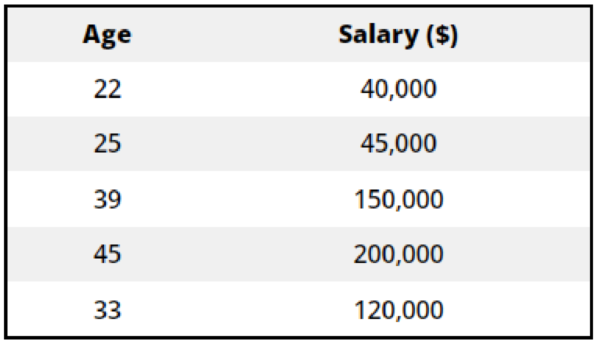
SET 1

1. Consider the age and salary of Individuals in a town.Find the correlation matrix, covariance, plot the correlation plot on dataset and visualize giving an overview of relationships among data on sample dataset



Age	Salary ($)
22	40,000
25	45,000
39	150,000
45	200,000
33	120,000

```
import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

age = [22, 25, 39, 45, 33]

salary = [40000, 45000, 150000, 200000, 1200000]

df = pd.DataFrame({'age': age, 'salary': salary})

corr_matrix = df.corr()

covariance = df.cov()

sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', linewidths=0.5, linecolor='white')

plt.title('Correlation Plot')

plt.show()
```

2. Consider a sample diabetic dataset and plot the following Univariate graphs to show the distribution of data from a single variable. Write the inference for each and every plot using bar graph and line chart.

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.datasets import load_diabetes

diabetes = load_diabetes()

df = pd.DataFrame(diabetes.data, columns=diabetes.feature_names)

for feature in df.columns:

    plt.figure(figsize=(10, 4))

    plt.subplot(1, 2, 1)
```

```
sns.histplot(df[feature], kde=False, bins=30, color='skyblue')

plt.title(f'Bar Graph for {feature.capitalize()}')

plt.subplot(1, 2, 2)

sns.kdeplot(df[feature], color='darkblue')

plt.title(f'Line Chart for {feature.capitalize()}')

plt.tight_layout()

plt.show()
```

3. Write a Pandas program to create and display a DataFrame from a specified dictionary data which has the index labels.
   Sample Python dictionary data and list labels:
   exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
   'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
   'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
   'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
   labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j'].

```
import pandas as pd

import numpy as np

exam_data = {

    'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

    'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']

}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data, index=labels)

print(df)
```

4. You are a data analyst working for a car manufacturing company. As part of your analysis, you have a dataset containing information about the fuel efficiency of different car models. The dataset is stored in a NumPy array named fuel_efficiency, where each element represents the fuel efficiency (in miles per gallon) of a specific car model. Your task is to calculate the average fuel efficiency and determine the percentage improvement in fuel efficiency between two car models. Calculate the average fuel efficiency and determine the percentage improvement in fuel efficiency between two car models?

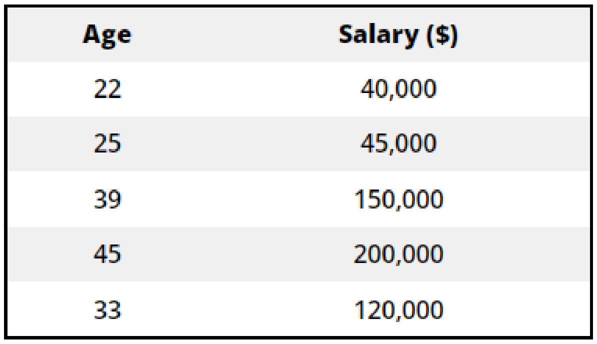
```
import numpy as np

np.random.seed(0)

fuel_efficiency = np.random.uniform(low=20.0, high=50.0, size=10)
```

```
average_efficiency = np.mean(fuel_efficiency)

print(f"Average fuel efficiency: {average_efficiency} miles per gallon")

model1, model2 = 0, 1

efficiency_improvement = (fuel_efficiency[model2] - fuel_efficiency[model1]) / fuel_efficiency[model1] * 100

print(f"Percentage improvement in fuel efficiency between model {model1} and model {model2}: {efficiency_improvement}%")
```

SET 2

1. Consider the above dataset and find the Correlation for the attributes age and salary. Check the linearity of the data using the following Correlation Coefficient.

- Pearson Correlation Coefficient
- Spearman’s Rank Correlation Coefficient



Age	Salary ($)
22	40,000
25	45,000
39	150,000
45	200,000
33	120,000

```
import numpy as np

from scipy.stats import pearsonr, spearmanr

age = np.array([22, 25, 39, 45, 33])

salary = np.array([40000, 45000, 150000, 200000, 120000])

pearson_corr, _ = pearsonr(age, salary)

print(f"Pearson Correlation Coefficient: {pearson_corr}")

spearman_corr, _ = spearmanr(age, salary)

print(f"Spearman's Rank Correlation Coefficient: {spearman_corr}")
```

2. Consider a sample housing dataset and plot the bivariate graphs to show the distribution of data from two variables for Scatterplot and Stacked Bar Chart

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.DataFrame({
   'var1': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
   'var2': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
})
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='var1', y='var2')
plt.title('Scatterplot of var1 vs var2')
plt.show()
df_grouped = df.groupby('var1')['var2'].value_counts().unstack()
df_grouped.plot(kind='bar', stacked=True)
```

```
plt.title('Stacked Bar Chart of var1 and var2')
plt.show()
```

3. Write a Pandas program to get the first 3 rows of a given DataFrame.
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

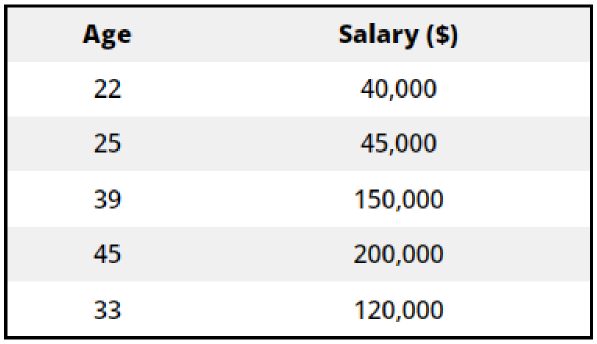
```
import pandas as pd

import numpy as np

exam_data = {

    'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

    'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']

}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data, index=labels)

print(df.head(3))
```

4. A company wants to know the most popular product they sell. They have a list of all the products they have sold in the past year, along with the number of times each product was sold. Write a program that will calculate the frequency distribution of products sold and print out the most popular product.

```
from collections import Counter

products_sold = ['Apple', 'Banana', 'Apple', 'Cherry', 'Banana', 'Apple', 'Apple', 'Banana', 'Cherry', 'Apple', 'Banana',
'Cherry', 'Cherry', 'Cherry']

product_frequency = Counter(products_sold)

most_popular_product = product_frequency.most_common(1)[0]

print(f"The most popular product is '{most_popular_product[0]}' with {most_popular_product[1]} sales.")
```

SET 3

1. Consider the below. Calculate Correlation and Covariance. Give the inference of both Covariance and Correlation values



Age	Salary ($)
22	40,000
25	45,000
39	150,000
45	200,000
33	120,000

```
import numpy as np
age = np.array([22, 25, 39, 45, 33])
salary = np.array([40000, 45000, 150000, 200000, 120000])
correlation = np.corrcoef(age, salary)[0, 1]
print(f"Correlation: {correlation}")
covariance = np.cov(age, salary)[0, 1]
print(f"Covariance: {covariance}")
```

2. Consider a sample car dataset and plot Mutlivariate graphs to show the distribution of data from multiple variables for Multivariate Scatterplot and Scatter Plot Matrix

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.DataFrame({
   'Mileage': [15000, 30000, 45000, 60000, 75000, 90000, 105000, 120000, 135000, 150000],
   'Price': [30000, 28000, 26000, 24000, 22000, 20000, 18000, 16000, 14000, 12000],
   'Horsepower': [150, 160, 170, 180, 190, 200, 210, 220, 230, 240]
})
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='Mileage', y='Price', hue='Horsepower')
plt.title('Multivariate Scatterplot of Mileage vs Price colored by Horsepower')
plt.show()
sns.pairplot(df)
plt.show()
```

4. Write a Pandas program to select the rows where the score is missing, i.e. is NaN.
   Sample Python dictionary data and list labels:
   exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
   'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
   'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
   'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
   labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

```
import pandas as pd

import numpy as np

exam_data = {

   'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

   'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

   'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

   'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data, index=labels)

missing_score_rows = df[df['score'].isna()]

print(missing_score_rows)
```

5. A school wants to know the most popular subject among their students. They have a list of all the subjects that their students have taken in the past year, along with the number of students who have taken each subject. Write a program that will calculate the frequency distribution of subjects taken and print out the most popular subject

```
from collections import Counter

subjects_taken = ['Math', 'English', 'Science', 'Math', 'History', 'Math', 'English', 'Math', 'History', 'Math', 'Science', 'Math']

subject_frequency = Counter(subjects_taken)

most_popular_subject = subject_frequency.most_common(1)[0]

print(f"The most popular subject is '{most_popular_subject[0]}' with {most_popular_subject[1]} students.")
```

SET 4

1. Write a Pandas program to calculate the sum of the examination attempts by the students.
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

```
import pandas as pd

exam_data = {

  'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

  'score': [12.5, 9, 16.5, None, 9, 20, 14.5, None, 8, 19],

  'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

  'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']

}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data, index=labels)

total_attempts = df['attempts'].sum()

print(f"The total number of examination attempts by the students is {total_attempts}.")
```

2. A weather station wants to know the most common types of weather in their area. They have a list of all the weather conditions that have occurred in the past year, along with the number of times each weather condition has occurred. Write a program that will calculate the frequency distribution of weather conditions and print out the most common weather type.

```
from collections import Counter

weather_conditions = ['Sunny', 'Rainy', 'Sunny', 'Cloudy', 'Sunny', 'Sunny', 'Rainy', 'Sunny', 'Cloudy', 'Sunny', 'Rainy', 'Cloudy', 'Cloudy', 'Cloudy']

weather_frequency = Counter(weather_conditions)

most_common_weather = weather_frequency.most_common(1)[0]

print(f"The most common weather type is '{most_common_weather[0]}' with {most_common_weather[1]} occurrences.")
```

3. Write a program to perform a regression analysis on a given dataset using both linear regression and logistic regression. Provide visualization for the same.

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression, LogisticRegression

import matplotlib.pyplot as plt

df = pd.DataFrame({

    'feature': np.random.randint(1, 100, 100),

    'target': np.random.randint(0, 2, 100)

})

X_train, X_test, y_train, y_test = train_test_split(df['feature'].values.reshape(-1,1), df['target'], test_size=0.2, random_state=42)

linear_reg = LinearRegression()

linear_reg.fit(X_train, y_train)

y_pred_linear = linear_reg.predict(X_test)

plt.figure(figsize=(10, 6))

plt.scatter(X_test, y_test, color='blue')

plt.plot(X_test, y_pred_linear, color='red')

plt.title('Linear Regression')

plt.xlabel('Feature')
plt.ylabel('Target')

plt.show()

logistic_reg = LogisticRegression()

logistic_reg.fit(X_train, y_train)

y_pred_logistic = logistic_reg.predict(X_test)

plt.figure(figsize=(10, 6))

plt.scatter(X_test, y_test, color='blue')

plt.scatter(X_test, y_pred_logistic, color='red')

plt.title('Logistic Regression')

plt.xlabel('Feature')

plt.ylabel('Target')

plt.show()
```

4. You are working on a classification problem to predict whether a patient has a certain medical condition or not based on their symptoms. You have collected a dataset of patients with labeled data (0 for no condition, 1 for the condition) and various symptom features. Write a Python program that allows the user to input the features of a new patient and the value of k (number of neighbors). The program should use the KNN classifier from the scikit-learn library to predict whether the patient has the medical condition or not based on the input features.

```
from sklearn.neighbors import KNeighborsClassifier

import numpy as np

X = np.array([[1, 2], [2, 3], [3, 4], [4, 5], [5, 6], [6, 7], [7, 8], [8, 9], [9, 10], [10, 11]])

y = np.array([0, 0, 0, 0, 1, 1, 1, 1, 1, 1])

k = 3

knn = KNeighborsClassifier(n_neighbors=k)

knn.fit(X, y)

new_patient = np.array([5, 6])

prediction = knn.predict(new_patient.reshape(1, -1))

if prediction[0] == 0:

    print("The patient does not have the condition.")

else:

    print("The patient has the condition.")
```

SET 5

1. Write a Pandas program to replace the 'qualify' column contains the values 'yes' and 'no' with True and False.
   Sample Python dictionary data and list labels:
   exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
   'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
   'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
   'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
   labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

```
import pandas as pd



# Sample Python dictionary data and list labels

exam_data = {

    'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

    'score': [12.5, 9, 16.5, None, 9, 20, 14.5, None, 8, 19],

    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']

}

labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']



# Create DataFrame

df = pd.DataFrame(exam_data, index=labels)



# Replace 'yes' and 'no' in 'qualify' column with True and False

df['qualify'] = df['qualify'].map({'yes': True, 'no': False})



print(df)
```

2. You are working on a data visualization project and need to create basic plots using Matplotlib. You have a dataset containing the monthly sales data for a company, including the month and corresponding sales values. Your task is to develop a Python program that generates line plots and bar plots to visualize the sales data. How would you develop a Python program to create a line plot and bar chart of the monthly sales data?

```
import matplotlib.pyplot as plt
```

```
# Sample dataset containing the months and corresponding sales values

months = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']

sales = [1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000, 5500, 6000, 6500]



# Create a line plot

plt.figure(figsize=(10, 6))

plt.plot(months, sales, marker='o')

plt.title('Monthly Sales Data (Line Plot)')

plt.xlabel('Month')

plt.ylabel('Sales')

plt.grid(True)

plt.show()



# Create a bar chart

plt.figure(figsize=(10, 6))

plt.bar(months, sales)

plt.title('Monthly Sales Data (Bar Chart)')

plt.xlabel('Month')

plt.ylabel('Sales')

plt.show()
```

3. A government agency wants to know the most common causes of accidents in their jurisdiction. They have a list of all the accidents that have occurred in the past year, along with the cause of each accident. Write a program that will calculate the frequency distribution of accident causes and print out the most common cause of accidents.

```
from collections import Counter

accident_causes = ['Speeding', 'Distracted Driving', 'Drunk Driving', 'Reckless Driving', 'Rain', 'Running Red Lights', 'Running Stop Signs', 'Teenage Drivers', 'Night Driving', 'Design Defects', 'Unsafe Lane Changes', 'Wrong-Way Driving', 'Improper Turns', 'Tailgating', 'Sleep-Deprived Driving', 'Speeding', 'Distracted Driving', 'Speeding', 'Drunk Driving', 'Speeding', 'Distracted Driving']
```

```
cause_frequency = Counter(accident_causes)

most_common_cause = cause_frequency.most_common(1)[0]

print(f"The most common cause of accidents is '{most_common_cause[0]}' with {most_common_cause[1]} occurrences.")
```

4. You are working on a project that involves analyzing the sales performance of a company over the past four quarters. The quarterly sales data is stored in a NumPy array named `sales_data`, where each element represents the sales amount for a specific quarter. Your task is to calculate the total sales for the year and determine the percentage increase in sales from the first quarter to the fourth quarter.Using NumPy arrays and arithmetic operations calculate the total sales for the year and determine the percentage increase in sales from the first quarter to the fourth quarter?

```
import numpy as np

sales_data = np.array([1000, 2000, 3000, 4000])  # replace with actual data

total_sales = np.sum(sales_data)

print(f"Total sales for the year: {total_sales}")

percentage_increase = ((sales_data[3] - sales_data[0]) / sales_data[0]) * 100

print(f"Percentage increase in sales from the first quarter to the fourth quarter: {percentage_increase}%")
```

SET 6

1. Write a Pandas program to iterate over rows in a DataFrame.
   Sample Python dictionary data and list labels:
   exam_data = [{'name':'Anastasia', 'score':12.5}, {'name':'Dima','score':9}, {'name':'Katherine','score':16.5}]

```
import pandas as pd

exam_data = [{'name':'Anastasia', 'score':12.5}, {'name':'Dima','score':9}, {'name':'Katherine','score':16.5}]

df = pd.DataFrame(exam_data)

for index, row in df.iterrows():

    print(row['name'], row['score'])
```

2. You are working on a data analysis project that involves analyzing the monthly temperature and rainfall data for a city. You have a dataset containing the monthly temperature and rainfall values for each month of a year. Your task is to develop a Python program that generates line plots and scatter plots to visualize the temperature and rainfall data. Develop a Python program to create a scatter plot and line plot of the monthly rainfall data.

```
import matplotlib.pyplot as plt

months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']

temperature = [20, 22, 25, 28, 30, 32, 35, 33, 31, 28, 25, 22]  # in degree Celsius

rainfall = [50, 40, 45, 70, 80, 90, 100, 90, 80, 70, 60, 55]  # in mm

plt.figure(figsize=(10, 4))
plt.plot(months, temperature, marker='o')

plt.title('Monthly Temperature')

plt.xlabel('Month')

plt.ylabel('Temperature (°C)')

plt.grid(True)

plt.show()

plt.figure(figsize=(10, 4))

plt.scatter(months, rainfall, color='r')

plt.title('Monthly Rainfall')

plt.xlabel('Month')

plt.ylabel('Rainfall (mm)')

plt.grid(True)

plt.show()
```

3. A hospital wants to know the most common diseases among their patients. They have a list of all the diseases that their patients have been diagnosed with in the past year, along with the number of patients who have been diagnosed with each disease. Write a program that will calculate the frequency distribution of diseases and print out the most common disease.

```
diseases = ['Flu', 'Cold', 'COVID-19', 'Diabetes', 'Hypertension', 'Flu', 'Cold', 'Flu', 'Diabetes', 'Flu', 'Cold', 'COVID-19']

frequency distribution of diseases

frequency_distribution = {}

for disease in diseases:

    if disease not in frequency_distribution:

        frequency_distribution[disease] = 0

    frequency_distribution[disease] += 1

for disease, frequency in frequency_distribution.items():

    print(f'{disease}: {frequency}')

most_common_disease = max(frequency_distribution, key=frequency_distribution.get)
```

```
    print(f'\nThe most common disease is {most_common_disease}.')
```

4. You have trained a machine learning model on a dataset, and now you want to evaluate its performance using various metrics. Write a Python program that loads a dataset and trained model from scikit-learn. The program should ask the user to input the names of the features and the target variable they want to use for evaluation. The program should then calculate and display common evaluation metrics such as accuracy, precision, recall, and F1-score for the model's predictions on the test data.

```
from sklearn import datasets

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

from sklearn.ensemble import RandomForestClassifier

from sklearn.preprocessing import LabelEncoder

iris = datasets.load_iris()

X = iris.data

y = iris.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

clf = RandomForestClassifier(random_state=42)

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)

precision = precision_score(y_test, y_pred, average='macro')

recall = recall_score(y_test, y_pred, average='macro')

f1 = f1_score(y_test, y_pred, average='macro')

print(f'Accuracy: {accuracy}')

print(f'Precision: {precision}')

print(f'Recall: {recall}')

print(f'F1-score: {f1}')
```

SET 7

1. Write a Pandas program to combining two series into a DataFrame.

```
import pandas as pd

s1 = pd.Series(['a', 'b', 'c'])

s2 = pd.Series([1, 2, 3])

df = pd.DataFrame({'col1': s1, 'col2': s2})

print(df)
```

2. You are a data scientist working for an e-commerce company. The marketing team has conducted an A/B test to evaluate the effectiveness of two different website designs (A and B) in terms of conversion rate. They randomly divided the website visitors into two groups, with one group experiencing design A and the other experiencing design B. After a week of data collection, you now have the conversion rate data for both groups. You want to determine whether there is a statistically significant difference in the mean conversion rates between the two website designs. "Based on the data collected from the A/B test, is there a statistically significant difference in the mean conversion rates between website design A and website design B?"

```
import numpy as np

from scipy import stats

conversion_rates_A = np.array([0.1, 0.15, 0.2, 0.2, 0.15, 0.2, 0.1])

conversion_rates_B = np.array([0.12, 0.18, 0.22, 0.21, 0.16, 0.21, 0.11])

t_stat, p_value = stats.ttest_ind(conversion_rates_A, conversion_rates_B)

print(f't-statistic: {t_stat}')

print(f'p-value: {p_value}')

if p_value < 0.05:

   print("There is a statistically significant difference in the mean conversion rates between website design A and website design B.")

else:

   print("There is no statistically significant difference in the mean conversion rates between website design A and website design B.")
```

3. A weather station wants to know if there is a correlation between the temperature and the amount of rainfall in a city. They have data on the temperature and rainfall each day for the past year in that city. Write a program that will calculate the correlation coefficient between temperature and rainfall, and create a scatter plot of the data.

```
import numpy as np

import matplotlib.pyplot as plt

temperature = np.array([20, 22, 25, 28, 30, 32, 35, 33, 31, 28, 25, 22])  # in degree Celsius

rainfall = np.array([50, 40, 45, 70, 80, 90, 100, 90, 80, 70, 60, 55])  # in mm

correlation_coefficient = np.corrcoef(temperature, rainfall)[0, 1]

print(f'Correlation coefficient: {correlation_coefficient}')

plt.scatter(temperature, rainfall)

plt.title('Temperature vs Rainfall')

plt.xlabel('Temperature (°C)')

plt.ylabel('Rainfall (mm)')

plt.grid(True)

plt.show()
```

4. Suppose you are working as a data scientist for a medical research organization. Your team has collected data on patients with a certain medical condition and their treatment outcomes. The dataset includes various features such as age, gender, blood pressure, cholesterol levels, and whether the patient responded positively ("Good") or negatively ("Bad") to the treatment. The organization wants to use this model to identify potential candidates who are likely to respond positively to the treatment and improve their medical approach. Build a classification model using the KNN algorithm to predict the treatment outcome ("Good" or "Bad") for new patients based on their features. Evaluate the model's performance using accuracy, precision, recall, and F1-score.Make predictions on the test set and display the results.

```
from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

from sklearn.preprocessing import LabelEncoder

import numpy as np

features = np.array([[50, 1, 120, 200], [30, 0, 110, 180], [60, 1, 140, 220], [40, 0, 130, 210]])

outcomes = np.array(['Good', 'Good', 'Bad', 'Bad'])

le = LabelEncoder()

outcomes = le.fit_transform(outcomes)

X_train, X_test, y_train, y_test = train_test_split(features, outcomes, test_size=0.2, random_state=42)
```

```
clf = KNeighborsClassifier(n_neighbors=3)

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)

precision = precision_score(y_test, y_pred, average='macro')

recall = recall_score(y_test, y_pred, average='macro')

f1 = f1_score(y_test, y_pred, average='macro')

print(f'Accuracy: {accuracy}')

print(f'Precision: {precision}')

print(f'Recall: {recall}')

print(f'F1-score: {f1}')
```

SET 8

1. A company wants to know if there is a correlation between the number of sales they make and the amount of advertising they spend. They have data on the number of sales they made each month for the past year, as well as the amount of advertising they spent each month. Write a program that will calculate the correlation coefficient between sales and advertising, and create a scatter plot of the data.

2. Write a Pandas program to select the specified columns and rows from a given data frame.Sample Python dictionary data and list labels:
Select 'name' and 'score' columns in rows 1, 3, 5, 6 from the following data frame.
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

3. Imagine you are an analyst for a popular online shopping website. Your task is to analyze customer reviews and provide insights on the average rating and customer satisfaction level for a specific product category. You will use the pandas library to calculate confidence intervals to estimate the true population mean rating. You have been provided with a CSV file named "customer_reviews.csv," which contains customer ratings for products in the chosen category.

4. A hospital wants to know if there is a correlation between the number of patients who smoke and the number of patients who get lung cancer. They have data on the number of patients who smoke and the number of patients who get lung cancer each year. Write a program that will calculate the correlation coefficient between smoking and lung cancer, and create a scatter plot of the data.

SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

Course Code:DSA0405 Course Title Fundamentals of Data Science For Economic Analysis

MODEL LAB EXAMINATION

SET 9

1. You are a data analyst working for a finance company. Your team is interested in analyzing the variability of stock prices for a particular company over a certain period. The company's stock data includes the closing prices for each trading day of the specified period. Your task is to build a Python program that reads the stock data from a CSV file, calculates the variability of stock prices, and provides insights into the stock's price movements.

2. Write a Pandas program to select the rows the score is between 15 and 20 (inclusive).
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}

3. A hospital wants to know if there is a correlation between the number of patients who smoke and the number of patients who get lung cancer. They have data on the number of patients who smoke and the number of patients who get lung cancer each year. Write a program that will calculate the correlation coefficient between smoking and lung cancer, and create a scatter plot of the data.

4. Write a python program will take in a dataset containing daily temperature readings for each city over a year and perform the following tasks:

1. Calculate the mean temperature for each city.
2. Calculate the standard deviation of temperature for each city.
3. Determine the city with the highest temperature range (difference between the highest and lowest temperatures).
4. Find the city with the most consistent temperature (the lowest standard deviation).

SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

Course Code:DSA0405 Course Title Fundamentals of Data Science For Economic Analysis

MODEL LAB EXAMINATION

SET 10

1. Write a Pandas program to create and display a DataFrame from a specified dictionary data which has the index labels.
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

2. You are a data scientist working for a company that sells shoes. You are tasked with writing a program that will calculate the frequency distribution of shoe sizes sold in the past year. The data is stored in a file called shoe_sales.csv. The file contains the following columns:

- shoe_size: The size of the shoe sold.
- quantity: The number of shoes sold in that size.
- Write a program that will read the data from the file and calculate the frequency distribution of shoe sizes. The program should output the frequency distribution table, as well as a bar chart showing the frequency of each shoe size.

3. A government agency wants to know if there is a correlation between the number of crimes committed in a city and the amount of poverty in that city. They have data on the number of crimes committed each year in each city, as well as the percentage of people living in poverty in each city. Write a program that will calculate the correlation coefficient between crime rate and poverty rate, and create a scatter plot of the data.

4. You work as a data scientist for an automobile company that sells various car models. The company has collected data on different car attributes, such as engine size, horsepower, fuel efficiency, and more, along with their corresponding prices. The marketing team wants to build a predictive model to estimate the price of cars based on their features. Write a Python program that perform linear regression modeling to predict car prices based on a selected set of features from the dataset. Additionally, you need to evaluate the model's performance and provide insights to the marketing team to understand the most influential factors affecting car prices.