EXPERIMENT – 1

WRITE A MACHINE LEARNING PROGRAM IN PYTHON TO SOLVE LINEAR REGRESSION

Aim: The aim of this program is to predict continuous values from input features.

Program:

```
from sklearn.linear_model import LinearRegression

import numpy as np

X = np.array([[1], [2], [3], [4], [5]])

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

model = LinearRegression()

model.fit(X, y)

prediction = model.predict([[6]])

print(prediction)
```

Input:

X = [[1], [2], [3], [4], [5]], y = [1, 2, 3, 4, 5]

Output: [6.]

Result: The model predicts the value `6` for the input `6`.

EXPERIMENT – 2

WRITE A MACHINE LEARNING PROGRAM IN PYTHON TO SOLVE LOGISTIC REGRESSION

Aim: The aim of this program is to classify data into binary categories.

Procedure:

```
from sklearn.linear_model import LogisticRegression

import numpy as np

X = np.array([[0], [1], [2], [3], [4], [5]])

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

model = LogisticRegression()

model.fit(X, y)

prediction = model.predict([[1.5]])

print(prediction)
```

Input:

X = [[0], [1], [2], [3], [4], [5]], y = [0, 0, 0, 1, 1, 1]

Output: [0]

Result: The model predicts the class `0` for the input `1.5`.

EXPERIMENT – 3

WRITE A MACHINE LEARNING PROGRAM IN PYTHON TO SOLVE DECISION TREE CLASSIFIER

Aim: The aim of this program is to classify data into multiple categories.

Procedure:

```
from sklearn.tree import DecisionTreeClassifier
import numpy as np
X = np.array([[0], [1], [2], [3], [4], [5]])
y = np.array([0, 0, 1, 1, 2, 2])
model = DecisionTreeClassifier()
model.fit(X, y)
prediction = model.predict([[3]])
print(prediction)
```

Input:

X = [[0], [1], [2], [3], [4], [5]], y = [0, 0, 1, 1, 2, 2]

Output: [1]

Result: The model predicts the class `1` for the input `3`.

EXPERIMENT – 4

WRITE A MACHINE LEARNING PROGRAM IN PYTHON TO SOLVE K-NEAREST NEIGHBORS

Aim: The aim of this program is to classify data based on the nearest neighbors.

Procedure:

```
from sklearn.neighbors import KNeighborsClassifier
import numpy as np
X = np.array([[0], [1], [2], [3], [4], [5]])
y = np.array([0, 0, 1, 1, 2, 2])
model = KNeighborsClassifier(n_neighbors=3)
model.fit(X, y)
prediction = model.predict([[3.5]])
print(prediction)
```

Input:

X = [[0], [1], [2], [3], [4], [5]], y = [0, 0, 1, 1, 2, 2]

Output: [1]

Result: The model predicts the class `1` for the input `3.5`.

EXPERIMENT – 5

WRITE A MACHINE LEARNING PROGRAM IN PYTHON TO SOLVE SUPPORT VECTOR MACHINE

Aim: The aim of this program is to classify the data using support vector classification.

Procedure:

```
from sklearn import svm
import numpy as np
X = np.array([[0], [1], [2], [3], [4], [5]])
y = np.array([0, 0, 1, 1, 2, 2])
model = svm.SVC()
model.fit(X, y)
prediction = model.predict([[4.5]])
print(prediction)
```

Input:

X = [[0], [1], [2], [3], [4], [5]], y = [0, 0, 1, 1, 2, 2]

Output: [2]

Result: The model predicts the class `2` for the input `4.5`.

EXPERIMENT – 6

WRITE A MACHINE LEARNING PROGRAM IN PYTHON TO SOLVE NAÏVE BAYES CLASSIFIER

Aim: The aim of this program is to classify data using probabilistic models.

Procedure:

```
from sklearn.naive_bayes import GaussianNB
import numpy as np
X = np.array([[1, 2], [2, 3], [3, 4], [4, 5], [5, 6]])
y = np.array([0, 0, 1, 1, 1])
model = GaussianNB()
model.fit(X, y)
prediction = model.predict([[3, 5]])
print(prediction)
```

Input:

X = [[1, 2], [2, 3], [3, 4], [4, 5], [5, 6]], y = [0, 0, 1, 1, 1]

Output: [1]

Result: The model predicts the class `1` for the input `[3, 5]`.