laSalle
UNIVERSITAT RAMON LLULL

Master en BIG DATA

DATA MINING

Análisis accidentes área metropolitana de Barcelona

La Salle, Universitat Ramon Llull

Autores:
Jaume Feliubadaló Rubio
Juan José Carbonell Moral
Miguel Ángel Tarrason Moron

Profesor:
Javier Ordenes Barbany

Fecha de entrega:
19/02/2019

Índice

Índice de figuras

1. Objetivo

Este proyecto se ubica en la ciudad de Barcelona, en dicha ciudad se dispone de una estadística sobre accidentes de tráfico la cual indica que se producen más de 9000 accidentes automovilísticos cada año. El propósito de este proyecto es de, con el uso de un modelo machine learning, detectar las zonas más peligrosas del área metropolitana de Barcelona con el claro fin de reducir el número de víctimas.

Se tratará de un modelo de machine learning del tipo supervisado, en el cual se dispone de la variable respuesta. Los modelos de los cuales se tiene conocimiento, son aquellos que se identifican como clustering. El análisis de clusters, también nombrado como segmentación de datos, tiene una variedad de objetivos. Dichos objetivos estan relacionados con agrupar o segmentar un set de datos en clusters, de modo que los que están dentro de cada agrupación están más estrechamente relacionados entre sí que los objetos asignados a diferentes agrupaciones. El Clustering también se usa para formar estadísticas descriptivas para determinar si los datos consisten o no en un conjunto de subgrupos distintos, cada grupo representa objetos con propiedades sustancialmente diferentes.

Destacar que para el modelado se dispone de un set de datos, el cual contiene información sobre la ubicación del accidente en formato de latitud y longitud, el gridID que es un identificador de cuadrícula (segmentación de la ciudad de Barcelona en cuadrículas), la fecha y el turno en formato; mañana, tarde o noche. La respuesta es de tipo binaria; 1 si se produjo accidente y 0 si no se produjo.

2. Pipeline

La línia de trabajo que se ha seguido para la elaboración de este análisis es la que se especificia a continuación:

- Búsqueda de información referente al proyecto
- Discusión Técnicas de Validación
- Primer estudio
- Conclusiones primer estudio
- Aumento riqueza dataset
- Segundo estudio
- Conclusiones segundo estudio
- Tercer estudio
- Conclusiones tercet estudio
- Cuarto estudio
- Conclusiones cuarto estudio
- Estudio quinto y sexto
- Conclusión final

2.1. Búsqueda de información referente al proyecto

Se realiza una búsqueda de información con referencia al proyecto mencionado, tal como noticias referentes a accidentes ocurridos en Barcelona y estadísticas varias. Esta primera parte del proyecto consiste en entrar en materia de accidentes y en especial, los que se dan en el área metropolitana de Barcelona. Se busca información tanto en portales webs de opendata, a nivel de comunidad autónoma, nacional y europeo cómo también se busca en periódicos locales y nacionales. Esta información nos da un enfoque de la problemática existente. Se dispone de antemano de información la cual marcará las decisiones futuras con respecto al análisis del dataset que disponemos.

2.2. Discusión Técnicas de Validación

En primera instancia se valoran varias técnicas aprendidas a lo largo del curso. Una vez realizamos el primer modelado con un algoritmo del tipo decision tree, establecemos la ténica de validación comocida como; validación cruzada. Esta técnica es muy utilizada en proyectos del entorno de inteligencia artificial. En nuestro caso, siendo el objetivo principal el de la predicción, la técnica cross-validation nos permite evaluar los resultados del análisis llevado a cabo.

Con la validación cruzada se garantiza la independencia de la partición realizada. Se realiza una partición del dataset del tipo, entrenamiento-test. La validación cruzada consiste en repetir y calcular la media aritmética obtenida de las medidas de evaulación sobre las diferentes particiones.

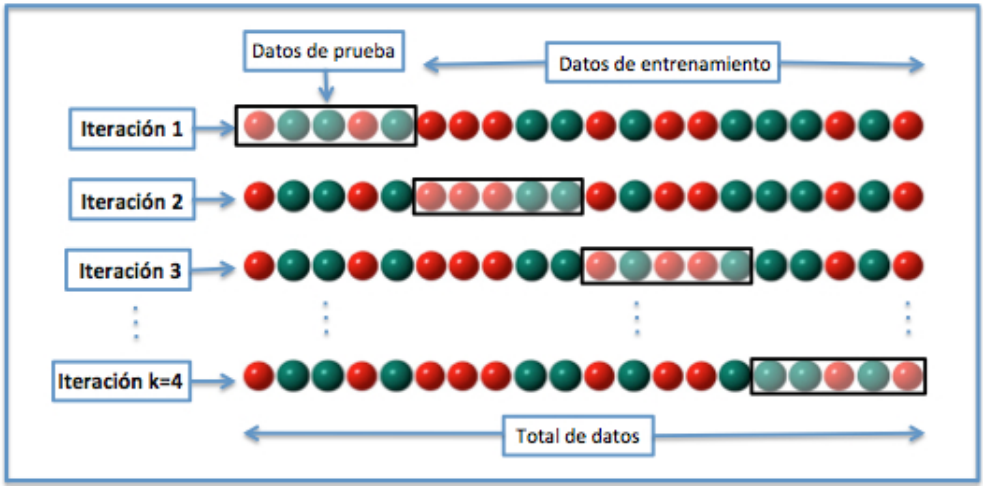

Figura 1: Cross-validation

2.3. Estudios y conclusiones de los mismos

Se han realizado un total de 6 estudios. En el primer estudio hacemos una aproximación en cuanto al procesado de la información contenida en el dataset y en el algoritmo. Destacar que antes de realizar cualquier procesado de información hemos hecho un primer modelado simple con el dataset original, el resultado de modelar un algoritmo del tipo decision-tree con la información original es una predicción alrededor de 0,79. En todos los estudios se realiza un enriqueciemiento del dataset, creando nuevas variables ya sean del tipo dummies ('shift') o features nuevas que nosotros consideramos que ayudarán a mejorar la predicción. De forma resumida, el tópico de cada estudio es el siguiente (para más detalle, se adjuntan los códigos ejecutados de cada estudio):

- Estudio 1: DTC con 0.7319 de predicción

- Estudio 2: Variable typeday. DTC con 0.68 de predicción.

- Estudio 3: DTC y regresión logística con 0.73 y 0.95 de predicción respectivamente.

- Estudio 4: Variable meteo. DTC con 0.54 de predicción (Overfitting).

- Estudio 5: Ensamble, Bagging, Boosting y Voting (DTC) con 0.73, 0.73, 0.75, 0.74 de predicción. Creación del mapa de barcelona.html.

- Estudio 6: Balanceo del dataset. Ensamble, Bagging, Boosting y Voting (DTC) con 0.77, 0.77, 0.74 y 0.75 de predicción.

3. Features

Este apartado da a conocer las nuevas variables introducidas en el dataset en los distintos estudios llevados a cabo en este proyecto. En este proyecto se han introducido un total de 11 variables. A continuación se detalla de forma sintetizada cada una de ellas. Destacar que todas ellas han sido introducidas a raíz de que se ha considerado oportuno su incorporación al dataset con el único de fin de enriquecer dicho dataset. Se enriquece el dataset con el objetivo de conseguir una mejora en la predicción de accidentes, es por eso que se escojen variables a introducir deducidas de la información recopilada anteriormente mencionada (ver apartado 1).

3.1. Dummies variable 'Shift'

En el dataset original existe la variable Shift, la cual representa los turnos divididos en 3 bloques: mañana (6h-13h), tarde (14h-20h) y noche (21h-5h). Se pretende aplicar dummies sobre esta variable. En la figura 2 se aprecia dicho procedimiento llevado a cabo. Posteriormente a la concatenación de los dummies, se elminina la variable Shift. De no eliminarse produciría overfitting al tener dos variables altamente correlacionadas.

```
dummy = pd.get_dummies(datos["Shift"])
dummy.head()
```

	Afternoon	Morning	Night
0	0	0	1
1	0	1	0
2	0	0	1
3	1	0	0
4	0	1	0

```
datos = pd.concat([datos, dummy], axis=1)
datos.head()
```

	GridID	date	Shift	Accident	Longitude.grid	Latitude.grid	Afternoon	Morning	Night
0	1	2010-10-08	Night	0	2.08	41.41	0	0	1
1	1	2011-02-16	Morning	0	2.08	41.41	0	1	0
2	1	2014-05-31	Night	0	2.08	41.41	0	0	1
3	1	2011-04-03	Afternoon	0	2.08	41.41	1	0	0
4	1	2013-02-20	Morning	0	2.08	41.41	0	1	0

Figura 2: Dummies variable Shift.

3.2. Week.day

Se decide crear la variable week.day, partiendo de la transformación en cuanto a formato de la variable ya existente 'date'. Una vez transformada dicha variable se realiza un bucle for con el cual se obtiene el número del día de la semana, destacar que se suma +1 ya que python empieza a contar a partir de 0, y 0 no tendría ningún sentido. Se tiene entonces que Lunes es el día 1, Martes es el día 2 y así siguiendo con la secuencia anteriormente mencionada.

```
from datetime import datetime, date, time, timedelta
import calendar
```

```
Week_Day = []
for fecha in datos['date']:
    dia_semana = datetime.weekday(fecha)+1
    Week_Day.append(dia_semana)

datos['Week.Day'] = Week_Day
del Week_Day
```

```
datos.head()
```

	GridID	date	Accident	Longitude.grid	Latitude.grid	Afternoon	Morning	Night	Week.Day
0	1	2010-10-08	0	2.08	41.41	0	0	1	5
1	1	2011-02-16	0	2.08	41.41	0	1	0	3
2	1	2014-05-31	0	2.08	41.41	0	0	1	6
3	1	2011-04-03	0	2.08	41.41	1	0	0	7
4	1	2013-02-20	0	2.08	41.41	0	1	0	3

Figura 3: Variable Week.day.

3.3. Festive

La variable Festive debido al razonamiento que se realiza en la primera parte del proyecto. Se considera que a partir de la información de la cual disponemos, los días festivos son una feature a incluir en nuestro dataset debido a que se sabe de ante mano que en dichos días, se produce un mayor número de desplazamientos dentro del área metropolitana de Barcelona.

Se aprecia en el trozo de código de la Figura 4, como se crea una lista con los días festivos de Barcelona. Se realiza un bucle for por el cual se adjudica a dichos días festivos un valor binario, si es día festivo 1 y 0 si no lo es. Se muestra también el total de días festivos de los cuales se disponen a lo largo del peridodo de tiempo en el cual se ubica el dataset.

```
festivos = ('1-1','6-1','19-4','22-4','1-05','10-6','24-6','15-8','11-9','24-9','12-10','1-11','6-12','25-12','26-12')
```

```
festive = []
for fecha in datos['date']:
    pasat = False
    for festivo in festivos:
        aux = festivo.split("-")
        aux2 = datetime.strftime(fecha, '%Y-%m-%d').split("-")
        if ( (aux2[1] == aux[1]) and (aux2[2] == aux[0]) and (pasat == False) ):
            festive.append(1)
            pasat = True
    if (pasat == False):
        festive.append(0)
```

```
datos['Festive'] = festive
del festive
del festivos
```

```
datos['Festive'].value_counts()
```

```
0    817881
1      6706
Name: Festive, dtype: int64
```

```
datos.head()
```

	GridID	date	Accident	Longitude.grid	Latitude.grid	Afternoon	Morning	Night	Week.Day	Festive
0	1	2010-10-08	0	2.08	41.41	0	0	1	5	0
1	1	2011-02-16	0	2.08	41.41	0	1	0	3	0
2	1	2014-05-31	0	2.08	41.41	0	0	1	6	0
3	1	2011-04-03	0	2.08	41.41	1	0	0	7	0
4	1	2013-02-20	0	2.08	41.41	0	1	0	3	0

Figura 4: Variable Festive.

3.4. Weekend

Esta variable se crea debido al razonamiento anteriormente expuesto para la variable Festive. En la Figura 5 se muestra el código y el head de los datos con dicha variable creada.

```
from datetime import datetime, date, time, timedelta
import calendar
```

```
Week_Day = []
for fecha in datos['date']:
    dia_semana = datetime.weekday(fecha)+1
    Week_Day.append(dia_semana)

datos['Week.Day'] = Week_Day
del Week_Day
```

```
datos.head()
```

	GridID	date	Accident	Longitude.grid	Latitude.grid	Afternoon	Morning	Night	Week.Day
0	1	2010-10-08	0	2.08	41.41	0	0	1	5
1	1	2011-02-16	0	2.08	41.41	0	1	0	3
2	1	2014-05-31	0	2.08	41.41	0	0	1	6
3	1	2011-04-03	0	2.08	41.41	1	0	0	7
4	1	2013-02-20	0	2.08	41.41	0	1	0	3

Figura 5: Variable Weekend.

4. Métodos y configuraciones utilizadas

Para el desarrollo del modelo predictivo cuyo objetivo es el de predecir el lugar y momento el cual se va a producir un accidente, se expone a continuación los distintos métodos y configuraciones más importantes que se han utilizado.

4.1. Label Encoder

Dentro de la parte de procesado de datos, sckit-learn ofrece este método para transformar variables categóricas en variables numéricas. Esto es debido a que las primeras no pueden ser procesadas por algoritmos.

Por ejemplo, en nuestra práctica, hacemos uso de dicha función para convertir la variable Season (Cold, Hot) en valores binarios. Destacar que dicha variable se ha utilizado en un estudio el cual no resulta ser el definitivo, es por ello que no se comenta la variable en el apartado de Features nuevas.

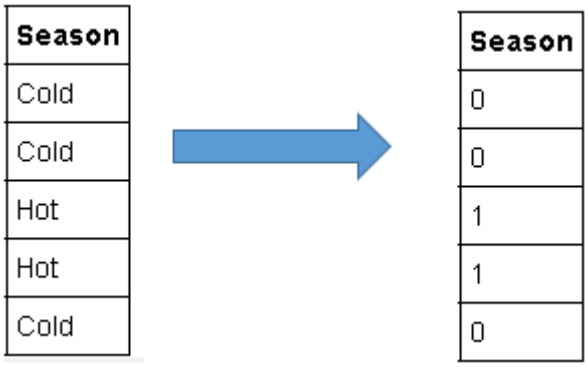

Figura 6: Label encoder. Fuente: Machine Learning Algorithms – Popular algorithms for data science and machine learning' by Guiseppe Bonaccorso.

4.2. DTC - Conditional Trees - Árboles de decisión

A la hora de aplicar este algoritmo no se debe tener presente la variable respuesta. Las variables predictoras pueden ser del tipo numéricas y categóricas. En este caso que nos ocupa se cumple que tenemos variables numéricas como predictoras y variable no categórica de estado en la respuesta. Representar árboles de mucha profundidad resulta de un grado de complejidad alto. Se realiza una selección de un 70 % sobre los datos de entrenamiento. Se realiza una primera inspección visual con un árbol de profundidad pequeño y apartir de ahí se toma la decisión de aumentar la profundidad del mismo.

Los árboles de decisión (DTC) son un método de aprendizaje supervisado con objetivo es crear un modelo que predice el valor de una variable objetivo mediante el aprendizaje de reglas de decisión simples deducidas de las características de los datos.

Por ejemplo, como se observa en la Figura 7, los árboles de decisión aprenden de los datos para aproximarse a una curva sinusoidal con un conjunto de reglas de decisión if-then-else. Cuanto más profundo es el árbol, más complejas son las reglas de decisión y más en forma el modelo.

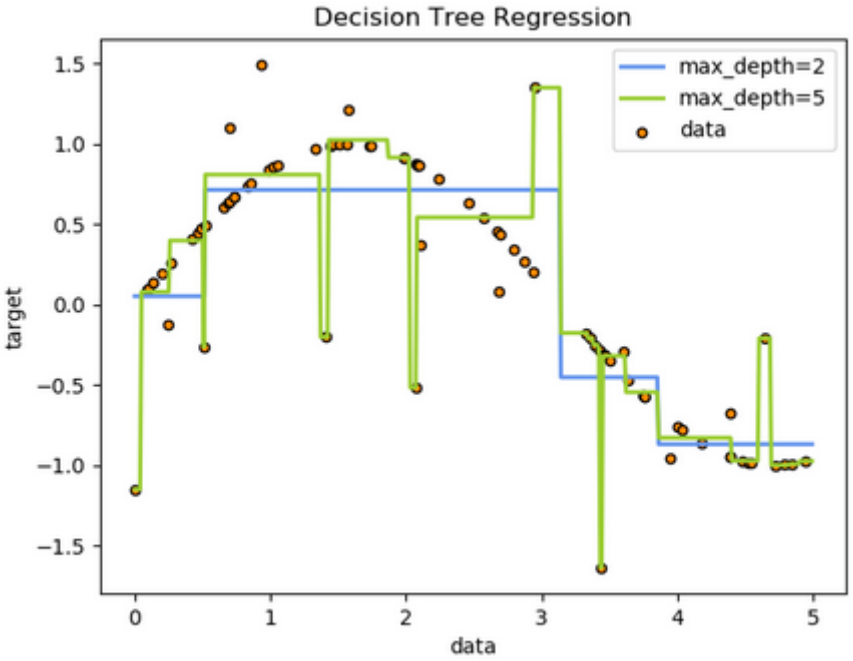

Figura 7: Ejemplo DTC.

4.3. Random Search

Se trata de un enfoque para el ajuste de parámetros por el algoritmo de una distribución aleatoria (es decir, uniforme) para un número fijo de iteraciones. Se construye y evalúa un modelo para cada combinación de parámetros elegidos.

En el caso de la práctica, haciendo uso de scikit-Learn, RandomizedSearchCV implementa una búsqueda aleatoria sobre parámetros, donde cada configuración se muestrea de una distribución sobre posibles valores de parámetros. Esto tiene dos ventajas principales sobre una búsqueda exhaustiva:

- Se puede elegir un alcance independientemente del número de parámetros y valores posibles.
- Agregar parámetros que no influyen en el rendimiento no disminuye la eficiencia.

4.4. Bagging, Boosting y Voting

A continuación se detallan los métodos más significativos basados en conjuntos de datos (Ensemble Learning) que también se han utilizado dentro del desarrollo de la práctica.

En estos casos, se tienen en cuenta las predicciones de un conjunto de hipótesis y, por lo general, el promedio de estas predicciones ofrece un resultado más preciso. Por el contrario, como es de esperar, conlleva un coste computacional mayor.

4.4.1. Bagging

En este caso el conjunto de datos se construye por completo. El entrenamiento se basa en una selección aleatoria de los sets de datos y las predicciones se toman en base al voto mayoriatorio del conjunto de sets.

Un ejemplo de Bootstrap es Random forests: Se trata de un conjunto de árboles de decision construidos sobre muestras aleatorias con un criterio de separación que busca encontrar el umbral que separa mejor los datos. En consecuencia, disponemos de varios árboles que producen una predicción totalmente distinta.

Importante destacar que pese a que el algoritmo interpreta los resultados escogiendo la clase más votada (democráticamente), scikit-learn lo implementa algo distinto ya que promedia los resultados.

4.4.2. Boosting

En este caso el conjunto se construye secuencialmente, centrándose en las muestras que han sido mal clasificados previamente.

Se considera unos ejemplos de Boosted el AdaBoost (Adaptative Bosting): La estructura básica es un árbol de decisión donde el conjunto de datos usado para el entrenamiento se adapta continuamente para forzar al modelo a centrarse en aquellas muestras que están mal clasificadas.

En cada iteración se aplica un factor de peso para aumentar la importancia de las muestras que se pronostican mal y disminuir la importancia de las otras. En otras palabras, el modelo itera comenzando como un alumno sin conocimiento hasta que el máximo de estimadores es alcanzado. Las predicciones siempre se toman por voto mayoritario.

4.4.3. Voting

La idea detrás de VotingClassifier es combinar métodos de aprendizaje automático conceptualmente diferentes y usar un voto mayoritario o el promedio de probabilidades pronosticadas (voto suave) para predecir las clases. Esto puede ser útil para un conjunto de modelos con un desempeño igual de bueno para equilibrar sus debilidades individuales.

Voting puede resultar una buena opción cuando una estrategia concreta no es suficiente para para alcanzar el umbral de acuracy deseado. Mientras se explotan los diferentes enfoques, es posible capturar muchoas microtendencias usando solo un pequeño grupo de

4.4.4. Apunte final

Y finalmente, para concluir este 4º apartado donde se detallan los distintos métodos y configuraciones utilizados, se incluye una visión global de la estrategia a seguir ante un nuevo dataset a analizar:

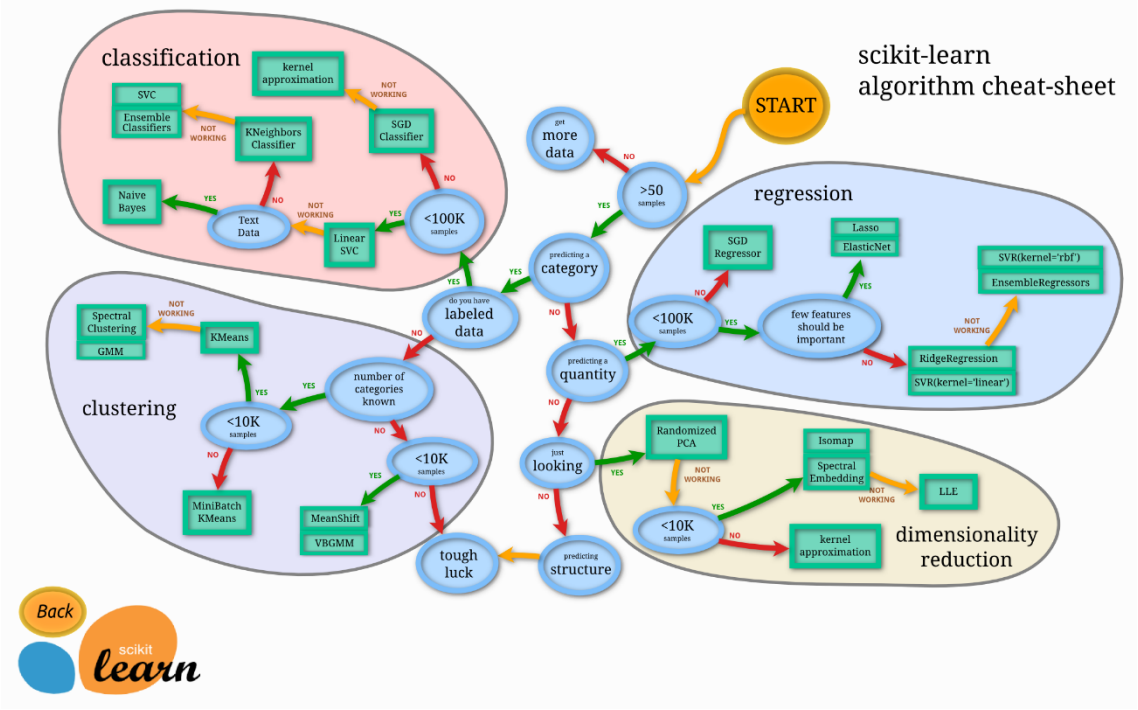

Figura 8: Estrategia. Fuente:scikit-learn.org/stable/tutorial/machine$_{l}earning_{m}ap/index.html$

5. Resultados y conclusiones

En este último apartado se exponen los resultados con sus debidas conclusiones sobre los métodos aplicados anteriormente mencionados sobre el dataset con el objetivo de predecir accidentes en Barcelona.

Se exponen únicamente los resultados del denominado caso base, el cual resulta ser el caso de estudio más favorable, en otras palabras, el estudio realizado el cual da una mejor predicción.

Destacar que este apartado se redacta con el fin de ser leído con el código adjuntado. Creemos que en este documento solo se debe presentar los resultados y conclusiones de forma sintetizada.

5.1. DTC

De todas las técnicas de machine learning aprendidas, se modela un algoritmo de tipo DTC (decision-tree-classifier). Se modela con una partición de los datos del tipo 70-30; 70 % de los datos son de entrenamiento y el 30 % son de test. Bajo los criterios de máxima profundidad no establecida y entropia, se consigue que sobre la muestra de entrenamiento el modelo tenga una capacidad predictiva del 88 % y del 73 % sobre la muestra test.

El párrafo anterior resume la primera aproximación que se realiza sobre el dataset. Es a partir de este primer modelado, que se empieza a discutir que técnicas pueden mejorar dicho resultado obtenido.

Destacar que obviamente solo se tendran en cuenta las técnicas utilizadas las cuales sean capaces de obtener una puntuación mejor a la obtenida. No obstante, se detalla algunas técnicas usadas con el fin de exponer en profundidad el estudio que se ha realizado.

5.2. Sobreajuste

Dichas técnicas bien podrían ser por ejemplo, el concepto de sobreajuste. En uno de los estudios realizados, se da un resultado notoriamente alto sobre la muestra de entrenamiento pero uno muy inferior en la muestra test. Este hecho nos conduce a realizar un estudio sobre el sobreajuste que se pueda obtener al realizar el modelado. En la figura 9 se observa el resultado del estudio sobre el Sobreajuste.

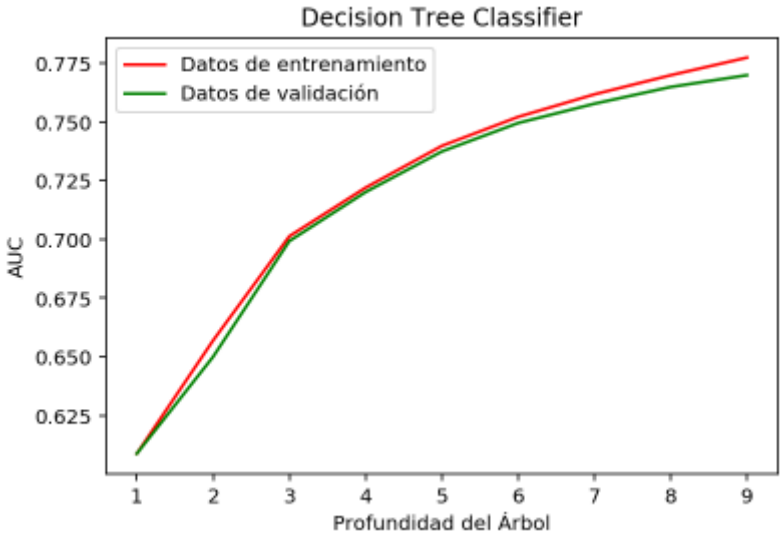

Figura 9: Sobreajuste

5.3. Label Encoder Shift

En este último subapartado, se detalla el mejor resultado obtenido en el estudio realizado sobre el dataset en cuestión. Se trata del estudio utilizando la técnica Label Encoder Shift que juntamente con el algoritmo DTC consigue como se aprecia en la Figura 10, una capacidad predictiva del 79 % sobre la muestra test.

```
print(roc_auc_score(y_train, y_train_pred[:, 1]), "Árbol de decisión - Datos de entrenamiento")
```

```
0.8213526334477538 Árbol de decisión - Datos de entrenamiento
```

```
y_test_pred = DTC.predict_proba(X_test)
```

```
print(roc_auc_score(y_test, y_test_pred[:, 1]), "Árbol de decisión - Datos de test")
```

```
0.7923023419575098 Árbol de decisión - Datos de test
```

Figura 10: Label Encoder Shift

6. Mapa de predicciones en el área metropolitana de Barcelona

En el proyecto, se incluye un gráfico interactivo en el cual se muestran los resultados. En la Figura 11 se puede observar dicho gráfico (formato mapa).

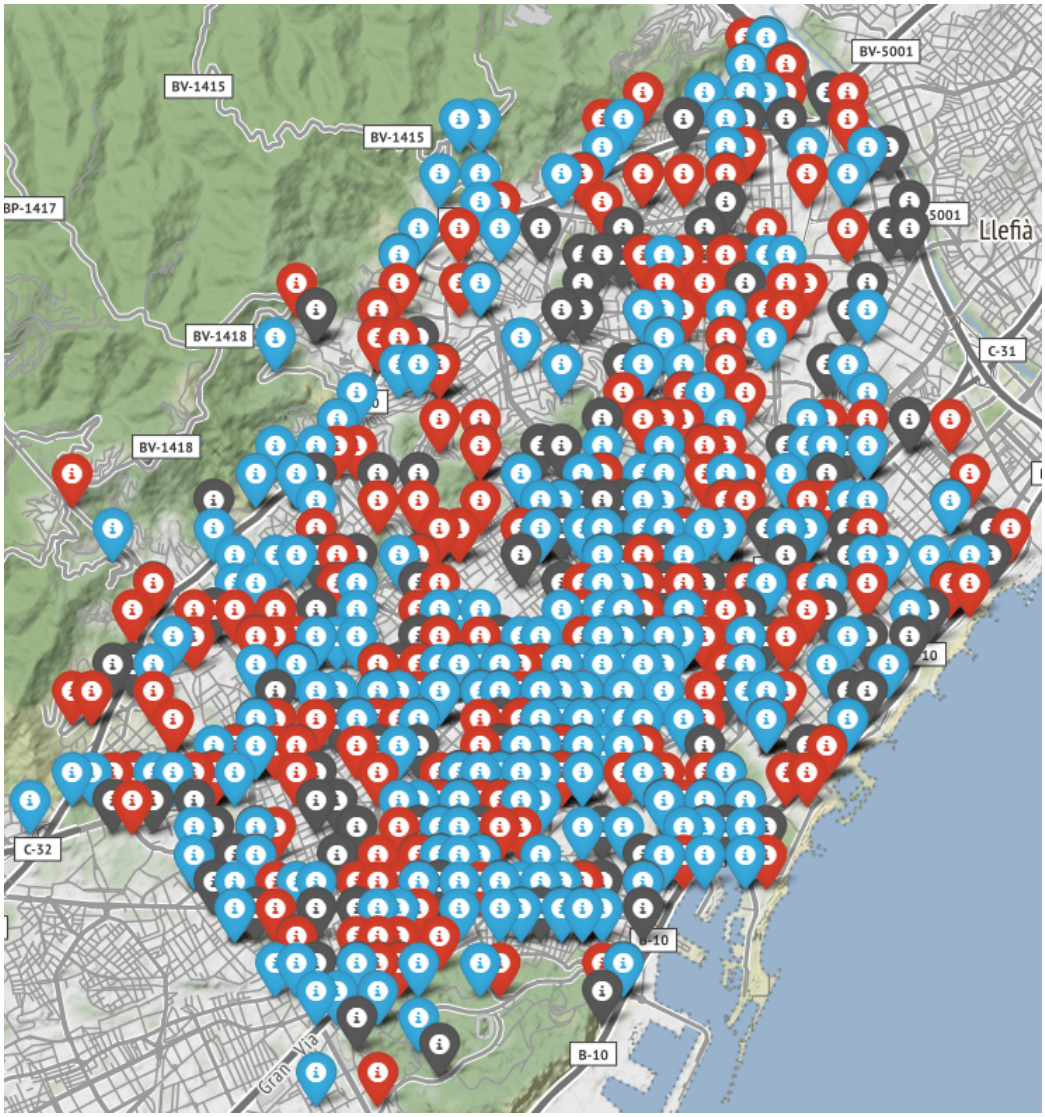

Figura 11: Mapa de predicciones en el área metropolitana de Barcelona