Practical No. 9 (e)

Aim : Predicting Housing Prices using K-Nearest Neighbors (KNN) Algorithm

Theory:

Data Preprocessing: The dataset containing information about properties such as square footage, number of bedrooms, location, etc., and their corresponding prices undergo preprocessing. This involves handling missing values, scaling features, encoding categorical variables, and splitting the data into training and testing sets.

Choosing the Value of K: The choice of the value of k (number of neighbors) is crucial. A smaller k leads to a more flexible model prone to overfitting, while a larger k makes the model more stable but might underfit the data. Techniques like cross-validation or grid search are often used to determine the optimal value of k.

Calculating Distance: For each data point in the test set, the distance between that point and every other point in the training set is calculated. Common distance metrics include Euclidean distance, Manhattan distance, or cosine similarity, depending on the nature of the data.

Finding Nearest Neighbors: The k-nearest neighbors of the test data point are identified based on the calculated distances. These neighbors are determined by selecting the k data points with the smallest distances to the test data point.

Making Predictions: For regression tasks like predicting housing prices, the predicted value for the test data point is usually the average of the target values of its k-nearest neighbors. Alternatively, a weighted average can be used, where closer neighbors have a higher influence on the prediction.

Model Evaluation: Finally, the performance of the KNN model is evaluated using metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), or R-squared on the test set.

Program Code:

```
# Importing necessary libraries

from sklearn.datasets import fetch_california_housing

from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsRegressor

from sklearn.metrics import mean_squared_error, r2_score


# Load the California housing dataset

housing = fetch_california_housing()

X = housing.data
```

```
y = housing.target

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create a KNN regressor
knn_regressor = KNeighborsRegressor(n_neighbors=5)

# Train the regressor
knn_regressor.fit(X_train, y_train)

# Predict on the test data
y_pred = knn_regressor.predict(X_test)

# Calculate evaluation metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error:", mse)
print("R-squared Score:", r2)
```

Output:

Result/Conclusion:

By following this lab content, students should gain practical experience in using the K-Nearest Neighbors algorithm for predicting housing prices, understanding its strengths, and exploring ways to improve its performance.