Proyecto del segundo parcial

Second part project

Francisco Parra [1]

[1]Escuela de Ingeniería Mecatrónica, Universidad Internacional del Ecuador, Quito, Ecuador
Autores Principales/Corresponding Authors, e-mail: frparrave@uide.edu.ec

***Abstract*— The best algorithm for classification of the dataset is created using hyperparameters and metrics to use the data as a reference to how good the algorithm is.**

***Keywords*—Algorithms, programming, learning models.**

***Resumen*— Se crea el mejor algoritmo para clasificación del dataset usando hiperparametros y métricas para usar los datos como referencia a que tan buen algoritmo es.**

***Palabras Clave*—Algoritmos, programaciòn, modelos de aprendizaje.**

I. Introducción

Limpieza de datos:

Tratar valores atípicos: Corregir errores de ortografía y formato: Los errores de ortografía y formato pueden dificultar el procesamiento de datos por parte del modelo. Es importante corregirlos para garantizar la coherencia y precisión de los datos. 2. Normalización de datos:

Escalar los datos: Los datos pueden tener diferentes escalas, lo que puede afectar el proceso de aprendizaje del modelo. La normalización implica transformar los datos a una escala común, como entre 0 y 1 o -1 y 1. Estandarizar los datos: La estandarización es similar a la normalización, pero implica centrar los datos en la media y ajustarlos a una desviación estándar de 1.

- **Eliminar valores ausentes:** Es común que los archivos CSV contengan valores faltantes, lo que puede afectar el rendimiento del modelo. Estos valores pueden eliminarse, imputarse (rellenarse con valores estimados) o excluirse de los análisis.
- **Tratar valores atípicos:** Los valores atípicos son puntos de datos que se desvían significativamente del resto de los datos. Pueden ser errores de medición o representar eventos inusuales. Es importante identificarlos y tratarlos adecuadamente para evitar que distorsionen el modelo.
- **Aprendizaje semi-supervisado:** Es una técnica intermedia entre el aprendizaje supervisado y el no supervisado. Realiza acciones en conjuntos de datos que tienen pocas etiquetas, así como en los que están sin etiquetar. Sin embargo, generalmente contiene datos sin etiquetar [**?**].
- **Aprendizaje por refuerzo:** Es un tipo de aprendizaje automático en el que no hay capacitación con datos clasificados o no clasificados [**?**]. Es una técnica del ML basada en la retroalimentación. Los agentes deben explorar el entorno y realizar acciones basadas en sus acciones [**?**].

II. Metodología

Para el primer modelo usamos Random Forest, que se utiliza como un algoritmo de aprendizaje automático robusto y versátil para realizar tanto tareas de clasificación como de regresión.

Random Forest generalmente logra una alta precisión en las predicciones, especialmente cuando se compara con modelos de aprendizaje automático individuales, como árboles de decisión simples.

XGBoost es un poderoso algoritmo de aprendizaje automático que sobresale en tareas como clasificación y regresión, lo que lo hace ideal para la predicción de datos en archivos CSV utilizando Python. Se basa en el concepto de potenciación de gradiente, donde se entrenan modelos múltiples (a menudo árboles de decisión) de forma secuencial, y cada modelo se enfoca en corregir los errores de los anteriores.

III. Resultados y Discusión

```python
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=100,
                            max_depth=None,
                            min_samples_split=3,
                            min_samples_leaf=4,
                            random_state=30,
                            bootstrap=True,
                            oob_score=True,
                            class_weight=None,
                            n_jobs=-2)
```

✓ 0.0s

Fig. 1. Ejemplo del primer modelo modelo.

```python
from xgboost import XGBClassifier
from sklearn.metrics import classification_report

xgb = XGBClassifier(n_estimators=70,
                    max_depth=6,
                    learning_rate=0.1,
                    gamma=0,
                    reg_lambda=3,
                    colsample_bytree=1,
                    colsample_bylevel=1,
                    min_child_weight=1,
                    max_delta_step=0,
                    seed=42)
```

✓ 0.0s

Fig. 2. Ejemplo del segundo modelo modelo.

```
Matriz de confusión:
 [[6659  360]
 [1213  768]]
Exactitud: 0.83
Precision: 0.68
Sensibilidad: 0.39
Especificidad: 0.95
f1-Score: 0.49
```

Fig. 3. Resultado del primer modelo.

```
Matriz de confusión:
 [[6674  345]
 [1227  754]]
Exactitud: 0.83
Precision: 0.69
Sensibilidad: 0.38
Especificidad: 0.95
f1-Score: 0.49
```

Fig. 4. Resultado del segundo modelo.