



EPiC Series in Computing

Volume 91, 2023, Pages 146–154

Proceedings of 38th International Conference on Computers and Their Applications



# Using Machine Learning to Predict Airport Passenger Throughput

Samuel Yi and Jiang Guo\*

California State University, Los Angeles, USA

syi13@calstatela.edu, jguo@calstatela.edu

## Abstract

The American commercial airline industry is a crucial part of United States infrastructure and is so large and widespread that it affects all of its citizens in one way or another. There are many moving pieces involved in this industry, but we believe that we can make a significant impact when it comes to forecasting future passenger throughput. We look to utilize machine learning to create a prediction model which can eventually be used by the Department of Homeland Security to improve security and overall customer experience at airport terminals. The results of this study show that a polynomial regression model can provide utility as well as predictions with an acceptable margin of error.

## 1 Introduction

The commercial aviation industry is a core part of the American economy having over 1.057 billion annual passengers at its peak in 2019 [1]. One of the many considerations that come with this massive number of passengers is the need for reliable passenger throughput forecasting. Forecasting in this case can aid not only with revenue but also in aspects of security, staffing, and overall customer experience. We seek to utilize machine learning to develop a polynomial regression model that is trained on readily available past data to predict future passenger throughput.

### 1.1 Purpose

Before diving into the data, problem, and solution, we must first explain why we want to predict passenger throughput. The Department of Homeland Security (DHS) and the Transportation Security Administration (TSA) provide security at American airports through various means. We focused on one of these, which are the TSA checkpoints found at terminals. At these terminals, TSA Agents provide “screening procedures [which are] intended to prevent prohibited items and other threats to transportation security from entering the sterile area of the airport and are developed in response to

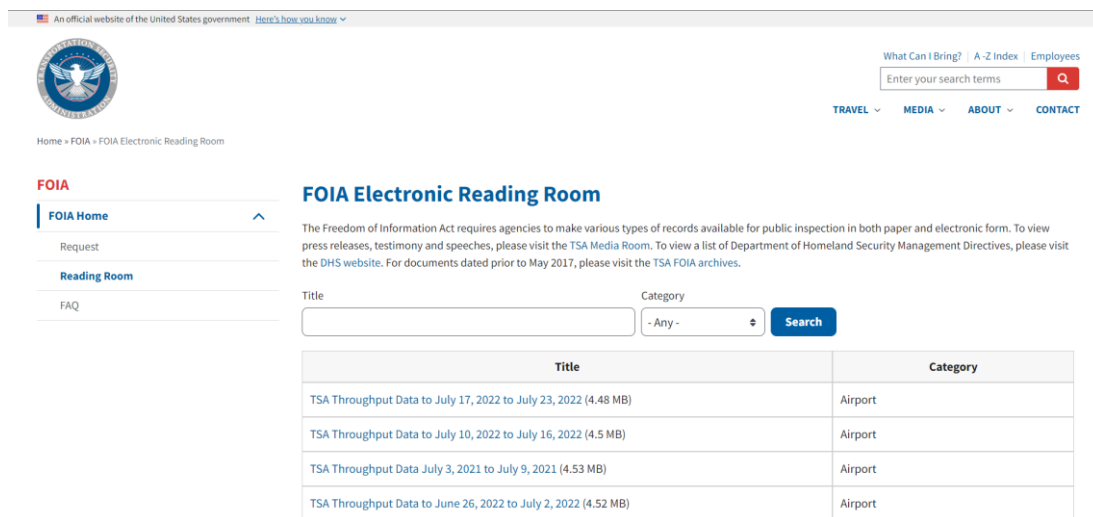
---

\* Corresponding author: Jiang Guo, jguo@calstatela.edu

information on threats to transportation security” [2]. These checkpoints are manned by varying amounts of TSA Agents throughout the day, depending on scheduling. We believe that integrating a polynomial regression model alongside existing tools could help the DHS and TSA to properly employ and schedule the necessary agents to accommodate the fluctuations of passengers throughout the days.

## 2 Dataset

The TSA collects data from these checkpoints and provides it for viewing on their website under the Freedom of Information Act (FOIA) as can be seen in Figure 1. There are various forms of information provided, but we focus on the weekly reporting of passenger throughput available in PDF format [3].



The screenshot shows the TSA's FOIA Electronic Reading Room. The page includes a search bar, a navigation menu with options like TRAVEL, MEDIA, ABOUT, and CONTACT, and a section for FOIA requests. A table lists the following requests:

Title	Category
TSA Throughput Data to July 17, 2022 to July 23, 2022 (4.48 MB)	Airport
TSA Throughput Data to July 10, 2022 to July 16, 2022 (4.5 MB)	Airport
TSA Throughput Data July 3, 2021 to July 9, 2021 (4.53 MB)	Airport
TSA Throughput Data to June 26, 2022 to July 2, 2022 (4.52 MB)	Airport

Figure 1: TSA's FOIA Reading Room

The data provided gives passenger throughputs at checkpoints across airport terminals in chronological order throughout the week. For this research, we used data from LAX's "Terminal 1 – Passenger" from 2015 – 2019.

### 2.1 Data Engineering

The data isn't usable directly in its original format, an example of which is shown below.

TSA Total Throughput  
7/25/2022

Date	Hour of Day	Airport	City	State	Checkpoint	Total Pax + KCM PAX					
7/17/2022	00:00	ANC	Ted Stevens Anchorage International	Anchorage	AK	South Checkpoint	326				
		ATL	Hartsfield Atlanta International	Atlanta	GA	Main Checkpoint	26				
		BQN	Rafael Hernandez	Aguadilla	PR	Rafael Hernandez Air	41				
		CLT	Charlotte/Douglas International	Charlotte	NC	B Checkpoint	2				
		DCA	Washington Reagan National	Arlington	VA	Concourse A	10				
						North Checkpoint	0				
						South Checkpoint	0				
		DEN	Denver International	Denver	CO	A Bridge	0				
						North	0				
						South	91				
		DTW	Detroit Metro Wayne County	Detroit	MI	Blue-2	26				
						Red 3	10				
		ELP	El Paso International	El Paso	TX	Consolidated Checkpoint	0				
		EWR	Newark International	Newark	NJ	CKPT-B3	1				
						CKPT-C1	31				
		FAI	Fairbanks International	Fairbanks	AK	ASAA-FAI	107				
		FAT	Fresno Air Terminal	Fresno	CA	FAT 01	84				
		GUM	Antonio B. Won Pat International	Tamuning	GU	GUM01	2				
		IAD	Washington-Dulles International	Washington, DC	VA	Passenger Screening Area: PreCheck	251				
		IND	Indianapolis International	Indianapolis	IN	Checkpoint A	8				
						Terminal 2	2				
						Terminal 4 Main	280				
						Terminal 5	62				
						Terminal 7	0				
		JFK	John F. Kennedy International	Jamaica	NY	Terminal 8	50				
						JNU	Juneau International	Juneau	AK	JNU-01	75
						LAS	Harry Reid International Airport	Las Vegas	NV	Term 1 - AB	2
		Term 1 - D	62								
		LAX	Los Angeles International	Los Angeles	CA	Terminal 3 - E Upper	234				
						TBIT Main Checkpoint	127				
						Terminal 4 - Passenger	1				
		LGA	LaGuardia	Flushing	NY	Terminal 5 - Passenger	15				
						Terminal 7 - Passenger	9				
		LGB	Long Beach Daugherty Field	Long Beach	CA	TB-CHK	17				
						TC CHK West	6				
		LGB	Long Beach Daugherty Field	Long Beach	CA	Checkpoint	0				
		MCI	Kansas City International	Kansas City	MO	MCI 37-45	0				
		MCO	Orlando International	Orlando	FL	EAST Checkpoint	0				
						WEST Checkpoint	0				
		MDW	Chicago Midway	Chicago	IL	Main Checkpoint	1				
MEM	Memphis International	Memphis	TN	Int'l Arrivals	0						

Figure 2: TSA FOIA PDF sample

Some data engineering is required. To begin, we must first convert the PDFs into CSV format using a converter tool as working with CSVs is much easier. After that, we must remove redundant information. The data is organized by date, hour, airport, city, state, and terminal along with throughputs. Essentially, all we need are the dates, hours, checkpoints, and throughputs. From the dates, we then extrapolate the days of the week and the week numbers to have as additional features when training our machine learning model. Finally, we separated and aggregated the data by checkpoint.

## 3 Polynomial Regression Model

The model we are proposing to use is a type of supervised learning model called a polynomial regression, which is essentially a regression model with a higher order or degree. We use this type of regression model because the output can't be properly captured with a straight line. Passenger throughput has peaks and valleys throughout the day so we need a model that can follow these trends which almost look like a sine-like function.

### 3.1 Overfitting

The concept of overfitting must be carefully considered when dealing with machine learning. This is especially the case with polynomial regression because if the degree is too high, the model becomes "too trained" on the sample data which inevitably leads to inaccurate predictions. However, we are looking to capture hourly, daily, and weekly trends in this case. Because of this, we do want overfitting to a degree.

### 3.2 One Hot Encoding

There is some preprocessing to be done before we can implement our polynomial regression model. First, we use one hot encoding to convert the features from categorical to discrete as most machine learning libraries don't play well (if at all) with categorical features. We use the Python library "sklearn" to access "OneHotEncoder" [4]. A sample of the code to extract out "dummy" features from a feature column is shown in the figure below.

```
from sklearn.preprocessing import OneHotEncoder

# get dummies from Hour
hour_cols = pd.get_dummies(df['Hour'])
hour_cols
```

**Figure 3:** Python code using OneHotEncoder

With one hot encoding, we convert each unique feature type into its own type, expanding the number of features from 4 to 85. An example of a datapoint expanded through one hot encoding would be: throughput for the Sunday of the 2<sup>nd</sup> week at 12:00 becoming a datapoint with the feature columns Sunday, Week 2, and 12:00 having a 1 (true) while the rest have 0s (false). See Figure 4 for an expanded DataFrame.

	Datetime	Throughput	0:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	...	51	52	53	Friday	Monday	Saturday	Sunday	Thursday	Tuesday	Wednesday		
0	1/1/2015 4:00	47	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0	0	
1	1/1/2015 5:00	189	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0	0	
2	1/1/2015 6:00	206	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0	0	
3	1/1/2015 7:00	547	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0	0	
4	1/1/2015 8:00	592	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	1	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
35448	12/31/2019 17:00	430	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1	0	
35449	12/31/2019 18:00	204	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1	0	
35450	12/31/2019 19:00	128	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1	0	
35451	12/31/2019 20:00	81	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1	0	
35452	12/31/2019 21:00	9	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1	0	

35453 rows × 86 columns

**Figure 4:** DataFrame after using OneHotEncoder

### 3.3 Principal Component Analysis

Next, we look to reduce the number of features with dimensionality reduction. With the expanded number of features, the model's complexity and runtime increase. Complexity is crucial with machine learning due to the large amounts of data being used as input. To decrease the (already high) complexity of a polynomial regression model, we use a form of dimensionality reduction called Principal Component Analysis (PCA). PCA is ideal in this situation because it allows us to capture and retain information from the original 85 features while reducing the overall number. Again, we use sklearn to access PCA [5] to find the optimal number of features and apply it to our data.

First, we determine the optimal number of features to reduce to. We can do this by graphing the relationship between the cumulative explained variance and the number of components (features) as seen in Figure 5 and Figure 6.

```

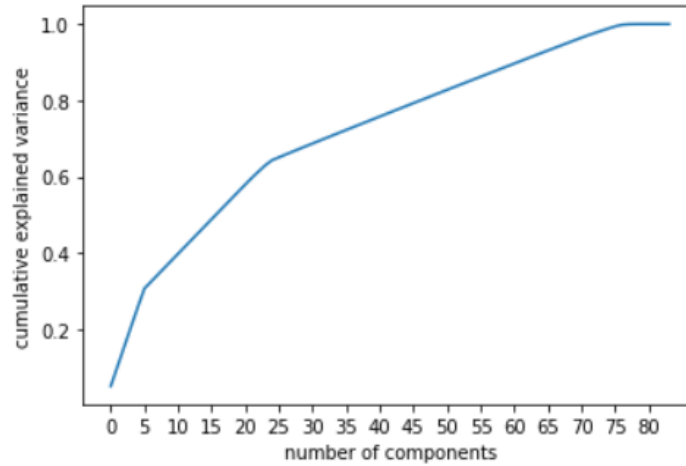
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

x_ticks = [0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80]

pca = PCA().fit(X_categorical)
plt.plot(pca.explained_variance_ratio_, linewidth=2)
plt.xlabel('number of components')
plt.ylabel('cumulative explained variance')
plt.xticks(x_ticks, x_ticks)

```

**Figure 5:** Python code to show the PCA explained variance ratio



**Figure 6:** Graph showing the PCA explained variance ratio

As seen above, there are a couple approximate points of interest: 6, 26, and 77. At each of these points, the rate of increase of cumulative explained variance decreases. Obviously, capturing as much explained variance as possible is ideal but it comes at the cost of added complexity. After some manual benchmarking, we determined that 25 was the optimal number of components. Once the 25 new components are created with PCA as demonstrated below, they can then be used with the polynomial regression model.

```
pca = PCA(n_components=25)
principal_components = pca.fit_transform(X_categorical)
columns = ['pc_1', 'pc_2', 'pc_3', 'pc_4', 'pc_5', 'pc_6', 'pc_7', 'pc_8', 'pc_9', 'pc_10', 'pc_11', 'pc_12', 'pc_13', 'pc_14']
X_principal = pd.DataFrame(data = principal_components, index=df['Datetime'], columns=columns)
```

**Figure 7:** Python code using PCA

### 3.4 Implementation

Implementation of the polynomial regression model is relatively simple due to sklearn. We simply need to choose the number of degrees and then use the classes `PolynomialFeatures` [6] and `LinearRegression` [7]. To determine the optimal number of degrees, we again did manual benchmarking which consisted of repeatedly increasing the degree by one and running the model. There was a decrease in accuracy moving from 2 to 3, so we decided on using 2 degrees. See Figure 8 for the implementation of the polynomial regression model.

```
from sklearn.preprocessing import PolynomialFeatures

poly = PolynomialFeatures(degree=2)
X_poly_train = poly.fit_transform(X_train)
X_poly_test = poly.transform(X_test)

from sklearn.linear_model import LinearRegression

linreg = LinearRegression()
linreg.fit(X_poly_train, y_train)
y_pred = linreg.predict(X_poly_test)
```

**Figure 8:** Python code using PolynomialFeatures and LinearRegression

### 3.5 Evaluation

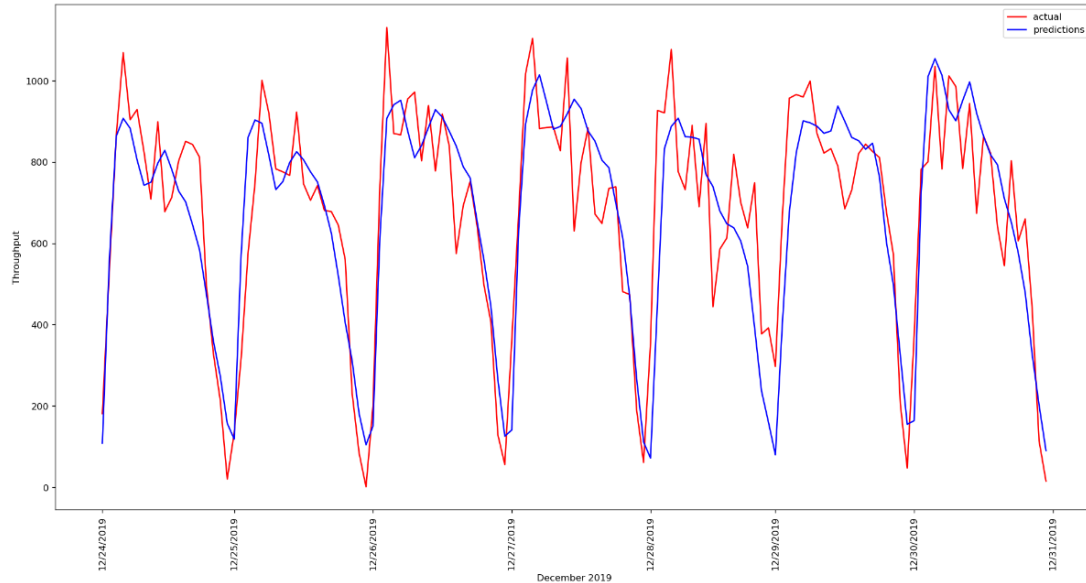
To measure the accuracy of our model, we use Root Mean Squared Error (RMSE). To provide context for this metric, we divide it by means of the actual throughputs (of 2019) to obtain Relative Root Mean Squared Error (RRMSE) percentages. Again, we used sklearn to access “mean\_squared\_error” [8] to calculate the RMSE. See the figure below for an example of that code.

```
from sklearn import metrics

mse = metrics.mean_squared_error(df2['Throughput'], df2['Prediction'])
rmse = np.sqrt(mse)
```

**Figure 9:** Python code using mean\_squared\_error

With this model, we achieve an overall RMSE of 146.143 and an RRMSE of 21.225% for 2019’s data. To highlight the utility of this model, see the figure below which compares the actual throughputs and predictions from 12/24/2019 – 12/31/2019 with an RMSE of 138.367 and RRMSE of 20.360%.



**Figure 10:** Graph comparing actual vs. predicted values from 12/24/2019 - 12/31/2019

## 4 Conclusion

Although this model doesn't capture all of the spikes that real-world data comes with, it does capture overall passenger throughput trends. The main strength of using a polynomial regression model comes from the ability to predict specific dates or ranges that tend to be harder to forecast due to shifting dates year-to-year. A prime example is shown in above in Figure 10. We feel that using a polynomial regression model to predict future passenger throughput alongside existing DHS and TSA tools can greatly benefit everyone, from those providing security at terminal checkpoints to everyday passengers. This research can be extended by integrating this model into an internal facing application for use by the DHS and TSA.

## Acknowledgement

This research was performed under an appointment to the U.S. Department of Homeland Security (DHS) Science & Technology (S&T) Directorate Office of University Programs Summer Research Team Program for Minority Serving Institutions, administered by the Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy (DOE) and DHS. ORISE is managed by ORAU under DOE contract number DE-SC0014664. All opinions expressed in this paper are the author's and do not necessarily reflect the policies and views of DHS, DOE or ORAU/ORISE.



## References

- [1] Federal Aviation Association, "Air Traffic by the Numbers," [Online]. Available: [https://www.faa.gov/air\\_traffic/by\\_the\\_numbers/media/Air\\_Traffic\\_by\\_the\\_Numbers\\_2022.pdf](https://www.faa.gov/air_traffic/by_the_numbers/media/Air_Traffic_by_the_Numbers_2022.pdf).
- [2] Transportation Security Administration, "Security Screening," [Online]. Available: <https://www.tsa.gov/travel/security-screening>.
- [3] Transportation Security Administration, "FOIA Electronic Reading Room," [Online]. Available: <https://www.tsa.gov/foia/readingroom>.
- [4] scikit learn, "sklearn.preprocessing.OneHotEncoder," [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html>.
- [5] scikit learn, "sklearn.decomposition.PCA," [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>.
- [6] scikit learn, "sklearn.preprocessing.PolynomialFeatures," [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html>.
- [7] scikit learn, "sklearn.linear\_model.LinearRegression," [Online]. Available: [https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LinearRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html).
- [8] scikit learn, "sklearn.metrics.mean\_squared\_error," [Online]. Available: [https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean\\_squared\\_error.html](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html).