COMPARACIÓN DE LIBRERÍAS DE VISUALIZACIÓN EN PYTHON

2024

Juan Luis Serradilla Tormos

UNIVERSIDAD DE MURCIA

Facultad de Informática

INTRODUCCIÓN

Características para comparar:

1. Características
2. Gráficas
3. Conclusiones

1. CARACTERÍSTICAS

Matplotlib

- Gran comunidad
- Versatilidad
- Curva de aprendizaje inicial

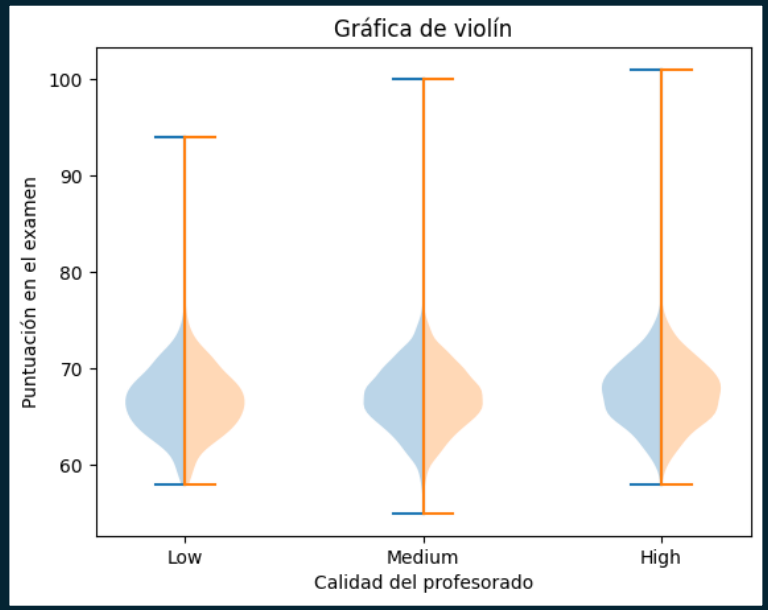
Seaborn

- Gran cantidad de funciones
- Gráficos estilizados con poca personalización
- Gran facilidad en estadística

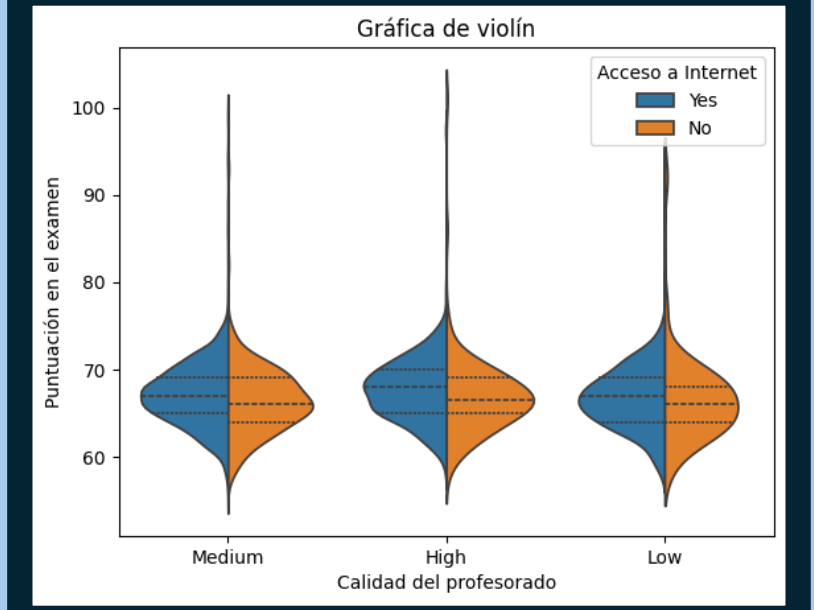
Plotly

- Gran cantidad de funciones
- Gráficos interactivos
- Generación sencilla para gráficos complejos

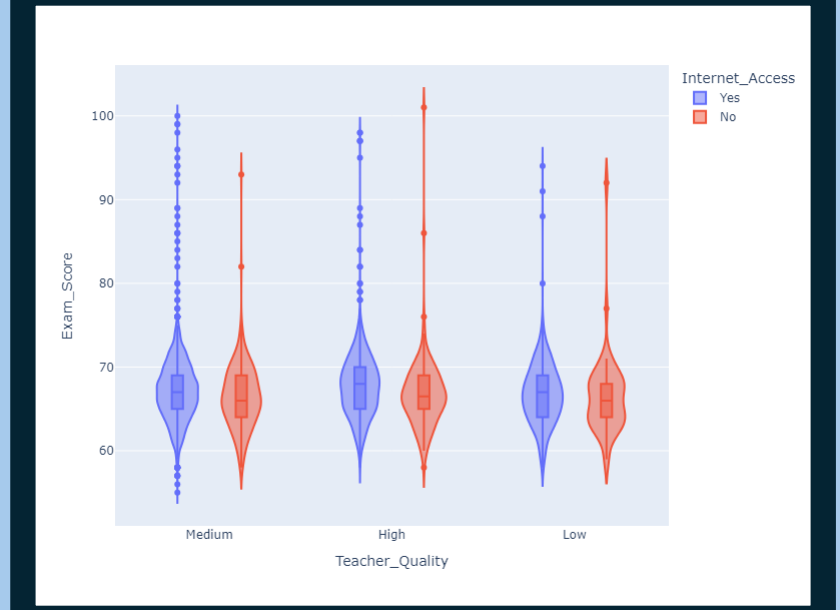
2.1. Gráfico de violín



Matplotlib



Seaborn



Plotly

2.1. Gráfico de violín

```python
plot_data = [
    data_students[data_students["Teacher_Quality"] == "Low"]["Exam_Score"].values,
    data_students[data_students["Teacher_Quality"] == "Medium"]["Exam_Score"].values,
    data_students[data_students["Teacher_Quality"] == "High"]["Exam_Score"].values
]
plt.violinplot(plot_data, side="low")
plt.violinplot(plot_data, side="high")
plt.xticks(np.arange(1, 4), labels=["Low", "Medium", "High"])
plt.title("Gráfica de violín")
plt.xlabel("Calidad del profesorado")
plt.ylabel("Puntuación en el examen")
plt.show()
```

Matplotlib

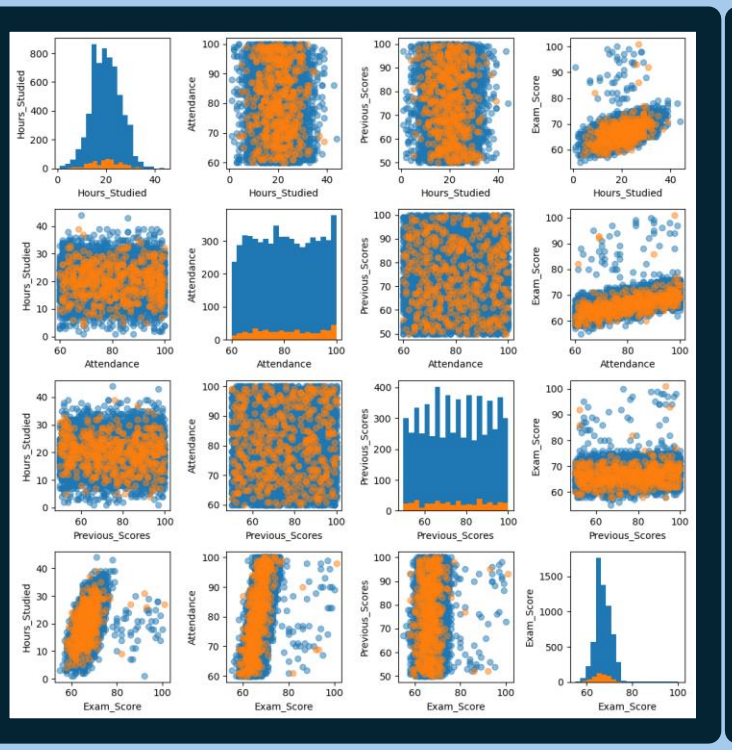
```python
sns.violinplot(data=data_students, x="Teacher_Quality", y="Exam_Score",
               hue="Internet_Access", split=True, inner="quart")
plt.title("Gráfica de violín")
plt.xlabel("Calidad del profesorado")
plt.ylabel("Puntuación en el examen")
legend = plt.gca().get_legend()
legend.set_title("Acceso a Internet")
plt.show()
```

Seaborn

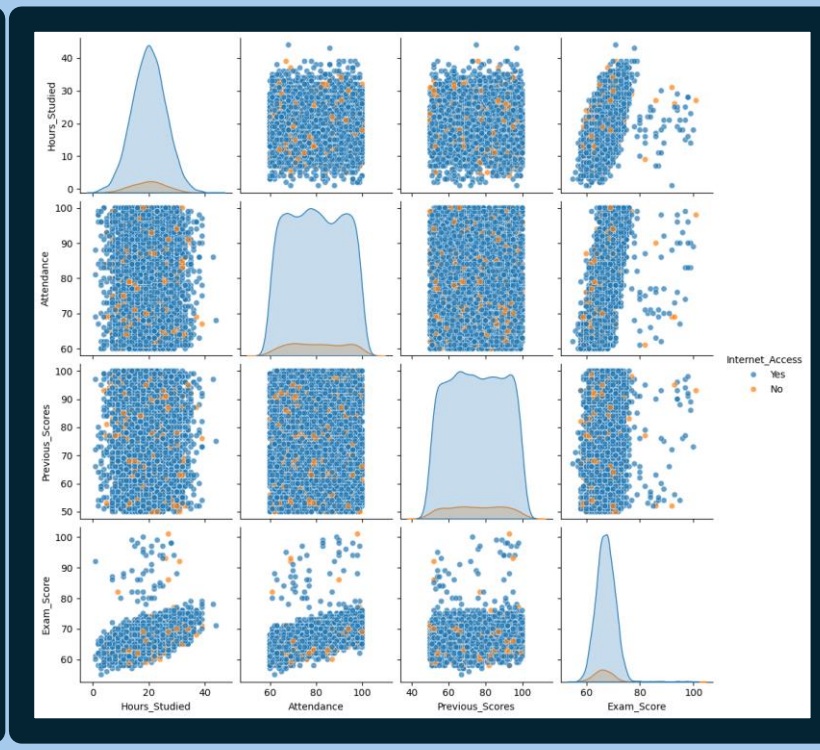
```python
fig = px.violin(data_students, x="Teacher_Quality", y="Exam_Score", color="Internet_Access", box=True)
fig.update_layout(height=600, width=775)
fig.show()
```

Plotly

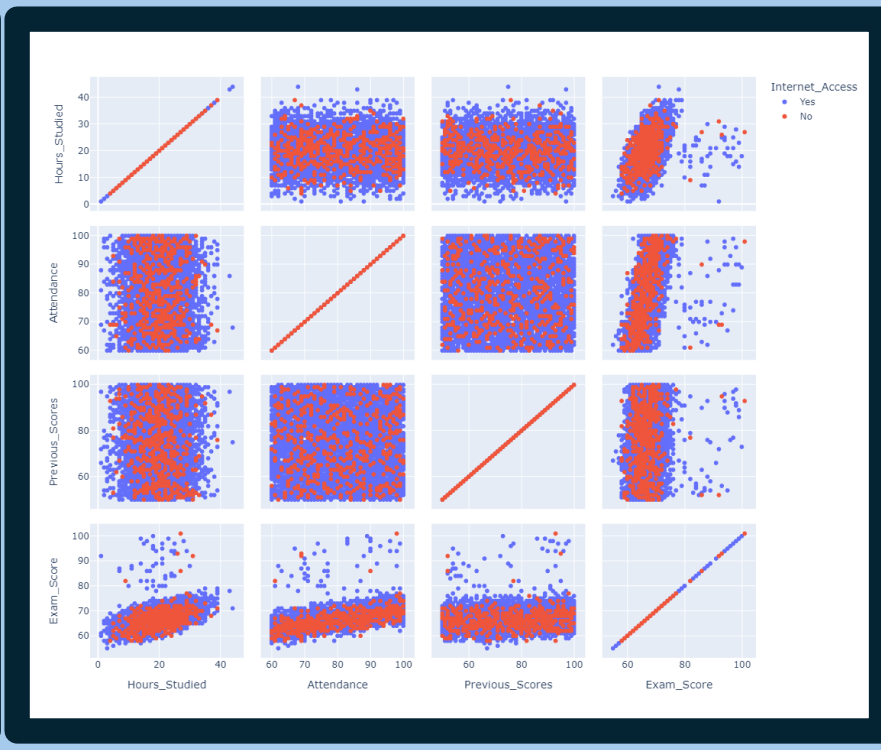
2.2. Gráfico de parejas



Matplotlib



Seaborn



Plotly

2.2. Gráfico de parejas

```python
numerical_columns = ["Hours_Studied", "Attendance", "Previous_Scores", "Exam_Score"]
categorical = "Internet_Access"
plot_dataset = data_students[numerical_columns]
fig, axes = plt.subplots(len(numerical_columns), len(numerical_columns), figsize=(10, 10))
for cat in ["Yes", "No"]:
    plot_dataset_filt = plot_dataset[data_students[categorical] == cat]
    for x,x_label in enumerate(numerical_columns):
        for y,y_label in enumerate(numerical_columns):
            ax = axes[x, y]
            if x == y:
                ax.hist(plot_dataset_filt[x_label], bins=20)
            else:
                ax.scatter(plot_dataset_filt[x_label], plot_dataset_filt[y_label], alpha=0.5)
            ax.set_xlabel(x_label)
            ax.set_ylabel(y_label)
plt.tight_layout()
plt.show()
```

→ **Matplotlib**

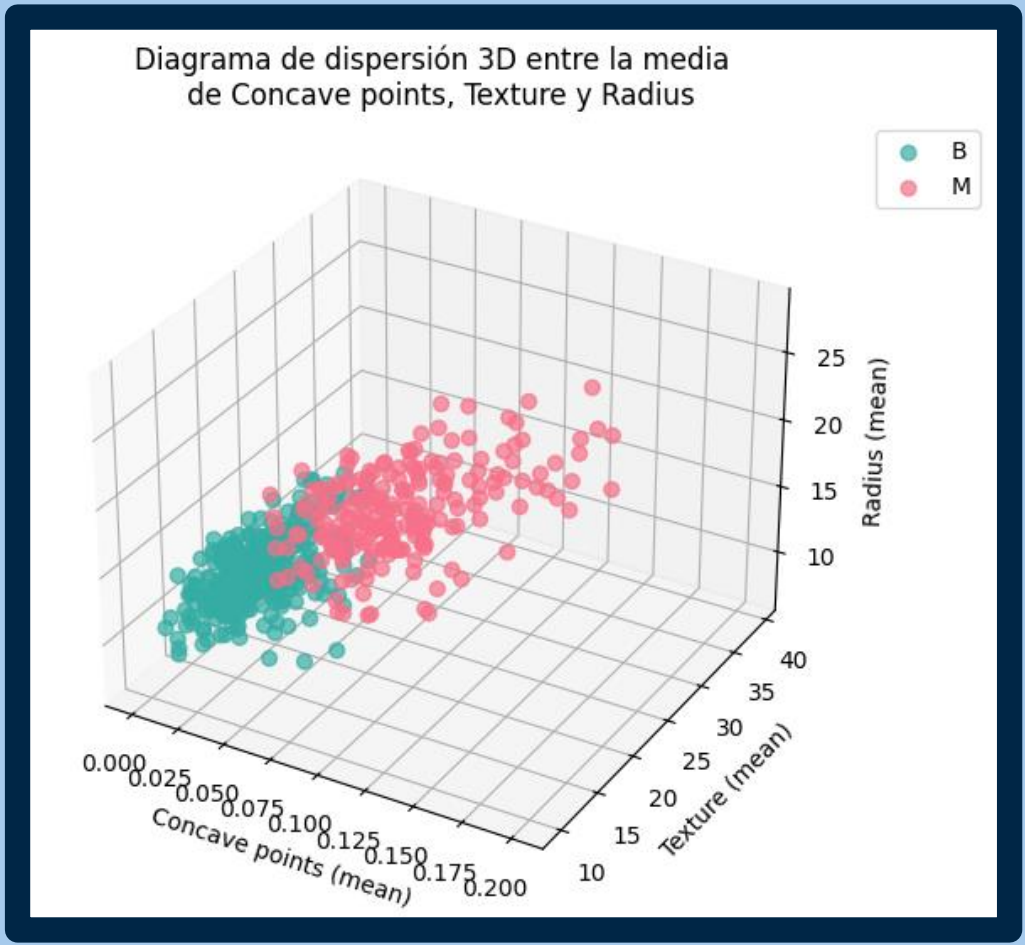
```python
numerical_columns = ["Hours_Studied", "Attendance", "Previous_Scores", "Exam_Score"]
categorical = "Internet_Access"
plot_dataset = data_students[numerical_columns]
numerical_columns.append(categorical)
plot_dataset = data_students[numerical_columns]
sns.pairplot(plot_dataset, hue=categorical, plot_kws={"alpha": 0.7})
plt.show()
```

→ **Seaborn**

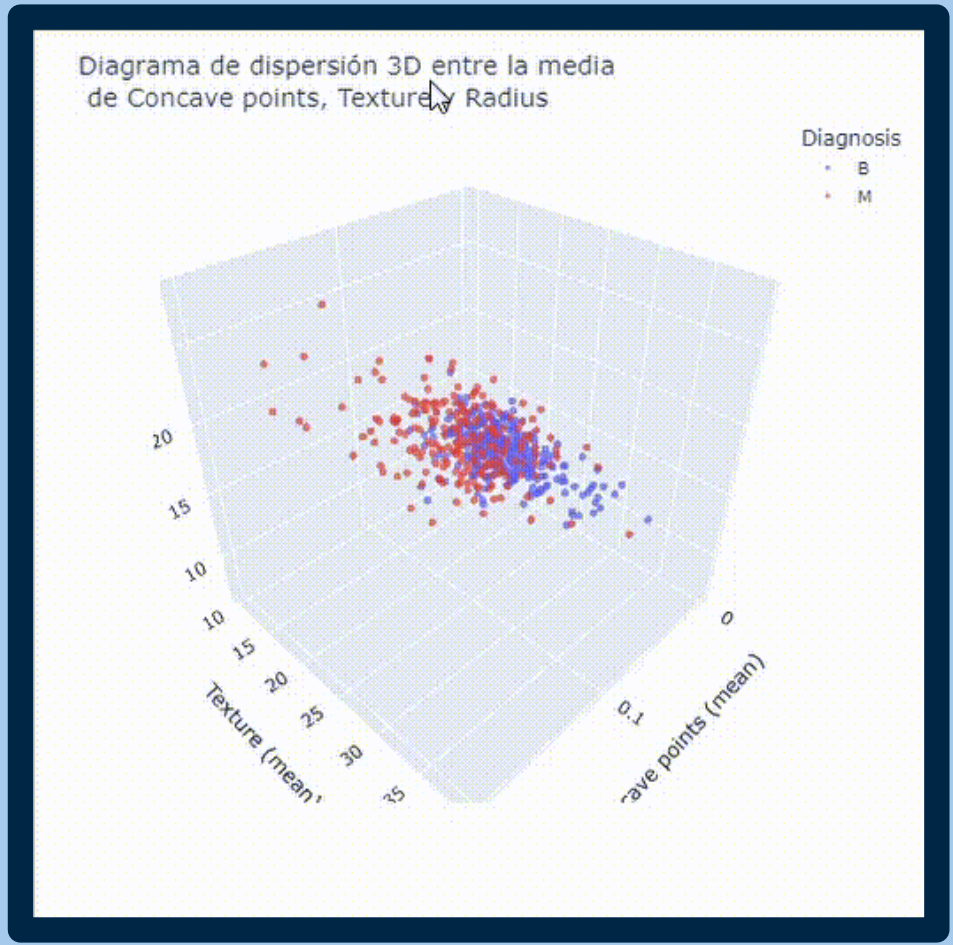
```python
numerical_columns = ["Hours_Studied", "Attendance", "Previous_Scores", "Exam_Score"]
categorical = "Internet_Access"
dataset_columns = numerical_columns.copy()
dataset_columns.append(categorical)
plot_dataset = data_students[dataset_columns]
fig = px.scatter_matrix(plot_dataset, dimensions=numerical_columns, color=categorical)
fig.update_layout(height=900, width=1100)
fig.show()
```

→ **Plotly**

2.3. Gráfico 3D



Matplotlib



Plotly

2.3. Gráfico 3D

```python
fig = plt.figure(figsize=(6, 6))
ax = fig.add_subplot(projection='3d')
cmap = ListedColormap(sns.color_palette("husl", 2).as_hex())

data_B = data_cancer[data_cancer["Diagnosis"] == "B"]
data_M = data_cancer[data_cancer["Diagnosis"] == "M"]
sc_B = ax.scatter(
    data_B["Concave points (mean)"],
    data_B["Texture (mean)"],
    data_B["Radius (mean)"],
    s=40, color=cmap(1), label="B", marker='o', alpha=0.7
)
sc_M = ax.scatter(
    data_M["Concave points (mean)"],
    data_M["Texture (mean)"],
    data_M["Radius (mean)"],
    s=40, color=cmap(0), label="M", marker='o', alpha=0.7
)

ax.set_xlabel('Concave points (mean)')
ax.set_ylabel('Texture (mean)')
ax.set_zlabel('Radius (mean)')
ax.set_title("Diagrama de dispersión 3D entre la media \n de Concave points, Texture y Radius")
plt.legend(bbox_to_anchor=(1.05, 1), loc=2)
plt.savefig("scatter_hue", bbox_inches='tight')
plt.show()
```

Matplotlib

```python
fig = px.scatter_3d(data_cancer,
                    x="Concave points (mean)", y="Texture (mean)", z="Radius (mean)", color="Diagnosis",
                    opacity=.7)
fig.update_traces(marker_size=3)
fig.update_layout(
    title_text="Diagrama de dispersión 3D entre la media <br> de Concave points, Texture y Radius",
    height=600, width=600
)
fig.show()
```

Plotly

3. CONCLUSIONES

Matplotlib

Se pueden crear gráficos que no están por defecto

Poco intuitiva para gráficos complejos y curva de aprendizaje pronunciada al final.

Seaborn

Muy sencilla de usar para gráficos estadísticos con resultados atractivos.

Gráficas resultantes muy estilizadas pero con pocas opciones de personalización.

Plotly

Gráficos interactivos muy útiles. Poca cantidad de código en muchas gráficas.

Tiempo de ejecución lento al renderizar gráficos interactivos.


FACULTAD DE INFORMÁTICA
UMU