UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

FACULTAD DE INGENIERÍA

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

MAESTRÍA EN INGENIERÍA EN COMPUTACIÓN

PREDICCIÓN DE LA UBICACIÓN FINAL DE UN AUTOMÓVIL POR MEDIO DE MODELOS DE APRENDIZAJE COMPUTACIONAL

TESIS

PARA OBTENER EL GRADO DE MAESTRO EN INGENIERÍA

PRESENTA

NORMANDO ALI ZUBIA HERNÁNDEZ

DIRECTOR DE TESIS

DR. LUIS CARLOS GONZALES GURROLA

CHIHUAHUA, CHIHUAHUA. OCTUBRE 2016

UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

FACULTAD DE INGENIERÍA

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

MAESTRÍA EN INGENIERÍA EN COMPUTACIÓN

PREDICCIÓN DE LA UBICACIÓN FINAL DE UN AUTOMÓVIL POR MEDIO DE MODELOS DE APRENDIZAJE COMPUTACIONAL

TESIS

PARA OBTENER EL GRADO DE MAESTRO EN INGENIERÍA

APROBADO:

DR. LUIS CARLOS GONZALES GURROLA, director

DR. FERNANDO MARTINEZ REYES, sinodal

M.I DAVID MALOOF FLORES, sinodal

OCTUBRE 2016

CHIHUAHUA, CHIHUAHUA.

13 de octubre de 2016

UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA
Facultad de Ingeniería

ING. NORMANDO ALÍ ZUBIA HERNÁNDEZ

Presente

En atención a su solicitud relativa al trabajo de tesis para obtener el grado de Maestro en Ingeniería en Computación, nos es grato transcribirle el tema aprobado por esta Dirección, propuesto y dirigido por el director **Dr. Luis Carlos González Gurrola** para que lo desarrolle como tesis, con el título: **"PREDICCIÓN DE LA UBICACIÓN FINAL DE UN AUTOMÓVIL POR MEDIO DE MODELOS DE APRENDIZAJE COMPUTACIONAL".**

ÍNDICE

Facultad de Ingeniería
Circuito No.1, Campus Universitario 2
Chihuahua, Chih. C.P. 31125
Tel. (614) 442-95-00 www.fing.uach.mx

UNIVERSIDAD INTERNACIONAL

4 ESTRATEGIA MALLA Y COMPARACIÓN DE MODELOS

4.1 Generación de estrategia y análisis de algoritmos

4.2 Procesamiento de datos para entrada de modelo

4.2.1 Criterio para reducción de trayectoria

4.2.2 Normalización de vectores

4.3 Clasificación de cuadrante destino

4.3.1 Configuración de los modelos

4.4 Resultados

4.4.1 Resultado de modelos utilizando 4 cuadrantes

4.4.2 Resultado de modelos utilizando 16 cuadrantes

4.4.3 Resultado de modelos utilizando 32 cuadrantes

4.5 Análisis de resultados

5 RED NEURONAL COMO ESTRATEGIA DE SOLUCIÓN

5.1 Estructura del modelo

5.2 Resultados

5.3 Análisis de resultados y estrategias de mejora

5.3.1 Inicios y destinos difíciles

5.3.2 Análisis de sectores difíciles

5.3.3 Uso de patrones individuales para mejorar la predicción de destinos

6 CONCLUSIONES

Referencias

Currículum vitae

Solicitamos a Usted tomar nota de que el título del trabajo se imprima en lugar visible de los ejemplares de las tesis.

A T E N T A M E N T E
"naturam subiecit aliis"

EL DIRECTOR

M.I. JAVIER GONZÁLEZ CANTÚ

FACULTAD DE INGENIERÍA U.A.CH. DIRECCIÓN

EL SECRETARIO DE INVESTIGACIÓN Y POSGRADO

DR. FERNANDO RAFAEL ASTORGA BUSTILLOS

Facultad de Ingeniería
Circuito No. 1, Campus Universitario 2
Chihuahua, Chih. C.P. 31125
Tel. (614) 442-9500 www.fing.uach.mx

Abstract

The enormous amount of data generated nowadays and the close relationship between society and technology opens new and challenging opportunities to solve several issues present within our environment. Nowadays transportation is one of the sectors with more problems, the government puts daily all its effort to improve this area in different ways, for example improving traffic monitoring, doing better transport networks, etc.

Particularly, in the city of Porto, Portugal and different countries that have changed their systems to electronic dispatching have an increasing interest to predict the final destination of any given user. This information could be exploited to better attend the demand of taxis, since knowing in advance where a trip may end, the dispatcher could arrange a new user service around. Also this information could be used in the benefit of the environment reducing the pollution with a better transportation service management.

Unfortunately the destination prediction has several problems, for example the inconsistency of trips trajectories due to a GPS error, similar trajectories that make difficult the identification of patterns and several problems caused by highways.

To tackle this problem, in this paper we used several machine learning models, specifically an ANN to predict the final destination of a trip. Also, we mentioned several problems that may arise in this sector and strategies to mitigate those problems and improve the accuracy of the model in the prediction of trip destination.

Resumen

La enorme cantidad de datos que se generan hoy en día y el acercamiento que existe entre la sociedad y la tecnología, otorgan grandes oportunidades y retos en la resolución de problemáticas presentes en nuestro entorno. Una de las áreas de oportunidad para el uso de esta información es el área de transporte, diariamente se ven esfuerzos por parte del gobierno para mejorar la infraestructura y los servicios que compete al sector vial, por ejemplo: la mejora de las vías de transporte, optimización y monitoreo de tráfico vehicular, mejoras en el manejo de servicios por parte de las empresas de transporte público y privado, entre otros.

Particularmente en la ciudad de Porto, Portugal y países en los cuales se ha cambiado el sistema de manejo de taxis a un sistema electrónico, se tiene un gran interés por conocer el destino de los viajes demandados por los usuarios. El conocimiento de dicha ubicación supondría un gran beneficio, debido a que se podría mejorar el manejo de sus recursos, conociendo que taxis se encontrarán en dicha zona en un futuro y de ser necesario, atender un servicio solicitado en dicha posición. A su vez otorgaría un beneficio al medio ambiente ya que no se utilizarían vehículos de forma innecesaria y disminuiría la contaminación.

Lamentablemente esta problemática trae consigo ciertos obstáculos, entre ellos: inconsistencia en datos causado por errores de GPS, trayectorias similares que impiden la identificación correcta de patrones, áreas difíciles de predecir debido a vías rápidas, etc.

A continuación en el presente documento se plantea el uso de modelos de aprendizaje máquina, especialmente el uso de un modelo de red neuronal para el modelado de trayectorias de vehículos y predicción de destinos. Así mismo se describen a detalle diversos obstáculos que se pueden presentar en problemas de esta índole y estrategias de integración de modelos para mitigarlos, resultando en el aumento de precisión de modelos para la predicción de ubicaciones finales.

Dedicatoria

Primeramente a Dios, ya que Él me ha acompañado durante el transcurso de toda mi vida, otorgándome innumerables bendiciones, logros y alegrías, pero en especial me ha acompañado durante estos dos años, ayudándome a terminar con éxito esta etapa de mi vida.

A mi familia, que siempre ha sido mi peldaño en todas las decisiones que he tomado, sin su gran apoyo, dirección y amor, nunca hubiera logrado llegar hasta esta instancia de mi vida.

Agradecimientos

Agradezco inmensamente al Consejo Nacional de Ciencia y Tecnología (CoNaCyT), a la Universidad Autónoma de Chihuahua y en especial a la Facultad de Ingeniería, por haberme dado la oportunidad de estudiar esta maestría y contribuir en gran medida a mi formación académica y personal. De igual manera agradezco a todos mis maestros de la maestría, que sin su ayuda y orientación, nunca hubiera alcanzado la culminación de la misma.

Así mismo, agradezco grandemente a la Secretaría de Investigación y Postgrado, a la Secretaría Administrativa, al director M.I. Ricardo Ramón Torres Knight y a todas las personas que hicieron posible la realización de una de las mejores experiencias de mi vida, la cual, fue una estancia de investigación en Inglaterra.

De la misma manera agradezco a *The University Of Nottingham* y en especial al Dr. Andrew J. Parkes, por haberme guiado en mi estancia de investigación.

A mi director de Tesis, el Dr. Luis Carlos González Gurrola, que siempre me apoyo en el desarrollo de mi tesis, agradezco grandemente la orientación, seguimiento y supervisión continúa de la misma. Así mismo, agradezco a mi sinodal, el Dr. Fernando Martínez Reyes, por las valiosas observaciones hechas a mi tesis y enriquecer con ellas el contenido de la misma.

Al M.I David Maloof Flores por todo el apoyo otorgado a lo largo de la maestría, comenzando por la orientación y seguimiento en el proceso de ingreso a la misma, por el trabajo realizado para que nuestra estancia en Inglaterra fuera un éxito y por último por darse el tiempo de revisar mi tesis y guiarme en el proceso de titulación.

Por último pero no menos importante, agradezco a Dios, a mi familia, a mi novia Margarita y a mis amigos por siempre apoyarme en todo momento. Sin su gran apoyo esto no hubiera sido posible.

Índice de Contenido

Índice de Figuras

Índice de Tablas

1

INTRODUCCIÓN

Hoy en día la tecnología crece constantemente, aumentando a su vez la influencia que tiene en la sociedad. Día a día se crean miles de dispositivos que ayudan a las personas en sus actividades cotidianas, recolectando grandes cantidades de información y resolviendo los problemas que se presenten en base al procesamiento de ella, ejemplo de estos dispositivos son los teléfonos y relojes inteligentes, bandas para el monitoreo de la salud, entre otros.

Una de las herramientas que también es utilizada frecuentemente y que a su vez genera miles de datos, es el "*Global Positioning System*", mejor conocido como GPS. Su objetivo principal es el de brindar a los usuarios los servicios de posicionamiento mediante el uso de satélites que orbitan alrededor de la tierra [1]. Esta tecnología ha crecido durante los últimos años y es usado en muchas áreas de desarrollo tales como la agricultura, topografía, aviación, entre otros. Es común que un usuario haya utilizado este tipo de tecnología en alguna etapa de su vida, puesto que se encuentra implantado comúnmente en dispositivos celulares y automóviles para auxiliar en la navegación o para buscar una dirección.

Debido a la gran cantidad de datos generados por esta herramienta se abre todo un campo de mejora y emprendimiento para diferentes áreas de desarrollo, especialmente en el área de transporte y navegación, ya que todos estos datos pueden ser analizados e interpretados de diferentes maneras para producir conocimiento que pueda ser útil en la resolución de los problemas presentes en dichas áreas.

En este capítulo se introduce al lector al trabajo de investigación. En la primera sección se mencionarán los problemas presentados en el área de transporte. En la sección 2 se proseguirá con la motivación del problema. En la sección 3 se realizará un análisis sobre el estado del arte. En la sección 4 se describirá la delimitación del trabajo de investigación. Terminando con los objetivos generales y específicos en la sección 5.

Así mismo este trabajo de investigación se encuentra dividido en 6 capítulos. Como ya se mencionó en

el párrafo anterior, el capítulo 1 es una introducción sobre el trabajo de investigación y un análisis sobre el estado del arte en este tema. El capítulo 2 presentan los fundamentos teóricos necesarios para el desarrollo del trabajo de investigación. En el capítulo 3 se abordará el procesamiento de los datos. En el capítulo 4 se describirá la primera estrategia implementada para atacar el problema de predicción de destinos. En el capítulo 5 se presentará el uso de un modelo de red neuronal para la predicción de destinos así como la propuesta de integración de modelos para mejorar la precisión de los mismos, terminando con el capítulo 6 que concierne a las conclusiones de este documento.

1.1. Planteamiento del Problema

El avance en la tecnología y sus cambios tan constantes, han originado transformaciones en el modo de realizar ciertos servicios, esto se presenta principalmente en el sector de transporte, en el cual, los esfuerzos por tener un servicio cada vez más eficiente y automático generan a su vez otras demandas que antes no se necesitaban. Esto ocurre especialmente en la industria de transporte privado, es decir la industria de taxis, en la cual, los sistemas de despacho han sido actualizados para lograr un manejo automático y de vanguardia.

Anteriormente, en el servicio de transporte privado, se usaban sistemas de despacho por medio de radio, en los cuales, tanto despachadores como taxistas podían tener una comunicación fluida y dar a conocer la ubicación actual y el destino de cada uno de los viajes, logrando así un servicio medianamente eficiente. En la actualidad, en países más desarrollados, el sistema de despacho por medio de radio ha sido cambiado por sistemas electrónicos, en los cuales, a pesar de ser posible el conocimiento de la ubicación actual del automóvil, no es posible conocer la ubicación final del viaje, o al menos no en todos los casos.

Aunado a esto, la comunicación de uno a muchos, la cual era posible en el sistema de despacho por medio de radios, ahora se convierte en comunicación uno a uno a consecuencia de los cambios realizados en el servicio de despacho, haciendo más difícil el trabajo de los administradores, los cuales tienen que comunicarse con varios taxis hasta encontrar el indicado. Un ejemplo verídico de este tipo de problemas recae en la ciudad de Porto, Portugal, donde la industria de taxis ha requerido ayuda explícita para solventar este problema que se presenta en la ciudad.

Existen otro tipo de alternativas como UBER, donde se podría pensar que este problema no afecta su funcionamiento y que el conocimiento de la ubicación final por medio de algoritmos ya no es necesaria, debido a que se puede ingresar el destino en la aplicación. A pesar de esto, no todos los usuarios ingresan la ubicación final del viaje, haciendo que el sistema o aplicación no conozcan a ciencia cierta donde terminará dicho recorrido y por ende, no tenga la capacidad de premeditar la posibilidad de que dicho automóvil estará

cerca en una futura demanda del servicio.

Así mismo, a pesar de que en México no se hayan actualizados los sistemas de despacho y la industria de taxis no se vea afectada por este tipo de cuestiones, el problema se puede ver apreciado en otras áreas de desarrollo, por ejemplo, el tráfico generado en ciertas horas del día y en ciertas áreas de la ciudad. Con una predicción del destino de diversos automóviles, se podría hacerse frente a esta problemática permitiendo que el tráfico fuera conocido con anticipación y que se pudieran tomar medidas al respecto.

Problemas de esta índole han sido muy poco atacados y no se conoce con exactitud cuál es la mejor de las soluciones, aunado a esto es difícil decidir que herramientas de deben de utilizar para poder atacar este tipo de problemáticas.

1.2. Motivación

Como podemos apreciar, el anticiparse a una próxima ubicación de un vehículo otorga una gran oportunidad para la resolución de problemas en el área de transporte.

La culminación de esta investigación y la implementación de modelos de aprendizaje máquina podría suponer una disminución de gastos de funcionamiento en el sector de transporte público y privado, específicamente en taxis, ya que el uso de dichas herramientas podrían otorgar información relevante para la toma de decisiones a las empresas taxistas, propiciando que puedan lograr predecir la ubicación futura de cada uno de los automóviles que conforman su flotilla y brindándoles una mejor administración de sus recursos.

Así mismo se tendría una disminución en la contaminación y tráfico debido a que se optimizaría el uso de los vehículos y sus trayectorias.

1.3. Estado del arte

Avances en la obtención de ubicaciones y uso de herramientas como GPS, han generado miles de datos espaciales en forma de trayectorias; donde una trayectoria se conoce como una secuencia de ubicaciones o coordenadas.

Existen diversos trabajos que tratan sobre el análisis de trayectorias sobre diferentes objetos, entre ellos encontramos el análisis de personas [2, 3, 4, 5, 6, 7, 8, 9], los cuales, se enfocan en la recolección de trayectorias individuales de usuarios con el fin de aprender sus patrones y dar un valor agregado a todas sus actividades diarias.

En [2] se utiliza un modelo oculto de markov (HMM), en el cual se recolecta la información de un individuo, e identificando las ubicaciones más visitadas por el usuario, el modelo identifica la secuencia más

probable de lugares que serán visitados por la personas. En dicho trabajo se logra una predicción del 70 % en un periodo de una hora sobre futuras locaciones.

En [3] se pretende predecir la trayectoria de estudiantes dentro de un campus universitario utilizando su dispositivo como identificador, en dicho documento se comparan diversos algoritmos como son el "*Prediction Partial Match*", "*Sampled Pattern Matching*", "*Lz-based*" y "*Markov Models*" mostrando alrededor de 72 % a 80 % de precisión.

Basado en trayectorias de señales GSM, Timothy et al. [4] clasifican la movilidad de un usuario en tres estados, los cuales consisten en: caminando, sin movimiento y conduciendo, logrando un porcentaje de precisión del 85 %. Similar a este trabajo encontramos las investigaciones de Patterson et al. [10] y Liao et.al [5] que usan trayectorias GPS para inferir la rutina de transporte de un usuario.

Zheng et al. [6, 7] clasifican la trayectoria de un usuario mediante su modo de transporte, los cuales son definidos como conduciendo, en bicicleta, en autobús o caminando. En dicho trabajo se usa un árbol de decisión para procesar las trayectorias de 65 usuarios recolectadas en un periodo de 10 meses, logrando un 72 % de precisión en el modelo.

En [8, 11] logran encontrar ubicaciones importantes y trayectorias comunes de viajes por medio de múltiples trayectorias de usuarios en coordenadas GPS. Así mismo Hasan et al. [9] analiza los patrones de personas en un ambiente urbano recolectados desde aplicaciones sociales multimedia.

Como se puede apreciar, la información sobre los patrones de un usuario puede llegar a otorgar gran valor agregado a sus actividades. Por ejemplo, si con el uso de trayectorias de ciertos usuarios se pudiera predecir si se dirigen o no a alguna zona comercial, los locales pudieran estar mejor preparados para otorgar un mejor servicio, o también se pudiera enviar algunas promociones vigentes y hacer más atractiva la visita al mismo.

Otro de los campos más importantes en lo que compete el análisis de trayectorias es el de transporte. Un gran número de vehículos ya contienen este tipo de tecnología integrados a ellos como son: taxis, autobuses vehículos privados, entre otros. La integración del GPS con tecnología vehicular permite que se generen miles de registros día con día, logrando que se pueda analizar este tipo de información para diversas áreas de interés como lo son: análisis de tráfico [12, 13] y recomendaciones de rutas [14, 15, 16] por mencionar algunos.

Dentro de esta área encontramos diferentes tipos de trabajos, los primeros que competen a la planeación de la mejor ruta entre dos o más puntos [14, 15, 16], los segundos tratan sobre la predicción de la ruta futura de un objeto determinado [17, 18, 19, 20] y por último encontramos trabajos que competen a la predicción de la próxima locación [21, 22, 23, 24, 25], de los cuales son en los que está basado este trabajo de investigación.

Dentro de los trabajos que tratan sobre la planeación de la mejor ruta encontramos el trabajo de Chen et. al [14, 15] donde se trata de buscar las mejores *k* trayectorias conectadas y encontrar la ruta más popular entre dos puntos. Así mismo tenemos el trabajo de Yuan et. al [16] en donde se minan las trayectorias históricas de un gran número de taxis para proveer al usuario la ruta más rápida entre dos ubicaciones. En este trabajo se utilizan las trayectorias de 3 meses de 33,000 taxis obteniendo que aproximadamente 50 % de las rutas obtenidas son 20 % más rápidas que otros trabajos realizados.

Entre los trabajos que competen a la predicción de la ruta futura se encuentra Jurgeb Wiest [17] en el cual tratan de predecir por medio de *Kalman Filter* la trayectoria de un carro cuando se está acercando a una intersección y así prever si pudiera chocar con otro vehículo al dar vuelta en ella. Así mismo tenemos a R. Fraile [18] que predice la trayectoria de un vehículo a corto plazo dentro de un estacionamiento para evitar choques por medio del HMM.

En [19] se propone el uso de un modelo de markov simple que usa los segmentos pasados de la trayectoria para predecir la ruta futura, logrando en este trabajo un 90 % de precisión en el modelo. Así mismo en Horvitz y Krumm [20] usan 7,000 trayectorias junto con un modelo de inferencia bayesiana para predecir la próxima locación de un conductor.

Como se mencionó anteriormente, existen trabajos que se enfocan en solamente la predicción de la próxima locación del viaje, dentro de estos trabajos encontramos a Álvarez García et al. [21] donde se presenta un sistema basado en la generación de un HMM a partir de las coordenadas GPS pasadas para predecir el próximo destino del usuario. En [22] se presenta la integración de tres modelos entrenados con diferentes tipos de patrones para mejorar la predicción de la ubicación futura de un usuario. Así mismo en Monreale et al. [23] propone el uso de la historia de movimientos que ha tenido un usuario para la construcción de un árbol que predice la ubicación futura.

Por otro lado la predicción de trayectorias en servicios de transporte como taxis ha ido creciendo en esta época, en dicho sector encontramos los trabajos de, Lam *et al.* [24] donde se predice la ubicación futura y el tiempo estimado del viaje procesando las trayectorias parciales de taxis por medio de métodos de ensamble. En Brebisson *et al.* [25] se utilizan diferentes tipos de redes neuronales para predecir la ubicación final de un viaje en taxi usando las coordenadas GPS del viaje.

Como se ha mencionado anteriormente a lo largo de este apartado, el tratamiento de los datos obtenido por medio de tecnología GPS brinda la oportunidad de obtener mucha información que a su vez es traducida en conocimiento. A pesar de esto, el uso de dichas técnicas para la predicción de ubicaciones no se ha explotado como debería ya que generalmente se tratan trayectorias muy pequeñas de recorrido o tratan comportamiento de usuarios por separado más no tratan trayectorias de usuarios dentro de vehículos a largo

plazo. Así mismo, en el campo del procesamiento de trayectorias queda todavía un camino muy largo por recorrer, ya que a pesar de todos los esfuerzos por mitigar este tipo de errores no se ha encontrado una solución absoluta para eliminarlos.

1.4. Delimitación

El conjunto de datos con el que se estará trabajando se compone de los recorridos de todo un año de aproximadamente 442 taxis de la ciudad de Porto, Portugal, cuyas fechas corresponden desde el 1 de Julio del 2013 al 30 de Junio del 2014. El objetivo principal de este problema es predecir el destino final de cada taxi en base a sus trayectorias iniciales por medio de modelos de aprendizaje máquina utilizados con éxito en trabajos anteriores para la predicción de locaciones y comportamiento de usuarios.

El conjunto de datos cuenta con aproximadamente 1.7 millones de registros de los cuales se utilizaron solamente 1.5 millones, lo cual supondría tanto una ventaja como un obstáculo debido a que se tiene gran cantidad de datos donde se podrían obtener gran número de patrones de movimiento e información interesante pero el pre-procesamiento de estos datos llevaría a un gran consumo de tiempo y de recursos computacionales.

Aunado a esto muchos de los registros que componen al *dataset* podrían tener datos incompletos o que no sirvan para obtener la información que se necesita.

1.5. Objetivos generales

Utilizar modelos de aprendizaje máquina para predecir la ubicación final de un taxi en base a su trayectoria inicial.

1.5.1. Objetivos específicos

- Depurar el conjunto de datos para obtener los registros que sean apropiados para el uso de modelos de aprendizaje máquina.
- Analizar que modelos de aprendizaje máquina son los adecuados para una posible solución a este problema.
- Implementar el modelo de aprendizaje máquina elegido.
- Predecir la próxima locación del taxi.

2

MARCO TEÓRICO

En este capítulo se presentarán los conceptos necesarios para entender los algoritmos utilizados en este trabajo. En la primera sección se explicara el concepto de aprendizaje máquina. Se continuará en la sección 2 con la descripción a detalle del algoritmo de redes neuronales, el cual es el algoritmo principal en la estrategia mostrada en este documento. En la sección 3 se abordará el algoritmo de árboles de decisión utilizado en la estrategia malla explicada en capítulos posteriores. Siguiendo en la sección 4 con la descripción de máquina de vectores de soporte y por último en la sección 5 se explicarán los métodos de ensamble ambos también utilizados en la estrategia malla.

2.1. Aprendizaje Máquina

Es el campo de estudio que diseña e implementa algoritmos de computadora para transformar datos en decisiones inteligentes. Dicha tarea es realizada por los algoritmos haciendo búsquedas de patrones dentro de los datos para así encontrar relación entre ellos y situaciones que se pueden observar en el entorno [26]. Se dice que una maquina aprende si puede utilizar experiencias pasadas para poder resolver experiencias futuras. Para que la máquina pueda aprender es necesario de tres componentes básicos:

- Datos.- Los cuales son utilizados para observaciones
- Abstracción.- Que representa la información en diferentes enfoques para la búsqueda de patrones
- Generalización.- Que usa los datos ya abstraídos en la etapa anterior para decidir la acción a realizar.

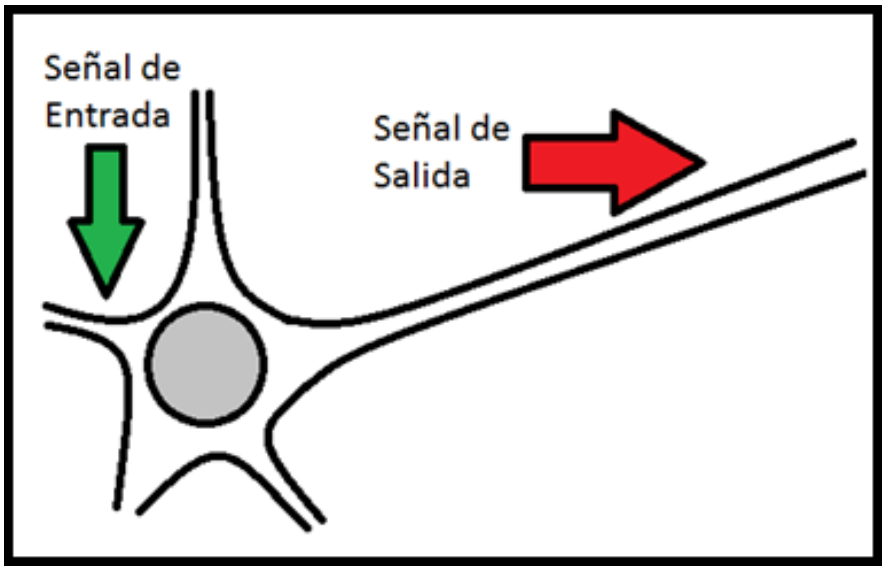


Figura 2.1: Ejemplo del procesamiento de señales que realiza la neurona por medio de la sinapsis.

2.2. Redes Neuronales

Una red neuronal es un algoritmo de aprendizaje máquina, el cual intenta representar la relación entre señales de entrada y señales de salida modelando así la reacción que se genera dentro del cerebro de un ser vivo y las conexiones que existen dentro de él [26]. En el cerebro existen miles de células interconectadas llamadas neuronas, por medio de ellas se transmite toda la información necesaria para poder realizar cualquier actividad que el cuerpo o el usuario necesite. El proceso de transmisión y procesamiento de estas señales consta de una señal de entrada, la cual al ser recibida por la neurona, es procesada y enviada a otra neurona, dicho proceso es llamado sinapsis (Véase Figura 2.1); dicho proceso es realizado hasta que dicha señal, habiendo sido procesada por todas las neuronas correspondientes, genere una salida la cual se percibe como una acción de un ser vivo.

El proceso anteriormente mencionado es representado en la red neuronal por medio de nodos que fungen como neuronas en la red, dichos nodos reciben valores de entrada que después son procesados y enviados a otra capa de nodos las cuales al final producen una salida. Generalmente una red neuronal se compone de los siguientes elementos (Véase Figura 2.2):

- Una capa de entrada
- Un vector de pesos para cada capa de neuronas
- Capa oculta con funciones de procesamiento

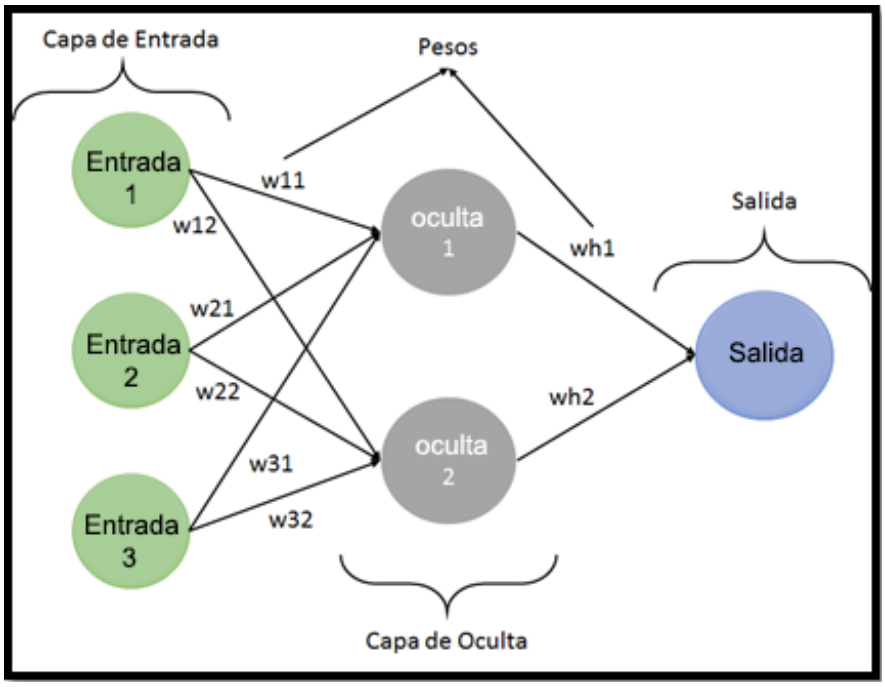


Figura 2.2: Red neuronal sencilla, 3 entradas, 1 capa oculta y una salida. Así mismo podemos observar los pesos de cada una de las capas.

- Capa de salida
- Algoritmo de entrenamiento

2.2.1. Back-Propagation

Como se describió anteriormente, la red neuronal necesita de algún algoritmo de entrenamiento para que pueda ser capaz de reconocer los patrones que se encuentran dentro de los datos. Uno de esos algoritmos es el algoritmo de *Back – Propagation* cuya función es registrar los desfases que se tienen de la salida correcta a la salida que se está obteniendo de la red neuronal, para luego ajustar los pesos en cada una de las capas y así en próximas clasificaciones la red neuronal pueda funcionar adecuadamente. Para ello el algoritmo de *back - propagation* consta de dos fases:

- Fase hacia adelante (Forward)
- Fase hacia atrás (Backward)

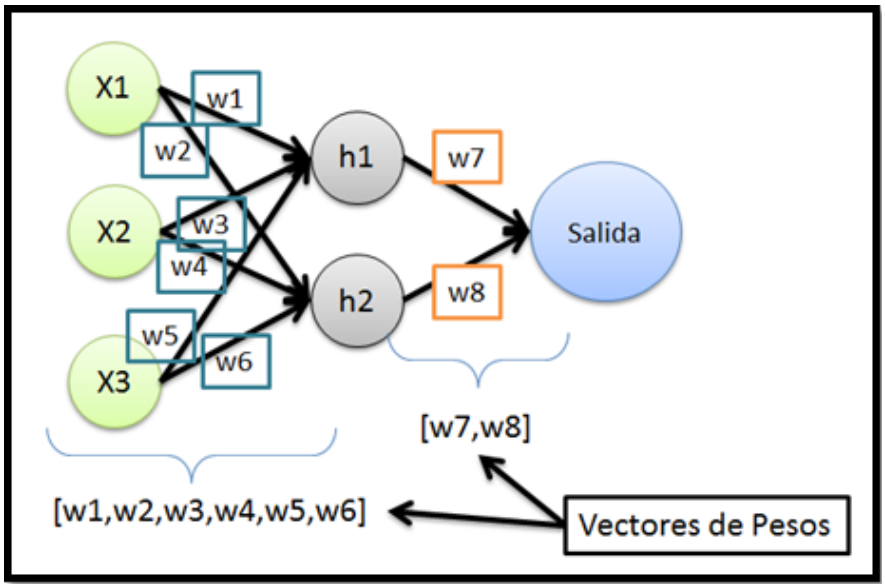


Figura 2.3: Inicialización de pesos en valores aleatorios. Los pesos inicializados observados en esa imagen son los de la capa de entrada y la capa oculta.

2.2.2. Fase hacia adelante

En la primera fase se obtienen los valores de entrada en cada uno de los nodos de la primera capa, así mismo se inicializan los pesos aleatoriamente en cada una de las capas y se realizan los cálculos necesarios en las capas ocultas para la generación de una salida.

2.2.3. Fase hacia atrás

En la segunda fase se compara la salida resultante con la salida objetivo (salida correcta). La diferencia entre estos valores, es decir, el error de la salida es propagado hacia atrás en la red para así ajustar los valores de los pesos en cada una de las capas.

El procedimiento para entrenar una red neuronal se puede observar a continuación.

2.2.4. Inicialización

El primer paso en el algoritmo es la inicialización de pesos de cada una de las capas, la longitud del vector de los pesos en la capa oculta dependerá del número de entradas que reciba la red neuronal, así mismo la longitud del vector de pesos en la capa de salida dependerá del número de neuronas en la capa oculta (Véase Figura 2.3).

2.2.5. Entrenamiento

En la primera fase hacia adelante, es necesario realizar el cálculo de la sumatoria de las multiplicaciones entre cada vector de entrada y su peso asignado por medio de la ecuación 2.1:

$$h_j = \sum x_i w_{ij} \tag{2.1}$$

Dónde:

- h_j es el valor de la sumatoria de multiplicaciones entre entradas y pesos
- x_i es el valor de la entrada en la neurona *i*
- w_{ij} es el valor del peso correspondiente a cada neurona de entrada *i* y neurona oculta *j*

Después de esto se aplica una función logística para generar la salida de cada neurona en la capa oculta por medio de la ecuación 2.2:

$$a_j = g(h_j) = \frac{1}{1 + \exp(-\beta h_j)} \tag{2.2}$$

Donde:

- a_j es la salida de la neurona oculta, la cual puede ser 0 o 1
- β es un valor positivo
- h_j es la salida de la neurona oculta obtenida de ecuación 2.1

El proceso descrito anteriormente puede ser observado en la Figura 2.4.

Después se calcula la salida por medio de la ecuación 2.3, la cual será el resultado de última neurona de la red. Dicho valor se obtiene sumando las multiplicaciones de los valores de salida de cada neurona con sus respectivos pesos:

$$y_k = g(h_j) = \sum a_j w_{jk} \tag{2.3}$$

Donde:

- y_k es el resultado de la neurona de salida *k*
- a_j es el valor de la neurona oculta *j*

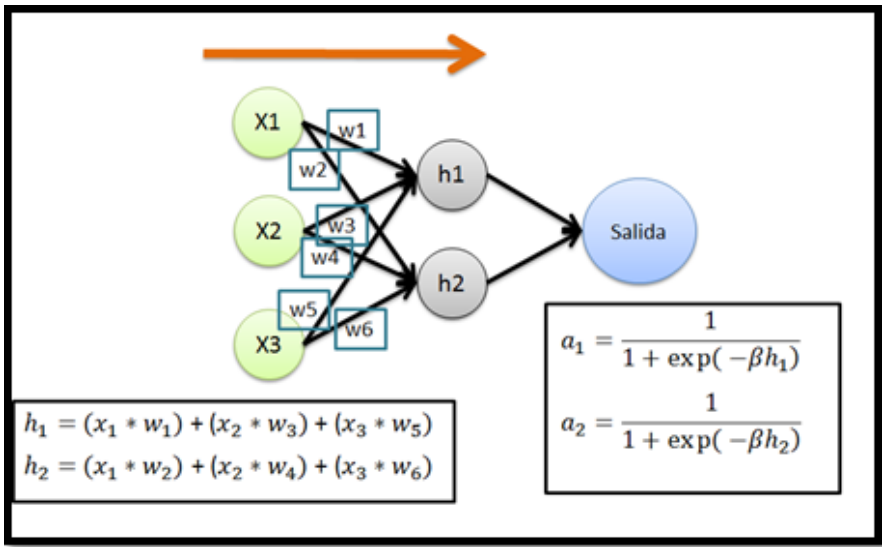


Figura 2.4: Procedimiento utilizado para calcular el valor de las neuronas ocultas y por consiguiente el estado de su activación.

- w_{jk} es el valor del peso correspondiente a cada neurona oculta *j* y neurona de salida *k*

El resultado puede ser sometido a diversos tipo de funciones dependiendo del fin que se desee, en este caso la salida se dejó tal cual debido a que es una función lineal. El procedimiento descrito anteriormente puede ser observado en la Figura 2.5.

Al haber obtenido la salida de la red neuronal se debe de proceder al ajuste de pesos, el cual se realiza en la fase hacia atrás. Para ello es necesario calcular la diferencia entre la salida de la red y la salida verdadera que debió haber tenido por medio de la ecuación 2.3:

$$\delta_o(k) = (o_k - t_k)(1 - o_k)o_k \tag{2.4}$$

Donde:

- $\delta_o(k)$ es el error total de la red neuronal
- o_k es la salida obtenida de la neurona de salida *k*
- t_k es la salida real que debe tener la neurona *k*

Este error obtenido es propagado hacia atrás en la red, modificando así los pesos de la capa de salida por medio de la ecuación 2.5:

$$w_{jk} = w_{jk} - \eta\delta_o(k)a_j \tag{2.5}$$

Donde:

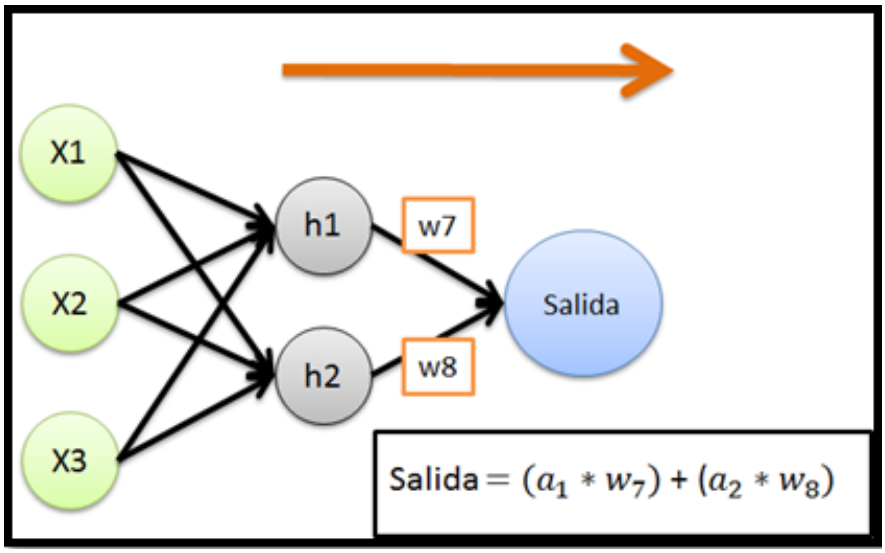


Figura 2.5: Cálculo realizado por parte de las neuronas de salida para conocer el resultado de la clasificación.

- w_{jk} es el valor del peso perteneciente a la neurona de salida *k* y neurona oculta *j*
- η que es el índice de aprendizaje de la red
- $\delta_o(k)$ es el error total de la red neuronal calculado en ecuación 2.4
- a_j es el valor de la neurona oculta *j* calculado en la ecuación 2.2

El proceso descrito anteriormente puede apreciarse en la Figura 2.6.

Después de haber actualizado los pesos de la capa de salida es necesario obtener los errores de la capa oculta con la ecuación 2.6:

$$\delta_h(j) = a_j(1 - a_j) \sum w_j \delta_o(k) \tag{2.6}$$

Donde:

- $\delta_h(j)$ es el error de la neurona oculta *j*
- a_j es el valor de la neurona oculta *j* calculado en la ecuación 2.2
- w_j valor de cada peso correspondiente a cada neurona oculta *j*
- $\delta_o(k)$ es el error total de la red neuronal calculado en ecuación 2.4

El último paso de la fase hacia atrás es ajustar los pesos de la capa oculta mediante la ecuación 2.7:

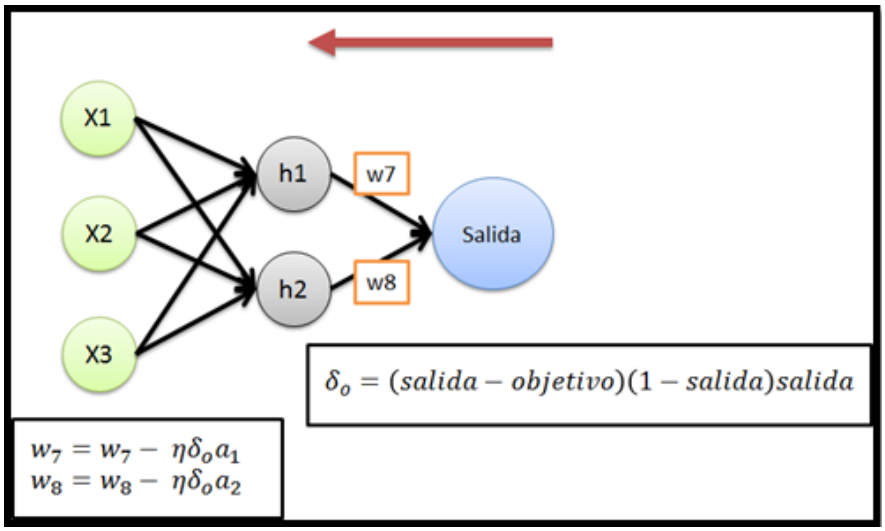


Figura 2.6: Ajuste de pesos en capa de salida.

$$w_i = w_i - \eta\delta_h(k)x_i \tag{2.7}$$

Donde:

- w_i es el valor del peso perteneciente a la neurona de entrada *i*
- η que es el índice de aprendizaje de la red
- $\delta_h(k)$ es el error de la neurona oculta *k* calculado en ecuación 2.6
- x_i es el valor de la entrada *i*

El proceso puede observarse en la Figura 2.7.

Se debe repetir todo este proceso hasta que la diferencia entre los errores por cada iteración baje de un cierto umbral o que se alcancen ciertas iteraciones.

2.3. Árbol de Decisión

El árbol de decisión es una metodología que utiliza decisiones jerárquicas para realizar una clasificación, dichas decisiones están basadas en los atributos que conforman a un conjunto de información [26]. Un árbol se compone de nodos y hojas, cada nodo representa una división cuyo objetivo es maximizar la separación de las diferentes clases existentes en el conjunto de datos, las cuales son representadas por las hojas del árbol. El ejemplo de un árbol puede ser observado en la Figura 2.8.

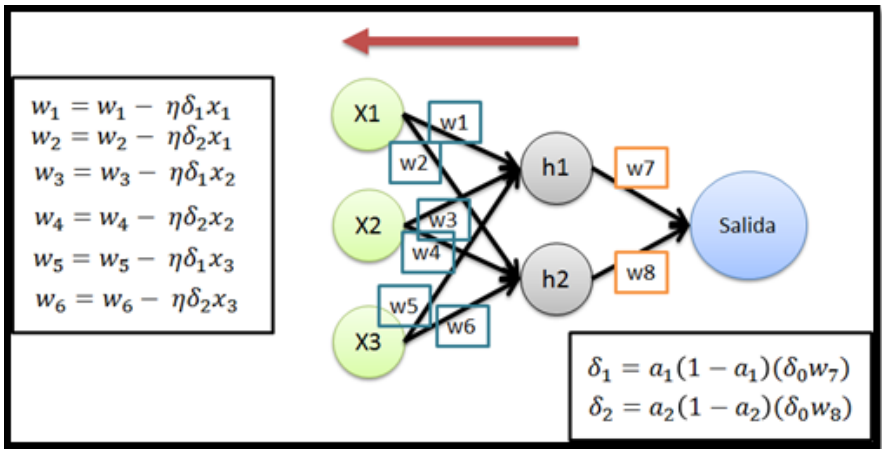


Figura 2.7: Ajuste de pesos en capa oculta.

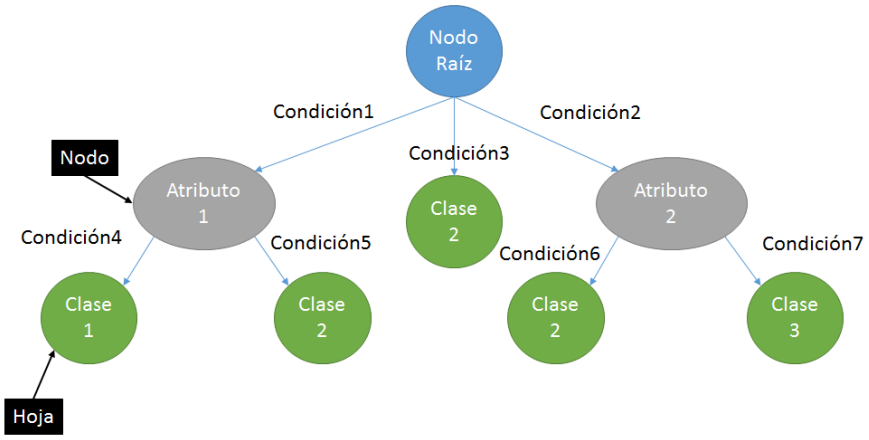


Figura 2.8: Árbol de decisión compuesto de tres dos nodos y 5 hojas. Cada nodo representa ciertas condiciones basadas en los atributos que componen al conjunto de datos, así como cada hoja representa la clasificación de una clase.

Para construir el árbol, primero se debe de construir un nodo raíz, dicho nodo representa al atributo que otorga más información para dividir las diferentes clases contenidas en los datos. Después se prosigue construyendo más nodos hasta que se hayan hecho todas las divisiones posibles en el árbol, los extremos de las ramas del árbol representarán la clase a la que pertenece cada porción del conjunto de datos. Para seleccionar aquellos atributos que otorgan más información se utiliza la ganancia de información expresada en la ecuación 2.9, la cual es calculada en base a la entropía, dicho parámetro indica el porcentaje de impurezas contenidas en un conjunto de atributos. La entropía es calculada con la ecuación 2.8:

$$Entropia(p) = -\sum_{i} p_i log_2 p_i \tag{2.8}$$

Donde:

- *Entropía(p)* es el porcentaje de impurezas en el conjunto de características
- p_i es la probabilidad de un resultado *i*

$$Ganancia(S, F) = Entropia(S) - \sum_{f \in values(F)} \frac{|S_f|}{|S|} Entropia(S_f) \tag{2.9}$$

Donde:

- *Ganancia(S, F)* es la ganancia de información de un conjunto de ejemplos *S* con características *F*
- $Entropia(S)$ Entropía de un conjunto de ejemplos *S*
- $Entropia(S_f)$ Entropía de un conjunto de ejemplos donde los valores de la característica *F* tiene valores *f*

El algoritmo para construir el árbol se puede observar a continuación:

Algorithm 1 Árbol de decisión

Crear un nodo raíz

repeat

Seleccionar un nodo en el árbol

Dividir el nodo seleccionado en dos o más nodos basados en la ecuación 2.9

until No existan más nodos a dividir

Quitar nodos con patrones muy específicos

Etiquetar cada hoja del árbol con su clase dominante

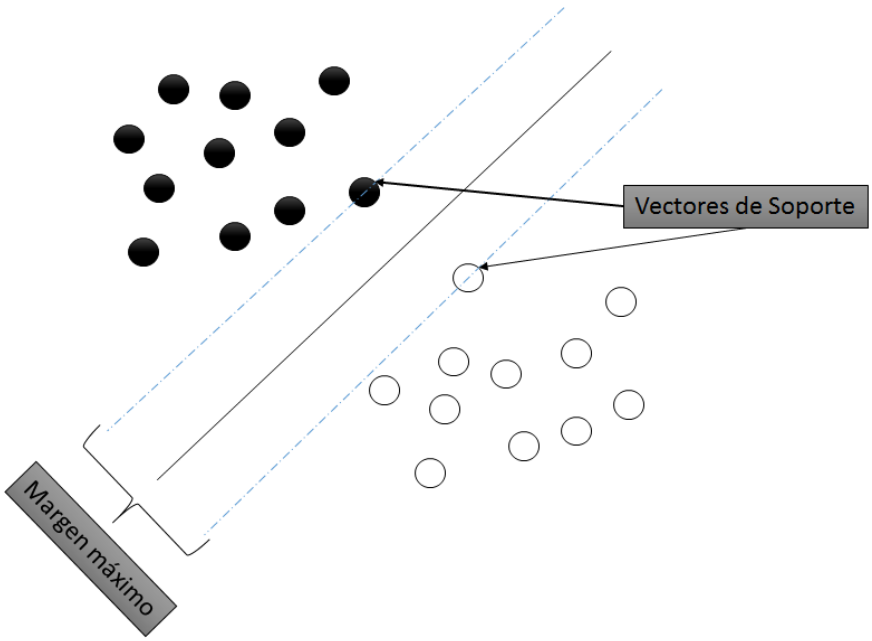


Figura 2.9: Algoritmo SVM separando dos clases de forma lineal. En la imagen se muestra el hiperplano de separación de color negro, así como el margen máximo representado en líneas punteadas.

2.4. Máquina de Vectores de Soporte

La máquina de vectores de soporte (SVM) es una técnica que forma una frontera o hiperplano que divide a diferentes clases en un espacio multidimensional constituido por los atributos del conjunto de datos (Véase Figura 2.9). El objetivo del algoritmo SVM es encontrar aquella división que maximice la separación entre los objetos que conforman diferentes clases, para así poder trabajar de forma correcta en información futura [27].

2.5. Métodos de Ensamble

Los métodos de ensamble surgieron de la hipótesis que dice ”dos cabezas piensan mejor que una”. La idea es que teniendo diversos modelos, los cuales aprendieron diferentes patrones y tienen diferentes resultados, se pueda tomar una mejor decisión uniendo las opiniones de cada uno de los modelos, en lugar de que un solo modelo aprenda todos los patrones dentro de los datos y en base a ello se tome una decisión [26].

Existen diferentes algoritmos dentro de esta categoría, cada uno de estos algoritmos tienen diferentes formas de generar sus modelos y de unir las opiniones de cada una de ellas, los métodos dentro de esta categoría son:

- Boosting
- Bagging
- Random Forest

2.5.1. AdaBoost

Esta técnica se basa en la generación de diversos modelos, cada uno con un aprendizaje muy pobre, pero que al unirlos se puede tener un resultado bastante acertado.

En cada iteración un nuevo clasificador es entrenado con el conjunto destinado a entrenamiento, cada uno de los registros tiene asignado un valor el cual indica qué tan difícil fue clasificarlo en la iteración anterior. Dichos valores son asignados inicialmente con un mismo valor, el cual es 1/*N*, donde *N* es el número de registros contenidos en el conjunto de entrenamiento. Con cada iteración el error ϵ de la clasificación es calculado por medio de la ecuación:

$$\epsilon_t = \sum_{n=1}^{N} w_n^{(t)} I(y_n \neq h_t(x_n)) \tag{2.10}$$

Donde:

- ϵ_t es el error de clasificación del modelo
- $w_n^{(t)}$ es el valor que indica que tan difícil es clasificar el registro *n*, dicho valor es tomado en cuenta si el resultado del clasificador es diferente a la salida real $I(y_n \neq h_t(x_n))$

Después los valores de los registros mal clasificados son actualizados usando la ecuación:

$$w_n^{(t+1)} = w_n^{(t)} \exp(\log(\frac{1-\epsilon}{\epsilon}))/Z_t), \tag{2.11}$$

Donde:

- $w_n^{(t+1)}$ es el nuevo valor del registro que indica que tan difícil es clasificar dicho registro en la tiempo $t+1$
- ϵ es el error de clasificación del modelo
- Z_t es una constante de normalización

El algoritmo termina cuando cierta cantidad de iteraciones es alcanzada o cuando todos los puntos son bien clasificados.

2.5.2. Bagging

Esta técnica consiste en tomar el conjunto de datos y generar *N* cantidad de subconjuntos a partir del mismo, dichos subconjuntos serán del mismo tamaño y serán formados por la elección de registros aleatorios tomados de los datos originales. Después de haber generado dichos subconjuntos se entrenará un modelo por cada uno de ellos.

La ventaja detrás de esta técnica es que se obtendrán diversos clasificadores que aprenderán patrones diferentes y al conjuntar sus resultados por medio de mayoría de votos, se tendrá un mejor resultado.

2.5.3. Random Forest

El algoritmo consiste en la generación de *N* cantidad de árboles, con la hipótesis de que si un árbol trabaja bien, una cantidad más grande de árboles trabajarían mejor. El algoritmo se basa en dos cuestiones: la primera, la cual toma el principio del algoritmo *bagging*, en el cual se toma un subconjunto de registros de la información original para entrenar diferentes árboles y el segundo, el cual decide la profundidad que cada árbol tendrá. Después de entrenar cada uno de los árboles, el resultado final será por mayoría de votos tomando los resultados de todos los árboles generados. El algoritmo puede ser observado a continuación:

Algorithm 2 Algoritmo Random Forest

for cada *N* árboles **do**

Crear un nuevo conjunto de entrenamiento

Usar este conjunto para entrenar un árbol de decisión

Por cada nodo del árbol, seleccionar aleatoriamente *m* atributos y calcular la ganancia de información usando ecuación 2.9.

Seleccionar el mejor atributo

Repetir hasta que el árbol este completo

end for

3

PRE-PROCESAMIENTO DE DATOS

Según la literatura [22, 28, 29, 30, 31], en la actualidad, se generan miles y miles de datos cada minuto producidos por diversos sectores laborales y académicos tales como: ingeniería, negocios, medicina, los cuales son almacenados en algún lugar esperando a ser utilizados.

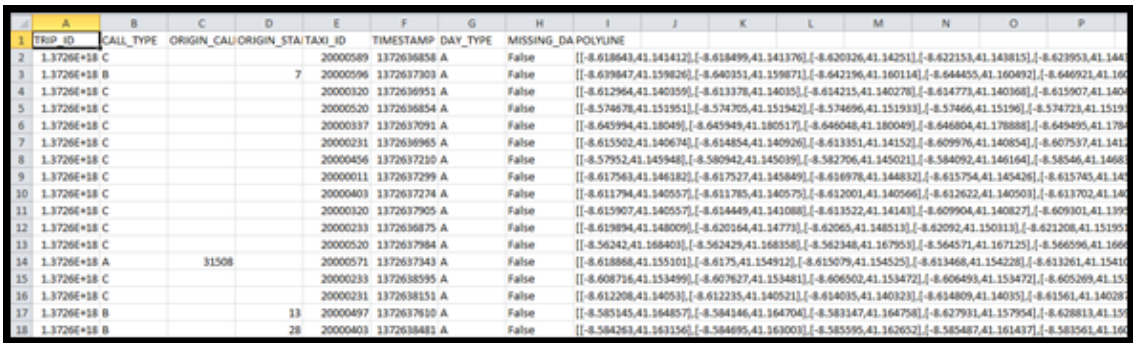
Esta enorme cantidad de datos genera nuevas oportunidades para explotar esta información en busca de información relevante y patrones que podría ser útil para mejorar en gran medida diversas áreas de desarrollo.

Lamentablemente la mayoría de dichos datos contienen ruido, valores faltantes e inconsistencia en su contenido, causados por un mal manejo por parte de los seres humanos, generando así mucha información errónea y por ende, si esta información es utilizada de esta manera, generaría patrones o resultados erróneos.

En este capítulo se mencionarán las inconsistencias encontradas en el conjunto de datos utilizado, además de las estrategias implementadas para mitigarlas. Dichas estrategias fueron utilizadas con el objetivo de obtener un conjunto de datos adecuado para los algoritmos utilizados en esta investigación y obtener un buen modelo de predicción. En la primera sección se describirán todos los atributos por los cuales se compone el conjunto de datos, así mismo en la segunda sección se explicarán aquellos problemas encontrados y las formas de tratar dichos obstáculos.

3.1. Composición del conjunto de datos

El conjunto de datos consiste en 1.7 millones de registros de los cuales se utilizaron 1.5 millones. Dichos registros corresponden a las trayectorias de 442 taxis en la ciudad de Porto, Portugal, en el periodo del 1 de Agosto del 2013 al 30 de Julio del 2014, dicho conjunto de datos está contenido en un archivo con extensión .csv, la composición del conjuntos de datos se puede observar en la Figura 3.1. Los campos de los cuales se compone el conjunto de datos son los siguientes:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	TRIP_ID	CALL_TYPE	ORIGIN_CALL	ORIGIN_STAND	TAXI_ID	TIMESTAMP	DAY_TYPE	MISSING_DATA	POLYLINE							
2	1.3726E+18	C			20000589	1372636858	A	False	[[-8.618643,41.141412],[-8.618499,41.141376],[-8.620326,41.14251],[-8.622153,41.143815],[-8.623953,41.144							
3	1.3726E+18	B		7	20000596	1372637303	A	False	[[-8.639847,41.159826],[-8.640351,41.159871],[-8.642196,41.160114],[-8.644455,41.160492],[-8.646921,41.16							
4	1.3726E+18	C			20000320	1372636951	A	False	[[-8.612964,41.140359],[-8.613378,41.14035],[-8.614215,41.140278],[-8.614773,41.140368],[-8.615907,41.140							
5	1.3726E+18	C			20000520	1372636854	A	False	[[-8.574678,41.151951],[-8.574705,41.151942],[-8.574696,41.151933],[-8.57466,41.15196],[-8.574723,41.1519							
6	1.3726E+18	C			20000337	1372637091	A	False	[[-8.645994,41.18049],[-8.645949,41.180517],[-8.646048,41.180049],[-8.646804,41.178888],[-8.649495,41.178							
7	1.3726E+18	C			20000231	1372636965	A	False	[[-8.615502,41.140674],[-8.614854,41.140926],[-8.613351,41.14152],[-8.609976,41.140854],[-8.607537,41.141							
8	1.3726E+18	C			20000456	1372637210	A	False	[[-8.57952,41.145948],[-8.580942,41.145039],[-8.582706,41.145021],[-8.584092,41.146164],[-8.58546,41.1468							
9	1.3726E+18	C			20000011	1372637299	A	False	[[-8.617563,41.146182],[-8.617527,41.145849],[-8.616978,41.144832],[-8.615754,41.145426],[-8.615745,41.14							
10	1.3726E+18	C			20000403	1372637274	A	False	[[-8.611794,41.140557],[-8.611785,41.140575],[-8.612001,41.140566],[-8.612622,41.140503],[-8.613702,41.14							
11	1.3726E+18	C			20000320	1372637905	A	False	[[-8.615907,41.140557],[-8.614449,41.141088],[-8.613522,41.14143],[-8.609904,41.140827],[-8.609301,41.139							
12	1.3726E+18	C			20000233	1372636875	A	False	[[-8.619894,41.148009],[-8.620164,41.14773],[-8.62065,41.148513],[-8.62092,41.150313],[-8.621208,41.15195							
13	1.3726E+18	C			20000520	1372637984	A	False	[[-8.56242,41.168403],[-8.562429,41.168358],[-8.562348,41.167953],[-8.564571,41.167125],[-8.566596,41.166							
14	1.3726E+18	A		31508	20000571	1372637343	A	False	[[-8.618868,41.155101],[-8.6175,41.154912],[-8.615079,41.154525],[-8.613468,41.154228],[-8.613261,41.1541							
15	1.3726E+18	C			20000233	1372638595	A	False	[[-8.608716,41.153499],[-8.607627,41.153481],[-8.606502,41.153472],[-8.606493,41.153472],[-8.605269,41.15							
16	1.3726E+18	C			20000231	1372638151	A	False	[[-8.612208,41.14053],[-8.612235,41.140521],[-8.614035,41.140323],[-8.614809,41.14035],[-8.61561,41.14028							
17	1.3726E+18	B		13	20000497	1372637610	A	False	[[-8.585145,41.164857],[-8.584146,41.164704],[-8.583147,41.164758],[-8.627931,41.157954],[-8.628813,41.15							
18	1.3726E+18	B		28	20000403	1372638481	A	False	[[-8.584263,41.163156],[-8.584695,41.163003],[-8.585595,41.162652],[-8.585487,41.161437],[-8.583561,41.16							

Figura 3.1: Muestra parcial de los datos mostrada desde una vista de excel, la cual consiste en los primeros 37 registros del *dataset*.

- TRIP_ID: (Cadena) contiene un identificador único para cada viaje.
- CALL_TYPE: (Caracter) Identifica el modo por el cual se demandó el servicio. Contiene tres posibles valores:
  - A.- Viaje despachado desde la central
  - B.- Viaje despachado desde un puesto de taxis (*stand*)
  - C.- En otro caso
- ORIGINAL_ALL: (Numérico) Contiene un indicador único para cada teléfono que haya sido usado para pedir un servicio desde la central. En caso de que el tipo de viaje no sea “A” es nulo.
- ORIGIN_STAND: (Numérico) Contiene un identificador único para cada (*stand*). Identifica el punto inicial del viaje si el tipo es “B”, sino es nulo.
- TAXI_ID: (Numérico) Identificador único para el conductor
- TIMESTAMP: (Numérico) Identifica el comienzo del viaje (en segundos)
- DAYTYPE: (Caracter) Identifica el tipo de día en el que se realizó el viaje. Puede tomar tres valores:
  - B.- Si es una fecha festiva
  - C.- Si es un día antes de un día festivo
  - A.- En otro caso
- MISSING_DATA: (Booleano) Es falso si el viaje está completo, verdadero si existe alguna perdida en los datos

- POLYLINE: (Cadena) Contienen la lista de coordenadas GPS en formato WGS84. Cada par de coordenadas fueron tomadas cada 15 segundos del viaje. El último par corresponde al final del recorrido y el primero al inicio del viaje.

Como se mencionó anteriormente existen registros los cuales tienen perdida de información indicados por el atributo *missing_data*, dichos registros necesitaron ser eliminados para mantener la integridad del *dataset*. A continuación se enumeran varios de los errores encontrados en el conjunto de datos.

3.2. Problemas presentes en el conjunto de datos

3.2.1. Lidiando con viajes vacíos

El conjunto de datos consta de aproximadamente 1.7 millones de viajes, los cuales cada una de sus coordenadas fueron tomadas cada 15 segundos.

Después de realizar un análisis, se encontró que 5,901 viajes pertenecientes al conjunto de datos duraron menos de 15 segundos, por ende dichos registros solo contenían un solo punto que indicaba el punto de partida de cada viaje más no su punto final. Debido a esto se optó por eliminar estos registros ya que no contenían la suficiente información para utilizarlos en el modelo.

La estrategia fue implementada en lenguaje python y consiste en verificar el largo de cada uno de los viajes contenidos dentro del conjunto de datos, si dicho viaje no contiene más de un punto el viaje es eliminado.

3.2.2. Viajes fuera de límite

Todos los viajes contenidos en el conjunto de datos son pertenecientes a la ciudad de Porto, Portugal, después de realizar un análisis se observó que muchos de esos viajes contienen coordenadas erróneas fuera de dicha ciudad generados por el GPS.

Para quitar estos viajes se eligieron las siguientes coordenadas, las cuales pertenecen los extremos de la ciudad, en la Figura 3.2 podemos apreciar el área que abarca dichos límites:

- Máxima Latitud.- 41.470679
- Mínima Latitud.- 41.00394
- Máxima Longitud.- -7.882110
- Mínima Longitud.- -8.777495

Figura 3.2: Área que abarca la ciudad de Porto, Portugal, correspondientes a las coordenadas 41.470679 y 41.00394 de máxima y mínima Latitud, así como -7.882110 y -8.777495 de máxima y mínima Longitud.

La estrategia para la eliminación de puntos fuera de límite, consistió en recorrer cada una de las coordenadas contenidas dentro del viaje, si alguna de dichas coordenadas no estaba dentro del rango elegido, todo el viaje era eliminado, como resultado se excluyeron 2,806 viajes del conjunto de datos; el resultado se puede apreciar en la Figura 3.3:

3.2.3. Errores en la trayectoria del viaje

El uso de aparatos como el GPS trae consigo una pequeña cantidad de errores debido a diversos factores como desviaciones de la señal por la atmósfera de la tierra, errores de reloj del receptor y satélite, dilución geométrica, entre otros factores [1], debido a ello fue necesario realizar un análisis exhaustivo sobre las rutas para encontrar errores dentro de dichas trayectorias.

Después del análisis realizado se detectaron ciertas discordancias en las rutas, las cuales contenían coordenadas que afectaban a la trayectoria causando ruido en ella y por ende era muy posible que resultara en patrones falsos o errores en el aprendizaje del modelo. El ruido contenido dentro de las rutas puede ser observado en la Figura 3.4, donde se muestran los errores encontrados en algunas rutas.

Para poder mitigar estos errores fue necesario implementar una estrategia que pudiera detectar aquellas coordenadas anómalas dentro de la ruta, para ello se prosiguió a seguir cada una de las coordenadas dentro de la ruta, por cada par de coordenadas en la secuencia se medió la distancia euclidiana en kilómetros, si la

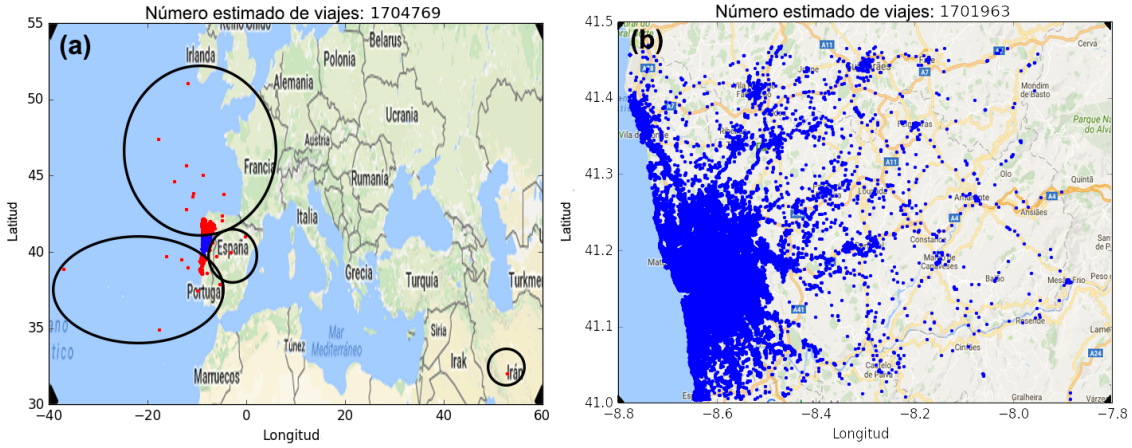


Figura 3.3: Resultado de viajes después de aplicar límites de la cuidad. De lado izquierdo (a) podemos observar todos los viajes contenidos en el conjunto de datos, en los cuales están de color rojo los viajes fuera de límites, así también se encuentran encerrados para mejor apreciación. De lado derecho (b) se encuentran los viajes dentro de los límites marcados

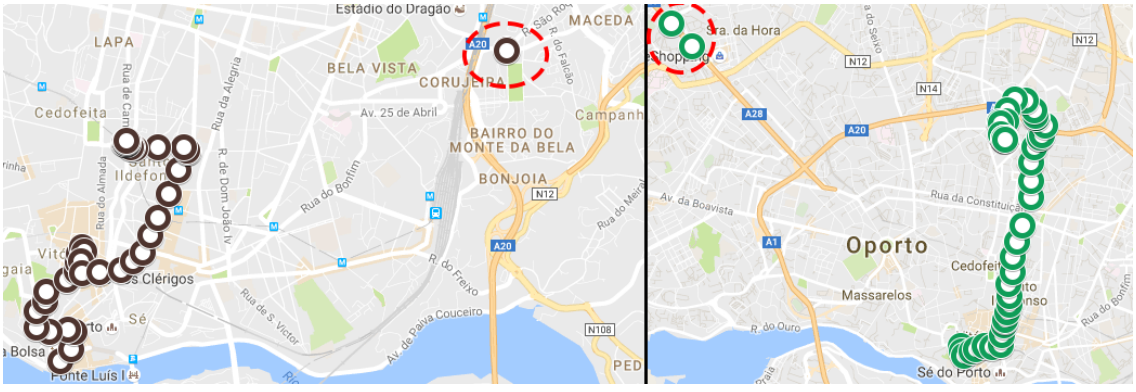


Figura 3.4: Ejemplo de trayectorias tomadas desde diferentes partes de la ciudad de Porto, Portugal, las cuales contienen errores dentro de ellas, dichos errores se encuentran encerrados en color rojo

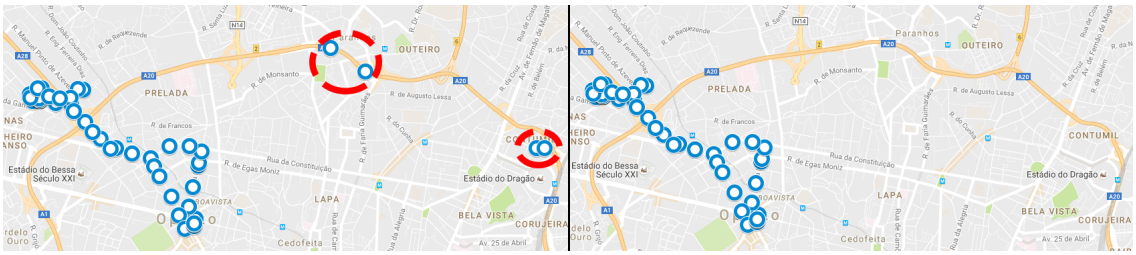


Figura 3.5: Resultado después de haber eliminado los errores en las trayectorias. De lado izquierdo se muestra la trayectoria antes de aplicar la estrategia de depuración y de lado derecho se muestra la trayectoria ya depurada.

distancia es mayor a 5 Km la coordenada es eliminada de la trayectoria, el resultado se puede observar en la Figura 3.5:

4

ESTRATEGIA MALLA Y COMPARACIÓN DE MODELOS

El siguiente paso después de haber realizado el pre-procesamiento de los datos (véase Capítulo 3) es la formulación de la estrategia que identifique los patrones dentro de la información obtenida y realice las operaciones necesarias para poder realizar la predicción del destino del viaje en automóvil.

En este capítulo se explicará la primera estrategia que se implementó para la predicción de destinos utilizando diferentes modelos de aprendizaje máquina. En la primera sección se explicará el proceso de investigación y elección de modelos para realizar los experimentos, así como la explicación de la estrategia implementada. En la segunda sección se describe el pre-procesamiento de los datos que se realizó para que los modelos pudieran manejar el conjunto de datos. A continuación se mostrarán los resultados obtenidos por cada uno de los modelos y se realizarán comparaciones entre ellos, por último se abordará el análisis de los resultados de este capítulo y cómo dichos resultados permitieron idear nuevas estrategias para atacar el problema.

4.1. Generación de estrategia y análisis de algoritmos

La estrategia implementada surge de la suma de principios ”divide y vencerás” y ”dos cabezas piensan mejor que una”. Debido a la inmensa cantidad de registros contenidos dentro del conjunto de datos es de suponer que existe una innumerable cantidad de patrones, los cuales muy probablemente se encuentran dispersos dentro de la ciudad, un solo modelo podría tener dificultades para lidiar con todos estos patrones, resultando en predicciones falsas o influidas por otros patrones. Es por ello que se tomó la decisión de asignar varios modelos que lidiaran con varias secciones de la ciudad, después se utilizarían los resultados de dichos

Figura 4.1: Segmento elegido para prueba de estrategia. El área elegida se encuentra encerrada en color rojo, la cual representa la sección con más aglomeración de viajes dentro de la ciudad de Porto, Portugal.

modelos para poder obtener un mejor resultado, algo parecido a lo que realizan los métodos de ensamble en su funcionamiento (véase sección 2.5).

Lamentablemente existe una gran dispersión de los viajes aún dentro del área elegida (véase sección 3.2.2), esto imposibilita en gran medida la implementación de la estrategia propuesta ya que dificulta la elección correcta de los cuadrantes para cumplir con un balance en los viajes y patrones en cada uno de ellos. Para lidiar con este problema se eligió un área representativa de los datos en la cual se contenía la mayor aglomeración de destinos de viajes, para probar la estrategia y conocer si es viable usarla en un área con mayor dispersión. El área elegida se puede observar en la Figura 4.1, dicha sección se encuentra limitada por las siguientes coordenadas:

- Máxima Altitud.- 41.17894463
- Mínima Altitud.- 41.12059775
- Máxima Longitud.- -8.554285
- Mínima Longitud.- -8.66589

La elección de las áreas donde se encuentran los modelos y su distribución, fue basada en el uso de un plano cartesiano que abarca la porción elegida de la ciudad. Se dividió la ciudad en 4, 16 y 32 secciones

Figura 4.2: Segmentación de la ciudad de Porto, Portugal en 4, 16 y 32 cuadrantes respectivamente.

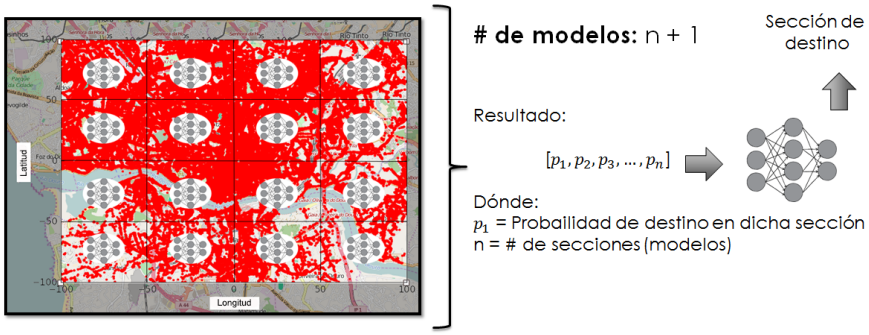


Figura 4.3: Metodología de secciones para predicción de zona de destino. Dicha metodología consiste en el uso de varios modelos para la obtención de un solo resultado, mismo resultado que es utilizado por una red neuronal para la obtención del área más probable para el destino del viaje.

rectangulares a lo largo de toda la sección elegida (Figura 4.2), las cuales llamaremos cuadrantes. Cada cuadrante, como mencionamos anteriormente, contienen un modelo que es entrenado solamente con los viajes que inician dentro del mismo, como cada uno de dichos viajes puede o no terminar dentro del cuadrante, cada modelo es entrenado para predecir esta información. Al final los resultados de todos los modelos son utilizados por una red neuronal, de la cual se obtiene el cuadrante más probable para ser el destino del viaje. La estrategia ejemplificada se puede observar en la Figura 4.3.

Al haber ideado la estrategia a implementar, se procedió a realizar un análisis para recolectar información sobre los algoritmos que fueran adecuados para atacar este problema. La primera actividad que se realizó fue la recolección de artículos científicos que hablaran sobre algoritmos utilizados para mitigar este tipo de problemas (véase Capítulo 1), así mismo se obtuvo información de varias fuentes para encontrar ventajas y desventajas de cada uno de los algoritmos analizados. El resultado de dicha investigación puede ser encontrada en las Tablas 4.1, 4.2 y 4.3. En [27, 26] se puede encontrar información más detallada de los algoritmos.

Después de un análisis exhaustivo se decidió implementar 6 algoritmos de aprendizaje máquina para la estrategia de cuadrantes, dichos algoritmos han mostrado buenos resultados en materia de clasificación atacando problemas parecidos. Dichos algoritmos fueron red neuronal, métodos de ensamble (Adaboost,

Tabla 4.1: Comparación de algoritmos.

Algoritmos	Fortalezas	Debilidades
Red Neuronal	▪ Puede ser adaptada a clasificación o regresión ▪ Está entre los modelos más precisos ▪ Hace estimaciones en base a la relación que tienen los datos	▪ Es costosa computacionalmente en fase de entrenamiento ▪ Es susceptible al ruido, haciendo que el modelo aprenda patrones incorrectos, (también conocido como sobre-entrenamiento) ▪ No se puede interpretar el funcionamiento interno tan fácilmente.
Árboles de decisión	▪ Puede trabajar bien con pérdida de datos ▪ Trabaja bien con grandes y pequeñas entradas de datos ▪ El funcionamiento del modelo puede ser comprendido fácilmente	▪ Es susceptible al ruido, haciendo que el modelo aprenda patrones incorrectos, (también conocido como sobre-entrenamiento) ▪ Puede ser difícil modelar ciertas relaciones ▪ Cambios pequeños en los datos de entrenamiento pueden resultar cambios grandes en la lógica de decisión

Tabla 4.2: Comparación de algoritmos.

Algoritmos	Fortalezas	Debilidades
HMM	▪ Maneja bien la incertidumbre en el conocimiento del estado actual ▪ Nos permite obtener una secuencia de estados ▪ La implementación y ejecución no es muy costosa computacionalmente	▪ Está limitada a utilizar pocos estados y pocas observaciones ▪ Trabaja bien con secuencias cortas pero se ve reducido su rendimiento al trabajar con secuencias largas. ▪ Depende mucho de las matrices de transiciones y confusión
Algoritmo Genético	▪ Poderoso para problemas de optimización ▪ Fácil de implementar y de entender su funcionamiento ▪ Puede trabajar con varios objetivos	▪ No se han usado muchas implementaciones en problemas estocásticos ▪ Difícil de crear una función objetivo adecuada para este problema

Tabla 4.3: Comparación de algoritmos.

Algoritmos	Fortalezas	Debilidades
Métodos de Ensamble	▪ Pueden ser usados en datos con un gran número de registros o de características ▪ Modelos que trabaja bien para la mayoría de los problemas ▪ Pueden manejar ruido en los datos y valores faltantes	▪ No es tan fácil de interpretar ▪ Pueden requerir de cierta configuración de parámetros para que los modelos trabajen adecuadamente con los datos
SVM	▪ Puede ser usado para clasificación o predicción numérica ▪ El modelo no se ve influenciado por ruido en los datos ▪ Regularmente maneja altos porcentajes de precisión	▪ Puede ser lento de entrenar, especialmente si se manejan muchas características o registros ▪ En ocasiones resulta en un modelo de caja negra complicado de interpretar

Bagging, Random Forest), SVM y Árbol de decisión.

4.2. Procesamiento de datos para entrada de modelo

Habiendo elegido los modelos, se tiene que analizar la forma que deben tener los datos para que el modelo pueda recibir dicho conjunto y pueda dar resultados fiables. En esta estrategia solamente se utilizó la trayectoria del viaje con el objetivo de crear un modelo general que pudiera ser utilizado en cualquier ambiente sin necesidad de utilizar información adicional que pudiera no ser recolectada. Como mencionamos anteriormente (véase Capítulo 3), la trayectoria tiene una longitud diferente para cada viaje, razón por la cual esta debe ser transformada a una longitud fija para que los modelos puedan aprender de ella, es por ello que fue necesario elegir un criterio de selección de coordenadas en cada uno de los viajes.

4.2.1. Criterio para reducción de trayectoria

Suponga un escenario en el cual la central de taxis quiere saber cuál es el vehículo más indicado para mandar a cierta zona donde existe demanda de transporte. La central tiene conocimiento de las rutas de todos los taxis en un tiempo determinado, es decir algunos taxis apenas habrán iniciados sus trayectorias, otros probablemente ya estarán por terminar su viaje. Lamentablemente estos nos dejaría con trayectorias de diferentes longitudes, las cuales no es posible usarlas como vector de entrada en los modelos debido a que todos ellos necesitan de vectores de características con longitud fija. Un ejemplo de este escenario puede ser observado en la Figura 4.4.

Para lidiar con este problema es necesario reducir las trayectorias, pero es necesario saber que coordenadas elegir de cada una de ellas para tomar la mayor información posible; si se eligiera solamente el inicio, muchas de las trayectorias serían parecidas y por ende las predicciones no serían fidedignas, si se seleccionara solamente el final de un cierto punto caeríamos en el mismo problema, dicho segmento pudiera ser igual en muchos viajes y el modelo se confundiría; además de ello perderíamos valiosa información sobre tendencias inicio - fin, es por ello que se tomó la decisión de juntar estos dos segmentos del viaje, el inicio y las últimas coordenadas de un punto aleatorio elegido dentro de la trayectoria. En [25] se utilizan 10 coordenadas totales, de las cuales 5 son las primeras coordenadas del viaje y las otras 5 se conforman de las últimas 5 coordenadas de un punto intermedio en la trayectoria; dicha elección dio los mejores resultados e indican que es la combinación que proporciona mayor aporte a este tipo de modelos. En la Figura 4.5 podemos observar la elección de las coordenadas.

Para reducir los viajes dentro del conjunto de datos se implementó un algoritmo en lenguaje python, el

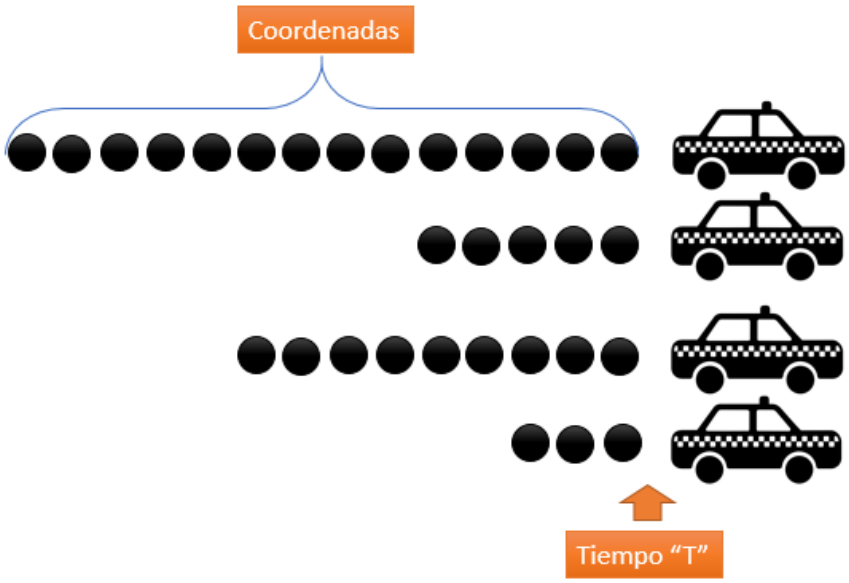


Figura 4.4: Ejemplo de un escenario real dónde la central verifica las trayectorias de diferentes taxis, las cuales cada una de ellas tendría una tendencia a tener longitud diferente de las otras.

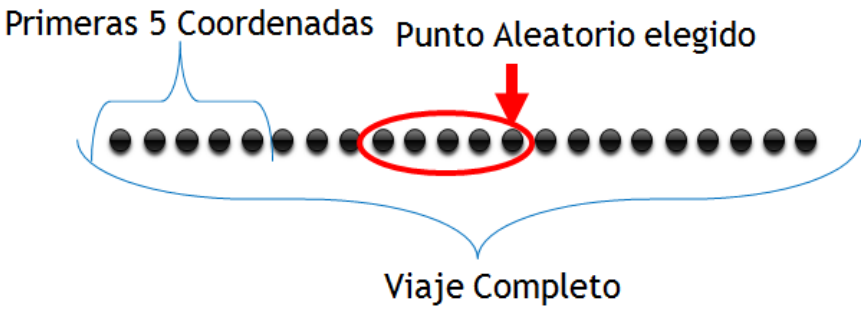


Figura 4.5: Estrategia de selección de coordenadas. La cual elige las primeras 5 coordenadas del viaje y las últimas 5 coordenadas de un punto aleatorio elegido dentro del mismo.

```python
for i in range(len(matrizCoordenadas)):
    j = 0
    #Elección de punto aleatorio dentro de la trayectoria
    puntoAleatorio = random.randint(9,len(matrizCoordenadas[i]) - 2)

    #Punto donde inicial las últimas 5 coordenadas del punto aleatorio
    k = (len(matrizCoordenadas[i]) - puntoAleatorio) + 4
    index = 0

    #Creacion de vector de caracteristicas
    while (j < 11):
        if j < 5:
            #data
            allData[i][index] = matrizCoordenadas[i][j][0]
            allData[i][index + 1] = matrizCoordenadas[i][j][1]
            index+=2
        if j >= 5 and j < 10:
            #data
            allData[i][index] = matrizCoordenadas[i][-k][0]
            allData[i][index + 1] = matrizCoordenadas[i][-k][1]
            index+=2
            k -=1

        if j == 10:
            #target
            allTarget[i] = matrizCoordenadas[i][-1]
        j+= 1
```

Figura 4.6: Algoritmo utilizado para la reducción de trayectorias implementado en lenguaje Python.

cual como se puede observar en la Figura 4.6 elige una coordenada aleatoria dentro del viaje para simular la consulta de la central de taxis en un determinado momento, así mismo como se eligieron 10 coordenadas para formar el vector de características, todos los viajes con longitud menor a esta cifra fueron descartados del conjunto de datos.

Adicional a la elección de coordenadas, se transformó el conjunto de datos de dos formas; primero se transformaron todas las coordenadas de los viajes a coordenadas de plano cartesiano, y segundo se realizó una normalización de los datos. La transformación de las coordenadas GPS a coordenadas cartesianas fue realizada debido a que las coordenadas GPS manejan cantidades de 6 decimales, lo cual posiblemente imposibilitaría al modelo encontrar los patrones adecuadamente así como el manejo de las cifras en ciertos lenguajes de programación. Al realizar dicha modificación nos permite observar de mejor manera el espacio donde se encuentran los viajes y facilita el manejo de los datos en los modelos elegidos. Las dimensiones utilizadas en el plano cartesiano en cada uno de sus ejes es de 100 unidades, las ecuaciones utilizadas para transformar cada una de las coordenadas pueden ser observadas a continuación:

$$Coordenada = \frac{Coordenada - ValorMedio}{Proporcion} \tag{4.1}$$

$$ValorMedio = \frac{(ValorMaximo - ValorMinimo)}{2} + ValorMinimo \tag{4.2}$$

$$Proporcion = \frac{(ValorMaximo - ValorMedio)}{100} \tag{4.3}$$

4.2.2. Normalización de vectores

La normalización de los datos ayuda a que todos los atributos tengan un peso igualitario al entrenar los modelos, así como transformar todas las cantidades de los atributos en rangos más pequeños y más manejables para los modelos, en la literatura se indica que modelos como redes neuronales, SVM y métodos de ensamble trabajan de una mejor manera y apoya en la etapa de aprendizaje si los datos están normalizados. Para normalizar los datos se restó la media de cada dato por columna y se dividió cada columna entre el valor máximo de la misma. Las ecuaciones utilizadas pueden ser observadas a continuación:

$$Dato = \frac{Dato - Media}{ValorMaximoColumna} \tag{4.4}$$

$$Media = \frac{ValorMaximo - ValorMinimo}{2} \tag{4.5}$$

4.3. Clasificación de cuadrante destino

Como se mencionó anteriormente se utilizaron 6 modelos de aprendizaje máquina utilizando la estrategia explicada anteriormente (sección 4.1). Dichos algoritmos son:

- Red neuronal
- Máquina de vectores de soporte (SVM)
- Árbol de decisión
- Métodos de Ensamble
    - *Adaboost*
    - *Bagging*
    - *Random Forest*

Para el caso de la red neuronal se implementaron 3 estrategias las cuales son:

- Usar los resultados de las redes como un vector binario (1 y 0)
- Usar un vector con los porcentajes que arrojó cada red
- Usar el mayor de los porcentajes arrojado por las redes como resultado.

4.3.1. Configuración de los modelos

La arquitectura usada en la red neuronal consiste en un modelo de 3 capas (capa entrada, oculta y salida), usando 5 neuronas ocultas y la función *softmax* definida por $x_i = \frac{exp(x_i)}{\sum_{j=1}^{C} exp(x_j)}$, donde x_i es el valor normalizado de todas las neuronas después de la fase de entrenamiento y x_j es el valor de cada una de las neuronas en la misma capa (Véase sección 2.2.5). Para el caso de los métodos de ensamble se utilizaron 100 estimadores sin límite de profundidad y para el clasificador SVM se utilizó como parámetros *gamma* = 2 y *C* = 1. Los parámetros fueron seleccionados después de realizar 5 corridas de 30 ejecuciones cada una, eligiendo después del experimento, los parámetros que obtuvieron mejores resultados.

4.4. Resultados

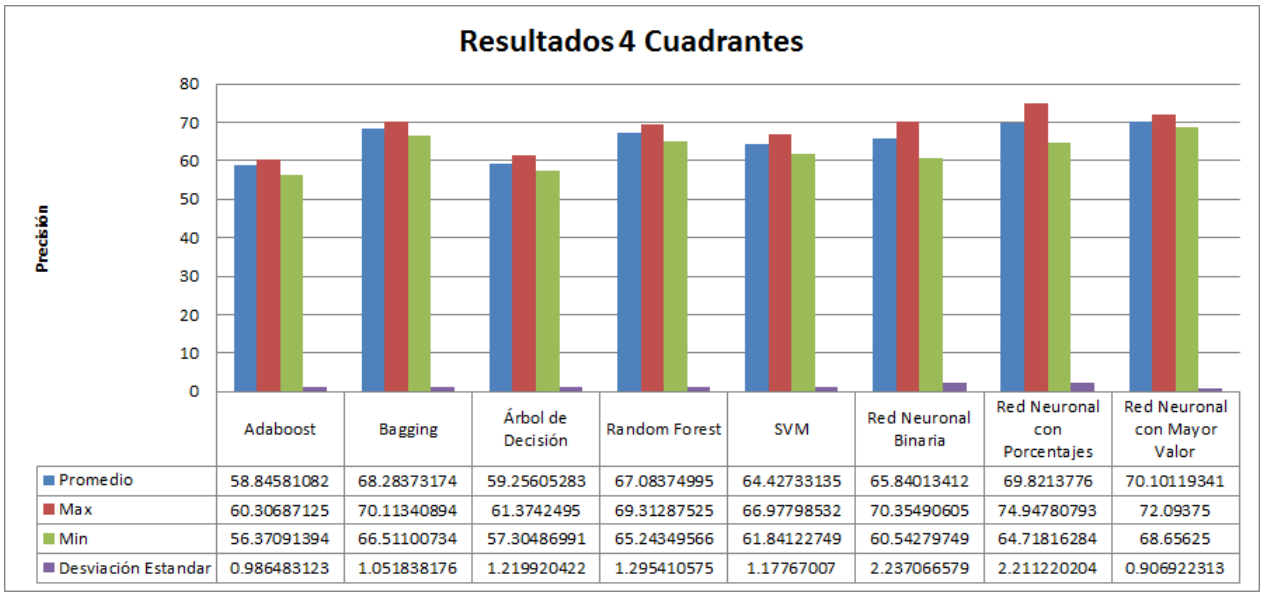
Como se mencionó en la sección 4.1, los resultados que se presentan en este apartado por cada algoritmo consisten en la generación de *N* modelos dependiendo de la cantidad de cuadrantes en los cuales se haya dividido la ciudad. Cada uno de dichos modelos es ejecutado 30 veces eligiendo para cada una de dichas ejecuciones un conjunto diferentes de entrenamiento y testeo (70 % y 30 % respectivamente). Cada uno de los *N* modelos es binario, por tanto al final se tendrá un nuevo vector de ceros y unos los cuales serán utilizados como entrada en una red neuronal; la predicción de dicho algoritmo es usado como resultado y la evaluación de dicho resultado se realiza por medio de la matriz de confusión cuyo porcentaje es obtenido de la siguientes manera:

$$Precision = \frac{VerdaderosPositivos + VerdaderosNegativos}{TotalInstancias} \tag{4.6}$$

El promedio de las 30 ejecuciones de cada modelo es usado como resultado representativo así como referencia para comparar que tan buenos son los algoritmos para atacar este problema. En seguida se muestran los resultados obtenidos por cada uno de los modelos usando diferentes cuadrantes.

4.4.1. Resultado de modelos utilizando 4 cuadrantes

Los resultados de los modelos utilizando 4 cuadrantes pueden ser observados en la Figura 4.7.

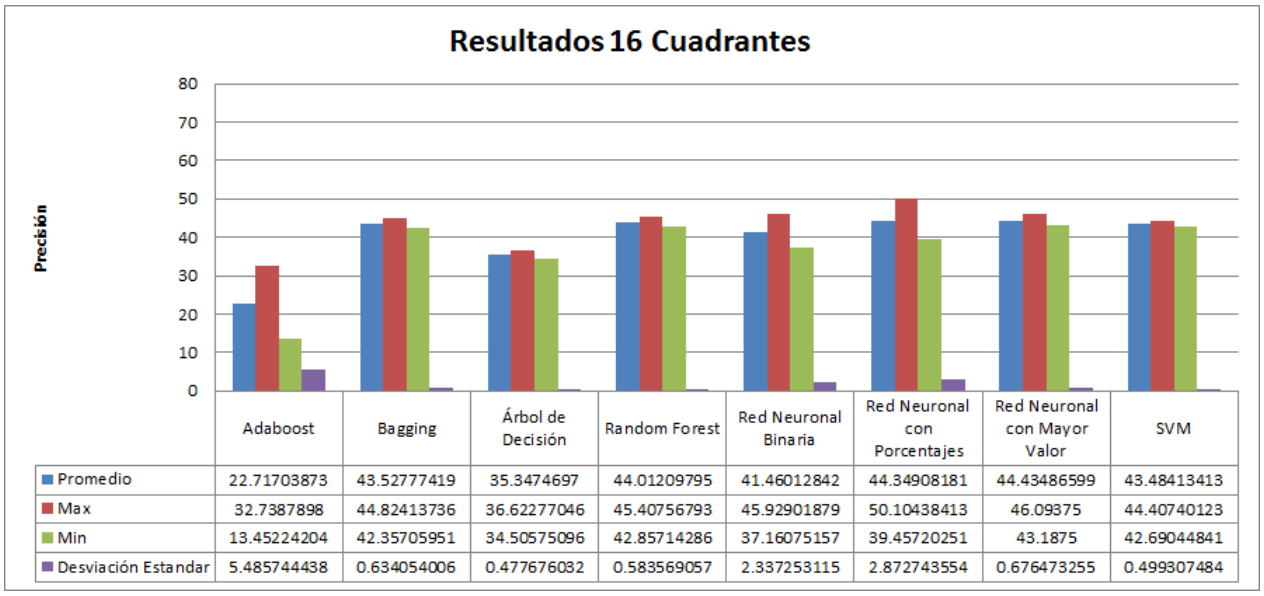


	Adaboost	Bagging	Árbol de Decisión	Random Forest	SVM	Red Neuronal Binaria	Red Neuronal con Porcentajes	Red Neuronal con Mayor Valor
Promedio	58.84581082	68.28373174	59.25605283	67.08374995	64.42733135	65.84013412	69.8213776	70.10119341
Max	60.30687125	70.11340894	61.3742495	69.31287525	66.97798532	70.35490605	74.94780793	72.09375
Min	56.37091394	66.51100734	57.30486991	65.24349566	61.84122749	60.54279749	64.71816284	68.65625
Desviación Estandar	0.986483123	1.051838176	1.219920422	1.295410575	1.17767007	2.237066579	2.211220204	0.906922313

Figura 4.7: Resultados de cada algoritmo dividiendo la ciudad en 4 cuadrantes.

4.4.2. Resultado de modelos utilizando 16 cuadrantes

Los resultados de los modelos utilizando 16 cuadrantes pueden ser observados en la Figura 4.8.

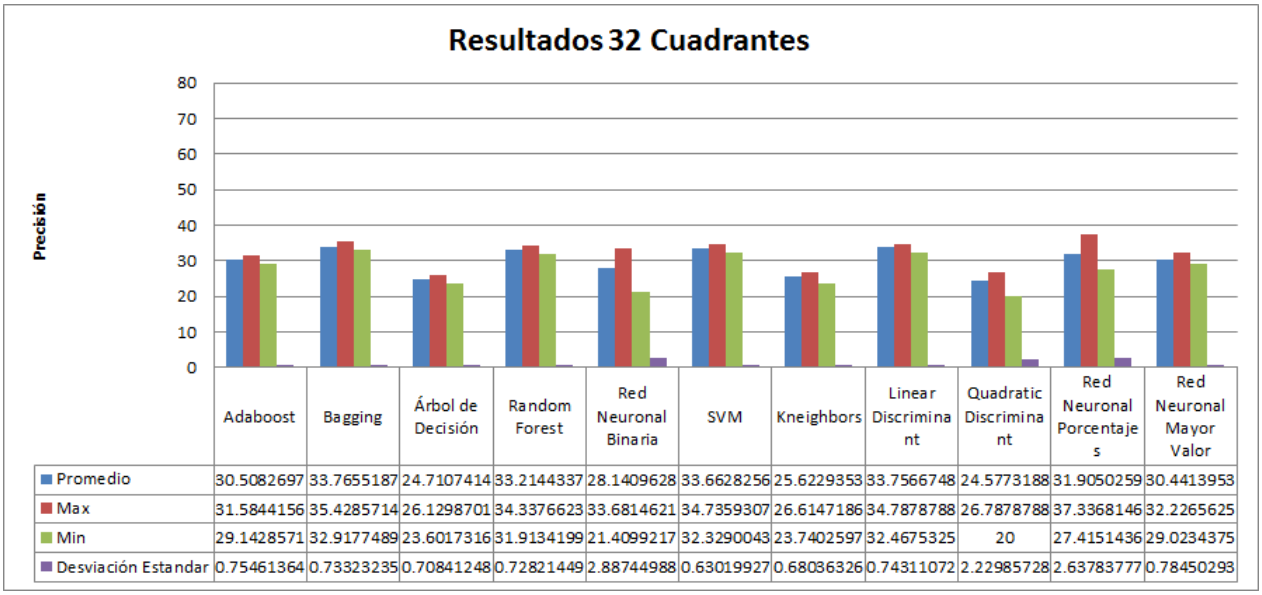


	Adaboost	Bagging	Árbol de Decisión	Random Forest	Red Neuronal Binaria	Red Neuronal con Porcentajes	Red Neuronal con Mayor Valor	SVM
Promedio	22.71703873	43.52777419	35.3474697	44.01209795	41.46012842	44.34908181	44.43486599	43.48413413
Max	32.7387898	44.82413736	36.62277046	45.40756793	45.92901879	50.10438413	46.09375	44.40740123
Min	13.45224204	42.35705951	34.50575096	42.85714286	37.16075157	39.45720251	43.1875	42.69044841
Desviación Estandar	5.485744438	0.634054006	0.477676032	0.583569057	2.337253115	2.872743554	0.676473255	0.499307484

Figura 4.8: Resultados de cada algoritmo dividiendo la ciudad en 16 cuadrantes.

4.4.3. Resultado de modelos utilizando 32 cuadrantes

Los resultados de los modelos utilizando 32 cuadrantes pueden ser observados en la Figura 4.9.

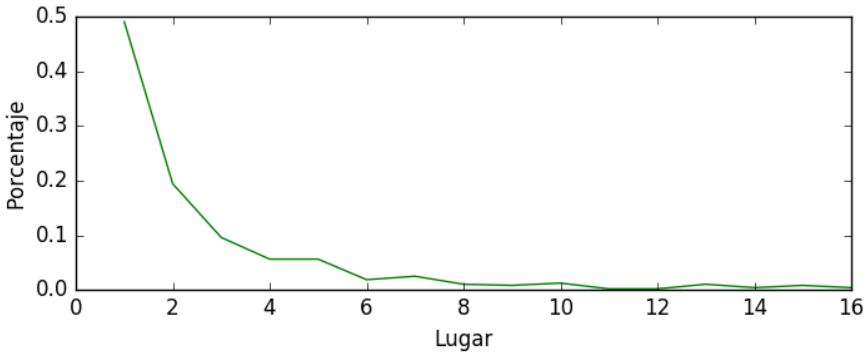


	Adaboost	Bagging	Árbol de Decisión	Random Forest	Red Neuronal Binaria	SVM	Kneighbors	Linear Discriminant	Quadratic Discriminant	Red Neuronal Porcentajes	Red Neuronal Mayor Valor
Promedio	30.5082697	33.7655187	24.7107414	33.2144337	28.1409628	33.6628256	25.6229353	33.7566748	24.5773188	31.9050259	30.4413953
Max	31.5844156	35.4285714	26.1298701	34.3376623	33.6814621	34.7359307	26.6147186	34.7878788	26.7878788	37.3368146	32.2265625
Min	29.1428571	32.9177489	23.6017316	31.9134199	21.4099217	32.3290043	23.7402597	32.4675325	20	27.4151436	29.0234375
Desviación Estandar	0.75461364	0.73323235	0.70841248	0.72821449	2.88744988	0.63019927	0.68036326	0.74311072	2.22985728	2.63783777	0.78450293

Figura 4.9: Resultados de cada algoritmo dividiendo la ciudad en 32 cuadrantes.

4.5. Análisis de resultados

Para evaluar los resultados y observar el verdadero comportamiento de los modelos, se procedió a la generación de una gráfica con la distribución de los cuadrantes verdaderos en términos de porcentajes otorgada por los mismos. La distribución observada en la Figura 4.10 muestra, que la mayoría de las veces el clasificador posiciona en los primeros 5 lugares al cuadrante correcto, es decir, aquel cuadrante donde se encuentra el destino del viaje (lo cual representa el 87 % del total de las predicciones). Esto demuestra que a pesar los bajos porcentajes de predicción mostrados en las gráficas de la sección anterior, la estrategia malla tiene un alto potencial de mejora. Es probable que realizando algunos cambios en el acomodo de los cuadrantes o ideando alguna estrategia para mitigar el error que posiciona al cuadrante correcto del segundo al cuarto lugar, se podría mejorar bastante la metodología implementada en este apartado. Dichas mejoras quedan como trabajo futuro y no competen a este trabajo debido a que se implementaron nuevas estrategias que serán explicadas posteriormente.



Lugar	1	2	3	4	5	6	7	8
Porcentajes	0.490605	0.194154	0.096033	0.056367	0.056367	0.018789	0.025052	0.010438
Lugar	9	10	11	12	13	14	15	16
Porcentajes	0.008351	0.012526	0.002088	0.002088	0.010438	0.004175	0.008351	0.004175

Figura 4.10: Comportamiento de la red neuronal con 16 cuadrantes.

5

RED NEURONAL COMO ESTRATEGIA DE SOLUCIÓN

A pesar de que la estrategia malla presentaba resultados prometedores en cuanto a la predicción del área más probable para el destino del viaje, el resultado del modelo es una sección muy grande y probablemente no sirva para fines más específicos como se mencionó en el capítulo de introducción, en donde podría ser necesaria una ubicación exacta o al menos aproximada para tomar decisiones en ciertos casos.

En este capítulo se explicará a detalle la implementación de una red neuronal que obtiene como resultado las coordenadas aproximadas del destino del viaje, así como el análisis de los resultados y mejoras futuras que se pudieran realizar. En la primera sección se explicará la estructura general del modelo y la estrategia de obtención de coordenadas. En la segunda sección se muestran las diferencias en la elección de *clusters*. Por último se abordará la comparación entre un modelo entrenado con los registros de varios usuarios, conocido como modelo general y modelos creados en base a registros de un solo usuario, conocidos como modelos individuales.

5.1. Estructura del modelo

La estructura de este modelo fue obtenido del equipo ganador de la competencia *ECML/PKDD discovery challenge on taxi destination prediction* [25], en dicho trabajo se implementa varios modelos de redes neuronales para la predicción del destino aproximado del viaje, es decir sus coordenadas más probables. El modelo utilizado consta de tres capas, una capa de entrada, una capa oculta y una capa de salida, que posteriormente es utilizada en una operación de producto punto para obtener las coordenadas más probables.

Cómo se mencionó en la sección 4.2.1, el vector de entrada se compone primeramente de las primeras 5

Feature	Amount
Taxi_id	448
Origin_call	57,106
Origin_stand	64
Day of the week	7
Week of the year	52
Quarter hour of the day	96

Tabla 5.1: Valores únicos de cada atributo

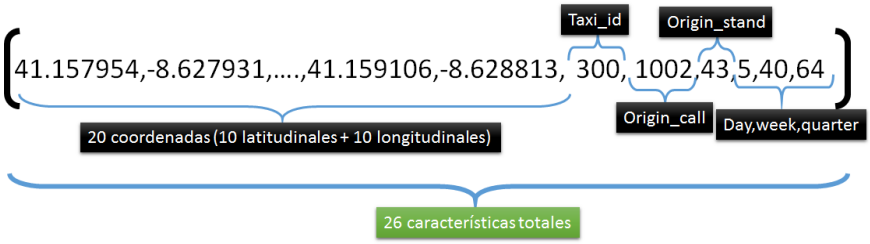


Figura 5.1: Formato del vector de características utilizado.

coordenadas y las últimas 5 coordenadas de un punto elegido aleatoriamente. En este modelo se adicionaron las características que acompañaban a la trayectoria (véase sección 3.1), dando como resultado un total de 26 neuronas en la capa de entrada; dichos atributos fueron analizados y se remplazó el valor de cada uno por un número consecutivo correspondiente a su posición en un arreglo de valores únicos, en el caso del atributo "*timestamp*", se dividió en día, semana del año y cuarto de hora tal como lo realizaron en el trabajo mencionado anteriormente. La cantidad de valores únicos de cada atributo puede observarse en la Tabla 5.1. El formato del vector de entrada puede observarse en la Figura 5.1.

Cada uno de los vectores conformados por cada registro es normalizado de la misma manera que en la sección 4.2.2, dicho proceso fue realizado para disminuir el peso que pudieran tener algunos atributos causado por su rango de valores.

La capa oculta está compuesta por 500 neuronas ReLu (*Rectifier Linear Units*), las cuales son neuronas cuya función de evaluación es $max(0, x)$, en dicha función en valor de la neurona se mantiene en 0 si el valor de la neurona no sobrepasa un umbral, de lo contrario se conserva el valor. En [32] se puede encontrar información detallada al respecto.

En [25] se menciona que el predecir la ubicación final sin ninguna información *apriori* da como resultado ruido en la predicción y resultados alejados de la ubicación real del destino, es por ello que la capa de salida

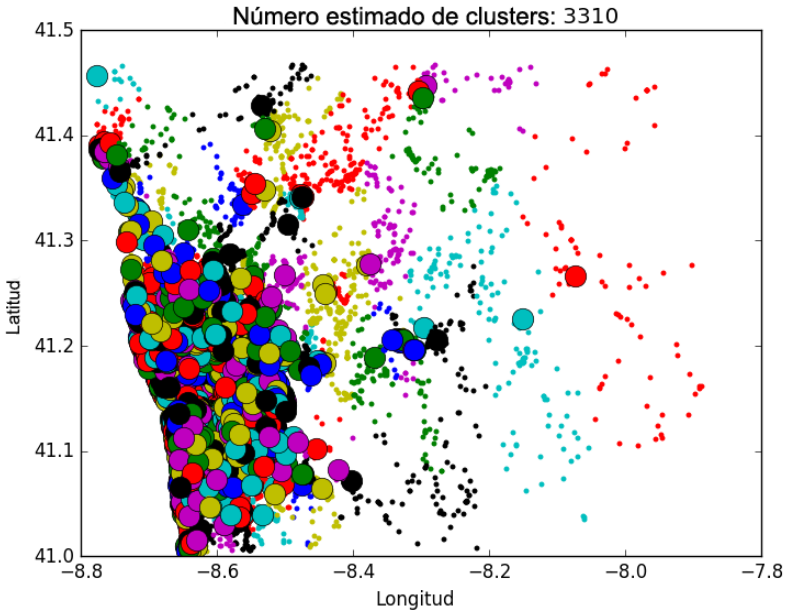


Figura 5.2: *Clusters* obtenidos con el algoritmo *mean-shift* dentro de la ciudad de Porto. Los *clusters* están representados con círculos, los destinos pertenecientes a cada destino están marcados con el color correspondiente a cada *clusters*.

está compuesta por neuronas que representan *clusters* creados en base a todos los destinos del conjunto de datos. Para definir las dimensiones de la capa de salida, es decir la obtención de las coordenadas de los *clusters*, se utilizó el algoritmo *mean-shift*, dicho algoritmo trabaja bien para datos dispersos y con gran volumen, cosa que para otros algoritmos como *K-means* sería demasiado intensivo de calcular. Se utilizaron todas las coordenadas del conjunto de datos depurado como se mencionó en el capítulo 3, dando como resultado un total de 3310 *clusters*. En la Figura 5.2 se muestra la distribución de dichos *clusters* en la ciudad de Porto, Portugal.

La salida de la red neuronal es un vector que contiene la probabilidad de cada cluster de ser el destino final dado cierta entrada. Dicha salida es utilizada después en conjunto con las coordenadas de los *clusters* en un producto punto, es decir, al final se obtendrá una coordenada la cual será el destino del viaje. Para obtener las probabilidades de la capa de salida es utilizada una función *softmax* la cual es representada a continuación:

$$p_i = \frac{exp(x_i)}{\sum_{j=1}^{C} exp(x_j)} \tag{5.1}$$

Donde:

- p_i es el valor normalizado de todas las neuronas en la capa de salida

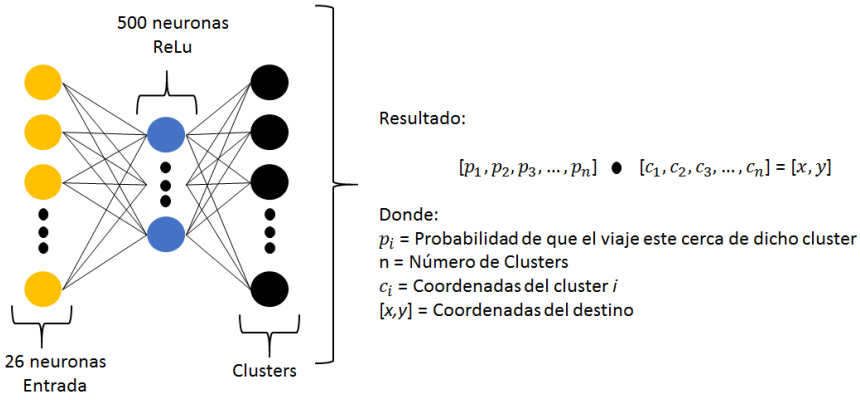


Figura 5.3: Modelo utilizado para predicción del destino.

Tabla 5.2: Resultados obtenidos por la red neuronal.

Error Mínimo	Error Máximo	Error Promedio	STD
0 Km	75 Km	1.80 Km	2.31

- x_i es el valor de la neurona que va a ser normalizado
- x_j es el valor de las demás neuronas contenidas en la capa de salida

En [26] se puede observar una descripción detallada de dicha función. Así mismo para la evaluación del modelo se utilizó la distancia en Km entre el destino real y el destino predicho. La estructura final del modelo puede ser apreciada en la Figura 5.3. Para el entrenamiento del modelo se utilizó el algoritmo de descenso de gradiente con un *momentum* de 0.9, usando bloques de 200 registros para entrenamiento.

5.2. Resultados

Cada modelo fue entrenado y evaluado 10 veces usando 1.5 millones de registros para poder obtener un promedio, es decir una tendencia en su comportamiento para la predicción de destinos. Para realizar las evaluaciones se dividió el conjunto de datos en entrenamiento y testeo eligiendo un 70 % y 30 % respectivamente. Como se comentó anteriormente para la fase de entrenamiento se utilizaron porciones de 200 registros en cada iteración de la red neuronal, así mismo la forma de evaluación del modelo es la distancia en kilómetros entre la distancia real y la distancia predicha. Los resultados de la red neuronal pueden ser apreciados en la Tabla 5.2

5.3. Análisis de resultados y estrategias de mejora

Para obtener una idea sobre posibles fenómenos que pudieran estar impactando en el funcionamiento de la red neuronal, se realizaron 4 tipos de análisis. Dichos análisis fueron realizados con el fin de investigar qué tipo de circunstancias causan el incremento del error en la predicción pero también para idear estrategias que puedan mejorar la precisión del modelo.

5.3.1. Inicios y destinos difíciles

Como primer hipótesis se tenía que el error que se presentaba en el modelo se debía a que la dificultad de predicción dependía solo de ciertas zonas, es decir, que en la ciudad pudieran existir sectores que por su naturaleza en la vida real el algoritmo no pudiera identificar los patrones dentro de la sección y por ende su predicción se viera disminuida.

El primer análisis consintió en un estudio espacial sobre los errores de cada uno de los viajes y la relación que pudieran tener con su posición de inicio o posición de destino. Para ello se graficaron todos los destinos e inicios con colores diferentes dependiendo del error que cada uno obtuvo en el modelo, dichos errores fueron catalogados en 5 categorías:

- Viajes con menos de 5 Km de error
- Viajes con 5 a 10 Km de error
- Viajes con 10 a 20 Km de error
- Viajes con 20 a 30 Km de error
- Viajes con más de 30 Km de error

Al realizar el análisis se creía que como resultado se iba a encontrar ciertas zonas de color determinado, es decir que probablemente las zonas más difíciles tendrían aglomerados a todos los viajes con más de 30 Km de error. Lamentablemente como se muestra en las Figuras 5.4 y 5.5, el error parece no depender solamente de donde inicia o donde termina el viaje ya que los errores están muy dispersos. Debido a esto, se prosiguió con un análisis de patrones entre zonas para encontrar las situaciones que causan el error en el modelo, dicho análisis es abordado en la sección siguiente.

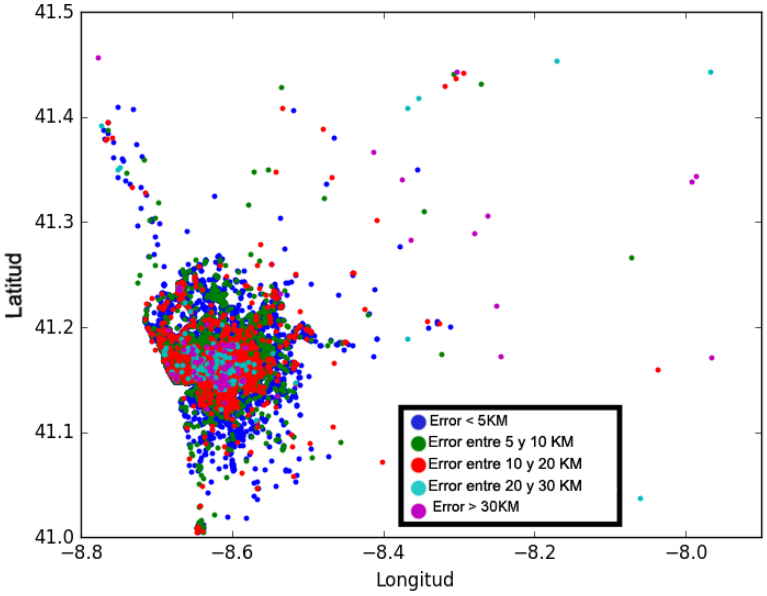


Figura 5.4: Relación entre zonas de inicio y error en los viajes

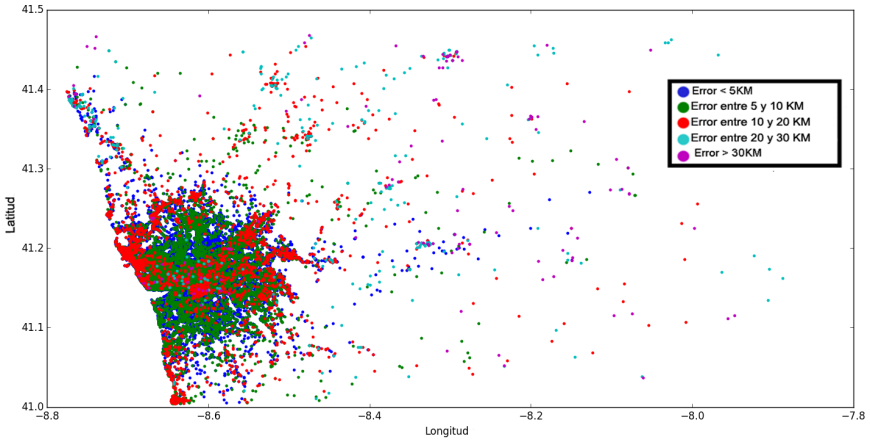


Figura 5.5: Relación entre zonas de destino y error en los viajes.

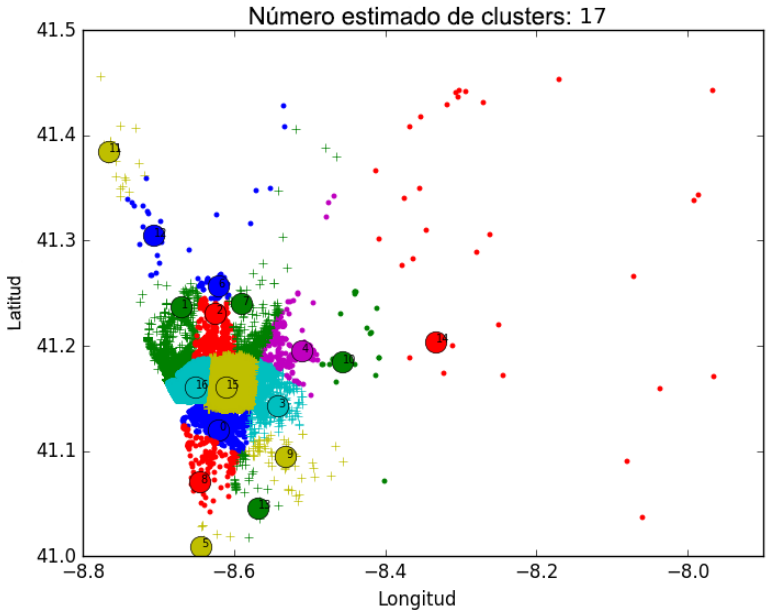


Figura 5.6: Ciudad de Porto, Portugal dividida en 17 sectores.

5.3.2. Análisis de sectores difíciles

El tercer análisis se enfocó en encontrar los patrones entre zonas que resultan difíciles de predecir y por ende causan un error grande. Para ello se dividió la ciudad de Porto en 17 sectores (véase Figura 5.6), dicho número de secciones fue seleccionado después de haber realizado diferentes aglomeraciones de destinos, dicho experimento dio como resultado el mínimo número de *clusters* necesarios para identificar la causa de aumento de error y facilitar así la identificación de patrones dentro de los viajes.

Después de realizar la división se generaron pares de secciones (inicio-fin del viaje), así mismo el error promedio de todos los viajes contenidos en cada par fue calculado.

Para representar el porcentaje de contribución de error de cada sección al error total (calculado por la red neuronal) se usó el término de influencia. Ese parámetro se utilizó para conocer cual par de secciones es más susceptible a mejoras y así, reducir el error total del modelo. La influencia de cada sección consiste en la multiplicación de error promedio por sección por el número de viajes contenidos en la misma, para obtener el porcentaje de influencia se divide cada valor de influencia por la suma total de dicha columna:

$$Influencia = ErrorPromedio * \#Viajes$$

$$\%deinfluencia = \frac{influencia}{TotalInfluencia}$$

Los viajes con mayor influencia en el error total pueden ser apreciados en la Tabla 5.3.

Tabla 5.3: Primeros 5 pares inicio-fin con el error más grande.

Inicio	Fin	Error Promedio	Influencia
15	15	1.030 Km	28 %
15	1	5.28 Km	15 %
15	16	2.19 Km	11 %
16	15	1.82 Km	9 %
15	3	2.46 Km	4 %

Como se puede apreciar, los viajes que inician en la sección 15 y terminan en la sección 1 son el par con mayor potencial de mejora debido a la alta tasa de error que manejan y el porcentaje de influencia en el error total. Analizando más a fondo, se encontró que para ir del sector 15 al sector 1 es necesario tomar una vía rápida (*highway*) para poder llegar a su destino, resultando así en muchas trayectorias similares.

5.3.3. Uso de patrones individuales para mejorar la predicción de destinos

Diversos trabajos [22, 2, 21] hacen uso de estrategias híbridas usando diferentes tipos de modelos para el reconocimiento de patrones mostrando una mejor precisión que usando los modelos separados. Para este análisis se creó un nuevo modelo el cual recibe solamente los viajes de un solo individuo como vector de entrada usando el atributo *origin_call* para diferenciarlos. El objetivo de este análisis es conocer si el uso de este tipo de modelos puede ayudar a mejorar la precisión del modelo.

Para el experimento se seleccionaron 9 usuarios haciendo uso del algoritmo K-means (véase Figura 5.7). Por cada usuario seleccionado se creó una red neuronal las cuales difieren solamente en su capa de salida dependiendo del número de destinos probables que tuviera cada persona. Cada red neuronal fue evaluada 10 veces. Para poner en perspectiva el análisis, la Tabla 5.4 presenta la diferencia de error promedio entre el modelo general y el modelo individual.

Como se puede apreciar, el uso de patrones individuales puede ser de ayuda para agregar valiosa información al modelo y como resultado mejorar su precisión, los resultados de este experimento son prometedores y el uso de un modelo híbrido puede ser una buena estrategia para reducir el error del modelo. Un ejemplo para la integración de los modelos sería la utilización de las salidas del modelo individual y el modelo general como entradas a otro algoritmo, en [33] se muestra la utilización de una red neuronal como modelo complementario, resultando así en una reducción del error del modelo utilizado como entrada. Así mismo se puede utilizar la misma metodología con los algoritmos vistos en el Capítulo 4, utilizando el algoritmo SVM

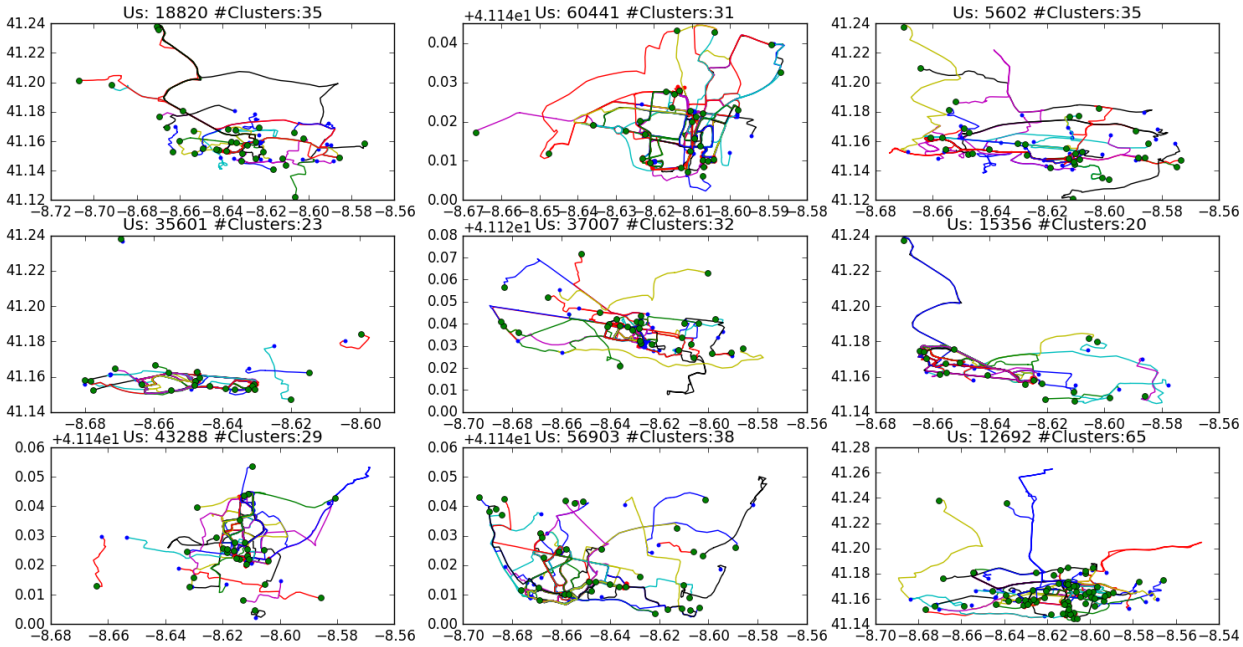


Figura 5.7: Usuarios elegidos para el análisis. En la imagen se puede apreciar los destinos frecuentes por cada usuario, así como todos los viajes contenidos de ese usuario en el conjunto de datos de entrenamiento.

Tabla 5.4: Comparación de errores entre modelo general y modelo individual

Usuario	Modelo General	Modelo Individual
12692	1.50 Km	1.77 Km
56903	2.42 Km	1.55 Km
43288	0.96 Km	0.92 Km
15356	1.07 Km	0.79 Km
37007	1.75 Km	0.65 Km
35601	2.36 Km	0.73 Km
5602	0.81 Km	1.64 Km
60441	0.85 Km	1.07 Km
18820	2.18 Km	1.66 Km

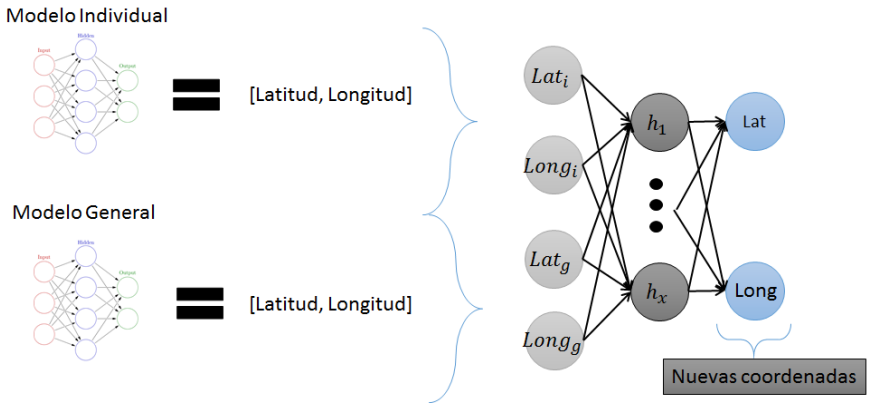


Figura 5.8: Ejemplo sobre uso de Red Neuronal para la integración de los modelos individuales y modelo general. La red neuronal se compondría de 4 neuronas de entrada que serían las coordenadas resultantes de los modelos individual y general, con dos neuronas de salida que darían como resultado las coordenadas del destino del viaje.

o Métodos de ensamble para realizar la integración de los dos modelos. En [22] se muestra como es utilizado un modelo de regresión lineal para la integración de modelos HMM para la predicción de destinos. Así mismo una triangulación de coordenadas podría ser empleada como metodología de integración, utilizando las dos coordenadas generadas por los dos modelos y la coordenada del *cluster* más cercano. En la Figura 5.8 podemos apreciar un ejemplo de cómo estos modelos podrían ser utilizados para obtener una salida más precisa utilizando una red neuronal.

El funcionamiento de dichas estrategias necesitan ser implementadas y evaluadas para realmente conocer el porcentaje de aumento en la precisión de los modelos; a pesar de que en este documento no se implementaron dichas estrategias se cree que el uso de sistemas híbridos puede mejorar la precisión en modelos, es por ello que dichas actividades serán realizadas en trabajo futuro.

6

CONCLUSIONES

El minado de trayectorias es un problema con muchos retos pero a la vez oportunidades. A pesar de que en esta época se tenga una gran cantidad de datos de esta índole, la mayoría de ellos contiene bastantes errores que dificultan en gran medida el correcto procesamiento y extracción de conocimiento desde ellos, por ejemplo, coordenadas falsas dentro de la trayectoria, errores en la obtención de coordenadas causadas por el GPS, inconsistencias en los datos debido a errores humanos, entre otros.

En este trabajo se presentaron diversas estrategias para la predicción de destinos, la primera de ellas es una estrategia de integración de modelos como son redes neuronales, SVM y métodos de ensamble para la predicción del área más probable. Así mismo, se implementó un modelo de una red neuronal asistido por un modelo de *clustering* para la obtención de las coordenadas más probables y se planteó una estrategia de integración de patrones generales y patrones individuales para el aumento en la predicción del modelo. Los resultados obtenidos sugieren que este tipo de modelos pueden ser utilizados para resolver el problema de la predicción de destinos, pero que diversas mejoras son necesarias para una resolución completa.

Uno de los problemas encontrados en el procesamiento de coordenadas es la similitud de trayectorias causado por autopistas. Dichas vías son utilizadas para llegar a diversos lugares dentro de una ciudad lo cual causa que muchas trayectorias se parezcan a pesar de que no se dirijan exactamente al mismo lugar, dicha problemática hace evidente la necesidad de una solución para este tipo de detalles. Como hipótesis, una mejor representación de la trayectoria (ej. uso de vectores, elección de diferentes coordenadas) podría ayudar al modelo a reconocer mejor los patrones dentro de las trayectorias, así mismo una mejor elección en el acomodo de los *clusters* podría ayudar al modelo a ser más preciso en la predicción de las coordenadas de destino.

Como trabajo futuro se pretende mejorar la representación de las trayectorias y el acomodo de los *clusters*, así mismo se desea implementar la integración de modelos generales con modelos individuales para

corroborar la disminución de error por medio de dicha estrategia.

Referencias

[1] R. Bajaj, S. L. Ranaweera, and D. P. Agrawal, "Gps: location-tracking technology," *Computer*, vol. 35, pp. 92–94, Apr 2002.

[2] J. Alvarez-lozano and J. A. Garc, "Crowd Location Forecasting at Points of Interest,"

[3] L. Song, D. Kotz, R. Jain, and X. He, "Evaluating next-cell predictors with extensive wi-fi mobility data," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 1633–1649, Dec 2006.

[4] T. Sohn, A. Varshavsky, A. LaMarca, M. Y. Chen, T. Choudhury, I. Smith, S. Consolvo, J. Hightower, W. G. Griswold, and E. de Lara, "Mobility detection using everyday gsm traces," in *Proceedings of the 8th International Conference on Ubiquitous Computing*, UbiComp'06, (Berlin, Heidelberg), pp. 212–224, Springer-Verlag, 2006.

[5] L. Liao, D. Fox, and H. Kautz, "Learning and inferring transportation routines," in *Proceedings of the 19th National Conference on Artifical Intelligence*, AAAI'04, pp. 348–353, AAAI Press, 2004.

[6] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on gps data," in *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, (New York, NY, USA), pp. 312–321, ACM, 2008.

[7] Y. Zheng, L. Liu, L. Wang, and X. Xie, "Learning transportation mode from raw gps data for geographic applications on the web," in *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, (New York, NY, USA), pp. 247–256, ACM, 2008.

[8] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, (New York, NY, USA), pp. 791–800, ACM, 2009.

[9] S. Hasan, X. Zhan, and S. V. Ukkusuri, "Understanding urban human activity and mobility patterns using large-scale location-based data from online social media," in *Proceedings of the 2Nd ACM*

SIGKDD International Workshop on Urban Computing, UrbComp '13, (New York, NY, USA), pp. 6:1–6:8, ACM, 2013.

[10] D. J. Patterson, L. Liao, D. Fox, and H. Kautz, "Inferring high-level behavior from low-level sensors," pp. 73–89, 2003.

[11] X. X. Yu Zheng, "Learning travel recommendations from user-generated gps traces," *ACM Transaction on Intelligent Systems and Technology*, January 2011.

[12] Y. Wang, Y. Zheng, and Y. Xue, "Travel time estimation of a path using sparse trajectories," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, (New York, NY, USA), pp. 25–34, ACM, 2014.

[13] J. Shang, Y. Zheng, W. Tong, E. Chang, and Y. Yu, "Inferring gas consumption and pollution emission of vehicles throughout a city," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, (New York, NY, USA), pp. 1027–1036, ACM, 2014.

[14] Z. Chen, H. T. Shen, X. Zhou, Y. Zheng, and X. Xie, "Searching trajectories by locations: An efficiency study," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, (New York, NY, USA), pp. 255–266, ACM, 2010.

[15] Z. Chen, H. T. Shen, and X. Zhou, "Discovering popular routes from trajectories," in *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*, ICDE '11, (Washington, DC, USA), pp. 900–911, IEEE Computer Society, 2011.

[16] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "T-drive: Enhancing driving directions with taxi drivers' intelligence," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, pp. 220–232, Jan 2013.

[17] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with gaussian mixture models," in *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pp. 141–146, June 2012.

[18] R. Fraile and S. Maybank, "Vehicle trajectory approximation and classification," in *British Machine Vision Conference*, 1998.

[19] J. Krumm, "A markov model for driver turn prediction," in *SAE Technical Paper*, SAE International, 04 2008.

[20] J. Krumm and E. Horvitz, "Predestination: Inferring destinations from partial trajectories," in *Proceedings of the 8th International Conference on Ubiquitous Computing*, UbiComp'06, (Berlin, Heidelberg), pp. 243–260, Springer-Verlag, 2006.

[21] J. A. Alvarez-Garcia, J. A. Ortega, L. Gonzalez-Abril, and F. Velasco, "Trip destination prediction based on past GPS log using a Hidden Markov Model," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8166–8171, 2010.

[22] M. Chen, X. Yu, and Y. Liu, "Mining moving patterns for predicting next location," *Information Systems*, vol. 54, pp. 156 – 168, 2015.

[23] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti, "Wherenext: A location predictor on trajectory pattern mining," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, (New York, NY, USA), pp. 637–646, ACM, 2009.

[24] H. T. Lam, E. Diaz-Aviles, A. Pascale, Y. Gkoufas, and B. Chen, "(Blue) Taxi Destination and Trip Time Prediction from Partial Trajectories," 2015.

[25] A. de Brébisson, É. Simon, A. Auvolat, P. Vincent, and Y. Bengio, "Artificial Neural Networks Applied to Taxi Destination Prediction," *arXiv Preprint*, pp. 1–12, 2015.

[26] S. Marsland, *Machine Learning: An Algorithmic Perspective*. Chapman & Hall/CRC, 1st ed., 2009.

[27] B. Lantz, *Machine Learning with R*. Packt Publishing, 2013.

[28] D. Dueker, M. Taher, J. Wilson, and R. McConnell, "Evaluating children's location using a personal GPS logging instrument: limitations and lessons learned," *Journal of Exposure Science and Environmental Epidemiology*, vol. 24, no. 3, pp. 244–252, 2014.

[29] F. Ding, G. Song, K. Yin, J. Li, and A. Song, "A gps-enabled wireless sensor network for monitoring radioactive materials," *Sensors and Actuators A: Physical*, vol. 155, no. 1, pp. 210 – 215, 2009.

[30] J. Yu and P. Lu, "Learning traffic signal phase and timing information from low-sampling rate taxi {GPS} trajectories," *Knowledge-Based Systems*, pp.–, 2016.

[31] S. M. Tomkiewicz, M. R. Fuller, J. G. Kie, and K. K. Bates, "Global positioning system and associated technologies in animal behaviour and ecological research," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 365, no. 1550, pp. 2163–2176, 2010.

[32] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11)* (G. J. Gordon and D. B. Dunson, eds.), vol. 15, pp. 315–323, Journal of Machine Learning Research - Workshop and Conference Proceedings, 2011.

[33] A. J. ABEBE and R. K. PRICE, "Managing uncertainty in hydrological models using complementary models," *Hydrological Sciences Journal*, vol. 48, no. 5, pp. 679–692, 2003.

Curriculum vitae

Normando Ali Zubia Hernández nace en la ciudad de Camargo, Chihuahua, México, en 1992.

Estudió la carrera de Ingeniería de Software en la Universidad Autónoma de Chihuahua, titulándose de ella en el año 2014. Durante su carrera, realizó prácticas profesionales en la empresa 1AAuto, trabajando como desarrollador web.

En 2014 entra al programa de Maestría en Ingeniería en Computación de la Facultad de Ingeniería de la Universidad Autónoma de Chihuahua, bajo la supervisión del Dr. Luis Carlos Gonzales Gurrola. Teniendo en el transcurso de la misma una estancia de investigación de tres meses en *The University of Nottingham* bajo la supervisión del Dr. Andrew J. Parkes. Como resultado de la maestría se encuentra este trabajo de investigación, y adicional mente se realizó un artículo de investigación para el *Workshop on Soft Computing Applications 2016* titulado "*Analysis of Taxi Destination Prediction Using a Neural Network*".

Actualmente trabaja como docente en la Facultad de Ingeniería de la Universidad Autónoma de Chihuahua. Para realizar contacto, mandar un correo a la dirección nzubiahdz@gmail.com o bien azubiah@uach.mx