288R Project Progress Report – *Flight Fare Forecasting—Machine Learning for Cost-Effective Travel*

Project Group #: 17

Authors: Sami Eltawil, Indumini Jayakody, Andre Melo, Nikolai Pastore
Emails: seltawil@ucsd.edu, ijayakody@ucsd.edu, amelo@ucsd.edu, npastore@used.edu

Background

The goal of this project is to develop a machine learning model that can accurately predict airfare prices. Given the dynamic nature of airline pricing, such a model has practical value for both consumers and industry stakeholders. This model is not only expected to optimize fare predictions but also contribute to cost-saving and booking efficiencies. By leveraging machine learning, we aim to provide a data-driven approach to understanding airfare trends, offering valuable insights for both passengers and the airline industry.

Dataset

Our dataset, titled *Flight Prices*, consists of detailed records of airline ticket prices. It includes comprehensive information such as fare breakdowns, flight durations, flight segments, departure and arrival times, airport codes, and airline names. Sourced from Kaggle, this dataset serves as the cornerstone for our predictive model and analytical efforts.

Data Pipeline

The flight fare dataset contains detailed information about airline ticket prices, including fare details, travel duration, and flight segments. Each record represents a flight search entry, capturing key attributes such as departure and arrival times, airport codes, airline names, and pricing breakdowns. The table below provides a comprehensive data dictionary, describing the structure, data types, and example values for each column. Understanding these attributes is crucial for effective preprocessing and feature engineering in our predictive modeling pipeline.

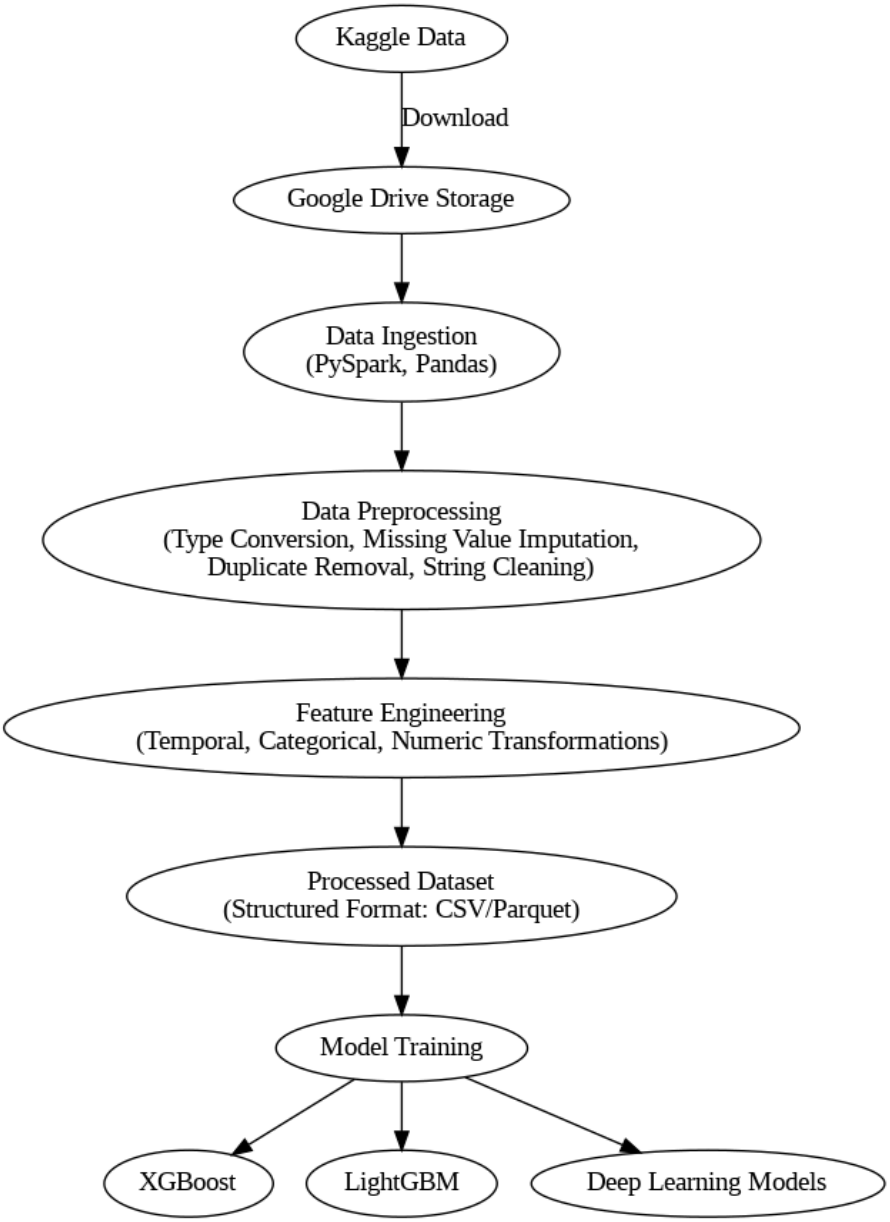
	Column Name	Data Type	Description	Example Value
0	legId	object	An identifier for the flight.	9ca0e81111c683bec1012473feefd28f
1	searchDate	datetime64[ns]	Date when this entry was recorded from Expedia.	2022-04-16 00:00:00
2	flightDate	datetime64[ns]	Date of the flight.	2022-04-17 00:00:00
3	startingAirport	category	Three-character IATA code for the departure airport.	ATL
4	destinationAirport	category	Three-character IATA code for the arrival airport.	BOS
5	fareBasisCode	category	The fare basis code.	LA0NX0MC
6	travelDuration	object	Total travel duration in hours and minutes.	PT2H29M
7	elapsedDays	Int64	Number of elapsed days (usually 0).	0
8	isBasicEconomy	boolean	Indicates whether the ticket is for basic economy.	False
9	isRefundable	boolean	Indicates whether the ticket is refundable.	False
10	isNonStop	boolean	Indicates whether the flight is non-stop.	True
11	baseFare	float64	Base price of the ticket (in USD).	217.670000
12	totalFare	float64	Total price of the ticket including taxes and fees.	248.600000
13	seatsRemaining	Int64	Number of seats remaining.	9
14	totalTravelDistance	float64	Total travel distance. This data is sometimes missing.	947.000000
15	segmentsDepartureTimeEpochSeconds	Int64	Unix time for departure of each segment. Entries are separated by '||'.	1650214620
16	segmentsDepartureTimeRaw	datetime64[ns, UTC-04:00]	ISO 8601 formatted departure time for each segment. Entries are separated by '||'.	2022-04-17 12:57:00-04:00
17	segmentsArrivalTimeEpochSeconds	Int64	Unix time for arrival of each segment. Entries are separated by '||'.	1650223560
18	segmentsArrivalTimeRaw	datetime64[ns, UTC-04:00]	ISO 8601 formatted arrival time for each segment. Entries are separated by '||'.	2022-04-17 15:26:00-04:00
19	segmentsArrivalAirportCode	category	IATA code for arrival airport of each segment. Entries are separated by '||'.	BOS
20	segmentsDepartureAirportCode	category	IATA code for departure airport of each segment. Entries are separated by '||'.	ATL
21	segmentsAirlineName	category	Name of the airline for each segment. Entries are separated by '||'.	Delta
22	segmentsAirlineCode	category	Two-letter airline code for each segment. Entries are separated by '||'.	DL
23	segmentsEquipmentDescription	category	Type of airplane used for each segment. Entries are separated by '||'.	Airbus A321
24	segmentsDurationInSeconds	Int64	Duration of the flight (in seconds) for each segment. Entries are separated by '||'.	8940
25	segmentsDistance	float64	Distance traveled (in miles) for each segment. Entries are separated by '||'.	947.000000
26	segmentsCabinCode	category	Cabin code for each segment (e.g., coach). Entries are separated by '||'.	coach

Our data pipeline is designed to efficiently process and prepare the flight fare dataset, ensuring it is structured for machine learning models. Given the dataset's large size (82+ million rows), we prioritize scalability and computational efficiency. The pipeline begins with data ingestion, where we download the dataset from Kaggle and store it in Google Drive for persistent access. Initially, Pandas was used for preliminary exploration, but given the large data volume, we transitioned to PySpark, which enables distributed computing and parallel processing. PySpark allows us to define a schema beforehand, ensuring each column is correctly formatted.

In the data preprocessing stage, we convert date-related columns into datetime format and ensure numerical and categorical variables are assigned appropriate data types. Missing values are handled using various strategies, such as imputing missing travel distances based on the average for each route and filling categorical gaps with "Unknown". We also remove duplicate entries to maintain data integrity and process multi-value columns, where values are separated by "||", extracting only the first available entry to standardize the dataset. Additionally, we clean and trim string-based columns to remove formatting inconsistencies.
To enhance predictive accuracy, we implement feature engineering by transforming the travel duration feature into a numeric format and calculating time until departure by measuring the difference between searchDate and flightDate. Categorical variables, such as airline and airport

codes, are encoded to ensure compatibility with machine learning models. Finally, the processed dataset is saved in a structured format, rendering it ready for model training. This pipeline ensures that the dataset is optimized for efficient computation while preserving key information necessary for accurate airfare prediction.

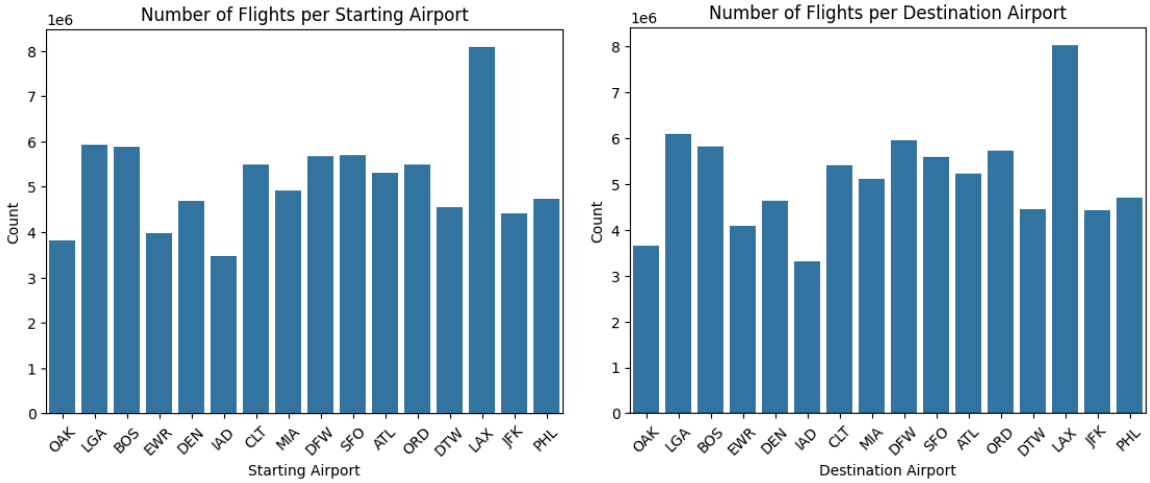
Below, is a schematic diagram of the data pipeline. This is provided to visually outline each step from ingestion to preprocessing and feature engineering.



EDA Description

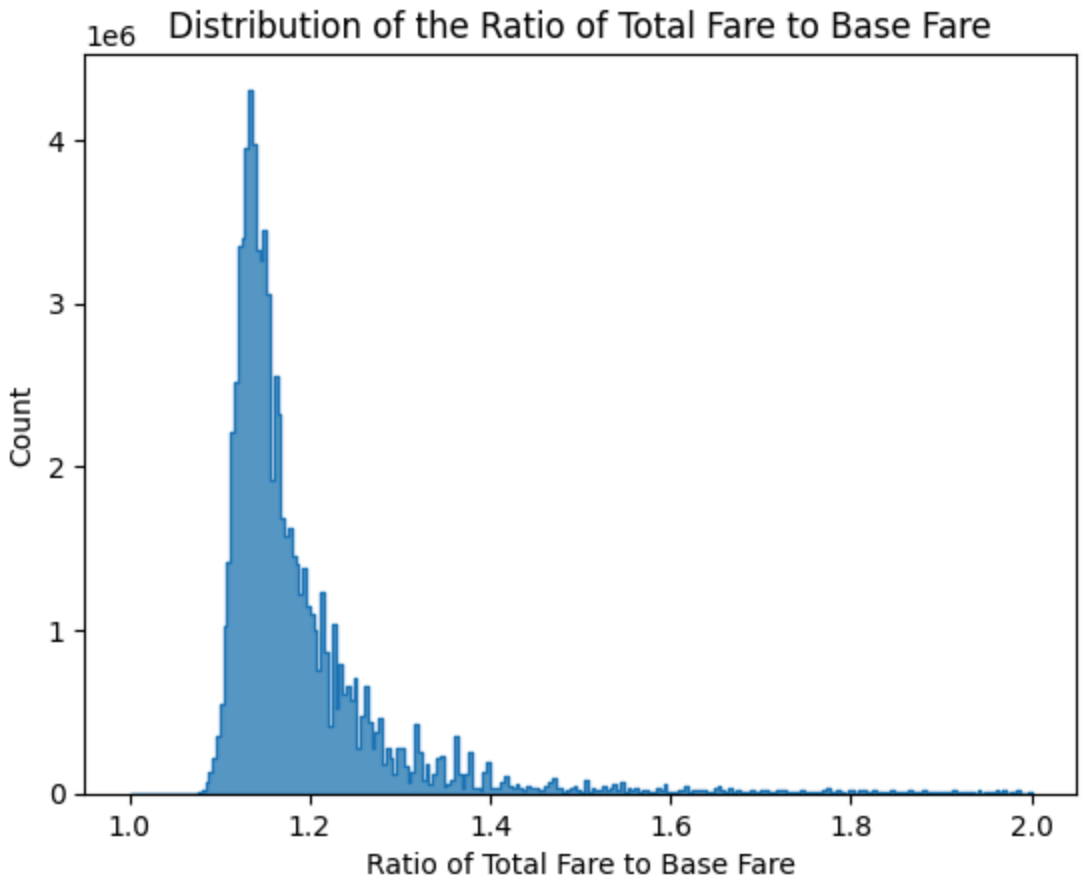
Because the dataset with which we are dealing is so bulky, and because the availability of big data visualization tools is somewhat lacking, we are largely restricted to univariate analyses in the graphical aspects of our exploratory data analysis. We proceed to outline the major findings of our exploratory data analysis below. We begin by analyzing the number of flights on a per-airport basis, which is largely summarized by the plots below.
Due to the dataset's scale, we optimized our EDA process by:

- Using a 1% stratified random sample (~800K rows) to maintain data distribution while also improving speed and usability.
- Computing summary statistics for only essential columns (baseFare, totalFare, totalTravelDistance) instead of other features that may not be as useful or relevant.
- Detecting missing values only for critical variables rather than the full dataset.
- Identifying outliers using percentile-based thresholds instead of boxplots, which are inefficient on large datasets.



These subtly different plots indicate that there are exactly sixteen unique starting airports and sixteen unique destination airports encompassed by our dataset. Moreover, these plots reveal a relatively uniform balance of classes both with respect to a given flight's starting/destination airports. This provides much assurance that the starting/destination airports may be easily manageable, worthwhile features to incorporate in training our predictive models and that no augmentation/downsampling would likely be necessitated to accommodate the use of such features.

We are also interested in the distribution of the ratio of total fare to base fare, which we observe through the plot below.



This distribution clearly reflects a rather low level of variance, suggesting that the total fare may be inferred fairly reliably from the base fare using the conversion factor specified by the peak of the above distribution (i.e., roughly, 1.15). Nonetheless, even though there appears to be hardly any variance in the above distribution, there is enough to suggest that using such a constant conversion may be faulty, even if only with very low probability. This suggests using constant conversion as a baseline in inferring total fare from base fare. Against this baseline, we would compare the performance of our predictive approach based on multiple regression. That is, we consider it worthwhile to adopt two labels (i.e., base fare and total fare) in training our models. One way of evaluating how our multiple regression model performs would be to compare the test accuracy of our model in predicting total fare to the baseline of the total fare simply predicted as a constant scalar multiple of the base fare.

Feature Engineering

There are many different features contained in our dataset, making it rather challenging to work with. Certainly, some basic feature engineering will likely be necessary for the models we plan to train. For example, the dataset contains a flight date in YYYY-MM-DD format per each flight entry. While this feature may indirectly suggest useful information for predicting a given flight price (e.g., in the month of the year or day of the week), we must represent this implicit

information in a format that is amenable for explicit processing. For our model, the year has no relevance since we already know that all entries in the dataset come from the same year (2022). However, we may easily extract the month of the year per flight date, and we may one-hot-encode months of the year to render this feature as a categorical variable. Similarly, we may extract and one-hot-encode days of the week as a separate categorical feature.

Our EDA revealed that the dataset contains features which identify starting and destination airports and that there are a total of sixteen unique values for each of these two categorical fields. It is intuitively sensible to assume that the airports associated with a given flight may influence the flight cost, so we plan to incorporate these features in training our model. To facilitate training, we could simply one-hot-encode all airport identifiers. On the other hand, a more sophisticated alternative approach might be to consider (and one-hot-encode) pairwise interactions between these two variables. This approach would seem rather sensible since it might be reasonable to assume that a given flight's price would depend largely on the route, which is determined effectively by the starting and destination airports. However, given the computational expense of deriving pairwise interactions and the unwieldy size of our dataset, this prospect remains strictly tentative.

The dataset also contains a column that indicates each flight's duration, and we anticipate that such information may be relevant for predicting flight prices. Because this information is formatted as a string with units, the only feature engineering necessary for this variable will likely be some basic string manipulation and subsequent casting to a numeric data type.

Additionally, the dataset contains three boolean variables concerning whether a given flight is a basic economy flight, whether a given flight is refundable, and whether a given flight is nonstop. All such boolean variables are, of course, categorical, and they may therefore be straightforwardly accounted for through one-hot-encoding. However, our EDA also suggests that many of these variables reflect significant class imbalance. For some of these variables, we may potentially account for class imbalance by downsampling our data at certain phases of model development and training. While direct augmentation would perhaps be a more preferable alternative, there is no apparent feasible means to that end, given the numerical nature of our ground truth labels. Moreover, even downsampling may not be effective for the variable that describes whether a given flight is refundable, since the class imbalance for this feature is so severe that nearly every instance in the dataset belongs to the same class (with respect to this feature). As such, we plan to drop this variable entirely.

Several feature transformations that were made to improve model performance include the following.

For most instances in the dataset, the total travel distance (in miles) is also provided, and we also expect this type of information to be potentially relevant in predicting flight prices. However, as revealed by our EDA, this variable is missing in a nontrivial proportion of data instances. Since this variable is numeric, we simply plan to impute missing values with the mean.

For ground-truth labels, we consider the variables that describe the base fare and the total fare (in dollars) for a given flight. While these variables are formatted as strings in the raw dataset, they are semantically numerical and are already rendered as such upon being processed through our data pipeline.

Models

The complexity of airfare pricing, influenced by multiple temporal, spatial, and categorical features, necessitates the use of machine learning models capable of capturing intricate relationships within the data. In our approach, we focus on two powerful gradient boosting models—XGBoost (Extreme Gradient Boosting) and LightGBM (Light Gradient Boosting Machine)—due to their effectiveness in structured data tasks, ability to handle mixed feature types, and computational efficiency. Additionally, we explore deep learning architectures for tabular data, which can dynamically learn feature representations and capture complex, non-linear interactions that may be difficult to model explicitly using traditional machine learning techniques.

Our choice of models is guided by previous work in airfare prediction, such as that conducted by Tziridis et al. (2017) and Biswas et al. (2022), which demonstrated the effectiveness of machine learning techniques in predicting flight prices within constrained settings. We aim to extend these methodologies to a broader, more detailed dataset, evaluating the effectiveness and generalizability of our approach.

Other methods and model stacking strategies may be used in future work to further enhance predictive accuracy as well as more basic models in order to create stronger baselines.

XGBoost

XGBoost is a powerful and efficient gradient-boosting algorithm that builds on traditional boosting methods by adding regularization, parallel processing, and smarter tree pruning techniques. It works particularly well for our dataset because it can handle both high-dimensional categorical data—like airline names and departure locations—and numerical features such as flight distance and duration. One of its biggest strengths is its built-in regularization, which includes L1 (Lasso) and L2 (Ridge) penalties to help prevent overfitting. Since airfare prices can be highly volatile due to seasonal trends, sudden demand shifts, and external factors, XGBoost's ability to model complex, non-linear relationships while still generalizing well makes it especially useful. Additionally, it applies a second-order Taylor approximation to optimize the loss function, leading to faster convergence and better predictive accuracy. Another key advantage is that XGBoost can handle missing data automatically, learning optimal split directions without requiring explicit imputation. To get the best performance, we plan to

fine-tune hyperparameters such as learning rate, maximum tree depth, number of estimators, and regularization terms using grid search and Bayesian optimization. We also plan to analyze feature importance scores to better understand which factors have the biggest influence on airfare pricing.
We will use k-fold cross-validation to ensure that our model generalizes well to unseen data.

LightGBM

We also plan to use LightGBM, another gradient boosting framework that is designed for speed and memory efficiency. Unlike XGBoost, which grows trees level by level, LightGBM takes a leaf-wise approach, which helps it achieve higher accuracy with fewer iterations. This makes it particularly useful for our dataset, which contains thousands of flight records with a mix of categorical and numerical features. LightGBM also uses a histogram-based splitting method, which cuts down on computation time and memory usage, making it highly scalable for large datasets. Another major advantage is that it handles categorical data natively, reducing the need for extensive preprocessing. This is especially helpful in airfare prediction, where categorical variables—like airlines, travel classes, and departure airports—play a key role in determining ticket prices.

Additionally, LightGBM is well-suited for dealing with imbalanced data, which is important given that airfare distributions can be skewed due to last-minute bookings and premium fare categories. Like XGBoost, we plan to fine-tune LightGBM by adjusting hyperparameters such as the number of leaves, learning rate, and boosting type, ensuring we strike the right balance between accuracy and interpretability. To better understand how the model makes predictions, we also plan to use SHAP (SHapley Additive exPlanations) to analyze feature importance, giving us insights into which factors have the biggest influence on airfare fluctuations.

Comparative performance analyses between LightGBM and XGBoost will be conducted to determine the most effective approach to reach our desired results.

Further Progress Report Details

As we progress in our project, our primary focus remains on refining the data pipeline and ensuring that the dataset is fully prepared for model training. The preprocessing phase is now operational, allowing us to efficiently clean, transform, and structure the dataset. However, we are still in the process of feature selection and engineering, particularly in handling categorical variables such as airline and airport codes. We are exploring different encoding techniques, including one-hot encoding and target encoding, to determine the most effective way to represent these variables for machine learning models. Additionally, we are working on strategies to handle missing values beyond simple imputation, such as leveraging clustering techniques to estimate missing totalTravelDistance values based on similar routes.

While we have not yet successfully trained initial versions of XGBoost and LightGBM, we are actively preparing for the modeling phase by evaluating the best hyperparameter tuning strategies and determining the optimal training setup. One challenge we are addressing is the high memory requirement for training models on such a large dataset. We are considering subsampling strategies and utilizing distributed computing techniques to facilitate efficient model training. Additionally, we plan to implement baseline models, such as linear regression and decision trees, to establish a performance benchmark before moving on to more complex approaches. In the coming weeks, our efforts will be directed toward finalizing feature engineering, implementing scalable model training workflows, and conducting exploratory training runs to evaluate the feasibility of different machine learning approaches.

Team Member Contribution

Andre contributed to drafting the project abstract and developing the data pipeline and added to multiple sections within this progress report.

Indumini contributed to exploratory analysis and added to multiple sections within this progress report.

Sami added to multiple sections within this progress report and developed plot visualizations that include all of our data instead of just a sample.

Niko generated a schematic diagram of the data pipeline, added to multiple sections within this progress report, and established the group's GitHub repository.

As we proceed with the next phases of our project, we anticipate that most of our efforts will be devoted to model training. As mentioned earlier, we will likely divide the entire model training process amongst all of ourselves to accommodate for the challenging volume of data with which we are dealing with.

Risks and Mitigation

One of the biggest challenges we currently face in our project is the sheer volume of data we're working with. This has already posed significant obstacles, particularly during exploratory data analysis (EDA), where long computation times and limited big data visualization options have made it difficult to extract insights efficiently. While there are techniques available for training models on large datasets, they come with trade-offs—primarily in terms of limited model selection and computational demands. Even within the subset of models that can handle big data, training and testing on the full dataset is likely to be a time-consuming process.

At this stage, our best strategy for managing these challenges is to take an iterative, team-based approach. One potential method is to randomly partition the dataset, distribute the partitions among team members, and have each member process and analyze their respective subset

independently. By regularly validating results across partitions, we can ensure consistency and reproducibility while keeping computation times manageable. This process may need to be repeated multiple times for different models to confirm that our findings are reliable across various segments of the data. While this approach is not without its limitations, it provides a practical, scalable way to navigate the computational constraints of working with big data. Additional risks, such as potential biases in data collection and model scaling will continue to be watched and plans for handling biases will be developed as needed.

References

Abdella, J. A., Zaki, N. M., Shuaib, K., & Khan, F. (2018). *Airline ticket price and demand prediction: A survey*. *Journal of King Saud University – Computer and Information Sciences*.

Biswas, P., Chakraborty, R., Mallik, T., Chakraborty, R., Uddin, S. I., Saha, S., Das, P., & Mitra, S. (2022). *Flight price prediction: A case study*. *International Journal for Research in Applied Science & Engineering Technology (IJRASET), 10*(VI).

Tuli, M., Singh, L., Malik, N., & Tripathi, S. (2023). *Prediction of flight fares using machine learning*. In *2023 IEEE Conference*.

Tziridis, K., Kalampokas, T., Papakostas, G. A., & Diamantaras, K. I. (2017). *Airfare prices prediction using machine learning techniques*. In *2017 25th European Signal Processing Conference (EUSIPCO)*.

Wong, D. (2022). *Flight prices* [Data set]. Kaggle. https://www.kaggle.com/datasets/dilwong/flightprices