UNIDAD TEMÁTICA 3: Algoritmos Lineales

Trabajo de Aplicación 7 - Análisis Discriminante Lineal utilizando Python y scikit-learn

Ejercicio 1 - Tutorial scikit-learn

El siguiente ejercicio es un tutorial para realizar clasificación a partir de un conjunto de datos, usando Análisis Discriminante Lineal y la librería scikit-learn de Python.

En cada paso verificar la documentación para ver cuales son las opciones de parámetros que se pueden usar en cada función.

Parte 1: carga de datos y preparación

1. Descargar de Webasignatura el dataset de ejemplo sample.csv

2. Crear un archivo Python y agregar las dependencias necesarias

```
import matplotlib
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split
```

3. Leer el dataset desde el archivo CSV utilizando la librería Pandas. Y ver como esta compuesto.

```
input_file = "sample.csv"
df = pd.read_csv(input_file, header=0)
print(df.values)
```

4. Graficar los datos utilizando la librería.

```
colors = ("orange", "blue")
plt.scatter(df['x'], df['y'], s=300, c=df['label'], cmap=matplotlib.colors.ListedColormap(colors))
plt.show()
```

5. Obtener a partir del dataset los datos y las clases.

```
X = df[['x', 'y']].values
y = df['label'].values
```

Parte 2: entrenamiento y testing

6. Dividir el conjunto de datos en 2, uno para entrenamiento y otro para prueba.
   train_X, test_X, train_y, test_y = train_test_split(X, y, test_size=0.25, random_state=0, shuffle=True)
7. Crear el un modelo de LDA y entrenarlo.
   lda = LinearDiscriminantAnalysis()
   lda = lda.fit(train_X, train_y)

Parte 3: evaluación

8. Predecir las clases para los datos del conjunto de prueba y ver los resultados.
   y_pred = lda.predict(test_X)
   print("Predicted vs Expected")
   print(y_pred)
   print(test_y)

9. Probar el modelo y ver el reporte. Observar las columnas y que significan.
   print(classification_report(test_y, y_pred, digits=3))

10. Ver la matriz de confusión y analizar los resultados.

    print(confusion_matrix(test_y, y_pred))

Parte 4 (opcional): Regresión Logística

11. Realizar el mismo procedimiento utilizando Regresión Logística y comparar los resultados.

    lr = LogisticRegression()

Ejercicio 2 – Dataset sport-training.csv

Realizar nuevamente el ejercicio del trabajo de aplicación 6 para la clasificación de deportistas utilizando scikit-learn. En cada paso comparar los resultados con los obtenidos utilizando RapidMiner.

1. Realizar el entrenamiento con los datos del archivo sports_Training.csv
2. Eliminar filas cuyo valor para el atributo 'CapacidadDecision' estan fuera de los limites. Esto se puede hacer de la siguiente forma utilizando la libreria Pandas

   ```
   data = data _original[(data _original['CapacidadDecision'] >= 3) &
                         (data _original['CapacidadDecision'] <= 100)]
   ```

3. Transformar atributos en string a numeros
   le = LabelEncoder()
   y_encoded = le.fit_transform(y)

   Para hacer el proceso inverso
   le.inverse_transform(y_encoded)
4. Usar los datos del archivo sports_Scoring.csv para clasificar los nuevos individuos uilizando el modelo entrenado anteriormente. Comparar los resultados con los obtenidos en el TA6.