" {↩
            ALA,ARG,ASN,ASP,CYS,GLN,GLU,GLY,HIS,ILE,LEU,LYS,MET,PHE,↩
            PRO,SER,THR,TRP,TYR,VAL}\n")
23      arfftra.write("@attribute numeroAminoaProteinaB"+str(k)+" ↩
            NUMERIC\n")
24      arfftra.write("@attribute nombreAminoaProteinaB"+str(k)+" {↩
            ALA,ARG,ASN,ASP,CYS,GLN,GLU,GLY,HIS,ILE,LEU,LYS,MET,PHE,↩
            PRO,SER,THR,TRP,TYR,VAL}\n")
25      arfftra.write("@attribute Dmin_Amin_A--Amin_B"+str(k)+" ↩
            NUMERIC\n")
26      arfftra.write("@attribute numeroAtom_Dmin_A"+str(k)+" ↩
            NUMERIC\n")
27      arfftra.write("@attribute nombreAtom_Dmin_A"+str(k)+" {C,CA,↩
            CB,CE,CE1,CD,CD1,CD2,CG,CG1,CZ,H,O,OD1,OD2,OE1,OE2,OG,OG1↩
            ,OH,N,NE,NE2,NH1,NH2,NZ,HZ3,HH11,HE,HZ2,HZ1,HD22,HH22,↩
            HH12,CG2,HD21,HG,HE22,ND2,CE2,SD,CZ2,CZ3,NE1,CH2,OXT,CE3,↩
            SG,HD1,ND1,HH21,HE1,HH,HE21}\n")
28      arfftra.write("@attribute numeroAtom_Dmin_B"+str(k)+" ↩
            NUMERIC\n")
29      arfftra.write("@attribute nombreAtom_Dmin_B"+str(k)+" {C,CA,↩
            CB,CE,CE1,CD,CD1,CD2,CG,CG1,CZ,H,O,OD1,OD2,OE1,OE2,OG,OG1↩
            ,OH,N,NE,NE2,NH1,NH2,NZ,HZ3,HH11,HE,HZ2,HZ1,HD22,HH22,↩
            HH12,CG2,HD21,HG,HE22,ND2,CE2,SD,CZ2,CZ3,NE1,CH2,OXT,CE3,↩
            SG,HD1,ND1,HH21,HE1,HH,HE21}\n")
30      arfftra.write("@attribute Dmax_Amin_A--Amin_B"+str(k)+" ↩
            NUMERIC\n")
31      arfftra.write("@attribute numeroAtom_Dmax_A"+str(k)+" ↩
            NUMERIC\n")
32      arfftra.write("@attribute nombreAtom_Dmax_A"+str(k)+" {C,CA,↩
            CB,CE,CE1,CD,CD1,CD2,CG,CG1,CZ,H,O,OD1,OD2,OE1,OE2,OG,OG1↩
            ,OH,N,NE,NE2,NH1,NH2,NZ,HZ3,HH11,HE,HZ2,HZ1,HD22,HH22,↩
            HH12,CG2,HD21,HG,HE22,ND2,CE2,SD,CZ2,CZ3,NE1,CH2,OXT,CE3,↩
            SG,HD1,ND1,HH21,HE1,HH,HE21}\n")
33      arfftra.write("@attribute numeroAtom_Dmax_B"+str(k)+" ↩
            NUMERIC\n")
34      arfftra.write("@attribute nombreAtom_Dmax_B"+str(k)+" {C,CA,↩
            CB,CE,CE1,CD,CD1,CD2,CG,CG1,CZ,H,O,OD1,OD2,OE1,OE2,OG,OG1↩
            ,OH,N,NE,NE2,NH1,NH2,NZ,HZ3,HH11,HE,HZ2,HZ1,HD22,HH22,↩
            HH12,CG2,HD21,HG,HE22,ND2,CE2,SD,CZ2,CZ3,NE1,CH2,OXT,CE3,↩
            SG,HD1,ND1,HH21,HE1,HH,HE21}\n")
```

```
35
36 arfftra.write("@data\n")
37 for i in range (0,largo):
38 arfftra.write(linea[i][0:14])
39 #aqui estan las 15 columnas que se repiten, meterlas en un for
40     for j in range(0,80):
41         arfftra.write(",")
42         arfftra.write(linea[i][14+107*j:22+107*j])# 14+107*0=121↩
               sumar 107 a cada posicion por ende seria 14*107*i ↩
           ver ejemplo
43         arfftra.write(",")
44         arfftra.write(linea[i][22+107*j:26+107*j])
45         arfftra.write(",")
46         arfftra.write(linea[i][26+107*j:30+107*j])
47         arfftra.write(",")
48         arfftra.write(linea[i][30+107*j:35+107*j])
49         arfftra.write(",")
50         arfftra.write(linea[i][35+107*j:39+107*j])
51         arfftra.write(",")
52         arfftra.write(linea[i][39+107*j:48+107*j])
53         arfftra.write(",")
54         arfftra.write(linea[i][48+107*j:54+107*j])
55         arfftra.write(",")
56         arfftra.write(linea[i][54+107*j:62+107*j])
57         arfftra.write(",")
58         arfftra.write(linea[i][62+107*j:70+107*j])
59         arfftra.write(",")
60         arfftra.write(linea[i][70+107*j:77+107*j])
61         arfftra.write(",")
62         arfftra.write(linea[i][77+107*j:88+107*j])
63         arfftra.write(",")
64         arfftra.write(linea[i][88+107*j:94+107*j])
65         arfftra.write(",")
66         arfftra.write(linea[i][94+107*j:103+107*j])
67         arfftra.write(",")
68         arfftra.write(linea[i][103+107*j:110+107*j])
69         arfftra.write(",")
70         arfftra.write(linea[i][110+107*j:120+107*j])                ↩

71     arfftra.write("\n")
72
73 for i in range (0,largo2):
```

```
74  arfftra.write(linea2[i][0:14])
75  #aqui estan las 15 columnas que se repiten, meterlas en un for
76      for j in range(0,80):
77          arfftra.write(",")
78          arfftra.write(linea2[i][14+107*j:22+107*j])# ↩
               14+107*0=121 sumar 107 a cada posicion por ende seria↩
                14*107*i ver ejemplo
79          arfftra.write(",")
80          arfftra.write(linea2[i][22+107*j:26+107*j])
81          arfftra.write(",")
82          arfftra.write(linea2[i][26+107*j:30+107*j])
83          arfftra.write(",")
84          arfftra.write(linea2[i][30+107*j:35+107*j])
85          arfftra.write(",")
86          arfftra.write(linea2[i][35+107*j:39+107*j])
87          arfftra.write(",")
88          arfftra.write(linea2[i][39+107*j:48+107*j])
89          arfftra.write(",")
90          arfftra.write(linea2[i][48+107*j:54+107*j])
91          arfftra.write(",")
92          arfftra.write(linea2[i][54+107*j:62+107*j])
93          arfftra.write(",")
94          arfftra.write(linea2[i][62+107*j:70+107*j])
95          arfftra.write(",")
96          arfftra.write(linea2[i][70+107*j:77+107*j])
97          arfftra.write(",")
98          arfftra.write(linea2[i][77+107*j:88+107*j])
99          arfftra.write(",")
100         arfftra.write(linea2[i][88+107*j:94+107*j])
101         arfftra.write(",")
102         arfftra.write(linea2[i][94+107*j:103+107*j])
103         arfftra.write(",")
104         arfftra.write(linea2[i][103+107*j:110+107*j])
105         arfftra.write(",")
106         arfftra.write(linea2[i][110+107*j:120+107*j])          ↩

107     arfftra.write("\n")
108 arfftra.close()
109 fcTrans.close()
110 fcOblig.close()
```

Bibliografía

[1] aihorizon. Machine learning, part i: Supervised and unsupervised learning. 2018. URL URL{http://www.aihorizon.com/essays/generalai/supervised_unsupervised_machine_learning.htm}. [Web; accedido el 27-04-2018].

[2] Ricardo Aler. Tutorial weka 3.6.0. 2009. URL URL{http://ocw.uc3m.es/ingenieria-informatica/herramientas-de-la-inteligencia-artificial/contenidos/transparencias/TutorialWeka.pdf}. [Web; accedido el 27-04-2018].

[3] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, y Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

[4] Michael J Berry y Gordon Linoff. *Data mining techniques: for marketing, sales, and customer support*. John Wiley & Sons, Inc., 1997.

[5] Botanical. Biomoléculas. 2017. URL URL{https://www.botanical-online.com/proteinas_formacion.htm}. [Web; accedido el 27-04-2018].

[6] Tatiana Gutiérrez Bunster. *Estudio de las características energéticas en zonas de interacción proteína-proteína, para identificación de interacciones transitorias y permanentes*. Tesis Doctoral, Universidad de Concepción, 2008.

[7] Carlos J Camacho y Chao Zhang. Fastcontact: rapid estimate of contact and binding free energies. *Bioinformatics*, 21(10):2534–2536, 2005.

[8] chem eng. Zeror. 2015. URL URL{http://chem-eng.utoronto.ca/~datamining/dmc/zeror.htm}. [Web; accedido el 20-07-2018].

[9] Temas Selectos de Ciencias. Aminoácidos. 2018. URL URL{http://temas-selectos-de-ciencias.blogspot.com/p/aminoacidos_3.html}. [Web; accedido el 27-04-2018].

[10] Victor de la Torre Russis, Alfredo Valles, Raúl Gómez, Glay Chinea, y Tirso Pons. Interacciones proteína-proteína: bases de datos y métodos teóricos de predicción. *Biotecnología Aplicada*, 20(3):201–208, 2003.

[11] Walt Disney. ¿qué es un átomo? 1962. URL URL{https://www.youtube.com/watch?v=-LcQcIcH1H4}. [Web; accedido el 27-04-2018].

[12] Gutierrez Cruz Doricela, Albarran Fernandez Yarosalf Aaron, y Rico Molina Ricardo. Manual para practicas de laboratorio: Taller con weka.

[13] ehu. Péptido. 2018. URL URL{http://www.ehu.eus/biomoleculas/peptidos/pep2.htm}. [Web; accedido el 27-04-2018].

[14] Ainhoa García. Biomoléculas. 2017. URL URL{https://sites.google.com/site/biomoleculasorganicas2obto/home}. [Web; accedido el 27-04-2018].

[15] Stephen R Garner et al. Weka: The waikato environment for knowledge analysis. En *Proceedings of the New Zealand computer science research students conference*, págs. 57–64. 1995.

[16] David S Goodsell y Arthur J Olson. Structural symmetry and protein function. *Annual review of biophysics and biomolecular structure*, 29(1):105–153, 2000.

[17] M Victoria Luque Guillén. Estructura y propiedades de las proteínas. 2009.

[18] George H John y Pat Langley. Intelligence, morgan kaufmann publishers, san mateo, 1995.

[19] Ozlem Keskin, Attila Gursoy, Buyong Ma, y Ruth Nussinov. Principles of protein- protein interactions: What are the preferred ways for proteins to interact? *Chemical reviews*, 108(4):1225–1244, 2008.

[20] Juan Manuel González Mañas. Protein data bank. 2018. URL URL{http://www.ehu.eus/biofisica/juanma/rasmol/pdb.htm}. [Web; accedido el 19-06-2018].

[21] Trudy McKee y James Robert McKee. *Biochemistry: the molecular basis of life*, tomo 108. McGraw-Hill New York, 2003.

[22] Julian Mintseris y Zhiping Weng. Structure, function, and evolution of transient and obligate protein–protein interactions. *Proceedings of the National Academy of Sciences*, 102(31):10930–10935, 2005.

[23] Diego García Morate. Manual de weka. *Disponível através do e-mail diego. garcia. morate@ mail. com*, 2008.

[24] Alberto Morán. Aminoácidos, péptidos y proteínas. 2016. URL URL{http://www.dciencia.es/aminoacidos-peptidos-y-proteinas}. [Web; accedido el 27-04-2018].

[25] Irene MA Nooren y Janet M Thornton. Diversity of protein–protein interactions. *The EMBO journal*, 22(14):3486–3492, 2003.

[26] Dr. José Antonio Martínez Oyanedel. Principios de estructura de proteínas. 1997. URL URL{http://www2.udec.cl/~jmartine/Indice.htm}. [Web; accedido el 27-04-2018].

[27] E. M. Phizicky y S. Fields. Protein-protein interactions: methods for detection and analysis. *Microbiological reviews*, 59(94):133, 1995.

[28] Francisco Gálvez Prada. ¿cuál es la estructura de los aminoácidos? 2018. URL `URL{https://www.hidden-nature.com/cual-es-la-estructura-de-los-aminoacidos}`. [Web; accedido el 27-04-2018].

[29] Jorge Enrique Rodríguez Rodríguez, Edwar Alonso Rojas Blanco, y Roger Orlando Franco Camacho. Clasificación de datos usando el método k-nn. *revista Vínculos*, 4(1):4–18, 2007.

[30] Juan Saavedra. *Identificacion de nuevos factores discriminantes basados en distancia entre aminoacidos en zona de interaccion entre proteinas.* Tesis Doctoral, Universidad del Bio-bio, 2017.

[31] Jesus Salgado. Tecnologías de la medicina molecular. 2016. URL `URL{https://www.uv.es/bbm/master/2016.pdf}`. [Web; accedido el 27-04-2018].

[32] Gopichandran Sowmya y Shoba Ranganathan. Protein-protein interactions and prediction: a comprehensive overview. *Protein and peptide letters*, 21(8):779–789, 2014.

[33] WEKA. Weka. 2018. URL `URL{https://www.cs.waikato.ac.nz/ml/weka/}`. [Web; accedido el 8-10-2018].

[34] Wikipedia. Polipéptido. 2018. URL `URL{https://es.wikipedia.org/wiki/Polip%C3%A9ptido}`. [Web; accedido el 27-04-2018].

[35] Wikipedia. Péptido. 2018. URL `URL{https://es.wikipedia.org/wiki/P%C3%A9ptido}`. [Web; accedido el 27-04-2018].