Reporte final del reto:
Driving Behavior Challenge

David Alejandro Velázquez Valdéz[1 [A01632648]], Diana Guadalupe García Aguirre[2 [A01276380]], Edgar Daniel Acosta Rosales[3 [A01276214]], and José Herón Samperio León[4 [A01276217]]

Instituto Tecnológico y de Estudios Superiores de Monterrey Campus Guadalajara, Av. Gral Ramón Corona 2514, GDL, MX

Abstract. Uno de los principales factores que influyen en la ocurrencia de accidentes de tráfico es un comportamiento de conducción agresivo. Según lo informado por la *AAA Foundation for Traffic Safety*, en el 55.7 por ciento de los accidentes de tráfico que fueron registrados a lo largo de cuatro años -es decir, 106727 accidentes fatales-, los conductores involucrados presentaron una o más conductas de conducción agresivas. Con base en esto, se realizará un modelo de **clasificación supervisada** que, además de describir las métricas de los tipos de comportamiento que se reflejan en los datos recopilados de tres pruebas de manejo, ayudará a predecir el momento en el que los automovilistas, debido a sus técnicas y estilos de manejo, se encuentren en riesgo de sufrir un accidente de tráfico fatal.

Keywords: Métricas · Comportamiento · Accidentes.

1 Antecedentes e introducción

Ion Cojocaru, Stefan Popescu y Cristian Mihaescu, estudiantes de la Universidad de Craiova, en Rumania, llevaron a cabo un experimento de conducción, en el que dos personas: un conductor y su copiloto, se subieron a un automóvil y, a lo largo de tres periodos de, aproximadamente, diez minutos, trabajaron de manera conjunta: mientras uno conducía el vehículo en un estilo determinado, el otro medía con una aplicación de Android, dos veces por segundo, los valores registrados por el giroscopio y el acelerómetro del dispositivo en tres dimensiones, para asociar dichos registros al estilo de manejo correspondiente. La labor de este último se concentraba en mantener el teléfono quieto y estable durante la toma de valores.

Los datos recopilados se centran en registrar conductas tales como excesos de velocidad, frenados y giros bruscos. Los autores del experimento aseguran que el origen de la idea de crear una aplicación que realice tales registros, es el hecho de que la gran mayoría de las personas en la actualidad posee un teléfono

inteligente y, de establecerse modelos de predicción adecuados, podría desarrollarse una aplicación que ayude a reducir la cantidad de accidentes fatales.

En la actualidad, la problemática antes descrita se encuentra cargada en la plataforma más grande de científicos de datos: Kaggle, una subsidiaria de Alphabet Inc., la compañía dueña de Google, el buscador más popular del mundo. En dicha plataforma se encuentran cargados los datos antes mencionados y el reto asociado se halla abierto y en espera de propuestas de modelos, de cualquier usuario del mundo, que ayuden a predecir con la mayor precisión posible las conductas de conducción agresivas con datos nuevos.

2 Análisis exploratorio de los datos

Para efectos de la obtención, entrenamiento y prueba de cualquier modelo de aprendizaje automático, es necesario contar con datos. En el caso que analizaremos, se cuenta con dos conjuntos de datos, uno de entrenamiento llamado `train_motion_data.csv` con 3644 observaciones y uno de prueba llamado `test_motion_data.csv` con 3084 observaciones, ambos documentando ocho variables, siendo siete de ellas variables independientes y una de ellas dependiente, llamada `Class`, que define cuál es el tipo de comportamiento descrito por cada evento registrado. La llamamos dependiente porque asumiremos que las demás variables influyen en ella. *A posteriori*, usaremos el conjunto de prueba para verificar que nuestro modelo sea capaz de predecir, a partir de las variables independientes y con una exactitud aceptable, el tipo de conducta de manejo del usuario del dispositivo que obtuvo los datos correspondientes.

Para ser más específicos con los datos recopilados, se incluye una descripción de cada una de las variables:

- **AccX** | Aceleración en el eje de las X, medida en metros por segundo al cuadrado (m/s^2)
- **AccY** | Aceleración en el eje de las Y, medida en metros por segundo al cuadrado (m/s^2)
- **AccZ** | Aceleración en el eje de las Z, medida en metros por segundo al cuadrado (m/s^2)
- **GyroX** | Rotación en el eje de las X, medida en grados por segundo ($^\circ/s$)
- **GyroY** | Rotación en el eje de las Y, medida en grados por segundo ($^\circ/s$)
- **GyroZ** | Rotación en el eje de las Z, medida en grados por segundo ($^\circ/s$)

– **Timestamp** | Tiempo en segundos, en formato Epoch Unix Timestamp

– **Class** | Tipo de conducta de manejo (SLOW, NORMAL, AGGRESSIVE)

	AccX	AccY	AccZ	GyroX	GyroY	GyroZ	Class	Timestamp
0	0.000000	0.000000	0.000000	0.059407	-0.174707	0.101938	NORMAL	3581629
1	-1.624864	-1.082492	-0.204183	-0.028558	0.051313	0.135536	NORMAL	3581630
2	-0.594660	-0.122410	0.220502	-0.019395	-0.029322	0.087888	NORMAL	3581630
3	0.738478	-0.228456	0.667732	0.069791	-0.029932	0.054902	NORMAL	3581631
4	0.101741	0.777568	-0.066730	0.030696	-0.003665	0.054902	NORMAL	3581631
...	...	...	...	...	...	...	...	...
3639	0.915688	-2.017489	1.687505	0.450360	0.384845	-1.236468	SLOW	3583789
3640	-1.934203	0.914925	-0.096013	0.321468	0.649350	-0.477162	SLOW	3583790
3641	-0.222845	0.747304	-0.887430	0.361174	-0.406836	0.054291	SLOW	3583790
3642	-0.349423	0.067261	0.394368	-0.132405	0.020159	-0.004963	SLOW	3583791
3643	-0.402428	0.406218	-0.423009	-0.053603	-0.006720	0.001145	SLOW	3583791

Fig. 1. Visualización de los datos de train_motion_data.csv

Ambos conjuntos de datos **no cuentan** con registros **nulos ni incompletos**, por lo que se podrá realizar un análisis que no conlleve imputación de datos faltantes. Al momento de analizar la variable dependiente `Class`, el número de registros que describen cada tipo de comportamiento es:

– **SLOW**: 1331 - Proporción del 36.5% de los datos.

– **NORMAL**: 1200 - Proporción del 33% de los datos.

– **AGGRESSIVE**: 1113 - Proporción del 30.5% de los datos.

Se puede notar que los porcentajes de los comportamientos están desbalanceados, lo que nos obligará en el futuro a tomar acciones de balanceo de datos para asegurar que nuestro modelo sea fiable. Asimismo, el análisis realizado con base en las marcas temporales registradas, una vez que separamos los datos de prueba en función de la variable `Class`, comprueba que los registros corresponden a intervalos de, aproximadamente, diez minutos de prueba de manejo de cada tipo de conducta y que los sensores registraron datos dos veces por segundo durante cada prueba. En esta parte, surge el cuestionamiento de si es realmente necesario clasificar los datos en tres categorías de estilo de manejo, ya que, si volvemos al planteamiento, observaremos que lo que realmente nos

interesa es detectar las conductas agresivas, las otras dos pueden considerarse normales y, por ende, pueden entrar en la misma categoría.

A continuación, vamos a visualizar de una manera más gráfica la correlación entre las variables, por medio de una matriz de Pearson, con un énfasis especial en la relación existente entre las variables independientes y la dependiente, para así proceder con la ideación de alguna hipótesis sobre cuáles de las independientes serán más relevantes en la obtención de nuestro modelo de predicción. Adicionalmente, puede resultar sumamente valioso observar si existe correlación entre las variables independientes, para así poder tomar las decisiones correspondientes en el tratamiento de datos:

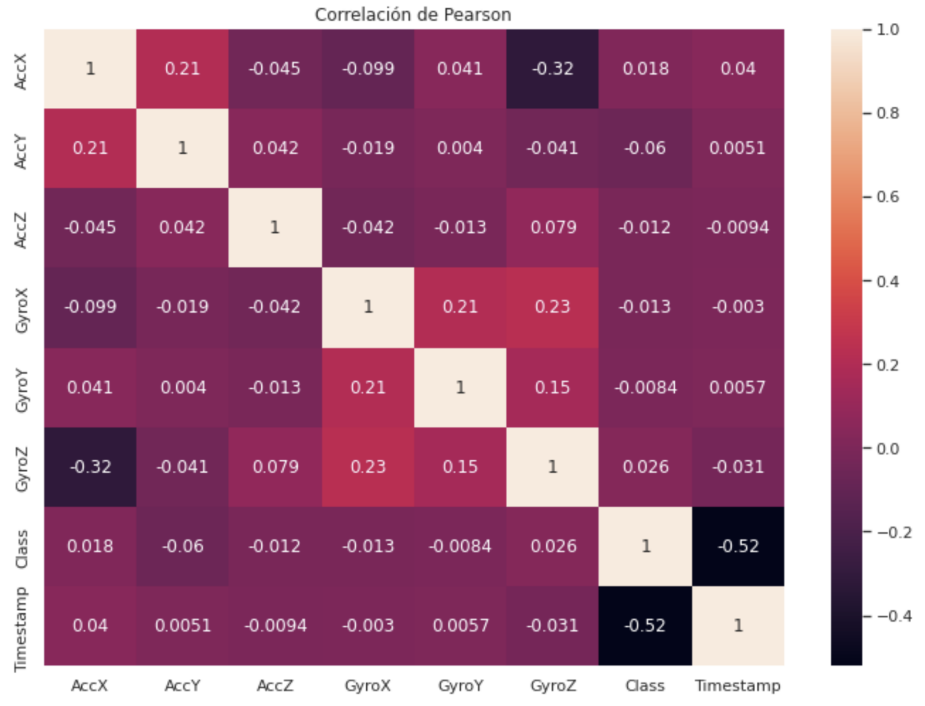

Fig. 2. Correlación de Pearson de los datos de train_motion_data.csv

Mediante el diagrama de correlación de Pearson, podemos visualizar que la variable dependiente `Class` se correlaciona en mayor medida con las variables independientes `AccX` -aceleración en el eje de las X medido en metros por segundo al cuadrado- y `GyroZ` -giroscopio en el eje de las Z medido en grados por segundo-. Descartaremos la relación con `Timestamp` por ahora, dado que claramente existirá una correlación por haber recolectado los datos a lo largo de tres momentos específicos de registro continuo de un comportamiento.

Recordemos que conocer la correlación entre las variables de este problema es uno de los pasos importantes para explorar los datos, sobre todo si queremos probar algún método estadístico. Sin embargo, no es el único dato relevante al realizar un análisis. Sabiendo que trabajamos con clases, resultaría muy interesante dividir el conjunto de datos en función de cada una de ellas y observar si hay algún hallazgo interesante si graficamos dicha clasificación, describiendo cada clase en función de cada una de las variables independientes:

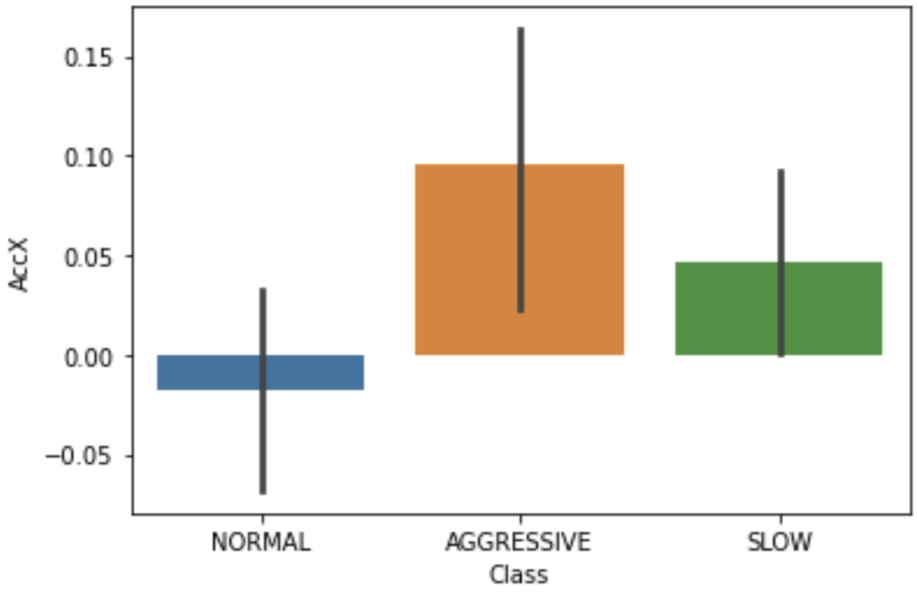

Fig. 3. Gráfico comparativo de aceleración en X para las tres clases de comportamiento.

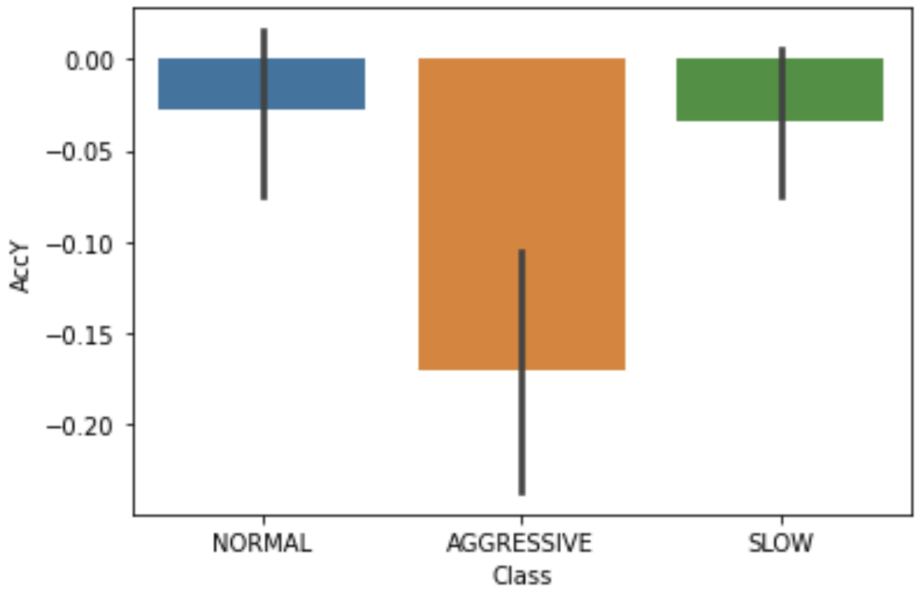

Fig. 4. Gráfico comparativo de aceleración en Y para las tres clases de comportamiento.

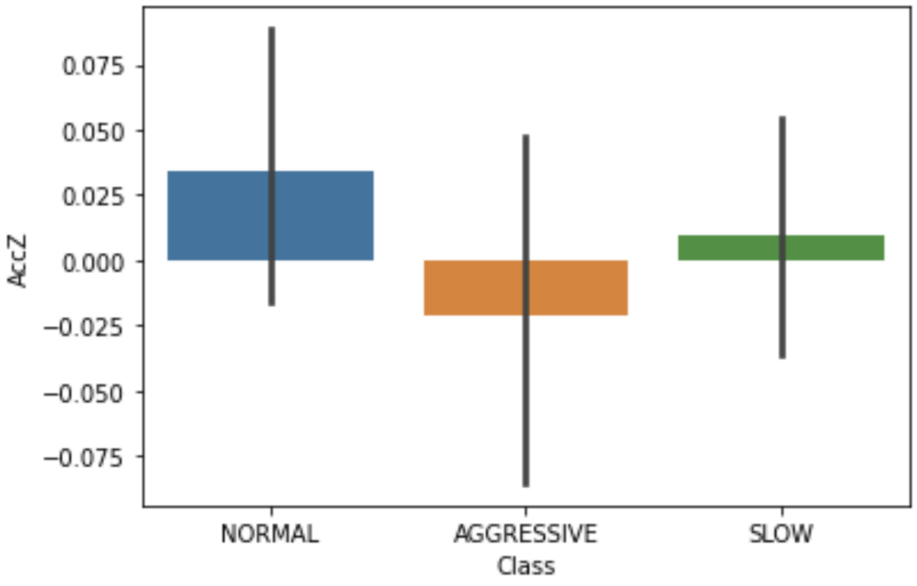

Fig. 5. Gráfico comparativo de aceleración en Z para las tres clases de comportamiento.

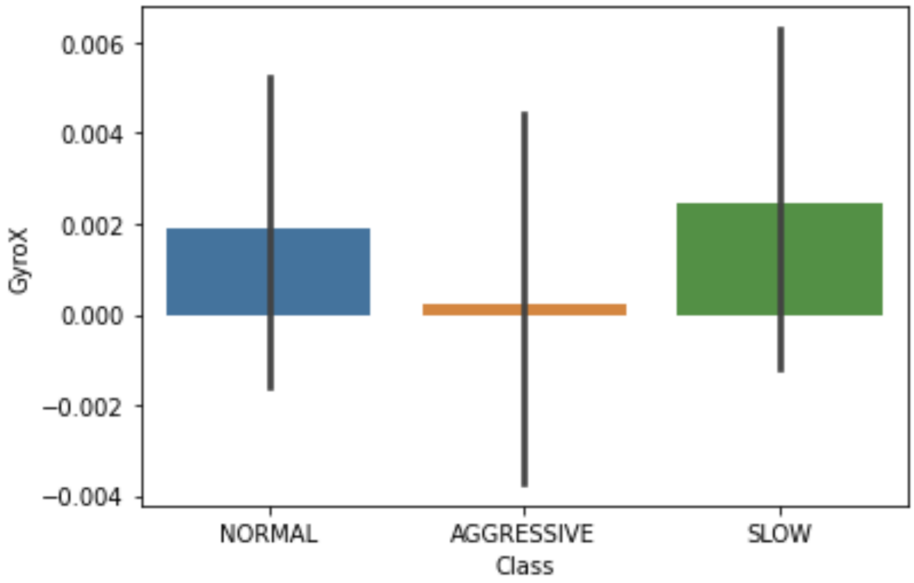

Fig. 6. Gráfico comparativo de giro en X para las tres clases de comportamiento.

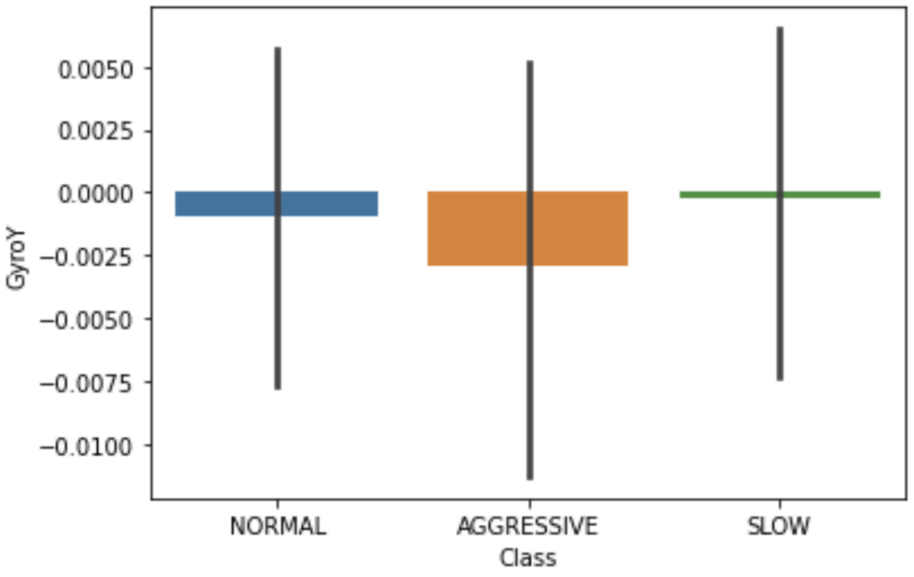

Fig. 7. Gráfico comparativo de giro en Y para las tres clases de comportamiento.

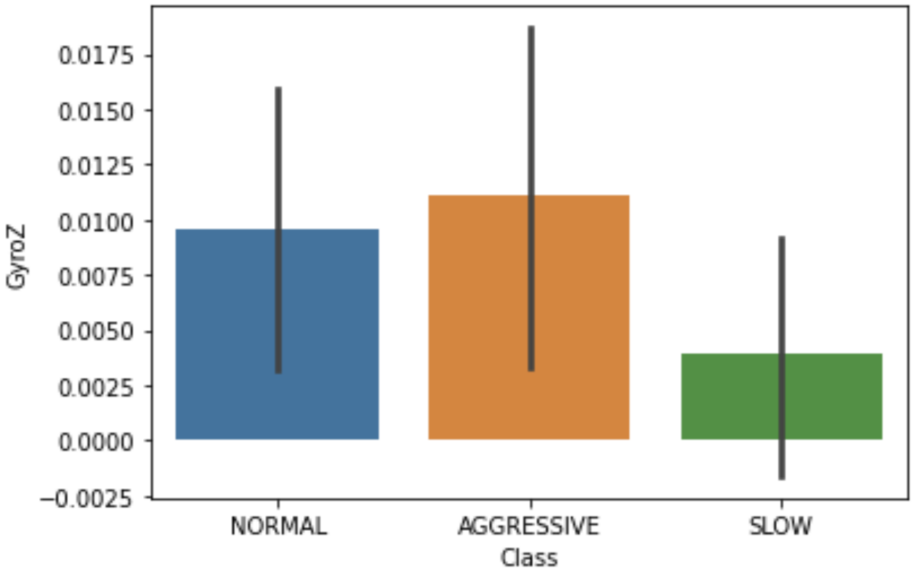

Fig. 8. Gráfico comparativo de giro en Z para las tres clases de comportamiento.

Al observar las gráficas anteriores, podemos darnos cuenta de que realmente podría ser una buena aproximación el hecho de unir las conductas de manejo lentas y normales en una sola categoría, ya que parece que los valores sí hacen una distinción interesante cuando se trata de conductas agresivas.

Una vez realizados estos análisis, se optó por realizar una normalización de los datos para optimizar su manejo en los distintos modelos a implementar y se eliminaron los registros atípicos de las variables independientes, en aras de obtener una mejor precisión en la predicción del comportamiento mediante un modelo óptimo.

3 Métodos de implementación

– K-Nearest Neighbors:
El algoritmo de K-vecinos más cercanos, por su traducción al español, es uno de los algoritmos más comúnmente usados dentro del aprendizaje de máquina, o *machine learning*, debido a que el modelo es muy simple de implementar y rápido en cuestión de cómputo, resolviendo problemas de clasificación. Este modelo es un método de clasificación supervisada no paramétrico, sirviendo para estimar la función de densidad $F(x/C_j)$ de las predictoras x por cada clase C_j, otorgando la probabilidad de que un elemento x pertenezca a a la clase C_j en un rango de k número de vecinos alrededor del elemento a evaluar, contando qué vecino es el que se repite más en el rango especificado por el usuario que realiza el modelo.

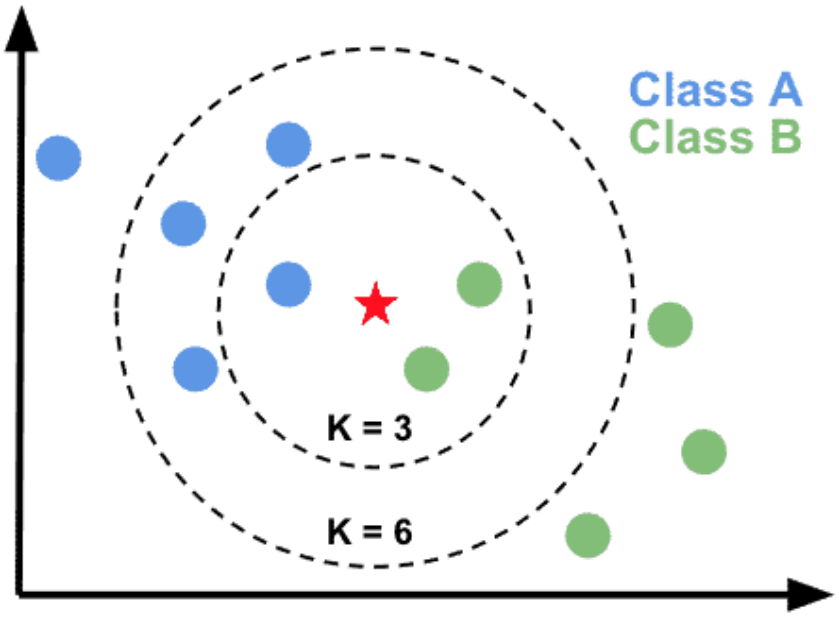

Fig. 9. Ejemplo de funcionamiento de K-Nearest Neighbors.

– Logistic Regression Algorithm:
El algoritmo de regresión logística, por su traducción al español, es un algoritmo que está relacionado con el área de estadística, usándose en problemas de clasificación binaria -con dos clases posibles-. Este algoritmo parte del uso de la función logística, -conocida también como función sigmoidal-, que se representa con una curva con forma de S que puede tomar cualquier número o valor real y mapearlo o convertirlo en un valor de rango entre 0 y 1 sin incluir dichos limites, siendo su función: $1/(1 + e^{-}value)$, donde e es la base de los logaritmos naturales y $value$ es el valor numérico que se busca transformar. La regresión logística es representada por una ecuación similar a la regresión linear, donde los valores de entrada (x) se combinan linealmente usando pesos o valores de coeficiente -representados como betas con subíndice n- para predecir un valor de salida (y), que está entre los rangos de 0 y 1. Un ejemplo de una ecuación de regresión logística es: $y = e^{(}b_0 + b_1 * x)/(1 + e^{(}b_0 + b_1 * x))$, donde $b0$ es el intercepto y b_1 o b_n es el coeficiente de la variable x_n. Gracias a este modelo se puede conocer la probabilidad de pertenencia a una clase en específico, representado como $P(X) = P(Y = 1/X)$, donde obtenemos dicha probabilidad con base en un valor de entrada x, y se es capaz de predecir, por ende, los valores de salida y.

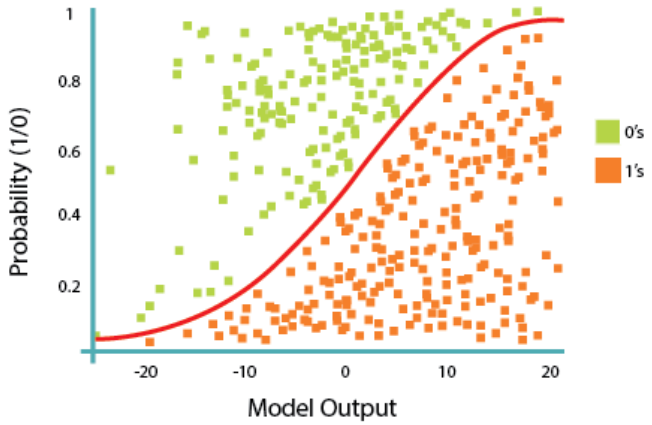

Fig. 10. Ejemplo de gráfica de regresión logística.

– Random Forest Algorithm:
El algoritmo de árboles aleatorios, por su traducción al español, es uno de los algoritmos más comúnmente usados, debido a que es muy versátil: puede usarse para resolver problemas tanto de regresión como de clasificación. Para explicar su funcionamiento, de manera breve, podemos remitirnos a su nombre: en efecto, el algoritmo construye varios árboles en su interior. Cada árbol tendrá el mismo objetivo: tomar una decisión respecto al problema planteado, ya sea elegir una clase o predecir un valor. Los árboles de decisión son estructuras simples, pero poderosas, que son capaces de elegir una salida tomando en cuenta todas las ramificaciones que las características de cada conjunto de datos otorga. Para ejemplificarlo mejor, tomemos como referencia un conjunto de datos que contiene tres variables independientes: *workToDo, outlook* y *friendsBusy*, con una variable de salida: *whatToDo*. Para determinar la salida de dicha variable, hemos de construir un árbol que incluya todas las posibles combinaciones de decisión de las tres variables independientes. Un ejemplo de lo anterior puede observarse en la figura 11:

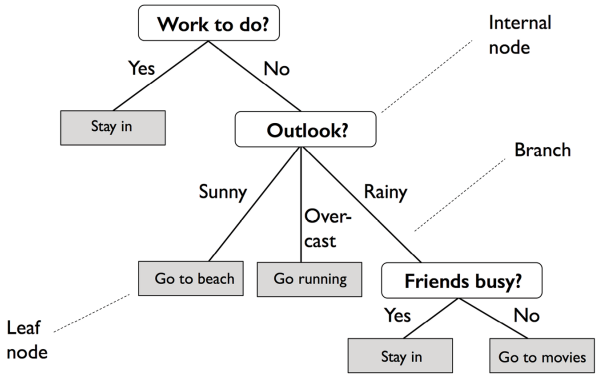

Fig. 11. Ejemplo de un árbol de decisión.

En el caso de nuestro conjunto de datos, hemos podido observar en las gráficas comparativas de clases que, en el caso de algunas de ellas, existe una diferencia marcada entre los valores de entrada de ciertas clases: por ejemplo, la aceleración en Z (AccZ), presenta valores negativos al producirse una conducta de manejo agresiva y la aceleración en X (AccX) hace lo mismo con conductas de manejo normales. De esta forma, asumimos, podemos construir un árbol de decisión que nos ayude a determinar la clase de la conducta observada con base en las variables independientes.

Adicionalmente, vale la pena destacar que uno de los mayores problemas del uso de árboles de decisión es aquello que conocemos como sobreentrenamiento, un fenómeno en el que el algoritmo aprende demasiado bien lo que sucede con el conjunto de datos de entrenamiento, pero se ajusta tanto que, *a posteriori* le resulta sumamente complejo trabajar con datos fuera de este conjunto de entrenamiento. La aleatoriedad del bosque yace en la elección de variables: se crean conjuntos de variables aleatorias, que, por ende, no se encuentran correlacionadas, lo que contribuye en la significativa disminución del problema tan común que representa el sobreentrenamiento del algoritmo. Es por esto último que hemos decidido usar un bosque aleatorio en lugar de un árbol de decisión unitario, que es más propenso a sufrir de este tipo de problemas.

– Multi-Layer Perceptron Classifier:
Un *perceptrón* es una neurona artificial basada en una función matemática de un modelo neuronal biológico $w0(t)$.

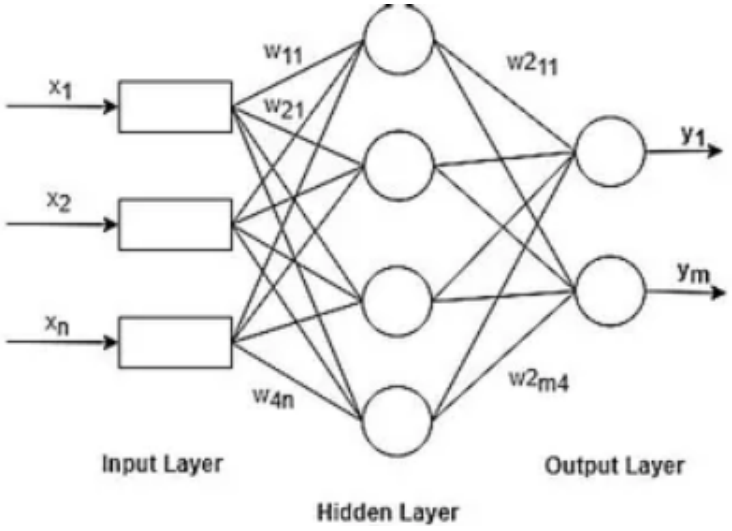

Fig. 12. Ejemplo de un modelo de clasificacion multicapa.

Es un algoritmo que, a diferencia de otros algoritmos de clasificación, se basa en una red neuronal artificial que puede ser aplicada en problemas no lineales con grandes cantidades de datos y no es necesario tener un gran banco de información para entrenarlo de manera apropiada, ya que se puede ajustar y obtener el mismo nivel de precisión que otros algoritmos.
La estructura de este algoritmo consta de:

- Datos de entrada
- *Hidden layer(s)* -pudiendo ser una o más-.
- Datos de salida

El perceptrón, matemáticamente, se describe de la siguiente forma: los datos de entrada se multiplican por los coeficientes de peso, donde el valor puede ser positivo o negativo, de ser positivo -y si el peso previamente calculado supera un umbral establecido-, la neurona se activa. La predicción se compara con un resultado conocido y de ser erróneo se transmite la información para ajustar los pesos.
Esto último no indica que no sea posible sobreentrenar el modelo y que como resultado las predicciones obtenidas con nuevos datos sean erróneas.

4 Resultados

– K-Nearest Neighbors:
La implementación del algoritmo K-Nearest Neighbors, con un valor de $k = 150$, obtuvo una precisión del 74% y una matriz de confusión de 100% verdaderos positivos y 99.75% verdaderos negativos. El reporte de clasificación obtenido se mostrará a continuación en la figura 13.

Reporte de clasificación:

	precision	recall	f1-score	support
0	0.74	1.00	0.85	2270
1	1.00	0.00	0.00	814
accuracy			0.74	3084
macro avg	0.87	0.50	0.43	3084
weighted avg	0.81	0.74	0.63	3084

Fig. 13. K-Nearest Neighbors - Reporte de clasificación.

– Logistic Regression Algorithm:
La implementación del algoritmo de regresión logística obtuvo una precisión del 74% y una matriz de confusión de 100% de verdaderos positivos y 100% de verdaderos negativos. El reporte de clasificación obtenido se mostrará a continuación en la figura 14.

Reporte de clasificación:

	precision	recall	f1-score	support
0	0.74	1.00	0.85	2270
1	0.00	0.00	0.00	814
accuracy			0.74	3084
macro avg	0.37	0.50	0.42	3084
weighted avg	0.54	0.74	0.62	3084

Fig. 14. Logistic Regression - Reporte de clasificación.

– Random Forest Algorithm:
La implementación del bosque aleatorio se dividió en dos fases: aquella en la que se consideraron las tres clases de salida proyectadas en el conjunto de datos original y aquella en la que sólo se consideraron dos clases, uniendo en una sola clase las conductas SLOW y NORMAL de manejo. Para la versión de tres fases, se obtuvo una precisión del 47.21%, mientras que la segunda tuvo una precisión del 74.35%. Curiosamente, al observar la matriz de confusión y el reporte de clasificación del algoritmo, en su segunda implementación, que es la que aquí documentaremos, nos percatamos de que el porcentaje de predicción para la clase SLOW-NORMAL es significativamente mayor que la de la clase AGGRESSIVE, por lo que suponemos que el desbalanceo de datos existente en el conjunto, entre ambas clases, podría estar provocando el problema. Al salir de los alcances del bloque TC3006C.101, no realizamos el balanceo de datos con el conjunto computado de prueba, pero, *a posteriori*, podría implementarse esta mejora, en aras de observar si contribuye con el mejoramiento de nuestro algoritmo, principalmente al detectar de forma precisa las conductas de conducción agresivas. Esto último es esencial, pues recordemos que es el objetivo primario de nuestro análisis y, si observamos la matriz de confusión con detenimiento y un poco de sentido común, podremos deducir que el porcentaje de dicha matriz que debemos enfocarnos en reducir es el de falsos negativos para la clase AGGRESSIVE. Según nuestro reporte, el algoritmo fue capaz de predecir únicamente el 25% de todos los registros de conductas agresivas, dejando fuera, lógicamente, el 75% de ellas. Recordemos que obtener un algoritmo que compute y realice predicciones con alta precisión es muy deseable, pero es aún más deseable que dicha precisión se concentre en los resultados más vitales del contexto del problema.

```
Reporte de clasificación:
              precision    recall  f1-score   support

           0       0.77      0.92      0.84      2270
           1       0.53      0.25      0.34       814

    accuracy                           0.74      3084
   macro avg       0.65      0.59      0.59      3084
weighted avg       0.71      0.74      0.71      3084


Accuracy score:
 0.743514915693904
```

Fig. 15. Random Forest - Reporte de clasificación y precisión.

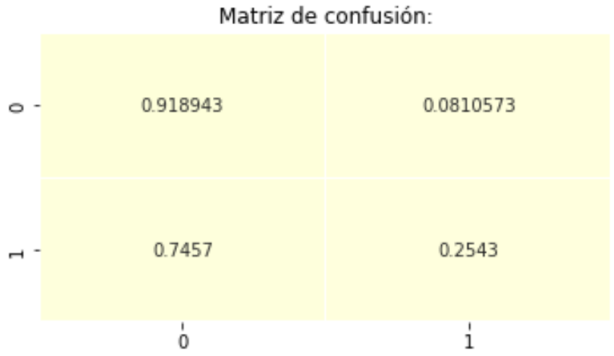

Fig. 16. Random Forest - Matriz de confusión.

– Multi-Layer Perceptron Classifier:
Cuando implementamos el modelo de clasificación multi capa primero establecemos las 4 entradas (AccX, AccY, GyroX, GyroY), 256 hidden layers y 3 resultados (SLOW, NORMAL, AGGRESSIVE). Los resultados de precisión oscilan entre 0.36% y 0.42%

Reporte de clasificación:

	precision	recall	f1-score	support
AGGRESSIVE	0.44	0.43	0.44	814
NORMAL	0.00	0.00	0.00	997
SLOW	0.47	0.85	0.61	1273
accuracy			0.47	3084
macro avg	0.31	0.43	0.35	3084
weighted avg	0.31	0.47	0.37	3084

Fig. 17. MLP Classifier - Reporte de clasificación y precisión.

Incluso cuando interactuamos con los parámetros de activación, cantidad de datos de entrenamiento y numero de épocas, el máximo obtenido no pudo pasar el 50%.

A pesar de este resultado significativamente mayor, el modelo no llegaba a clasificar las 4 entradas y proveer las 3 salidas.

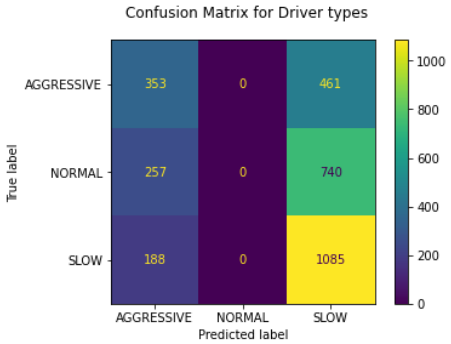

Fig. 18. MLP Classifier - Matriz de confusión.

5 Discusión

Al visualizar los resultados obtenidos por medio de los algoritmos implementados para predecir el tipo de comportamiento de manejo, podemos observar que todos ellos tienen medidas de precisión general similares. Elegiremos el algoritmo de MLP Classifier, debido a que fue capaz de predecir correctamente el 43% del total de las conductas de manejo agresivas del conjunto de prueba, el porcentaje más grande de todos los modelos computados. Sirve recordar que esta métrica es la más relevante del problema, por las razones anteriormente descritas.

6 Conclusión

Una de las características del reto planteado que vale la pena mencionar en este apartado, es el hecho de que la recolección de los datos podría no ser del todo objetiva. A reservas de indagar aún más en el experimento en el que se recolectaron los datos, todo parece indicar que las conductas de manejo fueron simuladas, por lo que es probable que no reflejen tan fielmente las métricas de conducción correspondientes y se encuentren un poco sesgadas. No obstante, creemos que los resultados obtenidos sí podrían resultar relevantes en el desarrollo de la aplicación que se plantea, aunque, de forma deseable, sería un buen ejercicio futuro el de reunir datos de conducción en escenarios reales, que colaboren con la investigación. Por otro lado, el desbalanceo de los datos antes mencionado y su repercusión en el modelo también es un factor importante en el análisis de nuestros resultados, pues ya se ha podido observar que el hecho de que se compute una precisión general del modelo aceptable, podría no significar necesariamente obtener tanta precisión para predecir ciertos aspectos que son, cabe decirlo, aún más relevantes que otros, en vista de que es esencial predecir la mayor cantidad de registros de conductas agresivas que sean posibles. Finalmente, podemos mencionar la increíble cantidad de cosas aprendidas durante estas cinco semanas, durante las cuales hemos ido descubriendo la importancia de la Ciencia de Datos y sus aplicaciones de una forma interesante y de valor. Este reto nos ha brindado la oportunidad de poner a prueba nuestros conocimientos y nuestra capacidad de inferencia, algo esencial en el quehacer ingenieril.