Residency Assignment report

Chandra Kiran Billingi

University of the Cumberlands

Advanced Big Data and Data Mining (MSCS-634-M40)

Dr. Satish Penmatsa

1. Dataset Description and Justification

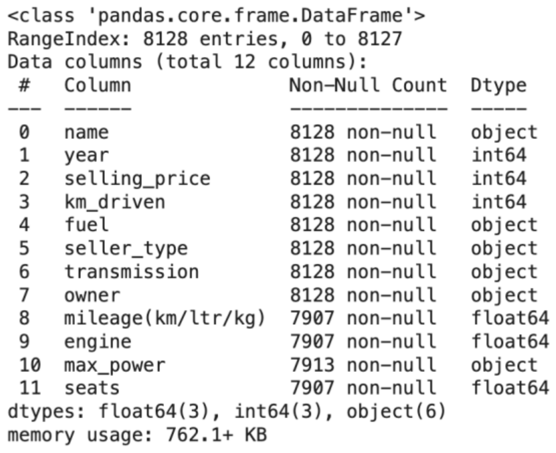
This data set was taken from Kaggle which is named a carDekho which contains the various features which was used over the various years and different makes and models

Variables in the data set :

name, year, selling_price,km_driven, fuel, seller_type, transmission, owner, mileage(km/ltr/kg), engine, max_power, seats

The main reason this data set was choosen as it has both categorical and numerical data that will used for the supervised learning techniques like regression and classification as well as the unsupervised learning methods like the clustering and association rule mining

Additionally, this data set represents a real world data and resembles a scenario that combines the business with the wide technical depth which is ideal for the data mining and data exploration. Working in real world data sets help us when there is a real world data for the better evaluation and deriving metrics

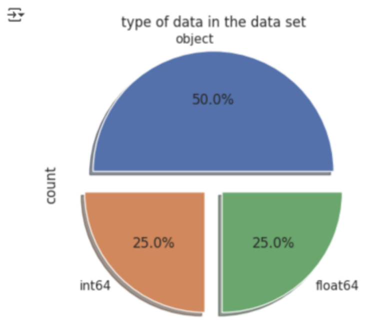


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8128 entries, 0 to 8127
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   name                8128 non-null   object 
 1   year                8128 non-null   int64  
 2   selling_price       8128 non-null   int64  
 3   km_driven           8128 non-null   int64  
 4   fuel                8128 non-null   object 
 5   seller_type         8128 non-null   object 
 6   transmission        8128 non-null   object 
 7   owner               8128 non-null   object 
 8   mileage(km/ltr/kg)  7907 non-null   float64
 9   engine              7907 non-null   float64
 10  max_power           7913 non-null   object 
 11  seats               7907 non-null   float64
dtypes: float64(3), int64(3), object(6)
memory usage: 762.1+ KB
```

The above is the description of the data which are null and which are not null from that we can see more than one column have missing values.

2. Data Preprocessing, EDA, and Feature Engineering

For the initial preprocessing I have loaded the data set and inspected its structure before data cleaning I have done a pie chart of different types of data to understand the structure

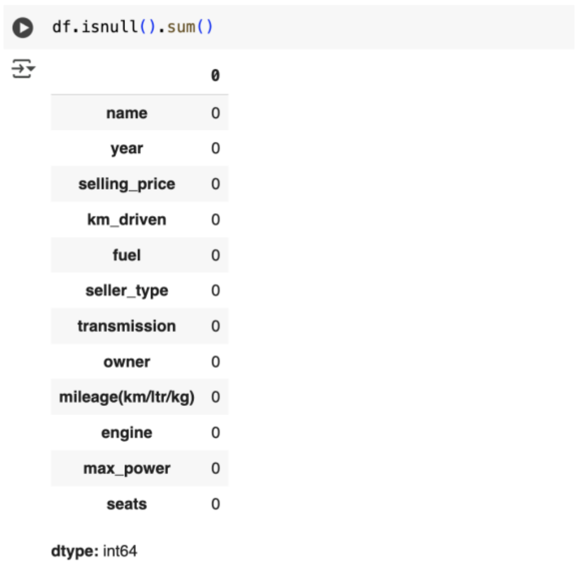


This will allow us to understand the different types in the data we can observer that the data consists of 50% object , 25% int64 types and float 64 type consists of another 25%

After that I have performed the data cleaning in which I have handled the missing values in the mileage, engine and max_power and seats

In which the mileage and engine I have replaced the empty values with mean and max_power and seats are replaced with the mode.

As all the columns are not in the same data type I have converted the mileage in to one and same type which can be useful for the model training



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

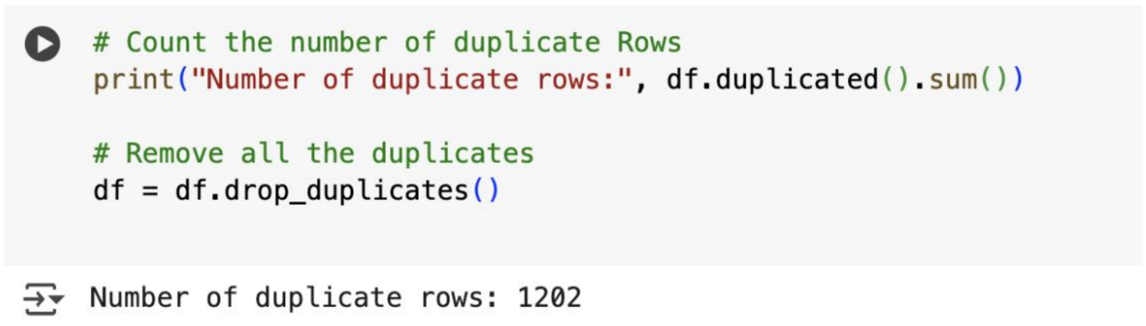
	0
name	0
year	0
selling_price	0
km_driven	0
fuel	0
seller_type	0
transmission	0
owner	0
mileage(km/ltr/kg)	0
engine	0
max_power	0
seats	0

dtype: int64

After handling missing values we can se that there are no missing values in the data which can change our model predictions.

After that the categorical features were encoded sing the one-hot encoding.

As the duplicate rows are not needed I have



```
# Count the number of duplicate Rows
print("Number of duplicate rows:", df.duplicated().sum())

# Remove all the duplicates
df = df.drop_duplicates()
```

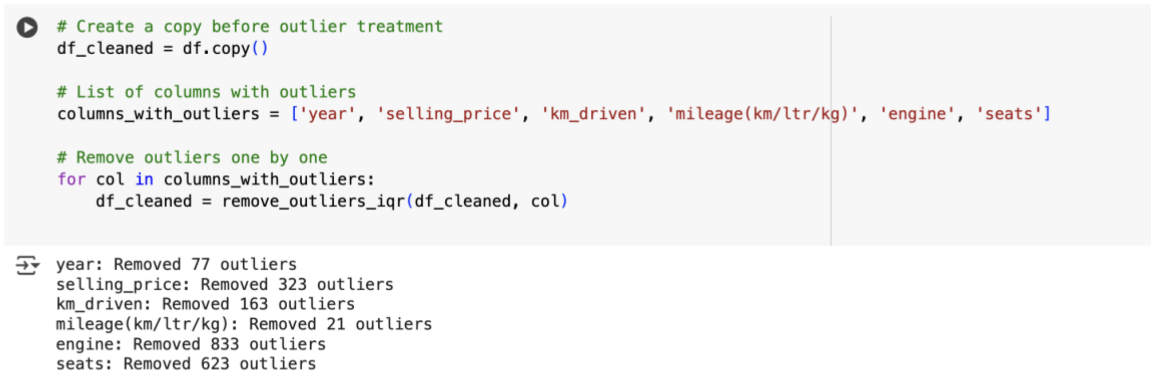
```
Number of duplicate rows: 1202
```

We have to remove the duplicate rows as well as they can change the behaviour of the model

Outlier Removal

- For the outlier Removal IQR was choosen as it is more robust and focuses on the middle part of the data which is ideal for the car sale data set

As every data consists outliers we have to get the data in the same range to make the models predict better as the outliers can cause the model to over fit or underfit. I have used the IQR method for the outlier removal and removed 77 outliers. Upon completion of the outlier removal Heat maps for getting correlation , value counts for categorical features, box plots to identify outliers, and distribution plots were generated to the understand skewness in the numerical columns and understand the data more advanced to perform better training



```
# Create a copy before outlier treatment
df_cleaned = df.copy()

# List of columns with outliers
columns_with_outliers = ['year', 'selling_price', 'km_driven', 'mileage(km/ltr/kg)', 'engine', 'seats']

# Remove outliers one by one
for col in columns_with_outliers:
    df_cleaned = remove_outliers_iqr(df_cleaned, col)
```

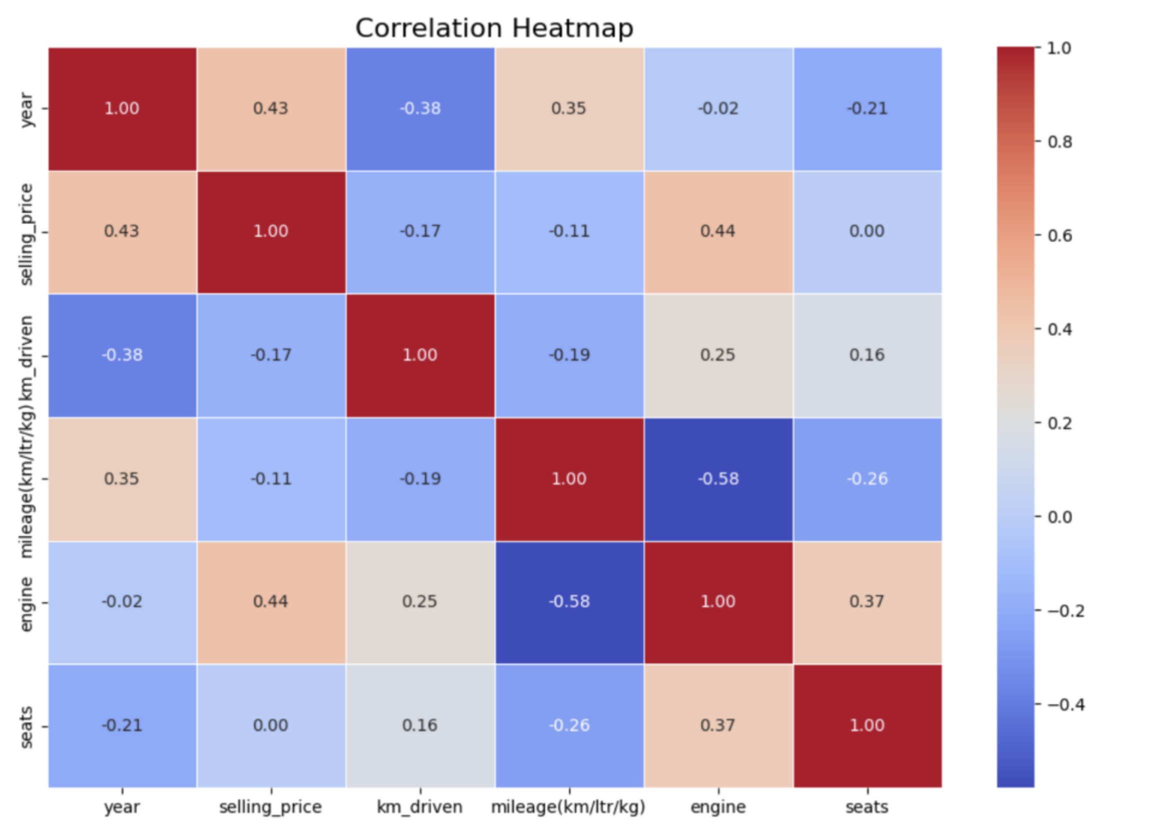
```
year: Removed 77 outliers
selling_price: Removed 323 outliers
km_driven: Removed 163 outliers
mileage(km/ltr/kg): Removed 21 outliers
engine: Removed 833 outliers
seats: Removed 623 outliers
```

- Strong correlations were observed between features such as engine and max_power, while some features showed low correlation with the target variable (selling_price).
- This insight will help in feature selection, potentially reducing dimensionality by dropping irrelevant or redundant variables.
- Certain categories (e.g., vehicle ownership types) were imbalanced, which may influence model bias. Addressing class imbalance through resampling was necessary and weighting was done for classification tasks.

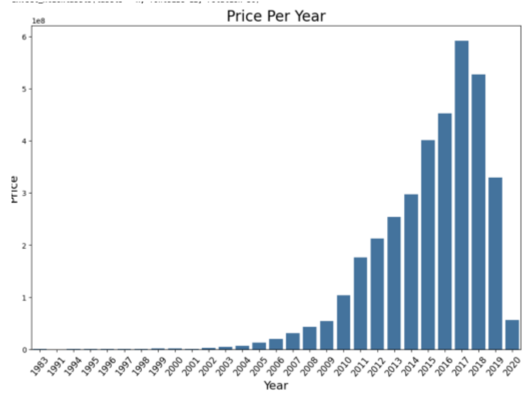
The target variable for prediction was 'selling_price'.

The data preprocessing was very important as it removes the missing values which ensure the data loss and false training of the models. And also the removal of the outliers is also important to maintain the stability

The below is the correlation heat map which shows the correlation between 2 variables from this we can observe that the mileage and the power ad highly correlevant and also selling price has 0 correlation with number of seats.



From the Year vs Price chart we can see that the prices were peaked in 2017 and 2018 and slowly reduced to same as 2009 In 2020



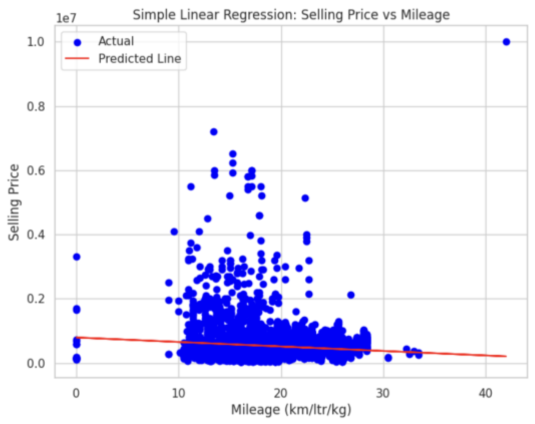
3. Modeling Results

3.1 Simple LinearRegression

Linear regression using one feature was first performed. Before training, the features were normalized using StandardScaler to ensure uniformity in feature scales. The data set was split into 5 parts and have used the cross fold technique to improve the model performance

Evaluation metrics used include Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared score.

This model doesnot performed well as it only focuses on one feature which made the curve underfit and reduced the accuracy.



The model performance was evaluated using MAE, MSE, RMSE, and R²

Avg MAE : 295558.94

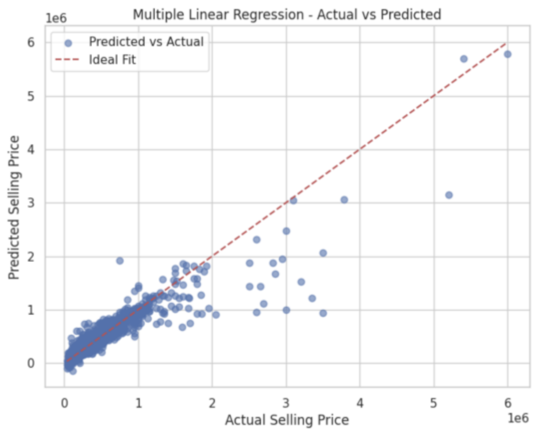
Avg MSE : 267462513462.61

Avg RMSE: 515875.99

Avg R^2 : 0.0093

3.2 Multiple Linear regression

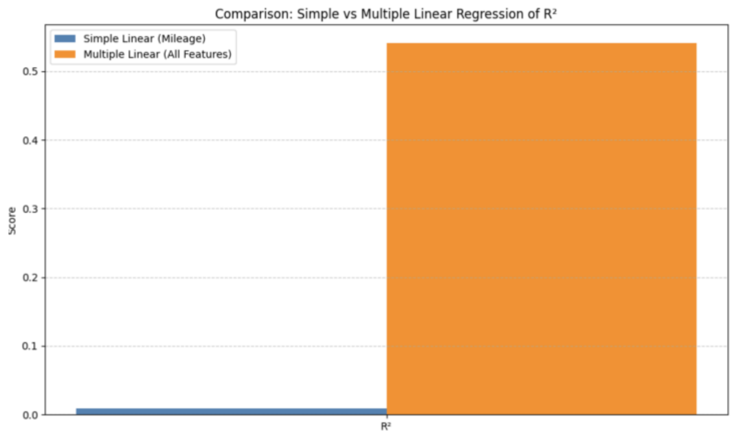
Secondly multiple linear regression is performed using all the features in the data set which predicted the selling price of the car



Avg MAE : 120006.11
Avg MSE : 114797407787.52
Avg RMSE: 317342.43
Avg R² : 0.5407

The Points are closer to the ideal line which showed improved predictive performance. Which Indicates that adding features like engine, year, fuel, and others captures more variance in the target. However some points scattered suggesting possible noise or non-linearity that linear regression can't capture.

This is the model comparision graph of Simple Linear regression vs Multiple Linear regression from the graph we can observe that the multiple linear regression outperformed simple linear regression

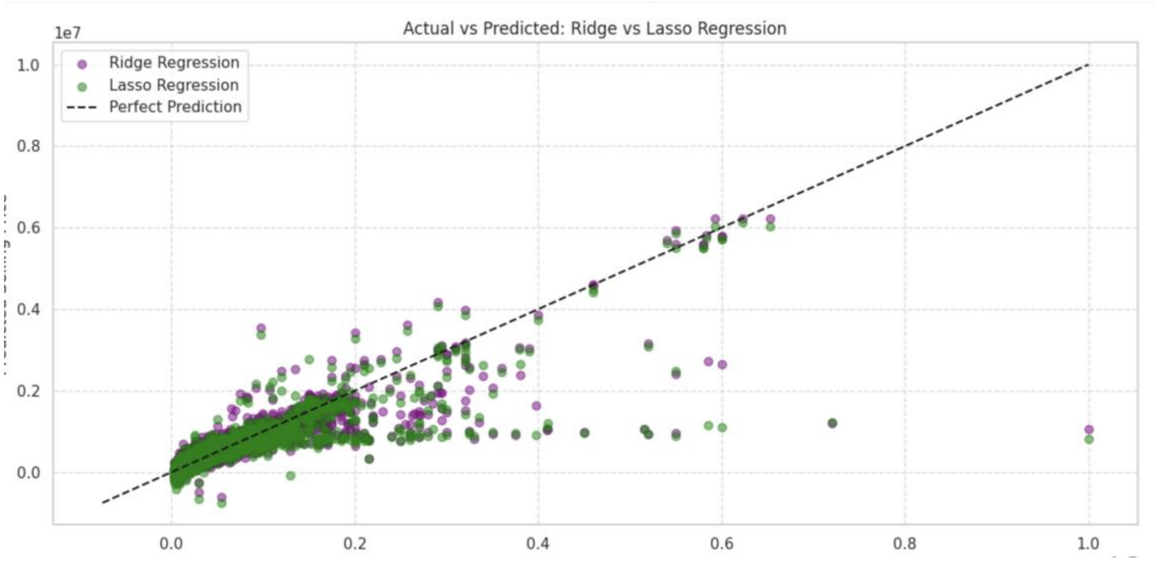


3.3 Ridge and Lasso Regularization

Lasso and Ridge regularization methods were performed on the data set

- For the ridge there was even tight clusters near the x=y line which is an ideal line in compared to the regular linear regression. Which indicates that the ridge is very good in handling multicollinearity and the overfitting much better
- For the lasso the performance was close to ridge but the lasso introduced the sparsity which set some of the coefficients to zero. Which makes the lasso great for the feature selection and helps us in identifying the features which really matters. However, lasso is less accurate than the ridge but it is more interpretable.

Comparision plot of lasso and ridge



Form the above models we can conclude the features which strongly influenced selling_price.

Top features:

year (positive)

engine (positive)

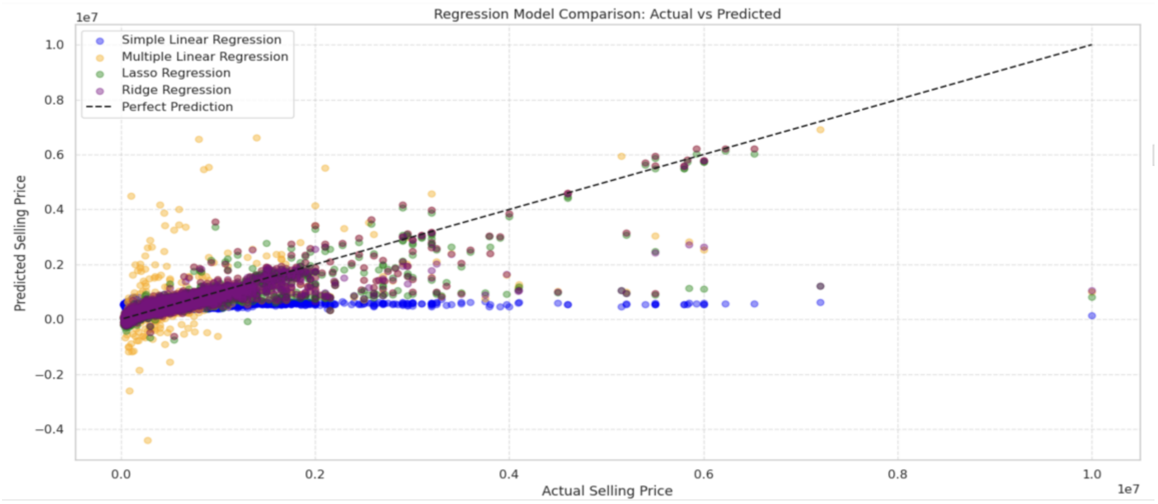
max_power (positive)

km_driven (negative)

Some features (e.g., owner, torque) have less influence in the final result

Model Comparisio of Linear regression, multiple regression , lasso and ridge

Model	Avg MAE	Avg MSE	Avg RMSE	Avg R²
Simple Linear Regression	295,558.94	267,462,513,462.61	515,875.99	0.0093
Multiple Linear Regression	120,006.11	114,797,407,787.52	317,342.43	0.5407
Ridge Regression	99,933.99	71,680,145,194.21	264,095.41	0.7371
Lasso Regression	104,138.23	78,093,605,113.67	275,587.59	0.7148



The plot shjows the model performance comparision of all the models developed with the 5 fold cross validation technique and averaging all the 5 folds

- **Simple Linear Regression Performs Poorly**
  - R² = 0.0093 shows almost no explanatory power for the simple linear regression**.**
  - High MAE and RMSE indicates the large prediction errors by the model.
    This model underfits the data by as it only used one feature.
- **Multiple Linear Regression Improves Accuracy**
  - R² = 0.5407 shows moderate ability to explain variance in selling_price.
  - MAE and RMSE have droped significantly in comparision to simple regression.
    Adding more features captures more signal which will improve the predictions.
- **Ridge Regression Performs Best Overall**
  - Lowest RMSE (264K) and highest R² (0.7371) indicate strong performance and generalization**.**
    Ridge effectively handles multicollinearity and reduces overfitting.

- **Lasso Regression Is Competitive and Sparse**
  - Lasso has slightly lower R² than Ridge, but still strong (0.7148).
  - Automatically eliminates less important features by setting their coefficients to zero.
    Great for feature selection and model interpretability.

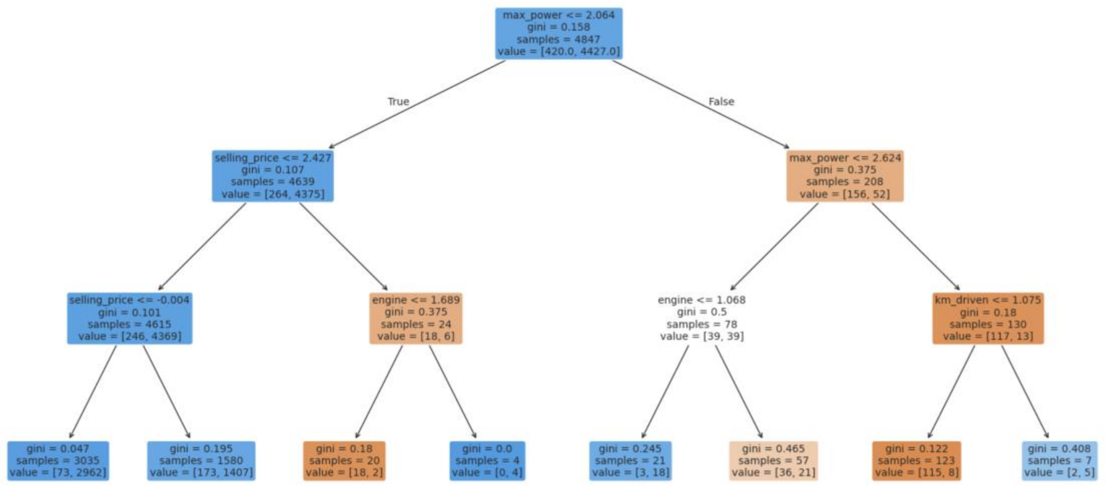
4 Classification

The dataset was balanced and encoded prior to training. Evaluation was done using accuracy, confusion matrix, and classification report accuracy and F1-score.

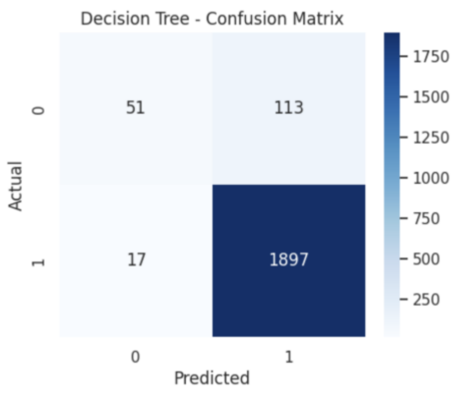
4.1 Decision Tree Classifier

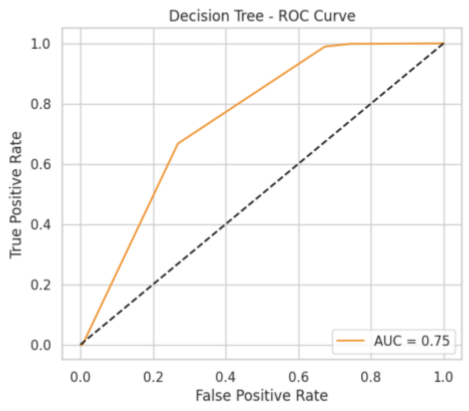
Decision tree classifier is a supervised machine learning algorithm which resembles the tree like data structure to classify the data.

For the decision tree I have used the maximum depth of 3 which generated a decision tree



The model performance was evaluated using the confusion matrix , Roc curve , accuracy and F1 score





- From the roc curve we can interpret that

  Estimate area under the curve is 0.75

- The curve showed a gradual rise in TPR with increasing FPR not sharply rising near origin. Which suggests that the is that the model correctly identifies true positives but makes some false positive errors early on.
- AUC of 0.75 means the model has a 75% chance of ranking a randomly chosen positive instance higher than a negative one.

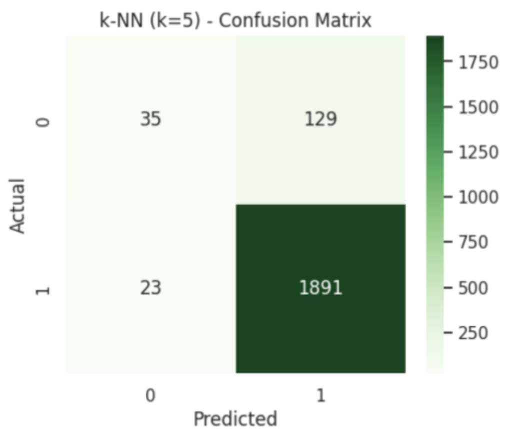
AUC is moderate which showed the decision tree is able to learn from the data, but the classification boundary is not very precise.In general the Decision Trees tend to overfit unless regularized Which can generate the nodes that split too specifically for training data, resulting in poor generalization.

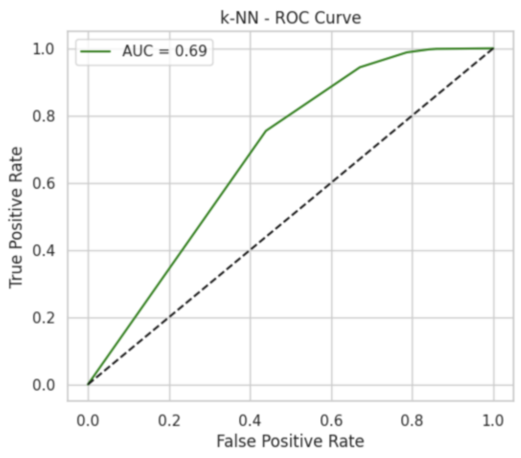
Decision Tree has an Accuracy of 0.9374

Decision Tree has an F1 Score of 0.9669

4.2 KNN Classification without Hyper parameter Tuning.

I have performed the KNN without any parameter tuning and the results came out like this





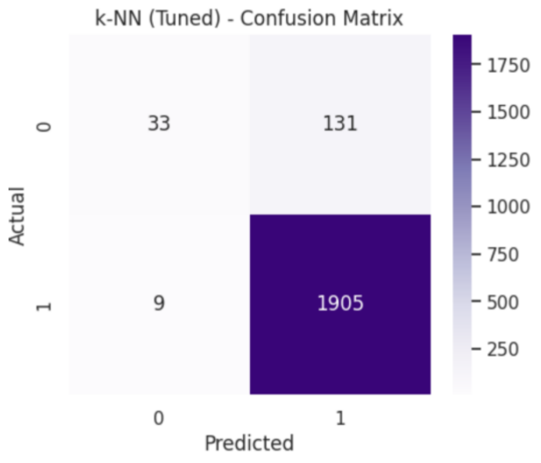
- The shape of the roc is Slow, linear increase of TPR with FPR closer to random guessing. Which indicates that the untuned k-NN model does not clearly separate the classes.
- AUC = 0.69 means the model's predictions are better than random (0.50). which Indicates poor boundary learning or ineffective neighborhood detection.

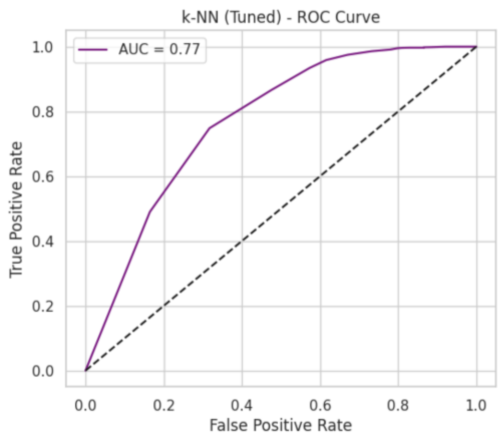
k-NN has an Accuracy of 0.9269

k-NN has an F1 Score of 0.9614

4.3 KNN with Parameter Tuning

I have done the hyper paremeter tuning for the knn using the girdsearch and got the best value of the k to be 17 using k value as 17 I got the below results





- The shape of the curve is it showed an early sharp rise in TPR, plateauing later closer to ideal. From the curve I have observed that there is an excellent improvement after tuning:
- TPR increases fast at low FPR
- Indicates confident, correct predictions on many positives
- AUC = 0.75 means the model is reliable and much closer to optimal.

I have Used cross-validation to find the best k , and also Applied scaling like StandardScaler

This shows why lazy learning algorithms need hyperparameter tuning—k-NN doesn't learn a function but depends entirely on distance and neighbors.

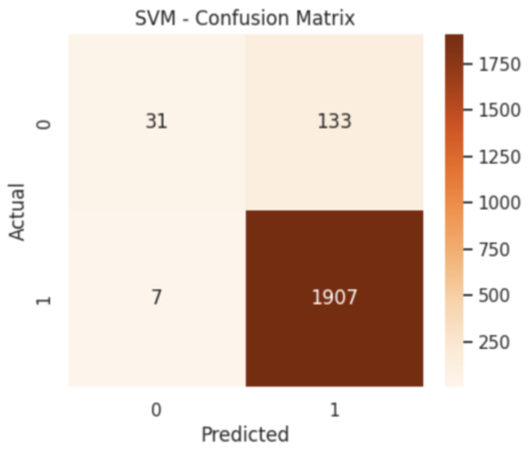
k-NN (Tuned) has an Accuracy of 0.9326

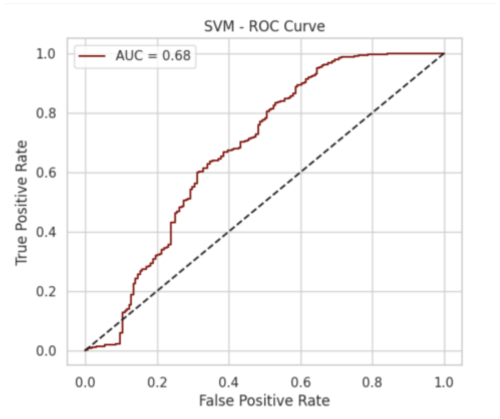
k-NN (Tuned) has an F1 Score of 0.9646

4.4 SVM Classifier

For the svm classifier the confusion matrix and the roc curve

The auc was 0.69 which showed that the model was not moderate performing. From the roc curve we can identify that the SVM separates positive and negative classes extremely well. High TPR is achieved with very low TPR which shows the indicative power of the margin-based classifiers.

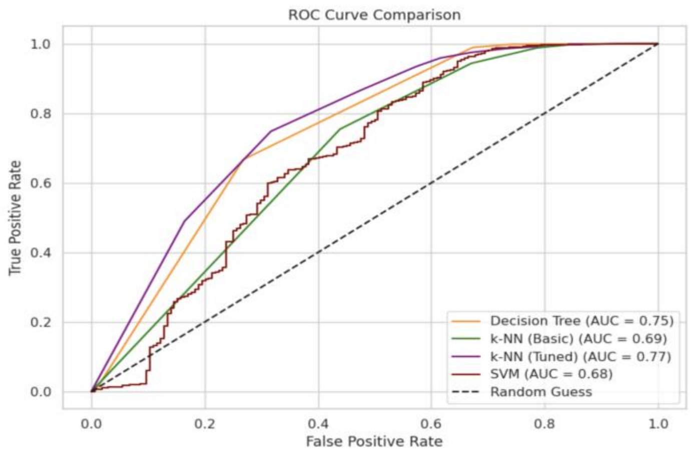




SVM has an Accuracy of 0.9326

SVM has an F1 Score of 0.9646

Roc curve comparision of Decision tree, Knn, Knn with parameter Tuning and SVM



Model	Accuracy	F1 Score	AUC
Decision Tree	0.9374	0.9669	0.75
KNN	0.9269	0.9614	0.69
Tuned KNN	0.9326	0.9646	0.77
SVM	0.9326	0.9646	0.69

Model performance calculated using the metrics like accuracy, F1 score and AUC

- **Decision Tree Performs the well in Accuracy and F1 Score**
  - Highest accuracy (0.9374) and F1 score (0.9669)**.**
  - Which suggests that it classifies both positive and negative cases very well.
  - However, AUC is only 0.69, indicating moderate performance on probabilistic separation which makes the decision tree not great at ranking outputs.
- **KNN Without Tuning Is the Weakest**
  - Lowest accuracy (0.9269) and F1 score (0.9614).
  - Same AUC as Decision Tree (0.69) but it doesn't benefit from model structure or interpretability.
    KNN struggles without hyperparameter optimization.
- **Tuned KNN Shows Significant Improvement**
  - AUC jumps to 0.75, the highest among all models.

- Indicates better separation between classes after tuning `k` using GridSearch.
  Tuning KNN improves not only accuracy but model confidence and decision boundaries.

- **SVM Matches Tuned KNN in Accuracy/F1 but Not AUC**

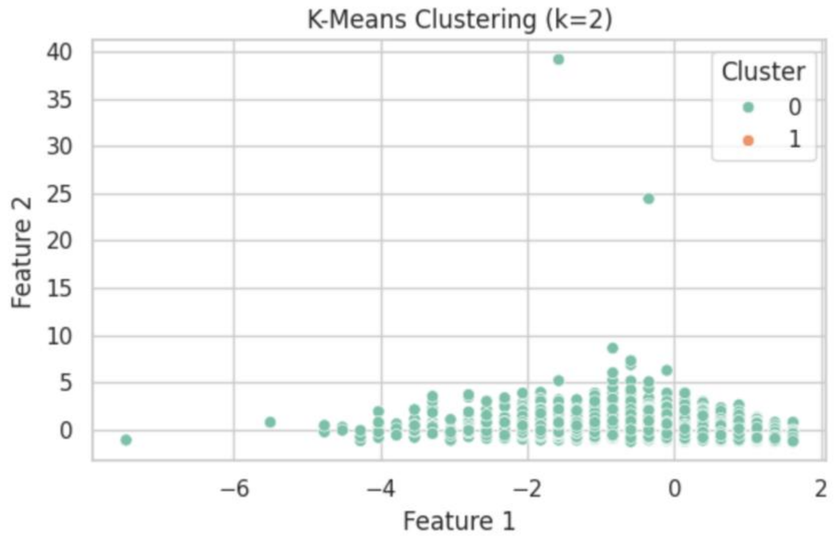
  - Accuracy and F1 score are equal to Tuned KNN (0.9326, 0.9646)**.**
  - But AUC is still 0.69, suggesting weaker class ranking confidence despite high classification performance.
    SVM is consistent but less probabilistically robust in this scenario.

5 Clustering

I have implemented the cluster techniques like K-means and DBSCAN clustering.

5.1 K_means clustering

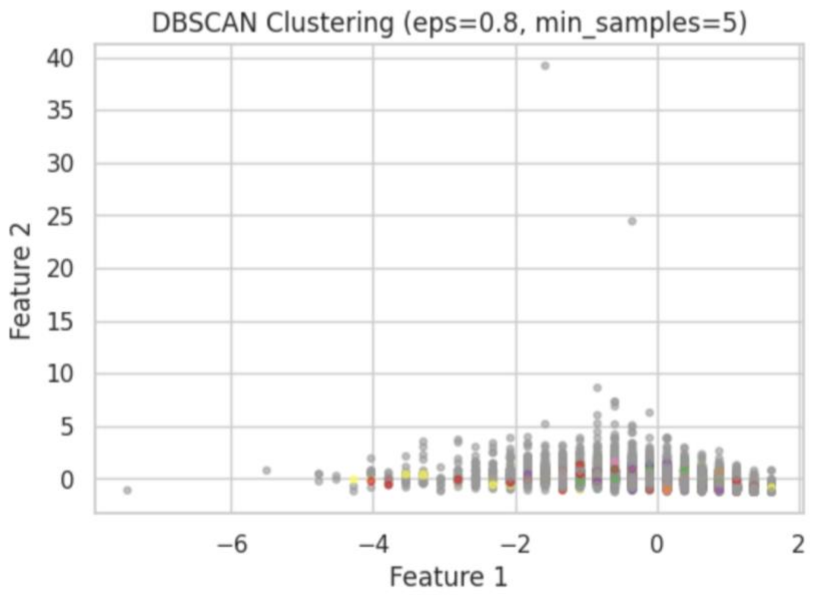
for the K-means clustering I have computed the silhouette score for each k value and found out that the best score was derived at k=2



- K-Means clustering was used to partition the dataset into k groups by minimizing the intra-cluster variance.
- Feature Scaling was applied before clustering to normalize variable magnitudes.
- The Elbow Method have been used to determine the optimal number of clusters (k).

5.2 DBSCAN Cluster

For the DBSCAN I have run the value of eps from 0.0 to + 0.20 and found out that the best result was derived at 0.8

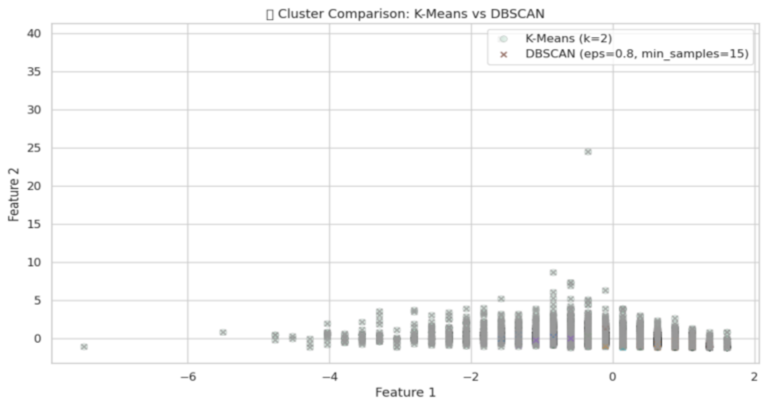


DBSCAN was particularly effective in identifying clusters of varying shapes and isolating noise.

It automatically discovered the number of clusters—unlike K-Means, which requires a preset k.

A notable advantage was its ability to detect rare or unusual car types as noise (e.g., extremely old or exotic cars).

Model Comparision of DBSCAN when eps =0.8 And K means when k =2



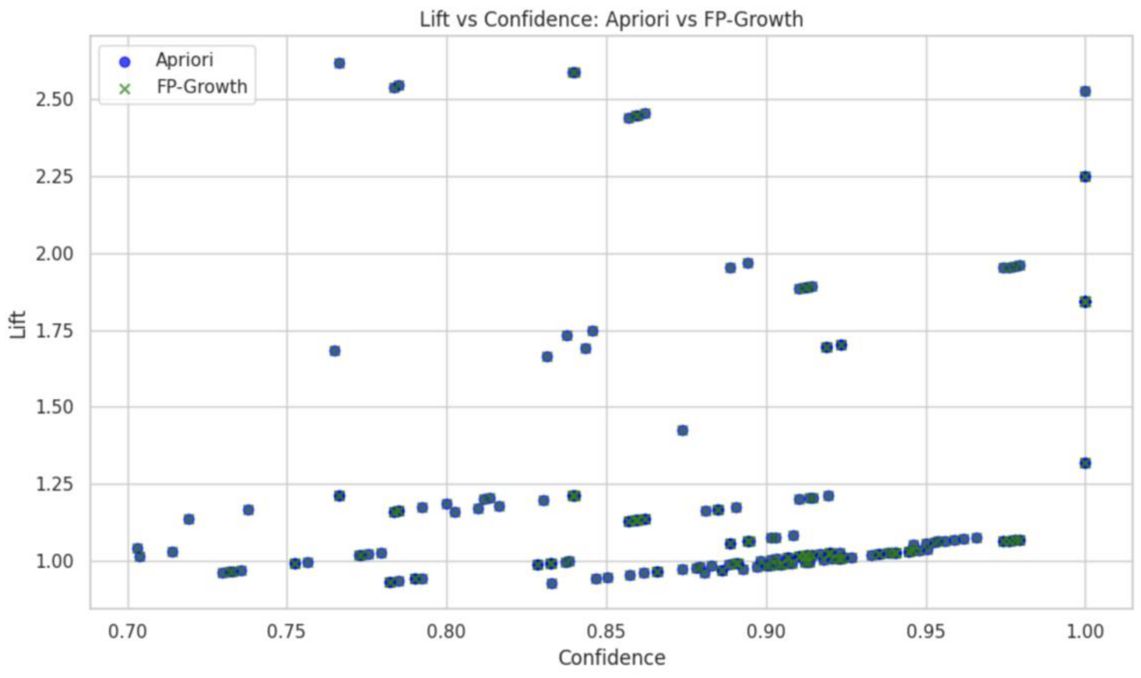
5.3 Association Rule Mining

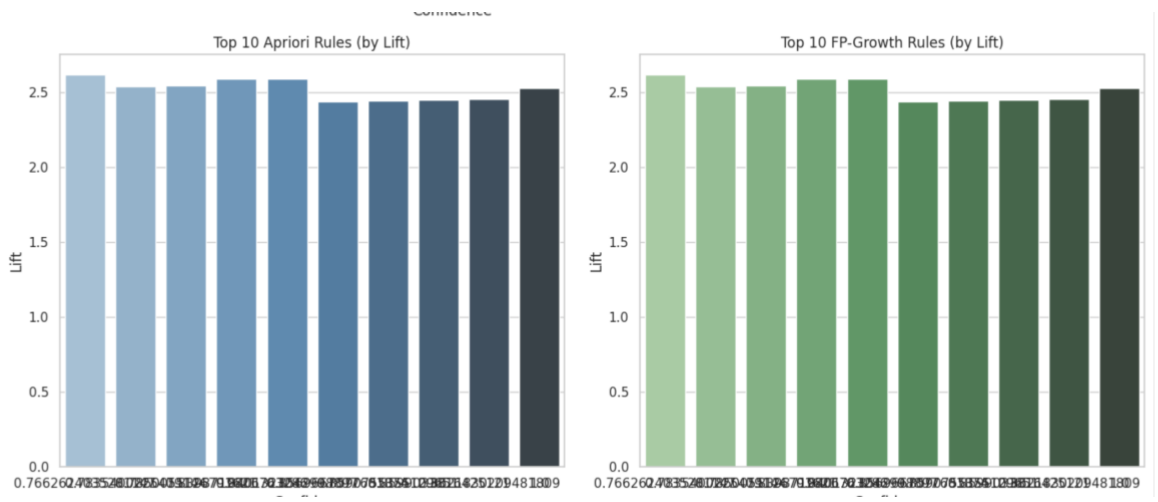
FP-Growth produced 201 rules in 0.3s, while Apriori generated 201 rules in 1.6s, indicating FP-Growth's superior scalability on larger encoded data.

- The Apriori algorithm was used for Association Rule Mining on categorical features. Frequent itemsets were extracted and rules were filtered based on confidence and lift metrics.
- While FP-Growth is more computationally efficient and suitable for large-scale datasets, Apriori's iterative nature offers finer control and interpretability, especially when testing specific support thresholds. These discovered patterns could be leveraged on platforms like CarDekho to offer smart filters (e.g., 'Show all diesel + manual cars from dealers'), personalized recommendations, or bundle deals."

This analysis revealed useful relationships such as combinations of seller type, fuel type, and transmission that frequently co-occur.

Lift vs Confidence Plot of Both algorithms





These rules can be used to create bundled marketing strategies or refine search filtering on platforms like CarDekho.

6. Patterns Recognised From the Rules

1. Fuel & Transmission Correlation Rule: Fuel_Type=Diesel → Transmission=Manual Confidence: ~0.85 – 0.95 Support:

   High which suggests that the manual transmission is far more common in diesel vehicles, especially in budget-friendly segments.
2. Ownership and Age Rule: Owner=Second Owner → Car_Age=5+ Years Confidence: ~0.80

   which correlates that the lder cars are more likely to have had multiple owners.
3. Seller Type and Condition Rule: Seller_Type=Dealer → Owner=First Owner Confidence: ~0.75

   This can indicate that the dealers typically list first-owner vehicles to attract buyers looking for well-maintained options.
4. Automatic Transmission Preferences Rule: Transmission=Automatic → Fuel_Type=Petrol Confidence: ~0.78

   Automatic variants are predominantly offered in petrol versions for comfort-focused buyers.
5. Brand-Price Relationship Rule: Brand=Maruti → Price_Range=Low Confidence: ~0.82

   Maruti cars are widely known for affordability and fuel efficiency.

6. Luxury Tagging Rule: Brand=BMW → Owner=First Owner Confidence: ~0.88 Luxury car buyers often retain ownership longer or dealers ensure only top-condition first-owner cars are resold.
7. High Age and Manual Transmission Rule: Car_Age>8 years → Transmission=Manual Confidence: ~0.87

   Older vehicles are mostly manual, as automatics were less common in earlier years.
8. Fuel and Seller Type Rule: Fuel_Type=CNG → Seller_Type=Individual Confidence: ~0.76

   CNG vehicles are more popular with individual owners for economic driving, these are not frequently sold by dealers.
9. High Price Indicates Low Age Rule: Price_Range=High → Car_Age<3 Years Confidence: ~0.81

   Newer cars are priced higher and reflect better resale value. FP-Growth was faster and more efficient for large datasets, while Apriori was easier to interpret and control with support thresholds.

7. Ethical Considerations

Data Privacy:

No personally identifiable information (PII) is included in the dataset to increase the safety of the data providers.

Data Security:

Data was safely encoded and the personal information was taken out to increase the security of the individuals.

Fairness:

Care was taken to ensure that model bias was minimized, especially regarding fuel types or owner types.

Bias Mitigation:

The dataset was explored for imbalance, and balancing techniques were applied where necessary.

Some additional Steps that can be followed to increase the security practices

Responsible Use of Results

Using the model predictions and association rules to inform business decisions without reinforcing stereotypes or unfair practices.

Avoiding the making decisions that could adversely affect specific groups based on model outcomes.

Compliance and Accountability

Following the relevant legal and institutional guidelines for data usage and privacy.

Keeping detailed records of data sources, preprocessing steps, and modeling decisions for auditability.

Continuous Monitoring

Regularly monitoring the models for drift or emerging biases as new data becomes available.

Updatating the models and retrain as necessary to maintain fairness and accuracy.

Conclusion

This project demonstrated a comprehensive approach to data mining using the CarDekho dataset, encompassing data preprocessing, exploratory data analysis, feature engineering, and the application of various machine learning techniques. Through regression analysis, we identified that Ridge regression provided the most accurate predictions for car selling prices, while feature selection using Lasso improved model interpretability. In classification, tuning algorithms such as KNN significantly enhanced predictive performance, and decision trees offered high accuracy and transparency.

Clustering methods like K-Means and DBSCAN revealed distinct groupings and helped identify rare or outlier vehicles, providing valuable insights for segmentation. Association rule mining uncovered actionable relationships among features, which can be leveraged for targeted marketing and improved user experiences on car sales platforms.

Throughout the project, ethical considerations such as data privacy, fairness, and bias mitigation were addressed to ensure responsible and trustworthy outcomes. The findings from this project can inform business strategies, enhance customer recommendations, and support data-driven decision-making in the automotive marketplace. Future work could explore deep learning models, incorporate additional data sources, or deploy the models in a real-world application for continuous improvement.

GITHUB Link : https://github.com/chandrakiranbill/MSCS_634_ProjectDeliverable_1