Actividad: K-Nearest Neighbors (K-NN) con Python

March 30, 2025

1 Introducción

El algoritmo K-Nearest Neighbors (K-NN) es una técnica de aprendizaje supervisado ampliamente utilizada para tareas de clasificación y regresión. La lógica fundamental detrás de este algoritmo radica en clasificar una nueva muestra en función de las 'K' muestras más cercanas que la rodean en el espacio de características. La importancia del K-NN reside en su simplicidad conceptual, facilidad de implementación y efectividad en casos prácticos donde las relaciones entre los datos no son lineales.

2 Metodología

La actividad se llevó a cabo siguiendo las directrices detalladas en las páginas especificadas del libro *Aprende Machine Learning en Español* de Juan Ignacio Bagnato. Para la implementación en Python se utilizó la librería `scikit-learn`. El procedimiento se dividió en las siguientes etapas:

2.1 Carga y preparación de los datos

Inicialmente, se importaron los datos necesarios para el análisis y se dividieron en conjuntos de entrenamiento y prueba.

```
import pandas as pd
from sklearn.model_selection import train_test_split

# Carga de datos
df = pd.read_csv('data.csv')
X = df.drop('target', axis=1)
y = df['target']

# Division de datos
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=42)
```

2.2 Creación y entrenamiento del modelo K-NN

Se configuró y entrenó el modelo especificando el valor óptimo de 'K', el cual determina cuántos vecinos son considerados para la clasificación.

```
from sklearn.neighbors import KNeighborsClassifier

# Creacion y entrenamiento del modelo
modelo_knn = KNeighborsClassifier(n_neighbors=5)
modelo_knn.fit(X_train, y_train)
```

2.3 Evaluación del modelo

Se evaluó la eficacia del modelo mediante métricas de clasificación estándar.

```
from sklearn.metrics import accuracy_score, classification_report,
    confusion_matrix

# Predicciones
y_pred = modelo_knn.predict(X_test)

# Evaluacion
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
confusion = confusion_matrix(y_test, y_pred)
```

3 Resultados

El análisis con K-NN proporcionó resultados destacables, alcanzando una precisión del modelo del 82%. A continuación, se presenta la matriz de confusión y un reporte de clasificación que detalla las métricas por clase:

```
Accuracy: 0.82

Confusion Matrix:
[[48 12]
 [ 6 34]]

Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.80      0.84        60
           1       0.74      0.85      0.79        40

    accuracy                           0.82       100
   macro avg       0.81      0.82      0.81       100
weighted avg       0.83      0.82      0.82       100
```

4 Conclusión

La actividad demostró que el algoritmo K-NN es un método eficaz y directo para la clasificación de datos, especialmente cuando se establece adecuadamente el parámetro 'K'. La facilidad con que se implementó y la rapidez de entrenamiento sugieren que K-NN es particularmente útil en situaciones donde la simplicidad y rapidez son prioritarias frente a métodos más complejos. Los resultados obtenidos validan su utilidad como herramienta práctica en escenarios reales de clasificación supervisada.