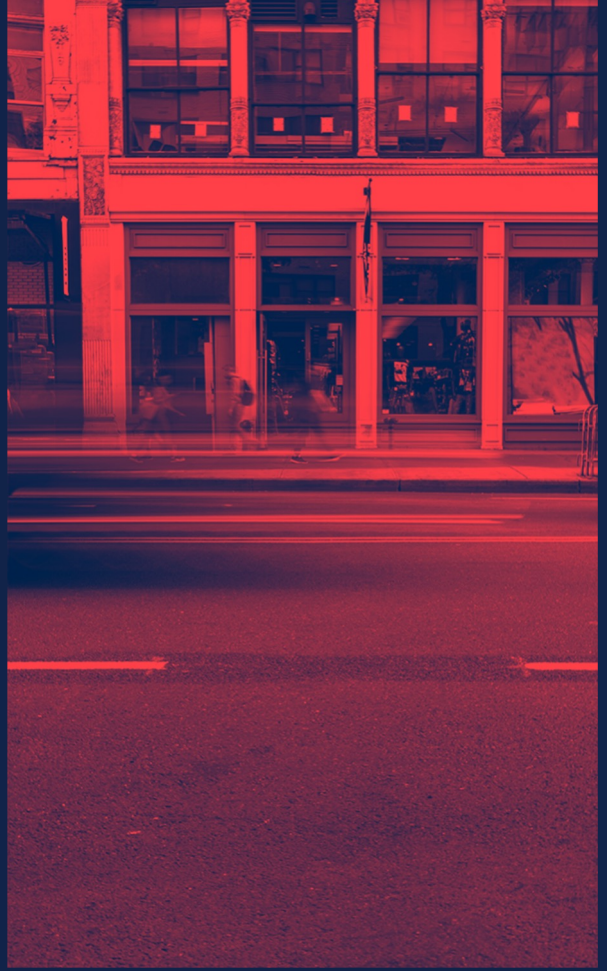


MTA Subway Station Traffic Forecasting

A Time-Series Analysis and forecasting using Random Forest Regression & Extreme Gradient Boosting

PRESENTER

- Thomas Joy II - Applied Mathematician / Data Scientist
- Graduating Fall 2022 from Hunter College with a Bachelor's & Master's of Arts in Mathematics / Applied Mathematics
- Graduated Galvanize's Data Science Boot Camp in January 2021
- Employed with Booz Allen Hamilton, DoD Contracting Firm
- Active-Duty Marine Corps Veteran & Active Air Force Reservist
- I have a dog: Cinny

BACKGROUND INFORMATION

- COVID-19 had a huge impact on MTA Subway ridership, causing ridership to decrease by 63.64%, from a daily weekday average of 5.5M passengers, down to roughly 2M; since then ridership still has still not bounced back to where it was pre-pandemic days.
- The MTA estimates that as of this past week, (30 Nov. - 08 Dec.), daily ridership is only back to 58.2% compared to pre-pandemic days, which is equivalent to 3.2M people using the subway daily.
- Even though ridership still is down, there are still a daily average of 15.4K people commuting in and out of any given subway station.

OVERVIEW OF PROJECT

DATA PROCESSING & CLEANING

- Before you can do anything, you must first load the data and perform any data cleaning necessary (typically one of the longest tasks on any data science project).

EXPLORATORY DATA ANALYSIS (EDA)

- Exploratory data analysis (EDA) is used to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods.
- It helps determine how best to manipulate data sources to get the answers you need, making it easier to discover patterns, spot anomalies, or check assumptions. This is where you search for the “story” that the data is telling.

FEATURE ENGINEERING & MODEL SELECTION

- Feature Engineering refers to the process of using domain knowledge to select and transform the most relevant variables from the raw data when creating a predictive model using ML or statistical modeling.
- Model Selection in Machine Learning is selecting the best model for your data. Different models will perform differently on different data sets and potentially by a large margin.

OVERVIEW OF PROJECT

MODEL TRAINING

- Model training is the phase in the project lifecycle where we try to fit the best combination of weights and bias to a machine learning algorithm to minimize a loss function over the prediction range.

MODEL EVALUATION

- Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses.
- Model evaluation is important to assess the efficacy of a model during initial research phases, and it also plays a role in model monitoring.

GRID SEARCH FOR FINE-TUNING HYPERPARAMETERS

- Hyperparameters are parameters whose values control the learning process and determine the values of model parameters that a learning algorithm ends up learning.
- Grid search is a process that searches exhaustively through a manually specified subset of the hyperparameter space of the targeted algorithm.
- After the best parameters are found, re-train and re-evaluate your model's performance.

THE DATA

Note: All time-series data was filtered between 10/2014 - 09/2022

TURNSTILE DATA FROM OCT. 2014 – SEPT. 2022

- 11 Features
- 83.13M Observations
- 10.62 GB memory

	C/A	UNIT	SCP	STATION	LINENAME	DIVISION	DATE	TIME	DESC	ENTRIES	EXITS
0	A060	R001	00-00-00	WHITEHALL ST	R1	BMT	2014-10-11	01:00:00	REGULAR	805439.0	1141080.0
1	A060	R001	00-00-00	WHITEHALL ST	R1	BMT	2014-10-11	05:00:00	REGULAR	805459.0	1141141.0
2	A060	R001	00-00-00	WHITEHALL ST	R1	BMT	2014-10-11	09:00:00	REGULAR	805589.0	1141257.0
3	A060	R001	00-00-00	WHITEHALL ST	R1	BMT	2014-10-11	13:00:00	REGULAR	805834.0	1141512.0
4	A060	R001	00-00-00	WHITEHALL ST	R1	BMT	2014-10-11	17:00:00	REGULAR	806150.0	1141903.0
...	...	...	...	...	...	...	...	...	...	...	...
83129623	TRAM2	R469	00-05-01	RIT-ROOSEVELT	R	RIT	2022-09-16	05:00:00	REGULAR	0.0	804.0
83129624	TRAM2	R469	00-05-01	RIT-ROOSEVELT	R	RIT	2022-09-16	09:00:00	REGULAR	0.0	804.0
83129625	TRAM2	R469	00-05-01	RIT-ROOSEVELT	R	RIT	2022-09-16	13:00:00	REGULAR	0.0	804.0
83129626	TRAM2	R469	00-05-01	RIT-ROOSEVELT	R	RIT	2022-09-16	17:00:00	REGULAR	0.0	804.0
83129627	TRAM2	R469	00-05-01	RIT-ROOSEVELT	R	RIT	2022-09-16	21:00:00	REGULAR	0.0	804.0

83129628 rows × 11 columns

THE DATA

Note: *"Data Set Station"* was manually made column, aligning the given Station name in the turnstile data with the correct Stop Name on the lookup table.

SUBWAY STATION LOOKUP TABLE

- 12 Features
- 617 Observations
- 57 KB memory

	Station ID	Complex ID	GTFS Stop ID	Data Set Station	Division	Line	Stop Name	Borough	Daytime Routes	Structure	GTFS Latitude	GTFS Longitude
0	119	119	L06	1 AV	BMT	Canarsie	1 Av	M	L	Subway	40.730953	-73.981628
1	119	119	L06	1 AVE	BMT	Canarsie	1 Av	M	L	Subway	40.730953	-73.981628
2	156	156	A18	103 ST	IND	8th Av - Fulton St	103 St	M	B C	Subway	40.796092	-73.961454
3	309	309	119	103 ST	IRT	Broadway - 7Av	103 St	M	1	Subway	40.799446	-73.968379
4	395	395	624	103 ST	IRT	Lexington Av	103 St	M	6	Subway	40.790600	-73.947478
...	...	...	...	...	...	...	...	...	...	...	...	...
612	171	624	E01	WORLD TRADE CTR	IND	8th Av - Fulton St	World Trade Center	M	E	Subway	40.712582	-74.009781
613	328	328	138	WTC-CORTLANDT	IRT	Broadway - 7Av	WTC Cortlandt	M	1	Subway	40.711835	-74.012188
614	235	235	F18	YORK ST	IND	6th Av - Culver	York St	Bk	F	Subway	40.701397	-73.986751
615	364	364	606	ZEREGA AVE	IRT	Pelham	Zerega Av	Bx	6	Elevated	40.836488	-73.847036
616	364	364	606	ZEREGA AV	IRT	Pelham	Zerega Av	Bx	6	Elevated	40.836488	-73.847036

617 rows × 12 columns

THE DATA

Note: Dates between 10/2014 – 03/2020 are considered to have zero COVID-19 cases

COVID-19 DATA BY DAY

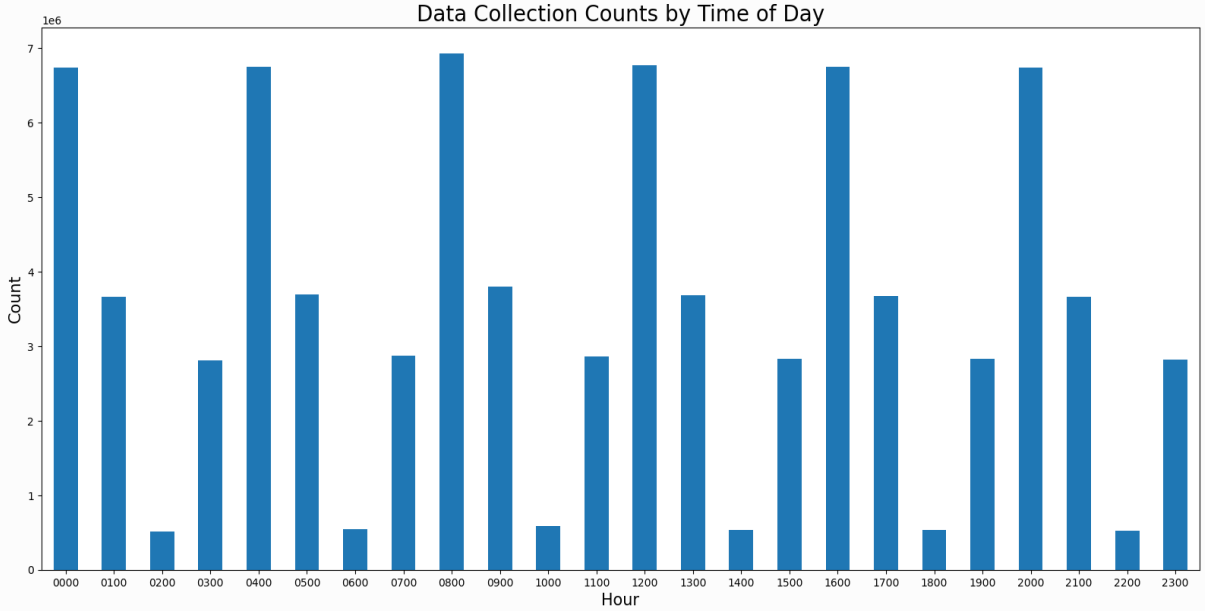
- 26 Features
- 1,009 Observations
- 212 KB memory

	date_of_interest	CASE_COUNT	PROBABLE_CASE_COUNT	CASE_COUNT_7DAY_AVG	ALL_CASE_COUNT_7DAY_AVG	BX_CASE_COUNT	BX_PROBABLE_CASE_COUNT	BX_CA
0	02/29/2020	1	0	0	0	0	0	
1	03/01/2020	0	0	0	0	0	0	
2	03/02/2020	0	0	0	0	0	0	
3	03/03/2020	1	0	0	0	0	0	
4	03/04/2020	5	0	0	0	0	0	
...	...	...	...	...	...	...	...	
1004	11/29/2022	3745	960	2614	3266	685	193	
1005	11/30/2022	3108	843	2676	3334	568	151	
1006	12/01/2022	3338	450	3011	3683	640	80	
1007	12/02/2022	2674	391	3074	3752	459	77	
1008	12/03/2022	1337	622	2979	3659	197	117	

1009 rows × 26 columns

CLEANING TURNSTILE DATA

- Assumptions:
- Aggregated to 4-hr intervals
- Removed Staten Island Railway due to data inconsistency
- Outlier trimming threshold set to 20K entries or exits of a single turnstile within 4-hr interval
- Total Time Spent Cleaning: 40+ hrs




```
Time: 00:00          Unique Stations where Data was collected: 568
Time: 01:00          Unique Stations where Data was collected: 359
Time: 02:00          Unique Stations where Data was collected: 56
Time: 03:00          Unique Stations where Data was collected: 361
Time: 04:00          Unique Stations where Data was collected: 567
Time: 05:00          Unique Stations where Data was collected: 446
Time: 06:00          Unique Stations where Data was collected: 426
Time: 07:00          Unique Stations where Data was collected: 551
Time: 08:00          Unique Stations where Data was collected: 572
Time: 09:00          Unique Stations where Data was collected: 562
Time: 10:00          Unique Stations where Data was collected: 537
Time: 11:00          Unique Stations where Data was collected: 535
Time: 12:00          Unique Stations where Data was collected: 569
Time: 13:00          Unique Stations where Data was collected: 480
Time: 14:00          Unique Stations where Data was collected: 405
Time: 15:00          Unique Stations where Data was collected: 493
Time: 16:00          Unique Stations where Data was collected: 570
Time: 17:00          Unique Stations where Data was collected: 461
Time: 18:00          Unique Stations where Data was collected: 334
Time: 19:00          Unique Stations where Data was collected: 451
Time: 20:00          Unique Stations where Data was collected: 567
Time: 21:00          Unique Stations where Data was collected: 408
Time: 22:00          Unique Stations where Data was collected: 213
Time: 23:00          Unique Stations where Data was collected: 404
```

CLEANING TURNSTILE DATA

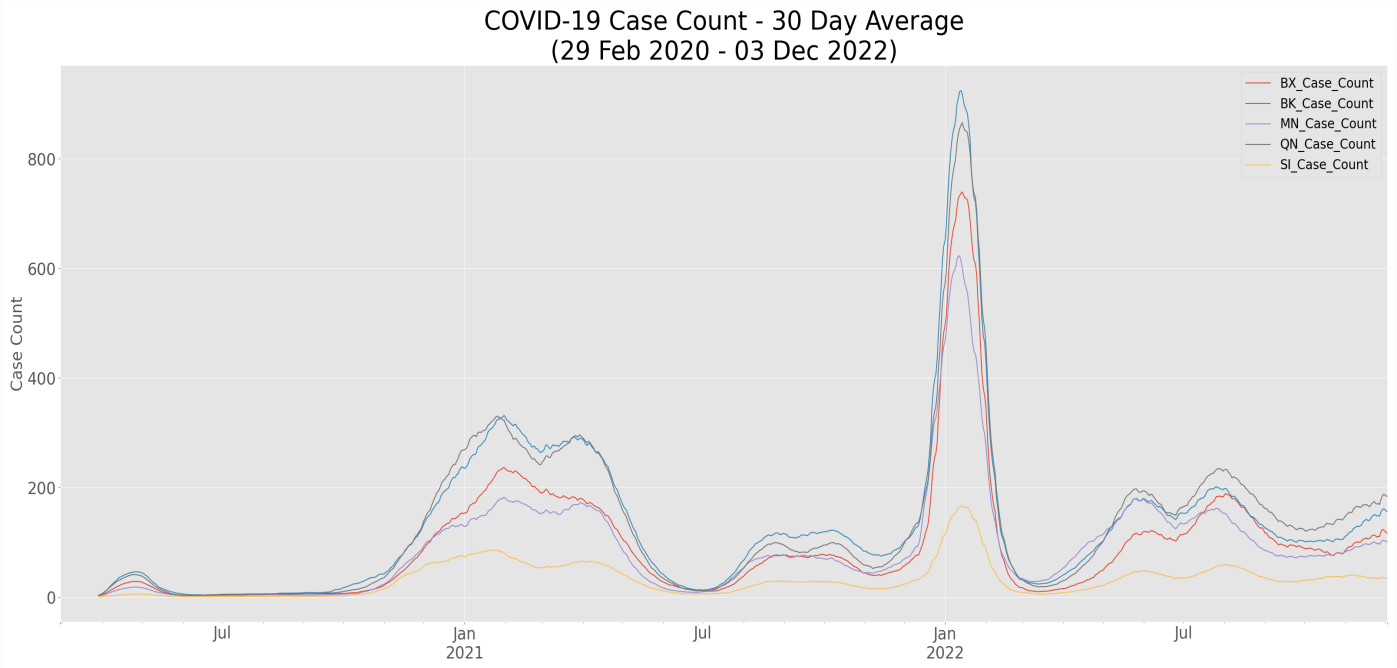
- Features: 22
- Observations: 77,076,625
- Unique Subway Stations: 474
- Unique Turnstiles: 4,852
- Start Date: 11 Oct. 2014
- End Date: 16 Sept. 2022

	DateTime	Date	Year	Month	Day	Hour	DOW	Turnstile_ID	Entries	Exits	Traffic	Station ID	Complex ID	GTFS Stop ID	Division	Line	Station	Borough	Daytime Routes	Structure	Latitude	Longitude
0	2014-10-11 00:00:00	2014-10-11	2014	10	11	0	5	A002-R051-02-00-00	0	0	0	275	612	F11	IND	Queens Blvd	Lexington Av/53 St	M	E M	Subway	40.757552	-73.969055
1	2014-10-11 00:00:00	2014-10-11	2014	10	11	0	5	A002-R051-02-00-00	0	0	0	7	613	R11	BMT	Astoria	Lexington Av/59 St	M	N W R	Subway	40.762660	-73.967258
2	2014-10-11 00:00:00	2014-10-11	2014	10	11	0	5	A002-R051-02-00-00	0	0	0	7	613	R11	BMT	Astoria	Lexington Av/59 St	M	N W R	Subway	40.762660	-73.967258
3	2014-10-11 00:00:00	2014-10-11	2014	10	11	0	5	A002-R051-02-00-00	0	0	0	223	223	B08	IND	63rd St	Lexington Av/63 St	M	F Q	Subway	40.764629	-73.966113
4	2014-10-11 04:00:00	2014-10-11	2014	10	11	4	5	A002-R051-02-00-00	45	4	49	275	612	F11	IND	Queens Blvd	Lexington Av/53 St	M	E M	Subway	40.757552	-73.969055
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
77076620	2022-09-16 04:00:00	2022-09-16	2022	9	16	4	4	R109-R305-03-00-02	27	16	43	328	328	138	IRT	Broadway - 7Av	WTC Cortlandt	M	1	Subway	40.711835	-74.012188
77076621	2022-09-16 08:00:00	2022-09-16	2022	9	16	8	4	R109-R305-03-00-02	1	9	10	328	328	138	IRT	Broadway - 7Av	WTC Cortlandt	M	1	Subway	40.711835	-74.012188
77076622	2022-09-16 12:00:00	2022-09-16	2022	9	16	12	4	R109-R305-03-00-02	27	32	59	328	328	138	IRT	Broadway - 7Av	WTC Cortlandt	M	1	Subway	40.711835	-74.012188
77076623	2022-09-16 16:00:00	2022-09-16	2022	9	16	16	4	R109-R305-03-00-02	55	39	94	328	328	138	IRT	Broadway - 7Av	WTC Cortlandt	M	1	Subway	40.711835	-74.012188
77076624	2022-09-16 00:00:00	2022-09-16	2022	9	16	0	4	R109-R305-03-00-02	137	44	181	328	328	138	IRT	Broadway - 7Av	WTC Cortlandt	M	1	Subway	40.711835	-74.012188

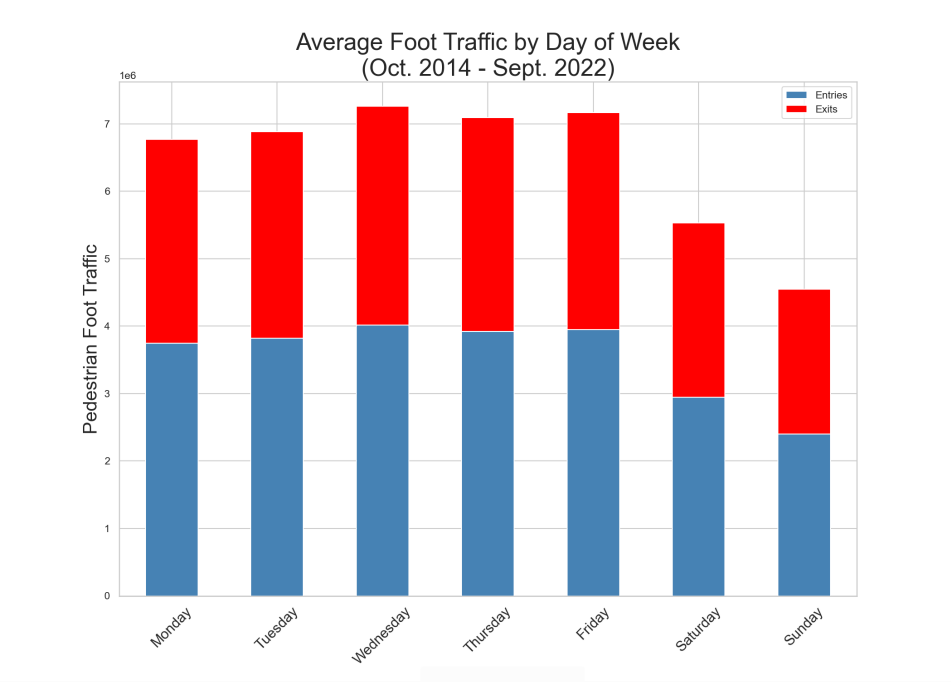
77076625 rows × 22 columns

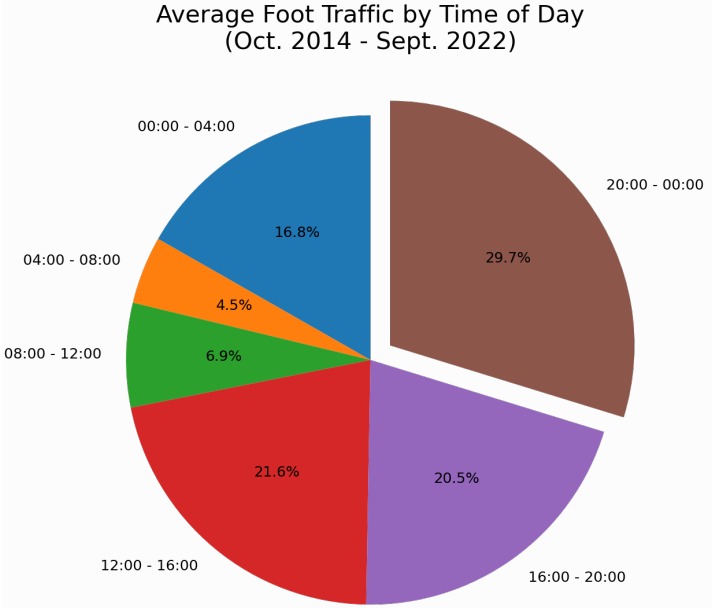
CLEANING COVID-19 DATA

- Features: 7
- Observations: 1,006
- Start Date: 29 Feb. 2020
- End Date: 16 Sept. 2022



EXPLORATORY DATA ANALYSIS
TURNSTILE DATA

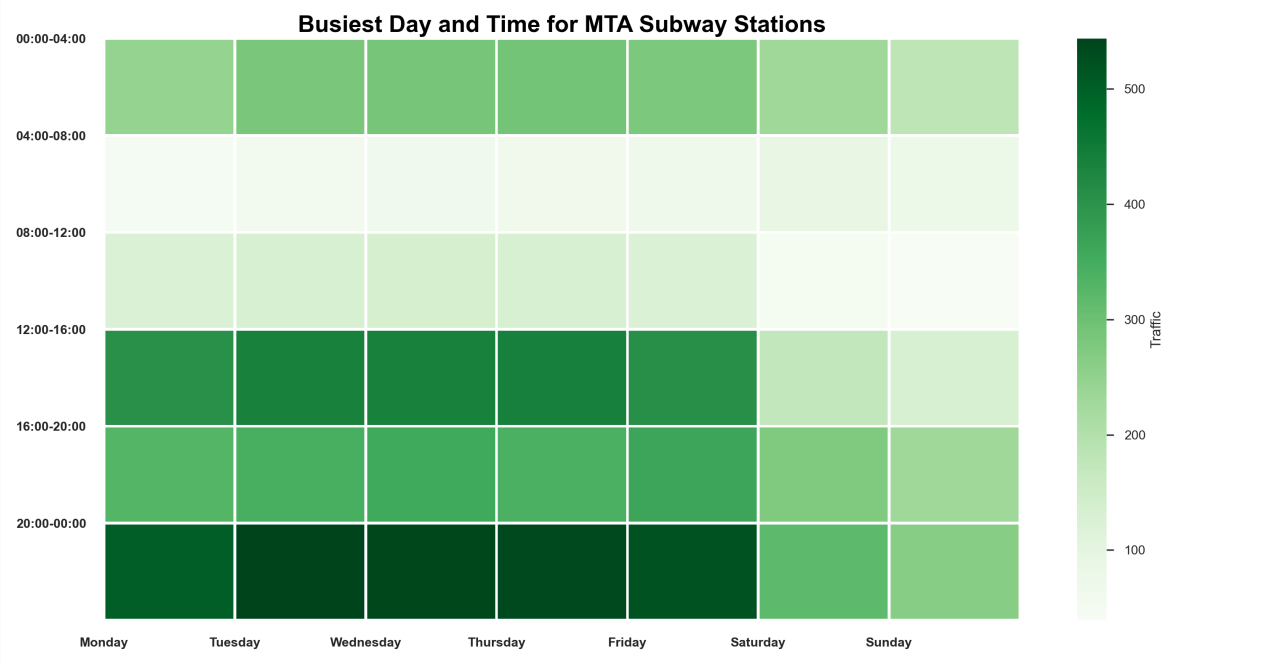




EXPLORATORY DATA ANALYSIS
TURNSTILE DATA

Busiest Date/Time for the MTA:

- Date: 13 Nov. 2014
- Time: 16:00-20:00
- Traffic: 5,897,128



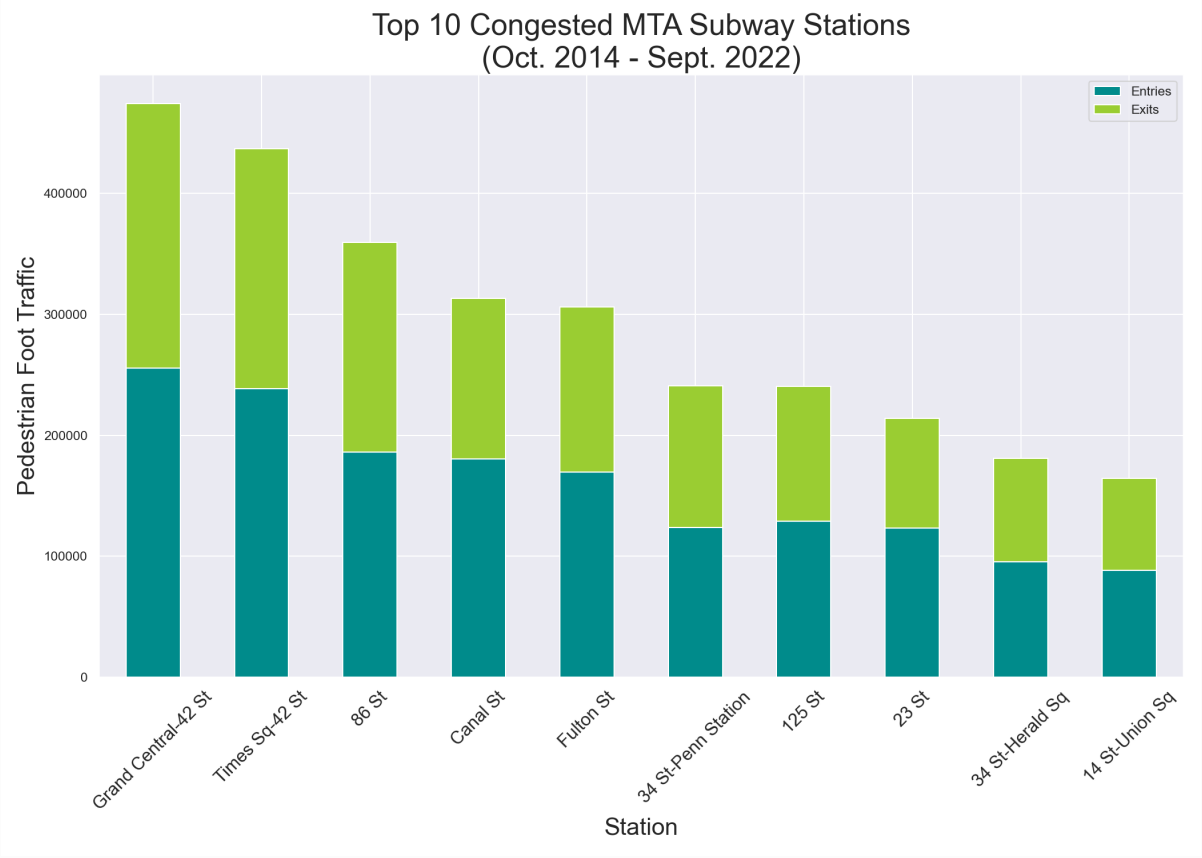
EXPLORATORY DATA ANALYSIS
TURNSTILE DATA

Busiest Day of the Week for Grand Central-42 St:

- Day: Wednesday
- Average Traffic: 614,429

Busiest Time of Day for Grand Central-42 St:

- Time: 16:00-20:00
- Average Traffic: 213,483



Note: Staten Island turnstile data was removed due to inconsistencies in the data.

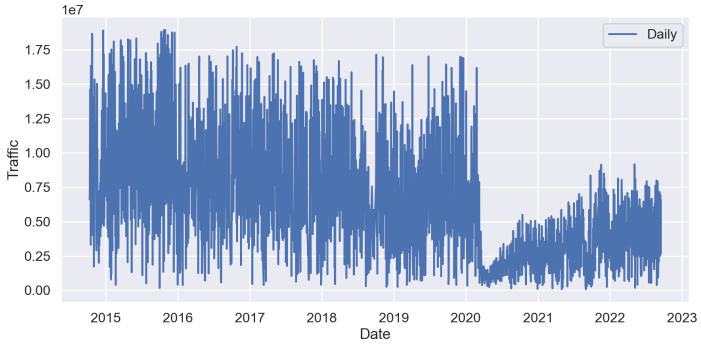
EXPLORATORY DATA ANALYSIS
TURNSTILE DATA

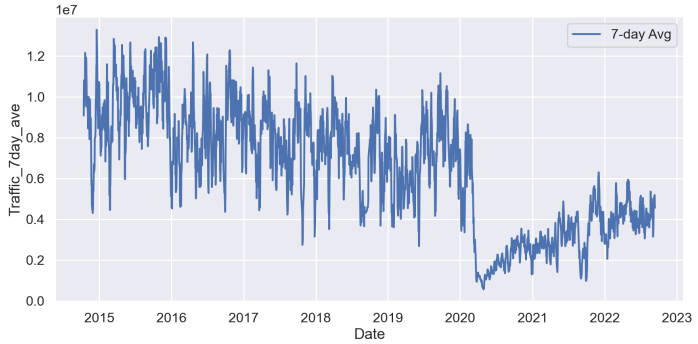
Average Daily Rider by Borough:

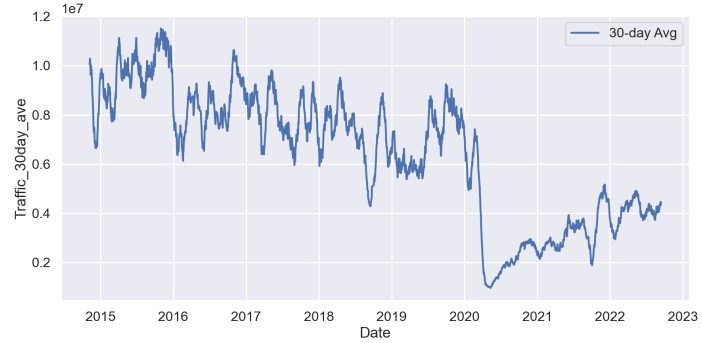
- Manhattan: 4,040,701
- Queens: 580,039
- Brooklyn: 1,507,526
- Bronx: 339,241

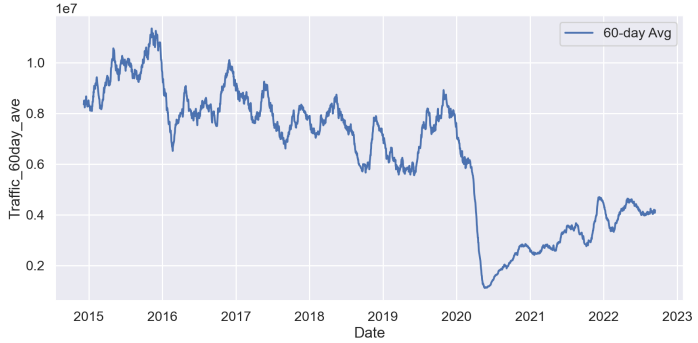
EXPLORATORY DATA ANALYSIS
TURNSTILE DATA

MTA Subway Traffic over Time
(Oct. 2014 - Sept. 2022)







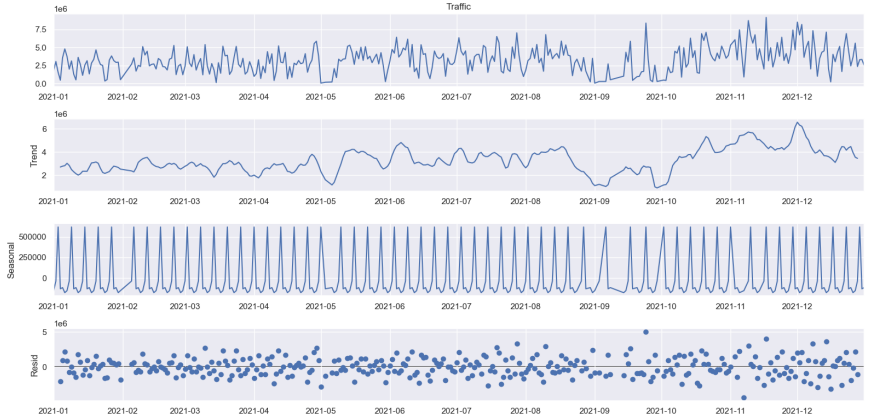


EXPLORATORY DATA ANALYSIS
TURNSTILE DATA

2019

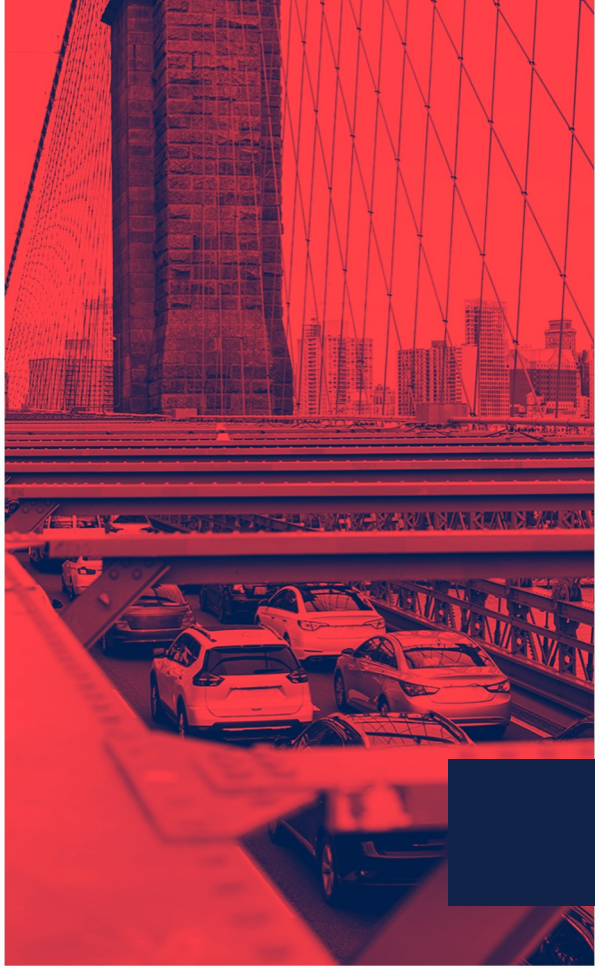
2020

2021



TIME TO BUILD A MACHINE LEARNING MODEL

Now that we are ready to start building our model, first we must prepare the data. This involves transforming the raw, but cleaned, data in to a form that can be modeled using machine learning algorithms.



FEATURE ENGINEERING

DateTime	Year	Month	Day	Hour	DOW	Traffic	Station ID	Complex ID	Borough	Structure_At Grade	Structure_Elevated	Structure_Open Cut	Structure_Subway	Structure_Viaduct	Date	COVID_Case_Count
2014-10-11 00:00:00	2014	10	11	0	5	0.000000	119	119	0	0	0	0	1	0	2014-10-11	0.000000
2014-10-11 04:00:00	2014	10	11	4	5	4140.000000	119	119	0	0	0	0	1	0	2014-10-11	0.000000
2014-10-11 08:00:00	2014	10	11	8	5	1902.000000	119	119	0	0	0	0	1	0	2014-10-11	0.000000
2014-10-11 12:00:00	2014	10	11	12	5	6239.000000	119	119	0	0	0	0	1	0	2014-10-11	0.000000
2014-10-11 20:00:00	2014	10	11	20	5	11610.000000	119	119	0	0	0	0	1	0	2014-10-11	0.000000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2022-09-16 00:00:00	2022	9	16	0	4	3045.000000	328	328	0	0	0	0	1	0	2022-09-16	70.000000
2022-09-16 04:00:00	2022	9	16	4	4	1017.000000	328	328	0	0	0	0	1	0	2022-09-16	70.000000
2022-09-16 08:00:00	2022	9	16	8	4	148.000000	328	328	0	0	0	0	1	0	2022-09-16	70.000000
2022-09-16 12:00:00	2022	9	16	12	4	2044.000000	328	328	0	0	0	0	1	0	2022-09-16	70.000000
2022-09-16 16:00:00	2022	9	16	16	4	3143.000000	328	328	0	0	0	0	1	0	2022-09-16	70.000000

4645824 rows × 16 columns

Date Features

- Dates and times are rich sources of information that can be used with machine learning models. However, these datetime variables do require some feature engineering to turn them into numerical data.

Categorical Features

- One-hot encoding in machine learning is the conversion of categorical information into a format that may be fed into machine learning algorithms to improve prediction accuracy.

- Another technique for dealing with categorical variables is Ordinal Encoding. In Ordinal encoding, each category value is assigned an integer value (i.e., Manhattan = 0, Queens = 1, etc.)

FEATURE ENGINEERING

Date	COVID_Case_Count	Next_Interval_Value	lag_station_1	lag_station_2	diff_station_1	diff_station_2	rolling_mean_station_6	rolling_mean_station_42	rolling_std_station_6	rolling_std_station_42
2014-10-20	0.000000	914.000000	9901.000000	8535.000000	-4369.000000	-319.000000	4245.333333	3499.119048	3266.557250	2541.431997
2014-10-20	0.000000	5473.000000	5695.000000	9901.000000	8757.000000	4388.000000	4380.000000	3660.690476	3527.362527	2743.500101
2014-10-20	0.000000	14136.000000	914.000000	5695.000000	-5713.000000	3044.000000	4177.833333	3767.500000	3470.956031	2684.189656
2014-10-20	0.000000	7011.000000	5473.000000	914.000000	4798.000000	-915.000000	5497.166667	3977.095238	3687.047893	2763.814356
2014-10-21	0.000000	1124.000000	14136.000000	5473.000000	-5160.000000	-362.000000	5891.000000	3965.880952	3315.648594	2762.201112
...	...	...	...	...	...	...	...	...	...	...
2022-09-15	80.000000	3045.000000	989.000000	3792.000000	6034.000000	411.000000	6334.500000	4021.857143	3300.352390	3052.123775
2022-09-16	70.000000	1017.000000	2786.000000	989.000000	-976.000000	5058.000000	6030.500000	4124.642857	3060.928928	3090.733134
2022-09-16	70.000000	148.000000	3045.000000	2786.000000	-4966.000000	-5942.000000	6109.333333	4146.119048	2937.461194	3075.073747
2022-09-16	70.000000	2044.000000	1017.000000	3045.000000	-2194.000000	-7160.000000	4750.833333	4146.119048	3521.306940	3075.073747
2022-09-16	70.000000	3143.000000	148.000000	1017.000000	6733.000000	4539.000000	4585.500000	4280.000000	3362.275643	3070.898216

Statistical Features

- Statistical-based feature selection methods involve evaluating the relationship between each input variable and the target variable using statistics and selecting those input variables that have the strongest relationship with the target variable.
- In our data, I chose to add lag, difference, rolling mean, and rolling standard deviation statistical-based features.

SPLITTING THE DATA

Date	COVID_Case_Count	Next_Interval_Value	lag_station_1	lag_station_2	diff_station_1	diff_station_2	rolling_mean_station_6	rolling_mean_station_42	rolling_std_station_6	rolling_std_station_42
2014-10-20	0.000000	914.000000	9901.000000	8535.000000	-4369.000000	-319.000000	4245.333333	3499.119048	3266.557250	2541.431997
2014-10-20	0.000000	5473.000000	5695.000000	9901.000000	8757.000000	4388.000000	4380.000000	3660.690476	3527.362527	2743.500101
2014-10-20	0.000000	14136.000000	914.000000	5695.000000	-5713.000000	3044.000000	4177.833333	3767.500000	3470.956031	2684.189656
2014-10-20	0.000000	7011.000000	5473.000000	914.000000	4798.000000	-915.000000	5497.166667	3977.095238	3687.047893	2763.814356
2014-10-21	0.000000	1124.000000	14136.000000	5473.000000	-5160.000000	-362.000000	5891.000000	3965.880952	3315.648594	2762.201112
...	...	...	...	...	...	...	...	...	...	...
2022-09-15	80.000000	3045.000000	989.000000	3792.000000	6034.000000	411.000000	6334.500000	4021.857143	3300.352390	3052.123775
2022-09-16	70.000000	1017.000000	2786.000000	989.000000	-976.000000	5058.000000	6030.500000	4124.642857	3060.928928	3090.733134
2022-09-16	70.000000	148.000000	3045.000000	2786.000000	-4966.000000	-5942.000000	6109.333333	4146.119048	2937.461194	3075.073747
2022-09-16	70.000000	2044.000000	1017.000000	3045.000000	-2194.000000	-7160.000000	4750.833333	4146.119048	3521.306940	3075.073747
2022-09-16	70.000000	3143.000000	148.000000	1017.000000	6733.000000	4539.000000	4585.500000	4280.000000	3362.275643	3070.898216

Training Set

- Start Date: 11 Oct. 2014
- End Date: 31 March 2021
- Shape: (3,785,273; 28)

Validation Set

- Start Date: 01 April 2021
- End Date: 16 Sept. 2022
- Shape: (840,301; 28)

MODEL SELECTION

When selecting the proper model, you must first have knowledge of how the model works and how your data is formatted. In our case, we formatted our data from a time-series to a supervised learning format, allowing us to experiment with ensembled models.

DECISION

In our experiment, we chose to test the Random Forest Regression & Extreme Gradient Boosting machine learning algorithms; as they tend to perform well and train quickly on larger data sets.

BASELINE MODEL

Baseline Metrics:
- Accuracy: 8.10%
- Mean Absolute Error: 4,660.32
- Root Mean Squared Error: 7,049.96

Random Forest Regression:
- Using Sci-Kit Learn Machine Learning in Python package
- Parameters: Default

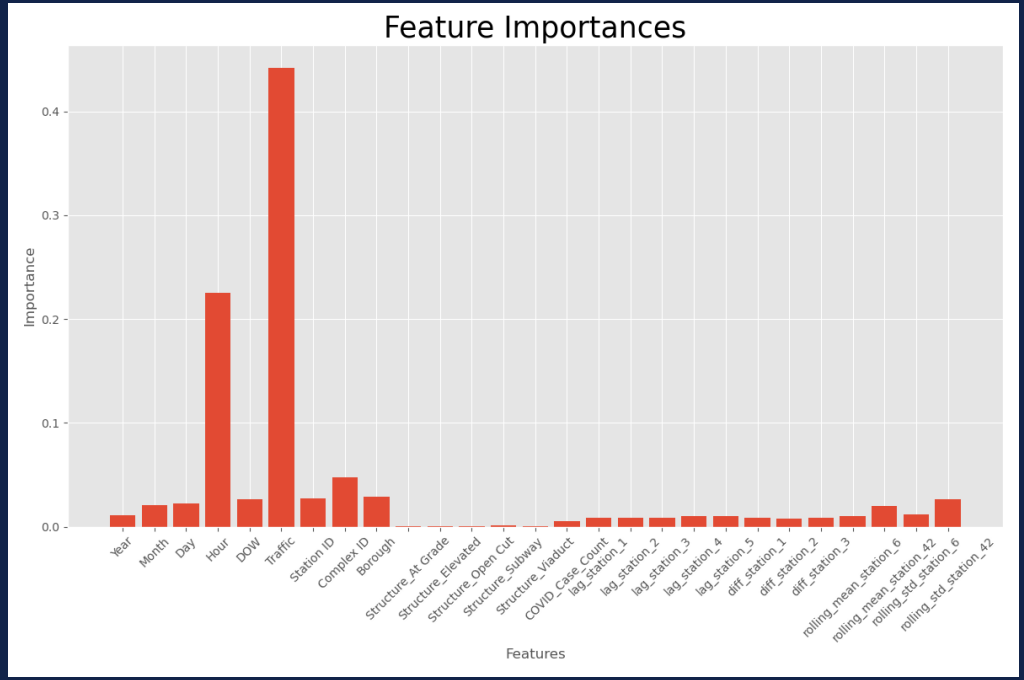
Evaluation Metrics:
- Accuracy: 16.32%
- Mean Absolute Error: 2,346.47
- Root Mean Squared Error: 3,484.92

Extreme Gradient Boosting Regression:
- Using XGBoost API
- Parameters: Default

Evaluation Metrics:
- Accuracy: 18.21%
- Mean Absolute Error: 2,096.55
- Root Mean Squared Error: 2,990.38

FEATURE SELECTION

- Feature Selection is the method of reducing the input variable to your model by using only relevant data and getting rid of noise in data.
- In our case, we are going to trim all the features that have less than a 2% importance.



GRID SEARCH ON HYPERPARAMETERS

Note: Parameters shown are for Random Forest, Extreme Gradient Boosting went through similar Grid Search

Step 1: Randomized Grid Search

Number of iterations: 100
Parameter Grid:

- Number of Estimators: (200, 300, 400, 500, 600, 800, 1000)
- Max Features: ('auto', 'sqrt')
- Max Depth: (10, 20, 30, 40, 50, 60, None)
- Min Sample Leafs: (1, 2, 4, 6, 10)
- Min Sample Splits: (2, 5, 10)
- Bootstrap: (True, False)

The goal of the Randomized Grid Search is to locate "General Area" of hyperparameters.

Note: Large Grid Search Parameters can consume large computation power & time. (E.g., This took 36+ hrs to run for each model)

Optimal Hyperparameters

- Number of Estimators: 300
- Max Features: 'auto'
- Max Depth: 40
- Min Sample Leafs: 4
- Min Sample Splits: 10
- Bootstrap: True

Step 2:

More Intensive Grid Search in the area of best parameters from the Randomized Grid Search.

Number of iterations: 100
Parameter Grid:

- Number of Estimators: (250, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350)
- Max Features: ('auto', 'sqrt')
- Max Depth: (20, 25, 30, 35, 40, 45)
- Min Sample Leafs: (3, 4, 5, 6, 7, 8)
- Min Sample Splits: (8, 9, 10, 11, 12)
- Bootstrap: (True, False)

RESULTS

❑ Random Forest Regression:
Using Sci-Kit Learn Machine Learning in Python package

➢ Parameters:
- Number of Estimators: 300
- Max Features: ‘auto’
- Max Depth: 40
- Min Sample Leafs: 4
- Min Sample Splits: 10
- Bootstrap: True

❑ Evaluation Metrics:
- Accuracy: 76.13%
- Mean Absolute Error: 707.88
- Root Mean Squared Error: 1,026.24

❑ Extreme Gradient Boosting Regression:
Using XGBoost API

➢ Parameters:
- Number of Estimators: 100
- Learning Rate: 0.15
- Max Depth: 5
- Objective: ‘reg:squarederror’
- Tree Method: ’Hist’

❑ Evaluation Metrics:
- Accuracy: 68.48%
- Mean Absolute Error: 1,004.62
- Root Mean Squared Error: 2,221.05

CONCLUSION

- Out of the models tested, the Random Forest Regression Model by SciKit-Learn's python package performed the best.

- Even though it is clear to the audience that COVID-19 was the cause of the drastic decrease in MTA ridership, given the format of the data given, we have shown that it was not of significant importance in the Traffic Forecasting models.

- Pros & Cons
  - Works well & performs well with large data sets.
  - Using a supervised learning method on time-series data can cause you to lose certain information that can only be found in time-series data (e.g., Seasonality, Trend, Autocorrelation, etc.).

NEXT STEPS

- Continue collecting more updated data to feed into the model for training.
- Experiment with more Statistical, Machine Learning, and Deep Learning algorithms; i.e., LSTM, VAR, ARIMA, etc.

RESOURCES

Links

- https://new.mta.info/coronavirus/ridership
- http://web.mta.info/developers/turnstile.html
- https://github.com/nychealth/coronavirus-data

THANKS!

Do you have any questions?

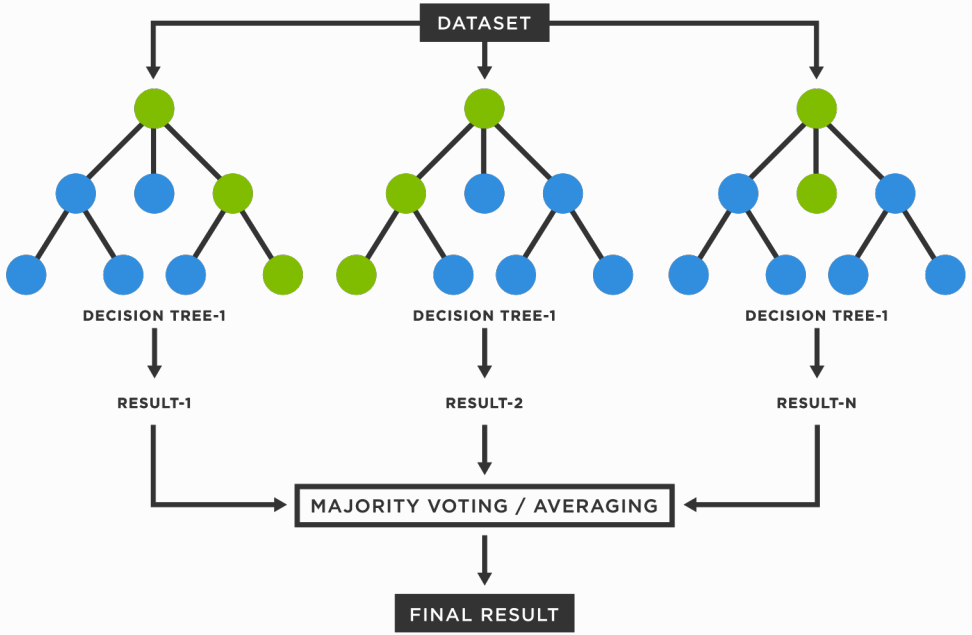
Thomas Joy II – Applied Mathematician | Data Scientist
tomjjoy123@gmail.com
+1 718 869 9409

https://github.com/tomjjoy
https://www.linkedin.com/in/thomas-joy-ii-37a246153/

This Presentation will be posted onto GitHub

RANDOM FOREST REGRESSION

- The Random Forest Algorithm is a type of supervised learning algorithm that uses ensemble methods (bagging) to solve both regression and classification problems.
- The algorithm operates by constructing a multitude of decision trees at training time and outputting the mean/mode of prediction of the individual trees.



EXTREME GRADIENT BOOSTING REGRESSION

- Extreme Gradient Boosting (XGBoost) is a popular and efficient open-source implementation of the gradient boosted trees algorithm. Gradient boosting is a supervised learning algorithm, which attempts to accurately predict a target variable by combining the estimates of a set of simpler, “weaker” models.

- XGBoost minimizes a regularized (L1 and L2) objective function that combines a convex loss function and a penalty term for model complexity. The training proceeds iteratively, adding new trees that predict the residuals or errors of prior trees that are then combined with previous trees to make the final prediction.

- It's called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.