

# **K.S. SCHOOL OF ENGINEERING AND MANAGEMENT, BENGALURU**

## **DEPARTMENT OF COMPUTER SCIENCE AND BUSINESS SYSTEMS**

### **MACHINE LEARNING LABORATORY (BCSL606)**



**Prepared by: -**

**Mrs. Frinkly Sathanga Shanija.  
Assistant Professor**

Dept. of CS & BS,  
KSSEM

**Ms. Harini Karan  
Assistant Professor**  
Dept. of CS & BS,  
KSSEM

## **LAB MANUAL**

**ACADEMIC YEAR 2025 – 2026**

# **Institution Vision & Mission**

## **VISION:**

“To impart quality education in engineering and management to meet technological, business and societal needs through holistic education and research”

## **MISSION:**

K.S. School of Engineering and Management shall,

- ❖ Establish state-of-art infrastructure to facilitate effective dissemination of technical and Managerial knowledge.
- ❖ Provide comprehensive educational experience through a combination of curricular and experiential learning, strengthened by industry-institute- interaction.
- ❖ Pursue socially relevant research and disseminate knowledge.
- ❖ Inculcate leadership skills and foster entrepreneurial spirit among students.

# **Department Vision & Mission**

## **VISION:**

“To provide competent learning ecosystem to develop the understanding of technology and business to produce innovative, principled and insightful leaders to meet the societal demands.”

## **MISSION:**

- ❖ To deliver high-quality education in the fields of technology and business through effective teaching-learning practices and a conducive learning environment.
- ❖ To create the center of excellence through collaborations with industries and various entities, addressing the evolving demands of society.
- ❖ To foster an environment that promotes innovation, multidisciplinary research, skill enhancement and entrepreneurship.
- ❖ To uphold and advocate for elevated standards of professional ethics and transparency.



K.S. SCHOOL OF ENGINEERING AND MANAGEMENT BENGALURU - 560109

DEPARTMENT OF COMPUTER SCIENCE AND BUSINESS SYSTEMS

**Course: MACHINE LEARNING Laboratory**

Type: Core	Course Code: BCSL606	Academic Year: 2024-2025
------------	----------------------	--------------------------

**No. of Hours per week**

Theory (LectureClass)	Practical/Field Work/Allied Activities	Total/Week	Total Number of LabContact Hours
0	0	2	20 Hours

**Marks**

Internal Assessment	Examination	Total	Credits
50	50	100	01

**Aim/Objective of the Course:**

1. To become familiar with data and visualize univariate, bivariate, and multivariate data using statistical techniques and dimensionality reduction.
2. To understand various machine learning algorithms such as similarity-based learning, regression, decision trees, and clustering.
3. To familiarize with learning theories, probability-based models and developing the skills required for decision-making in dynamic environments.

## Course Outcomes & CO-PO Mapping

Subject	Machine Learning Laboratory	BCSL606
<b>COURSE OUTCOMES</b>		
CO No.	On completion of this course, students will be able to:	RBT Level / Cognitive Level
BCSL606.1	To Illustrate the principles of multivariate data and apply dimensionality reduction techniques.	Applying (K3)
BCSL606.2	To Demonstrate similarity-based learning methods and perform regression analysis.	Applying (K3)
BCSL606.3	To Develop decision trees for classification and regression problems.	Applying (K3)
BCSL606.4	<b>Illustrate</b> Bayesian models for probabilistic learning.	Applying (K3)
BCSL606.5	To Implement the clustering algorithms to share computing resources.	Applying (K3)

### CO-PO-PSO MAPPING

CO No.	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO 9	P O 10	PO11	PO12	PS O 1	PS O 2
BCSL606.1	3	3	3	3	1	1	1	1	1	1	1	2	3	2
BCSL606.2	3	3	3	3	1	1	1	1	1	1	1	2	3	2
BCSL606.3	3	3	3	3	1	1	1	1	1	1	1	2	3	2
BCSL606.4	3	3	3	3	1	1	1	1	1	1	1	2	3	2
BCSL606.5	3	3	3	3	1	1	1	1	1	1	1	2	3	2

<b>Machine Learning lab</b>		Semester	<b>6</b>
Course Code	BCSL606	CIE Marks	50
Teaching Hours/Week (L:T:P: S)	0:0:2:0	SEE Marks	50
Credits	01	Exam Hours	100
Examination type (SEE)	Practical		

**Course objectives:**

- To become familiar with data and visualize univariate, bivariate, and multivariate data using statistical techniques and dimensionality reduction.
- To understand various machine learning algorithms such as similarity-based learning, regression, decision trees, and clustering.
- To familiarize with learning theories, probability-based models and developing the skills required for decision-making in dynamic environments.

<b>Sl.NO</b>	<b>Experiments</b>
1	Develop a program to create histograms for all numerical features and analyze the distribution of each feature. Generate box plots for all numerical features and identify any outliers. Use California Housing dataset.  <b>Book 1: Chapter 2</b>
2	Develop a program to Compute the correlation matrix to understand the relationships between pairs of features. Visualize the correlation matrix using a heatmap to know which variables have strong positive/negative correlations. Create a pair plot to visualize pairwise relationships between features. Use California Housing dataset.  <b>Book 1: Chapter 2</b>
3	Develop a program to implement Principal Component Analysis (PCA) for reducing the dimensionality of the Iris dataset from 4 features to 2.  <b>Book 1: Chapter 2</b>
4	For a given set of training data examples stored in a .CSV file, implement and demonstrate the Find-S algorithm to output a description of the set of all hypotheses consistent with the training examples.  <b>Book 1: Chapter 3</b>
5	Develop a program to implement k-Nearest Neighbour algorithm to classify the randomly generated 100 values of $x$ in the range of [0,1]. Perform the following based on dataset generated. <ol style="list-style-type: none"> <li>a. Label the first 50 points <math>\{x_1, \dots, x_{50}\}</math> as follows: if <math>(x_i \leq 0.5)</math>, then <math>x_i \in</math> Class1, else <math>x_i \in</math> Class1</li> <li>b. Classify the remaining points, <math>x_{51}, \dots, x_{100}</math> using KNN. Perform this for <math>k=1, 2, 3, 4, 5, 20, 30</math></li> </ol> <b>Book 2: Chapter – 2</b>

- |   |                                                                                                                                                                 |
|---|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 6 | Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs |
|---|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|

	<b>Book 1: Chapter – 4</b>
7	Develop a program to demonstrate the working of Linear Regression and Polynomial Regression. Use Boston Housing Dataset for Linear Regression and Auto MPG Dataset (for vehicle fuel efficiency prediction) for Polynomial Regression.  <b>Book 1: Chapter – 5</b>
8	Develop a program to demonstrate the working of the decision tree algorithm. Use Breast Cancer Data set for building the decision tree and apply this knowledge to classify a new sample.  <b>Book 2: Chapter – 3</b>
9	Develop a program to implement the Naive Bayesian classifier considering Olivetti Face Data set for training. Compute the accuracy of the classifier, considering a few test data sets.  <b>Book 2: Chapter – 4</b>
10	Develop a program to implement k-means clustering using Wisconsin Breast Cancer data set and visualize the clustering result.  <b>Book 2: Chapter – 4</b>
<b>Course outcomes (Course Skill Set):</b> At the end of the course the student will be able to: <ul style="list-style-type: none"><li>Illustrate the principles of multivariate data and apply dimensionality reduction techniques.</li><li>Demonstrate similarity-based learning methods and perform regression analysis.</li><li>Develop decision trees for classification and regression problems, and Bayesian models for probabilistic learning.</li><li>Implement the clustering algorithms to share computing resources.</li></ul>	
<b>Assessment Details (both CIE and SEE)</b>  The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. The minimum passing mark for the CIE is 40% of the maximum marks (20 marks out of 50) and for the SEE minimum passing mark is 35% of the maximum marks (18 out of 50 marks). A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each subject/ course if the student secures a minimum of 40% (40 marks out of 100) in the sum total of the CIE (Continuous Internal Evaluation) and SEE (Semester End Examination) taken together	
<b>Continuous Internal Evaluation (CIE):</b> CIE marks for the practical course are <b>50 Marks</b> . The split-up of CIE marks for record/ journal and test are in the ratio <b>60:40</b> . <ul style="list-style-type: none"><li>Each experiment is to be evaluated for conduction with an observation sheet and record write-up. Rubrics for the evaluation of the journal/write-up for hardware/software experiments are designed by the faculty who is handling the</li></ul>	



laboratory session and are made known to students at the beginning of the practical session.

- Record should contain all the specified experiments in the syllabus and each experiment write-up will be evaluated for 10 marks.
- Total marks scored by the students are scaled down to **30 marks** (60% of maximum marks).
- Weightage to be given for neatness and submission of record/write-up on time.
- Department shall conduct a test of 100 marks after the completion of all the experiments listed in the syllabus.
- In a test, test write-up, conduction of experiment, acceptable result, and procedural knowledge will carry a weightage of 60% and the rest 40% for viva-voce.
- The suitable rubrics can be designed to evaluate each student's performance and learning ability.
- The marks scored shall be scaled down to **20 marks** (40% of the maximum marks). The Sum of scaled-down marks scored in the report write-up/journal and marks of a test is the total CIE marks scored by the student.

#### **Semester End Evaluation (SEE):**

- SEE marks for the practical course are 50 Marks.
- SEE shall be conducted jointly by the two examiners of the same institute, examiners are appointed by the Head of the Institute.
- The examination schedule and names of examiners are informed to the university before the conduction of the examination. These practical examinations are to be conducted between the schedule mentioned in the academic calendar of the University.
- All laboratory experiments are to be included for practical examination.
- (Rubrics) Breakup of marks and the instructions printed on the cover page of the answer script to be strictly adhered to by the examiners. **OR** based on the course requirement evaluation rubrics shall be decided jointly by examiners.
- Students can pick one question (experiment) from the questions lot prepared by the examiners jointly.
- Evaluation of test write-up/ conduction procedure and result/viva will be conducted jointly by examiners.

General rubrics suggested for SEE are mentioned here, writeup-20%, Conduction procedure and result in -60%, Viva-voce 20% of maximum marks. SEE for practical shall be evaluated for 100 marks and scored marks shall be scaled down to 50 marks (however, based on course type, rubrics shall be decided by the examiners)

Change of experiment is allowed only once and 15% of Marks allotted to the procedure part are to be made zero.

The minimum duration of SEE is 02 hours

**Suggested Learning Resources:**

**Books:**

- S Sridhar and M Vijayalakshmi, “Machine Learning”, Oxford University Press, 2021.
- M N Murty and Ananthanarayana V S, “Machine Learning: Theory and Practice”, Universities Press (India) Pvt. Limited, 2024.

**Web links and Video Lectures (e-Resources):**

- [https://www.drssridhar.com/?page\\_id=1053](https://www.drssridhar.com/?page_id=1053)
- <https://www.universitiespress.com/resources?id=9789393330697>
- [https://onlinecourses.nptel.ac.in/noc23\\_cs18/preview](https://onlinecourses.nptel.ac.in/noc23_cs18/preview)

**Machine Learning Laboratory (BCSL606)**

**INDEX**

<b>SI No</b>	<b>Programs List</b>	<b>PageNo.</b>
1.	Develop a program to create histograms for all numerical features and analyze the distribution of each feature. Generate box plots for all numerical features and identify any outliers. Use California Housing dataset.	1
2.	Develop a program to Compute the correlation matrix to understand the relationships between pairs of features. Visualize the correlation matrix using a heatmap to know which variables have strong positive/negative correlations. Create a pair plot to visualize pairwise relationships between features. Use California Housing dataset.	4
3.	Develop a program to implement Principal Component Analysis (PCA) for reducing the dimensionality of the Iris dataset from 4 features to 2.	7
4.	For a given set of training data examples stored in a .CSV file, implement and demonstrate the Find-S algorithm to output a description of the set of all hypotheses consistent with the training examples.	9
5.	Develop a program to implement k-Nearest Neighbour algorithm to classify the randomly generated 100 values of $x$ in the range of [0,1]. Perform the following based on dataset generated. <ol style="list-style-type: none"> <li>Label the first 50 points <math>\{x_1, \dots, x_{50}\}</math> as follows: if <math>(x_i \leq 0.5)</math>, then <math>x_i \in \text{Class1}</math>, else <math>x_i \in \text{Class2}</math></li> <li>Classify the remaining points, <math>x_{51}, \dots, x_{100}</math> using KNN. Perform this for <math>k=1, 2, 3, 4, 5, 20, 30</math></li> </ol>	10
6.	Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs	18
7.	Develop a program to demonstrate the working of Linear Regression and Polynomial Regression. Use Boston Housing Dataset for Linear Regression and Auto MPG Dataset (for vehicle fuel efficiency prediction) for Polynomial Regression.	23
8.	Develop a program to demonstrate the working of the decision tree algorithm. Use Breast Cancer Data set for building the decision tree and apply this knowledge to classify a new sample.	26
9.	Develop a program to implement the Naive Bayesian classifier considering Olivetti Face Data set for training. Compute the accuracy of the classifier, considering a few test data sets.	28
10.	Develop a program to implement k-means clustering using Wisconsin Breast Cancer data set and visualize the clustering result.	31
11.	VIVA- QUESTIONS	34

## Experiment 1

**Develop a program to create histograms for all numerical features and analyze the distribution of each feature. Generate box plots for all numerical features and identify any outliers. Use California Housing dataset.**

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_california_housing

# Step 1: Load the California Housing dataset
data = fetch_california_housing(as_frame=True)
housing_df = data.frame

# Step 2: Create histograms for numerical features
numerical_features = housing_df.select_dtypes(include=[np.number]).columns

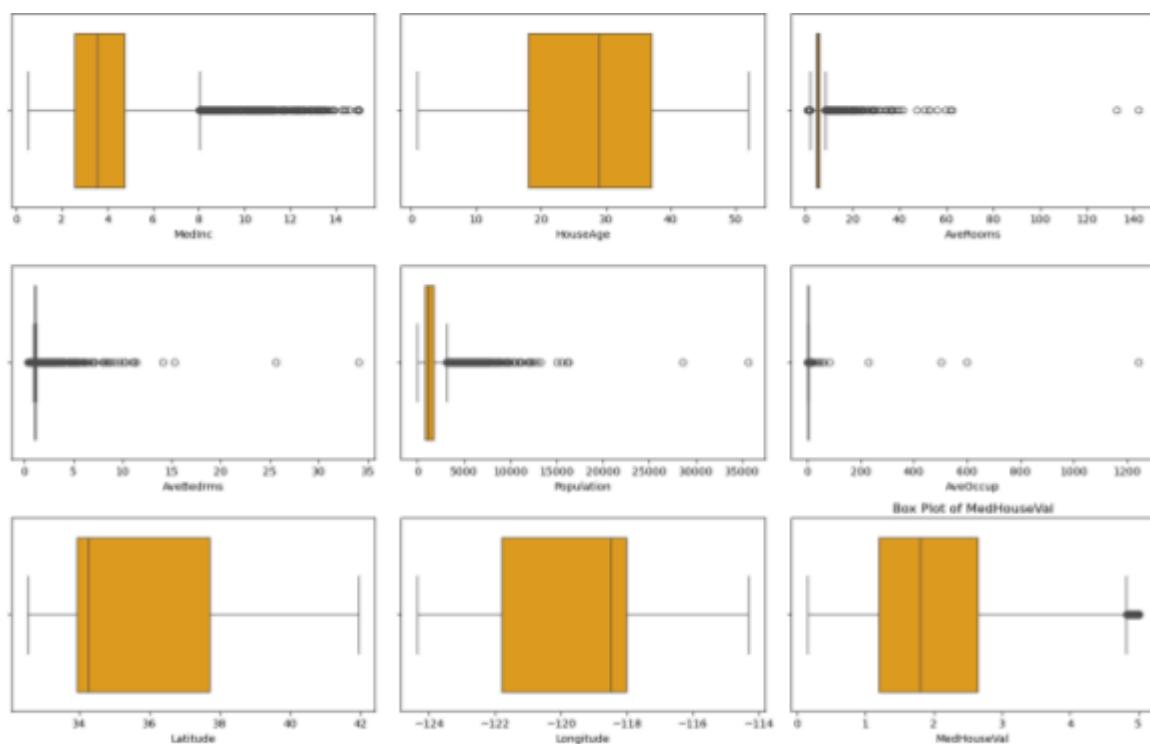
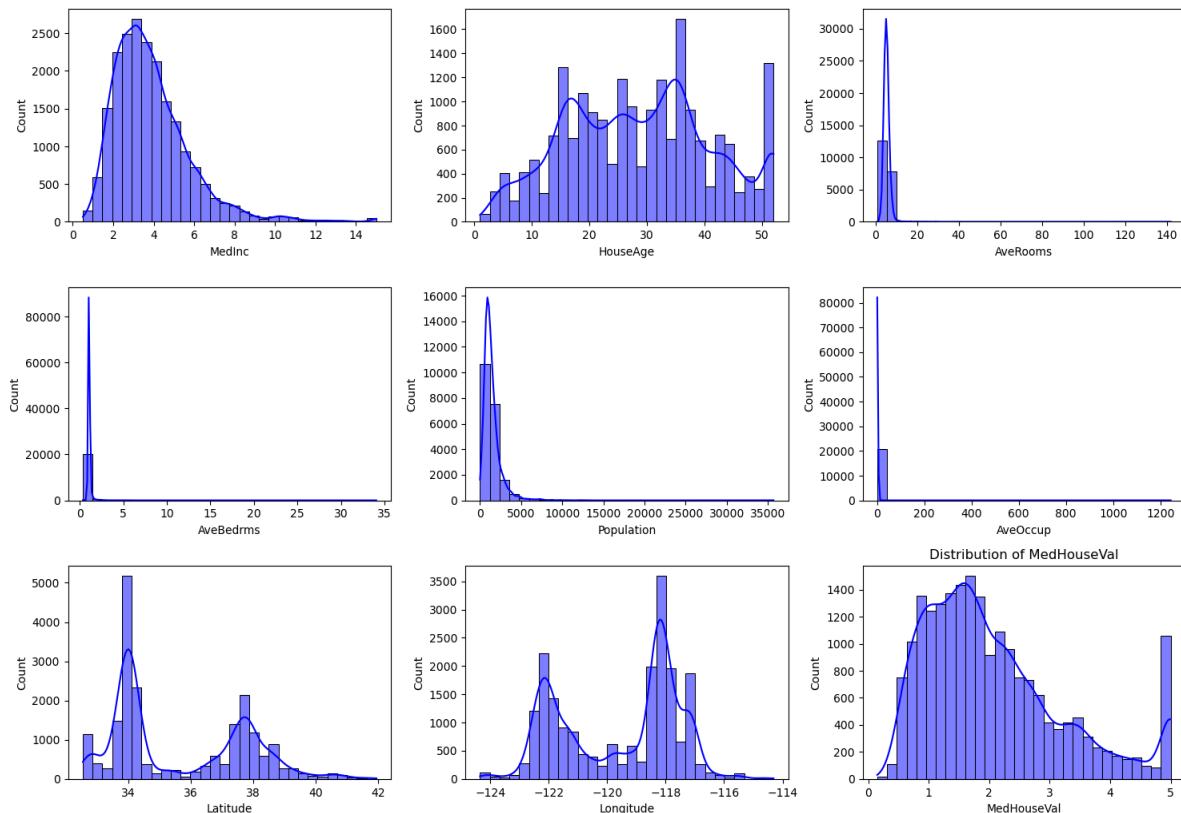
# Plot histograms
plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features):
    plt.subplot(3, 3, i + 1)
    sns.histplot(housing_df[feature], kde=True, bins=30, color='blue')
    plt.title(f'Distribution of {feature}')
plt.tight_layout()
plt.show()

# Step 3: Generate box plots for numerical features
plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features):
    plt.subplot(3, 3, i + 1)
    sns.boxplot(x=housing_df[feature], color='orange')
    plt.title(f'Box Plot of {feature}')
plt.tight_layout()
plt.show()

# Step 4: Identify outliers using the IQR method
print("Outliers Detection:")
outliers_summary = {}
for feature in numerical_features:
    Q1 = housing_df[feature].quantile(0.25)
    Q3 = housing_df[feature].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outliers = housing_df[(housing_df[feature] < lower_bound) | (housing_df[feature] > upper_bound)]
    outliers_summary[feature] = len(outliers)
    print(f'{feature}: {len(outliers)} outliers')
```

```
print(housing_df.describe())
```

## Output –





Outliers Detection:

MedInc: 681 outliers

HouseAge: 0 outliers

AveRooms: 511 outliers

AveBedrms: 1424 outliers

Population: 1196 outliers

AveOccup: 711 outliers

Latitude: 0 outliers

Longitude: 0 outliers

MedHouseVal: 1071 outliers

Dataset Summary:

	MedInc	HouseAge	...	Longitude	MedHouseVal
count	20640.000000	20640.000000	...	20640.000000	20640.000000
mean	3.870671	28.639486	...	-119.569704	2.068558
std	1.899822	12.585558	...	2.003532	1.153956
min	0.499900	1.000000	...	-124.350000	0.149990
25%	2.563400	18.000000	...	-121.800000	1.196000
50%	3.534800	29.000000	...	-118.490000	1.797000
75%	4.743250	37.000000	...	-118.010000	2.647250
max	15.000100	52.000000	...	-114.310000	5.000010

[8 rows x 9 columns]

## Experiment 2

**Develop a program to Compute the correlation matrix to understand the relationships between pairs of features. Visualize the correlation matrix using a heatmap to know which variables have strong positive/negative correlations. Create a pair plot to visualize pairwise relationships between features. Use California Housing dataset.**

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_california_housing

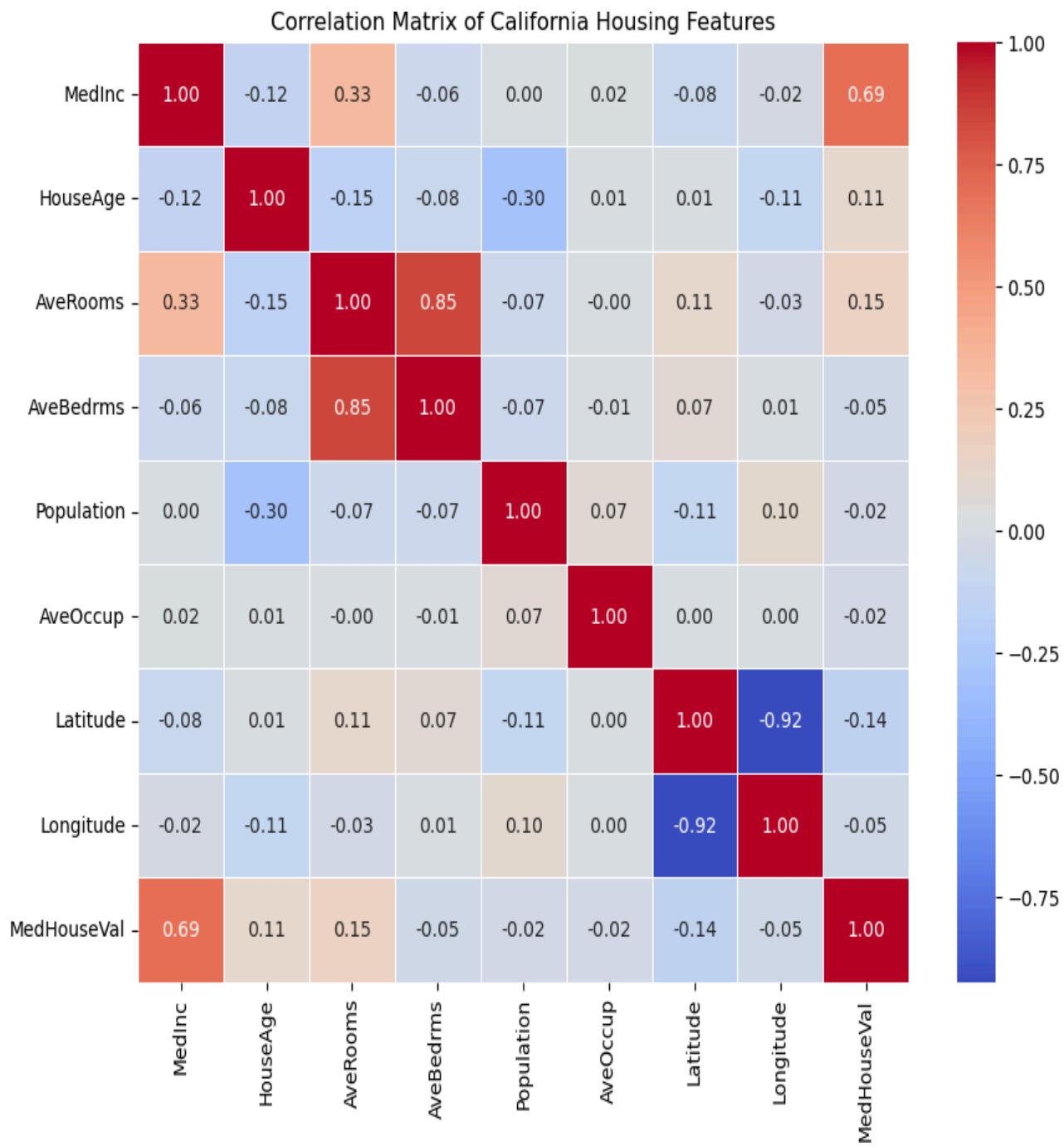
# Step 1: Load the California Housing Dataset
california_data = fetch_california_housing(as_frame=True)
data = california_data.frame

# Step 2: Compute the correlation matrix
correlation_matrix = data.corr()

# Step 3: Visualize the correlation matrix using a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Correlation Matrix of California Housing Features')
plt.show()

# Step 4: Create a pair plot to visualize pairwise relationships
sns.pairplot(data, diag_kind='kde', plot_kws={'alpha': 0.5})
plt.suptitle('Pair Plot of California Housing Features', y=1.02)
plt.show()
```

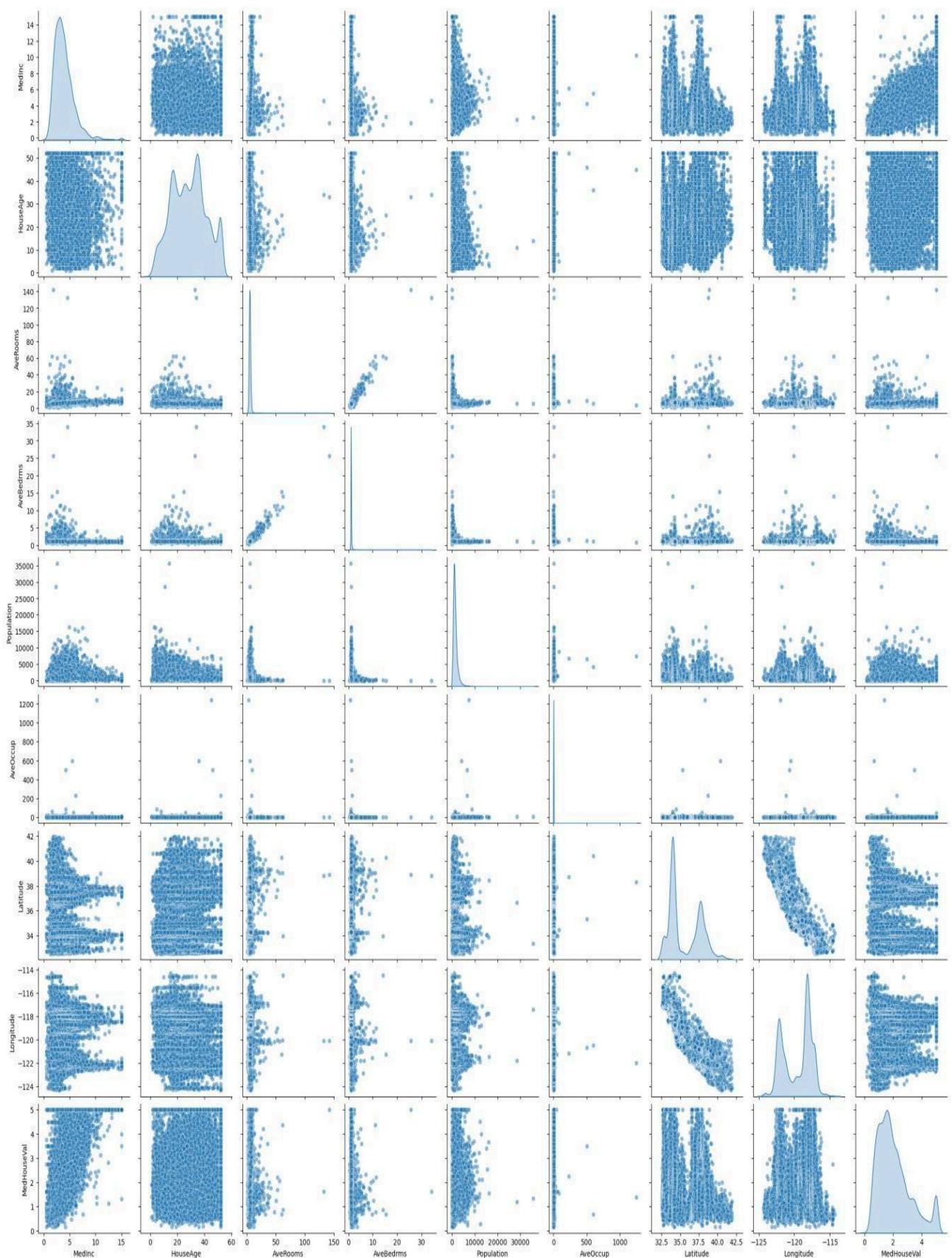
## OUTPUT:





# MACHINE LEARNING LABORATORY (BCSL606)

Pair Plot of California Housing Features



### Experiment 3

**Develop a program to implement Principal Component Analysis (PCA) for reducing the dimensionality of the Iris dataset from 4 features to 2.**

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Load the Iris dataset
iris = load_iris()
data = iris.data
labels = iris.target
label_names = iris.target_names

# Convert to a DataFrame for better visualization
iris_df = pd.DataFrame(data, columns=iris.feature_names)

# Perform PCA to reduce dimensionality to 2
pca = PCA(n_components=2)
data_reduced = pca.fit_transform(data)

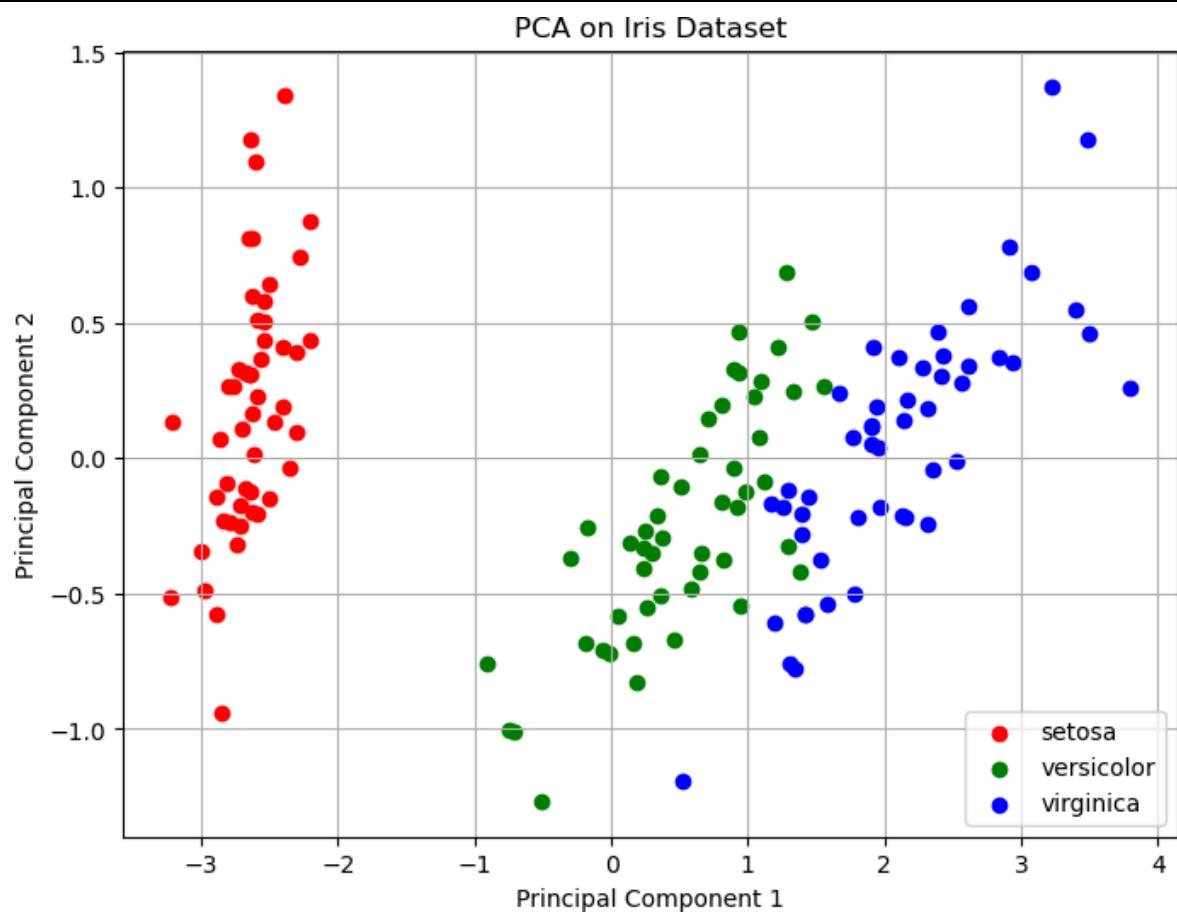
# Create a DataFrame for the reduced data
reduced_df = pd.DataFrame(data_reduced, columns=['Principal Component 1', 'Principal Component 2'])
reduced_df['Label'] = labels

# Plot the reduced data
plt.figure(figsize=(8, 6))
colors = ['r', 'g', 'b']
for i, label in enumerate(np.unique(labels)):
    plt.scatter(
        reduced_df[reduced_df['Label'] == label]['Principal Component 1'],
        reduced_df[reduced_df['Label'] == label]['Principal Component 2'],
        label=label_names[label],
        color=colors[i]
    )
```

```
plt.title('PCA on Iris Dataset')
```

```
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend()
plt.grid()
plt.show()
```

## OUTPUT-



2

## Experiment 4

**For a given set of training data examples stored in a .CSV file, implement and demonstrate the Find-S algorithm to output a description of the set of all hypotheses consistent with the training examples.**

```
import pandas as pd
```

```
def find_s_algorithm(file_path):
    data = pd.read_csv(file_path)

    print("Training data:")
    print(data)

    attributes = data.columns[:-1]
    class_label = data.columns[-1]

    hypothesis = ['?' for _ in attributes]

    for index, row in data.iterrows():
        if row[class_label] == 'Yes':
            for i, value in enumerate(row[attributes]):
                if hypothesis[i] == '?' or hypothesis[i] == value:
                    hypothesis[i] = value
                else:
                    hypothesis[i] = '?'

    return hypothesis
```

```
file_path = 'training_data.csv'
hypothesis = find_s_algorithm(file_path)
print("\nThe final hypothesis is:", hypothesis)OUTPUT:
```

Most Specific Hypothesis: ['Sunny' 'Warm' '?' 'Strong' '?' '?']

## **OUTPUT:**

Training data:

	Outlook	Temperature	Humidity	Windy	PlayTennis
0	Sunny	Hot	High	False	No
1	Sunny	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Rain	Cold	High	False	Yes
4	Rain	Cold	High	True	No
5	Overcast	Hot	High	True	Yes
6	Sunny	Hot	High	False	No

The final hypothesis is: ['Overcast', 'Hot', 'High', '?']

## Experiment 5

**Develop a program to implement k-Nearest Neighbor algorithm to classify the randomly generated 100 values of x in the range of [0,1]. Perform the following based on dataset generated.** a. Label the first 50 points {x<sub>1</sub>,.....,x<sub>50</sub>} as follows:  
if (x<sub>i</sub> ≤ 0.5), then x<sub>i</sub> ∈ Class1, else x<sub>i</sub> ∈ Class2 b. Classify the remaining points, x<sub>51</sub>,.....,x<sub>100</sub> using KNN. Perform this for k=1,2,3,4,5,20,30.

```
import numpy as np
import matplotlib.pyplot as plt
from collections import Counter

data = np.random.rand(100)

labels = ["Class1" if x <= 0.5 else "Class2" for x in data[:50]]

def euclidean_distance(x1, x2):
    return abs(x1 - x2)

def knn_classifier(train_data, train_labels, test_point, k):
    distances = [(euclidean_distance(test_point, train_data[i]), train_labels[i]) for i in range(len(train_data))]

    distances.sort(key=lambda x: x[0])
    k_nearest_neighbors = distances[:k]

    k_nearest_labels = [label for _, label in k_nearest_neighbors]
    return Counter(k_nearest_labels).most_common(1)[0][0]

train_data = data[:50]
train_labels = labels

test_data = data[50:]

k_values = [1, 2, 3, 4, 5, 20, 30]

print("--- k-Nearest Neighbors Classification ---")
print("Training dataset: First 50 points labeled based on the rule (x <= 0.5 -> Class1, x > 0.5 -> Class2)")
print("Testing dataset: Remaining 50 points to be classified\n")

results = {}

for k in k_values:
    print(f"Results for k = {k}:")
    classified_labels = [knn_classifier(train_data, train_labels, test_point, k) for test_point in test_data]
```

```
results[k] = classified_labels

for i, label in enumerate(classified_labels, start=51):
    print(f'Point x{i} (value: {test_data[i - 51]:.4f}) is classified as {label}')
print("\n")

print("Classification complete.\n")

for k in k_values:
    classified_labels = results[k]
    class1_points = [test_data[i] for i in range(len(test_data)) if classified_labels[i] ==
"Class1"]

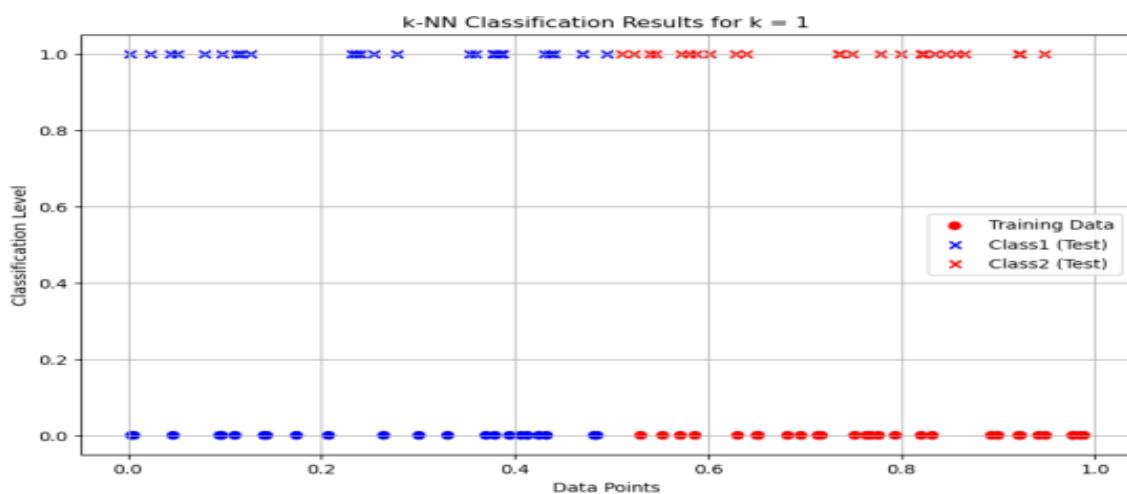
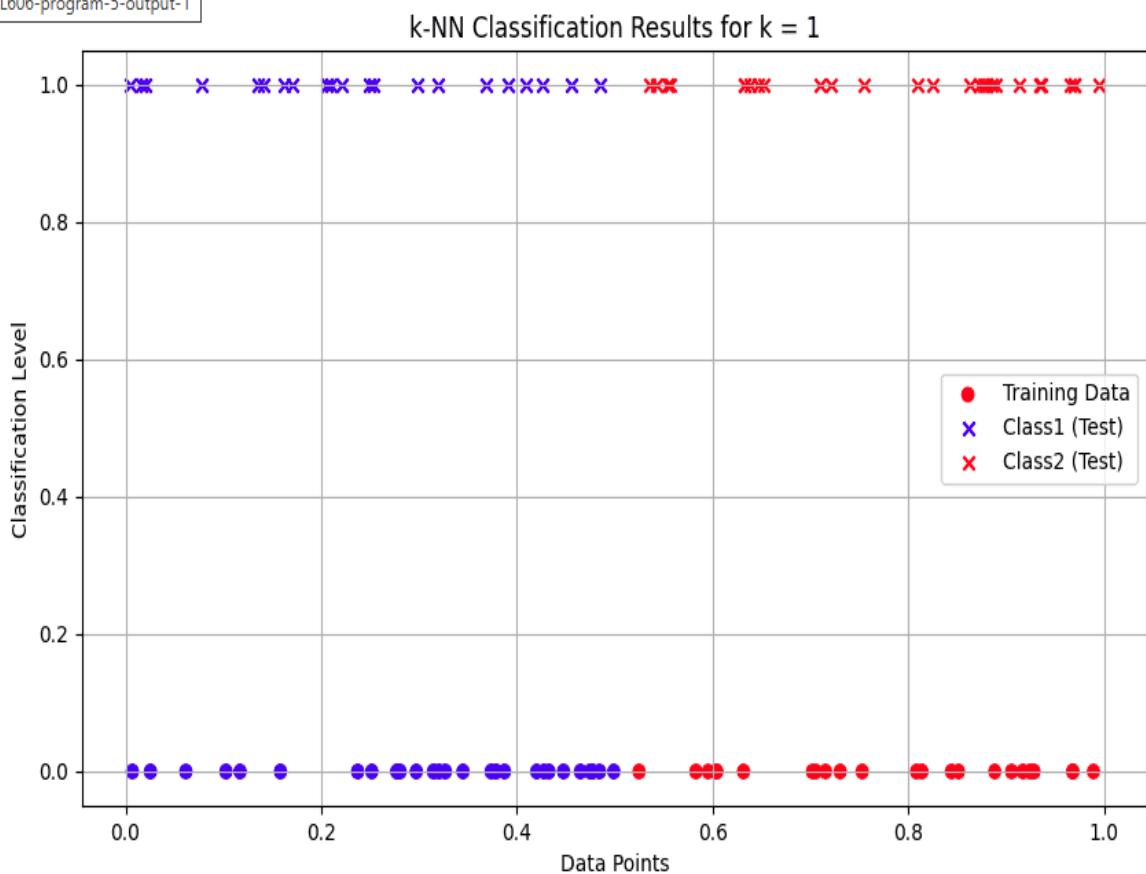
    class2_points = [test_data[i] for i in range(len(test_data)) if classified_labels[i] ==
"Class2"]

    plt.figure(figsize=(10, 6))
    plt.scatter(train_data, [0] * len(train_data), c=["blue" if label == "Class1" else "red"
for label in train_labels],
                label="Training Data", marker="o")
    plt.scatter(class1_points, [1] * len(class1_points), c="blue", label="Class1 (Test)",
marker="x")
    plt.scatter(class2_points, [1] * len(class2_points), c="red", label="Class2 (Test)",
marker="x")

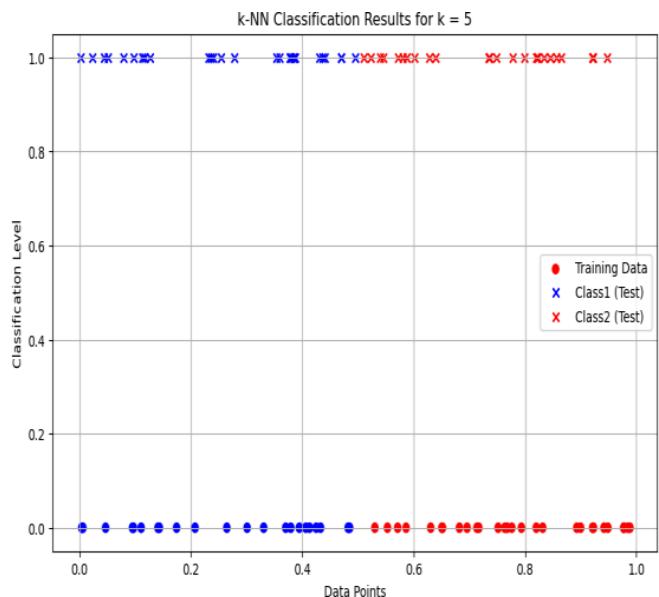
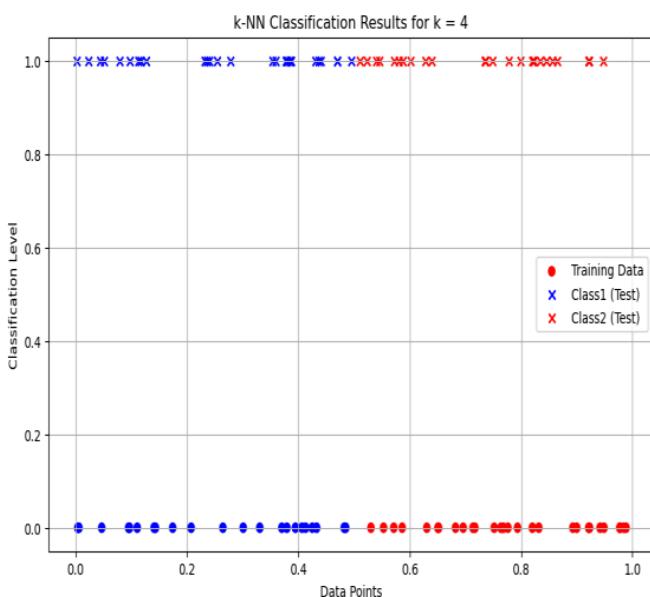
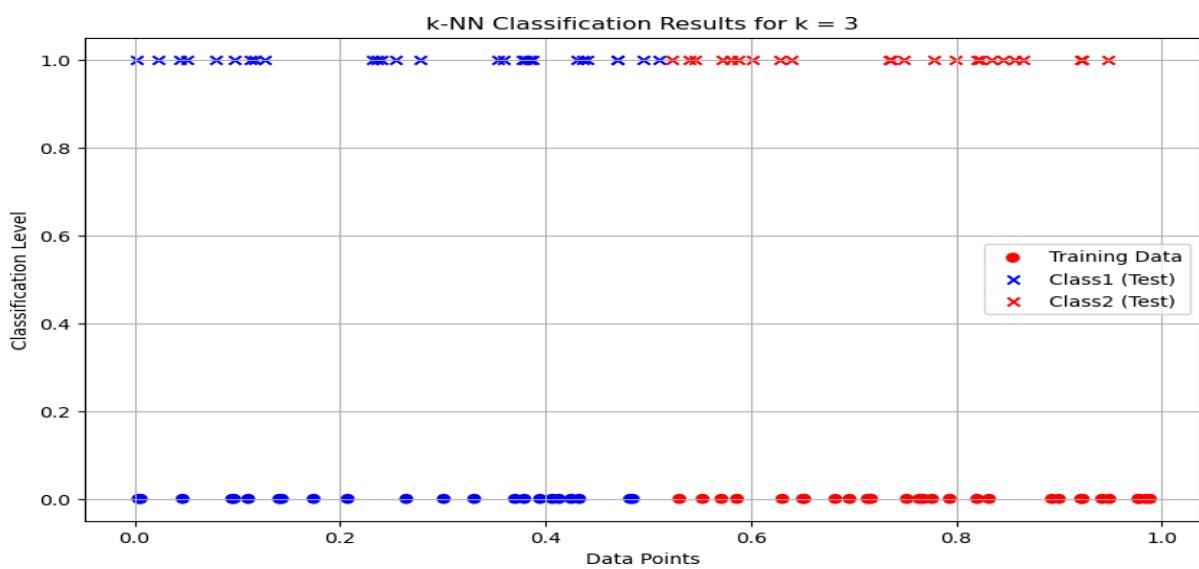
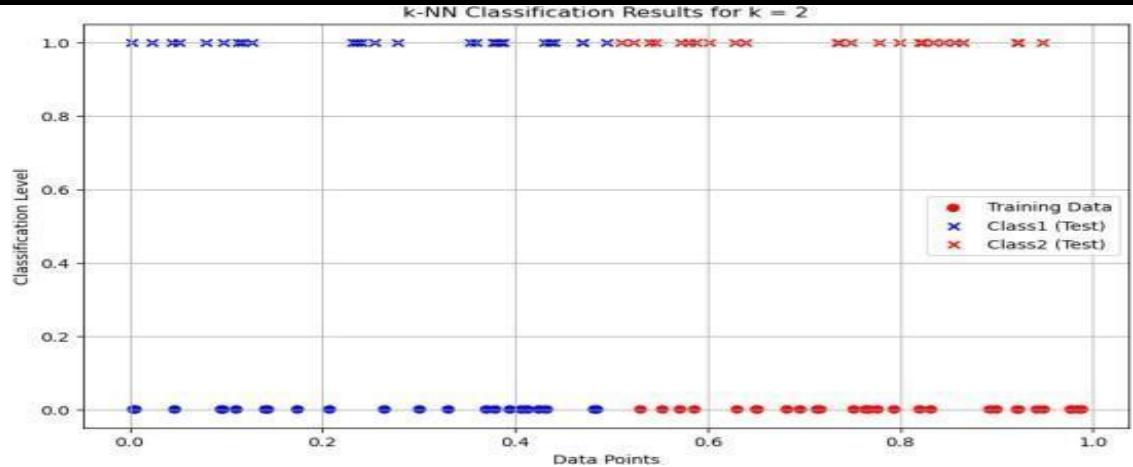
    plt.title(f"k-NN Classification Results for k = {k}")
    plt.xlabel("Data Points")
    plt.ylabel("Classification Level")
    plt.legend()
    plt.grid(True)
    plt.show()
```

**OUTPUT:**

BCSL606-program-5-output-1



## MACHINE LEARNING LABORATORY (BCSL606)



--- k-Nearest Neighbors Classification ---

Training dataset: First 50 points labeled based on the rule ( $x \leq 0.5 \rightarrow \text{Class1}$ ,  $x >$

0.5 -> Class2)

Testing dataset: Remaining 50 points to be classified

Results for k = 1:

Point x51 (value: 0.2059) is classified as Class1  
Point x52 (value: 0.2535) is classified as Class1  
Point x53 (value: 0.4856) is classified as Class1  
Point x54 (value: 0.9651) is classified as Class2  
Point x55 (value: 0.3906) is classified as Class1  
Point x56 (value: 0.8903) is classified as Class2  
Point x57 (value: 0.9695) is classified as Class2  
Point x58 (value: 0.2206) is classified as Class1  
Point x59 (value: 0.0203) is classified as Class1  
Point x60 (value: 0.1619) is classified as Class1  
Point x61 (value: 0.6461) is classified as Class2  
Point x62 (value: 0.6523) is classified as Class2

Point x63 (value: 0.8728) is classified as Class2  
Point x64 (value: 0.5435) is classified as Class2  
Point x65 (value: 0.8246) is classified as Class2  
Point x66 (value: 0.9347) is classified as Class2  
Point x67 (value: 0.5361) is classified as Class2  
Point x68 (value: 0.7215) is classified as Class2  
Point x69 (value: 0.9703) is classified as Class2  
Point x70 (value: 0.8764) is classified as Class2  
Point x71 (value: 0.7543) is classified as Class2  
Point x72 (value: 0.1406) is classified as Class1  
Point x73 (value: 0.1349) is classified as Class1  
Point x74 (value: 0.9705) is classified as Class2  
Point x75 (value: 0.2985) is classified as Class1  
Point x76 (value: 0.9948) is classified as Class2  
Point x77 (value: 0.4551) is classified as Class1  
Point x78 (value: 0.2101) is classified as Class1  
Point x79 (value: 0.5542) is classified as Class2  
Point x80 (value: 0.3202) is classified as Class1  
Point x81 (value: 0.6325) is classified as Class2  
Point x82 (value: 0.9345) is classified as Class2  
Point x83 (value: 0.0156) is classified as Class1  
Point x84 (value: 0.8859) is classified as Class2  
Point x85 (value: 0.2495) is classified as Class1  
Point x86 (value: 0.6380) is classified as Class2  
Point x87 (value: 0.7095) is classified as Class2  
Point x88 (value: 0.4259) is classified as Class1  
Point x89 (value: 0.0052) is classified as Class1  
Point x90 (value: 0.6322) is classified as Class2  
Point x91 (value: 0.1701) is classified as Class1  
Point x92 (value: 0.3693) is classified as Class1  
Point x93 (value: 0.4087) is classified as Class1  
Point x94 (value: 0.8103) is classified as Class2  
Point x95 (value: 0.0773) is classified as Class1

Point x96 (value: 0.8792) is classified as Class2

Point x97 (value: 0.9138) is classified as Class2  
Point x98 (value: 0.5567) is classified as Class2  
Point x99 (value: 0.8625) is classified as Class2  
Point x100 (value: 0.9363) is classified as Class2

Results for k = 2:

Point x51 (value: 0.2059) is classified as Class1  
Point x52 (value: 0.2535) is classified as Class1  
Point x53 (value: 0.4856) is classified as Class1  
Point x54 (value: 0.9651) is classified as Class2  
Point x55 (value: 0.3906) is classified as Class1  
Point x56 (value: 0.8903) is classified as Class2  
Point x57 (value: 0.9695) is classified as Class2  
Point x58 (value: 0.2206) is classified as Class1

Point x59 (value: 0.0203) is classified as Class1  
Point x60 (value: 0.1619) is classified as Class1  
Point x61 (value: 0.6461) is classified as Class2  
Point x62 (value: 0.6523) is classified as Class2  
Point x63 (value: 0.8728) is classified as Class2  
Point x64 (value: 0.5435) is classified as Class2  
Point x65 (value: 0.8246) is classified as Class2  
Point x66 (value: 0.9347) is classified as Class2  
Point x67 (value: 0.5361) is classified as Class2  
Point x68 (value: 0.7215) is classified as Class2  
Point x69 (value: 0.9703) is classified as Class2  
Point x70 (value: 0.8764) is classified as Class2  
Point x71 (value: 0.7543) is classified as Class2  
Point x72 (value: 0.1406) is classified as Class1  
Point x73 (value: 0.1349) is classified as Class1  
Point x74 (value: 0.9705) is classified as Class2  
Point x75 (value: 0.2985) is classified as Class1  
Point x76 (value: 0.9948) is classified as Class2  
Point x77 (value: 0.4551) is classified as Class1  
Point x78 (value: 0.2101) is classified as Class1  
Point x79 (value: 0.5542) is classified as Class2  
Point x80 (value: 0.3202) is classified as Class1  
Point x81 (value: 0.6325) is classified as Class2  
Point x82 (value: 0.9345) is classified as Class2  
Point x83 (value: 0.0156) is classified as Class1  
Point x84 (value: 0.8859) is classified as Class2  
Point x85 (value: 0.2495) is classified as Class1  
Point x86 (value: 0.6380) is classified as Class2  
Point x87 (value: 0.7095) is classified as Class2  
Point x88 (value: 0.4259) is classified as Class1  
Point x89 (value: 0.0052) is classified as Class1  
Point x90 (value: 0.6322) is classified as Class2  
Point x91 (value: 0.1701) is classified as Class1  
Point x92 (value: 0.3693) is classified as Class1  
Point x93 (value: 0.4087) is classified as Class1

Point x94 (value: 0.8103) is classified as Class2  
Point x95 (value: 0.0773) is classified as Class1  
Point x96 (value: 0.8792) is classified as Class2  
Point x97 (value: 0.9138) is classified as Class2  
Point x98 (value: 0.5567) is classified as Class2  
Point x99 (value: 0.8625) is classified as Class2  
Point x100 (value: 0.9363) is classified as Class2

Results for k = 3:

Point x51 (value: 0.2059) is classified as Class1  
Point x52 (value: 0.2535) is classified as Class1  
Point x53 (value: 0.4856) is classified as Class1

Point x54 (value: 0.9651) is classified as Class2  
Point x55 (value: 0.3906) is classified as Class1  
Point x56 (value: 0.8903) is classified as Class2  
Point x57 (value: 0.9695) is classified as Class2  
Point x58 (value: 0.2206) is classified as Class1  
Point x59 (value: 0.0203) is classified as Class1  
Point x60 (value: 0.1619) is classified as Class1  
Point x61 (value: 0.6461) is classified as Class2  
Point x62 (value: 0.6523) is classified as Class2  
Point x63 (value: 0.8728) is classified as Class2  
Point x64 (value: 0.5435) is classified as Class2  
Point x65 (value: 0.8246) is classified as Class2  
Point x66 (value: 0.9347) is classified as Class2  
Point x67 (value: 0.5361) is classified as Class2  
Point x68 (value: 0.7215) is classified as Class2  
Point x69 (value: 0.9703) is classified as Class2  
Point x70 (value: 0.8764) is classified as Class2  
Point x71 (value: 0.7543) is classified as Class2  
Point x72 (value: 0.1406) is classified as Class1  
Point x73 (value: 0.1349) is classified as Class1  
Point x74 (value: 0.9705) is classified as Class2  
Point x75 (value: 0.2985) is classified as Class1  
Point x76 (value: 0.9948) is classified as Class2  
Point x77 (value: 0.4551) is classified as Class1  
Point x78 (value: 0.2101) is classified as Class1  
Point x79 (value: 0.5542) is classified as Class2  
Point x80 (value: 0.3202) is classified as Class1  
Point x81 (value: 0.6325) is classified as Class2  
Point x82 (value: 0.9345) is classified as Class2  
Point x83 (value: 0.0156) is classified as Class1  
Point x84 (value: 0.8859) is classified as Class2  
Point x85 (value: 0.2495) is classified as Class1  
Point x86 (value: 0.6380) is classified as Class2  
Point x87 (value: 0.7095) is classified as Class2  
Point x88 (value: 0.4259) is classified as Class1  
Point x89 (value: 0.0052) is classified as Class1

Point x90 (value: 0.6322) is classified as Class2

Point x91 (value: 0.1701) is classified as Class1  
Point x92 (value: 0.3693) is classified as Class1  
Point x93 (value: 0.4087) is classified as Class1  
Point x94 (value: 0.8103) is classified as Class2  
Point x95 (value: 0.0773) is classified as Class1  
Point x96 (value: 0.8792) is classified as Class2  
Point x97 (value: 0.9138) is classified as Class2  
Point x98 (value: 0.5567) is classified as Class2  
Point x99 (value: 0.8625) is classified as Class2  
Point x100 (value: 0.9363) is classified as Class2

Results for k = 4:

Point x51 (value: 0.2059) is classified as Class1  
Point x52 (value: 0.2535) is classified as Class1  
Point x53 (value: 0.4856) is classified as Class1  
Point x54 (value: 0.9651) is classified as Class2  
Point x55 (value: 0.3906) is classified as Class1  
Point x56 (value: 0.8903) is classified as Class2

Point x57 (value: 0.9695) is classified as Class2  
Point x58 (value: 0.2206) is classified as Class1  
Point x59 (value: 0.0203) is classified as Class1  
Point x60 (value: 0.1619) is classified as Class1  
Point x61 (value: 0.6461) is classified as Class2  
Point x62 (value: 0.6523) is classified as Class2  
Point x63 (value: 0.8728) is classified as Class2  
Point x64 (value: 0.5435) is classified as Class2  
Point x65 (value: 0.8246) is classified as Class2  
Point x66 (value: 0.9347) is classified as Class2  
Point x67 (value: 0.5361) is classified as Class2  
Point x68 (value: 0.7215) is classified as Class2  
Point x69 (value: 0.9703) is classified as Class2  
Point x70 (value: 0.8764) is classified as Class2  
Point x71 (value: 0.7543) is classified as Class2  
Point x72 (value: 0.1406) is classified as Class1  
Point x73 (value: 0.1349) is classified as Class1  
Point x74 (value: 0.9705) is classified as Class2  
Point x75 (value: 0.2985) is classified as Class1  
Point x76 (value: 0.9948) is classified as Class2  
Point x77 (value: 0.4551) is classified as Class1  
Point x78 (value: 0.2101) is classified as Class1  
Point x79 (value: 0.5542) is classified as Class2  
Point x80 (value: 0.3202) is classified as Class1  
Point x81 (value: 0.6325) is classified as Class2  
Point x82 (value: 0.9345) is classified as Class2  
Point x83 (value: 0.0156) is classified as Class1  
Point x84 (value: 0.8859) is classified as Class2

Point x85 (value: 0.2495) is classified as Class1  
Point x86 (value: 0.6380) is classified as Class2

Point x87 (value: 0.7095) is classified as Class2  
Point x88 (value: 0.4259) is classified as Class1  
Point x89 (value: 0.0052) is classified as Class1  
Point x90 (value: 0.6322) is classified as Class2  
Point x91 (value: 0.1701) is classified as Class1  
Point x92 (value: 0.3693) is classified as Class1  
Point x93 (value: 0.4087) is classified as Class1  
Point x94 (value: 0.8103) is classified as Class2  
Point x95 (value: 0.0773) is classified as Class1  
Point x96 (value: 0.8792) is classified as Class2  
Point x97 (value: 0.9138) is classified as Class2

Point x98 (value: 0.5567) is classified as Class2  
Point x99 (value: 0.8625) is classified as Class2  
Point x100 (value: 0.9363) is classified as Class2

Results for k = 5:

Point x51 (value: 0.2059) is classified as Class1  
Point x52 (value: 0.2535) is classified as Class1  
Point x53 (value: 0.4856) is classified as Class1  
Point x54 (value: 0.9651) is classified as Class2  
Point x55 (value: 0.3906) is classified as Class1  
Point x56 (value: 0.8903) is classified as Class2  
Point x57 (value: 0.9695) is classified as Class2  
Point x58 (value: 0.2206) is classified as Class1  
Point x59 (value: 0.0203) is classified as Class1

Point x60 (value: 0.1619) is classified as Class1  
Point x61 (value: 0.6461) is classified as Class2  
Point x62 (value: 0.6523) is classified as Class2  
Point x63 (value: 0.8728) is classified as Class2  
Point x64 (value: 0.5435) is classified as Class2  
Point x65 (value: 0.8246) is classified as Class2  
Point x66 (value: 0.9347) is classified as Class2  
Point x67 (value: 0.5361) is classified as Class1  
Point x68 (value: 0.7215) is classified as Class2  
Point x69 (value: 0.9703) is classified as Class2  
Point x70 (value: 0.8764) is classified as Class2  
Point x71 (value: 0.7543) is classified as Class2  
Point x72 (value: 0.1406) is classified as Class1  
Point x73 (value: 0.1349) is classified as Class1  
Point x74 (value: 0.9705) is classified as Class2  
Point x75 (value: 0.2985) is classified as Class1  
Point x76 (value: 0.9948) is classified as Class2  
Point x77 (value: 0.4551) is classified as Class1  
Point x78 (value: 0.2101) is classified as Class1

Point x79 (value: 0.5542) is classified as Class2  
Point x80 (value: 0.3202) is classified as Class1  
Point x81 (value: 0.6325) is classified as Class2  
Point x82 (value: 0.9345) is classified as Class2

Point x83 (value: 0.0156) is classified as Class1  
Point x84 (value: 0.8859) is classified as Class2  
Point x85 (value: 0.2495) is classified as Class1  
Point x86 (value: 0.6380) is classified as Class2  
Point x87 (value: 0.7095) is classified as Class2  
Point x88 (value: 0.4259) is classified as Class1  
Point x89 (value: 0.0052) is classified as Class1  
Point x90 (value: 0.6322) is classified as Class2  
Point x91 (value: 0.1701) is classified as Class1

Point x92 (value: 0.3693) is classified as Class1  
Point x93 (value: 0.4087) is classified as Class1  
Point x94 (value: 0.8103) is classified as Class2  
Point x95 (value: 0.0773) is classified as Class1  
Point x96 (value: 0.8792) is classified as Class2  
Point x97 (value: 0.9138) is classified as Class2  
Point x98 (value: 0.5567) is classified as Class2  
Point x99 (value: 0.8625) is classified as Class2  
Point x100 (value: 0.9363) is classified as Class2

Results for k = 20:

Point x51 (value: 0.2059) is classified as Class1  
Point x52 (value: 0.2535) is classified as Class1  
Point x53 (value: 0.4856) is classified as Class1  
Point x54 (value: 0.9651) is classified as Class2  
Point x55 (value: 0.3906) is classified as Class1  
Point x56 (value: 0.8903) is classified as Class2  
Point x57 (value: 0.9695) is classified as Class2  
Point x58 (value: 0.2206) is classified as Class1  
Point x59 (value: 0.0203) is classified as Class1  
Point x60 (value: 0.1619) is classified as Class1  
Point x61 (value: 0.6461) is classified as Class2  
Point x62 (value: 0.6523) is classified as Class2

Point x63 (value: 0.8728) is classified as Class2  
Point x64 (value: 0.5435) is classified as Class1  
Point x65 (value: 0.8246) is classified as Class2  
Point x66 (value: 0.9347) is classified as Class2  
Point x67 (value: 0.5361) is classified as Class1  
Point x68 (value: 0.7215) is classified as Class2  
Point x69 (value: 0.9703) is classified as Class2  
Point x70 (value: 0.8764) is classified as Class2  
Point x71 (value: 0.7543) is classified as Class2  
Point x72 (value: 0.1406) is classified as Class1

Point x73 (value: 0.1349) is classified as Class1  
Point x74 (value: 0.9705) is classified as Class2  
Point x75 (value: 0.2985) is classified as Class1  
Point x76 (value: 0.9948) is classified as Class2  
Point x77 (value: 0.4551) is classified as Class1  
Point x78 (value: 0.2101) is classified as Class1

Point x79 (value: 0.5542) is classified as Class1  
Point x80 (value: 0.3202) is classified as Class1  
Point x81 (value: 0.6325) is classified as Class2  
Point x82 (value: 0.9345) is classified as Class2  
Point x83 (value: 0.0156) is classified as Class1  
Point x84 (value: 0.8859) is classified as Class2  
Point x85 (value: 0.2495) is classified as Class1

Point x86 (value: 0.6380) is classified as Class2  
Point x87 (value: 0.7095) is classified as Class2  
Point x88 (value: 0.4259) is classified as Class1  
Point x89 (value: 0.0052) is classified as Class1  
Point x90 (value: 0.6322) is classified as Class2  
Point x91 (value: 0.1701) is classified as Class1  
Point x92 (value: 0.3693) is classified as Class1  
Point x93 (value: 0.4087) is classified as Class1  
Point x94 (value: 0.8103) is classified as Class2  
Point x95 (value: 0.0773) is classified as Class1  
Point x96 (value: 0.8792) is classified as Class2  
Point x97 (value: 0.9138) is classified as Class2  
Point x98 (value: 0.5567) is classified as Class2  
Point x99 (value: 0.8625) is classified as Class2  
Point x100 (value: 0.9363) is classified as Class2

Results for k = 30:

Point x51 (value: 0.2059) is classified as Class1  
Point x52 (value: 0.2535) is classified as Class1  
Point x53 (value: 0.4856) is classified as Class1  
Point x54 (value: 0.9651) is classified as Class2  
Point x55 (value: 0.3906) is classified as Class1  
Point x56 (value: 0.8903) is classified as Class2  
Point x57 (value: 0.9695) is classified as Class2  
Point x58 (value: 0.2206) is classified as Class1  
Point x59 (value: 0.0203) is classified as Class1  
Point x60 (value: 0.1619) is classified as Class1  
Point x61 (value: 0.6461) is classified as Class2  
Point x62 (value: 0.6523) is classified as Class2  
Point x63 (value: 0.8728) is classified as Class2  
Point x64 (value: 0.5435) is classified as Class1  
Point x65 (value: 0.8246) is classified as Class2

Point x66 (value: 0.9347) is classified as Class2

Point x67 (value: 0.5361) is classified as Class1  
Point x68 (value: 0.7215) is classified as Class2  
Point x69 (value: 0.9703) is classified as Class2  
Point x70 (value: 0.8764) is classified as Class2  
Point x71 (value: 0.7543) is classified as Class2  
Point x72 (value: 0.1406) is classified as Class1  
Point x73 (value: 0.1349) is classified as Class1  
Point x74 (value: 0.9705) is classified as Class2

Point x75 (value: 0.2985) is classified as Class1  
Point x76 (value: 0.9948) is classified as Class2  
Point x77 (value: 0.4551) is classified as Class1  
Point x78 (value: 0.2101) is classified as Class1  
Point x79 (value: 0.5542) is classified as Class1

Point x80 (value: 0.3202) is classified as Class1  
Point x81 (value: 0.6325) is classified as Class2  
Point x82 (value: 0.9345) is classified as Class2  
Point x83 (value: 0.0156) is classified as Class1  
Point x84 (value: 0.8859) is classified as Class2  
Point x85 (value: 0.2495) is classified as Class1  
Point x86 (value: 0.6380) is classified as Class2  
Point x87 (value: 0.7095) is classified as Class2  
Point x88 (value: 0.4259) is classified as Class1  
Point x89 (value: 0.0052) is classified as Class1  
Point x90 (value: 0.6322) is classified as Class2  
Point x91 (value: 0.1701) is classified as Class1  
Point x92 (value: 0.3693) is classified as Class1  
Point x93 (value: 0.4087) is classified as Class1  
Point x94 (value: 0.8103) is classified as Class2  
Point x95 (value: 0.0773) is classified as Class1  
Point x96 (value: 0.8792) is classified as Class2  
Point x97 (value: 0.9138) is classified as Class2  
Point x98 (value: 0.5567) is classified as Class1  
Point x99 (value: 0.8625) is classified as Class2  
Point x100 (value: 0.9363) is classified as Class2

Classification complete.

## Experiment 6

**Implement the non-parametric Locally Weighted Regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs.**

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

def gaussian_kernel(x, xi, tau):
    return np.exp(-np.sum((x - xi) ** 2) / (2 * tau ** 2))

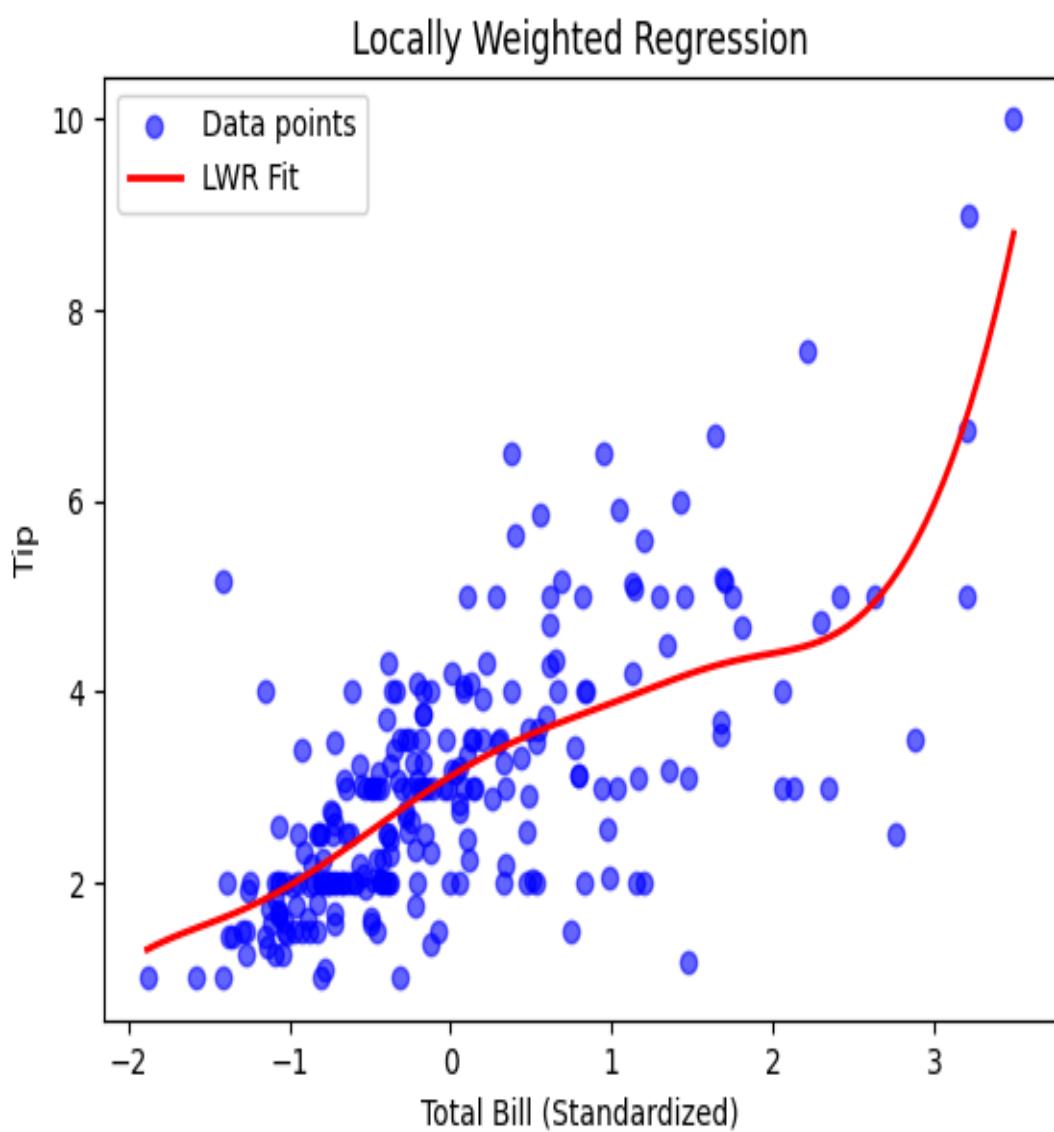
def locally_weighted_regression(x, X, y, tau):
    m = X.shape[0]
    weights = np.array([gaussian_kernel(x, X[i], tau) for i in range(m)])
    W = np.diag(weights)
    X_transpose_W = X.T @ W
    theta = np.linalg.inv(X_transpose_W @ X) @ X_transpose_W @ y
    return x @ theta

np.random.seed(42)
X = np.linspace(0, 2 * np.pi, 100)
y = np.sin(X) + 0.1 * np.random.randn(100)
X_bias = np.c_[np.ones(X.shape), X]

x_test = np.linspace(0, 2 * np.pi, 200)
x_test_bias = np.c_[np.ones(x_test.shape), x_test]
tau = 0.5
y_pred = np.array([locally_weighted_regression(xi, X_bias, y, tau) for xi in x_test_bias])

plt.figure(figsize=(10, 6))
plt.scatter(X, y, color='red', label='Training Data', alpha=0.7)
plt.plot(x_test, y_pred, color='blue', label=f'LWR Fit (tau={tau})', linewidth=2)
plt.xlabel('X', fontsize=12)
plt.ylabel('y', fontsize=12)
plt.title('Locally Weighted Regression', fontsize=14)
plt.legend(fontsize=10)
plt.grid(alpha=0.3)
plt.show()
```

**OUTPUT:**





## Experiment 7

**Develop a program to demonstrate the working of Linear Regression and Polynomial Regression. Use Boston Housing Dataset for Linear Regression and Auto MPG Dataset (for vehicle fuel efficiency prediction) for Polynomial Regression.**

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures, StandardScaler
from sklearn.pipeline import make_pipeline
from sklearn.metrics import mean_squared_error, r2_score

def linear_regression_california():
    housing = fetch_california_housing(as_frame=True)
    X = housing.data[["AveRooms"]]
    y = housing.target

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

    model = LinearRegression()
    model.fit(X_train, y_train)

    y_pred = model.predict(X_test)

    plt.scatter(X_test, y_test, color="blue", label="Actual")
    plt.plot(X_test, y_pred, color="red", label="Predicted")
    plt.xlabel("Average number of rooms (AveRooms)")
    plt.ylabel("Median value of homes ($100,000)")
    plt.title("Linear Regression - California Housing Dataset")
    plt.legend()
    plt.show()

    print("Linear Regression - California Housing Dataset")
    print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
    print("R^2 Score:", r2_score(y_test, y_pred))

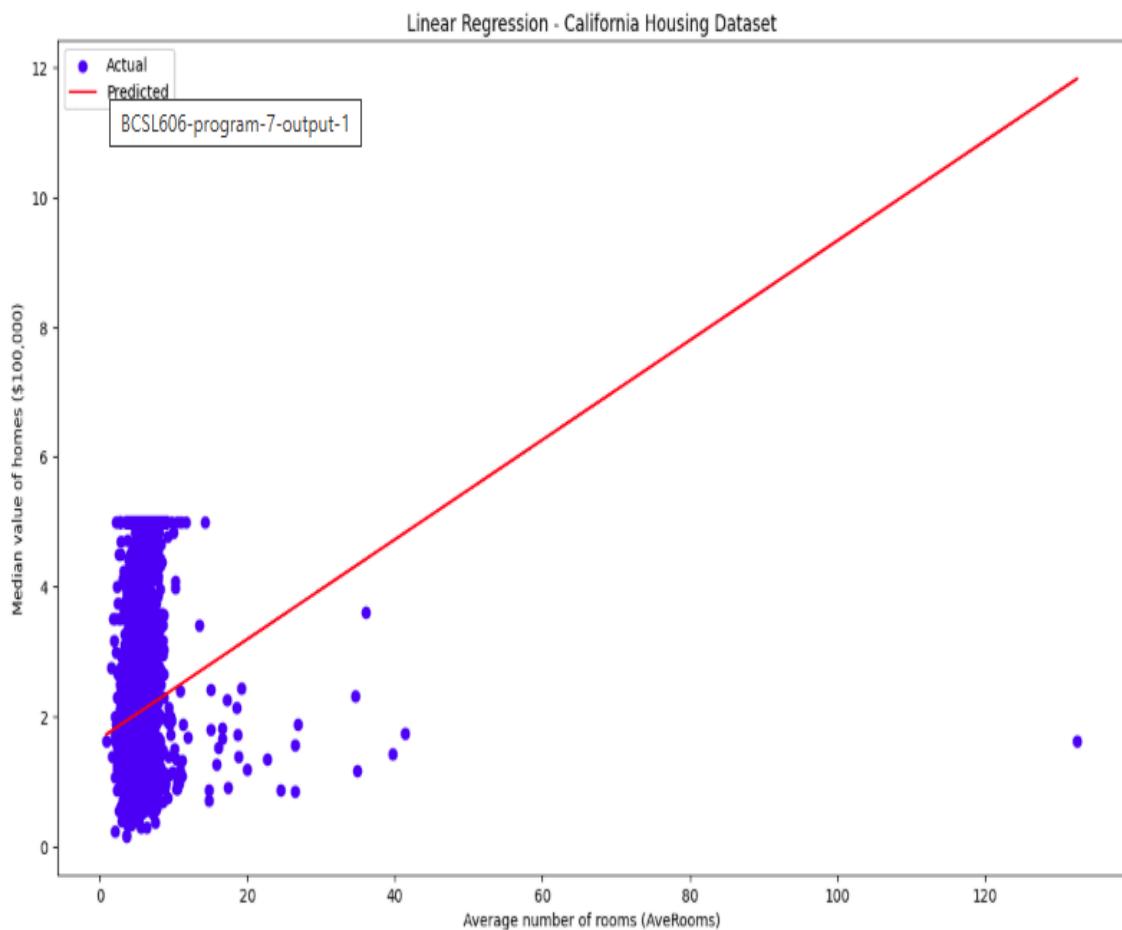
def polynomial_regression_auto_mpg():
    url =
"https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.da
ta"
    column_names = ["mpg", "cylinders", "displacement", "horsepower", "weight",
"acceleration", "model_year", "origin"]
    data = pd.read_csv(url, sep='\s+', names=column_names, na_values="?")
    data = data.dropna()

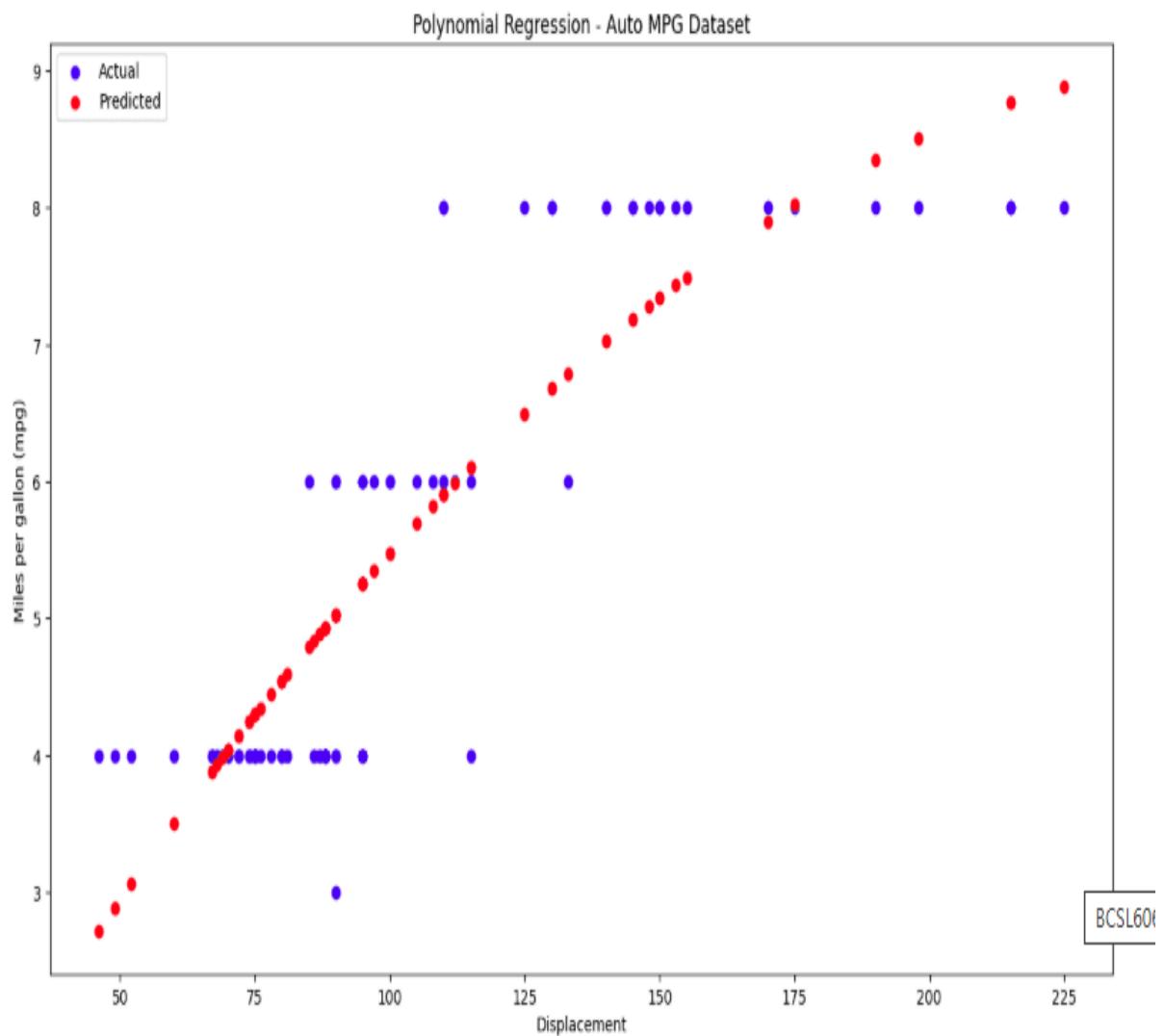
    X = data["displacement"].values.reshape(-1, 1)
    y = data["mpg"].values

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

```
poly_model = make_pipeline(PolynomialFeatures(degree=2), StandardScaler(),  
LinearRegression())  
poly_model.fit(X_train, y_train)  
  
y_pred = poly_model.predict(X_test)  
  
plt.scatter(X_test, y_test, color="blue", label="Actual")  
  
plt.scatter(X_test, y_pred, color="red", label="Predicted")  
  
plt.xlabel("Displacement")  
plt.ylabel("Miles per gallon (mpg)")  
plt.title("Polynomial Regression - Auto MPG Dataset")  
plt.legend()  
plt.show()  
  
print("Polynomial Regression - Auto MPG Dataset")  
print("Mean Squared Error:", mean_squared_error(y_test, y_pred))  
print("R^2 Score:", r2_score(y_test, y_pred))  
  
if __name__ == "__main__":  
    print("Demonstrating Linear Regression and Polynomial Regression\n")  
    linear_regression_california()  
    polynomial_regression_auto_mpg()
```

## OUTPUT:





## Experiment 8

**Develop a program to demonstrate the working of the decision tree algorithm. Use Breast Cancer Data set for building the decision tree and apply this knowledge to classify a new sample.**

```
# Importing necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn import tree

data = load_breast_cancer()
X = data.data
y = data.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

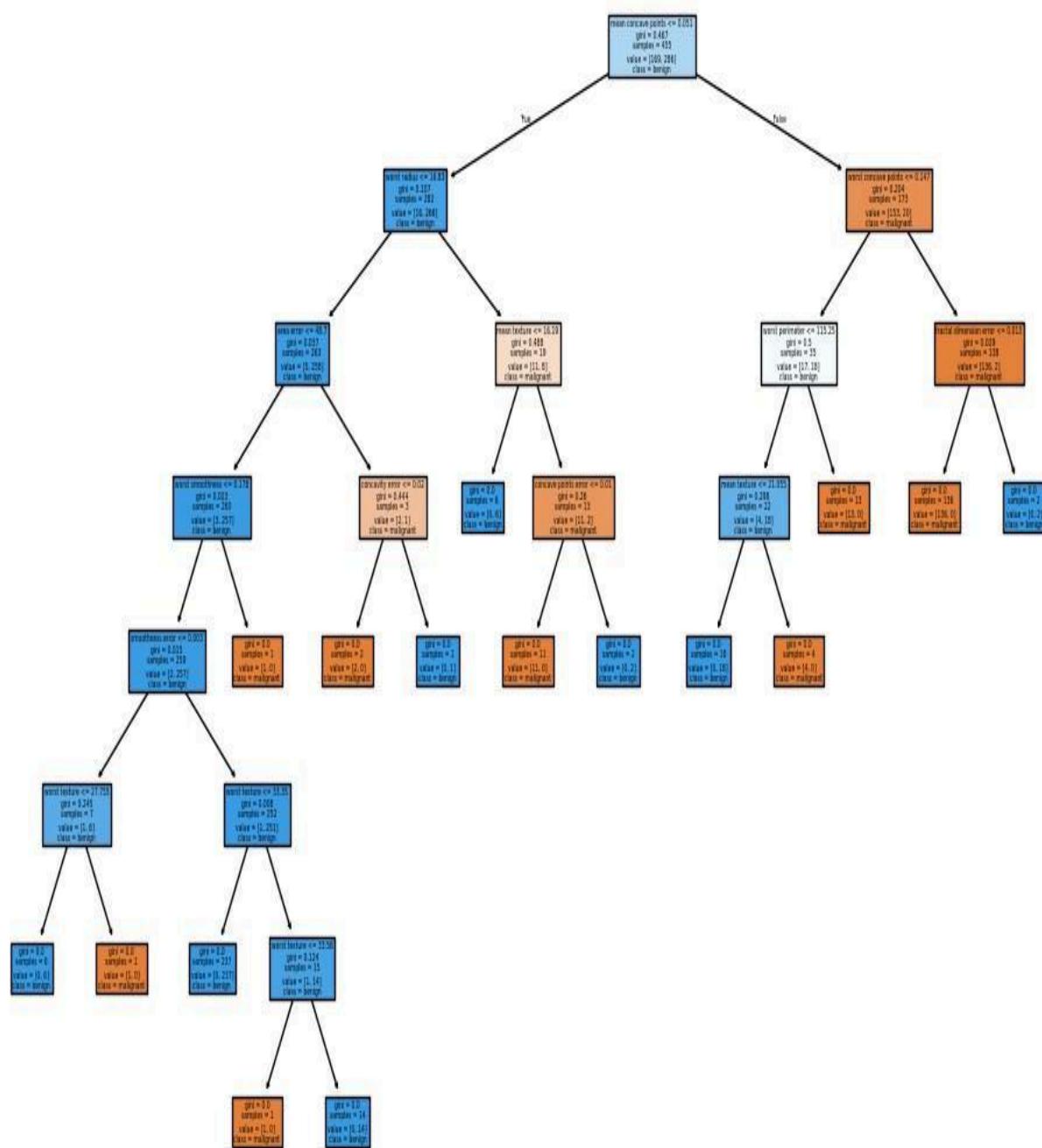
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy * 100:.2f}%")
new_sample = np.array([X_test[0]])
prediction = clf.predict(new_sample)

prediction_class = "Benign" if prediction == 1 else "Malignant"
print(f"Predicted Class for the new sample: {prediction_class}")

plt.figure(figsize=(12,8))
tree.plot_tree(clf, filled=True, feature_names=data.feature_names,
class_names=data.target_names)
plt.title("Decision Tree - Breast Cancer Dataset")
plt.show()
```

**OUTPUT:**

Decision Tree - Breast Cancer Dataset



## Experiment 9

**Develop a program to implement the Naive Bayesian classifier considering Olivetti Face Data set for training. Compute the accuracy of the classifier, considering a few test data sets.**

```
import numpy as np
from sklearn.datasets import fetch_olivetti_faces
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt

data = fetch_olivetti_faces(shuffle=True, random_state=42)
X = data.data
y = data.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

gnb = GaussianNB()
gnb.fit(X_train, y_train)
y_pred = gnb.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')

print("\nClassification Report:")
print(classification_report(y_test, y_pred, zero_division=1))

print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))

cross_val_accuracy = cross_val_score(gnb, X, y, cv=5, scoring='accuracy')
print(f'\nCross-validation accuracy: {cross_val_accuracy.mean() * 100:.2f}%')

fig, axes = plt.subplots(3, 5, figsize=(12, 8))
for ax, image, label, prediction in zip(axes.ravel(), X_test, y_test, y_pred):
```

```
ax.imshow(image.reshape(64, 64), cmap=plt.cm.gray)
ax.set_title(f"True: {label}, Pred: {prediction}")
ax.axis('off')

plt.show()
```

**OUTPUT:**

Accuracy: 80.83%

Classification Report:

	precision	recall	f1-score	support
--	-----------	--------	----------	---------

0	0.67	1.00	0.80	2
1	1.00	1.00	1.00	2
2	0.33	0.67	0.44	3
3	1.00	0.00	0.00	5
4	1.00	0.50	0.67	4
5	1.00	1.00	1.00	2
7	1.00	0.75	0.86	4
8	1.00	0.67	0.80	3
9	1.00	0.75	0.86	4
10	1.00	1.00	1.00	3
11	1.00	1.00	1.00	1
12	0.40	1.00	0.57	4
13	1.00	0.80	0.89	5
14	1.00	0.40	0.57	5
15	0.67	1.00	0.80	2
16	1.00	0.67	0.80	3
17	1.00	1.00	1.00	3
18	1.00	1.00	1.00	3
19	0.67	1.00	0.80	2
20	1.00	1.00	1.00	3
21	1.00	0.67	0.80	3
22	1.00	0.60	0.75	5
23	1.00	0.75	0.86	4
24	1.00	1.00	1.00	3
25	1.00	0.75	0.86	4
26	1.00	1.00	1.00	2
27	1.00	1.00	1.00	5
28	0.50	1.00	0.67	2
29	1.00	1.00	1.00	2
30	1.00	1.00	1.00	2
31	1.00	0.75	0.86	4
32	1.00	1.00	1.00	2
34	0.25	1.00	0.40	1
35	1.00	1.00	1.00	5
36	1.00	1.00	1.00	3
37	1.00	1.00	1.00	1

38	1.00	0.75	0.86	4
39	0.50	1.00	0.67	5
accuracy		0.81	120	
macro avg	0.89	0.85	0.83	120
weighted avg	0.91	0.81	0.81	120

Confusion Matrix:

$\begin{bmatrix} 2 & 0 & 0 & \dots & 0 & 0 & 0 \end{bmatrix}$

$\begin{bmatrix} 0 & 2 & 0 & \dots & 0 & 0 & 0 \end{bmatrix}$

$\begin{bmatrix} 0 & 0 & 2 & \dots & 0 & 0 & 1 \end{bmatrix}$

$\dots$

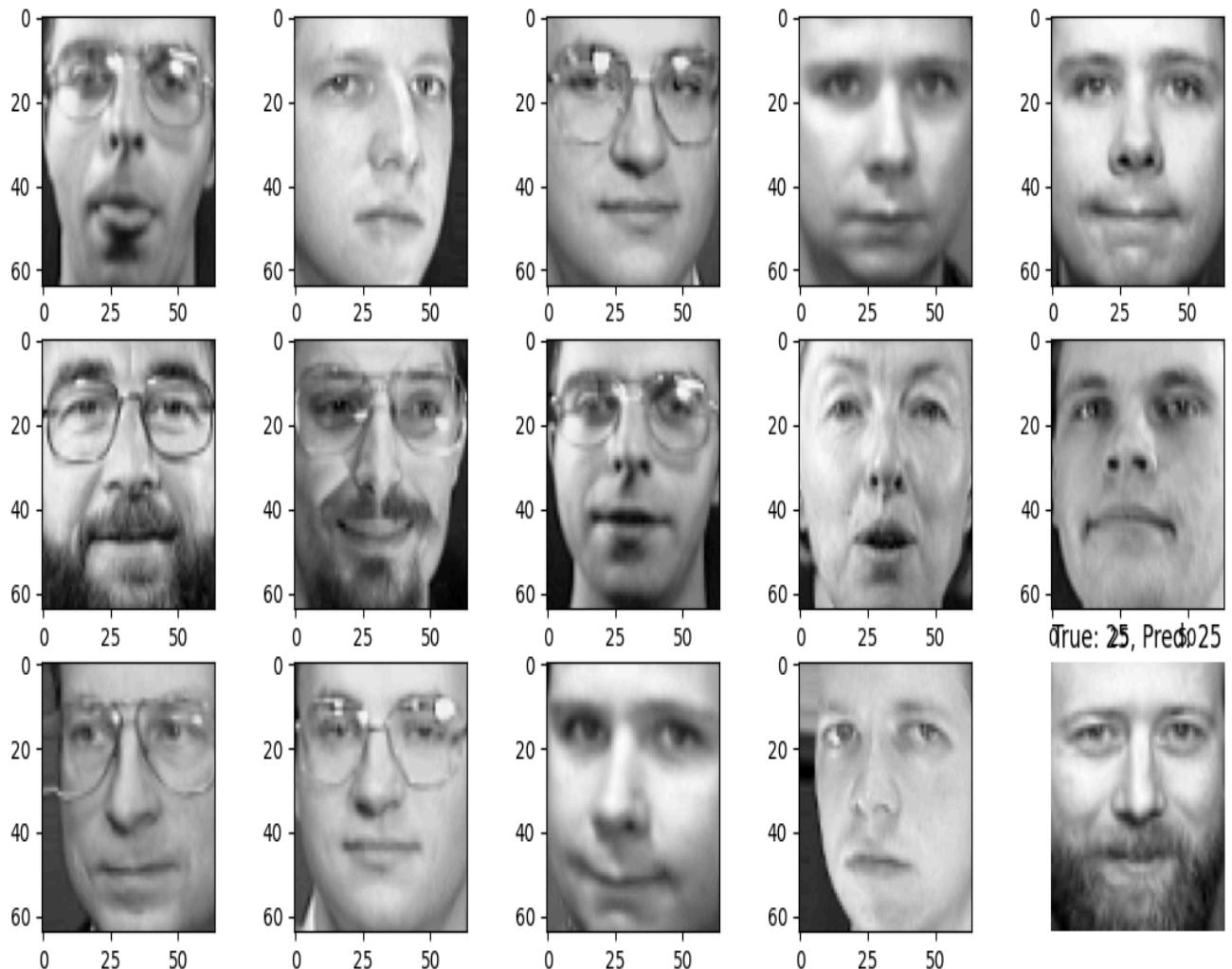
$\begin{bmatrix} 0 & 0 & 0 & \dots & 1 & 0 & 0 \end{bmatrix}$

$\begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 3 & 0 \end{bmatrix}$

$\begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 5 \end{bmatrix}$

Cross-validation  
accuracy:

87.25%



## Experiment 10

**Develop a program to implement k-means clustering using Wisconsin Breast Cancer data set and visualize the clustering result.**

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_breast_cancer
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import confusion_matrix, classification_report

data = load_breast_cancer()
X = data.data
y = data.target

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

kmeans = KMeans(n_clusters=2, random_state=42)
y_kmeans = kmeans.fit_predict(X_scaled)

print("Confusion Matrix:")
print(confusion_matrix(y, y_kmeans))
print("\nClassification Report:")
print(classification_report(y, y_kmeans))

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

df = pd.DataFrame(X_pca, columns=['PC1', 'PC2'])
df['Cluster'] = y_kmeans
df['True Label'] = y

plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='PC1', y='PC2', hue='Cluster', palette='Set1', s=100,
edgecolor='black', alpha=0.7)
plt.title('K-Means Clustering of Breast Cancer Dataset')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend(title="Cluster")
plt.show()

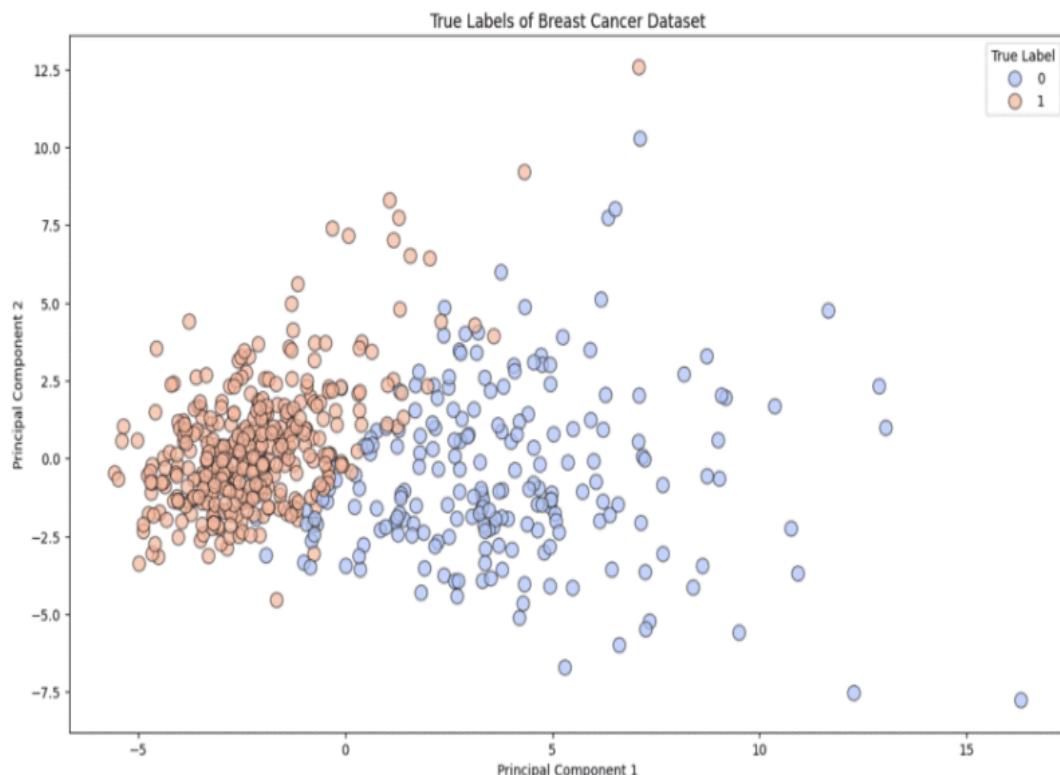
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='PC1', y='PC2', hue='True Label', palette='coolwarm',
s=100, edgecolor='black', alpha=0.7)
plt.title('True Labels of Breast Cancer Dataset')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend(title="True Label")
```

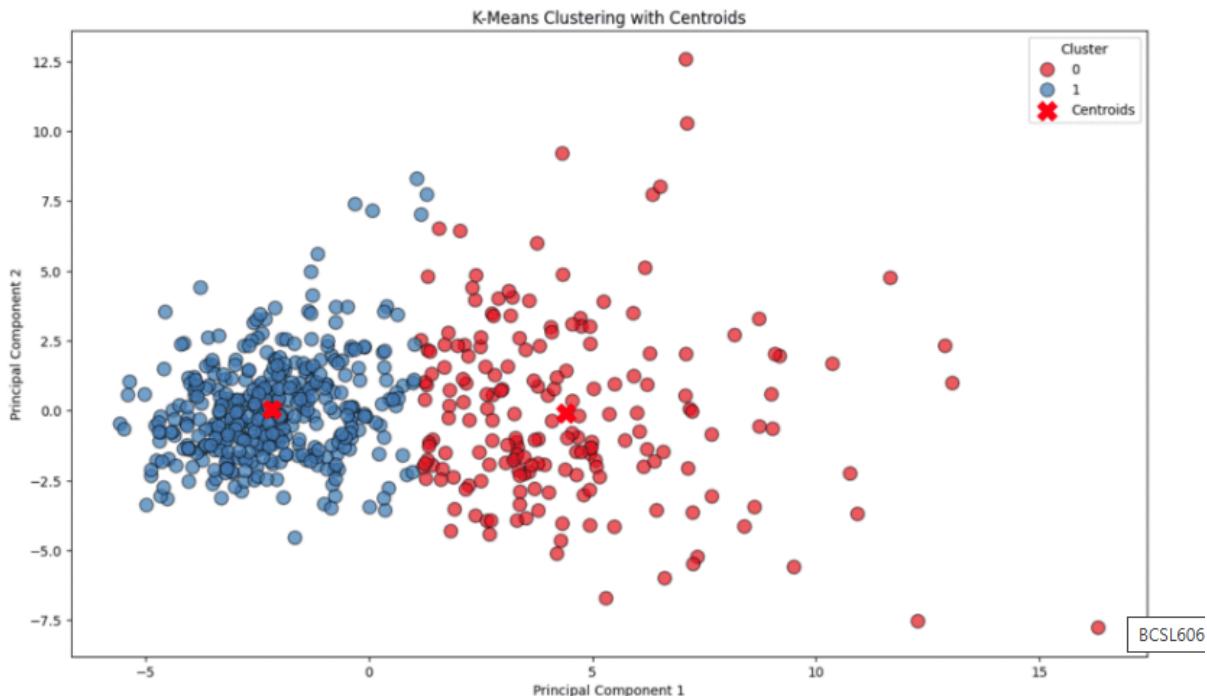
```
plt.show()
```

```
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='PC1', y='PC2', hue='Cluster', palette='Set1', s=100,
edgecolor='black', alpha=0.7)
centers = pca.transform(kmeans.cluster_centers_)
plt.scatter(centers[:, 0], centers[:, 1], s=200, c='red', marker='X', label='Centroids')
plt.title('K-Means Clustering with Centroids')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend(title="Cluster")
plt.show()
```

## OUTPUT:

Clustering Accuracy (approx.): 0.09





Confusion Matrix:

```
[[175 37]
 [ 13 344]]
```

Classification Report:

	precision	recall	f1-score	support
--	-----------	--------	----------	---------

0	0.93	0.83	0.88	212
1	0.90	0.96	0.93	357

accuracy		0.91	0.91	569
macro avg	0.92	0.89	0.90	569
weighted avg	0.91	0.91	0.91	569

## Machine Learning Lab

### Viva Questions

#### **1. What is the difference between supervised and unsupervised machine learning?**

A Supervised learning is a process where it requires training labeled data. When it comes to Unsupervised learning it doesn't require data labeling.

#### **2. How is KNN different from K-means clustering?**

KNN stands for K- Nearest Neighbours, it is classified as a supervised algorithm. K-means is an unsupervised cluster algorithm.

#### **4. How to handle or missing data in a dataset?**

An individual can easily find missing or corrupted data in a data set either by dropping the rows or columns. On contrary, they can decide to replace the data with another value. In Pandas there are two ways to identify the missing data, these two methods are very useful. `isnull()` and `dropna()`.

#### **5. What is the difference between an array and Linked list?**

Deep An array is an ordered fashion of collection of objects. A linked list is a series of objects that are processed in a sequential order.

#### **6. Explain why Navie Bayes is so Naive?**

It is based on an assumption that all of the features in the data set are important, equal and independent.

#### **7. Please state few popular Machine Learning algorithms?**

Nearest Neighbour

Neural Networks

Decision Trees etc

Support vector machines

#### **8. What are the different types of algorithm techniques available in machine learning?**

Some of them are :

Supervised learning

Unsupervised learning

Semi-supervised

learning Transduction

Learning to learn

#### **9. What are the three stages to build the model in machine learning?**

1. Model building
  2. Model testing
  3. Applying the model
-

**10. Name a few libraries in Python used for Data Analysis and Scientific computations**  
NumPy, SciPy, Pandas, SciKit, Matplotlib, Seaborn

**11. Which is the standard data missing marker used in Pandas?**

NaN

**12. Write the code to sort an array in NumPy by the nth column?**

Using argsort () function this can be achieved. If there is an array X and you would like to sort the nth column then code for this will be `x[x [: n-1].argsort ()]`

**13.What is pylab?**

A package that combines NumPy, SciPy and Matplotlib into a single namespace.

**14. Is all the memory freed when Python exits?**

No it is not, because the objects that are referenced from global namespaces of Python modules are not always de-allocated when Python exits.

**15. How can you randomize the items of a list in place in Python?**

Shuffle (lst) can be used for randomizing the items of a list in Python

**16. Which tool in Python will be used to find bugs if any?**

Pylint and Pychecker. Pylint verifies that a module satisfies all the coding standards or not. Pychecker is a static analysis tool that helps find out bugs in the course code.

**17. What are the supported data types in Python?**

Python has five standard data types –

- Numbers
- String
- List
- Tuple
- Dictionary

**19. What are the supported sequence types in Python?**

Python supports 7 sequence types. They are str, list, tuple, unicode, byte array, xrange, and buffer. where xrange is deprecated in python 3.5.X.

**20. What is pylab?**

A package that combines NumPy, SciPy and Matplotlib into a single namespace.

**21. What are the supported data types in Python?**

SciKit-Learn

---

**23. Is Python a case-sensitive programming language?**

Yes, it is a case-sensitive language.

**24. What is the difference between a tuple and a list?**

The basic difference between a tuple and a list is that the former is immutable and the latter is mutable.

**25. What is the difference between Xrange() and range()?**

Range() returns a list and xrange() returns an xrange object, which is kind of like an iterator and generates the numbers on demand.

**26. Optimize the below python**

```
code- word = 'word'  
      print word._len__()  
print 'word'._len_()
```

**27. What is PEP 8?**

PEP 8 is a coding convention that lets us write more readable code. In other words, it is a set of recommendations.

**28. What are Python decorators?**

A Python decorator is a specific change that we make in Python syntax to alter functions easily.

**29. What is Dict and List comprehensions are?**

They are syntax constructions to ease the creation of a Dictionary or List based on existing iterable.

**30. What is lambda in Python?**

It is a single expression anonymous function often used as inline function.

**31. What is pass in Python?**

Pass means, no-operation Python statement, or in other words it is a place holder in compound statement, where there should be a blank left and nothing has to be written there.

**32. In Python what are iterators?**

In Python, iterators are used to iterate a group of elements, containers like list.

**33. In Python what is slicing?**

A mechanism to select a range of items from sequence types like list, tuple, strings etc. is known as slicing.

---

**34. What is docstring in Python?**

A Python documentation string is known as docstring, it is a way of documenting Python functions, modules and classes.

**35. How can you copy an object in Python?**

To copy an object in Python, you can try `copy.copy()` or `copy.deepcopy()` for the general case. You cannot copy all objects but most of them.

**36. Explain how to delete a file in Python?**

By using a command `os.remove(filename)` or `os.unlink(filename)`

**37. Is It Mandatory For A Python Function To Return A Value?**

It is not at all necessary for a function to return any value. However, if needed, we can use `None` as a return value.

**38. What Is Whitespace In Python?**

Whitespace represents the characters that we use for spacing and separation. They possess an “empty” representation. In Python, it could be a tab or space.

**39. What Is Isalpha() In Python?**

Python provides this built-in `isalpha()` function for the string handling purpose. It returns `True` if all characters in the string are of alphabet type, else it returns `False`.

**40. What Does The Join Method Do In Python?**

Python provides the `join()` method which works on strings, lists, and tuples. It combines them and returns a united value.

**41. What Makes The CPython Different From Python?**

CPython has its core developed in C. The prefix ‘C’ represents this fact. It runs an interpreter loop used for translating the Python-ish code to C language.

**42. What Is A Tuple In Python?**

A tuple is a collection type data structure in Python which is immutable. They are similar to sequences, just like the lists. However, there are some differences between a tuple and list; the former doesn’t allow modifications whereas the list does.

**43. What is Artificial Intelligence?**

Artificial Intelligence is an area of computer science that emphasizes the creation of intelligent machine that work and reacts like humans.

---

**44. What are the various areas where AI (Artificial Intelligence) can be used?**

Artificial Intelligence can be used in many areas like Computing, Speech recognition, Bio-informatics, Humanoid robot, Computer software, Space and Aeronautics' etc.

---