ANÁLISIS PREDICTIVO Y MINERÍA DE DATOS

COMPRENDER Y APLICAR TÉCNICAS DE MINERÍA DE DATOS PARA LA CREACIÓN DE MODELOS PREDICTIVOS EN EL CONTEXTO DEL BIG DATA.

Andrei Alexandru Miu

Índice

Investigación teórica:

- **Explica qué es el análisis predictivo y su relación con la minería de datos.**

El **análisis predictivo** es un enfoque analítico que utiliza datos históricos y actuales para predecir eventos futuros o tendencias.
La **minería de datos** juega un papel fundamental en el análisis predictivo, ya que es el proceso de descubrir patrones, relaciones y tendencias significativas en grandes conjuntos de datos.

- **Describe al menos tres técnicas utilizadas en minería de datos para el análisis predictivo** (ejemplo: árboles de decisión, regresión logística, redes neuronales).

1. **Árboles de decisión**:
    - Los árboles de decisión son estructuras de flujo lógico que dividen iterativamente un conjunto de datos en subconjuntos más pequeños basados en atributos específicos.
    - Cada nodo del árbol representa una decisión o criterio, y las hojas representan los resultados finales. Es una técnica popular debido a su facilidad de interpretación y visualización.
2. **Regresión logística:**
    - Este método se utiliza para modelar relaciones entre una variable dependiente binaria (como "sí" o "no") y una o más variables independientes.
    - Se basa en el uso de una función logística para predecir probabilidades.
3. **Redes neuronales:**
    - Las redes neuronales son modelos inspirados en el cerebro humano que consisten en capas de neuronas artificiales interconectadas.
    - Son capaces de modelar relaciones complejas y no lineales en los datos, lo que las hace extremadamente poderosas en el análisis predictivo.

- **Investiga casos de uso reales donde se haya aplicado análisis predictivo en Big Data.**

Algunos casos de uso reales donde se haya aplicado el análisis predictivo:

1. **Sector de la salud:**

Instituciones médicas utilizan análisis predictivo para predecir la probabilidad de enfermedades en pacientes basándose en datos médicos históricos.

- Por ejemplo, los algoritmos predictivos ayudan a identificar pacientes con alto riesgo de enfermedades cardiovasculares.

2. **Sector financiero**

Bancos y entidades financieras aplican análisis predictivo para detectar fraudes en tiempo real.

- Por ejemplo, sistemas como los utilizados por Visa analizan patrones de transacciones para identificar actividades sospechosas.

3. **Plataformas digitales**

Plataformas como Amazon o Netflix aplican análisis predictivo para ofrecer recomendaciones personalizadas a los usuarios basadas en su historial de compras o preferencias.

- Esto mejora la experiencia del usuario y aumenta la conversión.

Aplicación práctica:

- **Elige un conjunto de datos abierto (ejemplo: Kaggle, UCI Machine Learning Repository).**

Elegimos un dataset sobre las precipitaciones de las regiones de España de 2021, de enero a diciembre y una columna Anual.
Puedes verlo pinchando aquí, o en el archivo adjunto a la práctica.

- **Preprocesa los datos (limpieza, transformación y reducción si es necesario).**
- **Implementa un modelo de análisis predictivo usando Python y una herramienta de minería de datos como Scikit-learn, TensorFlow o Weka.**

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import precision_score, recall_score, f1_score, classification_report, confusion_matrix

# Cargar datos
file_path = "PREC_2021_Provincias.csv"
df = pd.read_csv(file_path, sep=";")

# Preprocesamiento de datos
# Seleccionamos las columnas relevantes para predecir precipitaciones anuales altas/bajas
df["HighPrecipitation"] = (df["anual"] > 1000).astype(int)  # Etiquetamos si la precipitación anual es >1000 mm
X = df.iloc[:, 2:14]  # Datos de enero a diciembre
y = df["HighPrecipitation"]  # Etiqueta de clasificación

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Entrenar un modelo (Random Forest en este caso)
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

# Realizar predicciones
y_pred = clf.predict(X_test)

# Evaluar el modelo usando las métricas
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

# Mostrar resultados
print("Precisión:", precision)
print("Recall:", recall)
print("F1-Score:", f1)
print("\nReporte de Clasificación:")
print(classification_report(y_test, y_pred))
print("\nMatriz de Confusión:")
print(confusion_matrix(y_test, y_pred))
```

Cargamos los datos, preprocesamos y seleccionamos lo relevante.
Posteriormente, dividimos los datos y entrenamos un modelo.
Luego, hacemos predicciones y lo evaluamos mostrando finalmente los resultados.

- **Evalúa el rendimiento del modelo con métricas adecuadas (precisión, recall, F1-score).**

```
PS D:\WorkSpace_VSCode_IABD\SistemasDeBigData\Prácticas\Tema7> python .\ejercicio.py
Precisión: 1.0
Recall: 1.0
F1-Score: 1.0

Reporte de Clasificación:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        15
           1       1.00      1.00      1.00         1

    accuracy                           1.00        16




    accuracy                           1.00        16
   macro avg       1.00      1.00      1.00        16
weighted avg       1.00      1.00      1.00        16

    accuracy                           1.00        16
   macro avg       1.00      1.00      1.00        16

    accuracy                           1.00        16



    accuracy                           1.00        16
   macro avg       1.00      1.00      1.00        16
weighted avg       1.00      1.00      1.00        16


Matriz de Confusión:
[[15  0]
 [ 0  1]]
PS D:\WorkSpace_VSCode_IABD\SistemasDeBigData\Prácticas\Tema7>
```

Al ejecutar, nos salen los datos.