Parshvanath Charitable Trust's

A. P. SHAH INSTITUTE OF TECHNOLOGY

(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)
(Religious Jain Minority)

Class: BE **Subject: Machine Learning** **Sem-VII**

Experiment No:8

Course Outcome: CO3 **Blooms Level:L3**

Aim: To implement k-means algorithm.

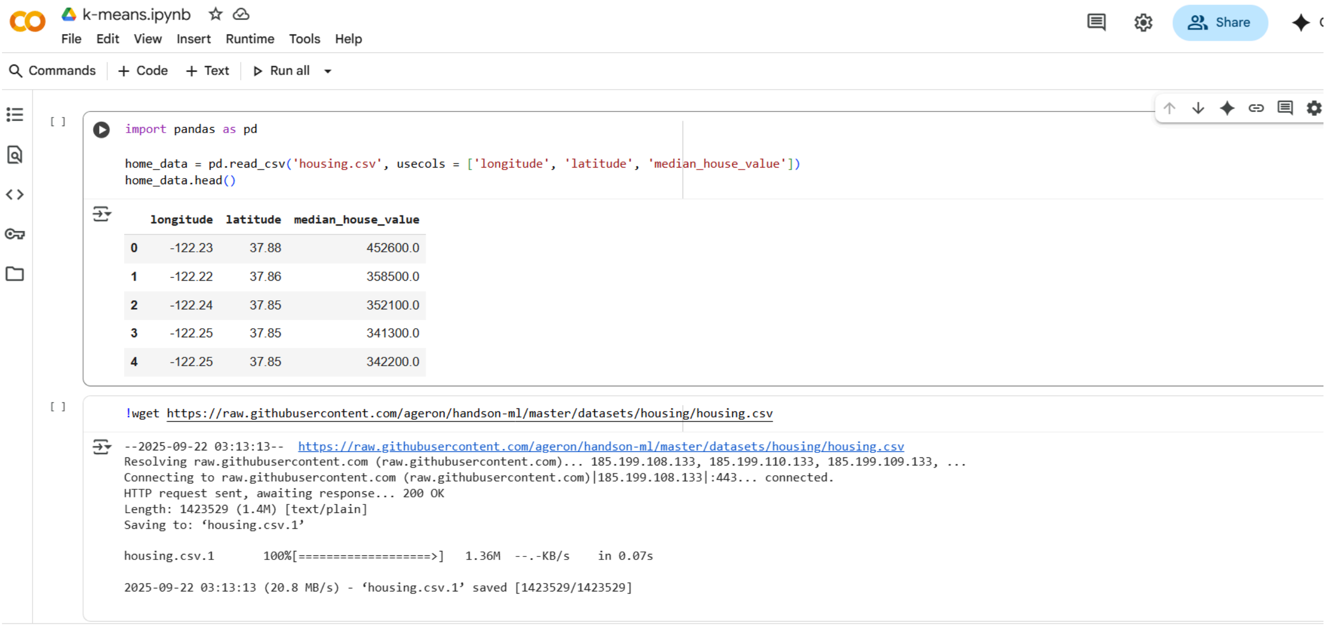
Abstract: k-means clustering is a popular clustering algorithm that does a good job of grouping spherical data together into distinct groups. This is very valuable as both an analysis tool when the groupings of rows of data are unclear or as a feature-engineering step for improving supervised learning models.

Sample Input and Output:

Theory:

In this tutorial, we will be using California housing data from Kaggle (here). We will use location data (latitude and longitude) as well as the median house value. We will cluster the houses by location and observe how house prices fluctuate across California. We save the dataset as a csv file called `'housing.csv'` in our working directory and read it using `pandas`.

Program:



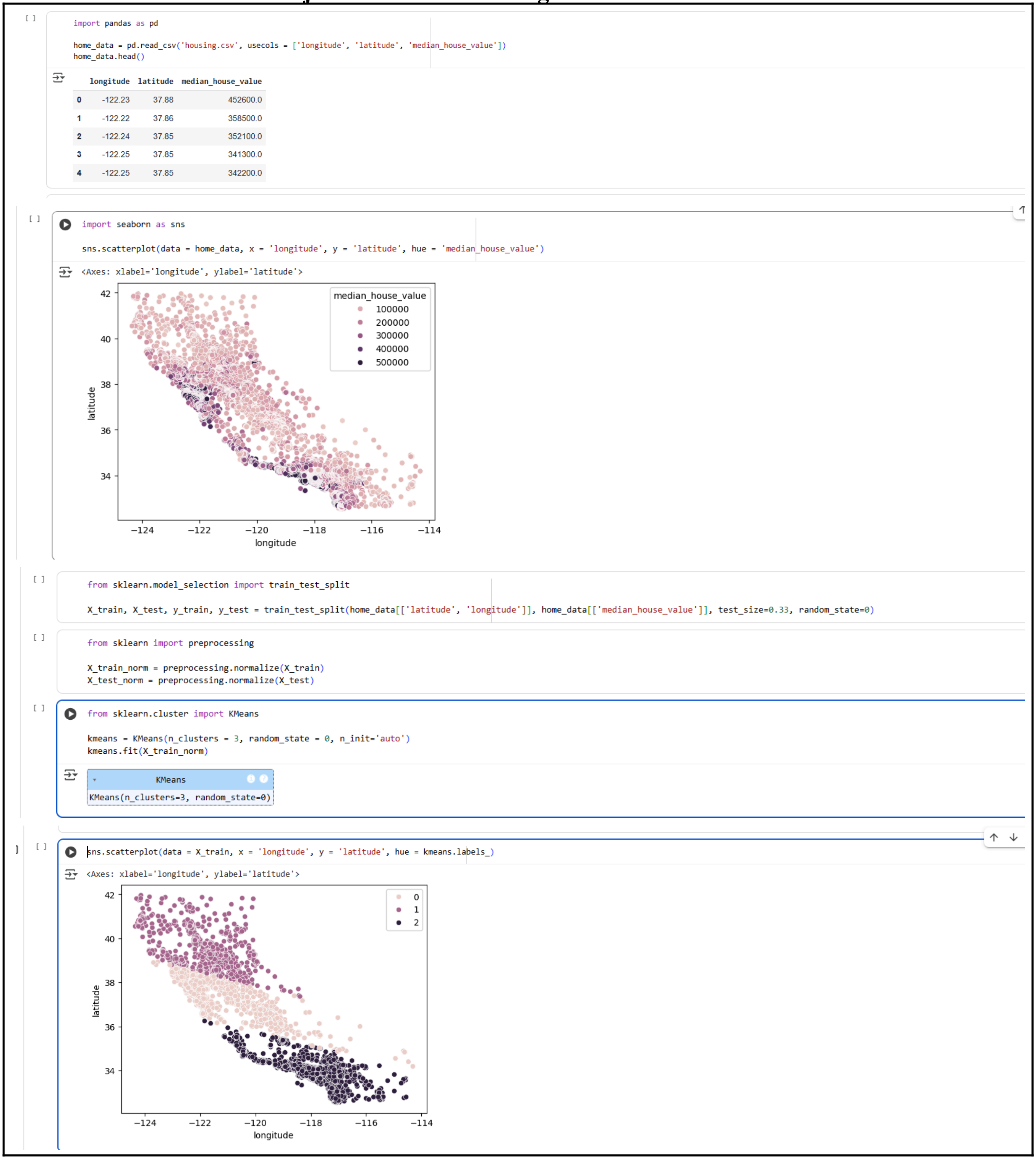
k-means.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text ▷ Run all

```
import pandas as pd

home_data = pd.read_csv('housing.csv', usecols = ['longitude', 'latitude', 'median_house_value'])
home_data.head()
```

	longitude	latitude	median_house_value
0	-122.23	37.88	452600.0
1	-122.22	37.86	358500.0
2	-122.24	37.85	352100.0
3	-122.25	37.85	341300.0
4	-122.25	37.85	342200.0

```
!wget https://raw.githubusercontent.com/ageron/handson-ml/master/datasets/housing/housing.csv
```

```
--2025-09-22 03:13:13--  https://raw.githubusercontent.com/ageron/handson-ml/master/datasets/housing/housing.csv
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.110.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1423529 (1.4M) [text/plain]
Saving to: ‘housing.csv.1’

housing.csv.1       100%[===================>]   1.36M  --.-KB/s    in 0.07s

2025-09-22 03:13:13 (20.8 MB/s) - ‘housing.csv.1’ saved [1423529/1423529]
```



```
import pandas as pd

home_data = pd.read_csv('housing.csv', usecols = ['longitude', 'latitude', 'median_house_value'])
home_data.head()
```

	longitude	latitude	median_house_value
0	-122.23	37.88	452600.0
1	-122.22	37.86	358500.0
2	-122.24	37.85	352100.0
3	-122.25	37.85	341300.0
4	-122.25	37.85	342200.0

```
import seaborn as sns

sns.scatterplot(data = home_data, x = 'longitude', y = 'latitude', hue = 'median_house_value')
```

```
<Axes: xlabel='longitude', ylabel='latitude'>
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(home_data[['latitude', 'longitude']], home_data[['median_house_value']], test_size=0.33, random_state=0)
```

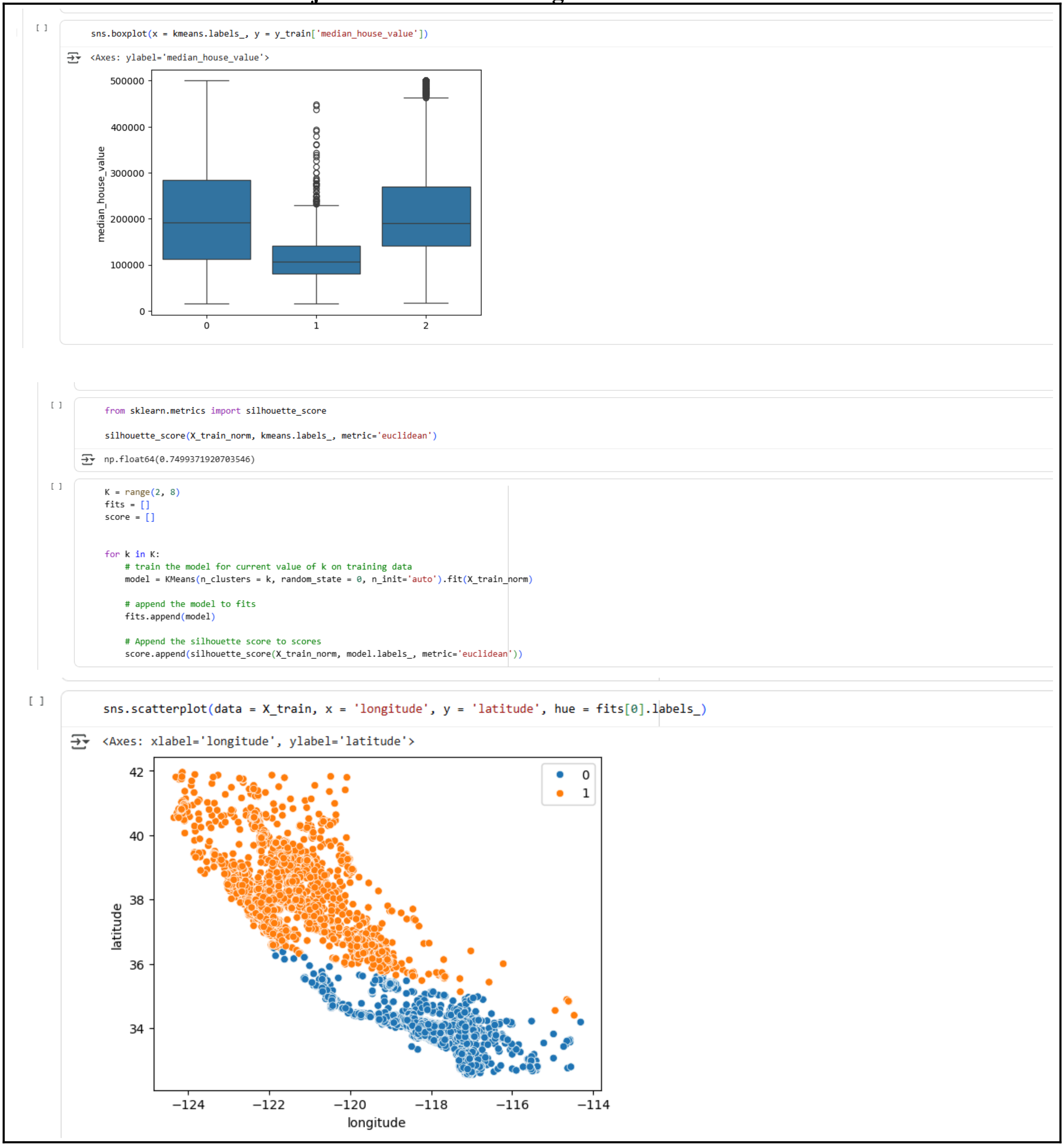
```
from sklearn import preprocessing

X_train_norm = preprocessing.normalize(X_train)
X_test_norm = preprocessing.normalize(X_test)
```

```
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters = 3, random_state = 0, n_init='auto')
kmeans.fit(X_train_norm)
```

```
KMeans
KMeans(n_clusters=3, random_state=0)
```

```
sns.scatterplot(data = X_train, x = 'longitude', y = 'latitude', hue = kmeans.labels_)
```

```
<Axes: xlabel='longitude', ylabel='latitude'>
```



```
sns.boxplot(x = kmeans.labels_, y = y_train['median_house_value'])
```

```
<Axes: ylabel='median_house_value'>
```

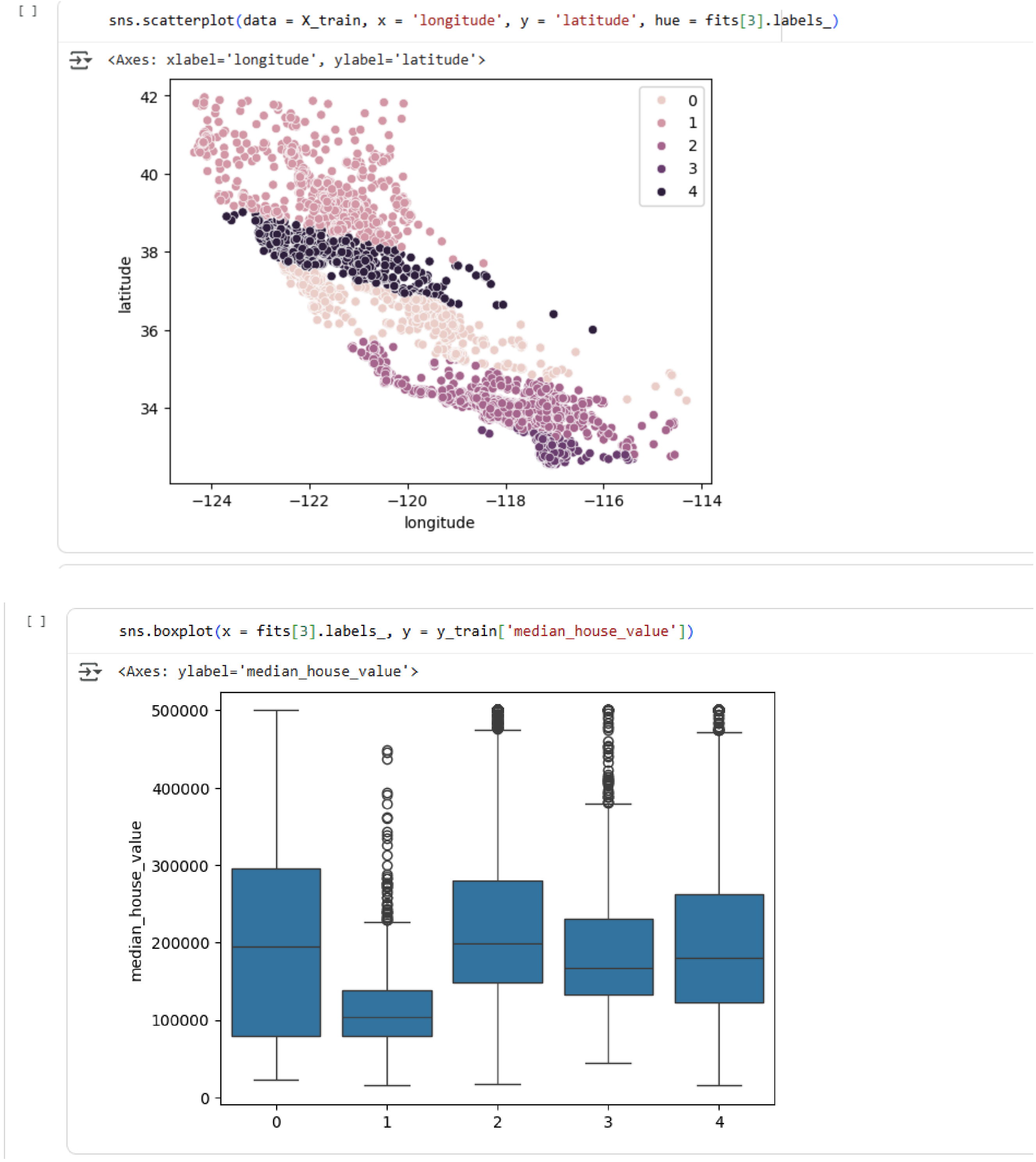
```
from sklearn.metrics import silhouette_score

silhouette_score(X_train_norm, kmeans.labels_, metric='euclidean')
```

```
np.float64(0.7499371920703546)
```

```
K = range(2, 8)
fits = []
score = []


for k in K:
    # train the model for current value of k on training data
    model = KMeans(n_clusters = k, random_state = 0, n_init='auto').fit(X_train_norm)

    # append the model to fits
    fits.append(model)

    # Append the silhouette score to scores
    score.append(silhouette_score(X_train_norm, model.labels_, metric='euclidean'))
```

```
sns.scatterplot(data = X_train, x = 'longitude', y = 'latitude', hue = fits[0].labels_)
```

```
<Axes: xlabel='longitude', ylabel='latitude'>
```



```
sns.scatterplot(data = X_train, x = 'longitude', y = 'latitude', hue = fits[3].labels_)
```

```
<Axes: xlabel='longitude', ylabel='latitude'>
```

```
sns.boxplot(x = fits[3].labels_, y = y_train['median_house_value'])
```

```
<Axes: ylabel='median_house_value'>
```

Output:

Conclusion: K-means clustering performs best on data that are spherical. Spherical data are data that group in space in close proximity to each other either. This can be visualized in 2 or 3 dimensional space more easily. Data that aren't spherical or should not be spherical do not work well with k-means clustering. For example, k-means clustering would not do well on the below data as we would not be able to find distinct centroids to cluster the two circles or arcs differently, despite them clearly visually being two distinct circles and arcs that should be labeled as such.

Exercise 1:

You are provided with the famous **Iris dataset**, which contains measurements of three different species of iris flowers. Each sample has the following attributes:

- Sepal Length (cm)
- Sepal Width (cm)
- Petal Length (cm)
- Petal Width (cm)

1. Apply **K-Means clustering** to group the flowers into clusters without using the species label.
2. Use **K = 3** (since there are 3 flower species) and fit the model.
3. Compare the clustering results with the actual species labels to check the accuracy.
4. Visualize the clusters in **2D** using Petal Length vs. Petal Width.
5. Discuss whether K-Means was effective in separating the species.

Students shall draw flowchart of exercise question in the writeup and submit.

Exercise 2:

You are given a dataset containing the **annual income (in ₹)** and **spending score (1–100)** of customers in a shopping mall.

Apply **K-Means clustering** on this dataset to group the customers into different clusters based on their purchasing behavior.

Use the **Elbow method** to determine the optimal number of clusters.

Visualize the clusters using a 2D scatter plot where:

X-axis represents **Annual Income**

Y-axis represents **Spending Score**

Different clusters are shown in different colors.

Interpret the characteristics of each cluster (e.g., high-income high-spending, low-income low-spending, etc.).

Suggest how the shopping mall can use these insights for **customer segmentation** and **targeted marketing strategies**.

Sample Dataset Columns:

CustomerID

Age

Annual Income (₹)

Spending Score (1–100)

Students shall draw flowchart of exercise question in the writeup and submit.