**Final Capstone Project:**

**Monthly Passenger Count Prediction for the San Francisco International Airport**

Team 1: Jamileh Jahangiry, Prachi Khanna and Se’Lina Lasher

Shiley-Marcos School of Engineering, University of San Diego

AAI 590: Capstone Project

Anna Marbut M.S.

August 12, 2024

# Table of Contents

[Table of Contents 1](#_pflq71lep468)

[Introduction 2](#_72sca0adh93k)

[Data Summary 2](#_42pvvvftidiv)

[Background on Chosen Experimental Methods 4](#_vfe7x6a9jnsc)

[Experimental Methods and Trainings 8](#_hdupw1tt8jkh)

[Results and Conclusions 13](#_fs6z2qjdx2w5)

[References 16](#_9ahwxe3lky9x)

[Visualizations 19](#_346yf5b6lix1)

# Introduction

In this project, we aim to predict the monthly total passenger count at San Francisco International Airport (SFO) using historical data from 2005 to 2018. Accurate passenger forecasting is crucial for airport management, as it enables better resource allocation, staff scheduling, and operational planning (Kamath, 2024). It also helps airlines optimize flight schedules and manage capacity, ultimately enhancing the passenger experience (Kamath, 2024).

Our dataset, sourced from Kaggle's [SF Air Traffic Passenger and Landings Statistics](https://www.kaggle.com/datasets/san-francisco/sf-air-traffic-passenger-and-landings-statistics), is provided by the city of San Francisco. This comprehensive dataset includes detailed records of passenger counts and landings, which we use to develop our predictive models.

The project has a dual focus: not only do we strive to build an accurate predictive model, but we also seek to evaluate whether advanced deep learning techniques, including pre-trained models, are necessary to achieve satisfactory results. With the rise of complex neural network architectures, there's growing interest in leveraging these models for time series forecasting. Pretrained models, in particular, offer a starting point for developing sophisticated predictions without extensive training from scratch.

By comparing deep learning models, including a pre-trained model, with more traditional forecasting approaches, we aim to determine if the added complexity of these techniques is justified by a significant improvement in predictive accuracy. This insight will help inform future decisions for data science projects in similar contexts, ensuring that the most efficient and effective methods are employed.

# Data Summary

Our dataset, sourced from Kaggle's SF Air Traffic Passenger and Landings Statistics, provides detailed monthly records of air traffic data at San Francisco International Airport. The air-traffic-passenger-statistics.csv file, which we selected for our analysis, contains the following features from **Table 1**:

| Variable | Data Type | Definition |
| --- | --- | --- |
| Activity Period | Integer | The year and month at which passenger activity took place. |
| Operating Airline | String | The Airline name for the operator of aircraft with passenger activity. |
| Operating Airline IATA Code | String | International Air Transport Association (IATA) two-letter designation for the airline. |
| Published Airline | String | Airline name that issues the ticket and books revenue for passenger activity. |
| Published Airline IATA Code | String | International Air Transport Association (IATA) two-letter designation for the airline. |
| GEO Summary | String | Location within the US (“domestic”) or outside the US (“international”) without stops. |
| GEO Region | String | A more detailed breakdown of the GEO Summary field to designate the region in the world. |
| Activity Type Code | String | Boarding a flight (“enplanements”), getting off a flight (“deplanements”), and transiting to another location (“in-transit”). |
| Price Category Code | String | Indicates whether the published airline is a low-cost carrier or not. |
| Terminal | String | The airport terminal designations at SFO where passenger activity took place. |

**Table 1.** Definitions and table provided by the City of San Francisco in the Kaggle Dataset (City of San Francisco, n.d.).

Before proceeding to data aggregation, we addressed the issue of missing data. Two variables, Operating Airline IATA Code and Published Airline IATA Code, each had 63 missing values out of 18,885 entries. After reviewing the data, we found that the missing values were linked to companies like Swissport USA, Pacific Aviation, Trego Dugan Aviation, Servisair, and Boeing Company. These companies are involved in aviation services, including ground handling and manufacturing, rather than airline operations (*Trego Dugan Aviation*, n.d.). As a result, they do not have IATA codes, which are used exclusively for airline identification (*IATA Codes*, n.d.). To handle this, we filled the missing IATA code values with "N/A" to indicate that these entities do not operate as traditional airlines.

To prepare the data for analysis, we aggregated the features by month, which involved performing counts for categorical variables like GEO Summary and Operating Airline to ensure that the data reflected a monthly perspective. The Activity Period was converted into a Datetime stamp to accurately represent the time series data. Following this aggregation, our dataset was reduced from 18,885 entries to 156. **Figure 1** illustrates what the monthly data looks like over the entire dataset.

Additionally, a new feature named Season was engineered and categorically encoded. This feature was added based on the evidence from the seasonal decomposition graph (**Figure 2**), which highlighted both seasonal and trend components in the data. The graph underscored the importance of capturing seasonal patterns, prompting us to include the Season feature to represent different times of the year. By focusing on aggregating data monthly and incorporating this new feature, we were able to better capture overall trends, seasonal variations, and other factors influencing passenger counts at SFO, providing a comprehensive view necessary for accurate forecasting across our various prediction methods.

# Background on Chosen Experimental Methods

To investigate the necessity of deep learning models for accurately predicting monthly passenger counts, we explored both traditional and advanced approaches. Linear Regression was selected as one of the traditional methods to serve as a baseline model.

Linear Regression is a statistical technique that models the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the observed data (Kibet, 2023). In the context of time series forecasting, this method can incorporate lagged values of the time series as features to account for temporal dependencies (Gomede, 2024). This approach allows the model to leverage past observations to predict future values, providing a simple yet informative baseline for comparison.

While Linear Regression may not capture complex patterns and nuances in the data as effectively as more sophisticated models, it is valuable for understanding basic trends and relationships. For instance, it can reveal whether there is a general upward or downward trend in passenger counts over time. The model's simplicity also makes it easier to interpret and implement, and it serves as a useful starting point for evaluating the added value of more complex models. Proper data preprocessing, such as detecting and handling outliers and missing values, is critical to ensure the model's accuracy and reliability.

In addition to our application, Linear Regression has been successfully applied in various time series forecasting scenarios, such as predicting stock prices, weather conditions, and economic indicators (Thilakarathne, 2020). These examples demonstrate its versatility and utility as a foundational forecasting tool.

We utilized the statistical method SARIMA (Seasonal AutoRegressive Integrated Moving Average) for our project on predicting monthly passenger counts at San Francisco International Airport (SFO). SARIMA is an extension of the ARIMA model that incorporates seasonal differencing to handle periodic fluctuations, making it well-suited for time series datasets with regular seasonal variations (Artley, 2022). In this context, SARIMA was particularly appropriate due to the clear annual seasonal trends identified in the SFO data (Figure 2). These trends, influenced by holidays, vacation periods, and annual events, significantly impact passenger traffic patterns (Bouwer et al., 2024).

The SARIMA model captures these seasonal patterns by including terms for seasonal autoregression, differencing, and moving averages (Artley, 2022). This approach allows the model to represent the increased passenger traffic during peak travel times, such as summer vacations and winter holidays, as well as the decreased volumes during off-peak periods (Bouwer et al., 2024). By accounting for these fluctuations, SARIMA can provide more accurate forecasts that reflect the cyclical nature of the data.

To illustrate SARIMA's effectiveness in capturing seasonal variations, consider its application in forecasting cooling degree-days (CDD). In this study, SARIMA was employed to predict CDD, a measure that quantifies the demand for cooling based on temperature variations (Bilgili, 2023). The study showed how SARIMA effectively managed seasonal fluctuations and trends in CDD data, demonstrating its ability to model complex seasonal patterns. This application is relevant to our work because it parallels the challenge of forecasting seasonal patterns in air traffic data at SFO.

In addition to using SARIMA for handling seasonal variations, we also incorporated the Prophet model developed by Facebook. Prophet is designed to manage time series data with strong seasonal effects and missing data points, making it a valuable tool for our analysis (Khare, 2023). Unlike deep learning models that require extensive computational resources and training time, Prophet offers a more resource-efficient approach by leveraging its pre-trained capabilities and fine-tuning options.

Prophet works by decomposing time series data into trend, seasonality, and holiday components. This decomposition allows Prophet to handle irregularities in the data, such as missing values and outliers, with robust forecasting performance (Khare, 2023). For our dataset, which exhibits clear seasonal trends and occasional outliers, Prophet's flexibility and interpretability make it particularly suitable. It allows for the easy incorporation of seasonal patterns and holiday effects, which are crucial for accurate passenger count predictions.

To illustrate Prophet's effectiveness, consider its application in forecasting retail sales. In a study by Taylor and Letham (2018), Prophet was used to forecast sales data with strong seasonal effects and irregular patterns. The model successfully captured seasonal fluctuations and provided accurate forecasts despite the presence of missing data and outliers. This example demonstrates how Prophet can manage complex time series data, similar to how it will be applied to our passenger count data at SFO. By incorporating Prophet, we enhance our forecasting capability, ensuring reliable predictions even with data irregularities and seasonal trends.

In our analysis, we first utilize Linear Regression, SARIMA, and Prophet to establish a baseline for forecasting performance. These models help us understand the effectiveness of traditional and statistical approaches in capturing the key patterns in our data.

Following this, we will evaluate whether advanced deep learning models—specifically Long Short-Term Memory (LSTM) networks and Transformers—can offer further improvements. The goal is to determine if these models provide significant benefits over the baseline models and if their complexity is justified.

LSTM networks are a type of Recurrent Neural Network (RNN) designed to address the vanishing gradient problem that can occur in traditional RNNs (Or, 2020). They use memory cells and gating mechanisms to retain and update information over long sequences, which is crucial for capturing temporal dependencies and trends (Or, 2020). In forecasting, LSTMs can learn from sequences of historical data to predict future values, making them useful for identifying long-term patterns and trends in time series data. For instance, a study by Fischer and Krauss (2018) demonstrated the application of LSTMs to stock market forecasting. The researchers found that LSTMs could effectively capture complex temporal patterns and dependencies in financial data, which is similar to the patterns we might encounter in passenger counts at SFO. This ability to model long-term dependencies makes LSTMs a promising choice for improving forecast accuracy in our context.

Transformers, on the other hand, utilize self-attention mechanisms to process entire sequences of data simultaneously. This allows them to capture complex relationships and long-range dependencies more effectively than traditional RNNs, which process data sequentially (Lanza, 2023). By incorporating positional encoding, Transformers maintain the order of elements in the sequence, which is essential for time series data. An example of their application is seen in a study by Zhang et al. (2022), where Transformers were used for electricity demand forecasting. The study showed that Transformers could manage intricate patterns and seasonal effects in time series data, highlighting their potential for handling the seasonal and trend components in our passenger count data.

By integrating these advanced models, we aim to assess whether LSTMs and Transformers can provide significant enhancements over Linear Regression, SARIMA, and Prophet. The goal is to determine if the added complexity and resource demands of these models are justified for improving our forecasts of monthly passenger counts.

# Experimental Methods and Trainings

The first model implemented to compare against the performance of our two deep learning models was a Linear Regression model. We explored several variations, including a simple Linear Regression with past target data only and another that incorporated correlated variables, such as the frequency of operating airlines within a month. Certain airlines consistently held a higher percentage of the total passenger count (**Figure 3**). However, for conciseness, we will focus only on the best-performing version among the various models and methods attempted.

For the Linear Regression model, the best-performing version utilized not only past values of the target variable but also incorporated seasonal information. The dataset, consisting of 156 entries created during the EDA and Feature Engineering stage, was processed using a sequence function. This function used the data from the last six months (entries) to create the input features (X), which included the previous six months of passenger counts and the associated seasons for those months. The target variable (y) was the passenger count for the following month. This X and y were then split into training, validation, and test sets, with 90 entries for training, 30 for validation, and 30 for testing.

To train the Linear Regression model, we used the **LinearRegression** class from the **sklearn.linear\_model** library. The model was fitted to the training data by calling the **fit** method, which calculated the optimal coefficients for the linear equation by minimizing the mean squared error between the predicted and actual passenger counts after training, the model's performance was evaluated on the validation and test sets. The metrics used for evaluation were MSE and the coefficient of determination (R²), providing measures of the model's accuracy and goodness-of-fit, respectively.

Following the implementation of Linear Regression, we employed the SARIMA model for forecasting. SARIMA is well-suited for time series data with strong seasonal patterns, such as our dataset, which exhibits clear annual trends.

The SARIMA model was implemented using the **SARIMAX** class from the **statsmodels.tsa.statespace.sarimax module**. We configured the model with an *order* of (1, 1, 1) and a *seasonal\_order* of (1, 1, 1, 12). The order parameter includes one autoregressive term, one differencing term for stationarity, and one moving average term. The *seasonal\_order* parameter captures the seasonal components with a periodicity of 12 months, corresponding to the annual cycle we’ve seen in **Figure 2**.

For evaluation, the SARIMA model was trained on the training set and tested on the same set of unseen data used for Linear Regression, which consisted of the last 30 months of data. This consistency in test data across models ensured a fair comparison of performance metrics, such as MSE and R².

SARIMA used only the passenger count data as its input, focusing on modeling the time series itself. By capturing both short-term fluctuations and long-term seasonal effects, SARIMA provided accurate forecasts of monthly passenger counts at SFO, leveraging its ability to handle complex seasonal patterns effectively.

Following the evaluation of SARIMA, which effectively captured seasonal patterns and trends in the passenger count data, we turned to Prophet to further explore its capabilities in time series forecasting. Prophet, developed by Facebook, is designed to handle time series data with strong seasonal effects and missing data points (Seasonality, Holiday Effects, And Regressors | Prophet, n.d.).

Prophet was implemented using the prophet library, which requires a specific DataFrame format. We first created a DataFrame with columns ds for dates and y for the target variable, which in this case is the monthly passenger count. After preparing this DataFrame, we split the data into training and testing sets. For consistency, the test set consisted of the last 30 months of data, the same period used in previous models.

Typically, Prophet utilizes the entire dataset for training, but in our case, we chose to reserve the last 30 months for testing to evaluate how Prophet performs against unseen data. This approach allowed us to assess Prophet's forecasting accuracy and compare it with the results from SARIMA and Linear Regression.

Following the implementation of Prophet, which effectively decomposed the time series into trend and seasonal components, we explored the potential of deep learning models to further enhance forecasting accuracy. The first deep learning model we decided to use was the Transformer.

For the Transformer model, we adapted the approach from Jeff Heaton's implementation, "Transformer-Based Time Series with PyTorch (10.3)." Transformers are particularly adept at handling sequential data due to their self-attention mechanisms, which can capture long-range dependencies. However, they do not inherently recognize the sequence of the data. To address this, Heaton's model includes a positional encoding mechanism that uses sinusoidal functions to encode positional information, allowing the model to account for the order of inputs.

The model architecture was constructed with several layers, including an encoder that maps the input data to a higher-dimensional space, a positional encoding component that adds positional information, and a transformer encoder to process the input sequence. The final layer is a decoder that maps the transformed data back to the output space, predicting the next value in the sequence.

The data preparation for the transformer involved segmenting the time series into sequences of six months, with the corresponding passenger count for the following month as the target. This structured the problem as a sequence-to-value prediction task. The dataset was split into training, validation, and test sets, consisting of 90, 30, and 30 sequences, respectively. The test set, comprising the last 30 months, was used consistently across all models for evaluation to ensure comparable performance metrics.

The model training involved using a MSE loss function, the Adam optimizer, and a learning rate scheduler to adjust the learning rate based on validation loss improvements. The training process included early stopping to prevent overfitting, stopping the training when the validation loss did not improve for five consecutive epochs. Despite experimenting with modifications, such as removing dropout layers to potentially enhance performance, the original configuration provided the best results. The model's effectiveness was measured against the same unseen test data used for all other models, evaluating its ability to generalize and accurately forecast future passenger counts based on past trends.

After assessing the performance of the Transformer model, we turned our attention to the LSTM network, our final deep learning model. Given its strength in capturing long-term dependencies in sequential data, we aimed to determine if the LSTM could provide additional insights or improvements over the previously tested models, including Linear Regression, SARIMA, Prophet, and Transformers.

To explore the effectiveness of LSTM networks, we applied a model specifically designed to capture temporal dependencies in sequential data. Our dataset preparation for LSTM was consistent with previous models: the data was split into training, validation, and test sets with 90, 30, and 30 sequences, respectively, using the past 6 months of passenger counts to predict the subsequent month's count. The test set, consisting of the last 30 months of data, remained consistent across all models for comparative evaluation.

The LSTM model underwent several optimization trials to fine-tune its architecture and hyperparameters. We experimented with different configurations, including varying the number of LSTM layers, adjusting the learning rate, and altering the dropout settings. Specifically, we tested models with higher learning rates than the baseline, removed dropout layers, and combinations of these adjustments. The best-performing configuration featured an LSTM model with the original learning rate of 0.01 and no dropout layers. We hypothesize that the relatively small size of the training dataset, comprising only 90 entries, made dropout less effective in preventing overfitting, as it can potentially hinder learning when the data size is limited.

The final model architecture consisted of two LSTM layers, with the first layer having 5 units and the second layer 3 units. These layers were followed by a Dense layer with a single output unit to predict the next value in the sequence. The model was compiled using the Adam optimizer with the original learning rate of 0.01 and theMSE) loss function. To prevent overfitting, we employed early stopping during training, monitoring the validation loss and stopping the training process when it did not improve for three consecutive epochs. This strategy allowed us to determine the optimal number of epochs dynamically, ensuring that the model did not overfit the training data.

The LSTM model's performance was evaluated on the same unseen test data as the previous models. This consistent approach ensured that we could fairly compare the predictive accuracy and generalization capabilities of the LSTM model against Linear Regression, SARIMA, Prophet, and Transformer models.

# Results and Conclusions

In evaluating the performance of various models for forecasting monthly passenger counts at SFO, we identified Linear Regression with seasonal components as the top performer. As shown in **Figure 4**, this model achieved the lowest MSE and highest R² on the test data, indicating its robust performance in predicting future passenger counts.

Linear Regression with Seasonal Components demonstrated strong performance across all metrics, with the lowest MSE and highest R² among the models. The model's effective capture of seasonal patterns contributed to its high accuracy. **Figure 5** illustrates the actual versus predicted values for this model, highlighting its precision in predicting the unseen test data.

Prophet, a pretrained model designed to handle time series data with strong seasonal effects and missing values, performed well but did not surpass Linear Regression. With an R² of 0.881 on the test data, Prophet effectively managed the seasonal patterns and outliers but was less accurate compared to Linear Regression. **Figure 6** shows the predictions made by Prophet alongside the actual values, providing insight into its forecasting capabilities.

SARIMA, showed moderate performance. It achieved an R² of 0.813 on the test set, indicating that while it captured the underlying trend and seasonality, it was less effective compared to the top models. **Figure 7** depicts the SARIMA model’s predictions, highlighting its alignment with actual values but also showing areas where it diverges.

The Transformer model, adapted from Jeff Heaton’s approach, was the least effective. Despite leveraging positional encoding to handle sequential data, it performed poorly with an R²

of -1.288 on the test data. **Figure 8** illustrates the Transformer’s predictions, revealing significant deviations from actual values and highlighting its struggles with the small dataset and univariate input.

LSTM, our final deep learning model, demonstrated limited effectiveness with an R² of 0.401. This model showed some potential but did not outperform the simpler models. **Figure 9** presents the LSTM’s predictions, indicating that despite various optimizations, the model did not achieve high accuracy.

The analysis of these models suggests that while deep learning approaches like Transformer and LSTM offer sophisticated methods for time series forecasting, they may not always outperform simpler models like Linear Regression with seasonal data included, particularly when the dataset is small and univariate. Deep learning models often require larger datasets and more diverse features to uncover complex patterns effectively.

In future work, expanding the dataset, exploring additional features, and experimenting with more advanced architectures could improve the performance of deep learning models. Additionally, incorporating multiple types of data and refining the training process may yield better results.

# 

# References

Artley, B. (2022, April 26). *Time Series Forecasting with ARIMA , SARIMA and SARIMAX*. Towards Data Science. Retrieved July 29, 2024, from https://towardsdatascience.com/time-series-forecasting-with-arima-sarima-and-sarimax-ee61099e78f6

Bilgili, M. (2023, July 31). Time series forecasting on cooling degree-days (CDD) using SARIMA model. *Natural Hazards*, *118*, 2569–2592. https://doi.org/10.1007/s11069-023-06109-4

Bouwer, J., Hausmann, L., Lind, N., Verstreken, C., & Xanthopoulos, S. (2024, January 8). *How airlines can handle seasonal travel demand*. McKinsey. Retrieved July 29, 2024, from https://www.mckinsey.com/industries/travel-logistics-and-infrastructure/our-insights/how-airlines-can-handle-busier-summers-and-comparatively-quiet-winters

City of San Francisco. (n.d.). *San Francisco International Airport (SFO) - Air Traffic Passenger and Landings Statistics*, Version 68. Retrieved August 11, 2024, from https://www.kaggle.com/datasets/san-francisco/sf-air-traffic-passenger-and-landings-statistics

Dolphin, R. (2020, October 21). *LSTM Networks | A Detailed Explanation*. Towards Data Science. Retrieved July 29, 2024, from https://towardsdatascience.com/lstm-networks-a-detailed-explanation-8fae6aefc7f9

Fischer, T., & Krauss, C. (2018, October 16). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*(2), 654-669. https://doi.org/10.1016/j.ejor.2017.11.054

Gomede, E. (2024, March 19). *Analyzing the Impact of Lagged Features in Time Series Forecasting: A Linear Regression Approach*. Cubed. Retrieved July 29, 2024, from https://blog.cubed.run/analyzing-the-impact-of-lagged-features-in-time-series-forecasting-a-linear-regression-approach-730aaa99dfd6

Grammarly. (2024). Grammarly. https://www.grammarly.com

Heaton, J. (2023, November 1). *Transformer-Based Time Series with PyTorch (10.3)*. YouTube. Retrieved July 31, 2024, from https://www.youtube.com/watch?v=NGzQpphf\_Vc

*IATA Codes*. (n.d.). IATA. Retrieved August 9, 2024, from https://www.iata.org/en/services/codes/

Kamath, P. (2024, February 6). *Future of Air Travel Forecast for Aviation Business Travel*. ITILITE's. Retrieved July 29, 2024, from https://www.itilite.com/blog/air-travel-forecast/

Khare, P. (2023, May 13). *Understanding FB Prophet: A Time Series Forecasting Algorithm*. Medium. Retrieved July 29, 2024, from https://medium.com/illumination/understanding-fb-prophet-a-time-series-forecasting-algorithm-c998bc52ca10

Kibet, M. (2023, April 21). *A Guide to Regression Analysis with Time Series Data*. InfluxData. Retrieved July 29, 2024, from https://www.influxdata.com/blog/guide-regression-analysis-time-series-data/

Lanza, E. (2023, August 2). *How to Apply Transformers to Time Series Models | by Intel | Intel Tech*. Medium. Retrieved July 29, 2024, from https://medium.com/intel-tech/how-to-apply-transformers-to-time-series-models-spacetimeformer-e452f2825d2e

Or, B. (2020, October 10). *The Exploding and Vanishing Gradients Problem in Time Series*. Medium. Retrieved July 29, 2024, from https://medium.com/metaor-artificial-intelligence/the-exploding-and-vanishing-gradients-problem-in-time-series-6b87d558d22

*Seasonality, Holiday Effects, And Regressors | Prophet*. (n.d.). Meta Open Source. Retrieved July 30, 2024, from https://facebook.github.io/prophet/docs/seasonality,\_holiday\_effects,\_and\_regressors.html

Taylor, S. J., & Letham, B. (2018). Forecasting at Scale. *THE AMERICAN STATISTICIAN*, *72*(1), 37-45.

Thilakarathne, A. L. (2020, October 9). *Creating a Model for Weather Forecasting Using Linear Regression*. Medium. Retrieved July 29, 2024, from https://medium.com/swlh/creating-a-model-for-weather-forecasting-using-linear-regression-b18c1590e8d7

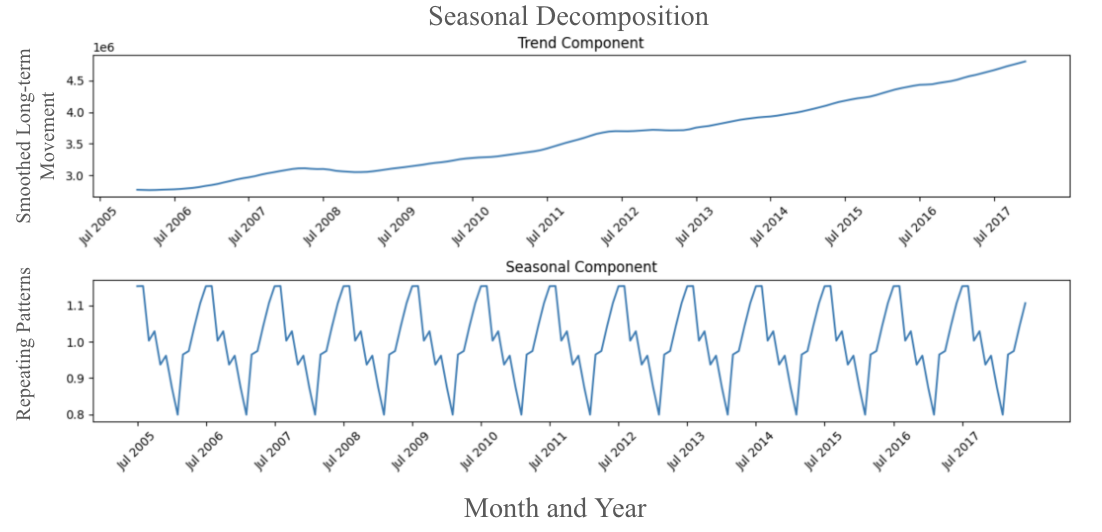
*Trego Dugan Aviation*. (n.d.). Trego Dugan Aviation - Home. Retrieved August 9, 2024, from https://www.trego-dugan.com/

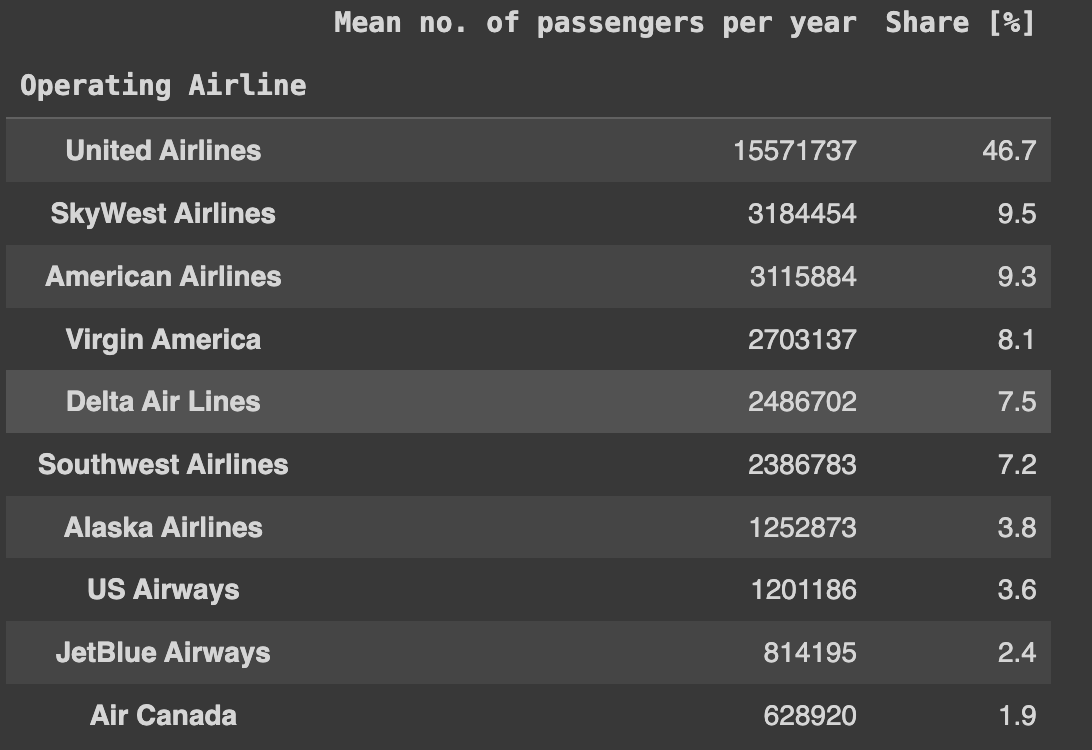
Zhang, G., Wei, C., Jing, C., & Wang, Y. (2022, August 18). Short-Term Electrical Load Forecasting Based on Time Augmented Transformer. *International Journal of Computational Intelligence Systems*, *15*(67). https://link.springer.com/article/10.1007/s44196-022-00128-y

# 

# Visualizations

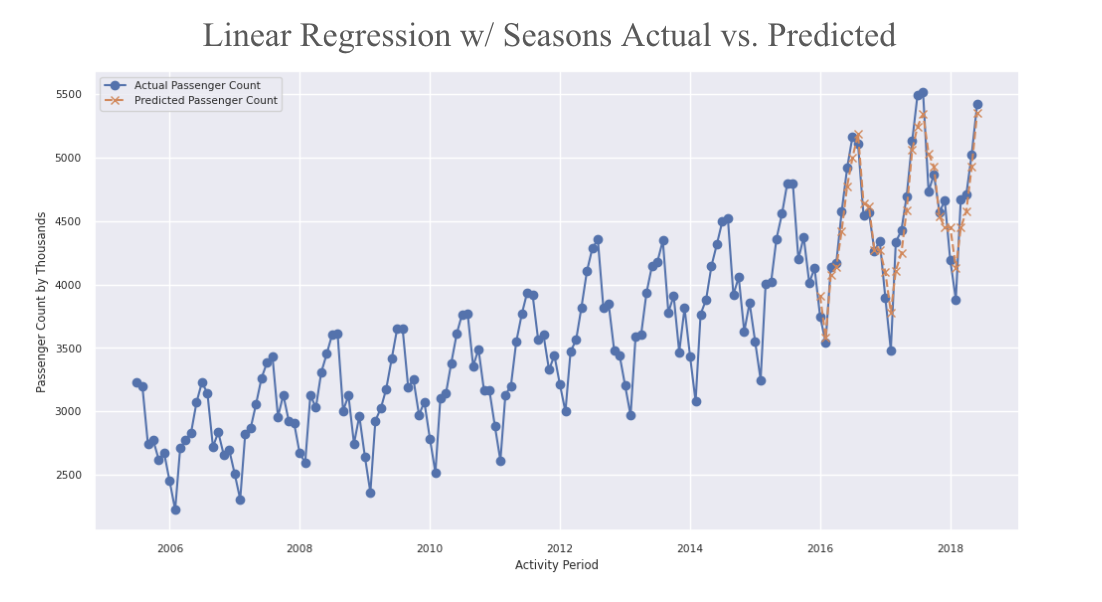
**Figure 1.** Monthly passenger count at San Francisco International Airport (SFO) from July 2005 to June 2018, showing a steady upward trend with seasonal fluctuations.

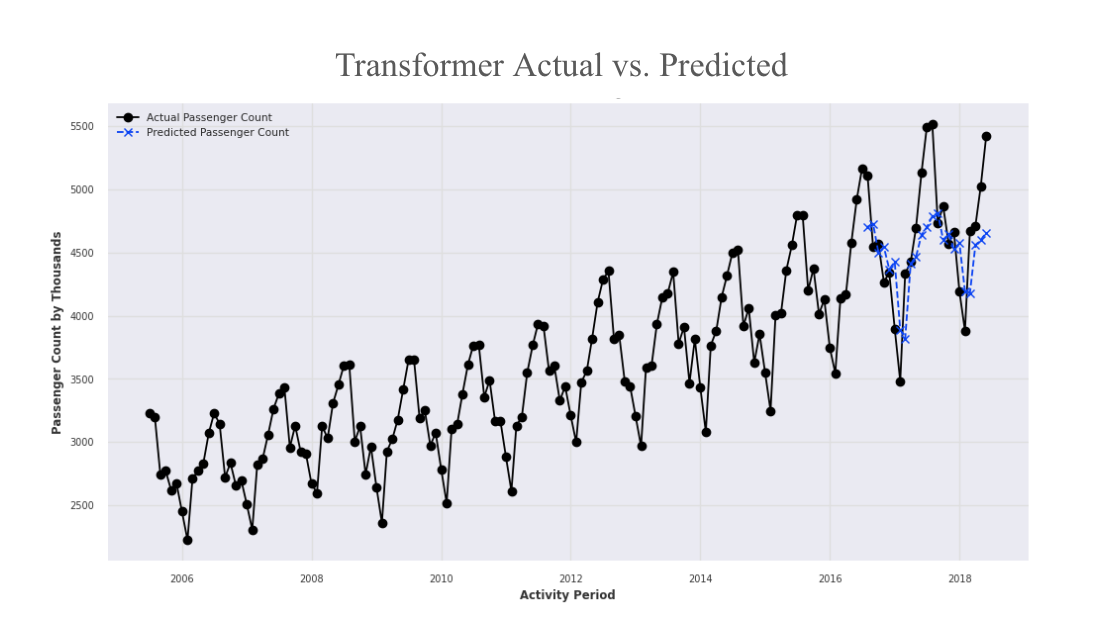
**Figure 2.** Seasonal decomposition of the monthly passenger count at San Francisco International Airport (SFO) from July 2005 to June 2018. The figure shows the trend component (top), indicating a steady long-term increase, and the seasonal component (bottom), illustrating regular annual fluctuations.

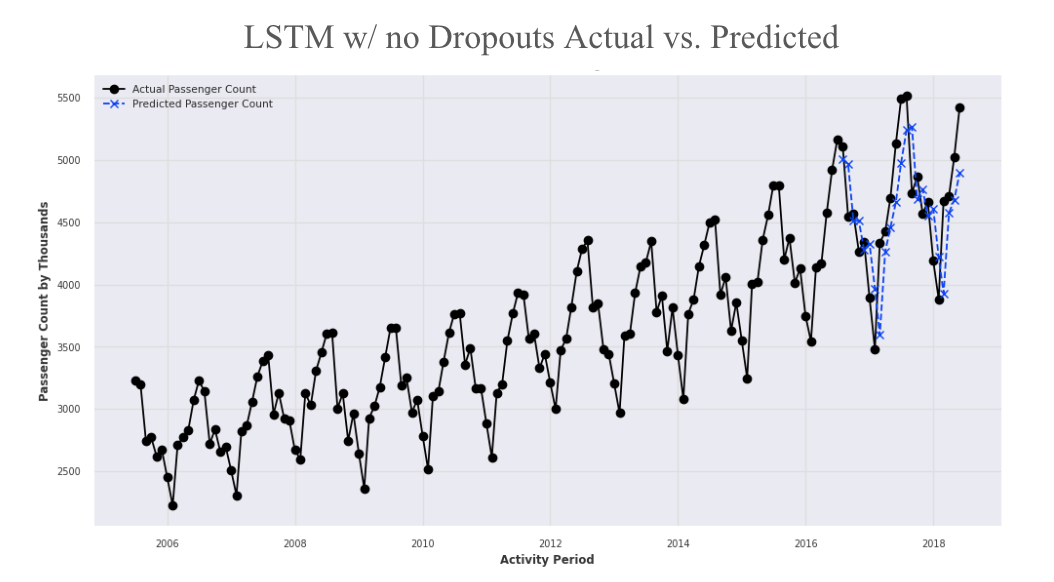
**Figure 3.** Average annual passenger count and market share by operating airline at San Francisco International Airport (SFO), with United Airlines holding the largest share at 46.7%.



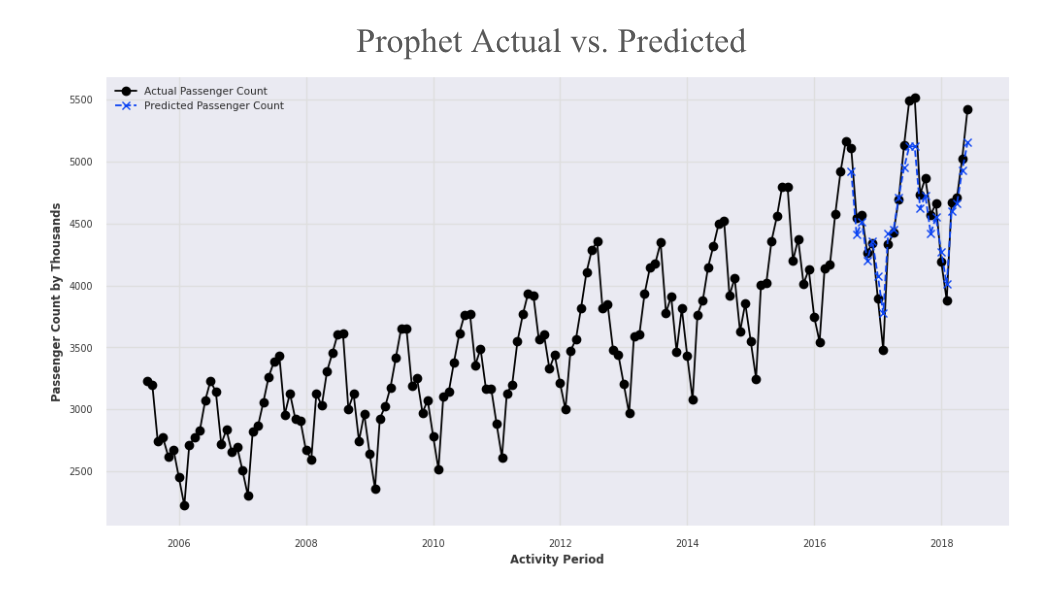
**Figure 4.** Performance comparison of the best-performing hyperparameters for various models in predicting monthly passenger counts at San Francisco International Airport (SFO). Metrics include Mean Squared Error (MSE) and R-squared (R²) for training, validation, and test datasets.

**Figure 5.** Comparison of Actual vs. Predicted Passenger Count at SFO Using Linear Regression Model Incorporating Seasonal Features. The model predicts passenger count one month ahead, showing close alignment between actual and predicted values over time with an ability to account for peak periods.

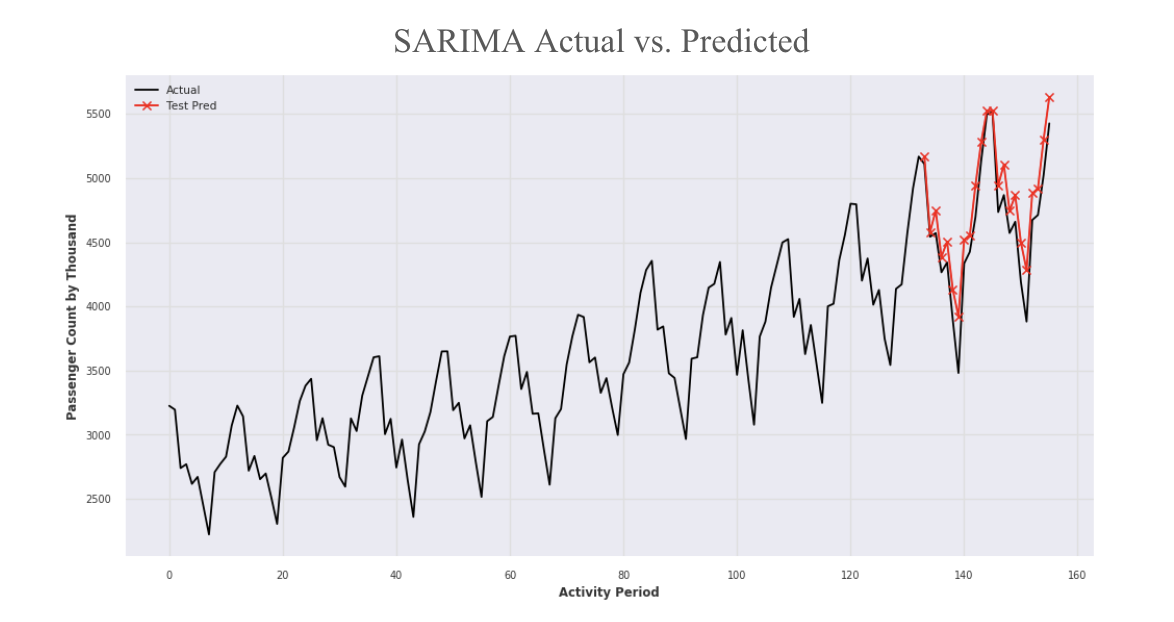
**Figure 6.** Comparison of Actual vs. Predicted Passenger Count at SFO Using Transformer Model with Dropout Layers on Unseen Data. The model's predictions (in blue) poorly match the actual passenger counts, with significant deviations at peak periods, indicating challenges in capturing seasonal trends and overall patterns.

**Figure 7.** Comparison of Actual vs. Predicted Passenger Count at SFO Using LSTM Model without Dropout Layers on Unseen Data. The predictions (in blue) show a limited alignment with actual passenger counts, particularly failing to accurately capture peak values, indicating challenges in the model's ability to generalize and predict future trends accurately.

**Figure 8.** Comparison of Actual vs. Predicted Passenger Count at SFO Using the Pre-Trained Prophet Model on Unseen Data. The predictions (in blue) show some alignment with actual passenger counts, but particularly failing to accurately capture peak values, indicating challenges in the model's ability to generalize and predict future trends accurately.

****

**Figure 9.** Comparison of Actual vs. Predicted Passenger Count at SFO Using the SARIMA Method on Unseen Data. The predictions (in red) show few alignment with actual passenger counts, particularly failing to accurately capture peak values as it tends to overestimate.

****