Open Source Software Lab Lab Test 2 Thursday – 3 to 5 PM

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.datasets import fetch_california_housing
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

In [46]:

```
housing = fetch_california_housing()
df = pd.DataFrame(housing.data, columns=housing.feature_names)
print(df.head(10))
```

```
   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
0  8.3252      41.0  6.984127   1.023810       322.0  2.555556     37.88
1  8.3014      21.0  6.238137   0.971880      2401.0  2.109842     37.86
2  7.2574      52.0  8.288136   1.073446       496.0  2.802260     37.85
3  5.6431      52.0  5.817352   1.073059       558.0  2.547945     37.85
4  3.8462      52.0  6.281853   1.081081       565.0  2.181467     37.85
5  4.0368      52.0  4.761658   1.103627       413.0  2.139896     37.85
6  3.6591      52.0  4.931907   0.951362      1094.0  2.128405     37.84
7  3.1200      52.0  4.797527   1.061824      1157.0  1.788253     37.84
8  2.0804      42.0  4.294118   1.117647      1206.0  2.026891     37.84
9  3.6912      52.0  4.970588   0.990196      1551.0  2.172269     37.84

   Longitude
0    -122.23
1    -122.22
2    -122.24
3    -122.25
4    -122.25
5    -122.25
6    -122.25
7    -122.25
8    -122.26
9    -122.25
```

In [53]:

```
print(df.describe())
```

```
             MedInc      HouseAge      AveRooms     AveBedrms    Population  \
```

```
count  20640.000000  20640.000000  20640.000000  20640.000000  20640.000000
mean       3.870671     28.639486      5.429000      1.096675   1425.476744
std        1.899822     12.585558      2.474173      0.473911   1132.462122
min        0.499900      1.000000      0.846154      0.333333      3.000000
25%        2.563400     18.000000      4.440716      1.006079    787.000000
50%        3.534800     29.000000      5.229129      1.048780   1166.000000
75%        4.743250     37.000000      6.052381      1.099526   1725.000000
max       15.000100     52.000000    141.909091     34.066667  35682.000000

          AveOccup      Latitude     Longitude
count  20640.000000  20640.000000  20640.000000
mean       3.070655     35.631861   -119.569704
std       10.386050      2.135952      2.003532
min        0.692308     32.540000   -124.350000
25%        2.429741     33.930000   -121.800000
50%        2.818116     34.260000   -118.490000
75%        3.282261     37.710000   -118.010000
max     1243.333333     41.950000   -114.310000
```

In [48]:

```
print(df.isnull().sum())
```

```
MedInc        0
HouseAge      0
AveRooms      0
AveBedrms     0
Population    0
AveOccup      0
Latitude      0
Longitude     0
dtype: int64
```

In [49]:

```
X=df[["HouseAge"]]
Y=df["MedInc"]
```

In [50]:

```
X_train,X_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=42)
linear_model = LinearRegression()
linear_model.fit(X_train, y_train)
y_pred_linear = linear_model.predict(X_test)
linear_mse=mean_squared_error(y_test, y_pred_linear)
linear_rmse = np.sqrt(mean_squared_error(y_test, y_pred_linear))
print("Linear Regression MSE:", linear_mse)
print("Linear Regression RMSE:", linear_rmse)
```

Linear Regression MSE: 3.5010405083205534
Linear Regression RMSE: 1.8711067602679847

In [52]:

```
def reverse_anagrams(arr):
    def is_anagram(s1, s2):
        return sorted(s1) == sorted(s2)

    result = []
    for i in range(len(arr)):
        reversed_str = arr[i][::-1]
        if any(is_anagram(arr[i], arr[j]) for j in range(len(arr)) if i != j):
            result.append(reversed_str)
        else:
            result.append(arr[i])
    return result

# Example usage:
strings = ["listen", "silent", "enlist", "hello", "world"]
print(reverse_anagrams(strings))
```

```
['netsil', 'tnelis', 'tsilne', 'hello', 'world']
```

In [ ]: