EDS PROJECT
SALARY PREDICTOR

GUIDED BY SHUBHANGI KALE

OUR TEAM

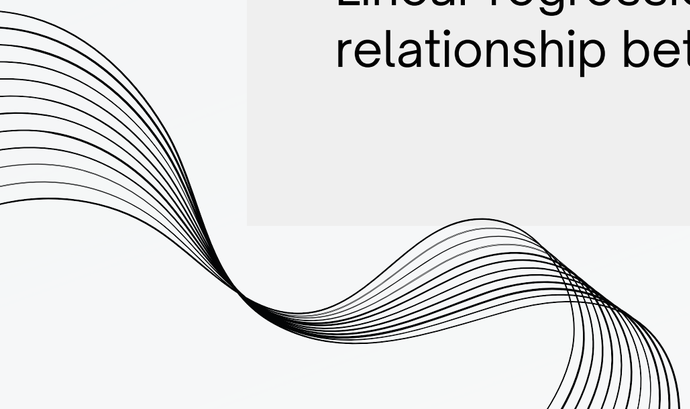
Monali Babde
205

Swarup Divekar
217

Pranav Falke
218

INTRODUCTION

- Data science is a multidisciplinary field that involves extracting insights and knowledge from data to solve complex problems and make data-driven decisions
- NumPy is a fundamental library for scientific computing in Python. It provides powerful N-dimensional array objects
- Pandas is a library that offers high-performance data manipulation and analysis tools
- It provides data structures like DataFrames and Series, which allow you to work with structured data easily
- Linear regression is a fundamental statistical modeling technique used to establish a linear relationship between a dependent variablea and one or more independent variables.



MOTIVATION

- The sink of Salary dataset always makes curiosity in mostly Students .
- How it to be manage huge dataset which includes enough no .of parameters to understand the python & data science can be understand with that dataset makes easy to understand
- Salary makes it includes numerical as well as character data types

DETAILS OF DATASET

Name: Salary dataset

Number of Features:

- **Age**: The age of individuals in the dataset
- **Education Level:** The highest level of education completed by individuals
- **Years of Experience:** The number of years of professional experience of individuals
- **Job Title:** The specific job title or position held by individuals
- **Gender :** Gender of individual

Number of records: Rows: 6704
Columns: 6

DATA MANIPULATION

```
import pandas as pd
df =pd.read_csv("/content/sample_data/salary.csv")
 #print all records of dataset
print(df)

# save DataFrame to a CSV file
df1.to_csv("Salary.csv",index=True)
# print all record through salary_data
salary_data=
pd.read_csv('/content/sample_data/salary.csv')
salary_data

# compute basic summary statistics of salary_data
 salary_data.describe()
```

OUTPUT:

	Age	Years of Experience	Salary
count	6702.000000	6701.000000	6699.000000
mean	33.620859	8.094687	115326.964771
std	7.614633	6.059003	52786.183911
min	21.000000	0.000000	350.000000
25%	28.000000	3.000000	70000.000000
50%	32.000000	7.000000	115000.000000
75%	38.000000	12.000000	160000.000000
max	62.000000	34.000000	250000.000000

```
# selecting salary >100000
print(df.loc[df['Salary']>100000])
```

OUTPUT:

```
       Age  Gender    Education Level                         Job Title  \
2     45.0    Male                PhD                    Senior Manager
4     52.0    Male           Master's                          Director
6     42.0  Female           Master's                   Product Manager
9     38.0    Male                PhD                   Senior Scientist
11    48.0  Female         Bachelor's                        HR Manager
...    ...     ...                ...                               ...
6690  42.0    Male  Bachelor's Degree                 Financial Manager
6693  43.0  Female    Master's Degree                     Sales Manager
6697  51.0  Female    Master's Degree  Senior Product Marketing Manager
6699  49.0  Female                PhD             Director of Marketing
6702  46.0    Male    Master's Degree                 Marketing Manager

      Years of Experience    Salary
2                    15.0  150000.0
4                    20.0  200000.0
6                    12.0  120000.0
9                    10.0  110000.0
11                   18.0  140000.0
...                   ...       ...
6690                 13.0  130000.0
6693                 14.0  140000.0
6697                 19.0  190000.0
6699                 20.0  200000.0
6702                 14.0  140000.0

[3772 rows x 6 columns]
```

```
# Apply multiple aggregation functions to Salary
df.groupby('Salary').agg(['mean', 'max', 'min'])
```

OUTPUT:

	Age			Years of Experience		
	mean	max	min	mean	max	min
Salary						
350.0	29.000000	29.0	29.0	1.500000	1.5	1.5
500.0	31.000000	31.0	31.0	4.000000	4.0	4.0
550.0	25.000000	25.0	25.0	1.000000	1.0	1.0
579.0	23.000000	23.0	23.0	1.000000	1.0	1.0
25000.0	24.781955	33.0	21.0	0.315789	1.0	0.0
...	...	...	...	...	...	...
220000.0	48.000000	49.0	44.0	21.090909	22.0	16.0
225000.0	50.000000	50.0	50.0	23.000000	23.0	23.0
228000.0	49.000000	49.0	49.0	23.000000	23.0	23.0
240000.0	51.000000	51.0	51.0	24.000000	24.0	24.0
250000.0	49.000000	52.0	45.0	23.333333	25.0	21.0

444 rows × 6 columns

# group by column and compute the given value from salary column
print(df.groupby('Salary').get_group(250000))

OUTPUT:

```
   Age Gender    Education Level                Job Title  \
30    50.0   Male        Bachelor's                      CEO
83    52.0   Male             PhD  Chief Technology Officer
5001  45.0   Male  Bachelor's Degree          Financial Manager

      Years of Experience    Salary
30                   25.0  250000.0
83                   24.0  250000.0
5001                 21.0  250000.0
```

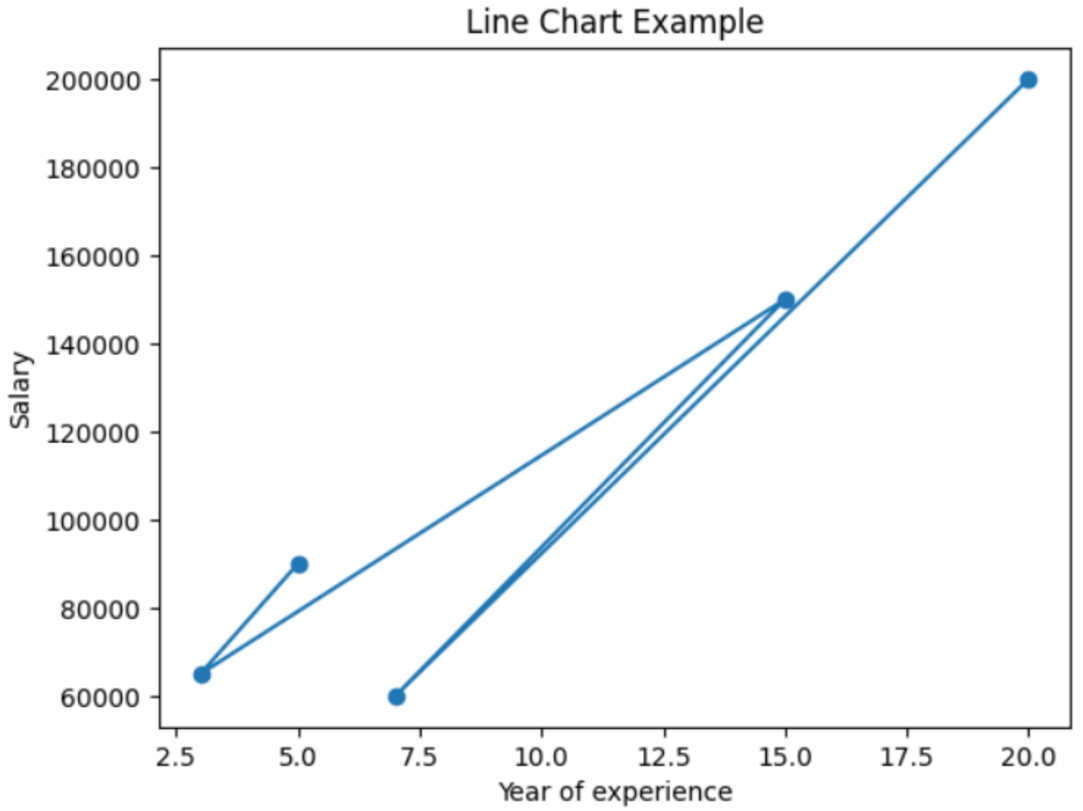
# compute the correlation between columns
print(df.corr())

OUTPUT:

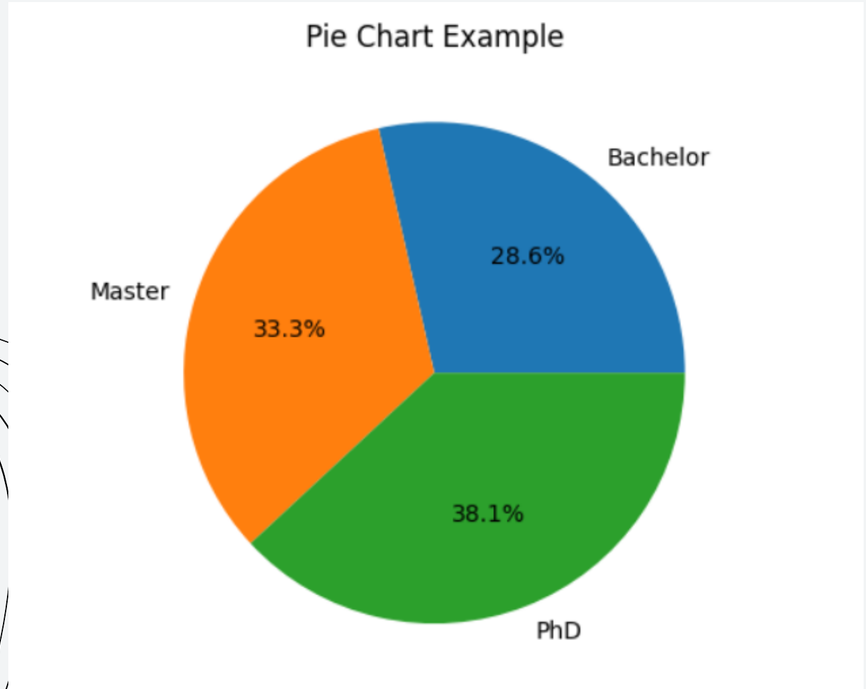
```
                         Age  Years of Experience    Salary
Age                 1.000000             0.937655  0.728053
Years of Experience 0.937655             1.000000  0.808969
Salary              0.728053             0.808969  1.000000
```

DATA VISUALIZATION

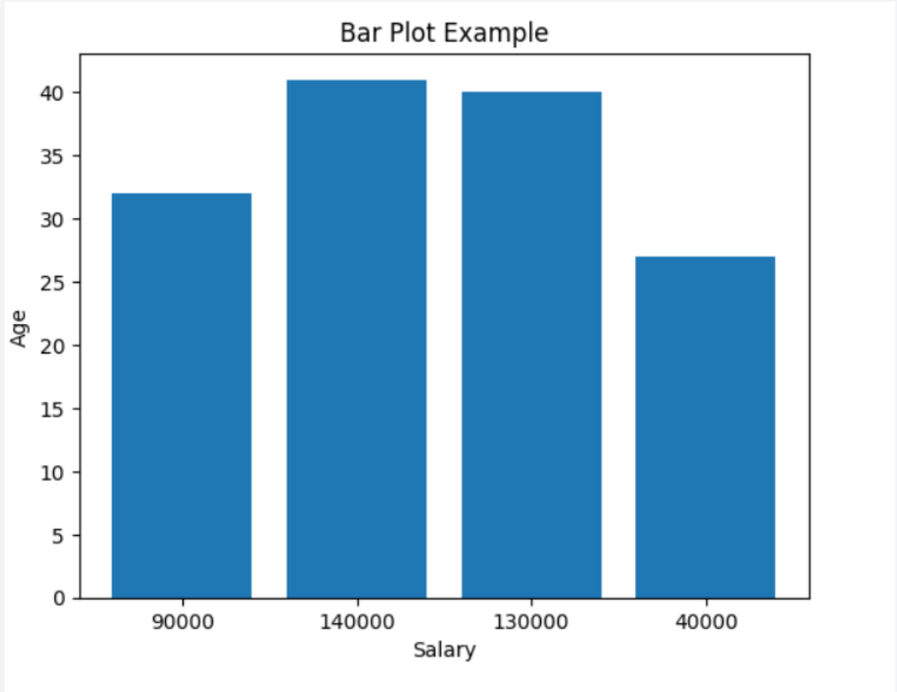
```
import matplotlib.pyplot as plt
import pandas as pd
df = pd.read_csv("/content/salary.csv")
df.head()

#Sample data
x= [5,3,15,7,20] # x-axis values (Year of experience)
y=[90000,65000,150000,60000,200000] # y-axis values (Salary)

# Create a line chart
plt.plot(x, y, marker='o')

# Customize the chart
plt.title("Line Chart Example")
plt.xlabel("Year of experience")
plt.ylabel("Salary")

# Display the chart
plt.show()
```



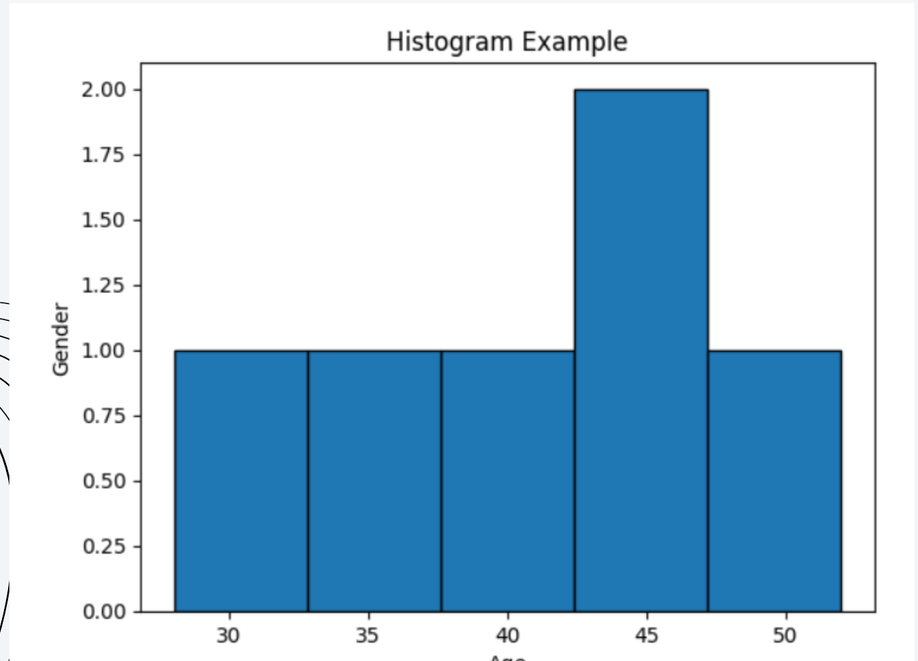
```
import matplotlib.pyplot as plt

# Example data
Education = ['Bachelor','Master','PhD']
Age = [30,35,40]

# Plotting the pie chart
plt.pie(Age,labels=Education, autopct='%1.1f%%')

# Adding a title
plt.title('Pie Chart Example')

# Displaying the pie chart
plt.show()
```



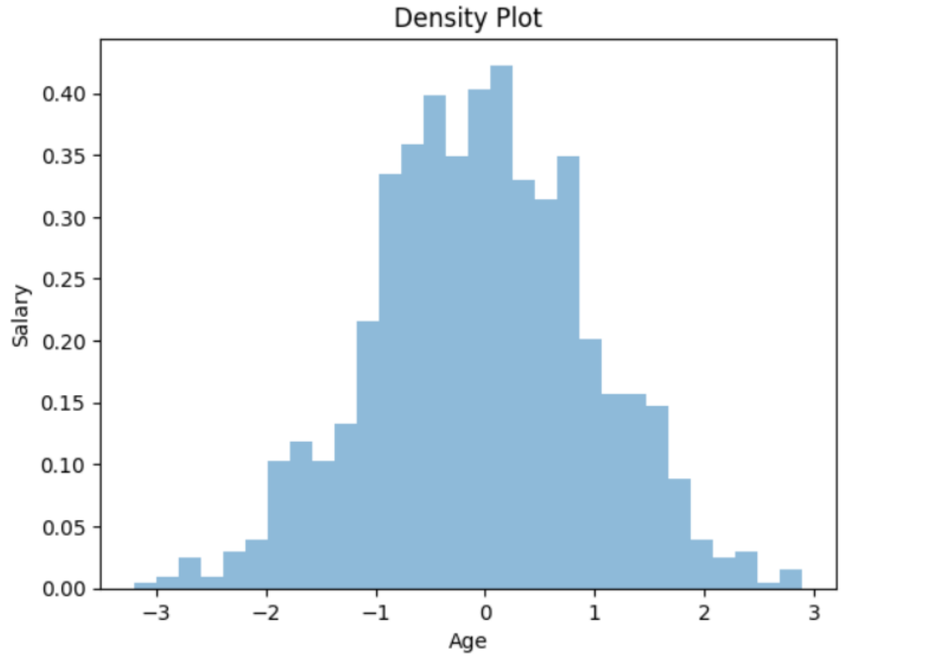

```
import matplotlib.pyplot as plt

# Sample data
Salary = ['90000', '140000', '130000','40000']
Age = [32, 41, 40, 27]

# Create a bar plot
plt.bar(Salary, Age) # Customize the plot
plt.title("Bar Plot Example")

plt.xlabel("Salary")
plt.ylabel("Age")

# Display the plot
plt.show()
```



```
import matplotlib.pyplot as plt

# Example data

data = [45,28,45,36,52,40]
#Plotting the histogram
plt.hist(data, bins=5, edgecolor="black")

# Adding labels and title
plt.xlabel('Age')
plt.ylabel('Gender')
plt.title('Histogram Example')

# Displaying the histogram
plt.show()
```




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

# Generate some random data
data = np.random.randn(1000)

# Create a density plot

plt.hist(data, density=True, bins=30, alpha=0.5)

# Add labels and title
plt.xlabel('Age')
plt.ylabel('Salary')
plt.title('Density Plot')

# Show the plot
plt.show()
```



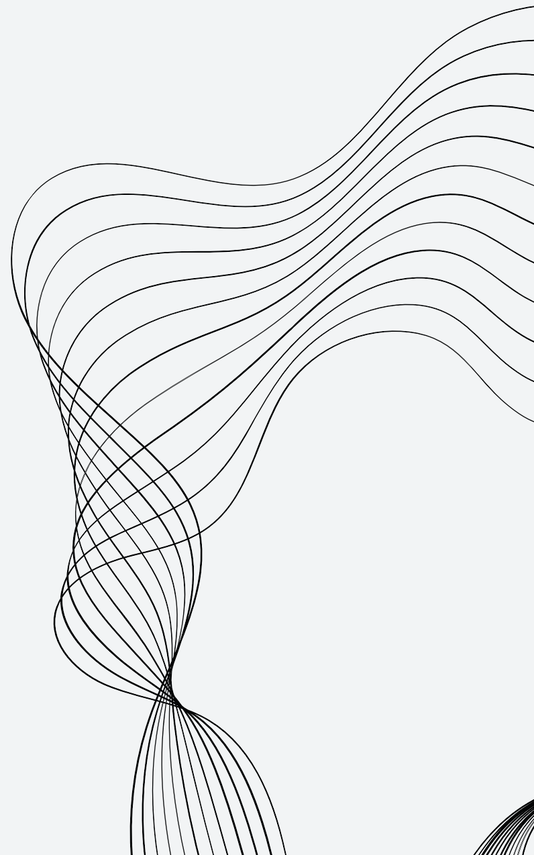
Predictive Technique (LR)

Linear Regression

```
import pandas as pd
df = pd.read_csv('/content/salary.csv')
print(df)

df1 = df.groupby('Age').max()
print(df1)
plt.plot(df1.index, df1['Salary'], marker='o')

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
X = df['Years of Experience']
df = df.dropna()
Y = df['Salary']

X = np.array(df['Years of Experience']).reshape(-1,1)
Y = np.array(df['Salary']).reshape(-1,1)

# Dropping any rows with Nan Values
X_train , X_test , y_train , y_test = train_test_split(X, Y, test_size = 0.25)

#Splitting the data into training and testing data
regr = LinearRegression()

regr.fit(X_train , y_train)
print(regr.score(X_test , y_test))
```

OUTPUT:

0.6394789681695328



Application

- By performing data manipulation techniques such as cleaning, filtering, and transforming the dataset, you can gain a deeper understanding of the data. Exploring summary statistics, distributions, and correlations between variables can provide insights into the characteristics and relationships within the dataset.
- Visualizing the Salary dataset can help uncover patterns, trends, and relationships between variables.
- Plots such as histograms, scatter plots and bar charts can provide visual representations.
- After performing data manipulation, visualizing the data, and clustering using Kmeans, the resulting clusters can serve as new features for predictive modeling.
- The cluster labels can be used as input features to build a classification model to predict survival or any other relevant outcome

Conclusion

- In conclusion, our analysis of the Salary dataset has provided valuable insights into the Employee information

- We discovered significant correlations between Employee and salarys such as age, gender, Education level, Job and Salary

- . Through data cleaning, preprocessing, visualization, and modeling, we were able to extract meaningful information

THANK YOU