Examples of Machine Learning with TBFM Data
SWIM Industry/FAA Team (SWIFT) Briefing

February 27th, 2020

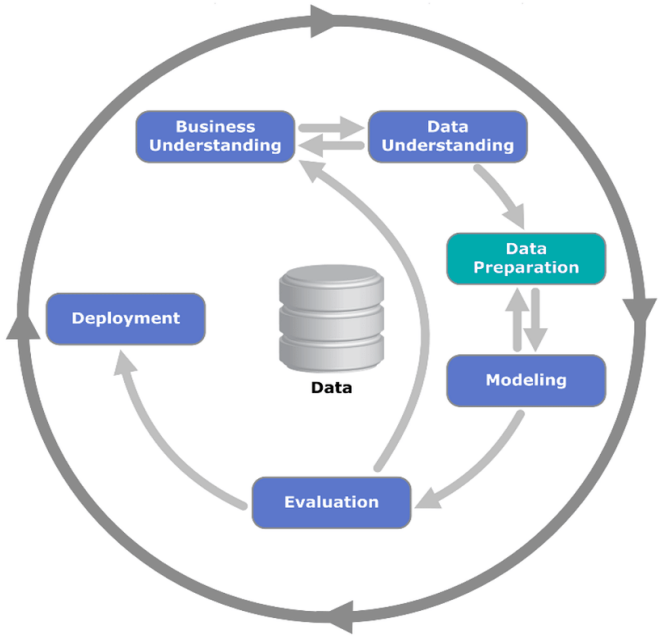
Al Capps - Al.Capps@nasa.gov

Objectives

- *Describe Problem Solving Workflow*
  - Apply a cross-industry standard collaborative workflow to aviation data mining
- *Provide Examples*
  - Demonstrate versatility of analytical solutions via the use of a collaborative workflow in rapid development of machine learning models on different NAS challenges
- *Demonstrate Machine Learning as a Service*
  - Demonstrate how solutions can be rapidly deployed for real-time operations, not just post operations analysis
- *Obtain required input on a specific problem*
  - Obtain problem definition clarification information from the aviation community on the ‘High TBFM Delay’ problem mentioned in prior SWIFT meetings
- *Identify who can meet more frequently for faster progress*
  - Convey a sense of urgency to work quickly and collaboratively on common analytical aviation needs

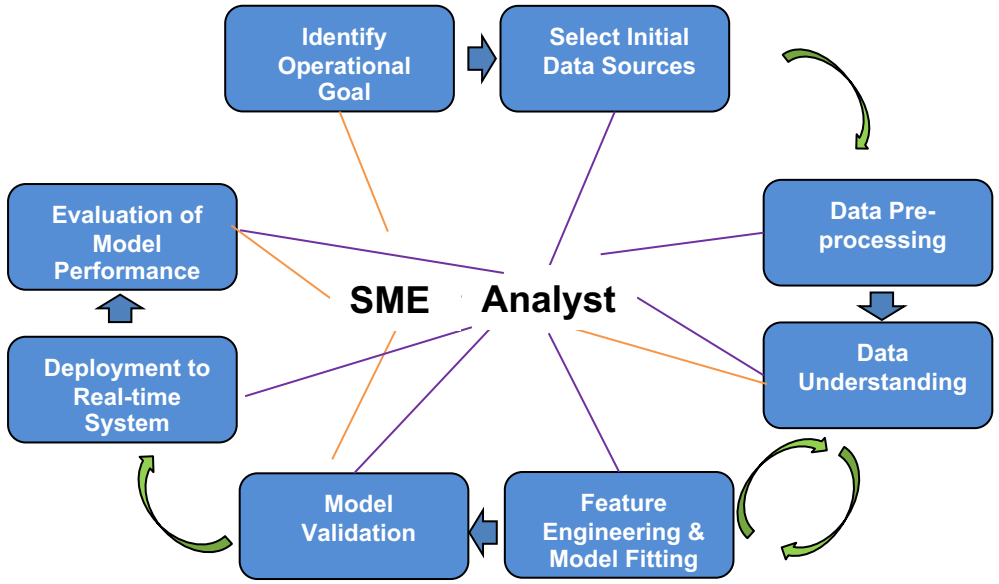
CRISP-DM Process Applied to Aviation

50K Foot View



Cross-Industry Standard Process for Data Mining (**CRISP-DM**)

Variation of CRISP-DM Used by NASA



A key point to providing these two workflows is to highlight the ***necessity*** of an iterative workflow between SMEs and Analysts.

Each step is important, and there are experts in each of these workflow areas that can perform these tasks quickly.

Example 1 - Motivation and Operational Goal

Motivation

- It may surprise some aviation enthusiasts to learn that at large multi-runway airports in the National Airspace System (NAS), the arrival landing runway is often not known ahead of the actual landing

- Knowledge of landing runway can provide benefits:
  - Landing time prediction
  - Taxi-in time prediction
  - Gate Conflict prediction
  - Gate resource utilization information (e.g. tugs, etc.)

Goals

- Allow operators and ATC systems to identify the most likely landing runway
- Provide this service in near real-time, targeting at least 1 hour ahead of landing
- Report the accuracy of the predictions provided
- Strive for maintainability and scalability across all NAS airports

Identify Operational Goal

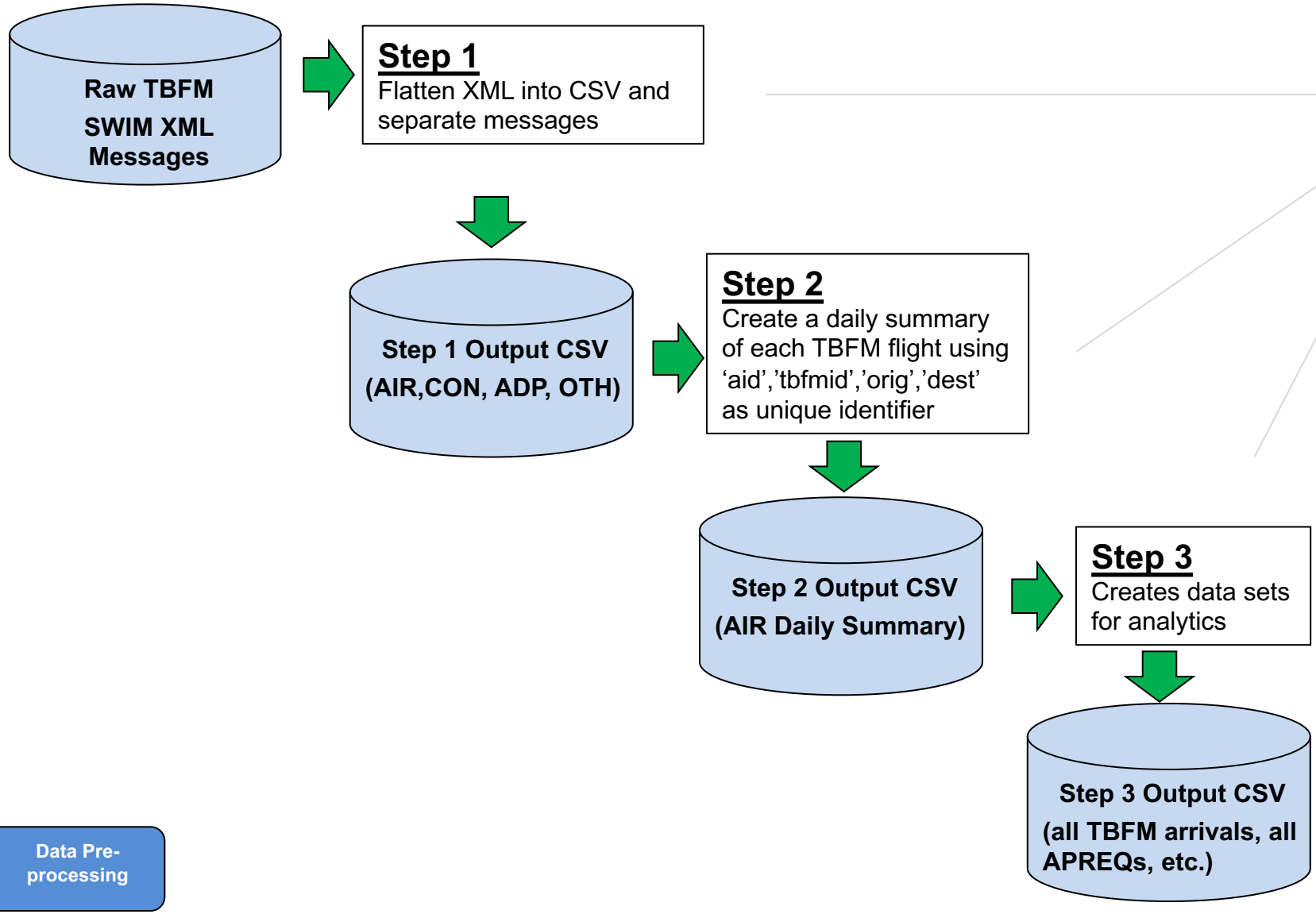
Focus on TBFM SWIM Data

- TBFM SWIM
  - Predicts a landed runway for its own internal processes. This could serve as a useful baseline to assess merit of machine learning comparison.
  - Has other attributes/features that may be beneficial to prediction. Tax payers spent significant funding adapting TBFM across the NAS so additional benefit is desirable.

- Truth data for actual landed runway
  - Controller scratchpad entries provided in TBFM (at some sites)
  - Analysis of alternative sources for larger truth data set likely leads to additional data sources that need to be merged with this initial data set.

Select Initial Data Sources

Python Scripts to Create TBFM SWIM "AIR" Dataset



For this presentation, python scripts were created for each step which pull from public TBFM SWIM data. There are many ways to accomplish this in other languages or designs.

Data Pre-processing

Data Understanding - Description

- For a description of the TBFM SWIM data source and its operational context, see
  - https://nsrr.faa.gov/nsrr-library-document/9298
  - Specifically, section 4.1.1 contains a definition of the data elements used

- Of the columns available in TBFM AIR messages, the ones utilized for this work were:
  - "trw"- TRACON runway assigned by the controller. This was considered "truth" runway. Note: other sources of runway truth are currently being evaluated, especially given controller scratchpad entries are not available for all airports.
  - "rwy" -Runway that TBFM used in its internal model. This was used only to assess TBFM accuracy.
  - "dap" – Departure airport
  - "apt" – Arrival airport
  - "typ" – Aircraft type (although this was later eliminated from modelling)
  - "mfx" – Arrival meter fix name as adapted in TBFM.
  - "gat" – Arrival gate name as adapted in TBFM.
  - "cfg" – The configuration, as listed in TBFM.
  - "scn" – Stream class name. This essentially tells TBFM which flights need to be separated from one another.

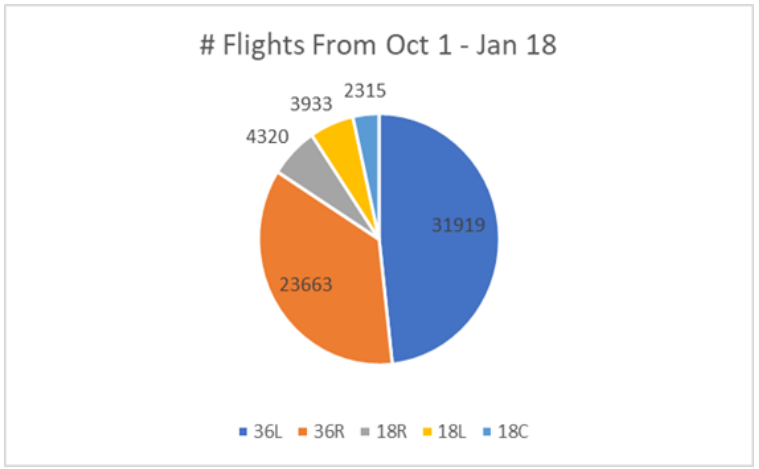
mfx	gat	scn	typ	eng	trw	rwy	match
ADOWN	SHINE	FILPZ_JET	A321/L	JET	18R	5	0
ADOWN	SHINE	FILPZ_JET	A321/L	JET	18R	5	0
BOATN	SHINE	PARQR_JE	C560/L	JET	18C	18R	0
ADOWN	SHINE	FILPZ_JET	A321/L	JET	18R	5	0
SUDSY	MAJIC	CHSLY_JET	B752/L	JET	18R	5	0
ADOWN	SHINE	FILPZ_JET	H/DC10/L	JET	36L	36L	1
FIBBR	CTF	STOCR_EA	C56X/L	JET	36R	36C	0
ADOWN	SHINE	FILPZ_JET	H/B763/L	JET	36L	36L	1
ADOWN	SHINE	FILPZ_JET	H/A306/L	JET	36L	36L	1

Here TBFM's runway (rwy- 36L) matches the target (trw- 36L)

Data Understanding

Data Understanding – Sample & Baseline Description

- CLT data was used given ATD-2 team's ability to verify accuracy (available at many other airports)
- Data was collected from Oct 1st 2019 through Jan 18th, 2020. Only rows with all required data elements properly populated were kept. With this filtering, **66,165** rows/flights remained for training and test.



Controller Runway	# Flights	% Flights
36L	31919	48.2
36R	23663	35.8
18R	4320	6.5
18L	3933	5.9
18C	2315	3.5
23	6	0.0

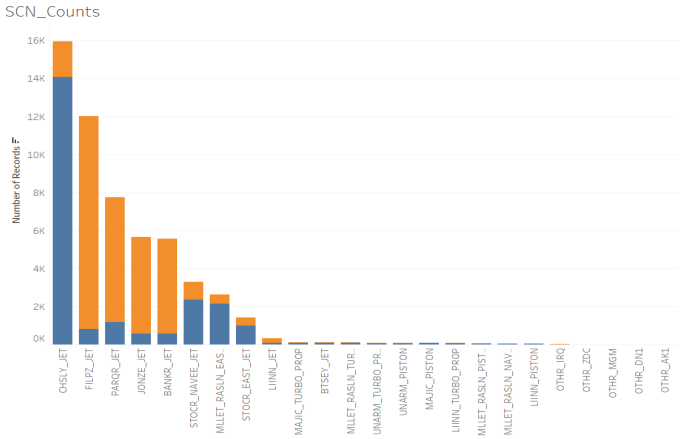


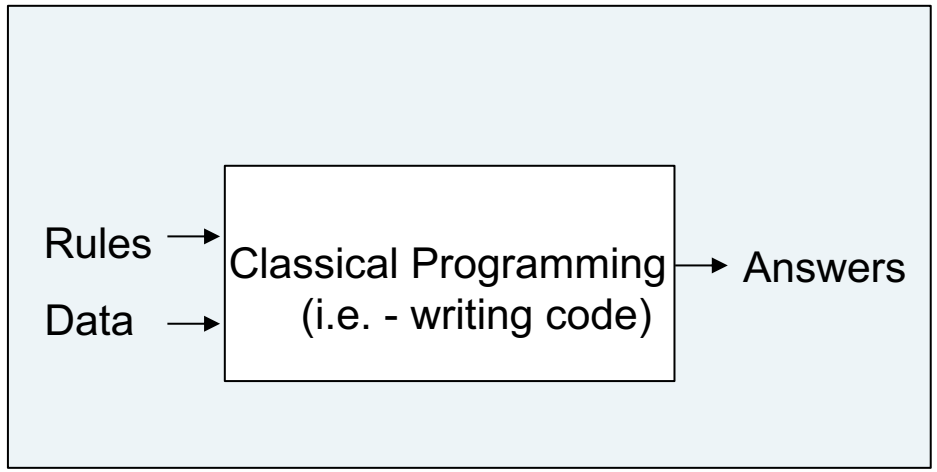
Tableau Visualization of SCN by Landed Runway

- Given the dominance of North landed runways (36L, 36R), the modelling was specifically developed to focus on this flow direction. This presumes that the configuration, or flow direction, would be passed into any algorithm that would seek to use this modelling.
  - With this decision, this becomes a 'binary classification' problem
- TBFM arrival runway prediction accuracy was approximately 70% accurate to these runways during this time.
  - This served as a baseline for the learner to attempt to beat

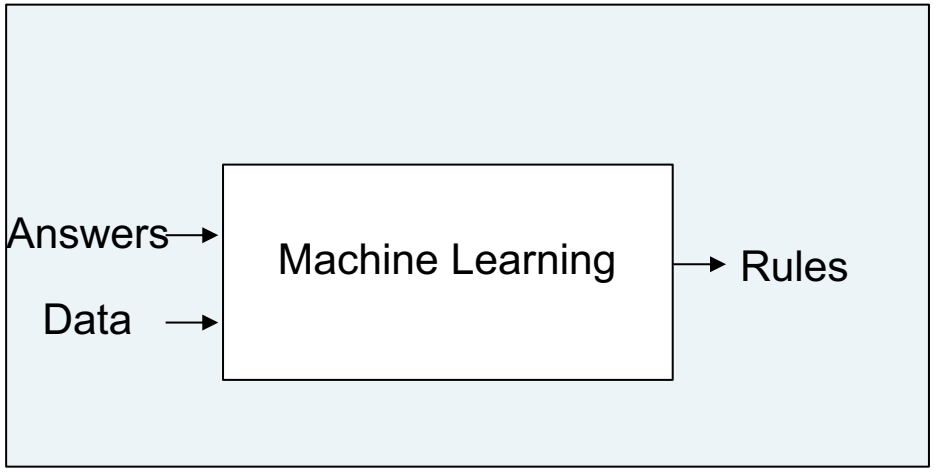
Data Understanding

How Machine Learning Differs from Classical Programming (50K Foot View)

Classical Programming



Machine Learning



The rules that machine learning automatically creates are applied to new data points to provide new answers.

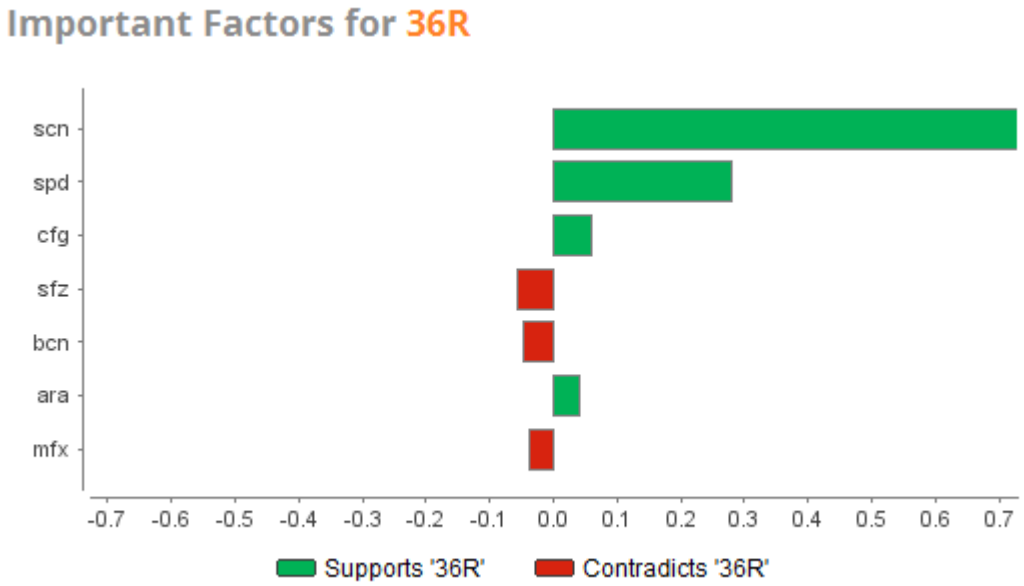
Input Attribute (Feature) Relative Importance

Feature Importance - Overall

Gradient Boosted Trees - Weights

Attribute	Weight
scn	0.352
spd	0.084
sfz	0.081
cfg	0.065
ara	0.045
tds	0.040
mfx	0.022
bcn	0.017
gat	0.017

Feature Importance – By Runway (target)



- Feature importance information from the models were used to continue the winnowing down (reduce dimensions) of the attributes used in the modelling to the most relevant, while also considering new features that would improve performance

Feature Engineering & Model Fitting

Develop Python Model and Validate

- Based on the promising initial results, a Light Gradient Boost (LGBoost) model was developed in python
  - LGB has been recently winning a number of the data science competitions due to its ability to evaluate a leaf of the tree without creating an entire branch (better performance)

- The steps were:
  - Read in the CSV file that came from previous steps into a Pandas data frame
  - Create two data frames. One for features, one for truth.
  - Create and store interim steps in ModelFamily wrapper class. More on this later.
  - Encode variables as required for the Light Gradient boost (LGB) model.
  - Split the dataset into train and test.
  - Specify initial hyperparameters and train the LGB model on the training dataset.
  - Use the trained model to generate new predictions on the test data.
  - Measure the performance of the test dataset.
  - When satisfied with performance of the model, store/save it for later use.

- N-fold validation is a commonly used technique to prevent “over fitting” and create a robust learner that will work well with data it has never seen

```
import lightgbm as lgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

```
##Convert categorical data using 'one hot' encoding
## This tends to be more robust to various models compared to int encoding
one_hot_scn=pd.get_dummies(arr_df.scn)
one_hot_mfx=pd.get_dummies(arr_df.mfx)
one_hot_eng=pd.get_dummies(arr_df.eng)
```

```
##Set paramaters
param = {'num_leaves':150, 'objective':'binary','max_depth':7,
         'learning_rate':.05,'max_bin':200}
param['metric'] = ['auc', 'binary_logloss']

#training the model using light gbm
num_round=50
start=datetime.now()
lgbm=lgb.train(param,train_data,num_round)
```

```
#converting probabilities into 0 or 1
for i in range(len(test_pred)):
    if test_pred[i]>=.5:
        test_pred[i]=1
    else:
        test_pred[i]=0

#calculating accuracy
accuracy_lgbm = accuracy_score(test_pred,y_test)
print(accuracy_lgbm)
```

Model Validation

Saving the Model for Later Use in Web Service

- The final model was saved/serialized for later use
  - In this case, we used ‘pickle’
  - This created a 0.5mb file on disk
- There are a number of different frameworks to use our new model (***as a service***).
- Deployment options need to consider many factors depending on how the service will be used
  - When using the model as a service, you do not need to load the model every request. It can be used efficiently to provide answers just like classical programming services.
  - In our case, we use python ‘flask’


```
##Ok, looks good. Before saving the model, train it on the entire dataset
train_data=lgb.Dataset(x,label=y)
lgbm=lgb.train(param,train_data,num_round)

##Serialize/Save the CLT LGB model to file
filename = 'clt_arr_runway_predictor.sav'
pickle.dump(lgbm, open(filename, 'wb'))
```

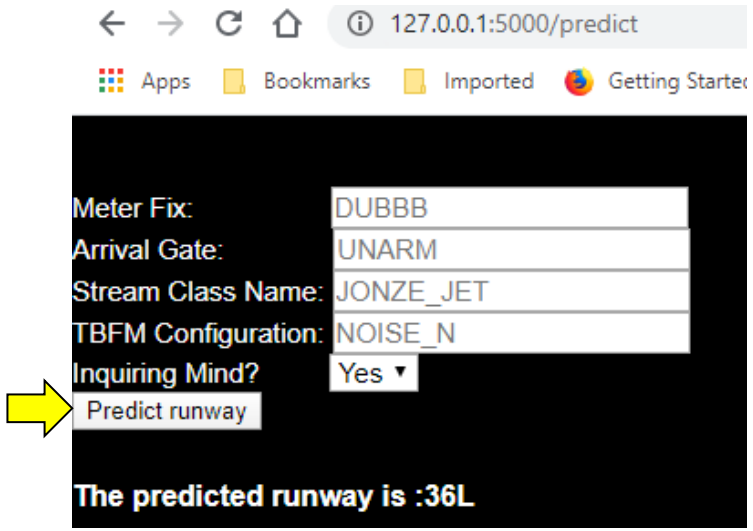
Open source python micro-framework to test services

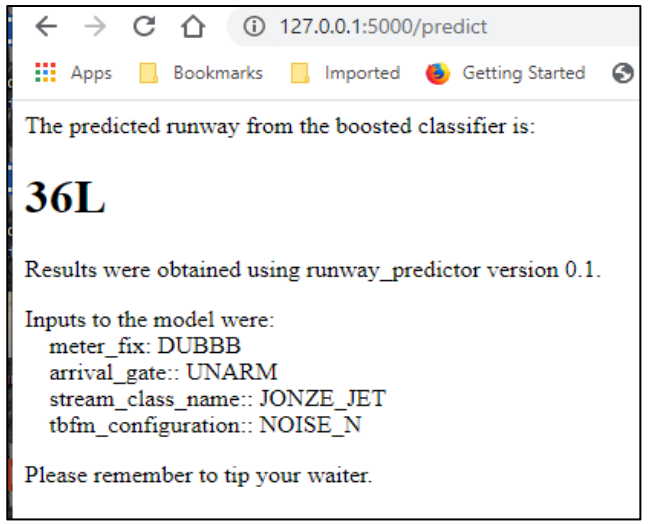
https://palletsprojects.com/p/flask/

Deployment to Real-time System

Evaluation of Model Performance

Testing Machine Learning as a Service (MLaaS)






```python
import pandas as pd
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
import ModelFamily
import lightgbm as lgb

##The following will be loaded and executed as soon as the service is started
app = Flask(__name__)
model_fam = pickle.load(open('arrival_runway_prediction_family.sav', 'rb'))
model=model_fam.models.get("clt_runway")
model_fam_name=model_fam.name
model_fam_version=model_fam.version

@app.route('/')
def home():
    return render_template('index.html')
```

- A simple web page was developed that allowed the user to enter the key inputs required by the model
  - This was just for test, likely different use in operations
- For operational deployment of python consider Heroku or other open source environments
  - For the time being, NASA team is likely to stick with python stack instead of predictive markup modeling language (PMML) given complexities/overhead

Deployment to Real-time System | **Evaluation of Model Performance**

Example 2 - Motivation and Operational Goal

Motivation

- At prior SWIFT meetings and other collaborative forums, operators have expressed the desire to find solutions for ***high and unpredictable delay*** from TBFM (APREQs)

- Knowledge of flights that have high delay can lead to:
  - Better understanding of the factors that influence this behaviour
  - Early information on flights that are likely to have high TBFM delay
  - Identification of procedural changes that could help flying public

Goals

- Assess how well the aviation community can estimate the size of TBFM APREQ delay
- Report the accuracy of the predictions provided
- Identify path to provide this information in near real-time (via service)
- Perform root cause analysis and/or recommended next steps to mitigate the problem

Identify Operational Goal

Focus on TBFM SWIM Data *plus 'Ready Time'*

- TBFM SWIM data shown in the prior learner was used as a starting point
  - Given the potential to predict TBFM assigned delay in near real-time, it was important to limit the dataset to those elements that are available prior to APREQ scheduling
  - Step 3 output from the python script was used to obtain all APREQs for a day and the unique TBFM identifiers.
  - This data was re-run through step1 output to only fill in data available up until scheduling

- TBFM delay data
  - NASA pulls down the TBFM system binary data from WJHTC every night
  - This data is in a TBFM proprietary form. It is translated to text.
  - The translated text is then processed for information.
  - The "ready time" from TBFM is a key data element to determining how much TBFM assigned delay a flight has

- TBFM Assigned Delay = *Scheduled Time of Departure from TBFM – Ready Time (in minutes)*

Select Initial Data Sources

Data Understanding - Description

- In addition to the elements earlier, the following TBFM SWIM data elements were used:
  - "ctm" – Coordination time. In our case for departures, that is TBFM's expected departure time from the airport.
  - "etm" – EDCT time, if it has one. We translated this into a yes/no hasEDCT boolean feature.
  - "scnname" – TBFM "con" message - Stream class name that is used with ssd (below) to determine MIT
  - "ssd" – TBFM "con" message – Super Stream Class distance

- Important note is the "scnname" and "ssd" from TBFM "con" gives the MIT separation for this flight
  - This needs to be synchronized/matched with the "air" messages, and becomes a new feature in our dataset

25 MIT over LGA_DYL

Features from "air" messages

dap	apt	mfx	scn	typ	spd	ara	ctm	etm
CVG	EWR	DJB	EWR_DJB	E170/L	431	31000	2019-11-02T00:30:00Z	2019-11-02T01:00:00Z
BOS	EWR	SHF	EWR_SHF	B739/L	354	16000	2019-11-02T00:00:00Z	2019-11-02T01:09:00Z
DTW	EWR	MILTON	EWR_MILTON	BCS1/L	456	37000	2019-11-02T01:22:00Z	2019-11-02T01:14:00Z
BOS	EWR	SHF	EWR_SHF	E190/L	393	16000	2019-11-02T00:20:00Z	2019-11-02T01:20:00Z
BWI	EWR	DYL	EWR_DYL	B737/L	338	23000	2019-11-02T01:59:37Z	2019-11-02T01:51:00Z
CMH	EWR	MILTON	EWR_MILTON	E170/L	425	31000	2019-11-02T00:00:00Z	2019-11-02T02:21:00Z
PIT	EWR	MILTON	EWR_MILTON	E75L/L	433	27000	2019-11-02T01:59:14Z	2019-11-02T02:23:00Z
CMH	EWR	MILTON	EWR_MILTON	E170/L	446	33000	2019-11-02T02:24:00Z	2019-11-02T02:26:00Z
DCA	EWR	DYL	EWR_DYL	E170/L	376	17000	2019-11-02T02:52:24Z	2019-11-02T03:02:00Z
ROC	ORD	PTN/MKG	ORD_PTN/MKG	A320/L	460	34000	2019-11-01T11:50:55Z	
CVG	ORD	MZZ	ORD_MZZ	H/B762/Z	483	28000	2019-11-01T10:58:00Z	
CVG	DFW	PXV	DFW_PXV	H/B763/L	467	32000	2019-11-01T10:55:00Z	
TYS	ORD	MZZ	ORD_MZZ	CRJ2/L	432	30000	2019-11-01T21:54:32Z	
YYZ	ORD	PTN/MKG	ORD_PTN/MKG	E75S/L	455	32000	2019-11-01T10:44:22Z	
CLE	EWR	MILTON	EWR_MILTON	B738/L	443	37000	2019-11-01T11:43:57Z	
PIT	MDW	BAGELMP	MDW_BAGELMP	B737/L	461	32000	2019-11-01T09:44:44Z	

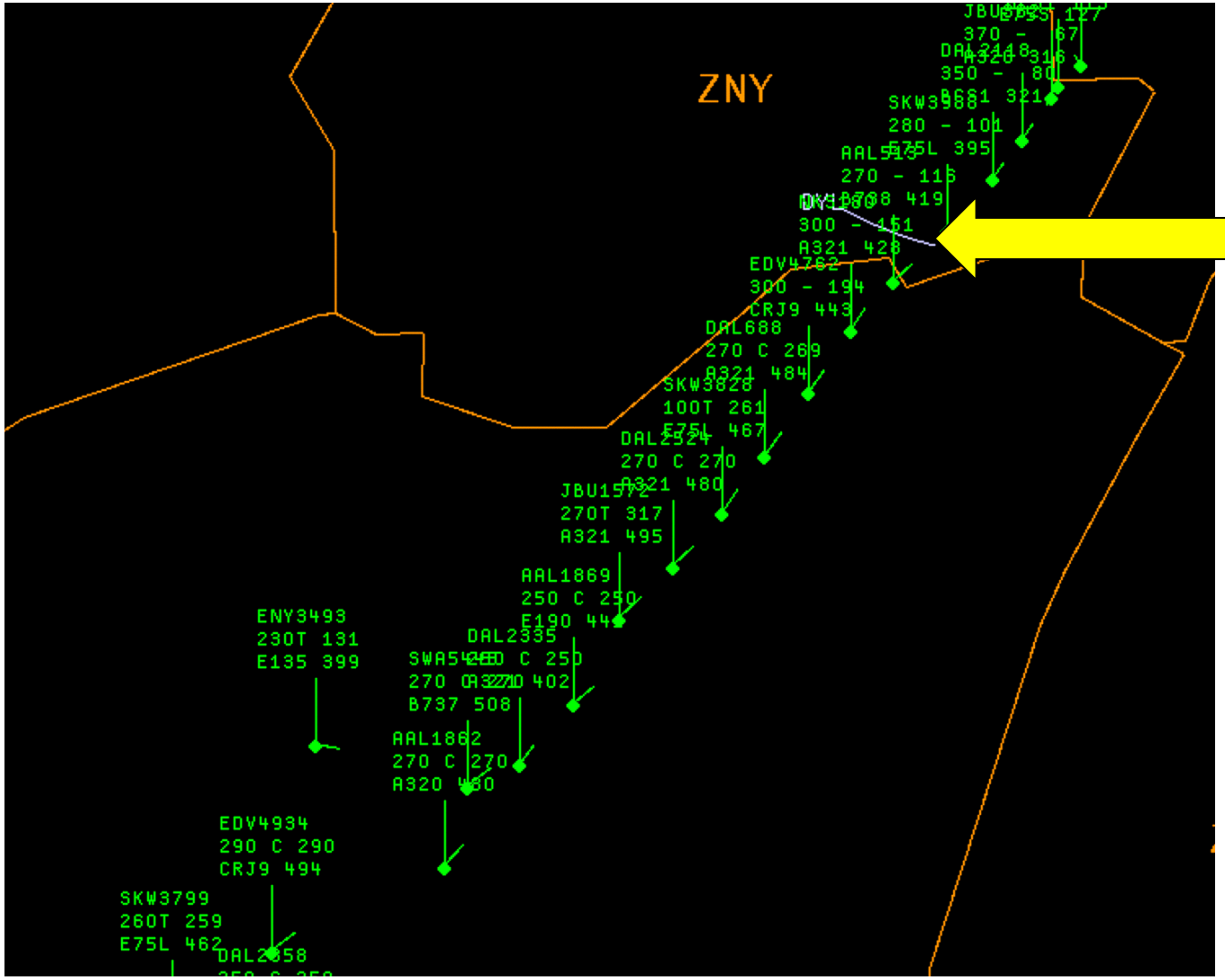
Derived feature from "con" messages


```
<ssc
sscType="NEW"><ssn>23</ssn><sscname>LGA_DYL</sscname><ssd>25.0</ssd><s
smin>0</ssmin><sstyp>SSC_MILES_IN_TRAIL</sstyp><scls><scl
sclType="NEW"><scname>LGA_DYL</scname><scmre>DYL</scmre></scl></scls><
ccs><cc
ccType="NEW"><apt>BWI</apt><apreq>SEMI</apreq><sch>ACCEPT</sch></cc><c
c
ccType="NEW"><apt>CLT</apt><apreq>SEMI</apreq><sch>ACCEPT</sch></cc><c
c
ccType="NEW"><apt>DCA</apt><apreq>SEMI</apreq><sch>ACCEPT</sch></cc><c
c
ccType="NEW"><apt>IAD</apt><apreq>SEMI</apreq><sch>ACCEPT</sch></cc><c
c
ccType="NEW"><apt>RDU</apt><apreq>SEMI</apreq><sch>ACCEPT</sch></cc><c
c
ccType="NEW"><apt>RIC</apt><apreq>SEMI</apreq><sch>ACCEPT</sch></cc></
ccs></ssc><ssc
sscType="NEW"><ssn>24</ssn><sscname>JFK_HOG</sscname><ssd>30.0</ssd><s
smin>0</ssmin><sstyp>SSC_MILES_IN_TRAIL</sstyp><scls><scl
sclType="NEW"><scname>JFK_HOG</scname><scmre>HOG</scmre></scl></scls><
ccs><cc ccType="NEW"><apt>BWI</apt><apreq>SEMI</apreq><sch>ACCEPT
```


Data Understanding

Importance of Super Stream Classes in TBFM

- LGA_DYL super stream class in TBFM is among the busiest in the county



LGA_DYL Metering Arc

Super stream classes have been adapted across the country in TBFM instances.

They define how flights are grouped coming out of, and leading into airports. Often, a MIT separation is used at these points to regulate the flow of traffic into important airspace/airport locations.

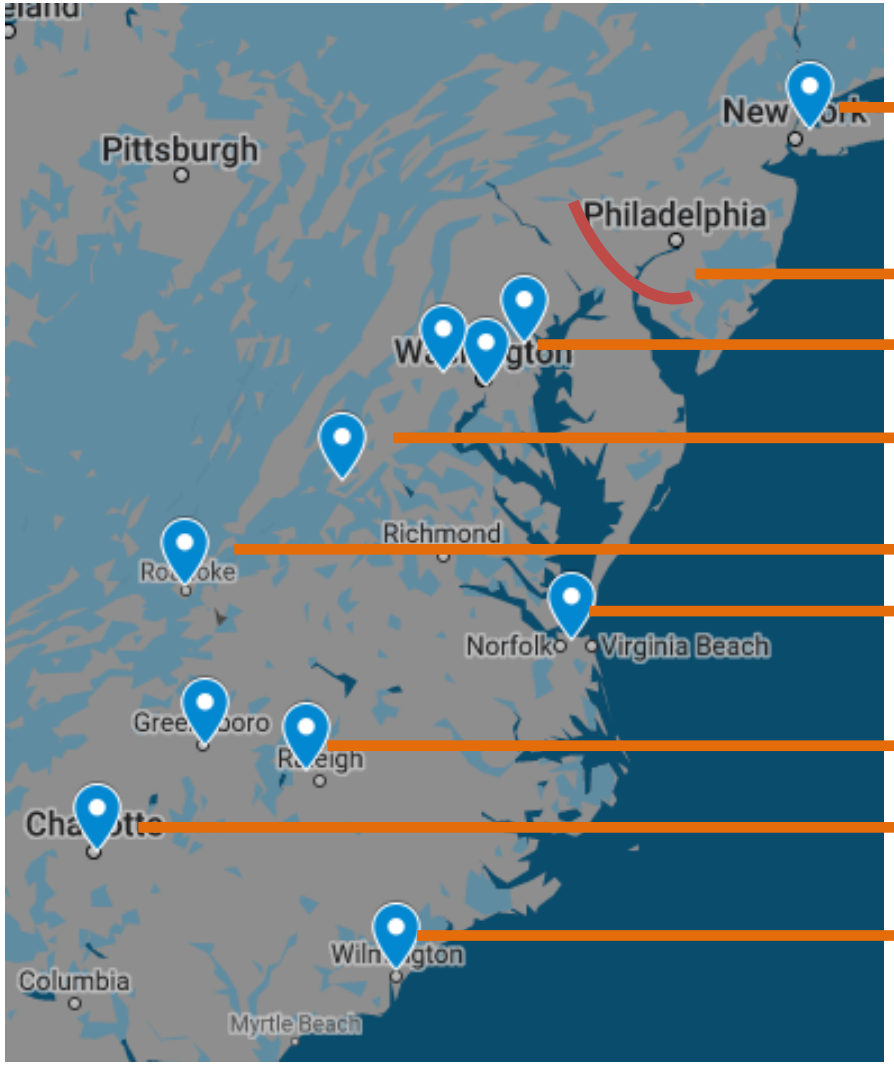
Data Understanding

Goldilocks Zone – Close in Airports are Hot- Have Higher Delay

Estimated 90th Percentile delay by Origin into LGA_DYL for all of 2019

Origin	90th % Delay (min)	Number Apreqs	Distance (nm)
BWI	39	89	132.6
DCA	26	5717	164.9
IAD	33	1110	188.4
RIC	22	2738	213.9
CHO	18	1316	253.9
ORF	17	2386	270.6
RDU	17	4420	329.8
ROA	12	288	347.0
GSO	15	2155	389.6
ILM	15	781	402.7
CLT	8	5036	451.9

Better



LGA Airport

LGA_DYL Metering Arc

32 minute delays (90th %)

18 minute delays (90th %)

12 minute delays (90th %)

17 minute delays (90th %)

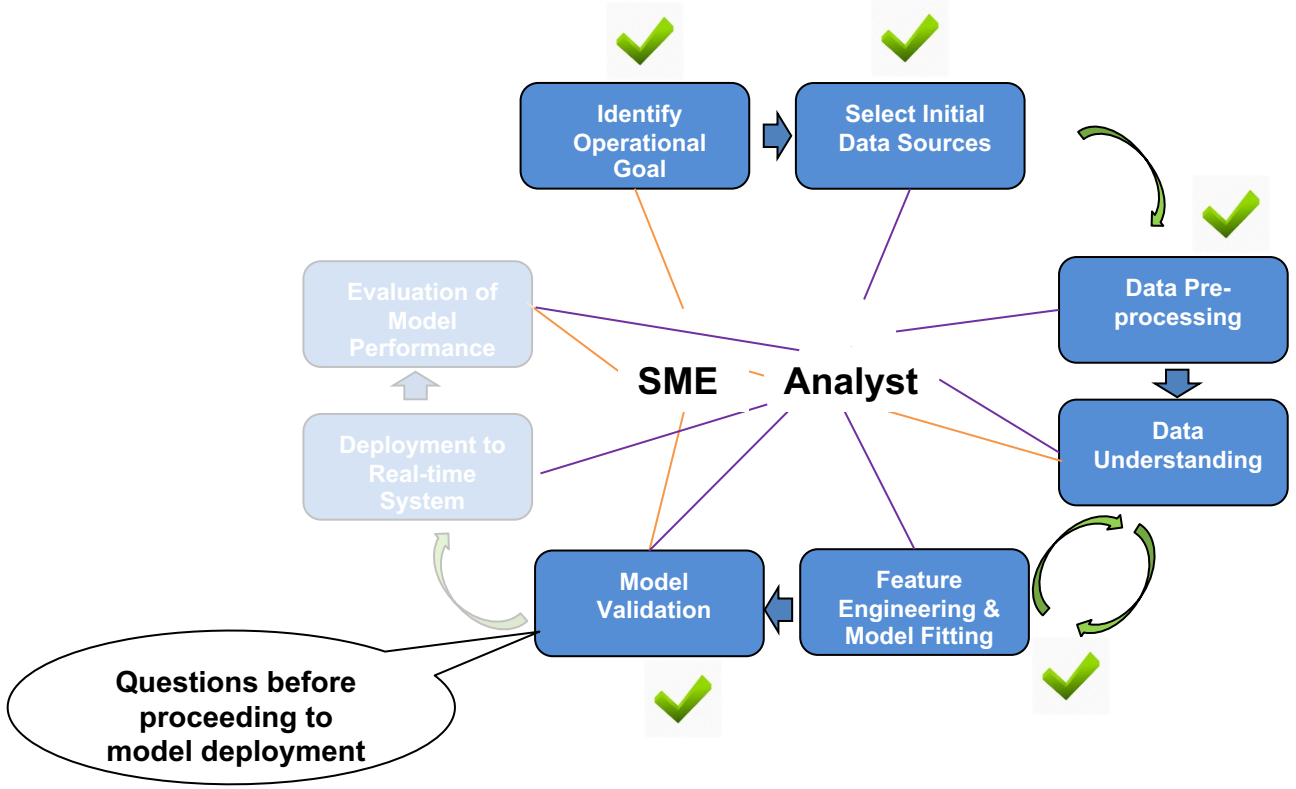
16 minute delays (90th %)

8 minute delays (90th %)

15 minute delays (90th %)

Data Understanding

Questions Before Proceeding



- Questions to community before proceeding:
  - Is a model that lets users know the predicted size and variance of TBFM assigned delay (by city pairs and stream class) valuable to the community even if the standard deviation is extremely large (e.g. 10 minutes)? Or do the predictions need to be more accurate (lower std) to be usable?
  - Can the community use data like that shown in this process to create a science/evidence-based definition of the “high TBFM delay” problem to help us focus our resources on the right problem?

Summary

- The aviation industry's biggest barrier to more fully benefitting from ML is ***constructing our problems in the right format*** to take advantage of ML breakthroughs that already exist (and are growing every day). This includes:
  - Creating an operationally meaningful problem as a ML challenge
  - Comporting our data into clean **datasets** with solid "ground truth" data
  - Creating initial **benchmarks** that ML experts can beat

- Can we predict the likely landing runway with high certainty? If so, how good are the predictions?
  - Yes, initial indications are very promising. The current day TBFM arrival runway prediction in this example was on the order of 70% accurate, whereas the gradient boost machine learning models achieve close to 90% accuracy. This is a significant improvement.

- Can models developed with machine learning be deployed and leveraged in near real-time?
  - Yes. In this example we deployed the model that was trained in post ops and leveraged it in a web service. Once the model is deployed as a service, near real-time data can be used to call it and get results from the model predictions.

- More specificity is needed from the community on the "high TBFM delay" operational problem mentioned in prior SWIFT meetings
  - This will help evidence-based solutions and can also be used to measure the benefits once solutions are deployed

- These problems would benefit from more frequent engagement by the aviation analytical community to prevent stovepipe solutions, ensure truth in reporting and leverage lessons learned across teams

Backup

- Backup

Winnowing Down the Columns/Features Used

Correlation Matrix of Potential Columns/Features to Evaluate Multicollinearity

	dap	mfx	gat	bcn	scn	typ	eng	spd	trw	sfz	rfz	ara	tds	cfx	est	a10	tcr	dfx	sfx	oma	ooa	o3a	ina	sus	man	cfg
dap	1	-0.00145	-0.03543	-0.07917	-0.04536	-0.05964	-0.00527	0.041068	0.010874	0.005316	-0.0065	0.00686	0.021061	0.00216	-0.00075	0.939629	0.630822	-0.0662	-0.0662	-0.03677	0.04478	0.087574	0.018306	-0.00944	0.015639	0.000765
mfx	-0.00145	1	-0.61734	-0.14466	-0.2768	-0.09103	0.039289	-0.05126	0.63309	0.003053	0.022903	0.005807	-0.01988	-0.21168	-0.2053	-0.01052	0.01601	-0.28961	-0.28961	0.355071	-0.46876	-0.14423	-0.0751	0.008983	0.012157	-0.01847
gat	-0.03543	-0.61734	1	0.149252	-0.21986	0.043135	-0.0246	0.099978	-0.67019	0.014579	-0.01275	0.190465	0.026502	0.157427	0.108977	-0.03924	0.013904	-0.19017	-0.19017	-0.01831	0.737105	0.60595	0.110905	-0.00259	-0.01166	0.03186
bcn	-0.07917	-0.14466	0.149252	1	0.082177	0.076444	0.01671	-0.04785	-0.10258	0.018138	0.005441	-0.00598	-0.0138	0.064384	-0.06876	-0.0574	-0.03162	0.087684	0.087684	0.038655	0.008395	0.053181	-0.04337	0.000566	-0.00194	-0.00364
scn	-0.04536	-0.2768	-0.21986	0.082177	1	0.025861	0.087732	-0.08029	-0.01274	-0.00237	0.003443	-0.15745	0.004668	-0.12774	0.150102	-0.01901	-0.10654	0.889123	0.889123	0.126792	-0.18135	-0.13053	-0.0536	0.005412	0.017104	-0.01268
typ	-0.05964	-0.09103	0.043135	0.076444	0.025861	1	0.062898	-0.24457	0.022719	-0.01226	0.002843	-0.0419	-0.01172	0.090266	-0.23045	-0.07153	-0.00372	0.061609	0.061609	-0.16052	-0.10963	-0.00898	-0.09471	0.01646	-0.01646	-0.00406
eng	-0.00527	0.039289	-0.0246	0.01671	0.087732	0.062898	1	-0.39034	0.118461	-0.03428	0.004122	0.151912	-0.06336	-0.00168	-0.00783	-0.00685	-0.00714	0.042251	0.042251	-0.18364	-0.12819	-0.06554	-0.11313	0.092118	0.012898	0.01105
spd	0.041068	-0.05126	0.099978	-0.04785	-0.08029	-0.24457	-0.39034	1	-0.13779	0.033595	-0.00969	0.009892	0.045927	0.01136	0.100406	0.021478	0.043368	-0.08328	-0.08328	0.271311	0.275261	0.165347	0.293994	-0.06926	9.34E-05	-0.00856
trw	0.010874	0.63309	-0.67019	-0.10258	-0.01274	0.022719	0.118461	-0.13779	1	-0.07459	0.010673	-0.04368	-0.06314	-0.15881	-0.13496	0.005074	-0.00161	-0.07591	-0.07591	0.153873	-0.54487	-0.38768	-0.08476	0.024496	0.002421	-0.02122
sfz	0.005316	0.003053	0.014579	0.018138	-0.00237	-0.01226	-0.03428	0.033595	-0.07459	1	0.031748	-0.00399	0.0253	-0.01374	-0.10927	0.001418	-0.01115	-0.00436	-0.00436	0.040926	0.037243	0.039809	0.119295	-0.00518	0.068094	0.11846
rfz	-0.0065	0.022903	-0.01275	0.005441	0.003443	0.002843	0.004122	-0.00969	0.010673	0.031748	1	-0.02644	0.002398	-0.01044	-0.02685	-0.00793	-0.00123	0.005047	0.005047	0.001459	-0.01961	-0.01352	-0.02097	0.031245	0.097663	0.003018
ara	0.00686	0.005807	0.190465	-0.00598	-0.15745	-0.0419	0.151912	0.009892	-0.04368	-0.00399	-0.02644	1	-0.03953	0.009266	0.08378	-0.00479	0.017537	-0.17273	-0.17273	0.144486	0.21842	0.194116	0.115919	0.019563	0.006438	-0.07133
tds	0.021061	-0.01988	0.026502	-0.0138	0.004668	-0.01172	-0.06336	0.045927	-0.06314	0.0253	0.002398	-0.03953	1	0.002363	-0.00601	0.024227	0.015447	0.00917	0.00917	-0.01066	0.025506	0.045873	0.001294	0.001263	0.003459	0.0307
cfx	0.00216	-0.21168	0.157427	0.064384	-0.12774	0.090266	-0.00168	0.01136	-0.15881	-0.01374	-0.01044	0.009266	0.002363	1	0.007487	0.007572	0.020511	-0.07965	-0.07965	-0.2451	0.105652	0.148035	-0.01915	0.001311	-0.00444	0.010108
est	-0.00075	-0.2053	0.108977	-0.06876	0.150102	-0.23045	-0.00783	0.100406	-0.13496	-0.10927	-0.02685	0.08378	-0.00601	0.007487	1	-0.00368	-0.05034	0.125765	0.125765	0.011832	0.228542	0.133347	0.141419	-0.01256	-0.00308	-0.03212
a10	0.939629	-0.01052	-0.03924	-0.0574	-0.01901	-0.07153	-0.00685	0.021478	0.005074	0.001418	-0.00793	-0.00479	0.024227	0.007572	-0.00368	1	0.643958	-0.03906	-0.03906	-0.06386	0.024247	0.102944	0.014117	-0.0089	0.013906	0.0036
tcr	0.630822	0.01601	0.013904	-0.03162	-0.10654	-0.00372	-0.00714	0.043368	-0.00161	-0.01115	-0.00123	0.017537	0.015447	0.020511	-0.05034	0.643958	1	-0.11257	-0.11257	-0.0473	0.035007	0.09045	0.025497	-0.00414	0.002124	-0.00122
dfx	-0.0662	-0.28961	-0.19017	0.087684	0.889123	0.061609	0.042251	-0.08328	-0.07591	-0.00436	0.005047	-0.17273	0.00917	-0.07965	0.125765	-0.03906	-0.11257	1	1	-0.01382	-0.0734	-0.05556	-0.10581	0.007529	0.028378	-0.01316
sfx	-0.0662	-0.28961	-0.19017	0.087684	0.889123	0.061609	0.042251	-0.08328	-0.07591	-0.00436	0.005047	-0.17273	0.00917	-0.07965	0.125765	-0.03906	-0.11257	1	1	-0.01382	-0.0734	-0.05556	-0.10581	0.007529	0.028378	-0.01316
oma	-0.03677	0.355071	-0.01831	0.038655	0.126792	-0.16052	-0.18364	0.271311	0.153873	0.040926	0.001459	0.144486	-0.01066	-0.2451	0.011832	-0.06386	-0.0473	-0.01382	-0.01382	1	0.068907	0.194124	0.28976	-0.03269	-0.00408	-0.01602
ooa	0.04478	-0.46876	0.737105	0.008395	-0.18135	-0.10963	-0.12819	0.275261	-0.54487	0.037243	-0.01961	0.21842	0.025506	0.105652	0.228542	0.024247	0.035007	-0.0734	-0.0734	0.068907	1	0.599109	0.277758	-0.02204	0.014754	0.040703
o3a	0.087574	-0.14423	0.60595	0.053181	-0.13053	-0.00898	-0.06554	0.165347	-0.38768	0.039809	-0.01352	0.194116	0.045873	0.148035	0.133347	0.102944	0.09045	-0.05556	-0.05556	0.194124	0.599109	1	0.181792	-0.01457	0.003687	0.030086
ina	0.018306	-0.0751	0.110905	-0.04337	-0.0536	-0.09471	-0.11313	0.293994	-0.08476	0.119295	-0.02097	0.115919	0.001294	-0.01915	0.141419	0.014117	0.025497	-0.10581	-0.10581	0.28976	0.277758	0.181792	1	-0.02792	-0.01066	0.004117
sus	-0.00944	0.008983	-0.00259	0.000566	0.005412	0.01646	0.092118	-0.06926	0.024496	-0.00518	0.031245	0.019563	0.001263	0.001311	-0.01256	-0.0089	-0.00414	0.007529	0.007529	-0.03269	-0.02204	-0.01457	-0.02792	1	0.014879	-0.00134
man	0.015639	0.012157	-0.01166	-0.00194	0.017104	-0.01646	0.012898	9.34E-05	0.002421	0.068094	0.097663	0.006438	0.003459	-0.00444	-0.00308	0.013906	0.002124	0.028378	0.028378	-0.00408	0.014754	0.003687	-0.01066	0.014879	1	-0.00735
cfg	0.000765	-0.01847	0.03186	-0.00364	-0.01268	-0.00406	0.01105	-0.00856	-0.02122	0.11846	0.003018	-0.07133	0.0307	0.010108	-0.03212	0.0036	-0.00122	-0.01316	-0.01316	-0.01602	0.040703	0.030086	0.004117	-0.00134	-0.00735	1

Run early models with/without certain features to determine if they are beneficial/required

Vars ->	dap	mfx	gat	bcn	scn	typ	spd	ara	tds	cfx	est	a10	tcr	oma	ooa	o3a	ina	cfg	ROC
	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0.8981
	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x	0.9011

Here the receiver operating characteristics (ROC) area under curve (AUC) shows that remove the filed flight plan (A10) yields equivalent or better performance.

- Some of the notably high correlations are:
- A10 vs DAP => .939629
- TCR vs DAP => .630822
- GAT vs MFX => -.61734
- OOA vs GAT => .737105
- O3A vs GAT => .60595
- DFX vs SCN => .889123
- **SFX vs SCN => .889123**
- TCR vs A10 => .643958
- SFX vs DFX => 1
- O3A vs OOA => .599109

Interestingly, the stream class name (SCN) contains useful information that allows removal of other columns without any loss in performance

Feature Engineering & Model Fitting

TBFM APREQs by Super Stream Class

- Using the same 110 day sample mentioned earlier, the table below lists the top 10 SSC across the NAS with flights subject to APREQ, **by count**

Super Stream Class	Count APREQs	Notes	Arrival %	EDC %	Top Airports Impacted
SE_JET	12197	Into ATL over JJEDI meter fix	100	0	MCO, JAX, CHS,SAV, CAE
LAX_FIM_JETS	12150	Into LAX over LOSHNW, SYMON	30.3	69.6	SFO, SJC, SMF,OAK,RNO
CHSLY_JET	12055	Into CLT over MAJIC	100	0	BWI,IAD,RDU,GSO,DCA
OZZZI_ZDC_JET	8850	Into ATL over OZZZI	100	0	DCA,BWI,RDU,IAD,RIC
FILPZ_JET	8463	Into CLT over SHINE	100	0	TYS,CVG,BNA,IND,AVL
SW_ZME_JET	7966	Into ATL over HOBTT	100	0	BHM,JAN,TPA,MGM,TLH
LAS_CLARR_JETS	7889	Into LAS over CLARR	100	0	LAX,BUR,SNA,SAN,VNY
ATL_J48_J75	7816	Into ATL over J48_J75	0	100	LGA,EWR,PHL,JFK,HPN
LGA_DYL	7536	Into LGA over DYY	0	100	DCA,CLT,RDU,RIC,ORF
CHPPR_JET	7216	Into ATL over CHPPR	100	0	IND,BNA,CHA,SDF,CVG

- Stream classes are what cause delay pass back, not necessarily city-pairs.
- This is just count, what about the delay size and variance?

Data Understanding

Feature Importance and Analytical Lessons

Feature Importance

- Approx. 20 features (not shown here) were developed to achieve greater predictability of TBFM delay
  - The results of this model yielded an overall accuracy of around 6.5 min error and 6.8 min standard deviation
  - Site specific (city pair) predictive accuracy can be as good as 3 min (error and standard deviation) or as high at 10 min
  - The machine learning regressor improved these predictions by 10-20% over any one statistical view

- Important lessons learned were:
  - Analysis of this problem benefits from grouping origin, destination and TBFM stream class name
  - The MIT in use (super stream class separation) is the most important attribute with predictive power/lift
  - Another important attribute is “what time of day do you need the stream class resource” (15 minute bin)

Feature Engineering & Model Fitting

Initial Models to Assess How Well Commonly Used Techniques Will Work with this Dataset

- 8 initial binary classification models were evaluated on the dataset
  - Naïve Bayes
  - Linear regression Model
  - Logistic regression
  - Fast Large Margin
  - **Deep Learning**
  - Decision Tree
  - Random Forest
  - **Gradient Boosted Trees**

- Gradient boosted trees had the highest accuracy and lowest variance
  - This was without any hyperparameter tuning (roughly out-of-the-box models)
  - While other models may appear to be close (e.g. within 1%, their performance on other key metrics beyond accuracy underperformed these two learners)

Feature Engineering & Model Fitting