Predicciones del precio de dispositivos móviles utilizando un clasificador Random Forest

Emiliano Vásquez Olea – A01707035

1. Introducción

Los avances tecnológicos en las últimas décadas muestran la importancia e interés que existe a nivel mundial por encontrar soluciones mejores y más eficientes para todos los problemas que existen. Conforme estos problemas se vuelven más complejos y los métodos y herramientas "tradicionales" ya no demuestran ser opciones viables, es necesario optar por nuevas soluciones a partir de los recursos con los que contamos.

El aprendizaje de maquina junto con el análisis de datos es uno de los ejemplos de este nuevo enfoque para resolver problemas complejos. En el presente documento se describirá el uso y ajuste del método de aprendizaje de máquina *Random Forest* para predecir los precios de dispositivos móviles a partir de sus especificaciones.

El objetivo de este modelo es utilizar datos recolectados de diferentes dispositivos, para poder estimar el precio que debe ser atribuido a nuevos dispositivos que sean fabricados. El *dataset* utilizado se encuentra en la plataforma *Kaggle* dentro de la siguiente liga: https://www.kaggle.com/datasets/iabhishekofficial/mobile-price-classification.

2. Separación del conjunto de datos

El *dataset* está compuesto por dos archivos, *train.csv* y *test.csv*, con 2000 y 1000 registros respectivamente. Sin embargo, para este análisis únicamente se hace uso del archivo *train.csv* debido a que solo este contiene las etiquetas correctas de la clasificación, que son necesarias al utilizar un modelo de aprendizaje supervisado. Entonces, se utilizarán estos 2000 registros para realizar la separación entre datos de entrenamiento, prueba y validación. Los registros contienen los siguientes atributos o columnas:

- battery_power
- blue
- clock_speed
- dual_sim
- fc
- four_g
- int_memory
- m_dep
- mobile_wt
- n_cores
- pc
- px_height
- px_width
- ram
- sc_h
- sc_w
- talk_time
- three_g
- touch_screen
- wifi

- price_range

En la descripción (*README.md*) del repositorio de la entrega se puede consultar la definición de cada uno de estos atributos. Es importante también mencionar que no se tienen datos faltantes, por lo que no fue necesario, en este caso, llevar a cabo algún proceso para rellenar esta información.

La primer división del *dataset* consistió en separar los datos de entrenamiento, con 85%, y un 15% para datos de prueba del *dataset* original. Después, del conjunto de entrenamiento se tomó otro 15% para la validación (12.75% del conjunto original).

Los datos de entrenamiento son utilizados, como su nombre lo indica, para entrenar nuestro modelo de aprendizaje automático, es decir, a partir de estos datos el método de *Random Forest* actualizará sus parámetros y sub-modelos para poder realizar predicciones. El conjunto de validación, por otra parte, son los datos usados para la actualización de hiperparámetros de nuestro modelo y será utilizado en el apartado de Ajuste de Parámetros en este documento. Por último, el conjunto de prueba permite evaluar el desempeño de nuestro modelo final a partir de información que no ha visto anteriormente.

Al separar un conjunto de validación y otro de prueba nos aseguramos de que podemos realizar ajustes a nuestro modelo sin forzar un cambio en el desempeño con los datos de prueba. Esto quiere decir que no estamos "metiendo mano" para tener una precisión más alta con los datos de prueba y mostrar un resultado mejor.

Antes de ser utilizados, los atributos que conforman la x, o los datos de entrada, son escalados utilizando el *Min Max Scaler* de *scikit-learn* para normalizar nuestras variables independientes.

3. Modelo de clasificación

Como fue mencionado anteriormente, el modelo utilizado para la clasificación de los precios es *Random Forest*, de forma específica, el modelo utilizará los atributos listados en la sección anterior para predecir el "price_range". A pesar de que se busca predecir los precios, esto se realiza sobre una clase en lugar de directamente el valor del costo, siendo estas clases las siguientes:

- 0: Costo bajo
- 1: Costo medio
- 2: Costo alto
- 3: Costo muy alto

Random Forest es considerado un modelo de *Ensemble*, que se refiere a aquellos algoritmos que utilizan y unen un conjunto de modelos más pequeños para obtener las predicciones deseadas. *Random Forest* utiliza árboles de decisión con el método de *bagging*, indicando que estos modelos son entrenados de forma paralela. Para la implementación de este modelo se utiliza la librería *scikit-learn* con la clase *RandomForestClassifier*. Los hiperparámetros iniciales para el modelo incluían el uso de 150 árboles dentro del modelo, aplicación de *Bootstrapping* y ningún límite para la profundidad máxima de cada árbol.

4. Resultados y Evaluación

Utilizando los parámetros iniciales y entrenando el modelo con el conjunto de datos de entrenamiento, se obtienen los siguientes resultados de precisión utilizando el método de *accuracy_score* de *scikit-learn*:

- Un puntaje del 100% para los datos de entrenamiento.
- Puntaje de 92.5% con los datos de validación.
- Puntaje de 88.6% para los datos de prueba.

Aunque la precisión es buena en general, la diferencia entre los puntajes de los conjuntos indica un posible sobre ajuste del modelo, por lo que más adelante se modifican los hiperparámetros con el objetivo de reducir este comportamiento.

Estos puntajes son desplegados en las líneas 121, 124 y 127 del código de la implementación encontrado en el repositorio. Utilizando la función *show_scatter_results* Se pueden graficar algunas de estas predicciones con atributos/ejes de ejemplo:

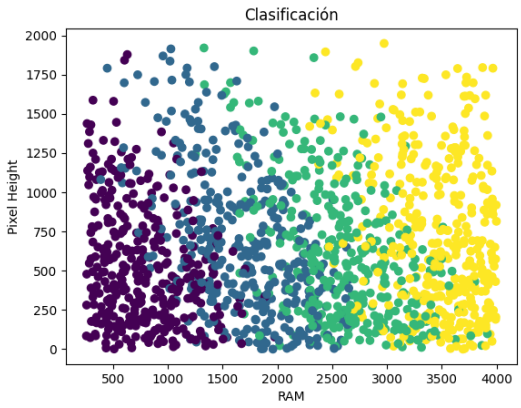

Fig 1. Clasificación de datos de entrenamiento.

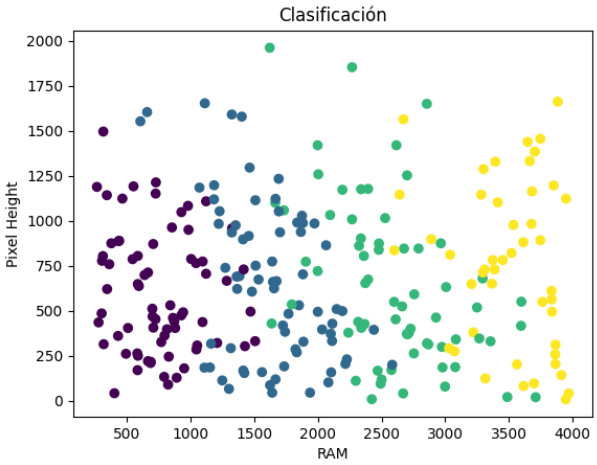

Fig 2. Clasificación de datos de validación.

El buen puntaje de precisión puede indicar un bajo *bias* en nuestro modelo, mientras que el posible sobre ajuste puede indicar un mayor nivel de varianza, sin embargo, no se ve extremadamente alterada la precisión con otros sets de datos.

5. Ajuste de Parámetros

Con el objetivo de mejorar nuestro modelo partiremos de el conjunto de datos de validación para ajustar los hiperparámetros en la clase de *RandomForestClassifier*. A continuación, se muestran una serie gráficas mostrando el desempeño al probar diferentes valores para algunos de estos parámetros:

En cuanto al número de árboles (*n_estimators*), se ve un mejor resultado al incrementar el valor de 150 a 200 para el entrenamiento, por lo que es una modificación que se puede probar.

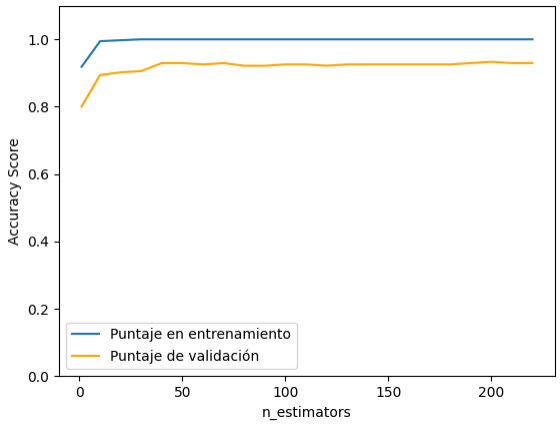

Fig 3. Precisión al entrenar con diferentes valores de *n_estimators*

Al probar con un rango de valores posibles para la máxima profundidad de cada árbol (*max_depth*) no se puede percibir un cambio del valor definido inicialmente. Posiblemente no beneficia al modelo reducir la profundidad.

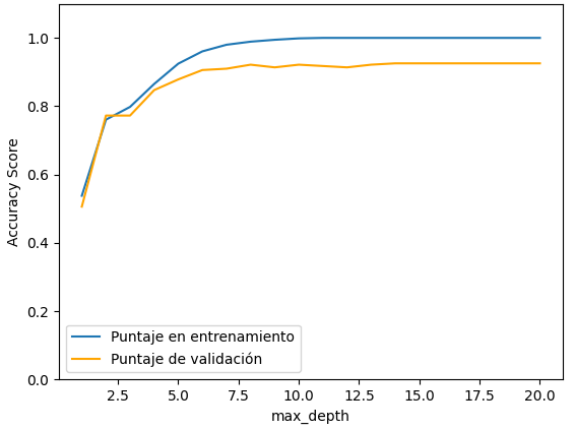

Fig 4. Precisión al entrenar con diferentes valores de *max_depth*

Por último, tampoco se notaron mejoras significativas en la precisión utilizando diferentes valores de *max_leaf_nodes*, que indica el máximo número de hojas para cada árbol en el bosque. Por lo tanto, este hiperparámetro se mantiene con el valor por defecto del modelo de *scikit-learn*

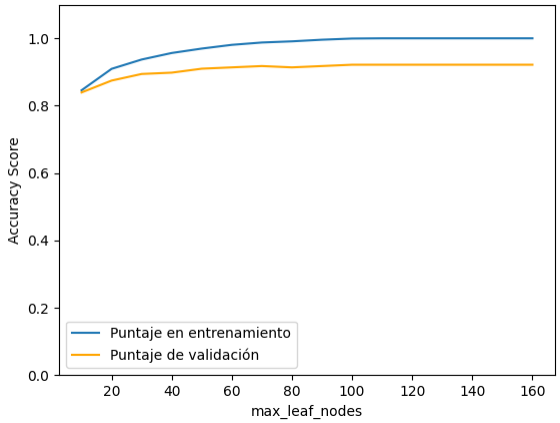

Fig 5. Precisión al entrenar con diferentes valores de *max_leaf_nodes*.

De estos tres atributos, el cambio realizado a los hiperparámetros se aplicó únicamente al número de árboles en el bosque. La obtención de estas gráficas se llevo a cabo utilizando la función *test_model_params*, definida a partir de la línea 17 del código, utilizando ciertas modificaciones para probar los diferentes atributos.

6. Conclusión y resultados finales

Al aplicar los cambios mencionados en la sección anterior, se obtienen los siguientes resultados en cuanto al puntaje de precisión:

- Un puntaje del 100% para los datos de entrenamiento.
- Puntaje de 93.3% con los datos de validación.
- Puntaje de 88.6% para los datos de prueba.

Vemos que únicamente hubo un leve incremento en la precisión con el conjunto de validación. Estos resultados indican que posiblemente se pueden realizar cambios para mejorar el modelo, sin embargo, el ajustar ciertos

hiperparámetros no se genera un beneficio notable.

Este modelo de clasificación con *Random Forest* demuestra tener un buen nivel de precisión al predecir el rango de precios de un dispositivo móvil, pero sigue existiendo cierta brecha entre la confiabilidad con diferentes conjuntos de datos. Teniendo la capacidad de realizar este tipo de predicciones, podemos facilitar el proceso de evaluación y asignación de precios para nuevos productos que salen al mercado.