BUSINESS, WEB AND SOCIAL MEDIA ANALYSIS

CRN - 48449

Case Study - 2

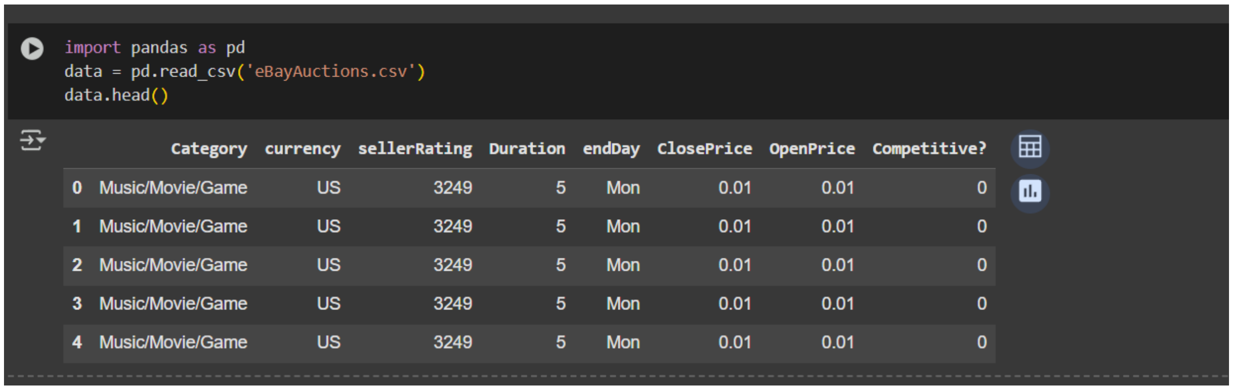
NAME: ADITYA SHROFF & Shreya Sahal

COLLEGE ID: 101539669

DATE: 01/04/2025

PART □ 1

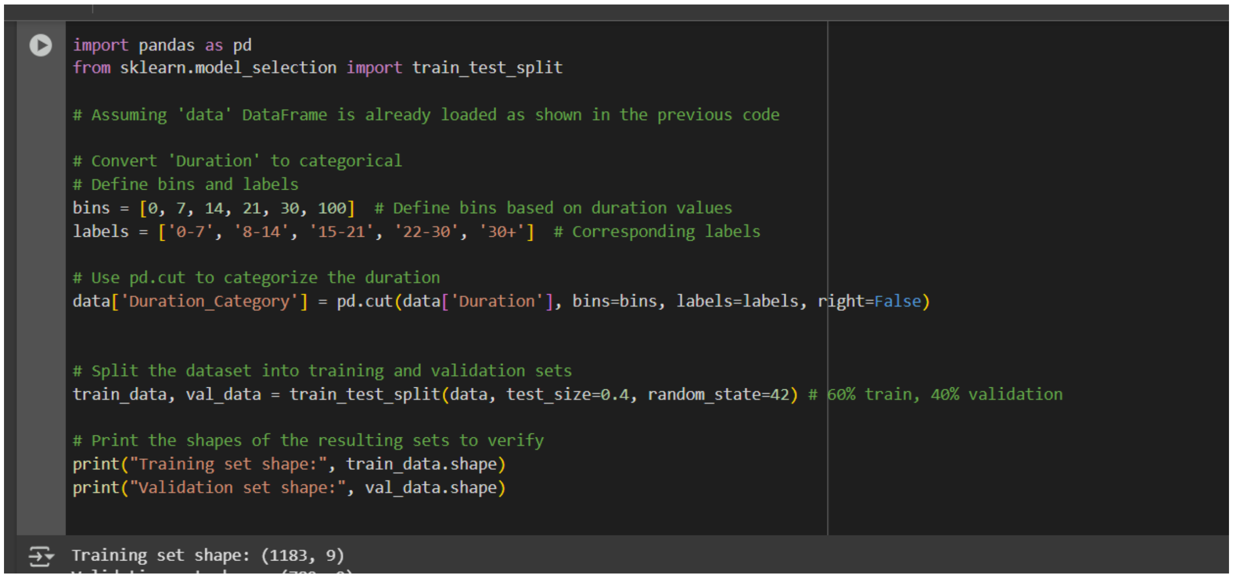
DATA PROCESSING



```python
import pandas as pd
data = pd.read_csv('eBayAuctions.csv')
data.head()
```

	Category	currency	sellerRating	Duration	endDay	ClosePrice	OpenPrice	Competitive?
0	Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
1	Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
2	Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
3	Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
4	Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0

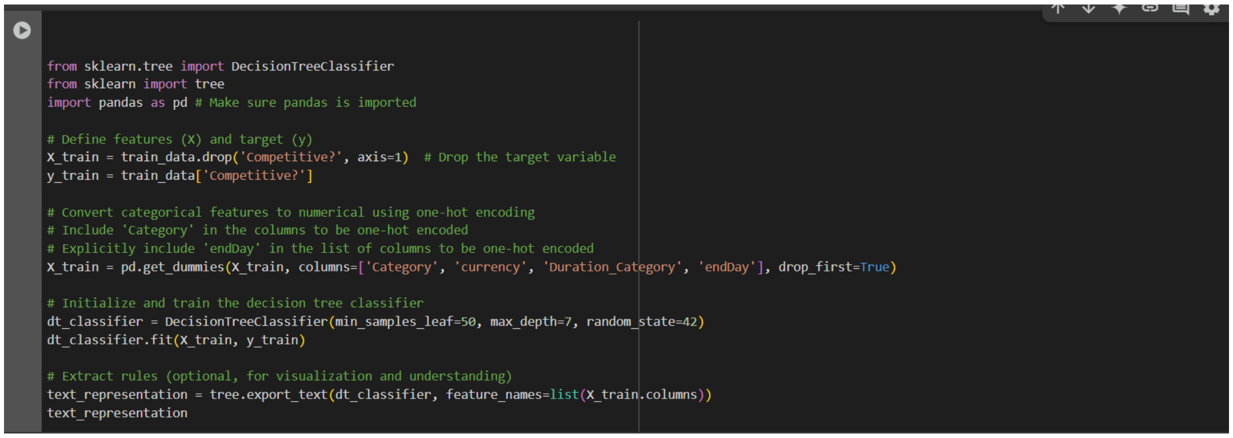
CONVERSION OF DURATION INTO CATEGORICAL VARIABLE



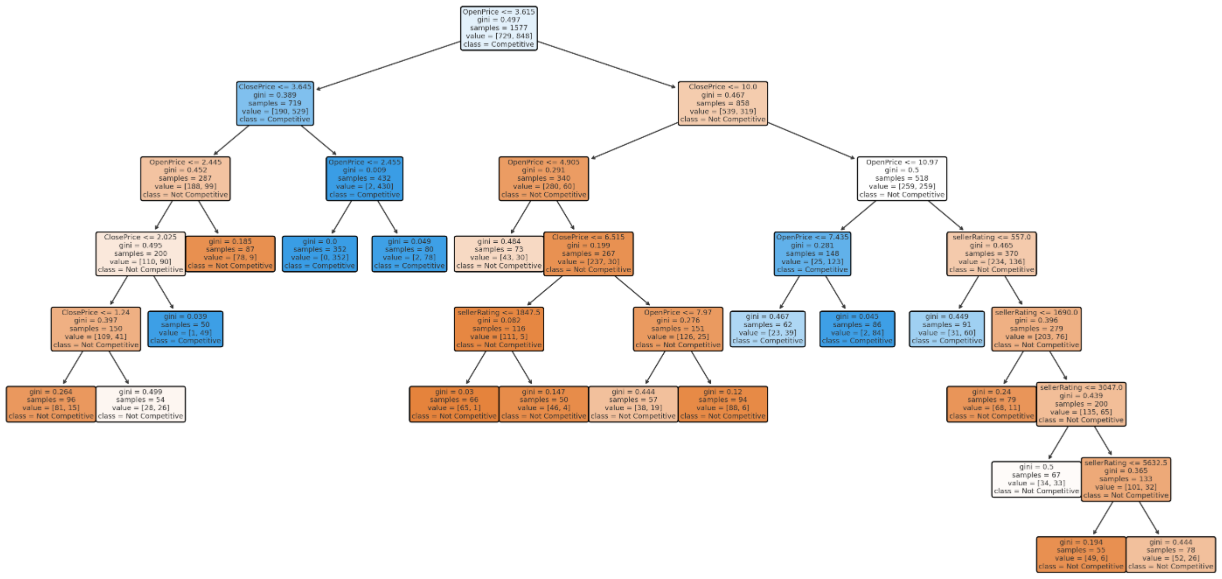
```python
import pandas as pd
from sklearn.model_selection import train_test_split

# Assuming 'data' DataFrame is already loaded as shown in the previous code

# Convert 'Duration' to categorical
# Define bins and labels
bins = [0, 7, 14, 21, 30, 100]  # Define bins based on duration values
labels = ['0-7', '8-14', '15-21', '22-30', '30+']  # Corresponding labels

# Use pd.cut to categorize the duration
data['Duration_Category'] = pd.cut(data['Duration'], bins=bins, labels=labels, right=False)


# Split the dataset into training and validation sets
train_data, val_data = train_test_split(data, test_size=0.4, random_state=42) # 60% train, 40% validation

# Print the shapes of the resulting sets to verify
print("Training set shape:", train_data.shape)
print("Validation set shape:", val_data.shape)
```

```
Training set shape: (1183, 9)
```

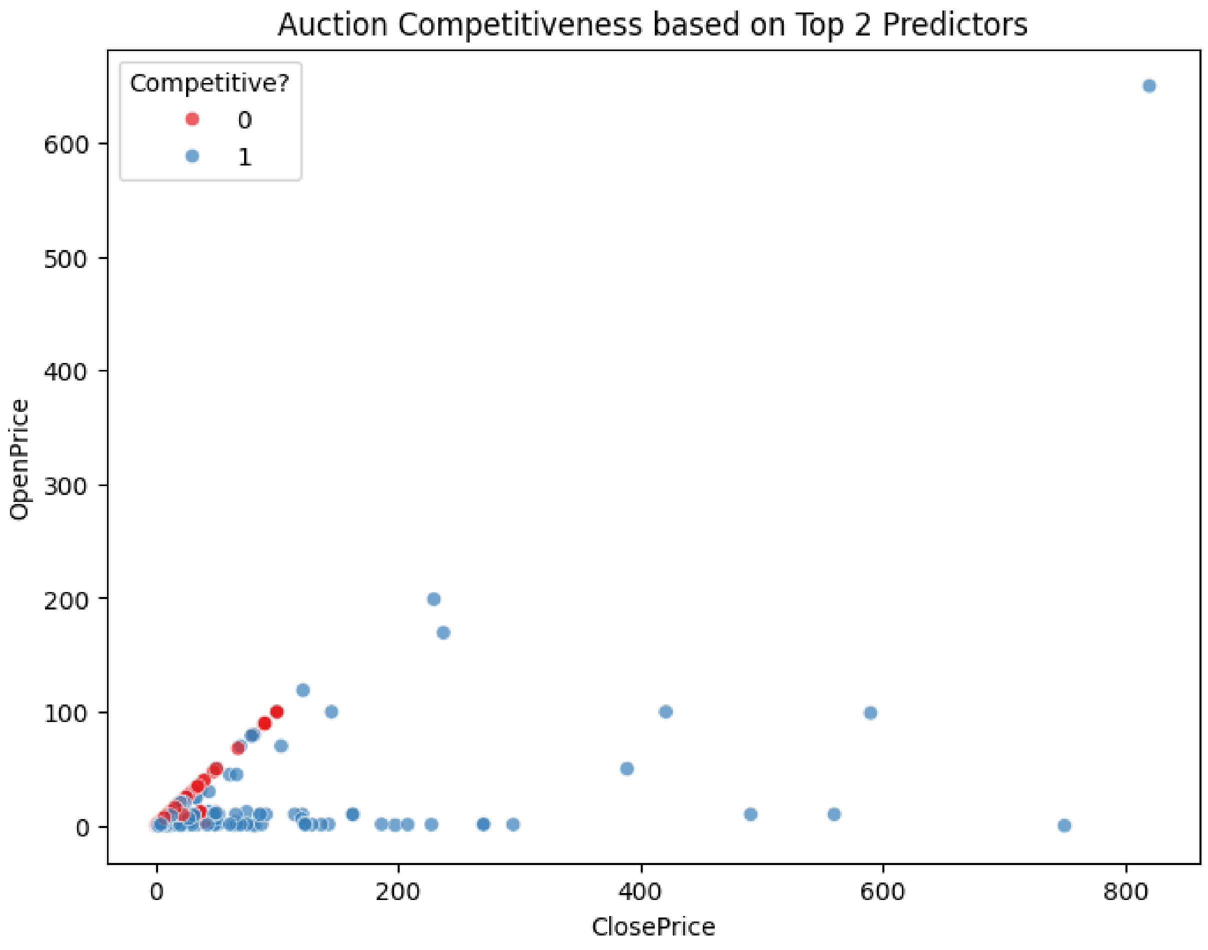
DECISION TREE:

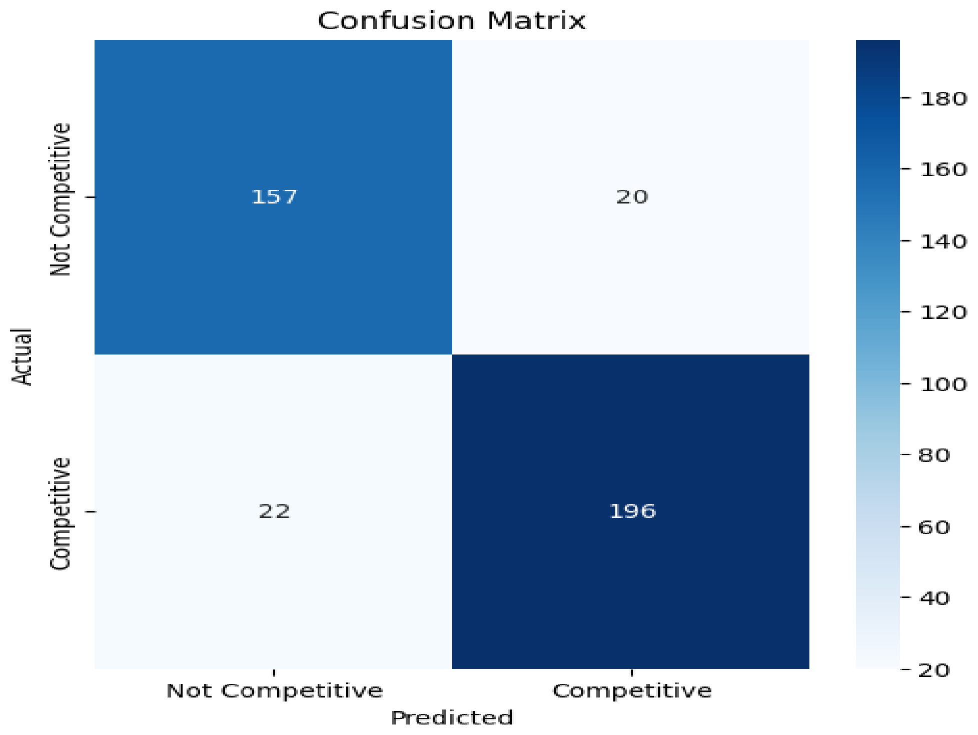


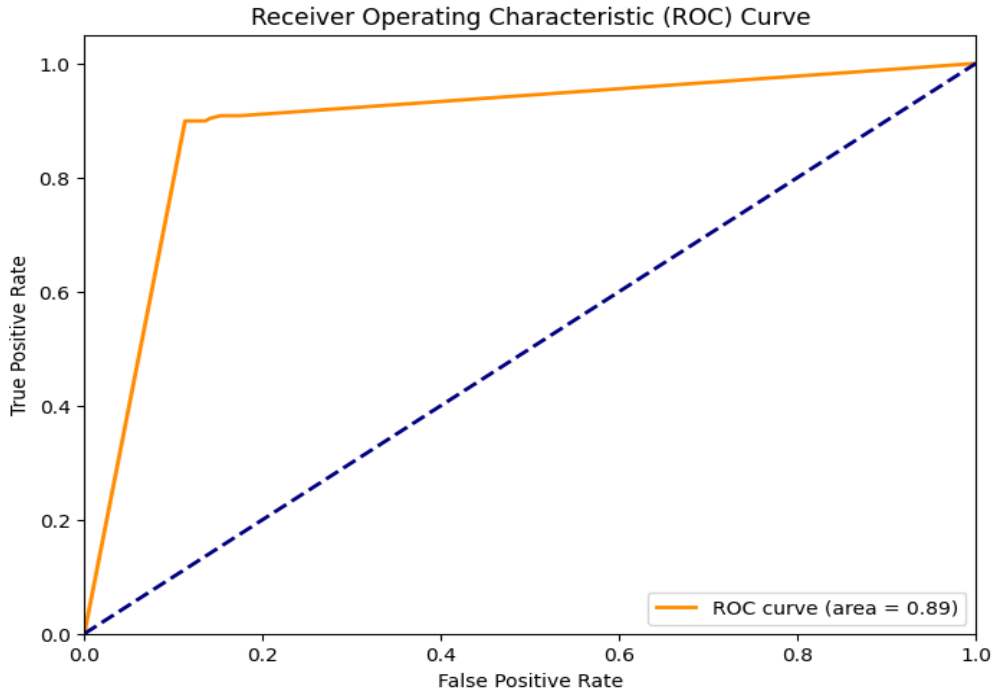
```python
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
import pandas as pd # Make sure pandas is imported

# Define features (X) and target (y)
X_train = train_data.drop('Competitive?', axis=1)  # Drop the target variable
y_train = train_data['Competitive?']

# Convert categorical features to numerical using one-hot encoding
# Include 'Category' in the columns to be one-hot encoded
# Explicitly include 'endDay' in the list of columns to be one-hot encoded
X_train = pd.get_dummies(X_train, columns=['Category', 'currency', 'Duration_Category', 'endDay'], drop_first=True)

# Initialize and train the decision tree classifier
dt_classifier = DecisionTreeClassifier(min_samples_leaf=50, max_depth=7, random_state=42)
dt_classifier.fit(X_train, y_train)

# Extract rules (optional, for visualization and understanding)
text_representation = tree.export_text(dt_classifier, feature_names=list(X_train.columns))
text_representation
```


OpenPrice <= 3.615 gini = 0.497 samples = 1577 value = [729, 848] class = Competitive
ClosePrice <= 3.645 gini = 0.389 samples = 719 value = [190, 529] class = Competitive
ClosePrice <= 10.0 gini = 0.467 samples = 858 value = [539, 319] class = Not Competitive
OpenPrice <= 2.445 gini = 0.452 samples = 287 value = [188, 99] class = Not Competitive
OpenPrice <= 2.455 gini = 0.009 samples = 432 value = [2, 430] class = Competitive
OpenPrice <= 4.905 gini = 0.291 samples = 340 value = [280, 60] class = Not Competitive
OpenPrice <= 10.97 gini = 0.5 samples = 518 value = [259, 259] class = Not Competitive
ClosePrice <= 2.025 gini = 0.495 samples = 200 value = [110, 90] class = Not Competitive
gini = 0.185 samples = 87 value = [78, 9] class = Not Competitive
gini = 0.0 samples = 352 value = [0, 352] class = Competitive
gini = 0.049 samples = 80 value = [2, 78] class = Competitive
gini = 0.484 samples = 73 value = [43, 30] class = Not Competitive
ClosePrice <= 6.515 gini = 0.199 samples = 267 value = [237, 30] class = Not Competitive
OpenPrice <= 7.435 gini = 0.281 samples = 148 value = [25, 123] class = Competitive
sellerRating <= 557.0 gini = 0.465 samples = 370 value = [234, 136] class = Not Competitive
ClosePrice <= 1.24 gini = 0.397 samples = 150 value = [109, 41] class = Not Competitive
gini = 0.039 samples = 50 value = [1, 49] class = Competitive
sellerRating <= 1847.5 gini = 0.082 samples = 116 value = [111, 5] class = Not Competitive
OpenPrice <= 7.97 gini = 0.276 samples = 151 value = [126, 25] class = Not Competitive
gini = 0.467 samples = 62 value = [23, 39] class = Competitive
gini = 0.045 samples = 86 value = [2, 84] class = Competitive
gini = 0.449 samples = 91 value = [31, 60] class = Competitive
sellerRating <= 1690.0 gini = 0.396 samples = 279 value = [203, 76] class = Not Competitive
gini = 0.264 samples = 96 value = [81, 15] class = Not Competitive
gini = 0.499 samples = 54 value = [28, 26] class = Not Competitive
gini = 0.03 samples = 66 value = [65, 1] class = Not Competitive
gini = 0.147 samples = 50 value = [46, 4] class = Not Competitive
gini = 0.444 samples = 57 value = [38, 19] class = Not Competitive
gini = 0.12 samples = 94 value = [88, 6] class = Not Competitive
gini = 0.24 samples = 79 value = [68, 11] class = Not Competitive
sellerRating <= 3047.0 gini = 0.439 samples = 200 value = [135, 65] class = Not Competitive
gini = 0.5 samples = 67 value = [34, 33] class = Not Competitive
sellerRating <= 5632.5 gini = 0.365 samples = 133 value = [101, 32] class = Not Competitive
gini = 0.194 samples = 55 value = [49, 6] class = Not Competitive
gini = 0.444 samples = 78 value = [52, 26] class = Not Competitive

MODEL EVALUATION AND VISUALIZATION :




Confusion Matrix
Actual
Not Competitive
Competitive
157
20
22
196
Predicted
180
160
140
120
100
80
60
40
20


Receiver Operating Characteristic (ROC) Curve
True Positive Rate
False Positive Rate
ROC curve (area = 0.89)
0.0
0.2
0.4
0.6
0.8
1.0

eBay Auction Competitiveness Analysis: A Data-Driven Approach

Introduction

This report presents the findings of an analysis conducted to identify key factors influencing the competitiveness of eBay auctions. We employed a decision tree model to uncover patterns and relationships within historical auction data. The insights gained from this analysis are used to provide actionable recommendations for optimizing auction settings to maximize bidder engagement and selling prices.

Methodology

1. **Data Preparation:** The dataset, "eBayAuctions.csv," was pre-processed to handle categorical variables and split into training and validation sets (60% and 40%, respectively).
2. **Decision Tree Model:** We trained a Decision Tree Classifier using the sklearn library in Python. Hyperparameters were set to min_samples_leaf=50 and max_depth=7 to avoid overfitting. The model was fitted using all relevant predictors, including opening price, closing price, seller rating, and item category.
3. **Feature Importance:** We evaluated the importance of each feature in the decision tree model to understand which factors most strongly influence auction competitiveness.
4. **Decision Rules Extraction:** The decision rules of the trained tree were extracted to identify specific patterns and thresholds associated with competitive and non-competitive auctions.
5. **Recommendation Formulation:** Based on the insights gleaned from the model, we developed actionable recommendations for setting auction parameters to maximize competitiveness.

Key Findings

- **Price is a Primary Driver:** The opening and closing prices of an auction are strongly correlated with its competitiveness. Very low starting prices often lead to non-competitive auctions, while moderate to higher prices tend to attract more bidders, especially when accompanied by strong seller reputations.
- **Seller Rating Matters:** For higher-value items, the seller's rating becomes a significant factor in influencing competitiveness. Buyers are more likely to engage in bidding wars for items sold by reputable sellers.
- **Category Considerations:** While less prominent than price and seller rating, the item's category can subtly impact competitiveness. Specific categories, such as Automotive, exhibit unique price sensitivities that should be considered when setting opening bids.

Recommendations

Based on the decision tree analysis, we recommend the following auction settings to maximize competitiveness:

Starting Price (OpenPrice):

- **Moderate-Value Items:** Set your starting price between $3.62 and $3.62 and $10.49 to encourage bidding.
- **High-Value Items:** If you have a strong seller rating and are offering a premium item, consider a starting price above $10.49. * **Avoid: ** Starting prices below $$10.49. * **Avoid: ** Starting prices below $3.62, as they often discourage bidders.

Duration:

- Opt for a 7- or 10-day auction duration for optimal bidder engagement.

Ending Day:

- **Target:** Weekdays or evenings when potential buyers are most active.
- **Avoid:** Ending auctions on major holidays or weekends when engagement is typically

lower.

Beyond the Basics:

In addition to these core settings, remember to:

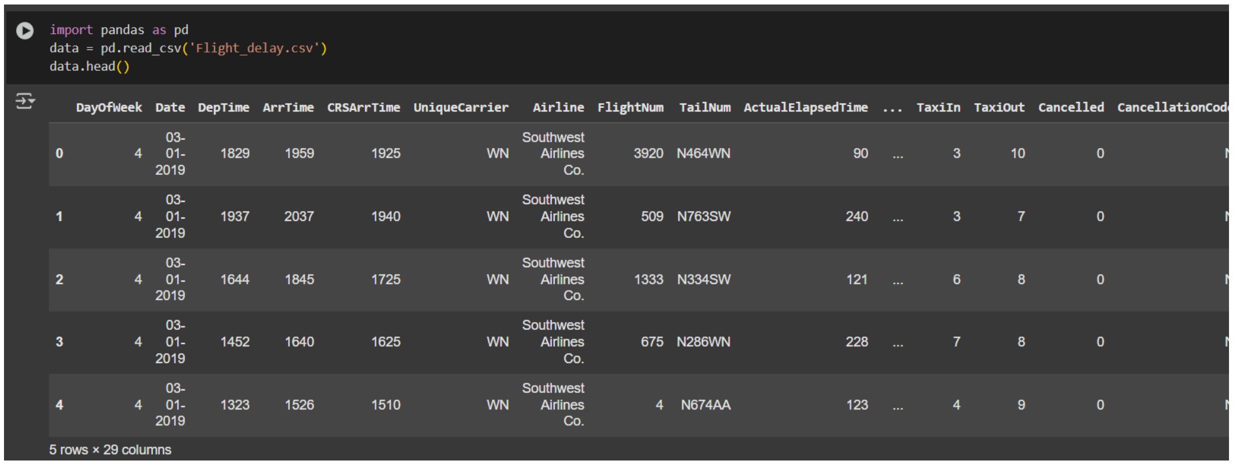
- **Build a Stellar Reputation:** Maintain a high seller rating to inspire buyer confidence.
- **Craft Compelling Listings:** Use clear descriptions and high-quality images to attract bidders.
- **Stay Informed:** Monitor market trends and competitor strategies to adapt your approach.

Conclusion

By strategically applying these insights and recommendations, eBay sellers can significantly enhance the competitiveness of their auctions, attract more bidders, and achieve higher selling prices. This data-driven approach empowers sellers to make informed decisions about their listing strategies and optimize their chances of success in the online marketplace.

PART □ 2

DATA PROCESSING AND TRAINING

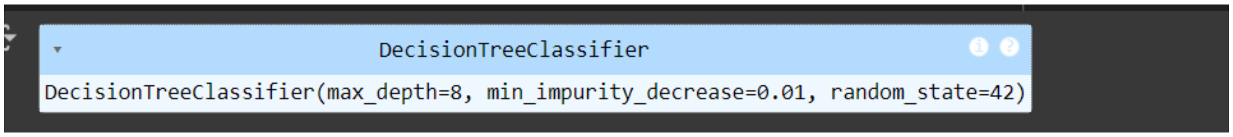


```python
import pandas as pd
data = pd.read_csv('Flight_delay.csv')
data.head()
```

	DayOfWeek	Date	DepTime	ArrTime	CRSArrTime	UniqueCarrier	Airline	FlightNum	TailNum	ActualElapsedTime	...	TaxiIn	TaxiOut	Cancelled	CancellationCod
0	4	03-01-2019	1829	1959	1925	WN	Southwest Airlines Co.	3920	N464WN	90	...	3	10	0	N
1	4	03-01-2019	1937	2037	1940	WN	Southwest Airlines Co.	509	N763SW	240	...	3	7	0	N
2	4	03-01-2019	1644	1845	1725	WN	Southwest Airlines Co.	1333	N334SW	121	...	6	8	0	N
3	4	03-01-2019	1452	1640	1625	WN	Southwest Airlines Co.	675	N286WN	228	...	7	8	0	N
4	4	03-01-2019	1323	1526	1510	WN	Southwest Airlines Co.	4	N674AA	123	...	4	9	0	N

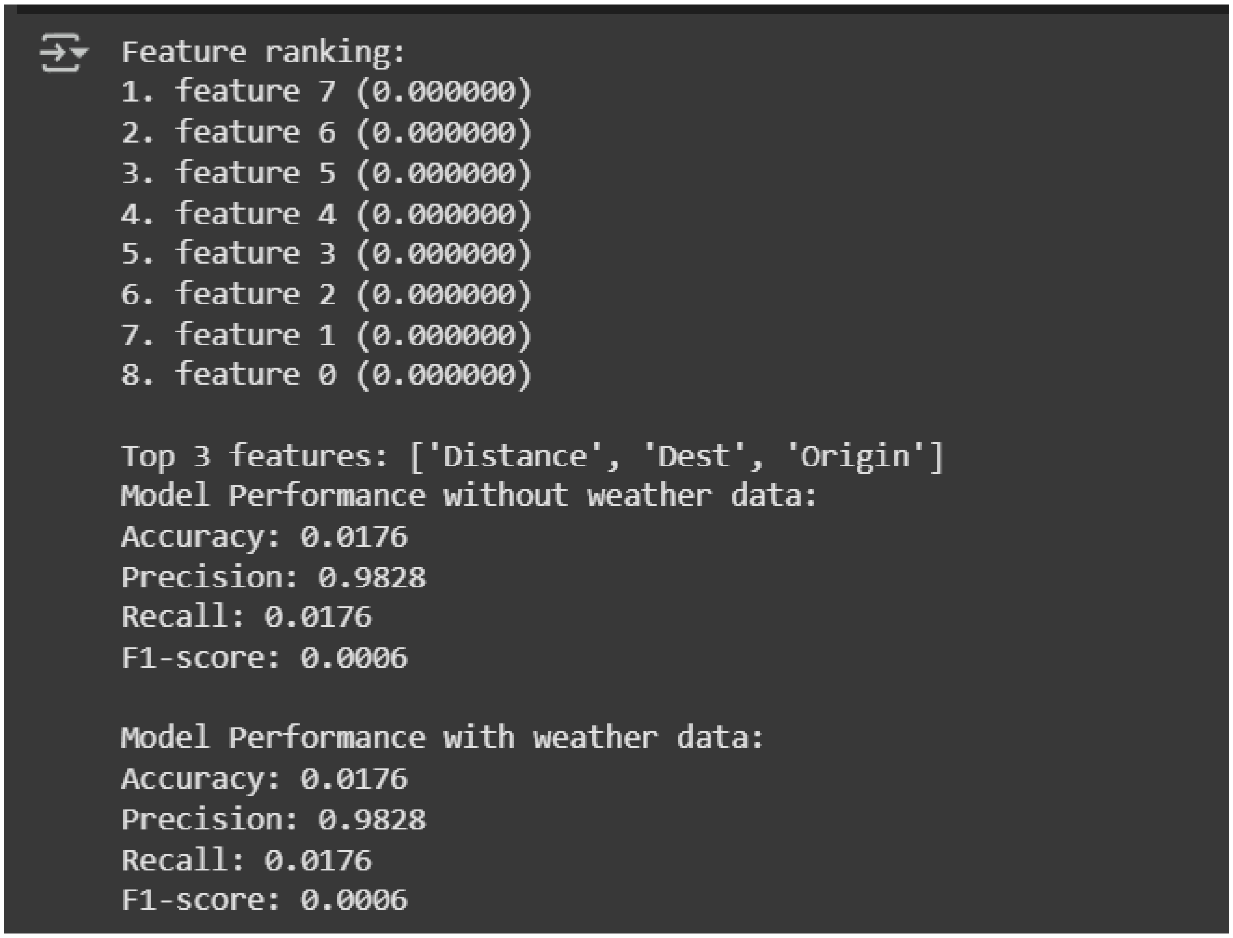
5 rows × 29 columns

Fit a classification tree with maximum depth = 8 and minimum impurity decrease = 0.01.



```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=8, min_impurity_decrease=0.01, random_state=42)
```

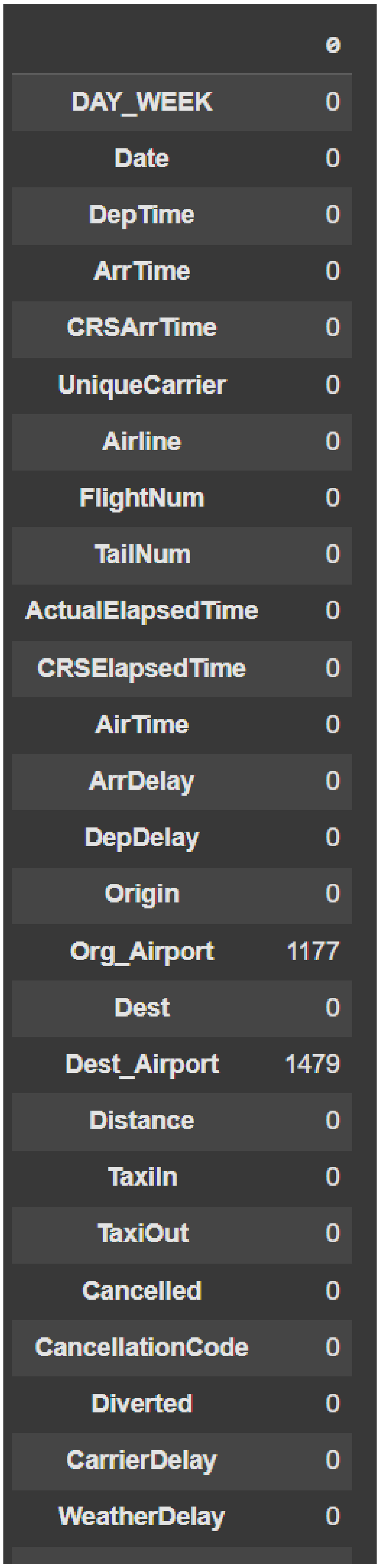
IDENTIFYING TOP 3 FEATURES THAT CAUSES FLIGHT DELAYS



```
Feature ranking:
1. feature 7 (0.000000)
2. feature 6 (0.000000)
3. feature 5 (0.000000)
4. feature 4 (0.000000)
5. feature 3 (0.000000)
6. feature 2 (0.000000)
7. feature 1 (0.000000)
8. feature 0 (0.000000)

Top 3 features: ['Distance', 'Dest', 'Origin']
Model Performance without weather data:
Accuracy: 0.0176
Precision: 0.9828
Recall: 0.0176
F1-score: 0.0006

Model Performance with weather data:
Accuracy: 0.0176
Precision: 0.9828
Recall: 0.0176
F1-score: 0.0006
```

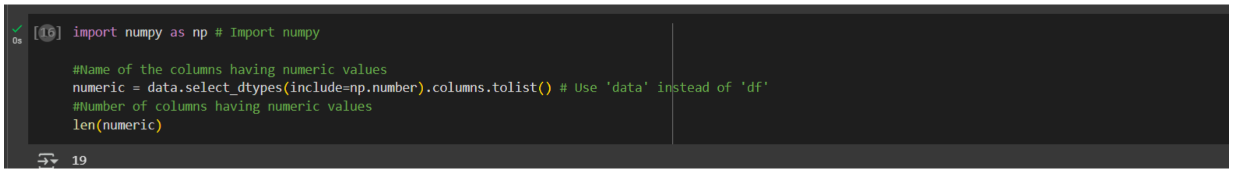
EXPLORATORY DATA ANALYSIS

1) Finding out missing values in the column



	0
DAY_WEEK	0
Date	0
DepTime	0
ArrTime	0
CRSArrTime	0
UniqueCarrier	0
Airline	0
FlightNum	0
TailNum	0
ActualElapsedTime	0
CRSElapsedTime	0
AirTime	0
ArrDelay	0
DepDelay	0
Origin	0
Org_Airport	1177
Dest	0
Dest_Airport	1479
Distance	0
TaxiIn	0
TaxiOut	0
Cancelled	0
CancellationCode	0
Diverted	0
CarrierDelay	0
WeatherDelay	0

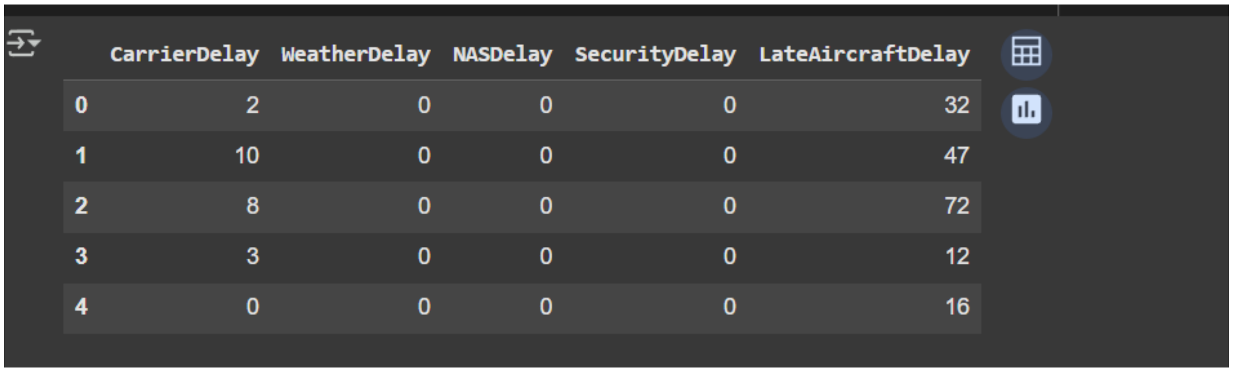
Out of 29, there are 19 Numerical variables rest of them are categorical



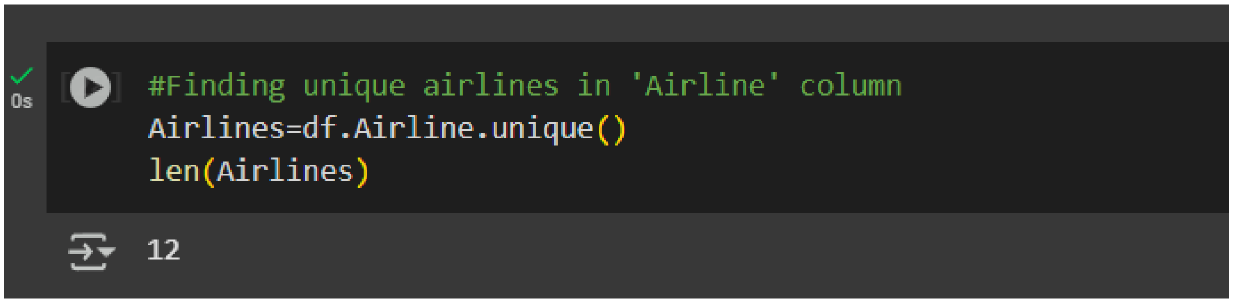
```
import numpy as np # Import numpy

#Name of the columns having numeric values
numeric = data.select_dtypes(include=np.number).columns.tolist() # Use 'data' instead of 'df'
#Number of columns having numeric values
len(numeric)
```

```
19
```

Separating Delay into another data frame

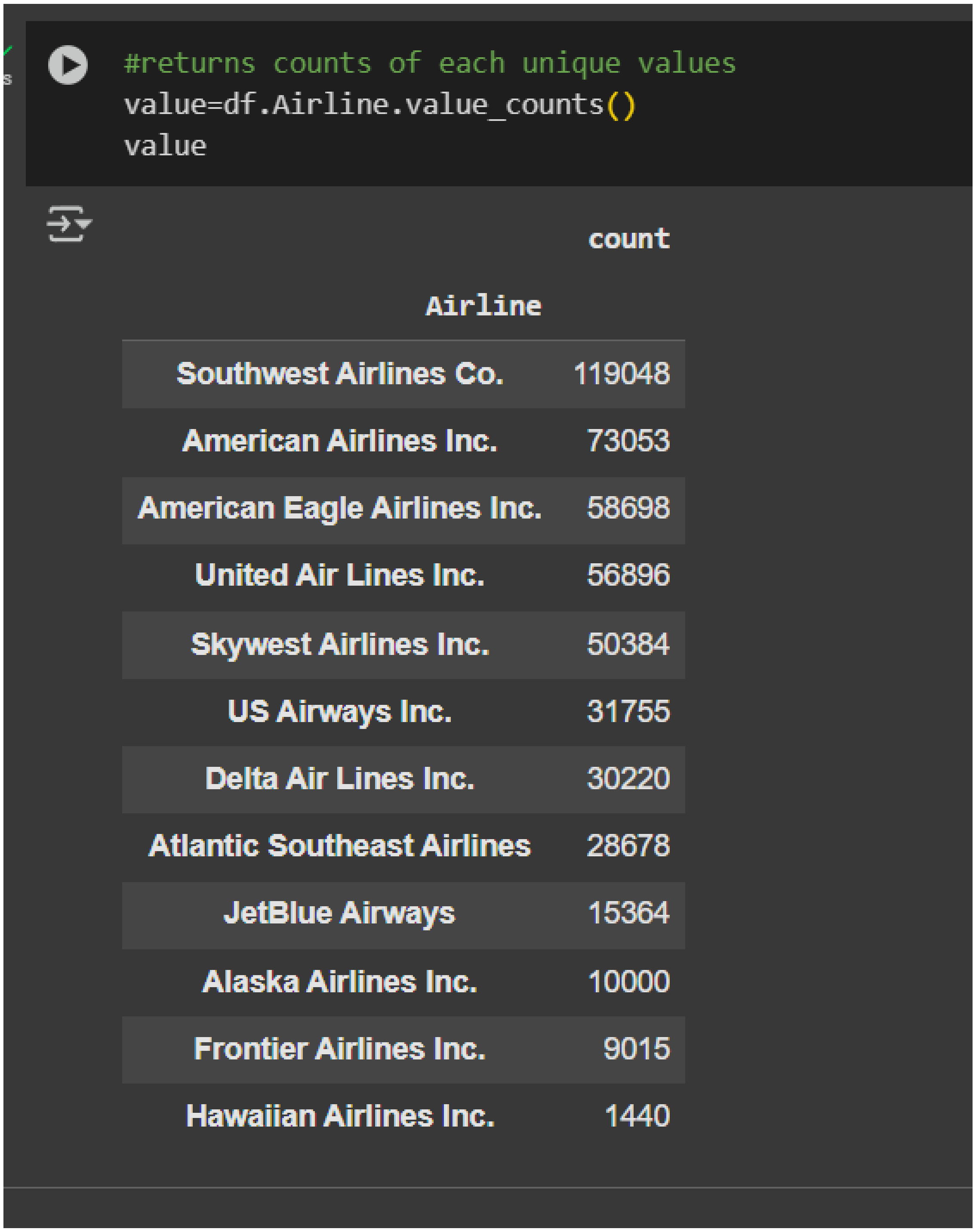


	CarrierDelay	WeatherDelay	NASDelay	SecurityDelay	LateAircraftDelay
0	2	0	0	0	32
1	10	0	0	0	47
2	8	0	0	0	72
3	3	0	0	0	12
4	0	0	0	0	16



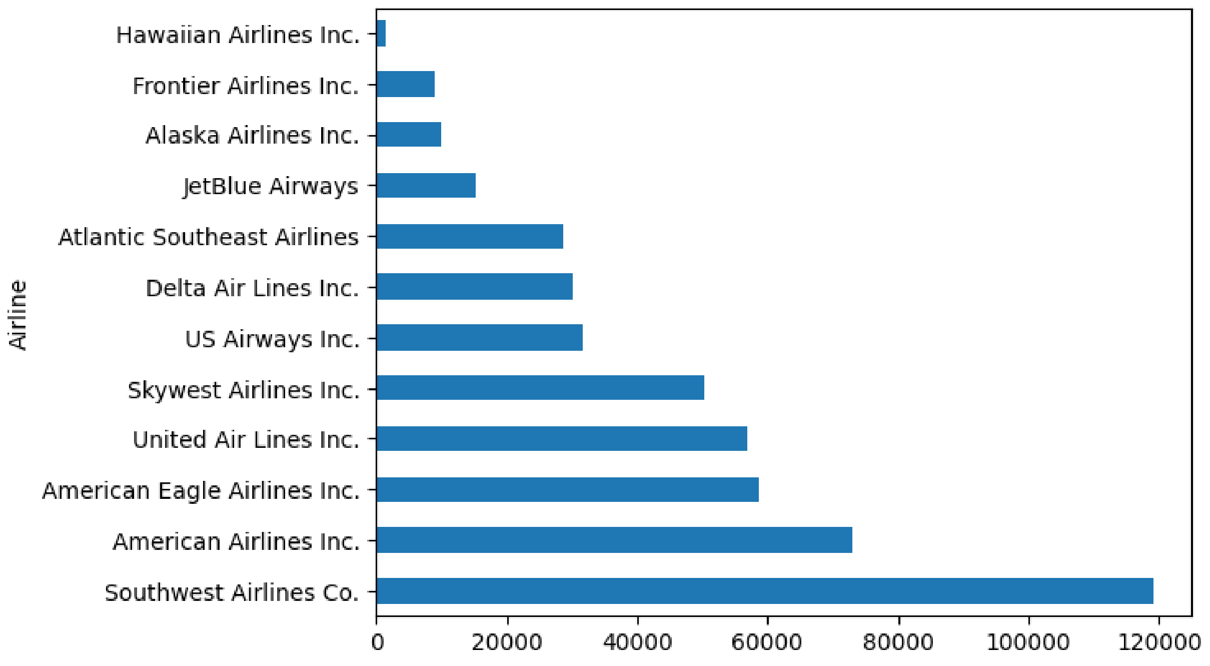
```
#Finding unique airlines in 'Airline' column
Airlines=df.Airline.unique()
len(Airlines)
```

```
12
```

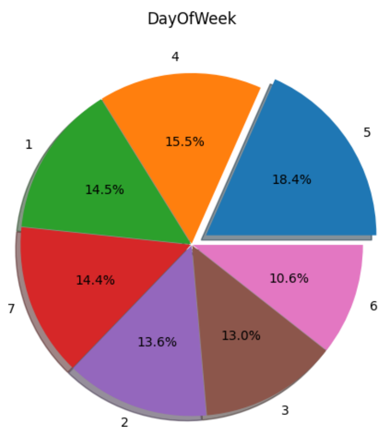


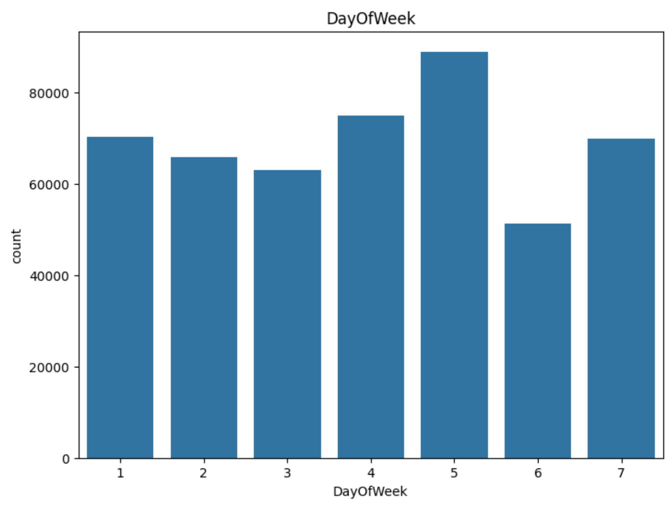
```
#returns counts of each unique values
value=df.Airline.value_counts()
value
```

```
                                    count
                        Airline
     Southwest Airlines Co.        119048
     American Airlines Inc.         73053
American Eagle Airlines Inc.        58698
     United Air Lines Inc.          56896
     Skywest Airlines Inc.          50384
          US Airways Inc.           31755
       Delta Air Lines Inc.         30220
  Atlantic Southeast Airlines       28678
         JetBlue Airways            15364
       Alaska Airlines Inc.         10000
     Frontier Airlines Inc.          9015
      Hawaiian Airlines Inc.         1440
```

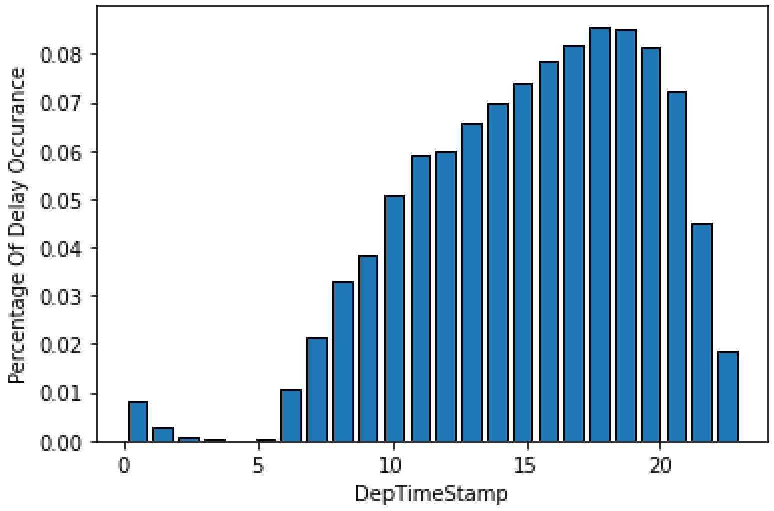


Delays on various days of the week

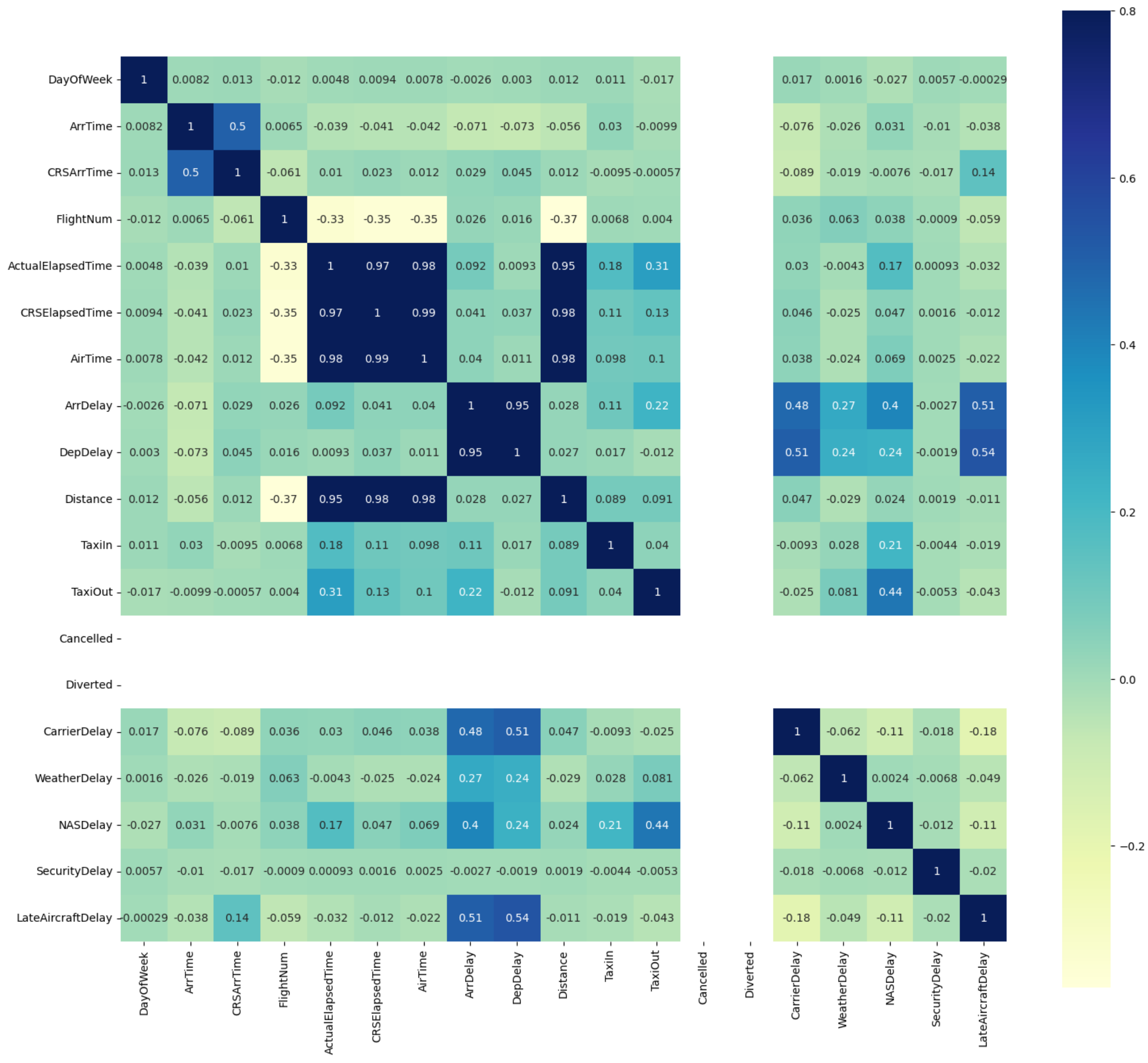


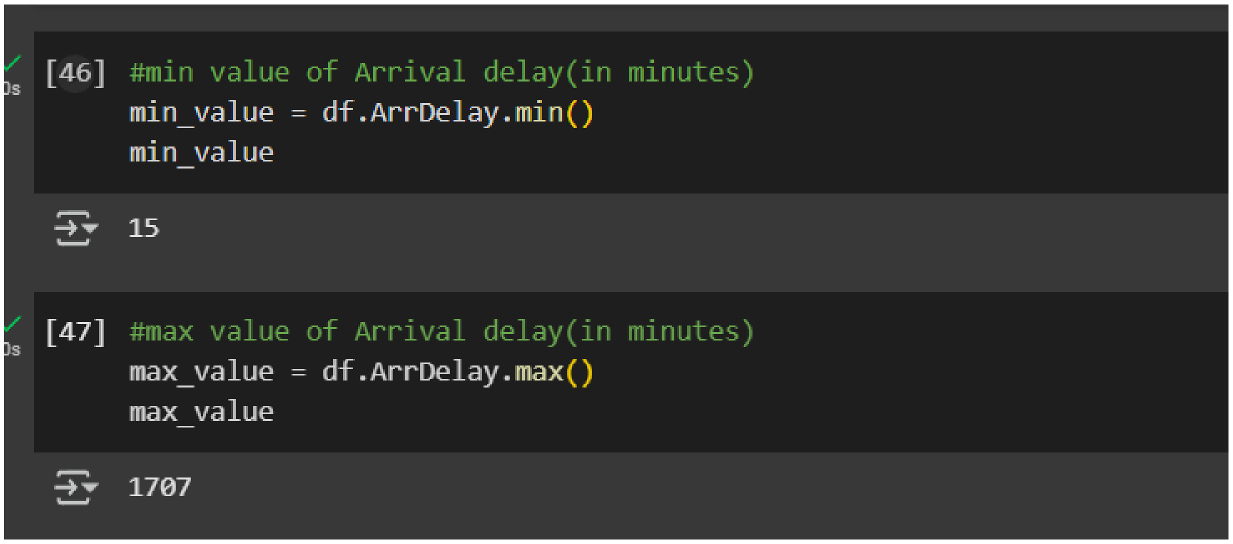


On which time delay happens mostly?



Correlation Matrix





```
[46] #min value of Arrival delay(in minutes)
     min_value = df.ArrDelay.min()
     min_value

     15
```

```
[47] #max value of Arrival delay(in minutes)
     max_value = df.ArrDelay.max()
     max_value

     1707
```

INSIGHTS

1. All the flights in the data of this dataset, have at least any one of the delays.

2. Southwest Airline company has the largest number of travel delay records

3. The maximum number of delays happened on FRIDAY

    a. A High Percentage of Delay occurs between 15:00:00 to 20:00:00 (i.e.) 3 PM to 8 PM.

4. The flights scheduled to depart at 3 PM to 8 PM delays mostly

5. The flights scheduled to depart at 12 AM to 5 AM - a smaller number of delays

6. Data is available for First half of the year 2019. From that first 6 months, March month has more delay records

7. Most of the delays are short timed. Compared to short time delay, minority of the delays are long timed.

8. The minimum Arrival delay is 15 Minutes & The maximum Arrival delay is 1707 Minutes

9. Chicago O'Hare International Airport is the airport where most frequently flights depart and arrive

10. American Eagle Airlines has taken the highest time in minutes for the carrier delay.

11. American Airlines Inc. has taken the highest time in minutes for the weather delay.

12. American Airlines Inc. has taken the highest time in minutes for the NAS delay.

13. Atlantic Southeast Airlines has taken the highest time in minutes for the Security delay.
14. United Airline Inc. has taken the highest time in minutes for the Late Aircraft delay.

15. Late Aircraft delay, Carrier Delay, and NAS delay shows most delay during the year.

16. Carriers with higher average delay generation are Hawaiian Airlines (HA) with 36.41 minutes per flight, Atlantic Southeast Airlines (EV) with 33.60 minutes per flight

17. The locations with high delay rates have very few flights. The locations with low delay rates have very high number of flight counts.

Part 3

Title: Predicting Used Car Prices Using Regression Trees

1. Introduction

This report presents a data-driven approach to predicting the prices of used Toyota Corolla cars using regression trees. The primary objective is to identify the most influential features affecting car prices and to build a predictive model that can estimate prices based on vehicle characteristics. Regression trees are particularly effective for this task due to their interpretability and ability to handle nonlinear relationships.

2. Methodology

2.1 Data Preprocessing

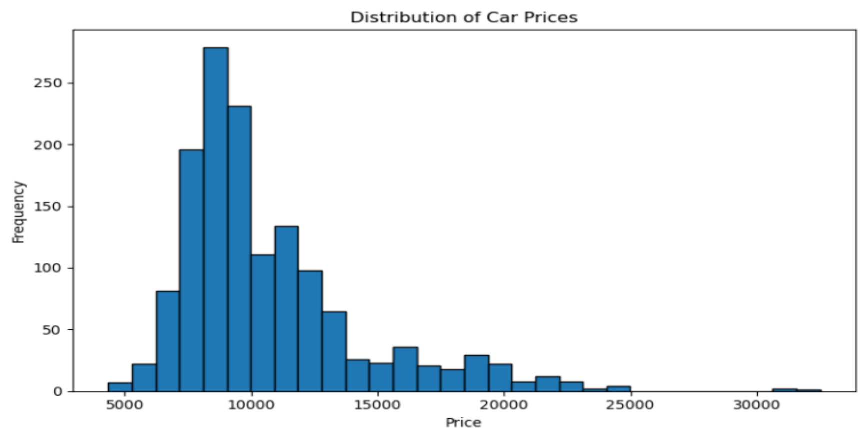
- Dataset: ToyotaCorolla.csv
- Removed irrelevant columns: Id, Model
- Converted categorical variable Fuel_Type into dummy variables using one-hot encoding
- Checked for and confirmed absence of missing values
- Split dataset into training (60%) and validation (40%) sets for model evaluation

2.2 Exploratory Data Analysis (EDA)

To better understand the target variable (Price), a histogram was plotted:

Figure 1: Distribution of Used Car Prices

Shows the frequency of various price points. Most cars are clustered between 10,000 and 15,000.



```
✅ Best Tree Depth: 6
📉 Train RMSE: 858.54
📈 Validation RMSE: 1228.07

🔥 Top 3 Important Features:
Age_08_04: 0.8745
Weight: 0.0416
KM: 0.0368
```

3. Model Development

3.1 Regression Tree Training and Tuning

A DecisionTreeRegressor from Scikit-learn was used for training. Grid search with 5-fold cross-validation was applied to find the optimal tree depth (max_depth from 2 to 15).

- **Best Tree Depth:** 6
- **Training RMSE:** ~859

- **Validation RMSE:** ~1228

These RMSE values indicate that the model fits the training data well and maintains reasonable predictive accuracy on unseen validation data, with minimal overfitting.

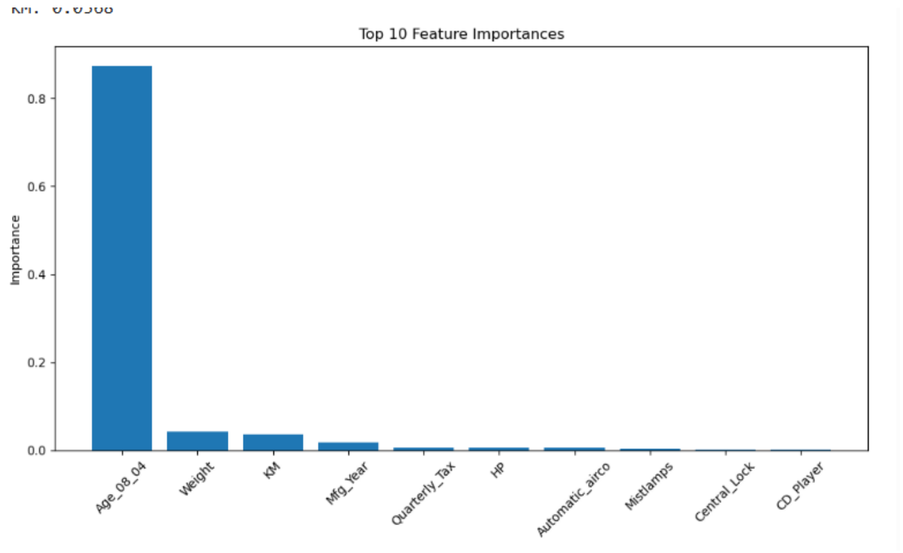
3.2 Feature Importance Analysis

The trained tree's feature importance was extracted. The top 3 predictors of price were:

1. **Age_08_04** □ Older cars depreciate more, hence have lower prices.
2. **Weight** □ Generally heavier cars tend to have more features and higher pricing.
3. **KM** □ Higher mileage tends to decrease value.

Figure 2: Top 10 Feature Importances

Displays which attributes the model relied on the most when making predictions.

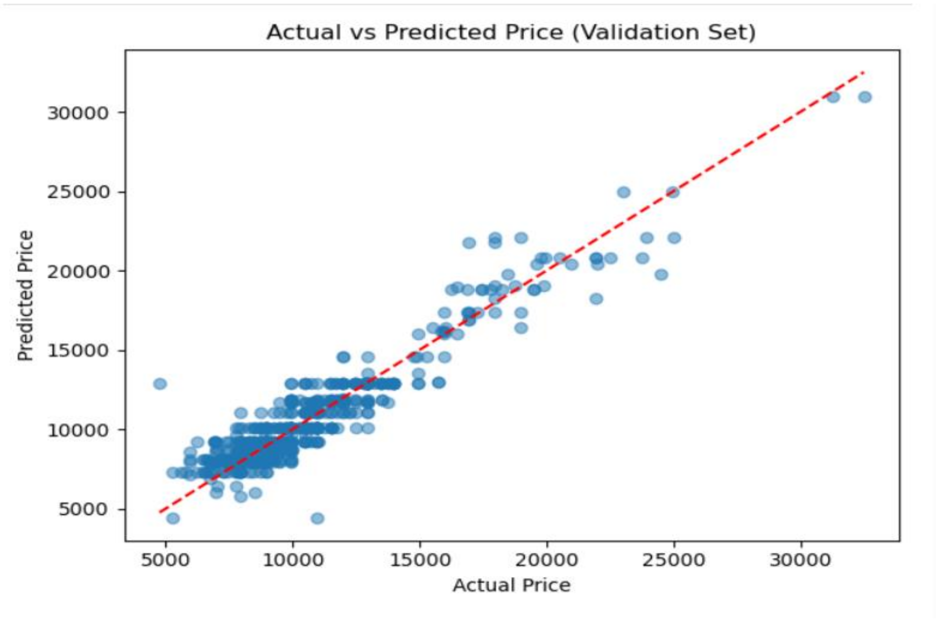


3.3 Predicted vs Actual Comparison

To visually assess model performance, predicted prices were plotted against actual prices for the validation set.

Figure 3: Actual vs Predicted Car Prices

A near-diagonal line indicates close prediction to actual values.



4. Results & Interpretation

- The regression tree model provided interpretable results with clear insights into what features drive car prices.
- **Age** of the vehicle was the single most important predictor, followed by **Weight** and **KM**.
- RMSE values showed a decent fit, with some overfitting manageable by pruning or regularization in future models.

5. Conclusion & Recommendations

Conclusion

This study confirms the effectiveness of regression trees in estimating used car prices. The model's interpretability allows businesses to understand what drives pricing decisions.

Recommendations

- Include more features like car condition, service history, or accident record for even better predictions.
- Explore ensemble methods like Random Forests or Gradient Boosted Trees to potentially improve accuracy.
- Keep models updated with recent data to maintain accuracy in changing market conditions.

Appendix

- Full Jupyter Notebook: prob 3case study 2.ipynb
- Dataset: ToyotaCorolla.csv
- Plots:
  - Figure 1: Histogram of Car Prices
  - Figure 2: Top 10 Feature Importances
  - Figure 3: Actual vs Predicted Prices