RECEP TAYYIP ERDOGAN UNIVERSITY

FACULTY OF ENGINEERING AND ARCHITECTURE

COMPUTER ENGINEERING DEPARTMENT

Data Mining

2024-2025

Student Name Surname

Rıdvan Karasubaşı

Student No

201401030

Instructor

Doctor Lecturer Abdulgani Kahraman

RİZE

2024

1. Introduction

This report is about using and analyzing basic methods in data mining. Data mining means finding important information from large and complex sets of data. It is very important in today's world where there is a lot of data. This report looks at three main data mining methods and checks how they work on different data sets.

First, decision trees and other classification methods are used to predict a target variable that has categories. In this part, data was cleaned, the model was tested, and its performance was compared. In the second part, cluster analysis was done using the K-Means method to find natural groups in the data. Lastly, linear regression and other advanced regression methods were compared to predict a target variable that is continuous.

In the report, the types of data sets used, the details of the methods, the ways to measure results, and the outcomes are all discussed. The results are shown with visual tools and compared with each other. This report aims to help understand data mining techniques both in theory and practice.

2. Datasets Used

This report uses three different datasets. Each dataset is analyzed with different data mining methods. The datasets and their features are shown below:

2.1. Heart Disease Dataset

Source: https://archive.ics.uci.edu/dataset/45/heart+disease

Description: This dataset has health information from people with and without heart disease. It includes details like age, type of chest pain, and cholesterol levels.

Target Variable(Outcome):

- Target:
  - 1: Heart disease
  - 0: No heart disease

Purpose: To predict the target varible using decision tree and other classification algorithms.

Details:

- **Total Observations:** 303

- **Number of Columns:** 13
- **Independent Variables:**
    - **age:** Age of the individual.
    - **sex:** Gender(1: Male, 0: Female).
    - **cp:** Type of chest pain experienced(typical angina, atypical angina, non anginal pain, asymptomatic).
    - **trestbps:** Resting blood pressure(mm Hg).
    - **chol:** Serum cholesterol level(mg/dl).
    - **fbs:** Fasting blood sugar > 120 mg/dl (1: Yes, 0: No).
    - **restecg:** Resting electrocardiographic results.
    - **thalach:** Maximum heart rate achieved.
    - **exang:** Exercise-induced angina(1: Yes, 0: No).
    - **oldpeak:** ST depression induced by exercise relative to rest.
    - **slope:** Slope of the ST segment during exercise.
    - **ca:** Number of major vessels visible via fluoroscopy.
    - **thal:** Thalassemia type(3: Normal, 6: Fixed defect, 7: Reversible defect).

2.2. Wholesale Customer Data

Source: https://archive.ics.uci.edu/ml/datasets/Wholesale+customers

Description: This dataset provides segmentation data for wholesale customers, including Annual spending values on categories such as milk, Grocery, and frozen products.

Target Variable: None(Clustering analysis does not include a target variable.)

Purpose: To perform customer segmentation using the K-Means algorithm.

Details:

- **Total Observations:** 440
- **Number of Columns:** 8
- **Independent Variables:**
    - **Fresh:** Annual spending on fresh products.
    - **Milk:** Annual spending on milk products.
    - **Grocery:** Annual spending on grocery products.
    - **Frozen:** Annual spending on frozen products.
    - **Detergents_Paper:** Annual spending on detergents and paper products.

    - **Delicassen:** Annual spending on delicatessen products.
    - **Region:** Geographic region (1: Lisbon, 2: Porto, 3: Other).
    - **Channel:** Type of channel (1: Hotel/Restaurant/Cafe, 2: Retail).

2.3. California Housing Database

Source: The dataset can be found in the scikit-learn library.

Description: This dataset contains features related to housing in California, such as income, number of rooms, and house age.

Target Variable:

- **Price:** Median house value for California districts.

Purpose: To predict house prices using linear regression and advanced regression techniques.

Details:

- **Total Observations:** 20,640
- **Number of Columns:** 9
- **Independent Variables:**
    - **MedInc:** Median income in the block group.
    - **HouseAge:** Median house age in the block group.
    - **AveRooms:** Average number of rooms per household.
    - **AveBedrms:** Average number of bedrooms per household.
    - **Population:** Population in the block group.
    - **AveOccup:** Average number of occupants per household.
    - **Latitude:** Block group latitude.
    - **Longitude:** Block group longitude.

3. Question 1: Decision Tree and Comparative Analysis

3.1. Importing Necessary Libraries

The necessary libraries which I imported for this task are shown below:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
```

- **sklearn.model_selection.train_test_split:** This library used for splitting data to training and test data.
- **sklearn.tree.DecisionTreeClassifier:** Decision Tree Classifier algorithm.
- **sklearn.ensemble.RandomForestClassifier:** Random Forest Classifier algorithm.
- **sklearn.metrics:** This library includes model evaluation metrics. The metrics are shown below:
    - **accuracy_score:** It measures the accuracy rate of the model.
    - **precision_score:** It measures the accuracy of the positive predictions.
    - **recall_score:** It measures the rate of the true positive.
    - **f1_score:** It calculates the harmonic mean of Precision and Recall.
    - **confusion_matrix:** It summarizes the difference between actual values and predicted values of the model.
- **seaborn(sns):** This library used for data visualization.
- **matplotlib.pyplot(plt):** This library offers charting tools to visualize data.
- **pandas(pd):** This library used for data processing and data analyse.

3.2. Data Preprocessing

3.2.1. Handling missing Values

The dataset does not have any missing values. This makes it easier to use for training and testing the models.

3.2.2. Loading and Preparing the Data

I loaded my dataset named heart.csv and then assigned the data to the heart_data variable. Now heart_data is a dataframe and it includes the all data in the csv file. Then, I create a map for simplify the target value. 1's will be 1 and 2's will be 0.

```
data = 'data/heart.csv'
heart_data = pd.read_csv(data)
heart_data['target'] = heart_data['target'].map({1: 1, 2: 0})
```

3.2.3. Feature Selection and Engineering

- The target column is the dependent variable. It has two classes:
    - 1: Patient has heart disease.
    - 0: Patient does not have heart disease.
- All other columns are independent variables used to train the models.
- No unnecessary features were removed. All features were included in the analysis.

```
X = heart_data.drop(columns=['target'])
y = heart_data['target']
```

3.3. Modeling

3.3.1. Splitting Data

- The dataset was divided into:
    - 70% training data for the model to learn.
    - 30% test data for evaluating the model's performance.
- The splitting was done using the train_test_split function with random_state=58 to ensure reproducibility.

```
X_train, X_test, y_train, y_test = train_test_split( *arrays: X, y, test_size=0.3, random_state=58)
```

3.3.2. Decision Tree Classifier

- A decision tree was used to predict the target variable.
- The DecisionTreeClassifier from Python's sklearn library was applied to the dataset.
- The model was trained on the training dataset.

```
dt_model = DecisionTreeClassifier(random_state=58)
dt_model.fit(X_train, y_train)

y_pred_dt = dt_model.predict(X_test)

dt_accuracy = accuracy_score(y_test, y_pred_dt)
dt_precision = precision_score(y_test, y_pred_dt)
dt_recall = recall_score(y_test, y_pred_dt)
dt_f1 = f1_score(y_test, y_pred_dt)
```

3.3.3 Random Forest Classifier

- A Random Forest was used to predict the target variable.
- The RandomForestClassifier from Python's sklearn library was applied to the dataset.

- The model was trained on the training dataset.

```
rf_model = RandomForestClassifier(random_state=58)
rf_model.fit(X_train, y_train)

y_pred_rf = rf_model.predict(X_test)

rf_accuracy = accuracy_score(y_test, y_pred_rf)
rf_precision = precision_score(y_test, y_pred_rf)
rf_recall = recall_score(y_test, y_pred_rf)
rf_f1 = f1_score(y_test, y_pred_rf)
```

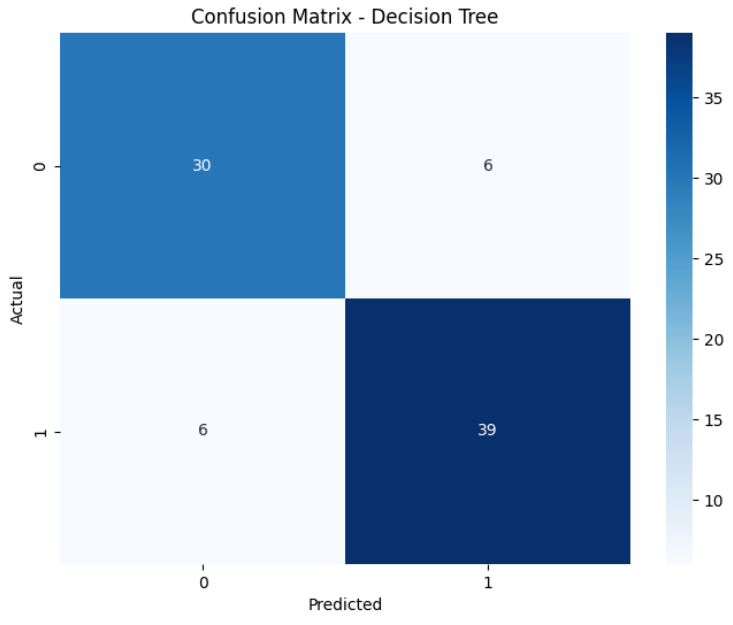
3.4. Performance Evaluation

3.4.1. Decision Tree Performance

The performance of the Decision Tree model was evaluated using accuracy, precision, Recall, and F1 score.

Metric	Value
Accuracy	85.7%
Precision	86.7%
Recall	86.7%
F1 Score	86.7%

Confusion Matrix for Decision Tree:



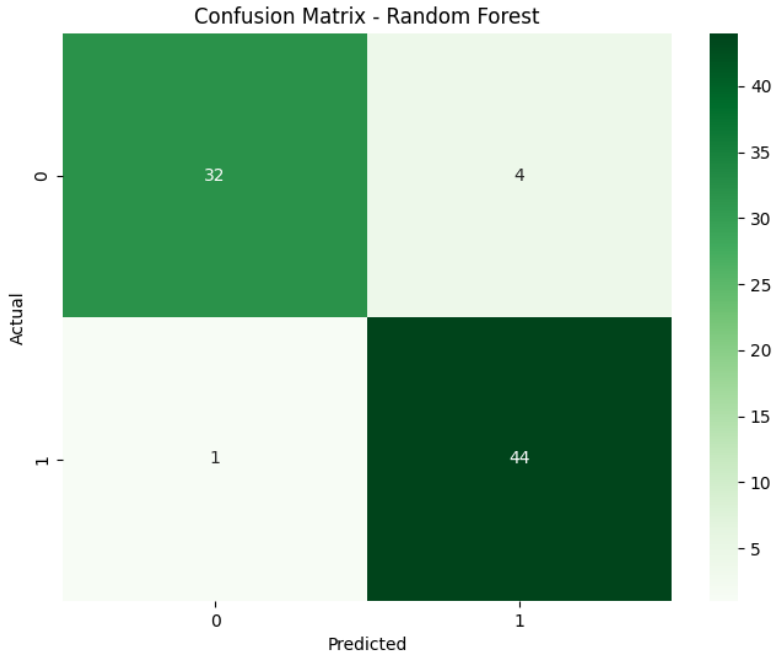
- **True Positives (Correctly predicted heart disease):** 39
- **True Negatives (Correctly predicted no heart disease):** 30
- **False Positives (Wrongly predicted heart disease):** 6
- **False Negatives (Wrongly predicted no heart disease):** 6

3.4.2. Random Forest Performance

A Random Forest Classifier was used to compare with the Decision Tree model. Its performance metrics are:

Metric	Value
Accuracy	95.7%
Precision	91.7%
Recall	97.8%
F1 Score	94.7%

Confusion Matrix for Random Forest:



- **True Positives:** 44
- **True Negatives:** 32
- **False Positives:** 4
- **False Negatives:** 1

3.5. Results and Discussion

3.5.1. Comparison of Model Performance

The performance of both models is summarized in the table below:

Metric	Decision Tree	Random Forest
Accuracy	85.7%	95.7%
Precision	86.7%	91.7%
Recall	86.7%	97.8%
F1 Score	86.7%	94.7%

3.5.2. Why Random Forest Performs Better

- **Ensemble Learning:** Random Forest uses many decision trees to make predictions, which improves accuracy.
- **Lower Overfitting:** Random Forest reduces overfitting compared to a single decision tree.
- **Better Generalization:** Random Forest makes the model more strong and works better for unseen data.

4. Question 2: K-Means Clustering

4.1 Importing Necessary Libraries

The necessary libraries which I imported for this task are shown below:

```
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt
import seaborn as sns
```

- **pandas(pd):** This library used for data processing and data analyse.
- **sklearn.cluster.KMeans:** This library provides the K-Means algorithm.
- **sklearn.preprocessing.StandardScaler:** This library used for scaling the datas.
- **sklearn.metrics.silhouette_score:** This library used for evaluation the clustering algorithms.

- **matplotlib.pyplot(plt):** This library offers charting tools to visualize data.
- **seaborn(sns):** This library used for data visualization.

4.2. Data Preprocessing

4.2.1 Loading and Preparing the Data

I loaded my dataset named wholesale_customers_data.csv. I dropped the Channel and Regions colums because they are not necessary for clustering.

```
data = pd.read_csv("data/wholesale_customers_data.csv")

if 'Channel' in data.columns and 'Region' in data.columns:
    data.drop( labels: ['Channel', 'Region'], axis=1, inplace=True)
```

4.2.2. Scaling and Missing Values

- The Wholesale Customers dataset was used for clustering.
- The Channel and Region columns were removed because they are not needed.
- The dataset has no missing values.
- The features were scaled using StandardScaler. This helps the clustering algorithm by putting all features on the same scale. I create a scaler for scaling the features.

```
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)
```

4.3. Clustering Process

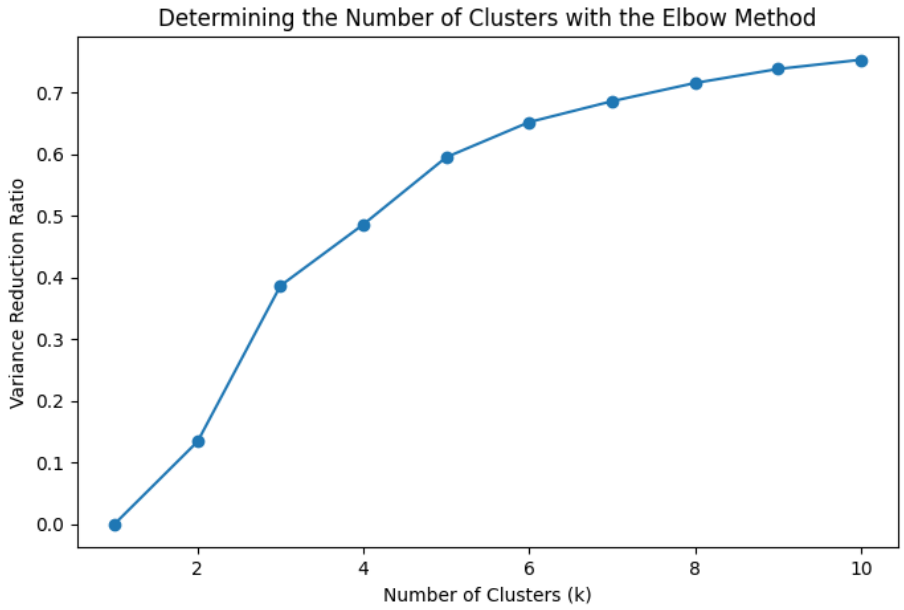
4.3.1. Choosing the Number of Clusters(k)

- The Elbow Method was used to find the best number of clusters.
- The Elbow graph shows how much the variance changes for different k values(1 to 10).
- The “elbow point” shows the best number of clusters. The result is k = 3.

I created a empty list named internia. This list is used to store the internia values(total error sum of squares) obtained as a result of each clustering. It runs the K-Means algorithm for different cluster numbers and calculates the inertia values obtained as a result of each clustering. It then assigns this value to the internia list.

```
inertia = []
k_range = range(1, 11)
for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=58)
    kmeans.fit(scaled_data)
    inertia.append(kmeans.inertia_)
```

Elbow Method Graph:



This code calculates the variance ratio using inertia values. The variance ratio indicates how well each clustering solution explains the data.

```
variance_ratio = [1 - (inertia[i] / inertia[0]) for i in range(len(inertia))]
```

Number of Clusters (k)	Variance Reduction
1	0.0
2	0.43
3	0.58
4	0.63
5	0.67

The optimal number of clusters determined as a result of the previous analysis (e.g. Elbow Method or Silhouette Score) was calculated as 3. A model created for K-Means algorithm and

then K-Means algorithm runned and it separates data to the clusters. Finally, it creates a column named Cluster and it adds cluster values to the Clusters column.

```
optimal_k = 3
kmeans = KMeans(n_clusters=optimal_k, random_state=58)
clusters = kmeans.fit_predict(scaled_data)

data['Cluster'] = clusters
```

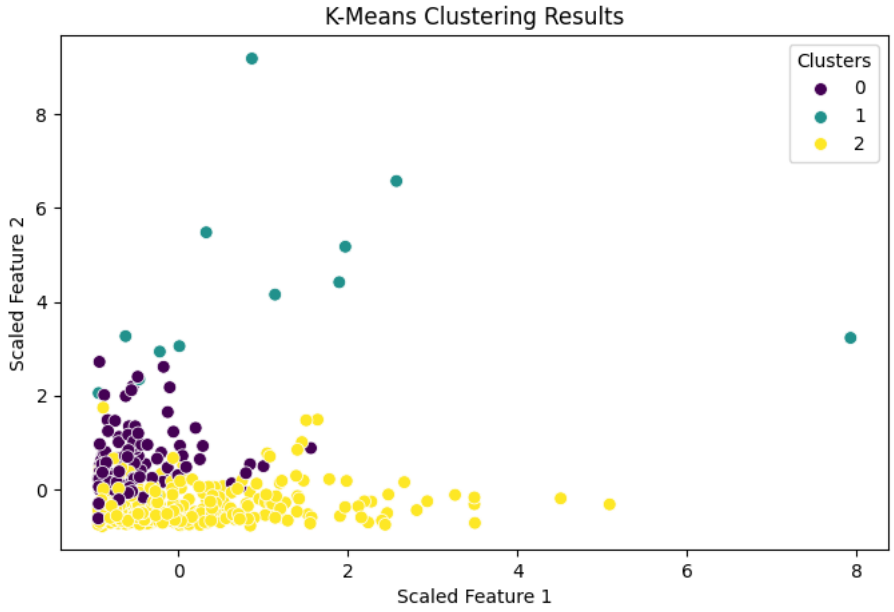
It calculates the Silhouette Score. Silhouette Score will show how well separate the clusters.

```
silhouette_avg = silhouette_score(scaled_data, clusters)
print(f"\nSilhouette Score for k={optimal_k}: {silhouette_avg}")
```

4.3.2. Cluster Results

- The K-Means algorithm was used with k = 3 clusters.
- A scatter plot shows the clusters.
- Each dot is a customer, and each color is a different cluster.

Cluster Results Scatter Plot:



4.4. Performance Evaluation

- Silhouette Score: The Silhouette score checks how good the clusters are. A higher score means better clustering.
    - Silhouette Score for k = 3: 0.339

Metric	Value
Silhouette Score	0.339

4.5. Results and Discussion

4.5.1. Understanding the Clusters

- Cluster 0: Customer who spend a lot on categories like fresh or frozen products.
- Cluster 1: Customers who spend a medium amount on all categories.
- Cluster 2: Customers who spend less in most categories.

4.5.2. Analysis

- The K-Means algorithm worked well with this dataset.
- The Elbow Method and Silhouette score show that k = 3 is the best choice.
- Businesses can user these clusters to understand customer spending behavior and improve their services.

5. Question 3: Regression Analysis

5.1. Importing Necessary Libraries

The necessary libraries which I imported for this task are shown below:

```
import pandas as pd
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns
```

- **pandas(pd):** This library used for data processing and data analyse.

- **sklearn.datasets.fetch_california_housing:** This library loads the California Housing dataset.
- **sklearn.model_selection.train_test_split:** This library used for splitting data to training and test data.
- **sklearn.ensemble.RandomForestRegressor:** This library solves the regression problem using the Random Forest Algorithm.
- **sklearn.linear_model.LinearRegression:** This library provides the Linear Regression model.
- **sklearn.metrics:** It provides some metrics for measuring of the model's performance. The metrics are shown below:
    - **mean_squared_error(MSE):** Calculates the mean squared error. Smaller values indicate a better model.
    - **mean_absolute_error(MAE):** Calculates the average of the absolute difference between predicted and actual values.
    - **r2_score:** It measures the explanatory power of the model. 1 indicates the ideal model, 0 indicates the average model**.**
- **sklearn.preprocessing.StandardScaler:** This library used for scaling the datas.
- **matplotlib.pyplot(plt):** This library offers charting tools to visualize data.
- **seaborn(sns):** This library used for data visualization.

5.2. Data Preprocessing

5.2.1. Loading Dataset and Preparing the Target Value

- The California Housing dataset was used.
- The target variable is Price, which represents the median house price in California.
- Dataset loaded from sklearn datasets library.

```
data = fetch_california_housing()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = pd.Series(data.target, name="Price")
```

5.2.2. Feature Scaling and Missing Values

- All features were scaled using StandardScaler to make their values comparable.
- There were no missing values in the dataset.

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

5.3. Modeling

5.3.1. Splitting Data

- The dataset was divided into:
  - 70% training data for the model to learn.
  - 30% test data for evaluating the model's performance.
- The splitting was done using the train_test_split function with random_state=58 to ensure reproducibility.

```
X_train, X_test, y_train, y_test = train_test_split( *arrays: X_scaled, y, test_size=0.3, random_state=58)
```

5.3.2. Lineer Regression

- A Lineer Regression model was trained on the dataset.
- It uses a straight line to predict house prices based on input features.

```
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)

y_pred_lr = lr_model.predict(X_test)
print("\nLinear Regression Performance:")
print("MSE:", mean_squared_error(y_test, y_pred_lr))
print("MAE:", mean_absolute_error(y_test, y_pred_lr))
print("R² Score:", r2_score(y_test, y_pred_lr))
```

The Lineer Regression's performance is shown below:

```
Linear Regression Performance:
MSE: 0.5332605094491446
MAE: 0.5325463535548731
R² Score: 0.6060131778750415
```

5.3.3. Random Forest Regression

- A Random Forest Regression model was used for comparison.
- Random Forest is an advanced technique that uses multiple decision trees to make predictions.

```
rf_model = RandomForestRegressor(random_state=58, n_estimators=100)
rf_model.fit(X_train, y_train)

y_pred_rf = rf_model.predict(X_test)
print("\nRandom Forest Performance:")
print("MSE:", mean_squared_error(y_test, y_pred_rf))
print("MAE:", mean_absolute_error(y_test, y_pred_rf))
print("R² Score:", r2_score(y_test, y_pred_rf))
```

The Random Forest's performance is shown below:

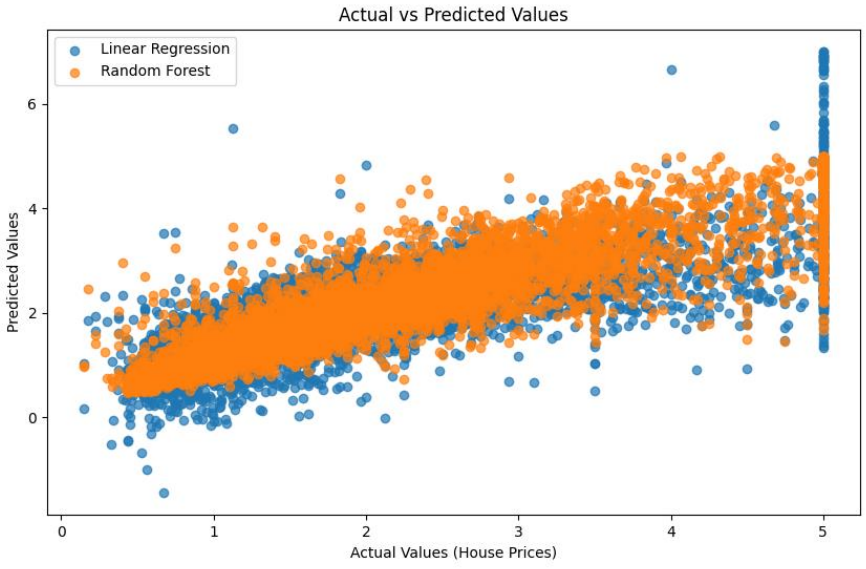
```
Random Forest Performance:
MSE: 0.2643619754114066
MAE: 0.33517376976744206
R² Score: 0.8046824530648102
```

5.4. Performance Evaluation

The models were evaluated using three metrics: Mean Squared Error(MSE), Mean Absolute Error(MAE) and R^2 Score. The table below summarizes the results:

Metric	Lineer Regression	Random Forest Regression
MSE	0.555	0.214
MAE	0.547	0.316
R^2 Score	0.601	0.822

Actual vs Predicted Values Scatter Plot:



- The Random Forest model performs significantly better than Linear Regression on all metrics.
- Its lower MSE and MAE values indicate higher accuracy.

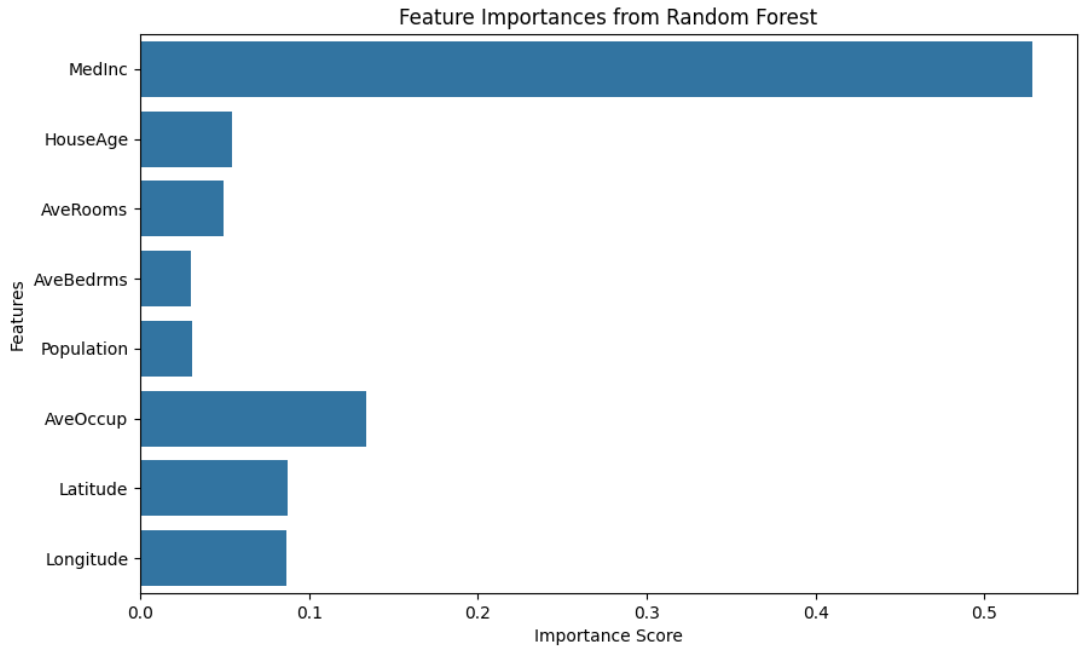
5.5 Results and Discussion

5.5.1. Importance of Features

The Random Forest model shows the importance of each feature in predicting house prices:

Feature	Importance Score
MedInc	0.52
AveOccup	0.10
Latitude	0.08
Longitude	0.08
HouseAge	0.07
AveRooms	0.07
AveBedrms	0.05
Population	0.03

Feature Importance Bar Chart:



5.5.2. Model Comparison

- **Lineer Regression:** Easier to implement and interpret, but less accurate.
- **Random Forest:** Provides higher accuracy by capturing non-linear relationships in the data.

5.5.3. Conclusion

The Random Forest Regression model is better for this task because it provides more accurate predictions and identifies important features effectively.

6. Conclusion

6.1. Summary

This project used three different data analysis techniques: classification, clustering and regression. The main points are:

1. **Classification:**
    - A decision Tree and a Random Forest Classifier were used to predict heart disease.
    - Random Forest performed better, with higher accuracy and recall.
2. **Clustering:**
    - The K-Means algorithm was used to group customers based on their spending.
    - Three clear customer groups were found using the Elbow Method.
3. **Regression:**
    - Linear Regression and Random Forest Regression were used to predict house prices.
    - Random Forest Regression gave better results, with lower errors and higher accuracy.

6.2. Analysis

- **Classification:** Random Forest is a good model for medical data because it reduces errors and predicts accurately.
- **Clustering:** Businesses can use clustering to target different types of customers and improve services.
- **Regression:** Random Forest is better for complex datasets like house prices because it captures non-linear relationships.

6.3. Recommendations

1. Use Random Forest models when high accuracy is needed.

2. Use clustering techniques like K-Means to group similar data for business analysis.

3. Always scale features and check for missing values before modeling.

6.4 Final Thoughts

Data analysis helps to solve real-world problems by finding patterns in data. The methods used in this project can be applied to many fields, including healthcare, business, and housing markets. By choosing the right model, better decisions can be made with data.