Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Monterrey

Inteligencia artificial avanzada para la ciencia de datos I
TC3006C.101

Módulo 2 Análisis y Reporte sobre el desempeño del modelo. (Portafolio Análisis)

Rodolfo Sandoval Schipper A01720253

4 sept 2023

Índice general

1.- Implementación e introducción del Modelo

En esta entrega, se propone crear un modelo de clasificación utilizando el algoritmo Random Forest. Para llevar a cabo este proceso, optamos por utilizar Python como lenguaje de programación y el framework de sklearn, ya que es conocido por su facilidad de uso y una amplia gama de bibliotecas disponibles para la manipulación y análisis de datos. Estaremos utilizando un dataset numérico para mejorar el rendimiento para crear el modelo de clasificación. A continuación se muestra el proceso, el dataset utilizado, el rendimiento, y la configuración del modelo.

El objetivo es hacer un análisis de separación y evaluación del modelo con un conjunto de prueba y un conjunto de validación (Train/Test/Validation).

Hacer un diagnóstico y explicar el grado de bias o sesgo: bajo medio alto.

Implementar diagnóstico y explicación el grado de varianza: bajo medio alto, diagnóstico y explicación del nivel de ajuste del modelo: underfit o overfit, e implementar técnicas de regularización para ajustar el modelo con los hiper parámetros óptimos. A continuación se entrega la liga del archivo collab en caso de no poder acceder al archivo en el repositorio: https://colab.research.google.com/drive/1j9ytO9ttK4igz3-l_9ni-yysltrV73IT?usp=sharing

1.1 Dataset Mobile Price Classification

Bob ha iniciado su propia empresa de telefonía móvil. Quiere dar una dura lucha a las grandes empresas como Apple, Samsung, etc.

No sabe cómo estimar el precio de los móviles que fabrica su empresa. En este competitivo mercado de la telefonía móvil no se pueden simplemente dar por sentado las cosas. Para solucionar este problema recopila datos de ventas de teléfonos móviles de varias empresas.

Bob quiere descubrir alguna relación entre las características de un teléfono móvil (por ejemplo: RAM, memoria interna, etc.) y su precio de venta. Pero no es tan bueno en Machine Learning. Entonces necesita tu ayuda para resolver este problema.

En este problema no es necesario predecir el precio real, sino un rango de precios que indique qué tan alto es el precio.

El dataset que estaremos utilizando es el siguiente: https://www.kaggle.com/datasets/iabhishekofficial/mobile-price-classification?select=train.csv

Los archivos del conjunto de datos están adjuntados al repositorio cómo train móvil y test_movil.

Ejemplo del dataset:

id	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	...	pc	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi
0	1043	1	1.8	1	14	0	5	0.1	193	...	16	226	1412	3476	12	7	2	0	1	0
1	841	1	0.5	1	4	1	61	0.8	191	...	12	746	857	3895	6	0	7	1	0	0
2	1807	1	2.8	0	1	0	27	0.9	186	...	4	1270	1366	2396	17	10	10	0	1	1
3	1546	0	0.5	1	18	1	25	0.5	96	...	20	295	1752	3893	10	0	7	1	1	0
4	1434	0	1.4	0	11	1	49	0.5	108	...	18	749	810	1773	15	8	7	1	0	1

2.- Analisis con (Train/Test/Validation)

Primero cargamos los datos iniciales. Estos datos contienen características (atributos) y etiquetas (los valores que se intentan predecir). Luego hicimos la división en conjuntos de entrenamiento y de prueba. El código utiliza la función train test split de scikit-learn para dividir el conjunto de datos en dos partes: entrenamiento y prueba. La proporción de división se controla mediante el argumento test size, que se estableció en 0.2, lo que significa que el 80% de los datos se usarán para entrenar y el 20% se reservará para probar el modelo. Luego utilizamos el conjunto de entrenamiento y de validación, los conjuntos de entrenamiento (X_train y y_train) se utilizan para entrenar el modelo. Esto significa que el modelo ajustará sus parámetros y aprenderá a hacer predicciones utilizando estos datos. Esta validación se logró hacer con Grid de sklearn donde se regulariza el modelo utilizando un conjunto de diferentes parámetros a probar para darle el mejor ajuste posible con de acuerdo a las mediciones del entrenamiento con los datos reales. Esto está desplegado por medio de las gráficas donde se diagnostica el sesgo y la varianza del resultado. La información y los resultados se pueden consultar. En la siguiente liga: https://colab.research.google.com/drive/1j9ytO9ttK4igz3-l_9ni-yysltrV73IT?usp=sharing

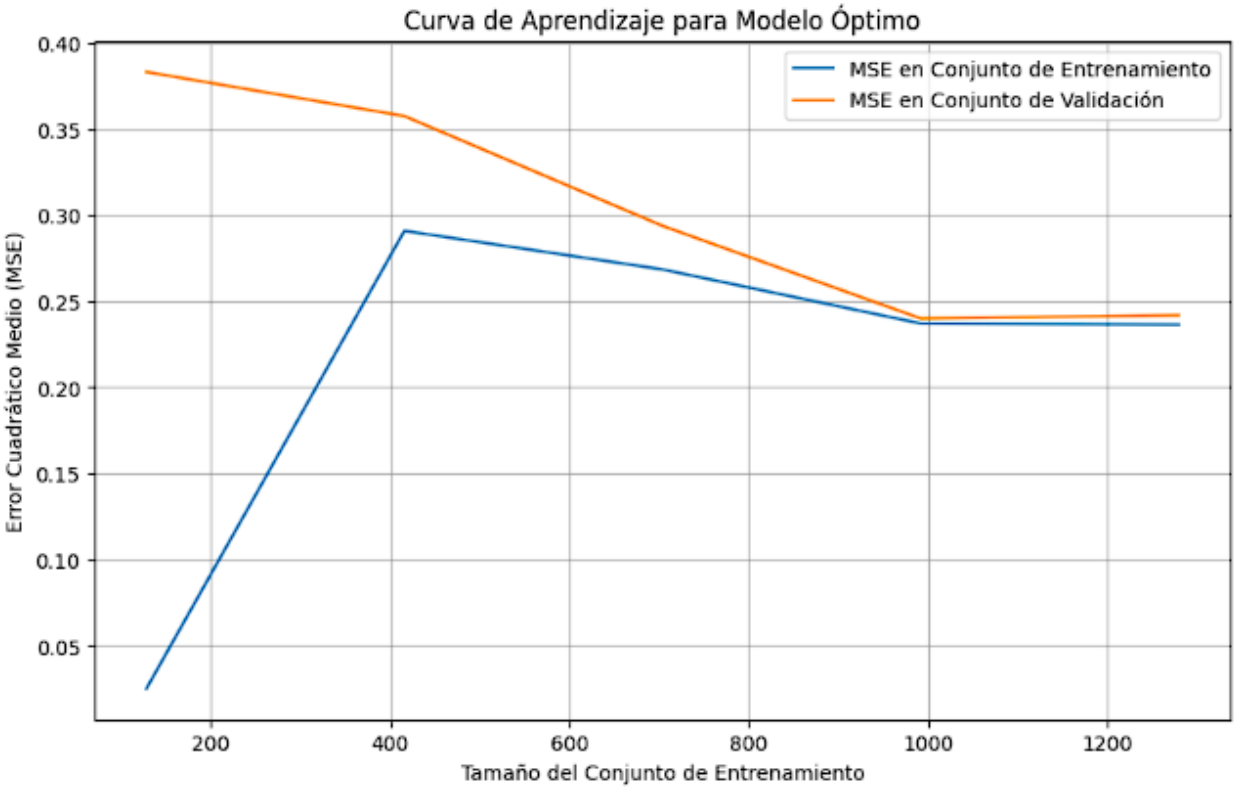
El porcentaje de precisión es de 75%. Sin embargo, era necesario obtener este porcentaje para ajustar el modelo con el mejor fitting posible. Los resultados/predicciones del modelo son los siguientes:

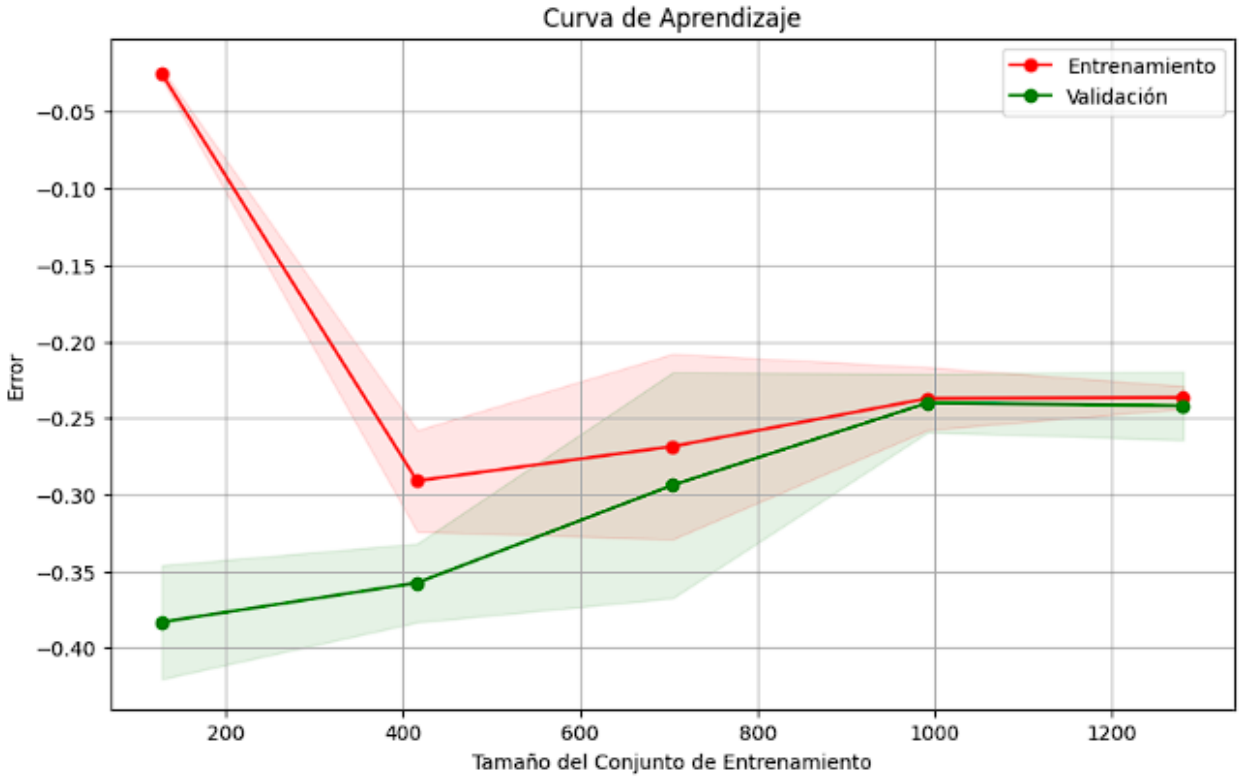
Predicciones en el conjunto de prueba:

```
[3 3 2 3 1 3 3 1 3 0 3 3 0 0 2 0 1 1 3 2 1 3 1 2 3 0 2 0 3 0 2 0 3 0 0 1 3
 1 3 2 0 2 0 0 0 1 1 2 1 2 2 0 2 1 3 1 3 1 1 3 3 3 1 1 1 1 1 2 1 1 1 1 2 3
 3 0 2 0 2 3 1 3 3 0 3 0 3 1 3 0 1 1 2 0 2 1 0 2 1 3 1 0 0 2 1 2 0 1 2 3 3
 3 1 3 3 3 3 1 3 0 0 3 2 1 1 0 3 2 3 1 0 1 1 2 3 1 1 0 3 2 1 3 2 2 3 3 3 2
 2 3 3 3 0 1 2 2 3 3 3 3 2 2 3 3 2 3 1 0 3 0 0 0 1 0 0 1 1 0 1 2 0 0 1 0 2
 2 2 1 0 0 0 0 0 3 1 1 2 1 2 3 1 2 3 3 3 1 2 0 0 1 1 2 1 2 3 3 1 2 0 3 2 2
 2 0 0 1 0 3 0 1 0 2 1 1 3 0 3 0 3 1 1 0 0 2 1 2 2 3 1 1 3 0 0 3 3 3 1 3 1
 0 3 2 1 3 3 3 3 1 0 1 1 3 1 1 3 1 0 3 1 1 2 0 0 3 2 3 2 2 1 3 2 2 3 2 2 1
 1 1 2 3 1 0 0 3 0 3 1 1 1 0 1 3 1 3 1 2 1 2 1 0 0 1 3 1 0 0 0 3 1 1 3 3 1
 3 3 2 3 1 3 3 2 2 3 3 3 0 3 1 3 1 3 1 3 3 0 1 1 3 1 3 1 3 0 0 1 0 3 0 1 1
 1 1 1 3 2 0 1 0 1 2 2 1 3 1 2 2 2 1 3 1 1 3 2 1 2 0 1 1 1 3 2 1 0 2 0 0 1
 0 0 0 0 2 2 3 2 3 0 3 1 3 1 1 1 1 2 0 2 2 3 3 1 2 1 3 1 3 2 1 2 2 2 1 0 0
 1 1 1 1 1 3 3 1 3 3 0 1 3 1 1 0 3 2 3 1 3 1 2 3 3 3 1 2 0 2 2 0 1 1 0 0 1
 1 2 3 3 3 2 2 1 1 2 2 3 3 1 1 3 2 1 2 1 0 2 3 0 1 0 3 1 1 3 2 2 0 3 0 2 2
 0 3 0 3 3 0 1 1 3 3 1 0 2 3 3 0 2 1 3 0 3 3 0 2 3 2 3 0 1 2 3 1 3 2 3 1 0
 1 0 3 1 0 3 2 3 1 0 3 3 3 2 3 3 1 1 1 3 3 3 1 0 1 1 1 2 2 1 0 2 2 3 2 0 3
 1 3 3 0 1 3 1 2 1 0 0 0 2 1 0 1 0 2 2 0 2 3 1 0 3 1 1 3 2 0 0 0 0 3 0 3 3
 1 3 2 1 3 3 1 1 1 3 2 2 1 0 3 0 1 1 2 1 1 1 1 1 2 1 3 1 3 2 1 1 3 2 0 1 3
 0 3 3 0 2 1 1 3 0 3 2 0 3 2 3 0 0 3 0 1 2 3 2 1 1 2 1 1 3 1 1 1 2 2 1 0 0
 1 0 0 3 0 1 1 0 1 1 0 2 0 3 2 3 0 0 1 2 2 1 0 1 2 0 1 2 0 0 3 3 1 3 1 2 3
 0 1 0 1 2 0 3 2 0 3 0 1 1 2 3 3 2 3 0 3 2 0 1 0 2 3 1 0 2 1 2 1 0 3 2 1 3
 1 1 1 1 1 3 1 2 1 2 0 0 1 1 0 2 0 0 1 1 3 3 3 3 0 1 2 1 1 0 0 3 1 0 2 0 2
 2 2 2 2 0 1 1 3 0 0 3 1 3 0 0 2 2 2 1 2 3 1 0 0 2 3 0 3 0 0 1 2 2 1 2 0 3
 2 1 2 2 3 0 1 1 2 0 3 2 0 1 3 1 1 3 1 2 3 1 1 1 2 3 3 0 3 3 0 2 3 2 2 2 3
 2 1 1 2 0 2 1 1 1 3 2 1 2 0 1 1 3 1 0 2 1 3 1 0 0 2 2 2 2 0 3 3 2 1 3 1 1
 3 1 2 1 2 2 3 0 3 0 2 2 0 2 1 2 2 1 1 3 3 1 0 1 1 3 1 3 0 2 2 0 2 3 3 3 0
 2 1 0 1 2 3 3 0 2 2 2 1 2 0 1 2 0 1 0 0 3 2 2 0 1 0 0 3 2 3 2 0 0 1 1 1 2
 2]
```

3.- Diagnóstico del sesgo y varianza

El diagnóstico del sesgo y la varianza en un modelo de aprendizaje automático se puede realizar a través del análisis de la curva de aprendizaje. Esta curva muestra cómo cambia el rendimiento del modelo a medida que se incrementa el tamaño del conjunto de entrenamiento.


Curva de Aprendizaje para Modelo Óptimo
MSE en Conjunto de Entrenamiento
MSE en Conjunto de Validación
Error Cuadrático Medio (MSE)
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
200
400
600
800
1000
1200
Tamaño del Conjunto de Entrenamiento


Curva de Aprendizaje
Entrenamiento
Validación
Error
−0.05
−0.10
−0.15
−0.20
−0.25
−0.30
−0.35
−0.40
200
400
600
800
1000
1200
Tamaño del Conjunto de Entrenamiento

3.1 Observaciones

Sesgo: Se puede observar que tanto el MSE en el conjunto de entrenamiento como en el conjunto de validación se estabilizan a medida que se aumenta el tamaño del conjunto de entrenamiento., significa que el modelo tiene un sesgo medio. Esto indica que el modelo tiene un buen equilibrio entre sesgo y varianza y es un buen modelo generalizador.

Varianza (Variance): Se puede observar que finalizando el entrenamiento no hay una brecha significativa entre el MSE en el conjunto de entrenamiento y el MSE en el conjunto de validación, esto indica una varianza óptima para los resultados del modelo.

Con de acuerdo a los aspectos más técnicos de cómo se implementó los métodos de validación, se reitera que se utiliza learning curve de sklearn para calcular el error cuadrático medio en el conjunto de entrenamiento y en el conjunto de validación. Luego, se convierten los valores de error en positivo multiplicando por -1, ya que learning curve devuelve valores negativos de error.

Para obtener los resultados de acuerdo al código, graficamos la curva de aprendizaje. En el eje x, se muestra el tamaño del conjunto de entrenamiento, y en el eje y, se muestra el MSE. La curva de aprendizaje se divide en dos líneas. Una representa el MSE en el conjunto de entrenamiento y otra que representa el MSE en el conjunto de validación.

4.- Diagnóstico del ajuste del modelo y parámetros

La configuración más óptima que se logró tener es de:

‘min_impurity _decrease’ →
Min impurity decrease establece un umbral en la reducción de la impureza. Si la reducción de la impureza obtenida al dividir un nodo es menor que min impurity decrease, entonces la división no se realizará, y el nodo se considerará una hoja. Si la reducción de la impureza es igual o mayor que min impurity decrease, entonces la división se realizará.

'n estimators': **Número de árboles**
'max_depth': **Profundidad de árbol**
'min_samples_split': **Número de división de nodos**

Parametro	Valor
min_impurity_decrease	0.02
max_depth	4
min_samples_split	10

n_estimators	50

Se ajustaron estos parámetros ya que el resultado nos da el mínimo MSE al finalizar el entrenamiento. Concluyendo que este ajuste es ideal para los datos que se están analizando en el modelo. Estos parámetros se ajustan en el modelo de modelo_optimo (ubicado en el código) para realizar el entrenamiento con estos hiper parámetros y fijar el mejor rendimiento posible con de acuerdo a los resultados graficados.