



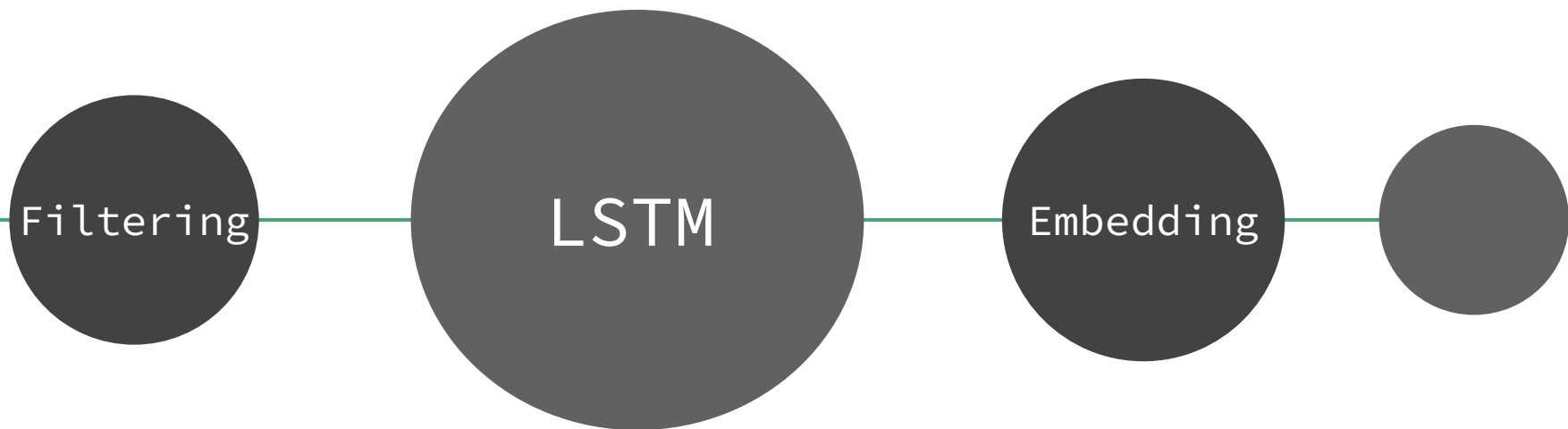
# Kagglistaz Final Presentation

Aritra Das, Ethan Lee, Luke Taylor, Asif Mahdin  
CSE 151B Spring 2023

# Summary

1. Major performance improvements of our model can be attributed to data cleaning and preprocessing
2. Our best performing model uses a LSTM Classifier to predict the travel time of the trip
3. By framing the problem in a new light major performance improvements can be made

Key Words



# Methodology

---

# Data Processing

Initial data size: 1,710,670 by 9 features

Prefiltered data size: 1,656,475

Filtered data size: 122,756

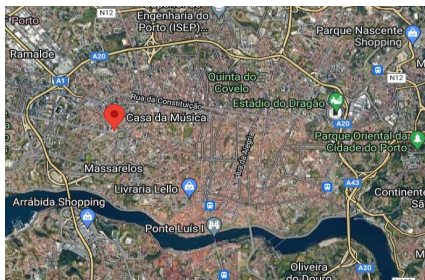
**Alterations:** converted polyline to trip length in seconds, convert timestamp into date/time

## Dropped:

- MISSING DATA
- Empty polylines
- Day type column
- Outliers

## Added:

- Latitude and longitude of each origin stand
- Tags for each origin stand (ex: hospitals, universities, stadiums)



## Included (to match test data):

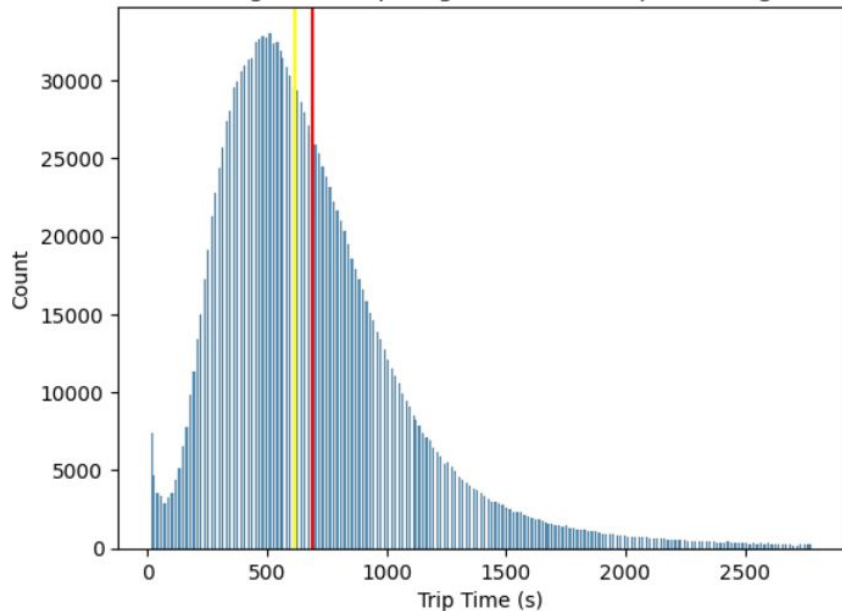
- Trips ending around 2:30PM, 5:45PM, 6:00PM, 4:00AM, or 8:30AM or starting up to 30 minutes before
- Trips with taxi ids matching those seen in the public test set
- Trips starting August or later

```
[ '11:39' '11:41' '12:17' '12:47' '13:18' '13:23' '14:11' '14:13' '14:16'  
'14:17' '14:18' '14:22' '14:23' '14:24' '14:25' '14:26' '14:27' '14:28'  
'14:29' '14:6' '15:12' '16:12' '16:14' '16:2' '16:23' '16:45' '16:48'  
'17:16' '17:17' '17:18' '17:19' '17:20' '17:21' '17:23'  
'17:24' '17:25' '17:27' '17:30' '17:31' '17:32' '17:33' '17:34' '17:35'  
'17:36' '17:37' '17:38' '17:39' '17:4' '17:40' '17:41' '17:42' '17:43'  
'17:44' '17:45' '17:46' '17:47' '17:48' '17:49' '17:50' '17:52' '17:53'  
'17:54' '17:55' '17:56' '17:57' '17:58' '17:59' '17:7' '2:40' '3:38'  
'3:39' '3:42' '3:43' '3:44' '3:47' '3:48' '3:49' '3:50' '3:51' '3:52'  
'3:53' '3:54' '3:55' '3:56' '3:57' '3:58' '3:59' '6:44' '7:2' '7:31'  
'7:46' '7:50' '7:53' '8:1' '8:10' '8:11' '8:12' '8:13' '8:14' '8:15'  
'8:16' '8:17' '8:18' '8:19' '8:20' '8:21' '8:22' '8:23' '8:24' '8:25'  
'8:26' '8:27' '8:29' '8:5' '8:7' '8:9' ]
```

```
mean: 687.1165698244766
std: 397.7629894750145
(1656475, 17)
```

```
Text(0.5, 0, 'Trip Time (s)')
```

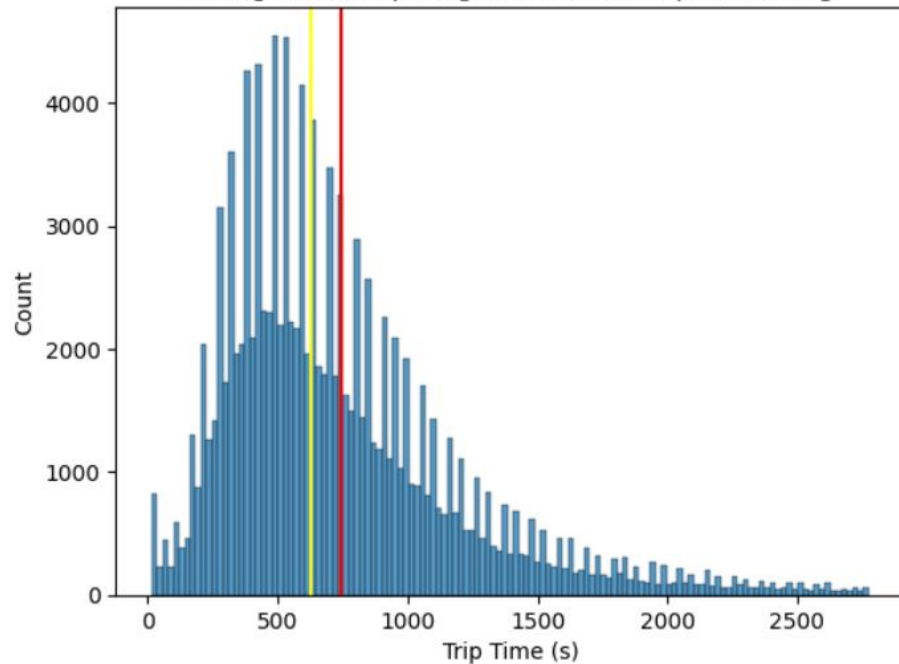
histogram of trip length in seconds : prefiltering



```
mean: 744.9569471146013
std: 461.8033643640038
(122756, 25)
```

```
Text(0.5, 0, 'Trip Time (s)')
```

histogram of trip length in seconds : postfiltering








# Deep Learning Model

Our **initial** deep learning models we experimented with were **Multi-Layered Perceptron(MLP)**, however, with this approach no matter how much we trained this model the **public loss would not break the 750 public loss**

After using MLP, our group decided to try Natural Language Techniques. We “vocabulary” the data to unique tokens to a **“vocab size” of 213**.

Most of the “vocabulary” consisted of the **prediction categories**

**Restricts model** to predicting values to **every fifteen second** time stamps

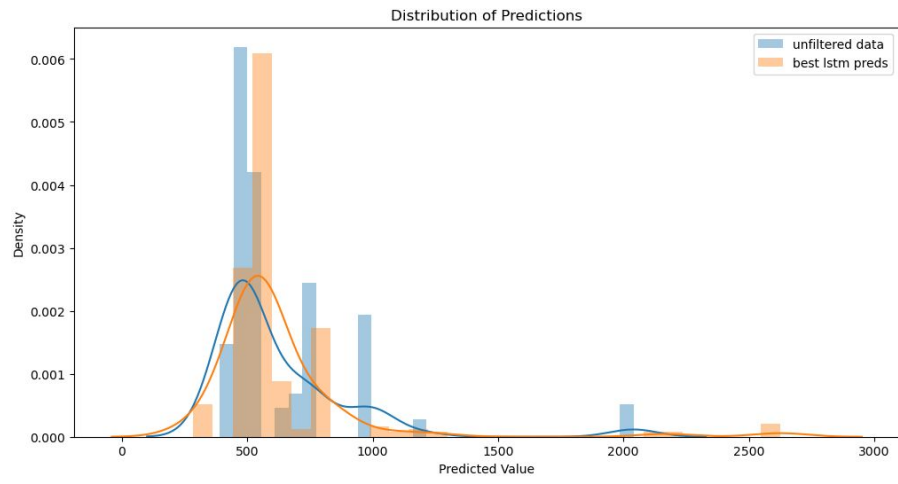
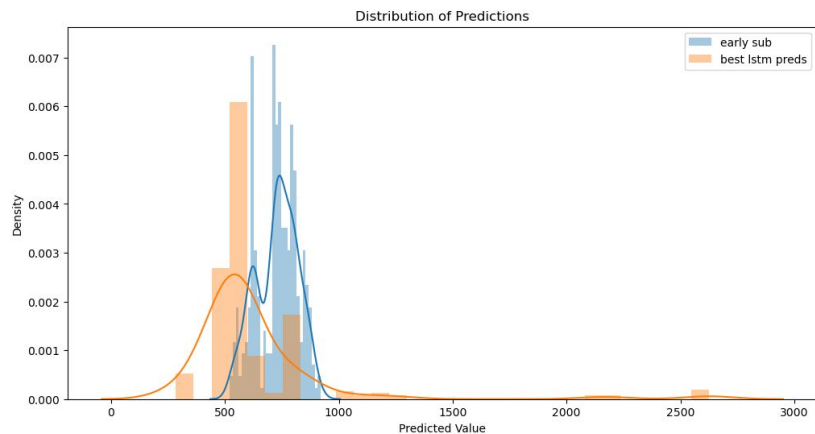
 <b>pred-9.csv</b> Complete · Asif Mahdin · 3d ago	<b>763.20947</b>	<input type="checkbox"/>
 <b>pred-8.csv</b> Complete · Asif Mahdin · 3d ago	<b>762.65786</b>	<input type="checkbox"/>
 <b>pred-7.csv</b> Complete · Asif Mahdin · 3d ago	<b>779.34391</b>	<input type="checkbox"/>
 <b>pred-6.csv</b> Complete · Asif Mahdin · 3d ago	<b>782.80369</b>	<input type="checkbox"/>
 <b>preds.csv</b> Complete · Luke Taylor12345 · 3d ago	<b>812.02183</b>	<input type="checkbox"/>

# Engineering Tricks

- Data Filtering + Augmentation

(Removing noise and unnecessary information, studying the distribution of the test set)

- Early Stopping
- Dropout (up to 50%)
- Feature Removal (empirical improvement)
- **Nvidia A100** to enhance training speed (1s per epoch)





# Experiments

---

# Experiment 1 - MLP Neural Network

Initially built four-layer multilayer perceptron. Sample implementation:

1st Dense Layer: 10 input features -> 256 output features

2nd Dense Layer: 256 -> 512

3rd Dense Layer: 512 -> 16

4th Dense Layer: 16 -> 1

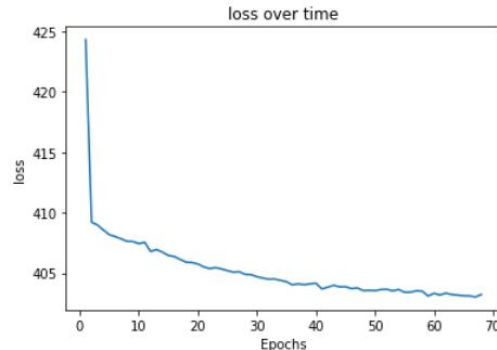
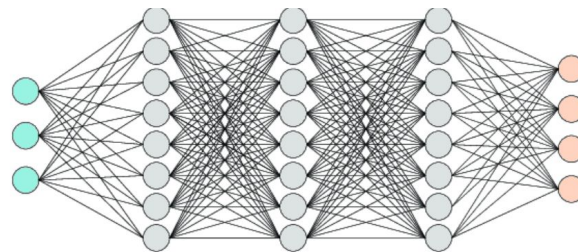
Adam + 1e-3 LR + ReLU for nonlinearity

With additional dropout layer between 1st and 2nd (also tried Batch Norm)

Parameter Count: 145,313 (tested between 40K-350K parameters)

**Result:**

Some learning but stubbornly high loss,  
Kaggle public test results not much better than  
using an average (lowest: ~774)



# Experiment 2: Random forest

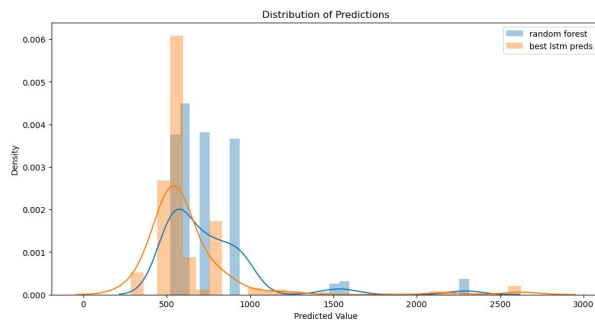
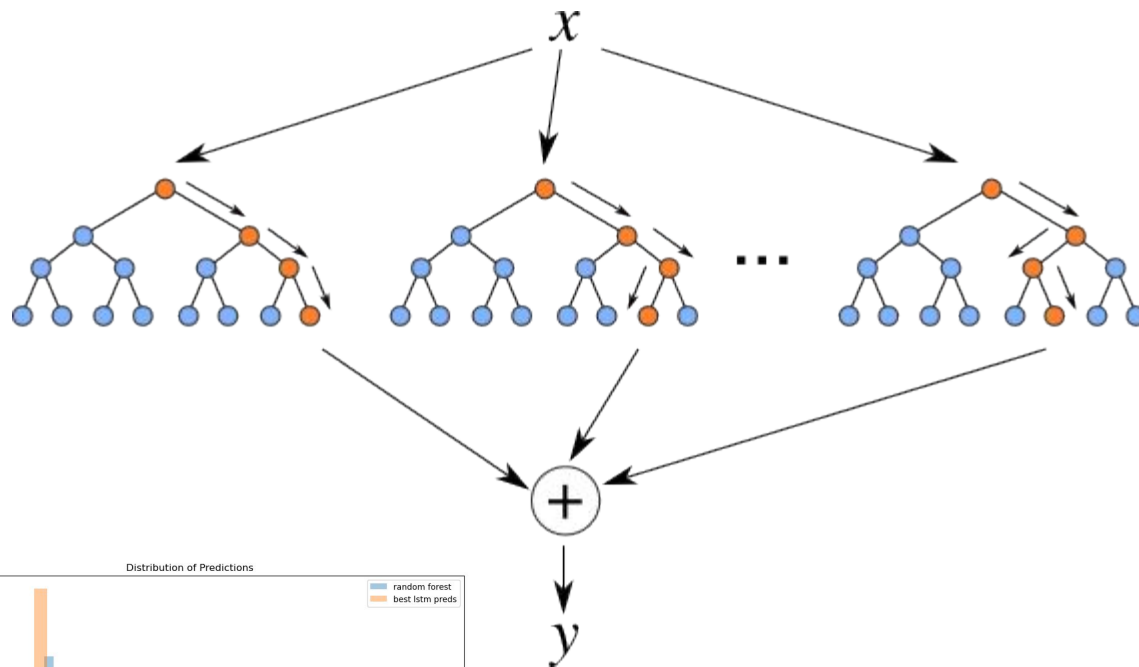
Best grid search parameters: 150  
estimators (100-200 considered), 10  
depth (5-30 considered)

Loss Function: mse

This took about 5 minutes to evaluate

Loss: 616 on processed data (122k  
points)

762 on raw data (1.6 million)

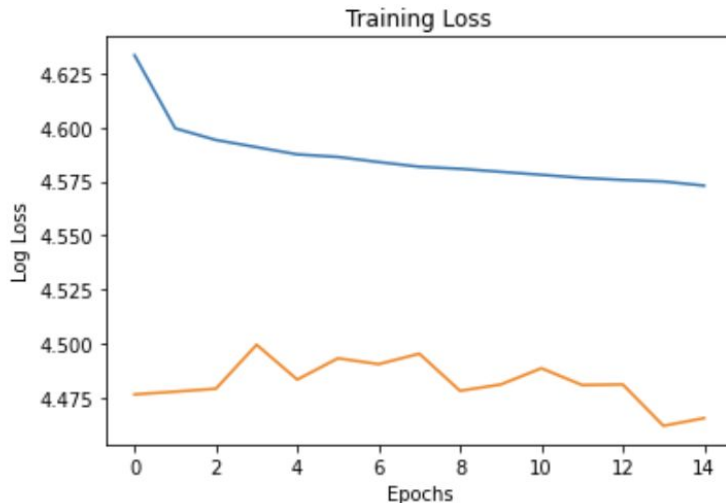


# Experiment 3: LSTM

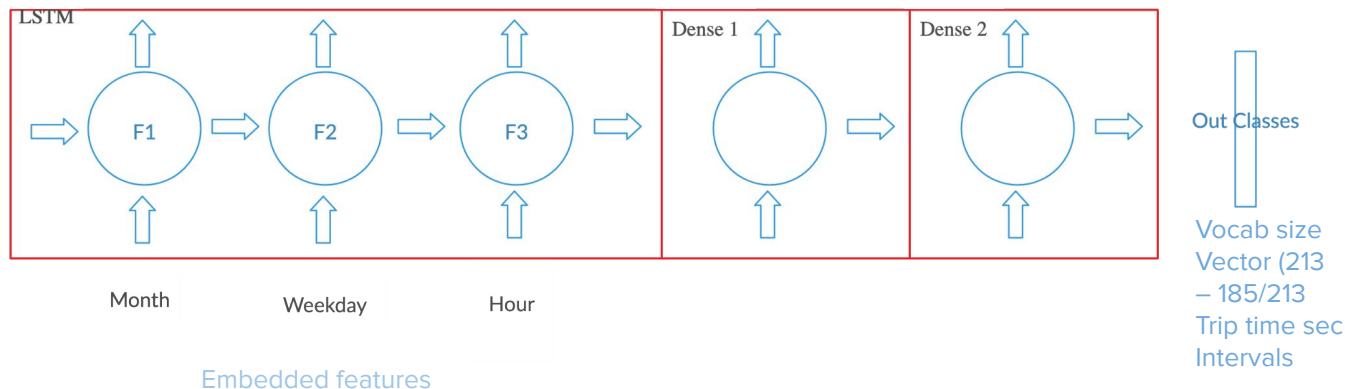
- Embedding Dim: 20
- Hidden Layers: 3
- Dropout: 0.5
- Hidden Dim: 256
- Epochs: 15
- Loss: Negative Log Likelihood
- Sequence (fixed length, order): 1.  
Month 2. Week Day 3. Hour →

## Time

Public Score ⓘ	Select
534.86928	<input type="checkbox"/>
534.97225	<input type="checkbox"/>
568.82073	<input type="checkbox"/>



## Model Inspiration: Shakespeare Sonnet HW



# Discussion

---

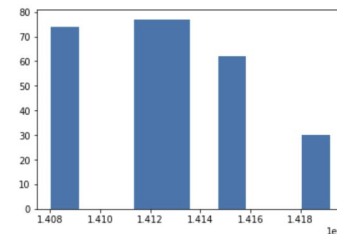
# What have you learned

We have gained significant insights into the practical applications of deep learning models throughout this project. We learned how crucial it is to understand the data we are working on and preprocessing it before model training.

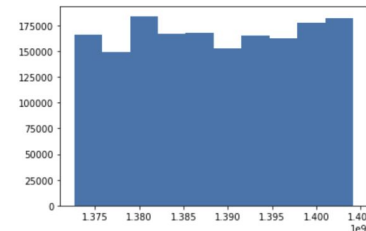
**Understanding the Importance of Exploratory Data Analysis (EDA):** Our initial attempts to apply machine learning models to the raw dataset did not yield any useful results, due to lack of data processing and understanding. This reiterates the importance of EDA in the data science pipeline, as it enables us to discover patterns and check relationships between variables. Ex: hour, quarter hour, location of starting, etc

**Power of Data Filtering and Featurig Engineering:** Through our EDA, we discovered that, the test set contains data from only specific timestamps and months. So, we decided to filter all irrelevant data and reduce the size of our dataset which enhanced our model's performance drastically.

```
In [51]: plt.hist(test['TIMESTAMP']);
```



```
In [52]: plt.hist(train['TIMESTAMP']);
```



# What have you learned

**More data does not mean better results:** The drop in rmse from **762** to **541** after reducing our dataset from 1.6 million points to 122,000 shows a larger dataset does not necessarily lead to better performance. This is because our dataset also contains a lot of noise and irrelevant data compared to the one in test set.

**Challenges in implementation of Machine learning models:** Implementing various machine learning models such as linear regression, random forest, MLP and LSTM we learned first hand the importance of parameter tuning, handling overfitting and making sure our model generalizes well to the unseen data.

**Teamwork:** Teamwork makes the dreamwork!

# Future Work

Moving forward there are several potential areas for improvement.

**More Feature Engineering:** Even though our current model have improved significantly, we still believe there is room for improvement through further feature engineering. Ex: using the polyline information to identify areas of high congestion, identifying and tagging more landmarks at each origin stand

**Hyperparameter Tuning:** We can use python libraries like Optuna to tune the Hyperparameter like embedding dimension, number of LSTM layers, dropout proportion, and other.

**Trying Different Model Architecture:** We could investigate other types of deep learning models such as transformers and other attention-based models which have proven to be excellent when it comes to sequence predictions task. We can also try ensembling methods where multiple models are chained together and their predictions are combined to give an even better accuracy.