Lab Exercise 1 - Data Exploration Parametric Methods

- Created by : Nileem Kaveramma C C | 2348441
- Created DATE:15-02-2024
- Edited Date: 15-02-2024

IMPORTED LIBRARIES

- numpy - for numerical, array, matrices (Linear Algebra) processing
- Pandas - for loading and processing datasets
- matplotlib.pyplot - For visualisation
- Saeborn - for statistical graph

```
from google.colab import drive
drive.mount('/content/drive')
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats
import plotly.express as px
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, ca
```

EMPLOYEE SALARY ANALYSIS he provided dataset captures information relevant to employee salary prediction, encompassing various attributes such as age, gender, education level, job title, years of experience, and salary. With a diverse set of features, the dataset offers valuable insights into the characteristics of individuals within an organizational context. This dataset becomes particularly relevant for exploring patterns and relationships that could contribute to predicting employee salaries. Through descriptive statistics, visualizations, and parametric tests, analysts can discern trends, potential disparities, and factors influencing salary variations among employees.

df is a commonly used variable name that often represents a DataFrame

```
df = pd.read_csv('/content/drive/My Drive/Salary Data.csv',encoding='unicode_es
df
```

	ï»¿Age	Gender	Education Level	Job Title	Years of Experience	Salary
0	32.0	Male	Bachelor's	Software Engineer	5.0	90000.0
1	28.0	Female	Master's	Data Analyst	3.0	65000.0
2	45.0	Male	PhD	Senior Manager	15.0	150000.0
3	36.0	Female	Bachelor's	Sales Associate	7.0	60000.0
4	52.0	Male	Master's	Director	20.0	200000.0
...	...	...	...	...	...	...
370	35.0	Female	Bachelor's	Senior Marketing Analyst	8.0	85000.0
371	43.0	Male	Master's	Director of Operations	19.0	170000.0
372	29.0	Female	Bachelor's	Junior Project Manager	2.0	40000.0
373	34.0	Male	Bachelor's	Senior Operations Coordinator	7.0	90000.0

df.shape - attribute is used to get the dimensions of the DataFrame.

```
df.shape
```

```
(375, 6)
```

df.columns attribute is used to retrieve the column labels or names of the DataFrame.

```
df.columns
```

```
Index(['ï»¿Age', 'Gender', 'Education Level', 'Job Title',
       'Years of Experience', 'Salary'],
      dtype='object')
```

df.dtypes attribute is used to retrieve the data types of each column in a DataFrame

```
df.dtypes
```

```
ï»¿Age                 float64
Gender                  object
Education Level         object
Job Title               object
Years of Experience    float64
Salary                 float64
dtype: object
```

df.head() method is used to display the first few rows of a DataFrame.

```
df.head()
```

	ï»¿Age	Gender	Education Level	Job Title	Years of Experience	Salary
0	32.0	Male	Bachelor's	Software Engineer	5.0	90000.0
1	28.0	Female	Master's	Data Analyst	3.0	65000.0
2	45.0	Male	PhD	Senior Manager	15.0	150000.0
3	36.0	Female	Bachelor's	Sales Associate	7.0	60000.0

The code df.isnull().count() in Pandas is used to count the total number of rows for each column in a DataFrame, including both missing (null or NaN) and non-missing values.

```
df.isnull().count()
```

```
ï»¿Age                 375
Gender                 375
Education Level        375
Job Title              375
Years of Experience    375
Salary                 375
dtype: int64
```

df.info() method in Pandas provides a concise summary of a DataFrame, including information about the data types, non-null values, and memory usage.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 375 entries, 0 to 374
Data columns (total 6 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   ï»¿Age               373 non-null    float64
 1   Gender               373 non-null    object 
 2   Education Level      373 non-null    object 
 3   Job Title            373 non-null    object 
 4   Years of Experience  373 non-null    float64
 5   Salary               373 non-null    float64
dtypes: float64(3), object(3)
memory usage: 17.7+ KB
```

The df.describe() method in Pandas is used to generate descriptive statistics that summarize the central tendency, dispersion, and shape of a dataset's distribution

```
df.describe()
```

	ï»¿Age	Years of Experience	Salary
count	373.000000	373.000000	373.000000
mean	37.431635	10.030831	100577.345845
std	7.069073	6.557007	48240.013482
min	23.000000	0.000000	350.000000
25%	31.000000	4.000000	55000.000000
50%	36.000000	9.000000	95000.000000
75%	44.000000	15.000000	140000.000000
max	53.000000	25.000000	250000.000000

Double-click (or enter) to edit

```
# Parametric methods (e.g., t-test)
# Example: Compare salaries between different education levels
education_levels = df['Education Level'].unique()

for level in education_levels:
    subset = df[df['Education Level'] == level]['Salary']
    print(f"\nSalary comparison for {level}:")
    print("Mean:", np.mean(subset))
    print("Standard Deviation:", np.std(subset))
```

```
Salary comparison for Bachelor's:
Mean: 74756.02678571429
Standard Deviation: 34699.55803057353

Salary comparison for Master's:
Mean: 129795.91836734694
Standard Deviation: 41446.53777511888

Salary comparison for PhD:
Mean: 157843.13725490196
Standard Deviation: 23162.99664739093

Salary comparison for nan:
Mean: nan
Standard Deviation: nan
```

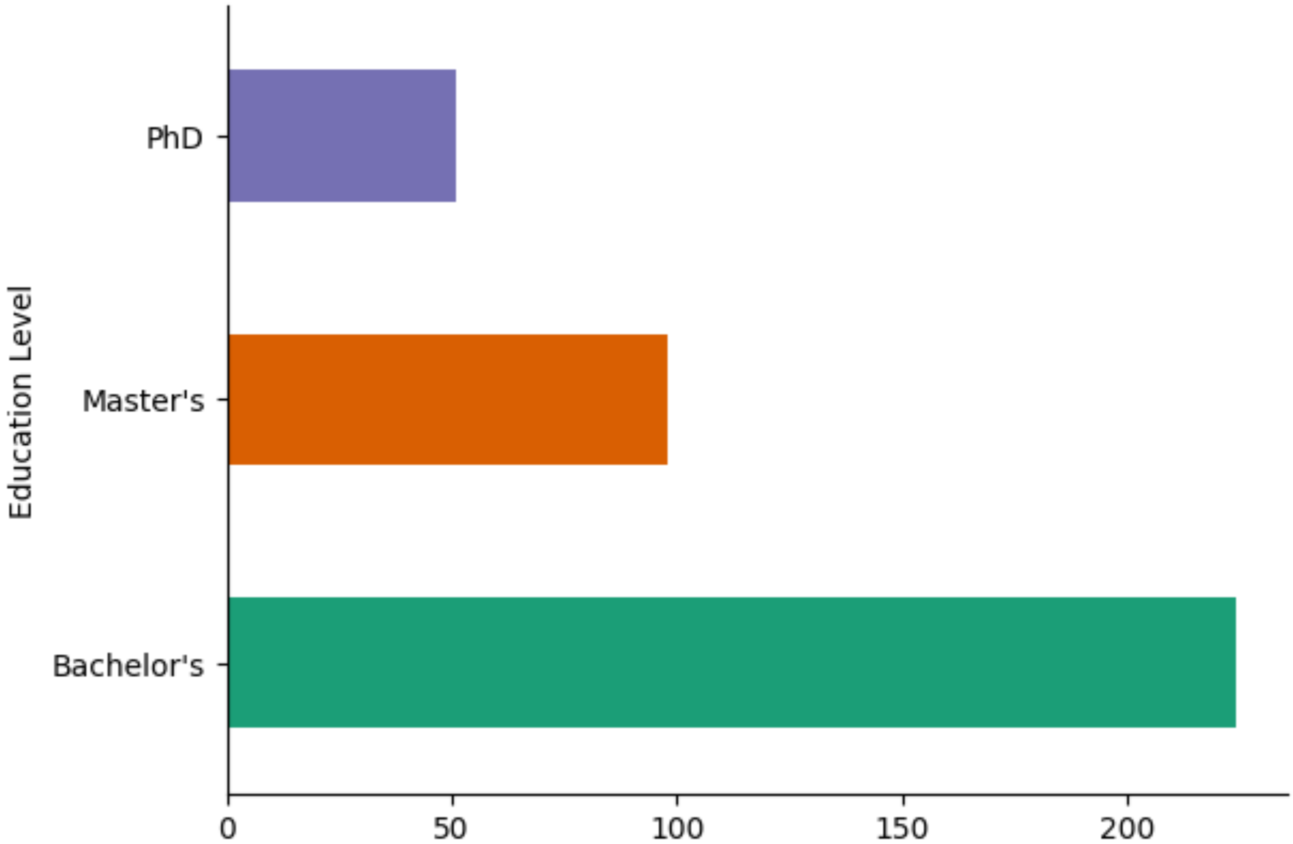
```
# Distribution of Salary using histogram and Q-Q plot
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
sns.histplot(df['Salary'], bins=20, kde=True)
plt.title('Distribution of Salary')
```

```
Text(0.5, 1.0, 'Distribution of Salary')
```

The provided code snippet utilizes the matplotlib library to generate a histogram for the 'Age' column in a DataFrame (df). The histogram is created with 20 bins, offering a visual representation of the distribution of ages in the dataset. The title of the plot is set to 'Age' for clarity.

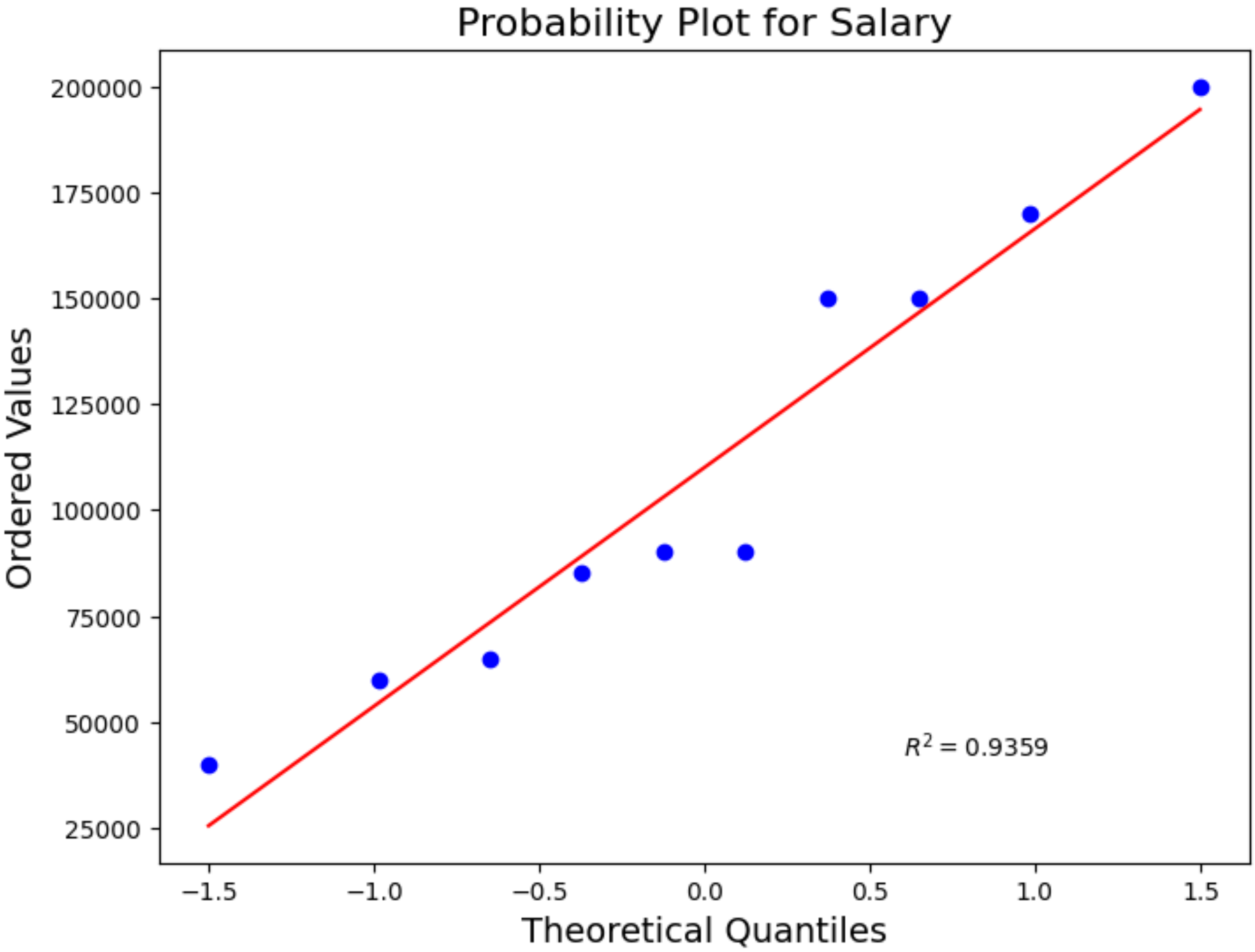
This code snippet utilizes both matplotlib and seaborn libraries to create a horizontal bar plot that visualizes the count of individuals in the DataFrame (df) based on their 'Education Level'.

```
from matplotlib import pyplot as plt
import seaborn as sns
df.groupby('Education Level').size().plot(kind='barh', color=sns.palettes.mpl_p
plt.gca().spines[['top', 'right',]].set_visible(False)
```

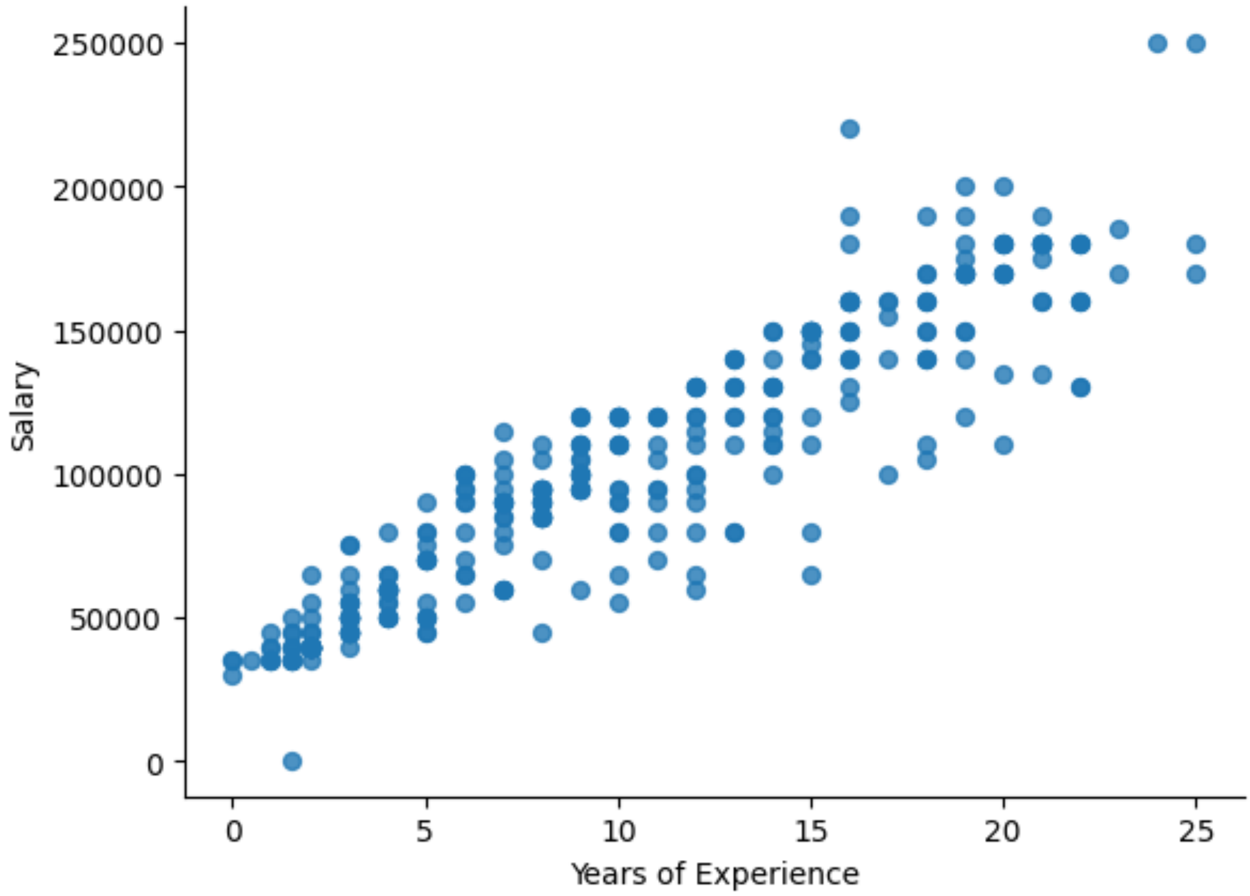



```
#Create probability plots to assess whether your data follow a specific distribu
#such as the normal distribution.

import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import probplot

# Create DataFrame
data = {
    'Age': [32.0, 28.0, 45.0, 36.0, 52.0, 35.0, 43.0, 29.0, 34.0, 44.0],
    'Gender': ['Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'Fe
    'Education Level': ["Bachelor's", "Master's", 'PhD', "Bachelor's", "Master's
    'Job Title': ['Software Engineer', 'Data Analyst', 'Senior Manager', 'Sales
    'Years of Experience': [5.0, 3.0, 15.0, 7.0, 20.0, 8.0, 19.0, 2.0, 7.0, 15.0
    'Salary': [90000.0, 65000.0, 150000.0, 60000.0, 200000.0, 85000.0, 170000.0,
}
```

```
df = pd.DataFrame(data)

# Create probability plot for 'Salary'
plt.figure(figsize=(8, 6))
probplot(df['Salary'], plot=plt, fit=True, rvalue=True)

# Customize plot
plt.title('Probability Plot for Salary', fontsize=16)
plt.xlabel('Theoretical Quantiles', fontsize=14)
plt.ylabel('Ordered Values', fontsize=14)

# Show plot
plt.show()
```

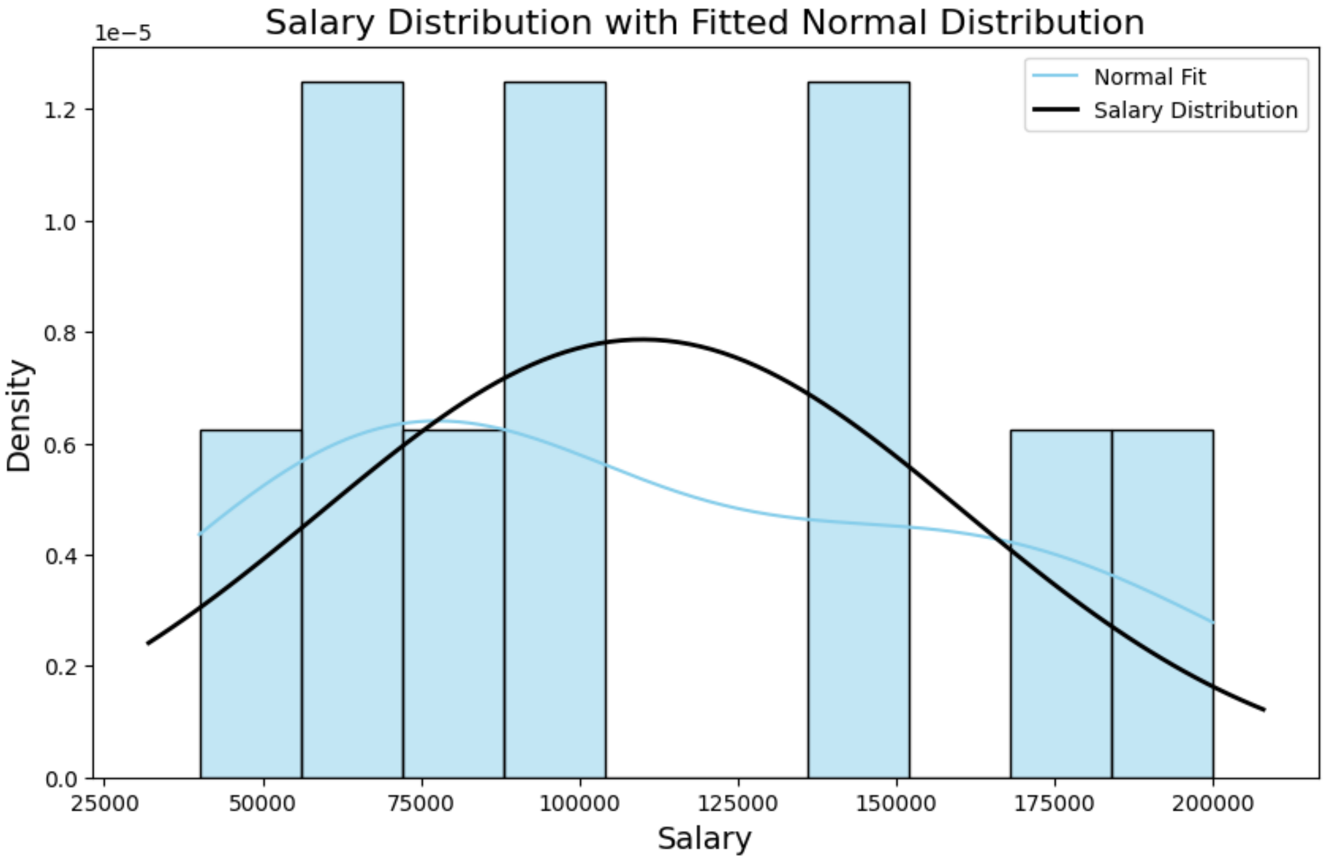


```
# Years of Experience vs Salary

from matplotlib import pyplot as plt
df.plot(kind='scatter', x='Years of Experience', y='Salary', s=32, alpha=.8)
plt.gca().spines[['top', 'right',]].set_visible(False)
```




```
# Plot histogram and fitted normal distribution

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm

# Create DataFrame
data = {
    'Age': [32.0, 28.0, 45.0, 36.0, 52.0, 35.0, 43.0, 29.0, 34.0, 44.0],
    'Gender': ['Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'F
    'Education Level': ["Bachelor's", "Master's", 'PhD', "Bachelor's", "Master'
    'Job Title': ['Software Engineer', 'Data Analyst', 'Senior Manager', 'Sales
    'Years of Experience': [5.0, 3.0, 15.0, 7.0, 20.0, 8.0, 19.0, 2.0, 7.0, 15.
    'Salary': [90000.0, 65000.0, 150000.0, 60000.0, 200000.0, 85000.0, 170000.0
}
```

```
df = pd.DataFrame(data)

# Plot histogram
plt.figure(figsize=(10, 6))
sns.histplot(df['Salary'], bins=10, kde=True, color='skyblue', stat='density')

# Fit normal distribution
mu, std = norm.fit(df['Salary'])
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)
plt.plot(x, p, 'k', linewidth=2)

# Customize plot
plt.title('Salary Distribution with Fitted Normal Distribution', fontsize=16)
plt.xlabel('Salary', fontsize=14)
plt.ylabel('Density', fontsize=14)
plt.legend(['Normal Fit', 'Salary Distribution'])

# Show plot
plt.show()
```


Salary Distribution with Fitted Normal Distribution
1e−5
Normal Fit
Salary Distribution
Density
Salary
0.0
0.2
0.4
0.6
0.8
1.0
1.2
25000
50000
75000
100000
125000
150000
175000
200000

```
#Perform parametric tests to compare groups or test hypotheses about your data.

# Assuming you have loaded your data into a DataFrame named 'df'
# If not, you can read your data from a file or another source

import pandas as pd
from scipy.stats import ttest_ind

data = {
    'Age': [32.0, 28.0, 45.0, 36.0, 52.0, 35.0, 43.0, 29.0, 34.0, 44.0],
    'Gender': ['Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'F
    'Education Level': ["Bachelor's", "Master's", 'PhD', "Bachelor's", "Master'
    'Job Title': ['Software Engineer', 'Data Analyst', 'Senior Manager', 'Sales
    'Years of Experience': [5.0, 3.0, 15.0, 7.0, 20.0, 8.0, 19.0, 2.0, 7.0, 15.
    'Salary': [90000.0, 65000.0, 150000.0, 60000.0, 200000.0, 85000.0, 170000.0
}

df = pd.DataFrame(data)

# Example: Compare salaries between Male and Female using t-test
male_salary = df[df['Gender'] == 'Male']['Salary']
female_salary = df[df['Gender'] == 'Female']['Salary']

# Perform independent t-test
t_statistic, p_value = ttest_ind(male_salary, female_salary)

# Print results
print("T-statistic:", t_statistic)
print("P-value:", p_value)

# Interpret the results
if p_value < 0.05:
    print("The difference in salaries between Male and Female is statistically
else:
    print("There is no significant difference in salaries between Male and Fema
```

```
T-statistic: 2.073284221395264
P-value: 0.07186138591486624
There is no significant difference in salaries between Male and Female.
```

plt.hist(female_heights, bins=5, density=True, alpha=0.6, color='g' label='Female Salary')

This line of code creates a histogram of the female heights.

It plots the distribution of female heights using a histogram with 5 bins.

The density=True parameter normalizes the histogram so that the area under the histogram equals 1, making it a probability density function.

The alpha=0.6 parameter controls the transparency of the histogram bars, and color='g' sets the color to green. Finally, label='Female Heights' assigns a label to the histogram for use in the legend.

xmin, xmax = plt.xlim() x = np.linspace(xmin, xmax, 100) p = stats.norm.pdf(x, mu, sigma)

These lines of code calculate the probability density function (PDF) of the fitted normal distribution for female heights.

It generates 100 equally spaced points (x) between the minimum and maximum observed heights (xmin and xmax).

Then, it computes the PDF (p) of the normal distribution with mean mu and standard deviation sigma at these points using stats.norm.pdf() from SciPy.

plt.plot(x, p, 'k', linewidth=2, label='Fitted Normal Distribution')

This line of code plots the fitted normal distribution on the same plot as the histogram. It uses plt.plot() to draw a line plot of the PDF (p) against the height values (x). The 'k' argument sets the line color to black, and linewidth=2 adjusts the line width. The label='Fitted Normal Distribution' assigns a label to this line for use in the legend.

Output Explanation:

The output of the code is a plot that consists of a histogram representing the distribution of female heights and a line plot representing the fitted normal distribution.

The histogram shows the frequency or density of female heights in different height ranges, while the fitted normal distribution provides an approximation of the underlying probability distribution of female heights assuming it follows a normal (Gaussian) distribution.

The plot allows visual comparison between the observed data and the fitted distribution, which can help assess how well the normal distribution fits the data.