FUNDAMENTALS OF MACHINE LEARING
FOR PATTERN DEVELOPERS
MLA0202

MODEL PRACTICAL

Lathika . S
192425066
AIML
3/02/2026
SET - 6

1. AIM: To analyze a housing dataset and predict house prices using machine learning to help mark make a buying decision.

ALGORITHM:

1. Import required libraries
2. Read the house dataset using pandas
3. Display first five records.
4. Perform basic statistical analysis
5. Display column names & data types
6. Replace missing values with mode
7. Visualize data using a heatmap.
8. Split data & Train a regression model.

CODE:

```
import pandas as Pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn. model - selection
import from train_ test - split
import from sklearn. linear . model import
Linear Regression
data = pd. read - csv ("house - data. csv")
Print (data. head ())
Print (data. describe ())
print (data . dtypes )
data. fillna (data. mode () . iloc [0], inplace = True)
sns. heatmap (data. corr () , annot = True)
plt. show ()
```

```
x = data.drop("Price", axis=1)
y = data ["Price"]
X_train, x_test, y_train, y_test = train_test_Split
  (x, y, test_size = 0.2)
model = Linear_Regression()
model. fit_train, y_train()
Print (price_pred).
```

OUTPUT:

```
[545000  6123000  478900  689400]
```

Result: Thus the program executed Sucessfully

2. AIM: To learn the most Specific hypothesis for identifying malignant tumors using the FIND-S algorithm.

ALGORITHM:

1. Initialize hypothesis with the most Specific values.
2. Consider only positive examples.
3. Generalize hypothesis to cover each positive example.
4. Ignore negative examples.
5. Output final hypothesis

CODE:

```
hypothesis = ['ϕ', 'ϕ', 'ϕ', 'ϕ', 'ϕ']
data = [
['circular', 'large', 'light', 'Smooth', 'Thick', 'Malignant']
['circular', 'large', 'light', 'Irregular', 'Thick', 'Malignant'],
['Oval', 'Large', 'Light', 'Irregular', 'Thick', 'Malignant)
]
```

```
for row in data:
    for i in range(len(hypothesis)):
    ~~for i in~~
        if hypothesis (i] = = 'ϕ':
            hypothesis (i] = row [i]
      elif hypothsis [i] ! = row [i]: hypothesis [i] = '?'
Print (hypothesis)
```

OUTPUT:

```
['?', 'Large', 'Light', '?', 'Thick']
```

Result: Thus the program executed Sucessfully

3. AIM: To impliment Linear Regression in Python and evaluate its performance.

ALGORITHM:

1. Import required libraries
2. Load the dataset
3. Split data into training & testing sets
4. Train the Linear Regression model
5. Predict output values
6. Evaluate model performance

CODE:

```
import pandas as pd
from sklearn.model_selection
import train_test_split
from sklearn.linear_model
import LinearRegression
from sklearn.metrics import
mean_squared_error, r2_score
```

```
X = [[1] [2] [3] [4] [5]]
y = [2,4, 6, 8, 10]
X_train, X_test, y_train, y_test =
  train_test_split (X, y, test_size = 0.2)
model = LinearRegression ()
model. fit (X_train, y_train)
y_pred = model. predict (X_test)
Print ("MSE", mean_squared_error (y_test, y_pred))
Print ("R2 Score:", r2_score (y_test, y_pred))
```

OUTPUT:

```
MSE : 0.0
R2 Score : 1.0
```

Result: Thus the program escuited Sucessfully

4. AIM: To implment the expectation - Mascimization (EM) algorithm using python to estimate paramitas of a Gaussian Mixture Model.

ALGORITHM:

1. Initialize paramitors (mean, variance, weights)
2. E - step : Compute responsibilities
3. M - step : update paramitors.
4. Repeat until Convergence
5. Display final paramitors

~~OUT~~
CODE:

```
import numpy as np
from SKlearn mixture import
Gaussianmixture

X = np. array ([[1], [2], [3] [4], [5], [6], [7]])
model = Gaussianmixture (n - components = 2)
model. fit (x)

Print ("Means:", model means-)
Print ("Weights:", model weights-)
```

OUTPUT:

```
Means : [[2.0] [9.0]]

weights: [0.5 0,5]
```

Result:-

Thus, the program Sucessfully executed