Programación de Inteligencia Artificial – Tarea – Unidad04

Usaremos Google Colab para ejecutar las líneas de código necesarias, el enlace a dicho cuaderno es:

Enlace → PIA04_Tarea_Notebook

Apartado 1: Explora los datos con Pandas

Para esta práctica usaremos el dataset 'Iris', este conjunto de datos contiene información sobre tres especies de flores (Setosa, Versicolor y Virginica) y sus medidas de pétalos y sépalos. El objetivo principal es visualizar y trabajar con esta distribución de datos.

Comenzamos importando las librerías necesarias, estas serán Pandas (manipulación de datos), NumPy (operaciones matemáticas) y Scikit-learn (manejo de datasets)

Importamos las librerías requeridas, e importamos la función 'load_iris':

```
[1]  import numpy as np
     import pandas as pd
     from sklearn.datasets import load_iris
```

Cargamos el dataset Iris y trabajamos con las principales caracteristicas

```
[2]  iris = load_iris()
```

```
print("Datos:", iris.data) #Mostramos las filas de datos
print("Etiquetas:", iris.target) # Mostramos los valores de las etiquetas
print("Nombres de las especies:", iris.target_names) # Nombres de las especies cargadas en el dataset
```

Mostrar salida oculta

Creamos un Dataframe de Pandas y añadimos la columna 'species' que contiene la información de la especie de cada flor:

```
[6]  df = pd.DataFrame(iris.data, columns=iris.feature_names)
     df['Species'] = iris.target
```

Visualizamos los primeros registros del dataframe creado para tener una idea de como se ven los datos con los que vamos a trabajar:

```
print("Primeros registros del dataset:")
print(df.head())
```

```
Primeros registros del dataset:
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

   Species
0        0
1        0
2        0
3        0
4        0
```

Obtenemos las estadísticas descriptivas del dataset, esto nos dará información estadística de cada variable, lo cual es útil para entender la distribución de datos

```
print("\nEstadísticas del dataset:")
print(df.describe())
```

```
Estadísticas del dataset:
       sepal length (cm)  sepal width (cm)  petal length (cm)  \
count         150.000000        150.000000         150.000000
mean            5.843333          3.057333           3.758000
std             0.828066          0.435866           1.765298
min             4.300000          2.000000           1.000000
25%             5.100000          2.800000           1.600000
50%             5.800000          3.000000           4.350000
75%             6.400000          3.300000           5.100000
max             7.900000          4.400000           6.900000

       petal width (cm)     Species
count        150.000000  150.000000
mean           1.199333    1.000000
std            0.762238    0.819232
min            0.100000    0.000000
25%            0.300000    0.000000
50%            1.300000    1.000000
75%            1.800000    2.000000
max            2.500000    2.000000
```

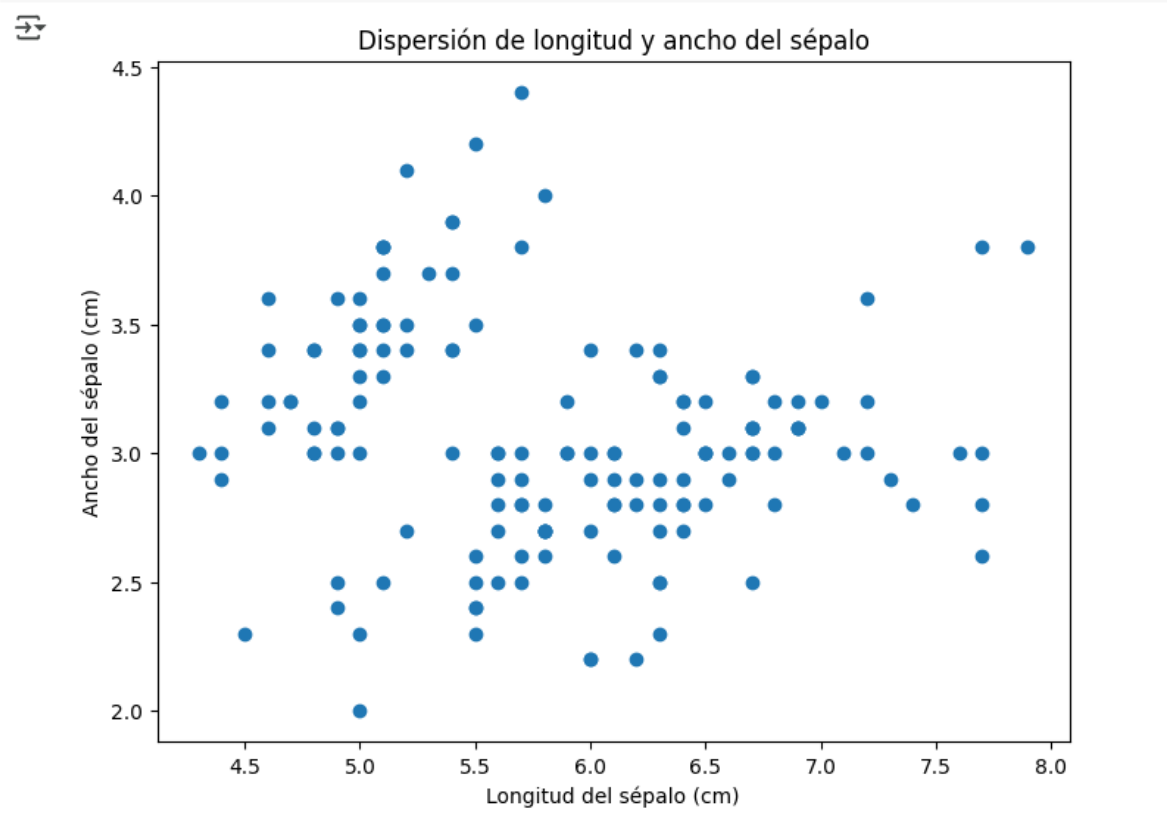
Apartado 2: Visualiza los datos con Pyplot

Importamos pyplot de la librería matplotlib la cual usaremos para crear gráficos:

```
[10] import matplotlib.pyplot as plt
```

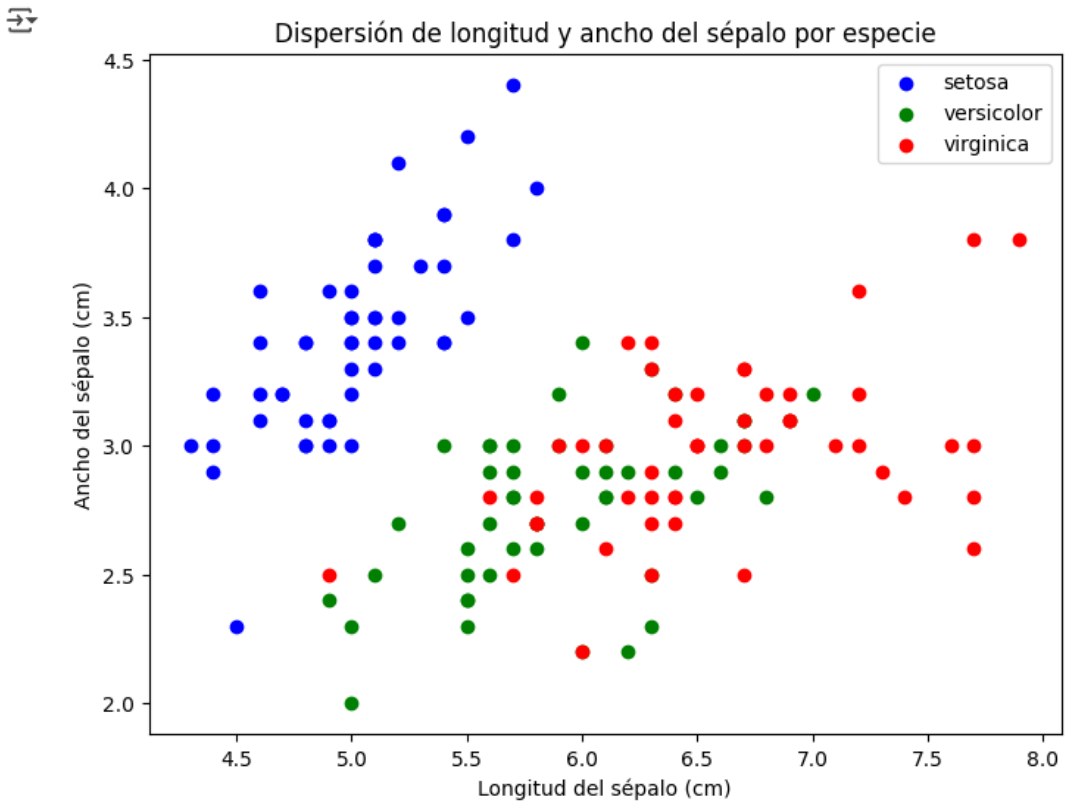
Creamos gráficos de dispersión para visualizar la relación entre las variables. En primer lugar representamos la longitud del sépalo frente al ancho del sépalo:

```
plt.figure(figsize=(8, 6))
plt.scatter(df['sepal length (cm)'], df['sepal width (cm)'])
plt.xlabel('Longitud del sépalo (cm)')
plt.ylabel('Ancho del sépalo (cm)')
plt.title('Dispersión de longitud y ancho del sépalo')
plt.show()
```



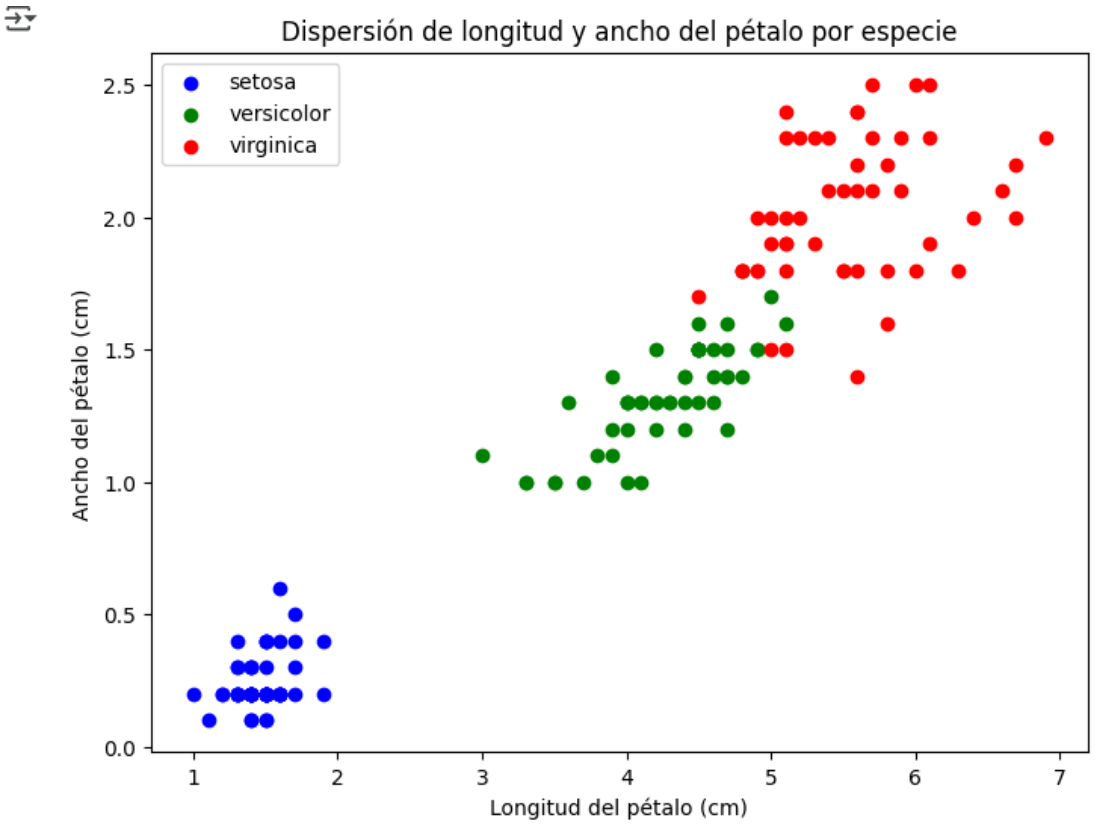
En segundo lugar, representamos lo mismo pero diferenciando por especies

```
colors = ['blue', 'green', 'red']
species_names = iris.target_names
plt.figure(figsize=(8, 6))
for i in range(len(species_names)):
    species_data = df[df['Species'] == i]
    plt.scatter(species_data['sepal length (cm)'], species_data['sepal width (cm)'], c=colors[i], label=species_names[i])

plt.xlabel('Longitud del sépalo (cm)')
plt.ylabel('Ancho del sépalo (cm)')
plt.title('Dispersión de longitud y ancho del sépalo por especie')
plt.legend()
plt.show()
```



Finalmente, representamos la longitud del pétalo frente al ancho para diferentes especies:

```
plt.figure(figsize=(8, 6))
for i in range(len(species_names)):
    species_data = df[df['Species'] == i]
    plt.scatter(species_data['petal length (cm)'], species_data['petal width (cm)'], c=colors[i], label=species_names[i])

plt.xlabel('Longitud del pétalo (cm)')
plt.ylabel('Ancho del pétalo (cm)')
plt.title('Dispersión de longitud y ancho del pétalo por especie')
plt.legend()
plt.show()
```



Apartado 3: Entrena modelos de aprendizaje automático con Scikit-learn

Importamos varios modelos de aprendizaje automático de Scikit-learn: Regresión Logística, Máquinas de Vectores de Soporte (SVC), K Vecinos Más Cercanos (KNN) y Árbol de Decisión.

```
[20] from sklearn.model_selection import train_test_split
     from sklearn.linear_model import LogisticRegression
     from sklearn.svm import SVC
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.metrics import accuracy_score
```

Separamos las características (X) de la variable objetivo (y), que en este caso es la especie de iris.

```
[21] X = df.drop('Species', axis=1)
     y = df['Species']
```

Dividimos los datos en conjuntos de entrenamiento y prueba usando ‘train_test_split’

```
[25] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

Entrenamos cada modelo con los datos de entrenamiento

```
models = {
    "Regresión Logística": LogisticRegression(),
    "SVC": SVC(),
    "KNN": KNeighborsClassifier(),
    "Árbol de Decisión": DecisionTreeClassifier()
}
```

Evaluamos la precisión de cada modelo en los datos de prueba, usando la función 'accuracy_score'

```
[28] for name, model in models.items():
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)
        print(f"Precisión de {name}: {accuracy}")
```

```
Precisión de Regresión Logística: 1.0
Precisión de SVC: 0.9333333333333333
Precisión de KNN: 0.9666666666666667
Precisión de Árbol de Decisión: 0.9333333333333333
```

Apartado 4: Entrena modelos de aprendizaje automático con pocas variables

Creamos un nuevo dataset que solo contiene las dimensiones de los sépalos y la especie

```
[30] df_sepalo = df[['sepal length (cm)', 'sepal width (cm)', 'Species']]
```

Repetimos el proceso de separar datos, dividir en conjuntos de entrenamiento y prueba, entrenar modelos y evaluar la precisión, pero esta vez usando solo las dimensiones de los sépalos.

```
[31] X_sepalo = df_sepalo.drop('Species', axis=1)
     y_sepalo = df_sepalo['Species']
```

```
[32] X_train_s, X_test_s, y_train_s, y_test_s = train_test_split(X_sepalo, y_sepalo, test_size=0.2)
```

```
[33] for name, model in models.items():
        model.fit(X_train_s, y_train_s)
        y_pred_s = model.predict(X_test_s)
        accuracy_s = accuracy_score(y_test_s, y_pred_s)
        print(f"Precisión de {name}: {accuracy_s}")
```

```
Precisión de Regresión Logística: 0.7333333333333333
Precisión de SVC: 0.7333333333333333
Precisión de KNN: 0.7666666666666667
Precisión de Árbol de Decisión: 0.6666666666666666
```

La reducción de variables (de usar sépalos y pétalos a solo sépalos) afecta negativamente el rendimiento de los modelos, aunque no considerablemente. Esto indica que los pétalos contienen información más relevante para la clasificación de especies que los sépalos, aunque estos últimos también aportan información útil.