Driving Behavior

Ernesto Adrián Álvarez Salazar A016
Carlos Javier Leal Beltrán A01638192
Carlos Moisés Chávez Jiménez A01637322
Luis Armando Salazar López A01114901

Agosto 2022 - Septiembre 2022

1 Introduction

El comportamiento agresivo al volante es el principal factor de los accidentes de tráfico. Según informa la Fundación AAA para la Seguridad Vial, en 106.727 accidentes mortales -el 55,7

La conducción agresiva incluye el exceso de velocidad, las pausas repentinas y los giros bruscos a la izquierda o a la derecha. Todos estos eventos se reflejan en los datos del acelerómetro y el giroscopio. Por lo tanto, sabiendo que hoy en día casi todo el mundo posee un smartphone que tiene una gran variedad de sensores, hemos diseñado una aplicación de recopilación de datos en android basada en los sensores del acelerómetro y el giroscopio.

2 Tratamiento Inicial de los Datos

Para comenzar a trabajar con los datos, es necesario que pasen por un proceso de preparación que nos permita obtener la mejor parte de ellos. Este proceso se divide en dos partes: Limpieza y Transformación. A continuación explicaremos ambas partes de esta etapa:

2.1 Limpieza de los datos

La limpieza es una parte fundamental del tratamiento de la información. En esta etapa, se busca eliminar la mayor cantidad de imperfecciones que nos pudieramos llegar a encontrar. Cosas como valores faltantes, datos fuera de rango, dividir la información disponible en "entrenamiento" y "pruebas", eliminar columnas innecesarias para el análisis, etc. Estos son algunos de los comandos que utilizamos para esta etapa:

```
# Importando las librerias
    import pandas as pd
```

```
    from sklearn import svm
    import matplotlib.pyplot as plt
    import numpy as np
# Veremos tambien la cantidad de registros y variables que
     tenemos disponibles
    Nrows = t_train.shape[0]
    Ncols = t_train.shape[1]
# Haremos un resumen a grandes rasgos de la informacion
     disponible
    t_train.describe()
# Mostraremos el tipo de dato de cada variable de la BD
    t_train.dtypes
# Funcion para revisar que el dataset de training no tenga
     datos faltantes
    pd.isna(t_train).sum()
```

En el caso de nuestra base de datos, no hicimos ninguna limpieza (de momento). Encontramos muchos valores atípicos en las variables relacionadas con el movimiento de los conductores (giroscopio y acelerometro), pero no los eliminamos. Al ser demasiados, puede generar que se vaya mucha información que puede ser valiosa para nuestro modelo. Aunque no hemos definido aún si los dejaremos o no, decidimos esperar hasta avanzar un poco más en nuestra manipulación de los datos. Pero lo importante, no encontramos valores faltantes, ni columnas innecesarias.

2.2 Transformación de los datos

Para la transformación de datos es un caso diferente. Pertenece igualmente a las fases previas al trabajo de los datos, pero no involucra descartar información. La transformación involucra convertir los datos a diagramas o elementos gráficos que faciliten el entendimiento de la información. También para esta etápa consideramos el cambiar algúnas variables de tipo de dato. Para la tranformación utilizamos los siguientes comandos:

```
# Importando las librerias
    import pandas as pd
    from sklearn import svm
    import matplotlib.pyplot as plt
    import numpy as np
# Conteo de tipos de conductores
    plt.hist(t_train['Class'])
    plt.show()
# Grafica de barras con conteo y los tipos de conductores
    sns.countplot('Class',data = t_train)
    plt.show()
# Diagrama de correlacion entre las variables
    ax = sns.heatmap(t_train.corr(),
        annot=True).set(title='Pearson Correlations');
```

```
# Clasificaremos nuestras variables conforme al tipo de
    conductor
   t_trainSlow = t_train[t_train['Class']=='SLOW']
   t_trainNormal = t_train[t_train['Class']=='NORMAL']
   t_trainAggressive =
       t_train[t_train['Class']=='AGGRESSIVE']
# Relizaremos diagramas de cajas y bigotes para conocer los
    valores atipicos
   sns.boxplot(t_train['AccX'],y=t_train['Class'])
   sns.boxplot(t_train['AccY'],y=t_train['Class'])
   sns.boxplot(t_train['AccZ'],y=t_train['Class'])
   sns.boxplot(t_train['GyroX'],y=t_train['Class'])
   sns.boxplot(t_train['GyroY'],y=t_train['Class'])
   sns.boxplot(t_train['GyroZ'],y=t_train['Class'])
```

Con base en los datos que contamos para relizar este trabajo, análizamos y decidimos qué diagramas utilizar para acercarnos al desarrollo de nuestro modelo. Seleccionamos los diagramas de correlación y cajas y bigotes.
El diagrama de correlación nos permite conocer qué variables son más afínes a la variable que estamos intentado predecir. Lo cual nos ahorra mucho trabajo para seleccionar las variables que estarán involucradas en nuestro modelo.
El diagrama de cajas y bigotes nos permitió conocer si hay valores atípicos en nuestros datos con respecto a las variables independientes. De esta forma nosotros podemos descartar estos valores para hacer un modelo más ajustado y apegado a la realidad.
Por último, vimos el tipo de dato en el que está cada variable y decidimos que no necesitamos cambiarlo. Los valores flotantes son ideales para este caso porque no se pierde información con los puntos decimales. Por lo tanto, podremos tener más precisión a la hora de predecir el tipo de conductor.