Flight Delay Prediction
Phase 5 Presentation

261 Final Project
Summer 2023
Section 5
Team 1

Kisha Kim
kisha.kim@berkeley.edu
Lead in Phase 1
Plan the Project

Chase Madson
chase_madson@berkeley.edu
Lead in Phase 2
Pipelines and Baselines

Eric Danforth
edanforth85@berkeley.edu
Lead in Phase 3
Improve our Baseline

Jess Stockham
jhsmith@berkeley.edu
Lead in Phase 4
Find Optimal Algorithm

Business Case and Developing a Model

Background:

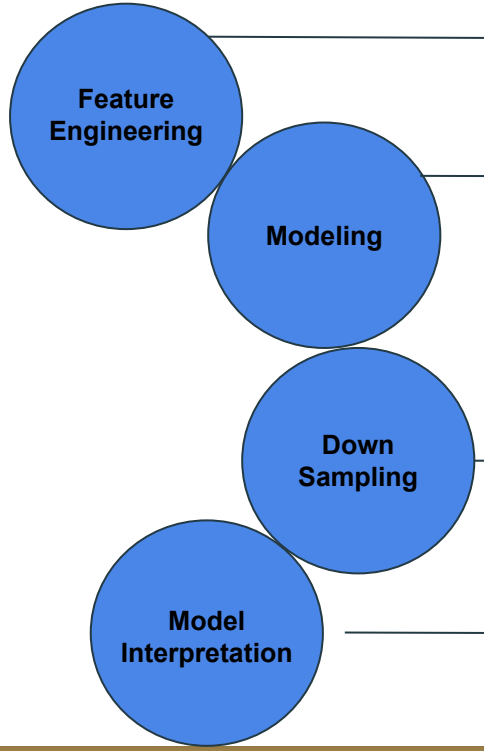
On-time Performance (OTP) for air travel has been in a state of steady decline. Airline industry expects to be profitable for the first time since the start of the pandemic in 2023. The losses that's due to flight delay has been steadily rising per the FAA from $19.2B in 2012 to estimated $33B in 2019.

Problem Statement:

The goal of this project is to develop a machine learning model that accurately predicts flight delays based on historical flight, airport station, and weather data spanning five years from 2015-2019 in the United States.

- **Delays** are defined as when flight takes off 15 minutes or more later than the predicted departure time.
- **Classification problem to predict the delay.** This is a **label,** where **delay=1** indicates a delay, **delay=0** as on time.
- **Per the business problem**, our **evaluation metric is** F-Beta that prioritizes recall (beta=2). We want to error on the side of predicting more flight delays and being wrong (focus on recall over precision)
- **Data**: Department of Transportation flights & Weather NOAA (pre-joined)

Our innovative approach



Graph-base Features

Created sophisticated graph features to represent each airport's strategic importance in the flight network

Modeling Techniques

- Used a LightGBM package to created Gradient Boosted Trees efficiently in the big data world
- Applied deep learning methods (neural networks)

Downsampling to address class imbalance

Initial data had ~80% of on-time flight

SHAP Value

Understand relative feature impact to the output model

Baseline Model = Random Forest

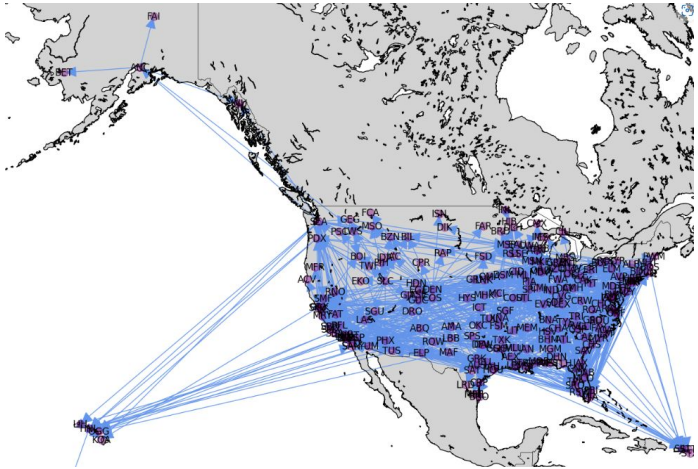
Estimator	Dataset	Key Parameters (defaults)	F-Beta (beta=2) Metric
Logistic Regression	Jan-Mar 2015 with 3 CV Folds	maxIter=100, regParam=0.0, elasticNetParam = 0.0	3.3%
Random Forest (BASELINE)	Jan-Mar 2015 with 3 CV Folds	maxDepth=5, numTrees=20	0.0%

Feature Families (700 columns total)

- **Numeric (2)**
  - Elevation
  - Distance
- **Categorical (7 columns, but 600+ w/ OHE)**
  - Day of Week
  - Month
  - Carrier
  - Origin
  - Destination
  - Origin_Type
  - Destination_Type

- Result is equivalent to the naive baseline that assumes all flights are on-time
- Expected random forest to outperform logistic regression as we refine the model.

Engineering New Features

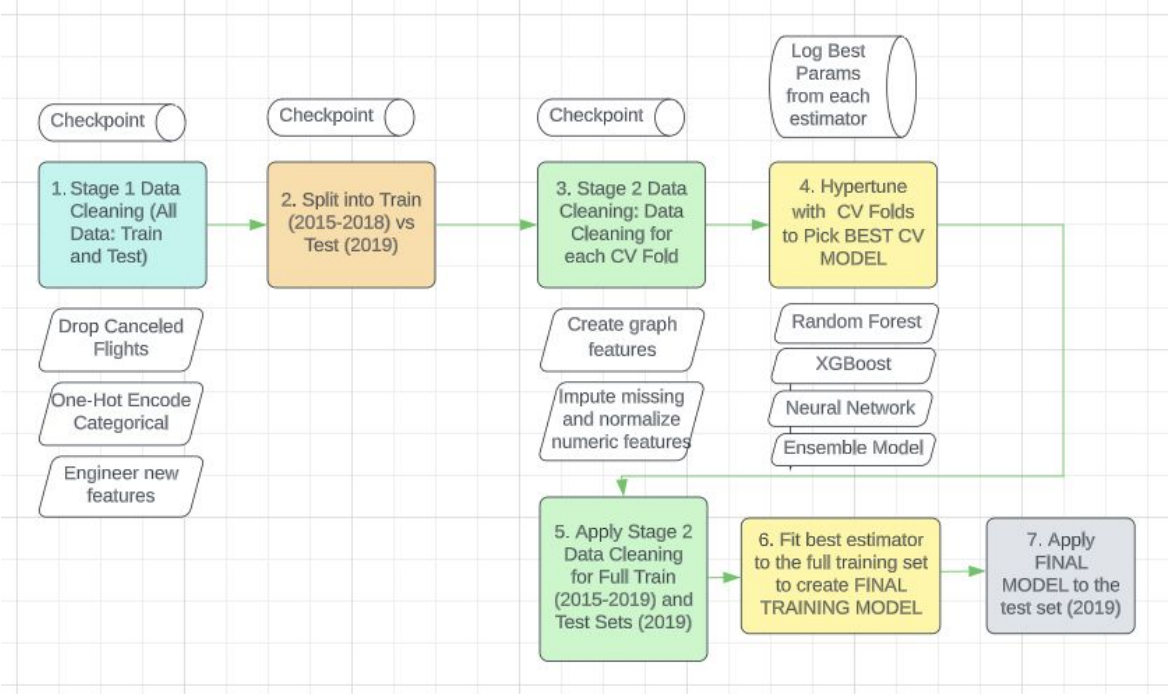


airport_iata_code	pagerank
MDW	16.667919932832607
DAL	12.015323092393754
TTN	6.727229976249611
UST	6.359073234678428
MAF	5.783570843374581
AMA	5.759743932028508
MHT	5.5422490941052605
OAK	3.051454431912101
CVG	2.616707398061823
MSP	2.2573752284359467

New Features Created To-Date:

- **Categorical:**
  - Holidays and holiday-adjacent indicators
  - First flight of the day indicator
- **Graph (numeric)**
  - PageRank
  - Triangle count
  - LPA (dropped because too compute intensive)
- **Lagged Time-Related Outcomes (numeric)**
  - Average Flight Delay Duration (from prior day)
  - Average Flight Duration by Flight Number (from prior day)
  - Expected Airport Congestion
  - Departure delay (in minutes) of its preceding flight

Data Pipeline to Minimize Data Leakage



Model Refinement: Hypertuning

- **Used Cross-validation Dataset Folds**
  - Reduces the risk that you are over-fitting to your particular training dataset
  - Blocked time-series splits over 2015-2018
  - Train: 200 days, val: 92
  - Used most recent 2 folds for hypertuning

- **Hypertuning** is trying different sets of parameters for your estimators to see what produces the best metric (f-beta)
  - Used baysean approach to hypertuning
  - Algorithm learn the best parameters as you perform more trials

Hypertuning Experiments (with new features)

#	Estimator	Dataset	Best Hyperparameters	Trials	Best F-Beta (beta=2) Metric
1	**Random Forest**	2017-2018 (2-folds) train: 200 days val: 92	maxDepth=12 numTrees=140	8	40.1%
2	**LightGBM**	2017-2018 (2-folds) train: 200 days val: 92	maxDepth=10 min_child_weight=2 min_data_in_leaf=1000 subsample=0.8 reg_alpha=1	10	28.7%
3	**Neural Network**	2017-2018 (2-folds) train: 200 days val: 92	layers=[81, 8, 4, 2]	2	43.4%

Feature Families (81 features total)

- **Numeric (10)**
  - Elevation
  - Distance
  - 4 graph features
  - 4 time-lagged features
- **Categorical (9 columns, but 71 w/ OHE)**
  - Day of Week
  - Month
  - Carrier
  - Origin_Type
  - Destination_Type
  - CRS_DEP_BUCKET
  - Holiday, holiday-adjacent
  - first_flight

Final Model Runs: Full Training Dataset

#	Estimator	Dataset	Hyperparameters	F-Beta (beta=2) Metric	AUC Metric
1	**Random Forest 1**	2015-2018 Train	maxDepth=12 numTrees=140	40.5%	50.3%
2	**Neural Network 1**	2015-2018 Train	layers=[81, 8, 4, 2]	40.5%*	66.9%
3	**Neural Network 2**	2015-2018 Train	layers=[81, 32, 16, 4, 2]	40.3%	67.0%
4	**Random Forest 2**	DOWNSAMPLED 2015-2018 Train	maxDepth=12 numTrees=140	60.4%	72.0%
5	**Neural Network 3**	DOWNSAMPLED 2015-2018 Train	layers=[81, 8, 4, 2]	63.1%	71.8%

Feature Families (81 features total)

- **Numeric (10)**
  - Elevation
  - Distance
  - 4 graph features
  - 4 time-lagged features
- **Categorical (9 features, but 71 w/ OHE)**
  - Day of Week
  - Month
  - Carrier
  - Origin_Type
  - Destination_Type
  - CRS_DEP_BUCKET
  - Holiday, holiday-adjacent
  - first_flight

Downsampling

Label	Training Dataset Row Count (original) 2015-2018	Training Dataset After Downsampling Row Count 2015-2018	Test Dataset Row Count 2019
No Delay	19,607,943	4,312,859	5,905,453
Delay	4,312,859	4,312,859	1,353,702
Total	**23,920,802**	**8,625,718**	**7,259,155**

Downsampled training dataset is 34% of the size of the original training dataset

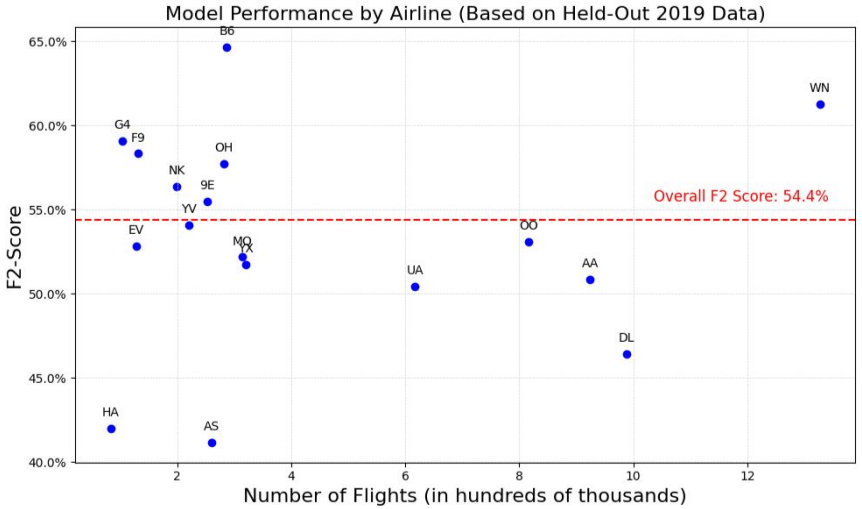
Final Result: Held-Out Dataset (2019)

- Sigmoid activation for the hidden layers, softmax for the output layer
- Limited-memory BFGS for the optimizer
- Block Size of 128
- Learning Rate of 0.03
- Max Iterations of 100

Estimator	Dataset	Hyperparameters	F-Beta (beta=2) Metric	AUC Metric
Neural Network 3	DOWNSAMPLED 2015-2018 Train	layers=[81, 8, 4, 2]	54.4%	70.9%

Group Name	Evaluation metric used (maybe different to ML loss function)	cross fold validation performance	Performance on blind test set (2021)
Group 10 Section 3	Accuracy, Precision, Recall,F1	Accuracy: 77% Precision: 71% Recall: 71% F1: 0.673	
Group 1 Section 3	F1	Best regularization parameter : 0.01 F1 score average: 0.7	
Group 2 Section 3	AUC-ROC	AUC: 80.9%	AUC: 79.6%
Group 1 Section 5	F-Beta	f-beta: 63.1%, AUC: 71.8%	f-beta: 54.4%, AUC: 70.1%

Post-Mortem Gap Analysis on Held-Out (2019)

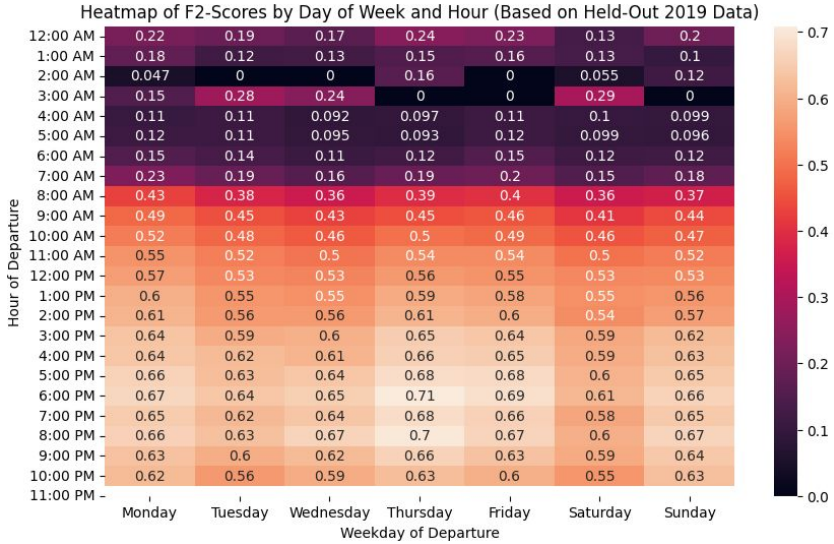


Error Patterns Across Airlines

- Poor on Alaska Airlines and Hawaii Airlines
- Poor on United, Delta, American

Error Patterns Across Time of the Week

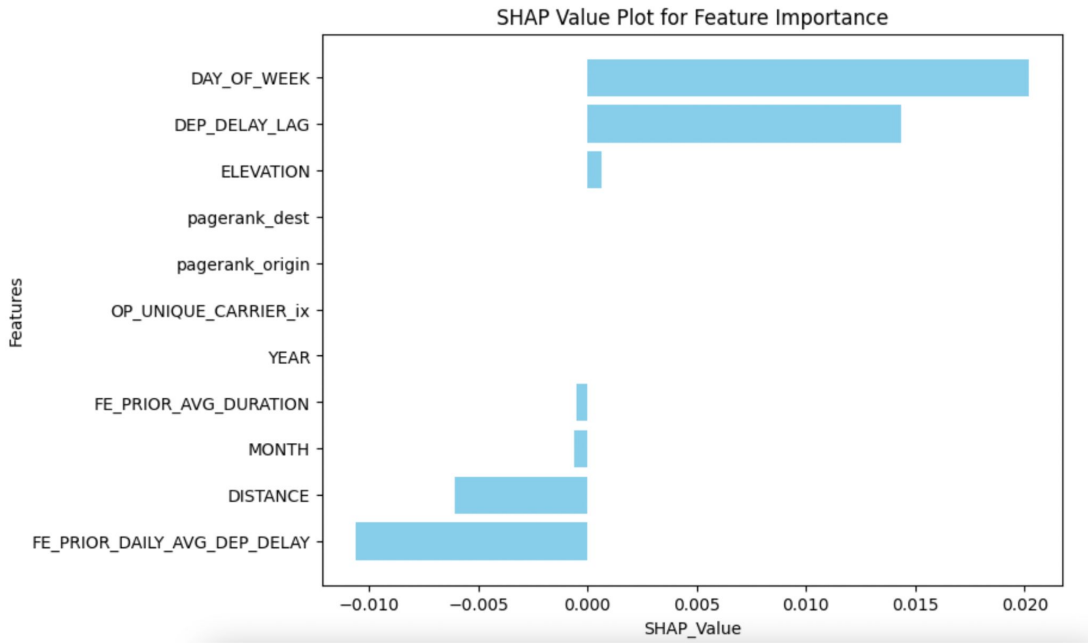
- Better on weekday afternoons and evenings
- Poor on mornings, especially before 7:00 a.m.



Shap Values

With 500 rows of data in test set

	0
DAY_OF_WEEK	0.020177
DISTANCE	-0.006085
ELEVATION	0.000640
FE_PRIOR_DAILY_AVG_DEP_DELAY	-0.010594
FE_PRIOR_AVG_DURATION	-0.000483



Conclusion and Next Steps

- Downsampling the majority class was key to model improvement
- Need to add more predictive features to improve performance
- **Planned improvements**
  - Features
    - Daily or weekly seasonal trend
    - Weather data features - taking care to avoid leakage
    - Interaction terms (e.g. MONTH*HOLIDAY)
  - Datasets
    - Major Events
    - Airport (e.g. maintenance, layout)
    - Airline (e.g. reputation, marketing)
    - Plane details (e.g. model, manufacturer, num of seats)

Other Packages we Tried

- tempo (Databricks labs)
  - Wraps the Spark data frame with a TSDF object
  - Creates smaller partitions that that are time aware and allow you to slice across partitions.
- prophet (Facebook)
  - Forecasting time series data with seasonality trends
  - Time blocked crossed validation
  - Limitations:
    - Very memory intensive
    - Difficult to serialize model
- fugue + statsforecast
  - Faster alternative to prophet
  - Limited documentation
- xgboost
  - Computationally heavy gradient boosting
  - Often replaced with lightgbm in Spark

Lessons Learned with Big Data

- Lessons Learned
  - Do your hypertuning on a subset of data
  - Checkpoint often
  - Create a robust data validation/quality assurance module as part of the data pipeline
  - Track experiments
  - Implement version control
- Challenges
  - More limited hypertuning due to time constraints
  - Team productivity challenges when sharing compute
  - Monitoring resources and maximizing use and efficiency
  - Limiting scope based on difficulty of working with data at scale

Questions?

Exploratory Data Analaysis

Figure 1

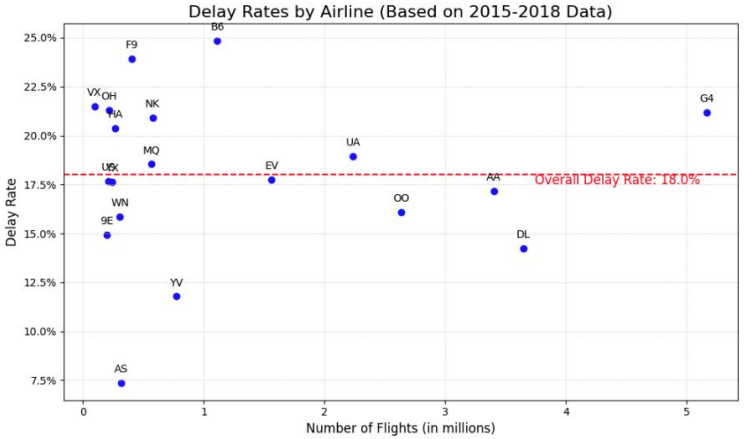


Figure 2