slant-up

May 20, 2024

Universidad Politécnica Salesiana

Graficas del Datasaurus

Estudiante: Carlos Saico

Carga del Dataset

[1]:
```
import pandas as pd
import seaborn as sns

print("Modulos Importados")
```

```
Modulos Importados
```

[2]:
```
data = pd.read_csv('DatasaurusDozen.tsv', sep='\t')
data.head()
```

[2]:

	dataset	x	y
0	dino	55.3846	97.1795
1	dino	51.5385	96.0256
2	dino	46.1538	94.4872
3	dino	42.8205	91.4103
4	dino	40.7692	88.3333

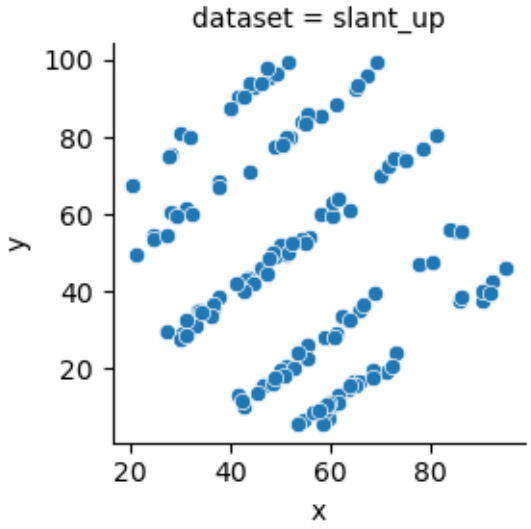
[3]:
```
data['dataset'].unique()
```

[3]:
```
array(['dino', 'away', 'h_lines', 'v_lines', 'x_shape', 'star',
       'high_lines', 'dots', 'circle', 'bullseye', 'slant_up',
       'slant_down', 'wide_lines'], dtype=object)
```

Diagrama de dispoerción Slant_up

[4]:
```
import seaborn as sns
import matplotlib.pyplot as plt
# Filtrar el conjunto de datos solo para la clase 'slant_up'
data_slant_up = data[data['dataset'] == 'slant_up']
# Crear una cuadrícula de gráficos
grid_scatterplot = sns.FacetGrid(data_slant_up, col="dataset", hue="dataset")
grid_scatterplot.map_dataframe(sns.scatterplot, x="x", y="y")
```

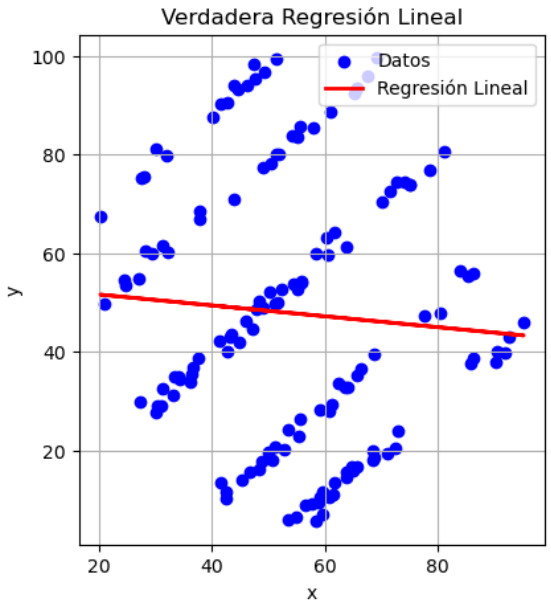
```
# Mostrar los gráficos
plt.show()
```



Gráfica función lineal

[5]:
```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

# Cargar los datos
data = pd.read_csv('DatasaurusDozen.tsv', sep='\t')

# Filtrar el conjunto de datos solo para la clase 'slant_up'
data_slant_up = data[data['dataset'] == 'slant_up']

# Separar las características (X) y la variable objetivo (y)
X = data_slant_up[['x']]
y = data_slant_up['y']

# Entrenar el modelo de regresión lineal
model = LinearRegression()
model.fit(X, y)

# Calcular las predicciones
y_pred = model.predict(X)
```

```python
# Calcular el R cuadrado
accuracy = r2_score(y, y_pred)

# Visualizar los datos y la regresión lineal
plt.figure(figsize=(10, 5))

# Subplot para la gráfica de regresión
plt.subplot(1, 2, 1)
plt.scatter(X, y, color='blue', label='Datos')
plt.plot(X, y_pred, color='red', linewidth=2, label='Regresión Lineal')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Verdadera Regresión Lineal')
plt.legend()
plt.grid(True)

# Subplot para la imagen "Real.jpeg"
plt.subplot(1, 2, 2)
img = mpimg.imread(r'C:\Users\carlo\Desktop\Mineria\Python\Datasaurus\Real.
 ↪jpeg')
plt.imshow(img)
plt.axis('off')  # Desactivar los ejes para la imagen
plt.title('Estimación de la Regresión Lienal')

plt.show()
```



[6]:
```
# Calcular las predicciones
y_pred = model.predict(X)

# Calcular el R cuadrado
accuracy = r2_score(y, y_pred)
print("Accuracy función lineal:", accuracy)
```

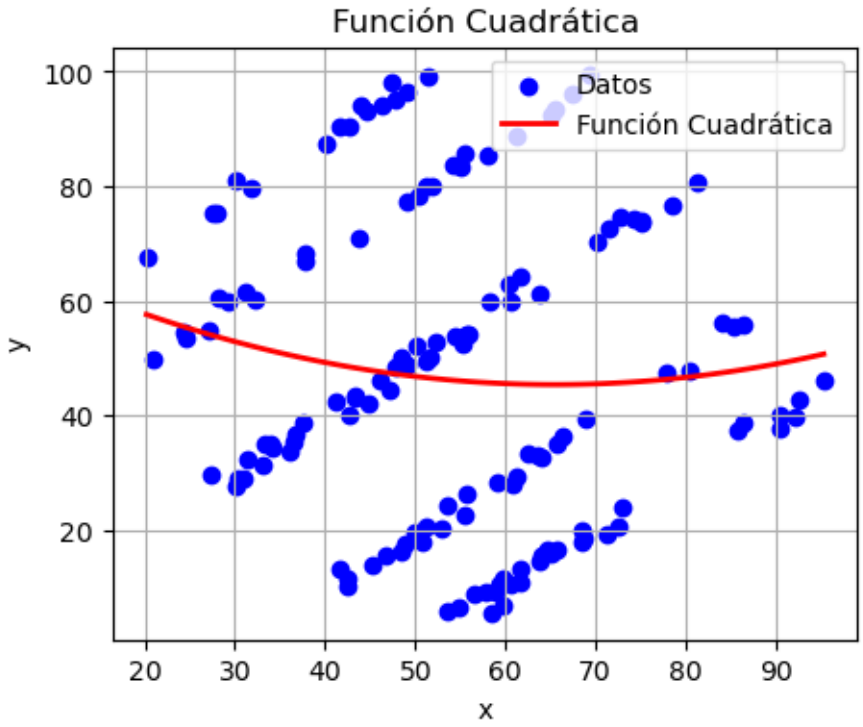
```
Accuracy función lineal: 0.004707223209015843
```

Gráfica función cuadrática

[7]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

# Cargar los datos
data = pd.read_csv('DatasaurusDozen.tsv', sep='\t')

# Filtrar el conjunto de datos solo para la clase 'slant_up'
data_slant_up = data[data['dataset'] == 'slant_up']

# Separar las características (X) y la variable objetivo (y)
X = data_slant_up[['x']]
y = data_slant_up['y']

# Ajustar un modelo de regresión cuadrática
poly = PolynomialFeatures(degree=2)
X_poly = poly.fit_transform(X)
model = LinearRegression()
model.fit(X_poly, y)

# Calcular las predicciones
y_pred = model.predict(X_poly)

# Calcular el R cuadrado
accuracyrc = r2_score(y, y_pred)

# Ordenar los valores de X para trazar la curva de regresión
X_plot = np.linspace(X.min(), X.max(), 100)
X_plot_poly = poly.transform(X_plot.reshape(-1, 1))
y_plot = model.predict(X_plot_poly)

# Visualizar los datos y la regresión cuadrática
plt.figure(figsize=(5, 4))
```

```
plt.scatter(X, y, color='blue', label='Datos')
plt.plot(X_plot, y_plot, color='red', linewidth=2, label='Función Cuadrática')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Función Cuadrática')
plt.legend()
plt.grid(True)
plt.show()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning: X
does not have valid feature names, but PolynomialFeatures was fitted with
feature names
  warnings.warn(
```



[8]:
```
print("Accuracy con la función cuadrática:", accuracyrc)
```

```
Accuracy con la función cuadrática: 0.010897814779311399
```

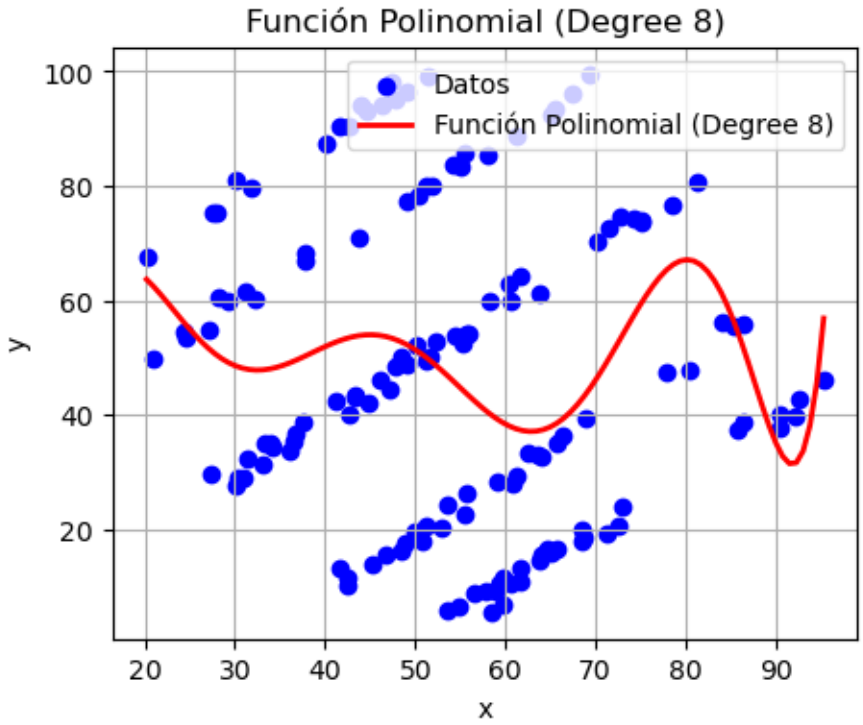
Gráfica función polinomial

[9]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

```
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

# Cargar los datos
data = pd.read_csv('DatasaurusDozen.tsv', sep='\t')

# Filtrar el conjunto de datos solo para la clase 'slant_up'
data_slant_up = data[data['dataset'] == 'slant_up']

# Separar las características (X) y la variable objetivo (y)
X = data_slant_up[['x']]
y = data_slant_up['y']

# Definir el grado del polinomio
grado = 8
# Ajustar un modelo de regresión polinomial
poly = PolynomialFeatures(degree=grado)
X_poly = poly.fit_transform(X)
model = LinearRegression()
model.fit(X_poly, y)

# Calcular las predicciones
y_pred = model.predict(X_poly)

# Calcular el R cuadrado
accuracyfp = r2_score(y, y_pred)

# Ordenar los valores de X para trazar la curva de regresión
X_plot = np.linspace(X.min(), X.max(), 100)
X_plot_poly = poly.transform(X_plot.reshape(-1, 1))
y_plot = model.predict(X_plot_poly)

# Visualizar los datos y la regresión polinomial
plt.figure(figsize=(5, 4))
plt.scatter(X, y, color='blue', label='Datos')
plt.plot(X_plot, y_plot, color='red', linewidth=2, label=f'Función Polinomial␣
 ↪(Degree {grado})')
plt.xlabel('x')
plt.ylabel('y')
plt.title(f'Función Polinomial (Degree {grado})')
plt.legend()
plt.grid(True)

plt.show()
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning: X
```

```
does not have valid feature names, but PolynomialFeatures was fitted with
feature names
  warnings.warn(
```



[10]:
```
print("Accuracy de la regresión cuadrática:", accuracyfp)
```

```
Accuracy de la regresión cuadrática: 0.07695269788252657
```