Data Mining Final Project:
California Housing Dataset Analysis

Nguyen Minh Duc
20202026
Department of Computer Science Engineering
UNIST, South Korea

Nguyen Thu Phuong
20202027
Department of Computer Science Engineering
UNIST, South Korea

1 Problem Statement

1.1 California Housing Dataset

In this project, we choose a California Housing Dataset, a widely recognized dataset in the field of Machine Learning. The dataset can be obtained via github.com/ageron. The dataset contains information on the variables using all the block groups in California from the 1990 Census, with each block consists of 1425.5 individuals living in a geographically compact area. Figure 1 shows a preview of this dataset. The dataset has 20.640 entries. Each entry represents a block and includes the following 10 attributes:

- **Longitude**: Geographic coordinate (numeric).
- **Latitude**: Geographic coordinate (numeric).
- **Housing Median Age**: Median age of houses in the district (numeric).
- **Total Rooms**: Total number of rooms in the district (numeric).
- **Total Bedrooms**: Total number of bedrooms in the district (numeric).
- **Population**: Number of people residing in the district (numeric).
- **Households**: Number of households in the district (numeric).
- **Median Income**: Median income of households in the district (numeric).
- **Median House Value**: Median value of houses in the district (numeric).
- **Ocean Proximity**: Categorical attribute indicating proximity to the ocean (nominal).

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY
...	...	...	...	...	...	...	...	...	...	...
20635	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	1.5603	78100.0	INLAND
20636	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	2.5568	77100.0	INLAND
20637	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	1.7000	92300.0	INLAND
20638	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	1.8672	84700.0	INLAND
20639	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	2.3886	89400.0	INLAND

20640 rows × 10 columns

Figure 1: Data preview

1.2 Hypothesis Construction

Classification hypothesis Looking at the data description, there is only one data that is suitable for classification, namely *Ocean Proximity*. In particular, we would like to know whether it is possible to classify the block groups' proximity to the sea given other attributes. If yes, which attribute plays the most important role in classifying the proximity? At first glance, both *longitude* and *latitude* seem to be the most important information when the block's ocean proximity is inferred as they are the absolute coordinates of the block. The confirmation of the hypothesis can be found in Section 3.1. By classifying the ocean proximity of the block, real estate clients can utilize the given information to see whether the place fits their preferences.

Regression Hypothesis Looking at the data description, there is a variety of options to conduct regression on. In this project, we are interested in finding out whether it is possible to predict the house price (house value) given other attributes. If yes, which attribute plays the most important role in predicting the median house value? At first glance, the median income of the neighborhood and the ocean proximity might have larger influences on the median house value than others as rich people tend to buy a more expensive house compared to ones with lower wages. The confirmation of the hypothesis can be found in Section 3.1. As Housing price prediction is a crucial aspect of real estate and financial planning, predicting housing prices is not only valuable for homebuyers and sellers but also for investors, lenders, and policymakers.

Clustering Hypothesis In this section, we are interested in finding out if the given census block groups form any meaningful clusters on the high dimensional space given by the attributes. One possible segmentation of the entries could be based on their economic groups, i.e., high-paying, middle-class, or poor neighborhoods. The confirmation of the hypothesis can be found in Section 3.1. Clustering the data points helps identify neighborhoods with similar housing dynamics, uncover outliers, and enhance decision-making processes for both buyers and sellers.

2 Exploratory Data Analysis (EDA)

2.1 Data Visualization

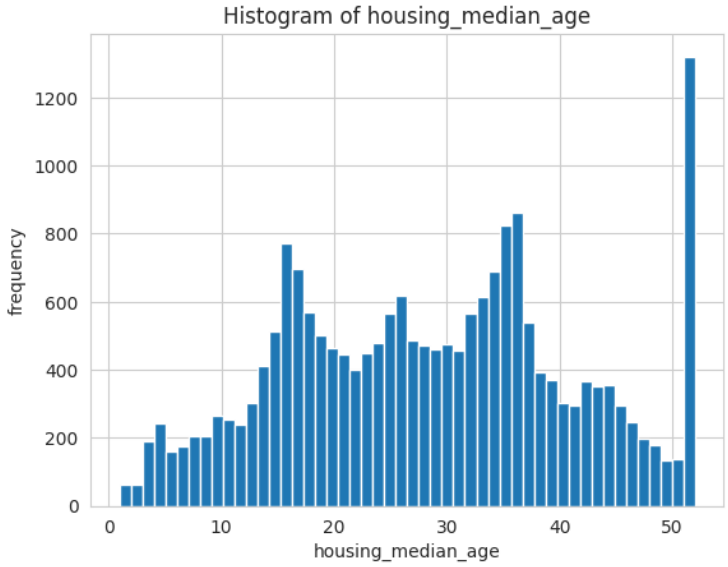

Figure 2: Housing_median_age distribution histogram

First, the histograms of all attributes that are not 'ocean_proximity' which contains texted data are examined to see the data distributions. These two histograms below 2, 3, for example, help us to see that realistic data contribution is usually not normal distribution at all, they are quite skewed and there

probably are some outliers. For the outliers of the skewed features (total_rooms, total_bedrooms) we argued that there could be the total amount of a housing building, not just a private house so we should not remove or transform it. Drawing the graphs of population, latitude, and longitude helps to see the graph of the California region and the population distribution which inspires us to do the classification work later. For example, as we can see from the graph 4, the population is more densely distributed on the east coast of California than on the west coast, which is closely related to the coordinates of the house, so house prices will be higher when the demand for housing is higher, which are all helpful for classification the 'ocean_proximity' categories.

Taking a look at the correlation among ["median_house_value", "median_income", "total_rooms", "housing_median_age"] in graph 5, we can see that, there is a high correlation between median_house_value and median_income.

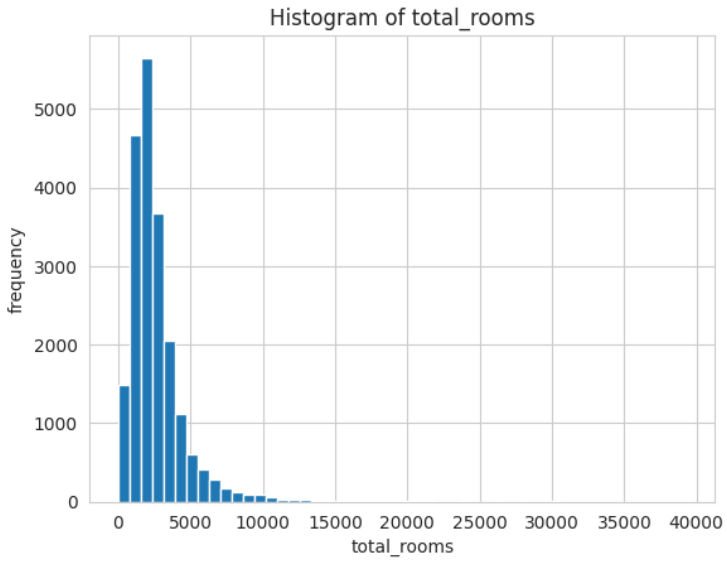

Figure 3: Total_rooms distribution histogram

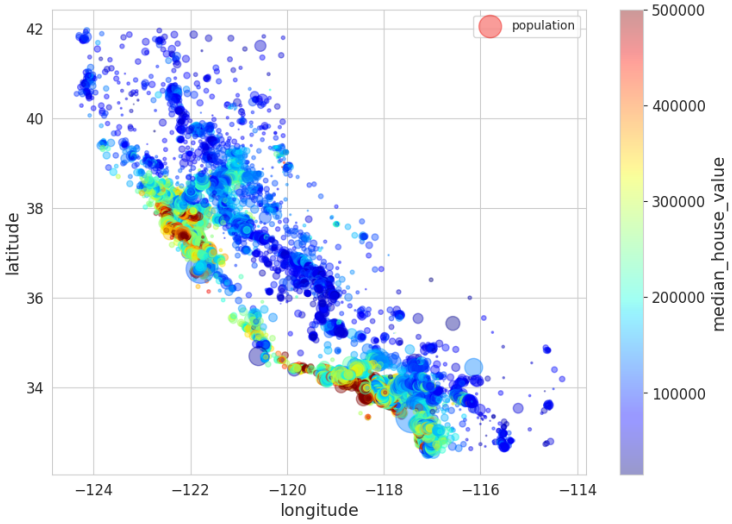

Figure 4: California population distribution.

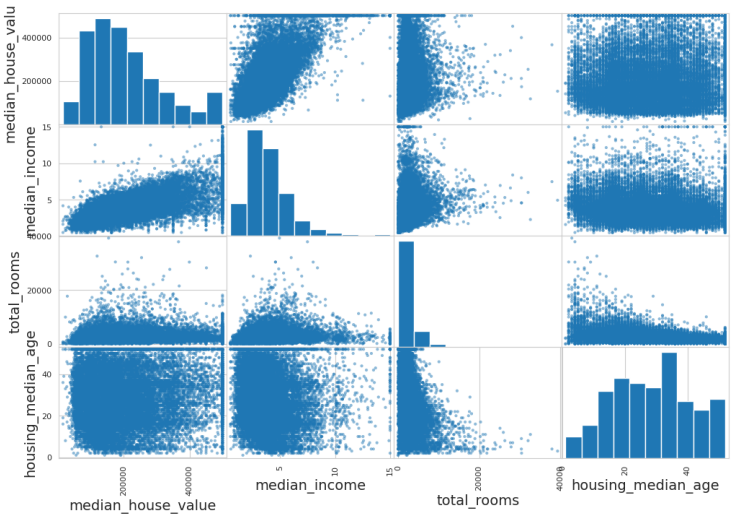

Figure 5: Correlation among 4 data attributes.

2.2 Data preproessing

First, we want to make sure that in the data there is no 'NaN' values. The statistics showed that there are no missing values in other attributes except total_bedrooms with 207 missing values, which we suppose that this is Missing completely at random so just fill the 'NaN' with the median values of the total_bedrooms attribute.

Besides, rather than using the raw data of total_rooms and, total_bedrooms, we prefer to use some more meaningful features, for example, rooms_per_household and bedrooms_per_room.

Finally, we do the feature calling by using the StandardScaler, to ensure that all features contribute equally to the computation of distances, and similarities during the training of the models later.

We also do one-hot encoding for ocean_proximity to prepare data for the classification part in figure 6.

ocean_proximity_<1H OCEAN	ocean_proximity_INLAND	ocean_proximity_ISLAND	ocean_proximity_NEAR BAY	ocean_proximity_NEAR OCEAN
0	0	0	1	0
0	0	0	1	0
0	0	0	1	0
0	0	0	1	0
0	0	0	1	0
0	0	0	1	0
0	0	0	1	0
0	0	0	1	0
0	0	0	1	0
0	0	0	1	0

Figure 6: One-hot encoding for ocean_proximity.

2.3 Dimensionality reduction PCA

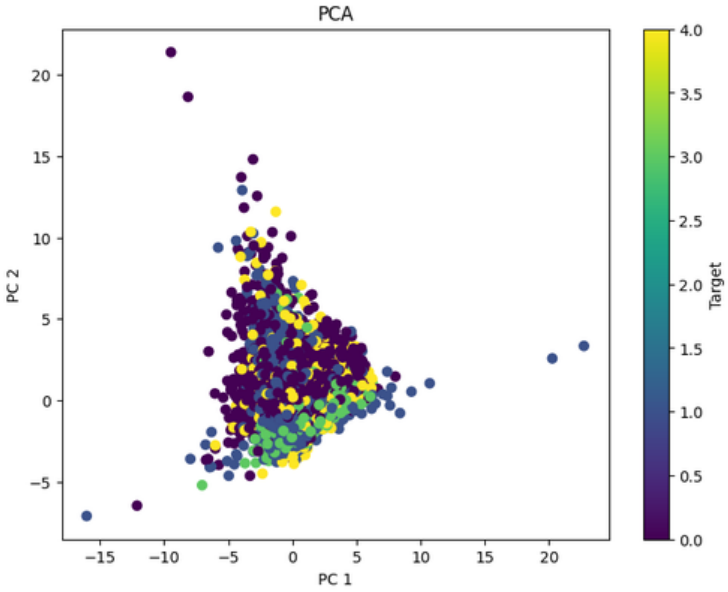

Figure 7: Linear PCA.

As we have a lot of numbers of features, the amount of data required to generalize well also increased exponentially. PCA can help to reduce the number of features while retaining the most important information and still improving the performance of the training process and making it computationally efficient. And we try with the Linear PCA and Kernel PCA, in graph 7 and 8 respectively. As we can see, both methods cannot separate the data at all, all the parts are overlapped each other. Because of this reason, we may not use PCA results for later analysis.

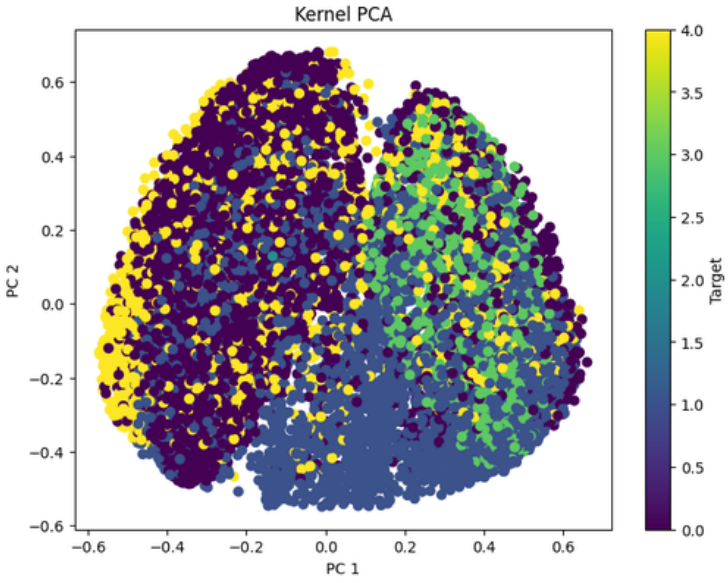

Figure 8: Kernel PCA.

3 Modeling

3.1 Classification

For this classification problem, we compare the performance among multiple models: Linear Support Vector Machine (SVM), Soft Margin SVM, Kernel SVM, Decision Tree, Random Forest, and Linear Discriminant Analysis.

Before conducting the classification, we normalize all of the attributes except the categorical attribute by using Standard Normalization.

Reasons choosing each model Linear SVM is chosen for its effectiveness in separating data into classes using a hyperplane while maximizing the margin. It works well when the classes are linearly separable. Soft Margin SVM is an extension of Linear SVM that allows for some misclassification, making it more robust when dealing with noisy or overlapping data. Kernel SVM is selected to handle non-linearly separable data. It employs kernel functions to map data into a higher-dimensional space, where it becomes linearly separable. Decision Trees are interpretable and easy to visualize. They can capture non-linear relationships in the data and are suitable for problems with complex decision boundaries. Random Forest is an ensemble method that combines multiple decision trees to improve accuracy and reduce overfitting. It is known for its robustness and ability to handle high-dimensional data. Linear Discrimination Analysis is a dimensionality reduction technique that can also be used for classification. It projects the data into a lower-dimensional space while maximizing class separability, making it useful for multi-class classification. Each model has its strengths and weaknesses, and the goal of comparing them is to determine which one performs best on the ocean proximity classification problem.

Model selection and hyper-parameter For Linear SVM, we use the default hyper-parameters given by `scikit-learn` and use the *linear* kernel. For Soft Margin SVM, we also use the default hyper-parameters with *linear* kernel and a small margin of $C = 0.95$. For Kernel SVM, we also use the default hyper-parameters with the Radial Basis Function kernel. For the Decision Tree, we grow the tree with the maximum depth of 17. As for the Random Forest, we build the forest with 128 random estimators. Lastly, for Linear Discriminant Analysis, we use the default hyper-parameters given by `scikit-learn`. Table 1 summarizes the models' hyper-parameters.

Model	Hyper-parameters
Linear SVM	Default
Soft Margin SVM	$C = 0.95$
Kernel SVM	RBF kernel
Decision Tree	`max_depth=17`
Random Forest	`n_estimators=128`
Linear Discriminant Analysis	Default

Table 1: Classification model's hyper-parameters

Evaluation We conduct the experiment on all six models on the California Housing Dataset, and the results are summarized in Table 2.

Model	Accuracy	Precision	Recall	F1
Linear SVM	79.31%	57.06%	55.23%	53.05%
Soft Margin SVM	79.36%	57.10%	55.25%	53.08%
Kernel SVM	85.56%	66.23%	64.07%	64.21%
Decision Tree	**97.46%**	**97.44%**	**97.80%**	**97.62%**
Random Forest	96.51%	76.63%	76.83%	76.71%
Linear Discriminant Analysis	75.90%	55.45%	53.07%	50.92%

Table 2: Models' performance

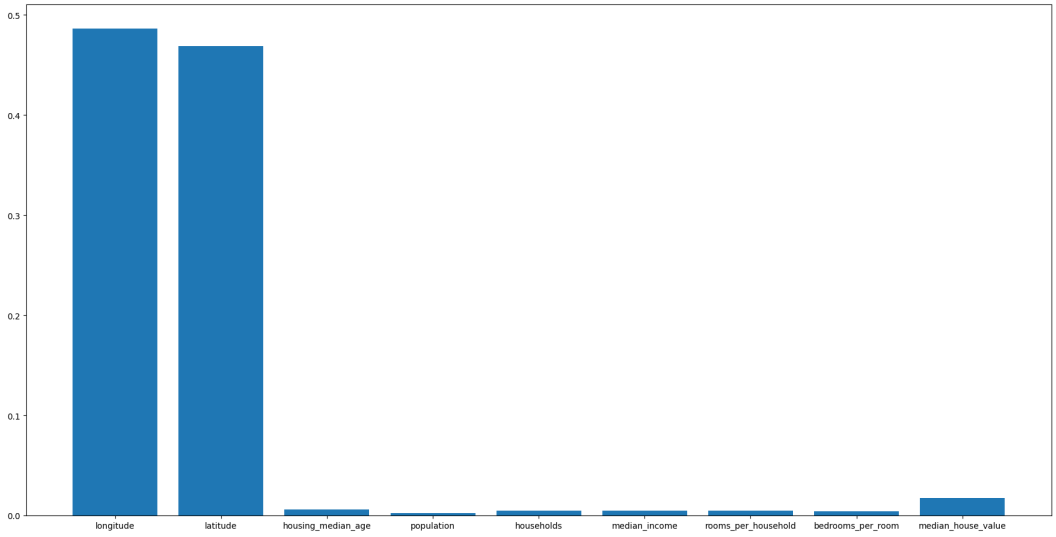

Figure 9: Classification Feature Importance

Explanations of the results Both Linear SVM and Soft Margin SVM achieve similar levels of accuracy, precision, recall, and F1-score, with an accuracy of around 79%. These models perform reasonably well but may not capture complex relationships in the data. The Kernel SVM model outperforms Linear SVM and Soft Margin SVM with an accuracy of approximately 85.56%. It demonstrates better performance in capturing non-linear relationships within the dataset. The Decision Tree model stands out as the top-performing model with an accuracy of 97.46%. It also achieves high precision, recall, and F1-score, indicating its ability to create a highly accurate classification model for the California Housing Dataset. Surprisingly, Random Forest performs worse than Decision Tree with slightly lower scores in all metrics. Linear Discriminant Analysis has the lowest accuracy among the models at 75.90%. It shows lower precision, recall, and F1-score compared to other models, suggesting it may not be the most suitable choice for this dataset. Figure 9 shows the features' importance when classifying using a Decision Tree. As expected, Both longitude and latitude have the most importance and they are about equally important.

3.2 Regression

For this regression problem, we compare the performance among multiple models: Linear Regression, Ridge Regression, Lasso Regression, Multi-Layer Perceptron, Decision Tree, and Random Forest.

Before conducting the classification, we one-hot encode the categorical attribute *ocean proximity* into a bit-vector of dimension 5. We also normalize all of the attributes except the categorical attribute by using Standard Normalization.

Reasons choosing each model Linear Regression serves as a baseline model and provides a simple and interpretable way to model the relationship between features and the target variable. It was chosen for its ease of implementation and understanding. Ridge Regression is used to mitigate multicollinearity and overfitting by adding a regularization term to the linear regression model. It helps improve model stability and generalization. Lasso Regression is selected for its ability to perform feature selection by introducing L1 regularization. It can effectively reduce the impact of irrelevant features on the model's predictions. Neural networks, specifically the Multi-Layer Perceptron (MLP), are chosen for their capability to capture complex, non-linear relationships in the data. They excel in modeling intricate patterns. Decision Trees are interpretable and well-suited for problems with non-linear relationships. They can handle a mix of numerical and categorical features and are useful for variable importance analysis. Random Forest is an ensemble method that combines multiple decision trees, providing robustness against overfitting and high prediction accuracy. It is particularly useful when dealing with high-dimensional data.

Model selection and hyper-parameter For Linear Regression, we use the default hyper-parameters given by `scikit-learn` and use the *linear* kernel. For Ridge Regression, we also use the default hyper-parameters with $\alpha = 100$. For Lasso Regression, we also use the default hyper-parameters with $\alpha = 1000$. For Multi-Layer Perceptron, we use `ReLU` activation function, the hidden layer sizes are 32, 64 and 32, and the maximum number of iterations is 1000. For the Decision Tree, we grow the tree with the maximum depth of 10. As for the Random Forest, we build the forest with 32 random estimators. Table 3 summarizes the models' hyper-parameters.

Model	Hyper-parameters
Linear Regression	Default
Ridge Regression	$\alpha = 100$
Lasso Regression	$\alpha = 1000$
Multi-Layer Perceptron	`ReLU` activation, `hidden_size=(32, 64, 32)`, and 1000 iterations
Decision Tree	`max_depth=10`
Random Forest	`n_estimators=32`

Table 3: Regression model's hyper-parameters

Evaluation We conduct the experiment on all six models on the California Housing Dataset, and the results are summarized in Table 4.

Model	$\mathbf{R^2}$ score
Linear Regression	63.09%
Ridge Regression	63.34%
Lasso Regression	63.53%
Multi-Layer Perceptron	81.32%
Decision Tree	73.03%
Random Forest	**82.18%**

Table 4: Models' performance on the regression task

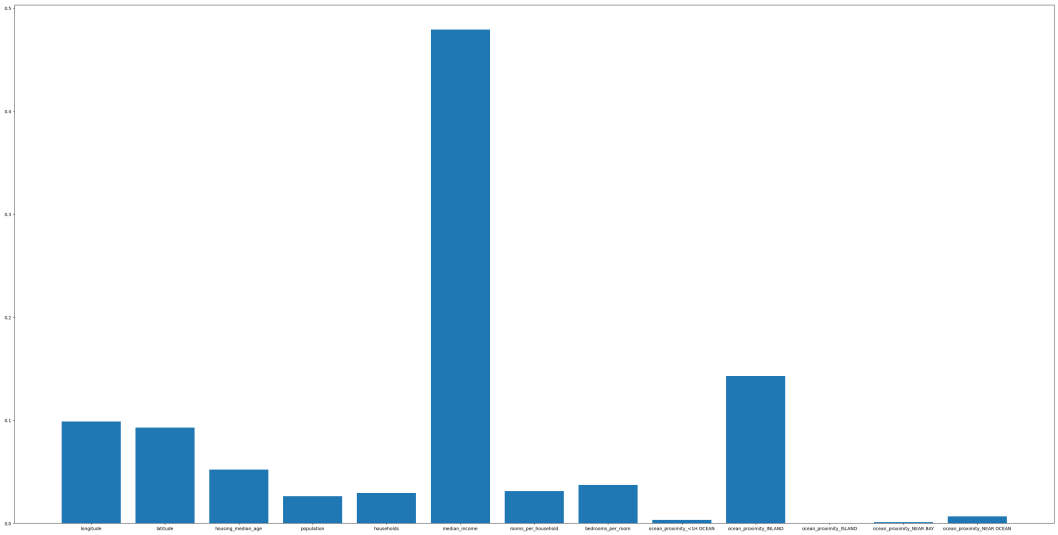

Figure 10: Regression Feature Importance

Explanations of the results These linear regression models achieve similar R^2 scores, ranging from approximately 63.09% to 63.53%. They provide a baseline for the regression task but may not capture complex relationships in the data. The MLP neural network model outperforms the linear regression models with an R^2 score of 81.32%, demonstrating the ability to capture non-linear

relationships within the dataset with the `ReLU` activation function. The Decision Tree model performs well with an R^2 score of 73.03%. It can capture non-linear relationships while being an interpretable model. Random Forest outperforms all other models with a slightly higher R^2 score of 82.18% compared to MLP thanks to its combination of multiple decision trees to improve accuracy and its ability to capture complex relationships in the data. Figure 10 shows the features' importance when classifying using a Random Forest. As expected, median income has the most importance coeficient.

3.3 Clustering

For this clustering problem, we compare the cluster results obtained from three different algorithms: K-means, DBSCAN, and Hierarchical Clustering.

Reasons choosing each model K-means is a widely used clustering algorithm known for its simplicity and efficiency. It is chosen for its ability to partition data into clusters based on the similarity of data points to centroids, making it suitable for datasets with well-defined clusters. DBSCAN is selected for its ability to identify clusters of arbitrary shapes and effectively handle noisy data. It is density-based, which means it can discover clusters of varying sizes and shapes without requiring the number of clusters to be specified in advance. Hierarchical Clustering is chosen for its capability to create a hierarchical representation of clusters. It can be used to explore the data at different levels of granularity, which is valuable for understanding the hierarchical structure within the data.

Model selection and hyper-parameter For the KMeans, after checking some cluster values from 2-10, we choose 6 as the best which can show the separation of different clusters quite well. For the DBSCAN algorithm, we choose the eps = 0.3 and the number of min_samples as 5. For the hierarchical clustering, the method ward means that in an agglomerative hierarchical clustering algorithm, we want to merge clusters that lead to the smallest increase in overall variance within the new cluster. Table 5 summarizes the models' hyper-parameters.

Model	Hyper-parameters
KMeans	n_clusters = 6
DBSCAN	eps = 0.3, min_samples = 5
Hierarchical Clustering	method = ward

Table 5: Clustering model's hyper-parameters

Evaluation We conduct the experiment on all three models on the California Housing Dataset, and the results can be found in Figure 11, Figure 12, and Figure 13.

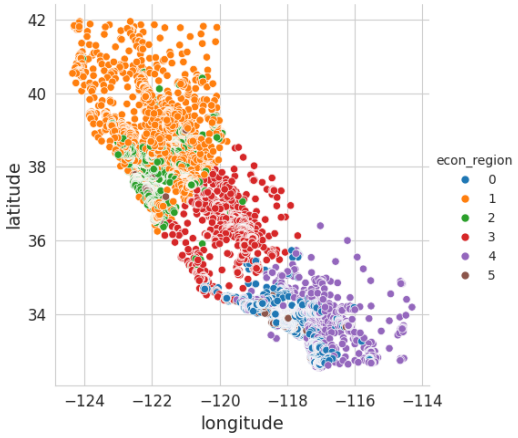

Figure 11: K-Means clustering

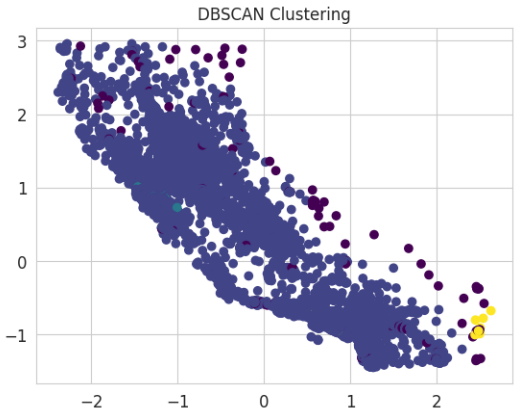

Figure 12: DBSCAN clustering

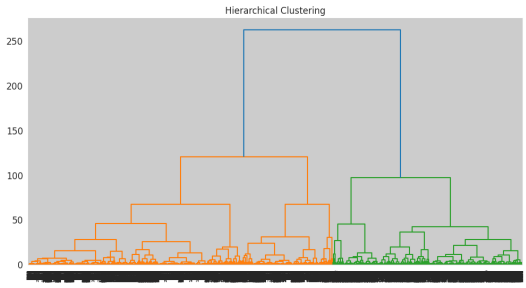

Figure 13: Hierarchical clustering

Explanations of the results DBSCAN does not seem to yield any meaningful clusters as most of them overlap. Hierarchical clustering produces a dendrogram that shows the hierarchical order of the cluster, but it cannot easily be interpreted. Out of the three, K-means seems to provides the best clusters as they are somewhat moderately segmentable and clearly shows each region of the city.

4 Discussion and Conclusion

Discussion In the classification task, we compared six different models, including Linear SVM, Soft Margin SVM, Kernel SVM, Decision Tree, Random Forest, and Linear Discriminant Analysis. Among these, the Decision Tree model demonstrated exceptional performance, achieving the highest accuracy of 97.46%. It excelled in both precision and recall, indicating its ability to create highly accurate classification models. Kernel SVM also performed well in capturing non-linear relationships within the data. For the regression task, we also evaluated six models, including Linear Regression, Ridge Regression, Lasso Regression, Multi-Layer Perceptron (MLP), Decision Tree, and Random Forest. The Random Forest regression model emerged as the top performer, achieving the highest R^2 score of 82.18%. It slightly outperformed the linear regression models and demonstrated its capability to capture complex relationships in the data. As for the clustering, K-means emerges as the best tool for this task, while DBSCAN has difficulty in capturing the relationship in the data as the data might not be in the expected complex shape in its algorithm, and Hierarchical clustering is hard to interpret.

Conclusion For the classification task, the Decision Tree model stands out as the most accurate and reliable choice for the California Housing Dataset. It excels in both accuracy and interpretability, making it a strong candidate for this type of problem. In the regression task, the Random Forest regression model outperforms other models, providing the highest R^2 score and thus the most accurate predictions for housing prices. It showcases the power of ensemble methods in capturing complex relationships within the data. Lastly, in clustering task, K-means seems to be the best tool for this dataset.