Flight Price Prediction - Group 4

Somaiah Kekada Arjuna (SXK230083), Shrutika Pujari (SXP230057), Neha Kumar (NXK230051),
Jayesh Saini (JXS230007), Ann Chepkoech (AXC230064)

Motivation

Airline ticket pricing is a dynamic and multi-faceted process influenced by demand, competition, seasonality, and operational costs. This project aims to analyze the flight booking dataset obtained from the "Ease My Trip" website, a platform for booking flight tickets. The primary goal is to understand price fluctuations and identify key factors influencing these variations by conducting various machine-learning models to develop meaningful insights for passengers, airlines, and travel agencies. Accurate price predictions benefit stakeholders such as:

- **Passengers**: Enable better planning and cost savings by booking at optimal times.
- **Airlines**: Assist in dynamic pricing strategies for maximizing revenue.
- **Travel Agencies**: Enhance recommendations and packages for customers.

Why This Topic Matters

The airline business strongly relies on data, and one case study that can take advantage of previous data is predicting flight pricing through machine learning techniques. Understanding these changes is extremely important to airlines, travel agents, and passengers. Season, demand, and flight features (e.g., number of stops and departure time) are some of the elements that greatly affect the price of airline tickets. We can provide significant insights into pricing variances by examining flight data and forecasting future ticket prices. Some other reasons involve:

- **Economic Impact**: The airline industry is heavily data-driven, and even small optimizations can bring significant economic advantages.
- **Research Gap**: Existing models often oversimplify the relationship between features or lack the sophistication to handle interactions among variables like time, demand, and route.
- **Practical Relevance**: Real-world applications include price monitoring tools, booking platforms, and operational decision-making support.

This project seeks to analyze the factors influencing airline ticket prices, focusing on the following key questions:

1. **Variation with Airlines**: Does the choice of airline affect ticket prices? Different airlines, such as budget vs. premium carriers, employ distinct pricing strategies, which will be explored by analyzing the airline column.

2. **Timing of Purchase**: How does the timing of purchase influence prices? Tickets bought closer to the departure date tend to be more expensive due to higher demand and limited availability, which will be analyzed by comparing ticket prices to the purchase date.
3. **Departure and Arrival Times**: Does the time of day affect ticket prices? Flights at peak times (e.g., early morning or late evening) may have higher prices, which will be examined by analyzing the departure and arrival times.
4. **Source and Destination**: How do origin and destination cities affect ticket prices? The route's popularity, demand, and distance between cities can influence pricing, which will be investigated using the 'From' and 'To' columns.
5. **Economy vs. Business Class**: How do prices differ between economy and business classes? Although the dataset doesn’t explicitly mention class, price patterns will be used to infer class distinctions and compare prices across airlines and routes.

Data

Dataset Overview & Source - The project utilized flight booking data collected over fifty days from Kaggle, spanning Business and Economy classes, to capture various pricing variations across India's six major cities, providing route diversity and varying demand patterns.

Dataset Description:

There are two datasets with 12 columns each and 300,153 rows for the two classes, Business and Economy. Data was collected for fifty days from February 11th to March 31st, 2022. The dataset contains detailed flight information, including flight schedules, routes, and ticket prices for India’s top six metropolitan cities. The key attributes in the dataset are:

- **ID:** Row number
- **Airline:** The airline operating the flight.
- **Flight:** The flight number.
- **Source City:** Departure city/airport.
- **Departure Time:** Departure time of the flight.
- **Arrival Time:** Time of arrival at the destination.
- **Stops:** Number of stops during the flight.
- **Destination City:** Destination city/airport.
- **Flight Class:** Economy or business class.
- **Duration:** Duration of the flight (hrs)
- **Days left:** Days left until departure during the time of booking.
- **Price:** The price of the flight ticket (target variable).

Data Cleaning and Preprocessing

The following measures were taken:

- Missing entries were checked in key columns like price, duration, and stops, and outliers were removed
- Irrelevant columns like ID and flight were removed as they were unnecessary.
- Class was renamed to flight_class for clarity.
- Label encoding was included to replace string values with an integer for categorical variables for KNN and Decision Tree models.
- One-hot encoding was used to transform categorical variables for linear regression.

Models Used

Linear Regression

Linear Regression was used as a baseline model to capture linear relationships between features and the target variable (flight price).

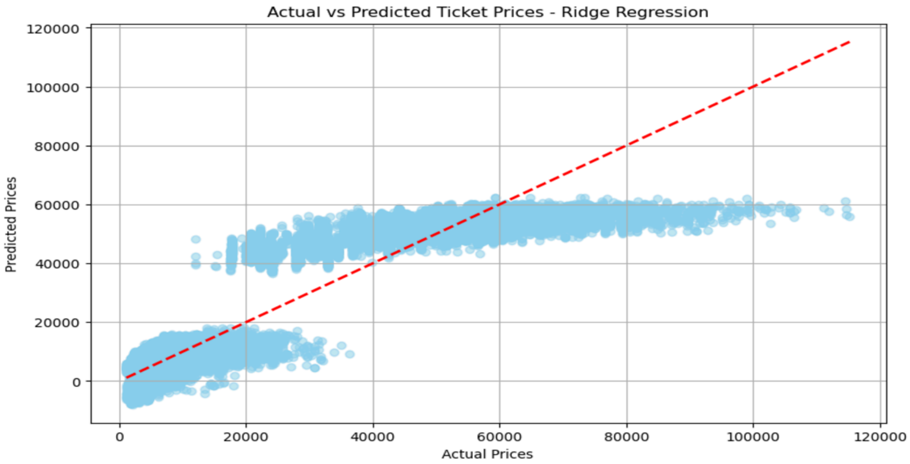
Hyperparameter Tuning:

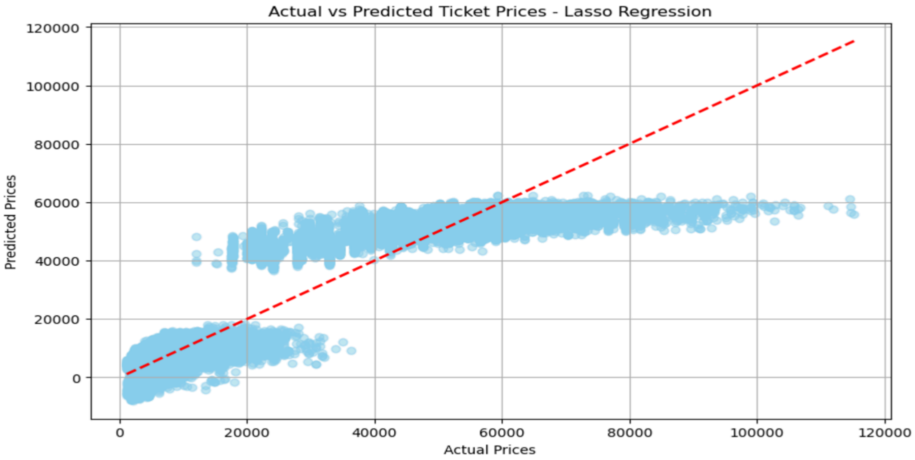
Grid Search: We used GridSearchCV to tune the model to get the optimal alpha value for both Ridge and Lasso, searching over a logarithmic scale of values [0.001,0.01,0.1,1,10,100,1000] with 5-fold cross-validation ensuring that each combination of alpha is evaluated thoroughly and the best model is selected based on R-squared performance.

Why These Choices?

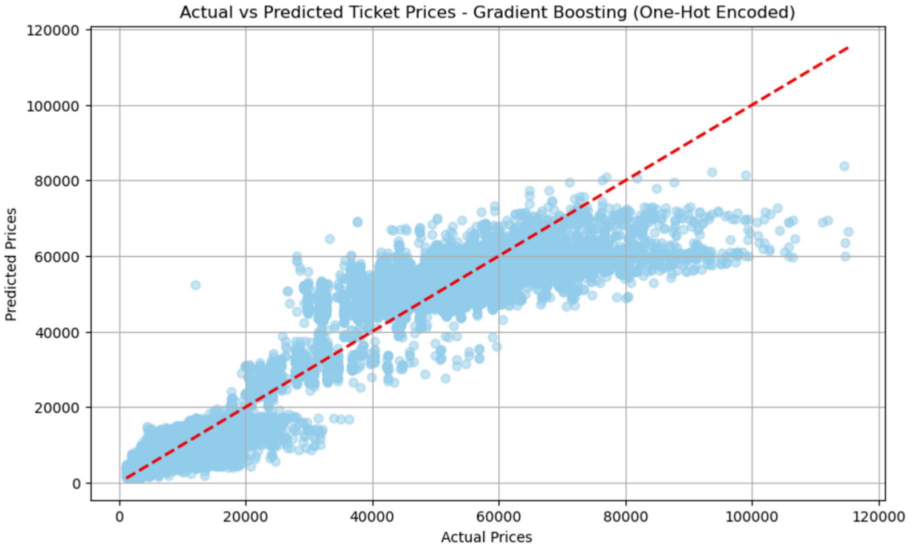
Ridge minimizes model complexity while retaining all features. Lasso simplifies the model by feature selection, removing less impactful predictors.

The graph below plots the actual vs predicted ticket prices for Ridge and Lasso Regression. Most of the points deviate from the red line, especially as the actual price increases. This suggests that the model tends to underestimate higher ticket prices. The variance in predictions for the second cluster (higher actual prices) implies that the model has difficulty accurately predicting Business class ticket prices compared to Economy class prices, most likely because business class tickets are higher and less frequent than economy class, so the model focuses more on accurately predicting the majority class (Economy), leading to underperformance for the minority class (Business prices).

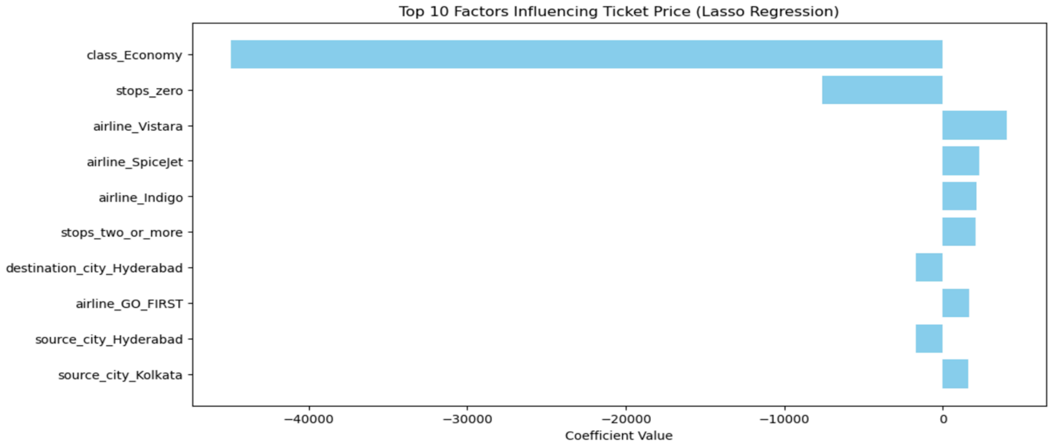




To solve this issue, gradient boosting was used as it reduces underestimation, improves accuracy for higher ticket prices, and enhances model interpretability.



Once gradient boosting was applied, there was a change in results. It has reduced the prediction errors for high ticket prices as there is a much tighter clustering of points around the red dashed line.



The graph above shows the top ten contributors to the variation in ticket pricing. The features with the negative coefficient, such as economy class, zero stops, or having Hyderabad as the destination city, are associated with low ticket pricing. Similarly, features such as Vistara Airlines or Indigo Airlines have a positive coefficient and increase ticket prices.

K-Nearest Neighbors (KNN)

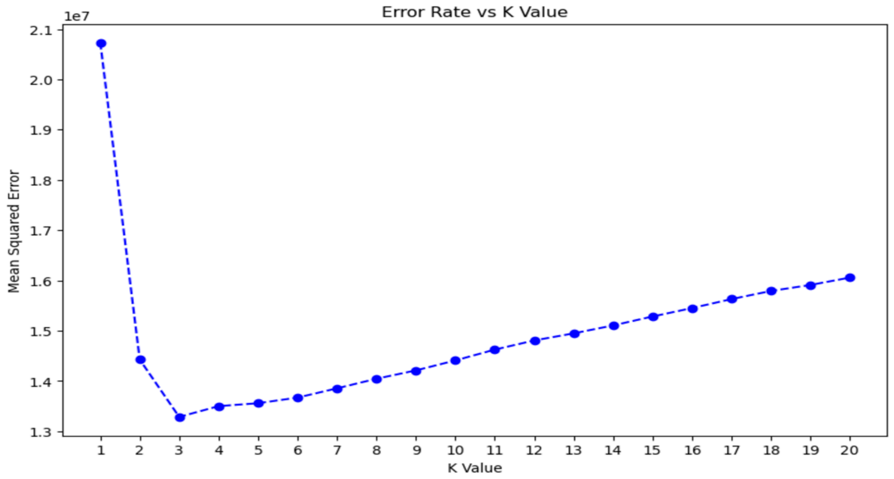
This model models non-linear relationships by predicting prices based on proximity to similar data points. The dataset is scaled using a standard scalar, which ensures that feature magnitudes are standardized. This step is crucial for KNN, as it is sensitive to feature scales.

Hyperparameter Tuning: The number of neighbors (k) tested values between 1 and 20.

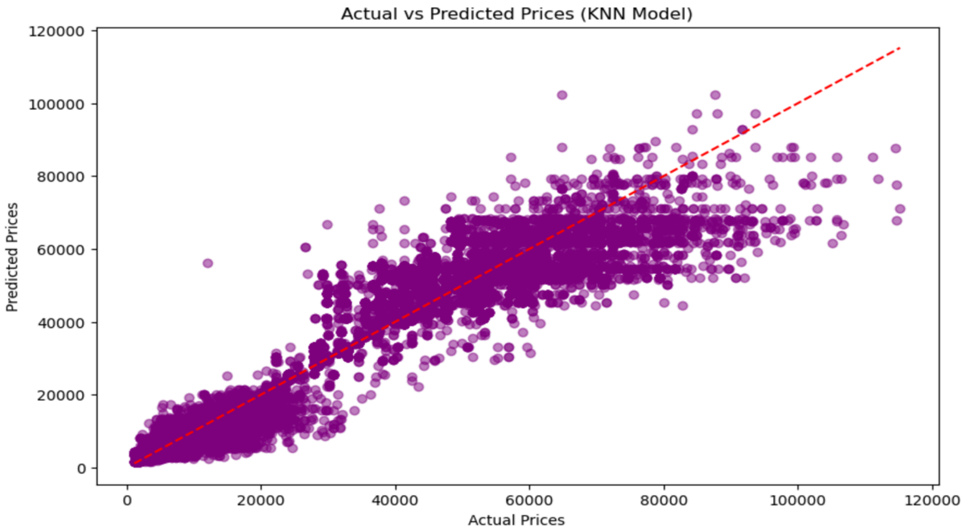
Grid Search: While we did not explicitly use GridSearchCV, we employed a cross-validation-like approach by iterating over k values and calculating the Mean Squared Error (MSE) for each. The error rates are stored, and the optimal k is selected as the one with the minimum error, determining the best hyperparameter.

Why These Choices?

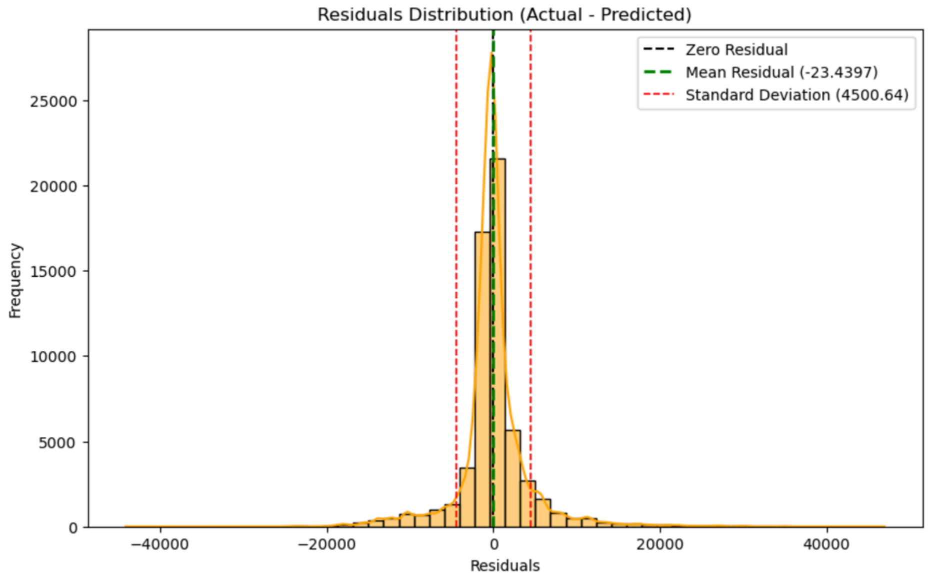
KNN is sensitive to k; a smaller value captures local patterns but risks overfitting, while larger values generalize better. The mean squared error value was checked for all the values of k between 1 and 20 and received 3 as our best k value.



The scatter plot below shows the relationship between the actual prices (on the x-axis) and the predicted prices (on the y-axis) in a K-Nearest Neighbors (KNN) regression model. There is a clear positive correlation between actual and predicted prices, suggesting the KNN model is capturing the relationship reasonably well. However, points deviate from the red line, especially at higher actual prices. This indicates that the predictions are less accurate for high-priced data points. Additionally, heteroskedasticity seems to be present as the spread of residuals appears to increase with the actual price.



The residual distribution plot below indicates that the model's residuals are mostly centered around zero, with a reasonable spread, and follow a roughly normal distribution. This suggests that the model has performed well in predicting the values, with errors that are mostly small and balanced around the true values.



Decision Trees

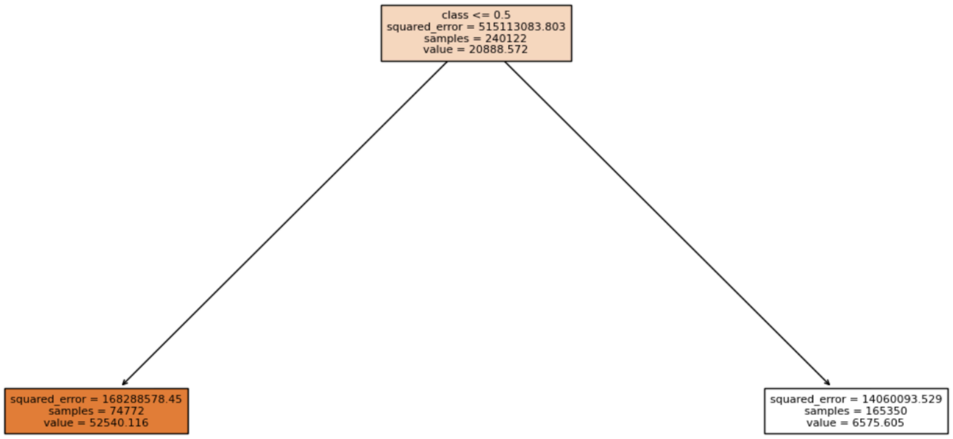
This model captures feature interactions effectively, even with non-linear and categorical variables.

Hyperparameter Tuning:

- **Max Depth**: Limited to 10 to prevent overfitting.
- **Splitting Criterion**: Feature importance

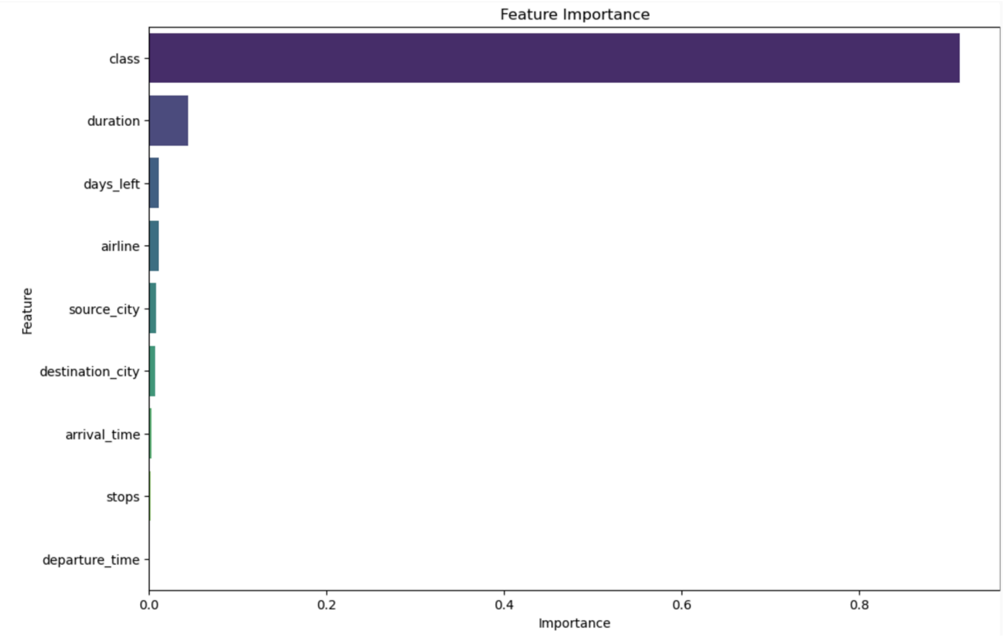
Why These Choices?

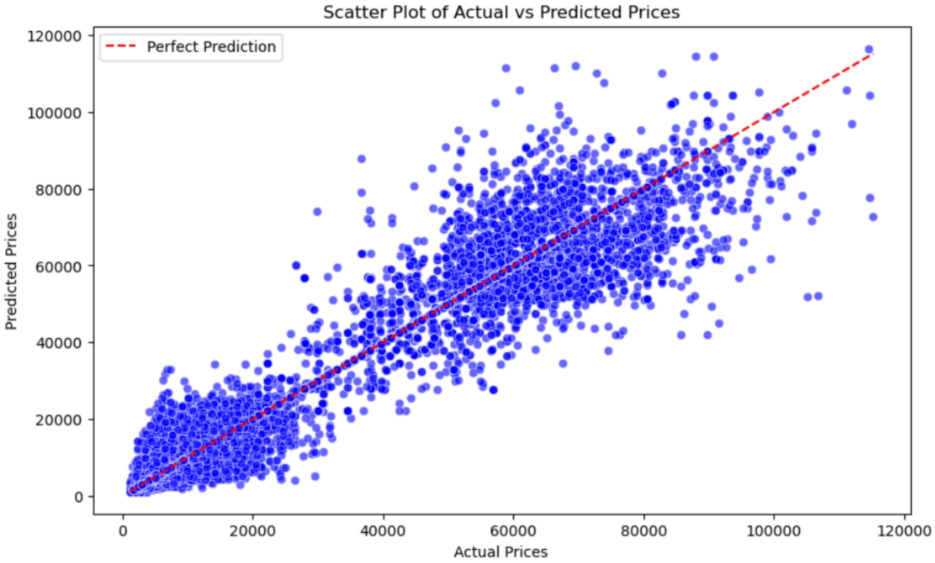
A max depth of 10 prevents the model from splitting too deeply, preventing overfitting. Additionally, we limited the splitting to feature_importance > 0.05 to identify the most important features. The purpose is to create a more compact and interpretable decision tree model by focusing on the most important features and not creating a decision tree with too much clutter that takes too long to run. By visualizing this new model, you can better understand the relationships between the important features and the target variable.



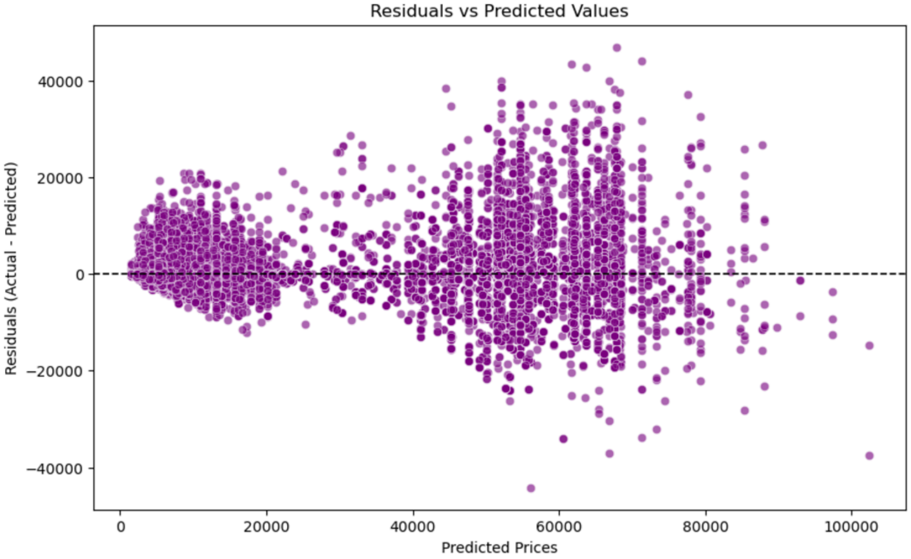
The decision tree visualization created was split according to class. The model shows that the class <= 0.5 group has a higher predicted value, while the class > 0.5 group has a lower predicted value. For class <= 0.5, it is likely characterized by features or conditions that lead to higher values of the target variable (price), meaning that it can be associated with business class. Therefore, the class > 0.5 group is associated with the economy class.

This decision makes sense as class is the most important feature in the model.





This scatter plot compares actual flight prices with predicted prices. The points closely align with the red "perfect prediction" line, showing that the model accurately captures price trends. Most predictions fall near the line, indicating strong performance. However, there's a slight spread at higher price levels, suggesting the model may struggle a bit with extremely high values. Overall, the model provides reliable predictions, though it could benefit from adjustments to improve accuracy for higher-priced flights.



The spread or variance of the residuals does not appear to be constant across the range of predicted values. The residuals show a wider spread at lower predicted values and a narrower spread at higher predicted values. This pattern suggests the presence of heteroscedasticity, which means the variance of the residuals is not constant.

Model Comparison

Performance Metrics

Model	RMSE	R2 Score	MAE	MSE	Key Notes
Gradient Boosting	5001.12	0.95	2518.45	20255930.09	Best performance among non-ensemble models with balanced generalization.
Linear Regression (Ridge)	6761.71	0.9113	-	-	Best Ridge Alpha: 1.0; Linear models struggle with

					non-linear data.
Linear Regression (Lasso)	6761.71	0.9113			Best Lasso Alpha: 0.001; Performance similar to Ridge.
K-Nearest Neighbors (KNN)	3645.10	0.97	1670.23	13286741.25	Best K value: 3; Strikes a balance between overfitting and underfitting.
Decision Tree	4500.66	0.96	2518.45	20255930.09	High variance but performs well due to its structure; prone to overfitting.

Performance Summary

1. **Gradient Boosting**: The model performed well with an RMSE of 5001.12 and an R^2 of 0.95, showing good accuracy and generalization. Residual analysis revealed no systematic bias, with errors centered around zero.
2. **K-Nearest Neighbors (KNN)**: This was the best-performing model, achieving an RMSE of 3645.10 and an R^2 of 0.97. The optimal value of K=3 helped avoid overfitting while keeping the accuracy high.
3. **Ridge and Lasso Regression**: Both models had similar performance, with an RMSE of 6761.71 and an R^2 of approximately 0.91. These models faced challenges in capturing non-linear relationships, leading to slightly lower performance.
4. **Decision Tree:** It performed well with an RMSE of 4500.66 and an R^2 of 0.96; though its high training and test R^2 scores suggest slight overfitting, it still delivers reliable predictions.

Discussion

Model Performance Analysis

- **Gradient Boosting**: This model performed very well on structured data thanks to its robustness against outliers and iterative learning process. While it achieved a strong R^2 of 0.95, it lags slightly behind KNN, indicating that there's room for improvement in capturing finer details.
- **K-Nearest Neighbors (KNN)**: With the best R^2 of 0.97 and the lowest RMSE of 3645.10, KNN performed exceptionally well. However, KNN is sensitive to distance metrics, meaning that its performance relies on proper feature scaling. It excelled with the optimal K=3, which balanced accuracy and avoided overfitting.
- **Ridge and Lasso Regression**: Both models struggled because of their linear assumptions, which limited their ability to capture non-linear relationships in the data. While regularization helped reduce overfitting compared to plain linear regression, it still wasn't enough to achieve high accuracy, resulting in lower performance than Gradient Boosting and KNN.
- **Decision Tree**: It exhibited signs of overfitting due to the high number of leaves (874), which indicates that the model excessively captured noise and nuances in the training data. However, its hierarchical splitting capability enables it to identify complex feature interactions, allowing for strong predictions despite reduced generalization to unseen data.

Overall, KNN emerged as the top performer, with Gradient Boosting offering a strong alternative, while Decision Tree delivered reliable predictions despite overfitting issues, and Ridge and Lasso regression were less effective due to their limitations in handling non-linearity.

Trade-offs

1. **Accuracy vs Interpretability**:
   - Gradient Boosting and KNN excel in accuracy but lack interpretability.
   - Ridge and Lasso are easier to interpret but fall short on performance.
   - Decision Tree offers moderate accuracy with better interpretability due to its hierarchical structure but can overfit, limiting its generalizability.
2. **Complexity vs Scalability**:
   - KNN is computationally expensive for larger datasets.
   - Gradient Boosting strikes a better balance, offering scalability with reasonable computational cost.
   - Decision Tree is computationally efficient and scalable but may sacrifice accuracy for simplicity in certain cases.

Residual Analysis

- **Centered Residuals**: All models, including the Decision Tree, exhibit residuals that are close to zero, indicating minimal systematic bias. This means the models are not consistently overestimating or underestimating true values.
- **Spread**: Gradient Boosting and KNN show narrower spreads in their residuals, suggesting lower error variability and consistent predictions. The Decision Tree shows a slightly wider spread, likely due to its tendency to overfit, resulting in greater variability in predictions.
- **Shape**: The residuals for the best-performing models, Gradient Boosting and KNN, follow a near-normal distribution, indicating reliability and proper capture of underlying data patterns. The Decision Tree's residuals, however, may deviate slightly from normality due to its overfitting, reflecting some structured error patterns rather than purely random distribution.

Appendix

After analyzing the initial results of the linear regression model, it was observed that the model's accuracy was not satisfactory. This could be due to its inability to capture complex, non-linear relationships in the data. Therefore, we included gradient boosting in the linear regression model after reviewing how to improve the accuracy. By adding Gradient Boosting to the model, we can take advantage of its ability to handle non-linear relationships, interactions between features, and the ability to fit complex data patterns more effectively than linear regression.

The hybrid model (combining linear regression and Gradient Boosting) benefited from both the simplicity and interpretability of linear regression and the powerful predictive capabilities of Gradient Boosting. This combination enabled the model to perform better on the dataset by addressing both linear and non-linear trends, leading to higher predictive accuracy compared to the initial linear regression-only model.