Act 13: Programando Random Forest en Python

Ana Isabel Loera Gil

30 de marzo del 2025

1 Introducción

Random Forest es un ensamble en Machine Learning en donde se combinan diversos árboles. Es un modelo de aprendizaje supervisado para clasificación. Funciona de la siguiente manera:

- Se selecciona k features de m totales (siendo k menor a m) y se crea un árbol de decisión con esas k características.
- Se crean n árboles variando siempre la cantidad de k features y también se puede variar la cantidad de muestras que se pasan esos árboles.
- Se toma cada uno de los n árboles y se hace una clasificaión. Se guarda el resultado de cada árbol en n salidas.
- Se calculan los votos obtenidos por cada "clase" seleccionada y se considera la mas votada como la clasificación final del "bosque"

2 Metodología

2.1 Descargar el archivo csv de Kaggle

https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud/data

2.2 Importación de librerias

```
import pandas as pd
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
```

2.3 Leer el archivo csv y mostrar los primeros 5 registros

```
dataframe= pd.read_csv(r"creditcard.csv")
print(dataframe.head())
```

2.4 Se cuenta cuántas clases hay

```
count_classes =pd.value_counts(dataframe['Class'],sort=True)
```

2.5 Creación del gráfico de barras de las clases

```
count_classes.plot(kind='bar',rot=0)
LABELS = ["No Fraud", "Fraud"]
plt.xticks(range(2),LABELS)

plt.title("Frequency by observation number")
plt.xlabel("Class")
plt.ylabel("Number of OBservations")
plt.show()
```

2.6 Definición de etiquetas y features

```
y = dataframe['Class']
X = dataframe.drop('Class', axis=1)
```

2.7 Definición de sets de entrenamiento y test

```
X_train, X_test, y_train, y_test=train_test_split(X, y, train_size=0.7)
```

2.8 Creación y entrenamiento del modelo

```
model = RandomForestClassifier(n_estimators=100,
                               bootstrap=True, verbose=2,
                               max_features='sqrt')
model.fit(X_train, y_train)
```

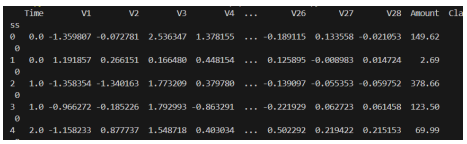
2.9 Reporte de clasificación

```
predictions = model.predict(X_test)
print("Reporte de clasificacion")
print(classification_report(y_test,predictions))
```

2.10 Representación en mátriz de confusion del reporte de clasificación

```
conf_matrix= confusion_matrix(y_test, predictions)
plt.figure(figsize=(12,12))
sns.heatmap(conf_matrix, xticklabels=LABELS, yticklabels=LABELS, annot=True, fmt="d")
plt.title("Confusion matrix")
plt.ylabel("True class")
plt.xlabel("Predict class")
plt.show()
```

3 Resultados



	Time	V1	V2	V3	V4	...	V26	V27	V28	Amount	Class
0	0.0	-1.359807	-0.072781	2.536347	1.378155	...	-0.189115	0.133558	-0.021053	149.62	0
1	0.0	1.191857	0.266151	0.166480	0.448154	...	0.125895	-0.008983	0.014724	2.69	0
2	1.0	-1.358354	-1.340163	1.773209	0.379780	...	-0.139097	-0.055353	-0.059752	378.66	0
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	...	-0.221929	0.062723	0.061458	123.50	0
4	2.0	-1.158233	0.877737	1.548718	0.403034	...	0.502292	0.219422	0.215153	69.99	0

Figure 1: Primeros registros

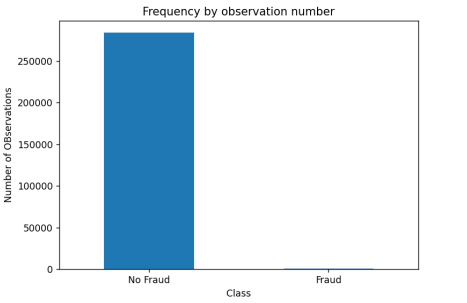


Figure 2: Gráfico de barras de la frecuencia del número de observacion

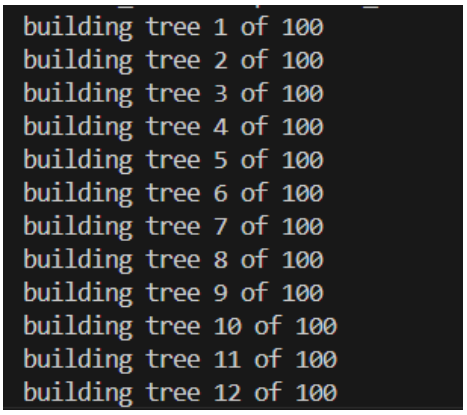


Figure 3: Entrenamiento del modelo

Reporte de clasificacion

	precision	recall	f1-score	support
0	1.00	1.00	1.00	85285
1	0.97	0.78	0.86	158
accuracy			1.00	85443
macro avg	0.98	0.89	0.93	85443
weighted avg	1.00	1.00	1.00	85443

Figure 4: Reporte de clasificación

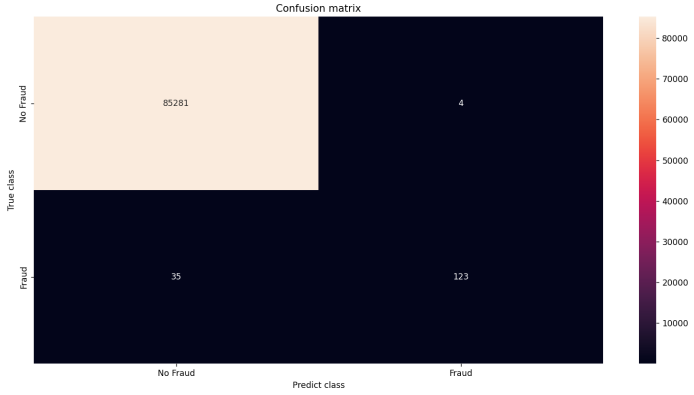


Figure 5: Mátriz de confusión del reporte de clasificación

4 Conclusión

El Random Forest es un modelo que, aunque es menos intuitivo que un solo árbol de decisión, ayuda a evitar el overfitting y a obtener mejores resultados. Me di cuenta de que es un modelo rápido y fácil de usar. A medida que sigo explorando diferentes modelos, entiendo mejor cómo elegir la mejor opción según el problema que quiero resolver.

5 Referencias bibliograficas

Ignacio Bagnato, J. (2020). Aprende machine learning. Leanpub.