Atharva Prashant Pawar (9427) - Comps - A [Batch - D]

ML - Exp - 1

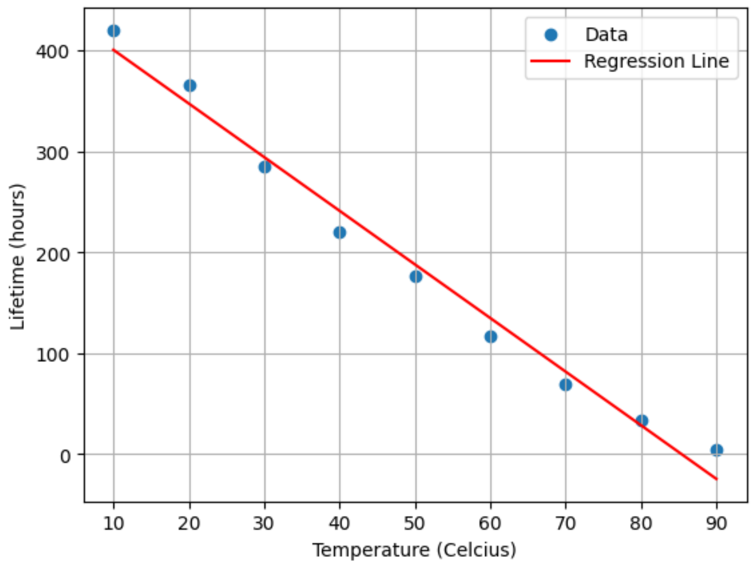
1. Linear Regression life time model:

A company manufactures and electronic device to be used in a very wide temperature in the company knows that increased temperature shortens the lifetime of the device and a study is therefore performed in which the lifetime is determined as a function of temperature the following data is found: dataset: temperature in Celcius(t) :10, 20, 30, 40, 50, 60, 70, 80, 90 Life time in hours(y) : 420, 365, 285, 220, 176, 117, 69, 34, 5

calculate the 95% confidence interval for the slope in the usual linear regression model, which expresses the life time as a linear function of the temperature.

```
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt

temperature = np.array([10, 20, 30, 40, 50, 60, 70, 80, 90])
lifetime = np.array([420, 365, 285, 220, 176, 117, 69, 34, 5])

slope, intercept, r_value, p_value, std_err = stats.linregress(temperature, lifetime)
n = len(temperature) # Degrees of freedom
df = n - 2

t_critical = stats.t.ppf(0.975, df) # Critical t-value for a 95% confidence interval
SE_slope = std_err # Standard error of the slope estimate
margin_of_error = t_critical * SE_slope # The margin of error

# Confidence interval for the slope
confidence_interval_x, confidence_interval_y  = (slope - margin_of_error, slope + margin_of_error)

def roundfun(value, roundby=3): # number round-off function
    return round(float(value), roundby)

print("Slope:", roundfun(slope))
print("95% Confidence Interval for the Slope:", roundfun(confidence_interval_x), roundfun(confidence_interval_y))

plt.scatter(temperature, lifetime, label="Data")
plt.plot(temperature, slope * temperature + intercept, color='red', label='Regression Line')
plt.xlabel("Temperature (Celcius)")
plt.ylabel("Lifetime (hours)")
plt.legend()
plt.grid(True)
plt.show()
```

```
Slope: -5.313
95% Confidence Interval for the Slope: -5.918 -4.709
```

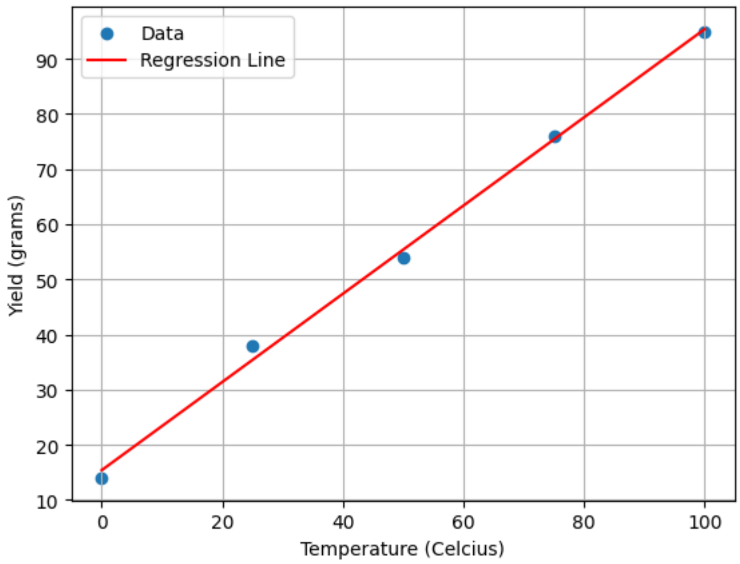


2. Yield of chemical process:

The yield Y of a chemical process is a random variable whose value is considered to be a linear function of the temperature the following data of corresponding values of X and Y is found: temperature in Celcius(x) : 0,25,50,75,100 Yield in grams(y) : 14, 38, 54, 76, 95

```
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt

temperature = np.array([0, 25, 50, 75, 100])
yield_grams = np.array([14, 38, 54, 76, 95])

slope, intercept, r_value, p_value, std_err = stats.linregress(temperature, yield_grams)
equation_of_line = f"Yield = {slope:.2f} * Temperature + {intercept:.2f}" # Equation of the line

def roundfun(value, roundby=3): # number round-off function
    return round(float(value), roundby)

print("Linear Regression Results:")
print("Slope:", roundfun(slope))
print("Intercept:", roundfun(intercept))
print("R-squared:", roundfun(r_value**2))
print("Equation of the line:", equation_of_line)

# Create points to plot the regression line
x_values = np.linspace(min(temperature), max(temperature), 100)
y_values = slope * x_values + intercept

# Plotting the scatter plot and regression line
plt.scatter(temperature, yield_grams, label="Data")
plt.plot(x_values, y_values, color='red', label='Regression Line')
plt.xlabel("Temperature (Celcius)")
plt.ylabel("Yield (grams)")
plt.legend()
plt.grid(True)
plt.show()
```

```
Linear Regression Results:
Slope: 0.8
Intercept: 15.4
R-squared: 0.997
Equation of the line: Yield = 0.80 * Temperature + 15.40
```



3. The value of y and their corresponding value of y shown in the table below

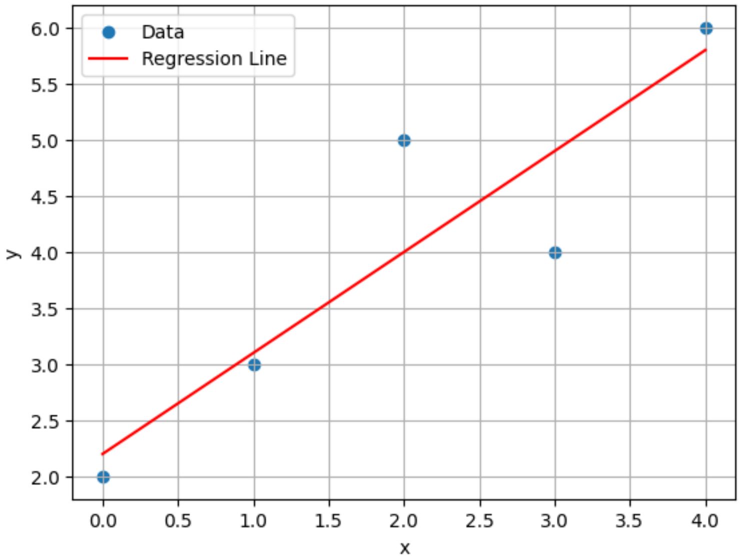
x: 0,1,2,3,4 y: 2,3,5,4,6

a. Find the least square regression line y=ax+b. b. Estimate the value of y when x = 10.

```
import numpy as np
import matplotlib.pyplot as plt

x = np.array([0, 1, 2, 3, 4])
y = np.array([2, 3, 5, 4, 6])

# Cal: the coefficients a and b for the least sq reg line
n = len(x)
sum_x, sum_y = np.sum(x), np.sum(y)
```

```
sum_x_squared, sum_xy = np.sum(x**2), np.sum(x * y)

# Cal: least sq reg line coefficients (a and b)
a = (n * sum_xy - sum_x * sum_y) / (n * sum_x_squared - sum_x**2)
b = (sum_y - a * sum_x) / n

equation_of_line = f"y = {a:.2f}x + {b:.2f}"
print("Least Squares Regression Line:", equation_of_line)

# (testing...)
x_new = 10
y_estimate = a * x_new + b
print(f"Estimated value of y when x = 10: {y_estimate:.2f}")


x_values = np.linspace(min(x), max(x), 100) # Create points to plot the reg line
y_values = a * x_values + b

plt.scatter(x, y, label="Data")
plt.plot(x_values, y_values, color='red', label='Regression Line')
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.grid(True)
plt.show()
```

```
Least Squares Regression Line: y = 0.90x + 2.20
Estimated value of y when x = 10: 11.20
```



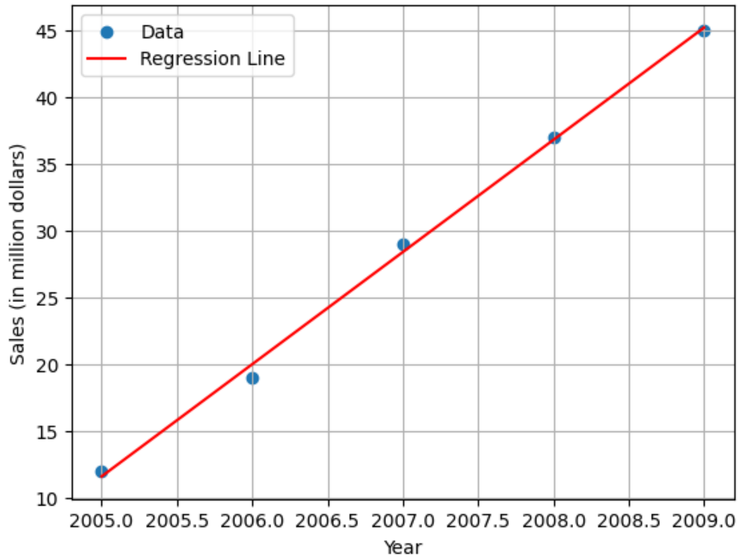
4. The sales of company in million dollar for each year are shown in the table below:

x(year) : 2005, 2006, 2007, 2008, 2009 y(sales): 12, 19, 29, 37, 45 a. Find the least square regression line Y = a x + b b. Use the least square regression line as a model to estimate the sales of company in 2012

```
import numpy as np
import matplotlib.pyplot as plt

years = np.array([2005, 2006, 2007, 2008, 2009])
sales = np.array([12, 19, 29, 37, 45])

# Calculate the coefficients a and b for the least squares regression line
n = len(years)
sum_years, sum_sales = np.sum(years), np.sum(sales)
sum_years_squared = np.sum(years**2)
sum_years_sales = np.sum(years * sales)

# Calculate least squares regression line coefficients (a and b)
a = (n * sum_years_sales - sum_years * sum_sales) / (n * sum_years_squared - sum_years**2)
b = (sum_sales - a * sum_years) / n

# Equation of the least squares regression line
equation_of_line = f"Y = {a:.2f}x + {b:.2f}"
print("Least Squares Regression Line:", equation_of_line)

# Estimate the sales of the company in 2012 using the regression line
x_new = 2012
```

```
sales_estimate = a * x_new + b
print(f"Estimated sales in 2012: {sales_estimate:.2f} million dollars")

# Plotting the scatter plot and regression line
plt.scatter(years, sales, label="Data")
plt.plot(years, a * years + b, color='red', label='Regression Line')
plt.xlabel("Year")
plt.ylabel("Sales (in million dollars)")
plt.legend()
plt.grid(True)
plt.show()
```

```
Least Squares Regression Line: Y = 8.40x + -16830.40
Estimated sales in 2012: 70.40 million dollars
```



5. You have to study the relationship between the monthly E-Commerce sales and the online advertising cost you have the survey result for 7 online stores for the last year.

online_store: 1,2,3,4,5,6,7; monthly_ecommerce_sales(in_1000S): 368, 340, 665, 954, 331, 556, 376; online_advertising_dollars(1000S): 1.7, 1.5, 2.8, 5, 1.3, 2.2, 1.3;

a. Find the least square regression line y = a x + b b. Use the least square regression line as a model to estimate the sales of the company when 8 dollar spent for advertisement

```
import numpy as np
import matplotlib.pyplot as plt

online_store = np.array([1, 2, 3, 4, 5, 6, 7])
monthly_ecommerce_sales = np.array([368, 340, 665, 954, 331, 556, 376])
online_advertising_dollars = np.array([1.7, 1.5, 2.8, 5, 1.3, 2.2, 1.3])

# Calculate the coefficients a and b for the least squares regression line
n = len(online_store)
sum_store = np.sum(online_store)
sum_sales = np.sum(monthly_ecommerce_sales)
sum_store_squared = np.sum(online_store**2)
sum_store_sales = np.sum(online_store * monthly_ecommerce_sales)

# Calculate least squares regression line coefficients (a and b)
a = (n * sum_store_sales - sum_store * sum_sales) / (n * sum_store_squared - sum_store**2)
b = (sum_sales - a * sum_store) / n

# Equation of the least squares regression line
equation_of_line = f"y = {a:.2f}x + {b:.2f}"
print("Least Squares Regression Line:", equation_of_line)

# Estimate the sales when 8 dollars are spent on advertisement using the regression line
dollars_spent = 8
sales_estimate = a * dollars_spent + b
print(f"Estimated sales when $8 are spent on advertisement: {sales_estimate:.2f} (in 1000s)")

# Create points to plot the regression line
x_values = np.linspace(min(online_store), max(online_store), 100)
y_values = a * x_values + b

# Plotting the scatter plot and regression line
```

```
# Plotting the scatter plot and regression line
plt.scatter(online_store, monthly_ecommerce_sales, label="Data")
plt.plot(x_values, y_values, color='red', label='Regression Line')
plt.xlabel("Online Store")
plt.ylabel("Monthly E-commerce Sales (in 1000s)")
plt.legend()
plt.grid(True)
plt.show()
```

```
Least Squares Regression Line: y = 4.36x + 495.43
Estimated sales when $8 are spent on advertisement: 530.29 (in 1000s)
```