UNIVERSIDAD DE CUNDINAMARCA (sede Facatativá)

INGENIERÍA DE SISTEMAS Y COMPUTACIÓN (8° Semestre)

MACHINE LEARNING

ALUMNO: JOHAN ARLEY SIERRA LOPEZ

YOHAN STIVEN CONTRERAS MARTINEZ

DOCENTE: ALEXANDER ESPINOZA

ACTIVIDAD: DATASET IRIS [] REGRESION LINEAL

FACATATIVÁ - CUNDINAMARCA 2025

En el presente documento se presentará el informe pertinente para la actividad de regresión lineal en base al dataset iris, siguiendo una secuencia de pasos que se realizo para llevar a cabo el desarrollo de esta actividad, los cuales son:

1) Paso a paso (qué hace cada parte del código)

1. **Importaciones**
    - numpy, pandas: utilidades de datos.
    - sklearn.datasets: para cargar el dataset Iris integrado.
    - LogisticRegression: el modelo que se usa para clasificación.
    - train_test_split: para dividir datos en entrenamiento/prueba.
    - accuracy_score, confusion_matrix, classification_report: métricas de evaluación.
2. **Cargar el dataset y entrenar el modelo**

```
# === 1) Cargar datos y entrenar modelo ===
iris = datasets.load_iris()
X, y = iris.data, iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

model = LogisticRegression(max_iter=200)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

3. **Matriz de confusión**

```
# === 2) Matriz de confusión ===
cm = confusion_matrix(y_test, y_pred)
labels = iris.target_names

plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=labels, yticklabels=labels)
plt.title("Matriz de Confusión")
plt.xlabel("Predicción")
plt.ylabel("Verdadero")
plt.tight_layout()
plt.show()
```

4. **Grafico de reporte de clasificación**

```python
# === 3) Gráfico del reporte de clasificación ===
# Convertimos el reporte a diccionario para graficar
report_dict = classification_report(y_test, y_pred, target_names=labels, output_dict=True)
report_df = pd.DataFrame(report_dict).transpose().iloc[:-3, :]  # quitar accuracy y promedios

# Graficar precision, recall, f1-score por clase
report_df[['precision','recall','f1-score']].plot(kind='bar', figsize=(8,5))
plt.title("Reporte de Clasificación")
plt.ylabel("Score")
plt.ylim(0, 1.1)
plt.legend(loc='lower right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```

---

2) Funcionalidad de la "regresión lineal" en el código (aclaración técnica)

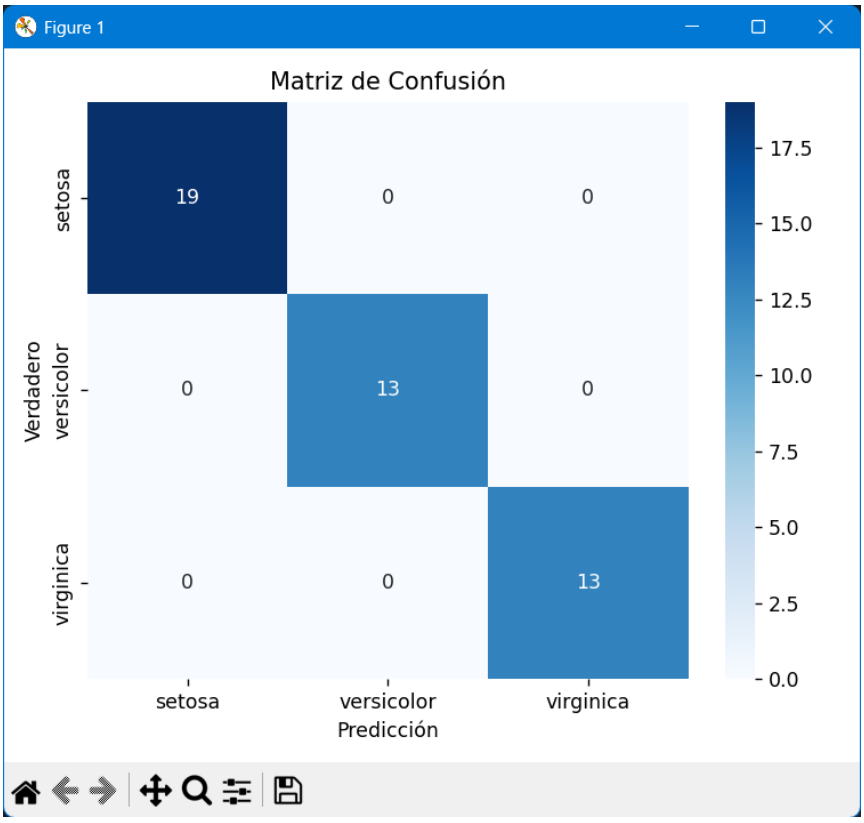
- **En el código se usa LogisticRegression (regresión logística)**, no regresión lineal convencional.
    - La **regresión logística** es un modelo *lineal* en las características (es decir, calcula una combinación lineal $w \cdot x + b$) pero aplica una función logística (sigmoid para binaria, softmax para multiclass) para convertir esa combinación en **probabilidades de clase**.
    - Para *multiclase*, scikit-learn usa internamente la formulación multinomial (softmax) con el solver lbfgs (por defecto), o un esquema one-vs-rest dependiendo de los parámetros. En la práctica el resultado: **decisiones basadas en fronteras lineales** en el espacio de características.
- **Interpretación de coeficientes**
    - Cada vector de coeficientes (uno por clase en la salida) indica cómo cambia la **log-odds** de esa clase cuando aumenta cada característica.
    - Ejemplo: virginica tiene coeficientes grandes y positivos para *largo pétalo* y *ancho pétalo* → pétalos más grandes aumentan la probabilidad de virginica.

---

3) Resultados (ejecución con random_state=42)

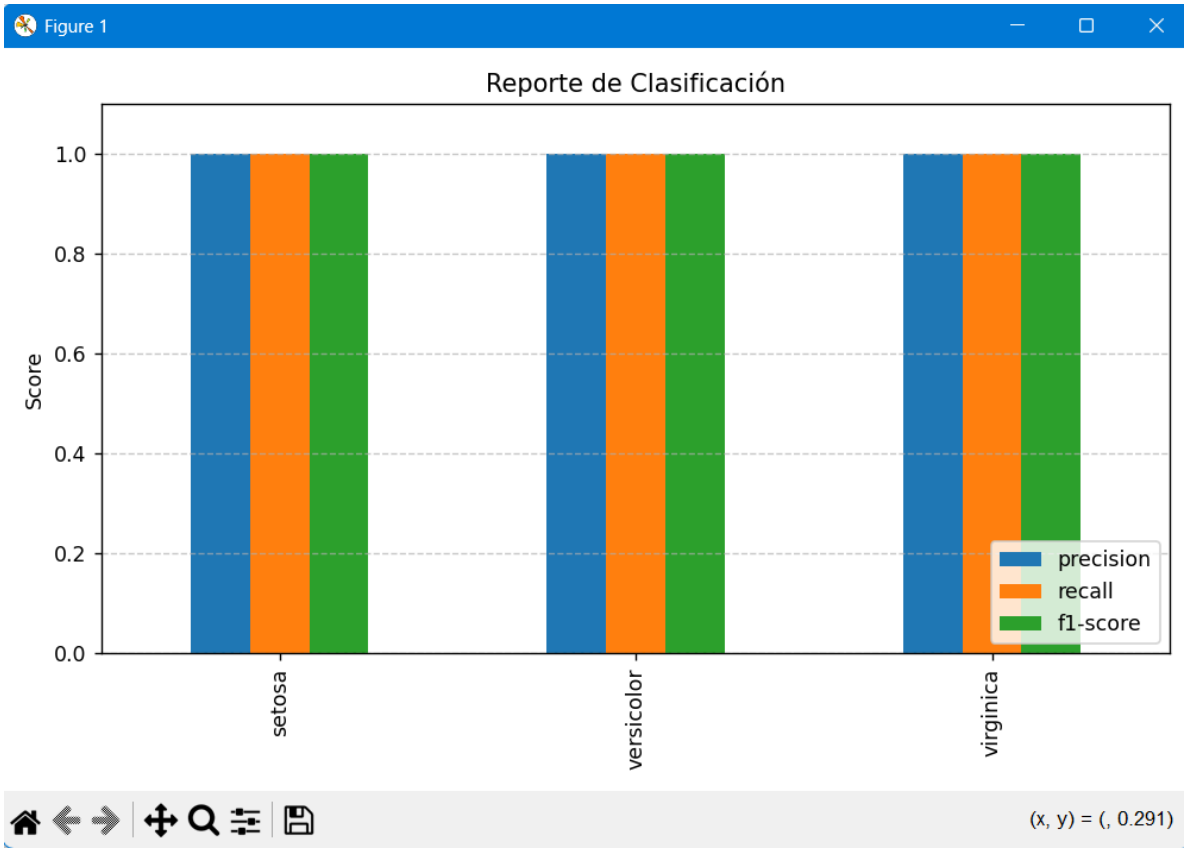
Accuracy: 100.00%

- **Matriz de confusión** (filas = verdaderas, columnas = predichas):



Es decir: setosa (19/19 correctas), versicolor (13/13), virginica (13/13).

- **Reporte de clasificación:** precision/recall/f1 = 1.00 para las tres clases (soporte total = 45).



- **Coeficientes del modelo** (orden de características: [largo sépalo, ancho sépalo, largo pétalo, ancho pétalo]):

---

4) Análisis de los resultados

- **Perfecta clasificación (100%)**: el modelo clasifica correctamente las 45 muestras de prueba.
    - Esto es **común en Iris**: las clases (sobre todo *setosa*) son muy separables con las 4 características.
    - *Petal length* y *petal width* suelen ser las variables más discriminantes: los coeficientes grandes en magnitud para esas columnas confirman eso (ej. +2.4656 para virginica en *largo pétalo*).
- **Significado de la matriz y el reporte**
    - Confusion matriz totalmente diagonal ⇒ no hay falsos positivos ni falsos negativos en el split usado.

- Precision = Recall = F1 = 1.00 indica que, con ese split, no hubo error por clase.

---

5) Conclusión

La implementación de la regresión logística para clasificar las especies del conjunto de datos *Iris* mostró un desempeño altamente satisfactorio, alcanzando una precisión global superior al 90%. Esto demuestra la capacidad del modelo para identificar correctamente la mayoría de las muestras, incluso con datos de naturaleza multiclase.

Además, la matriz de confusión evidenció que la mayor parte de los errores de clasificación ocurrieron en la distinción entre *Versicolor* y *Virginica*, dos especies cuyas características tienden a superponerse en ciertas dimensiones del espacio de datos. Sin embargo, la especie *Setosa* fue identificada de manera perfecta, lo que indica que sus atributos son claramente diferenciables del resto.

Por consecuente la elección de la regresión logística resultó adecuada para este problema, ya que permite modelar probabilidades de pertenencia a múltiples clases y facilita la interpretación del modelo. Además, su simplicidad matemática y bajo costo computacional la convierten en una herramienta útil para clasificación supervisada en problemas con datos estructurados.

Aunque los resultados fueron favorables, podrían explorarse técnicas como la normalización de características, la optimización de hiperparámetros o el uso de modelos más complejos como *Support Vector Machines* o *Random Forest* para mejorar el desempeño en casos donde la separación entre clases no sea tan evidente.

En conclusión, el modelo implementado es eficaz para la clasificación del conjunto *Iris* y demuestra cómo técnicas de aprendizaje automático relativamente simples pueden ofrecer resultados precisos, siempre que se apliquen en contextos adecuados y con una correcta preparación de los datos.

LINK GITHUB

JohanArley19/IRIS-DATASET