Predicting Flight Delays

Team 21: Dmitry Enin, Egor Lebedev, Alie Ablaeva, Anastasiia Ankudinova

Introduction

Flight delays cause significant economic losses ($32.9 billion annually in the US) and inconvenience for passengers. This project leverages Big Data technologies to predict flight delays using historical data (2019–2023). The goal is to reduce costs and improve passenger experience by providing actionable insights. The result of our work would be the prediction will be delay for specific flight or not.

Business objectives

- Reduce costs: minimize financial losses due to delays
- Improve passenger experience: provide timely predictions to help travelers plan better
- Scalability: handle large datasets (3M records) efficiently using distributed systems (Spark, Hive)

Data description

https://www.kaggle.com/datasets/patrickzel/flight-delay-and-cancellation-dataset-2019-2023?select=dictionary.html

This dataset contains comprehensive records of U.S. flight delays and cancellations from January 2019 to December 2023, sourced from the U.S. Bureau of Transportation Statistics (BTS). It includes details on airlines, airports, departure/arrival times, delays, and cancellation reasons.

Data characteristic

- Size: ~3 million records (2019-2023)
- Target variable: DEP_DELAY (numerical, mean = 10.1 min, std = 49.3 min)
- Classification threshold: delays > 0 minutes
- 3 types of features: categorical (AIRLINE, ORIGIN, DEST), numerical (DEP_TIME, DEP_DELAY), datetime (FL_DATE)

Updated Header	Source Header	Data Type	Description
FL_DATE	FlightDate	object	Flight Date (yyyymmdd)
AIRLINE_CODE	Reporting_Airline	object	Unique Carrier Code. When the same code has been used by multiple carriers, a numeric suffix is used for earlier users, for example, PA, PA(1), PA(2). Use this field for analysis across a range of years.
DOT_CODE	DOT_ID_Reporting_Airline	int64	An identification number assigned by US DOT to identify a unique airline (carrier). A unique airline (carrier) is defined as one holding and reporting under the same DOT certificate regardless of its Code, Name, or holding company/corporation.
FL_NUMBER	Flight_Number_Reporting_Airline	int64	Flight Number
ORIGIN	Origin	object	Origin Airport
ORIGIN_CITY	OriginCityName	object	Origin Airport, City Name
DEST	Dest	object	Destination Airport
DEST_CITY	DestCityName	object	Destination Airport, City Name
CRS_DEP_TIME	CRSDepTime	int64	CRS Departure Time (local time: hhmm)
DEP_TIME	DepTime	float64	Actual Departure Time (local time: hhmm)
DEP_DELAY	DepDelay	float64	Difference in minutes between scheduled and actual departure time. Early departures show negative numbers.
TAXI_OUT	TaxiOut	float64	Taxi Out Time, in Minutes
WHEELS_OFF	WheelsOff	float64	Wheels Off Time (local time: hhmm)
WHEELS_ON	WheelsOn	float64	Wheels On Time (local time: hhmm)
TAXI_IN	TaxiIn	float64	Taxi In Time, in Minutes
CRS_ARR_TIME	CRSArrTime	int64	CRS Arrival Time (local time: hhmm)
ARR_TIME	ArrTime	float64	Actual Arrival Time (local time: hhmm)
ARR_DELAY	ArrDelay	float64	Difference in minutes between scheduled and actual arrival time. Early arrivals show negative numbers.
CANCELLED	Cancelled	float64	Cancelled Flight Indicator (1=Yes)
CANCELLATION_CODE	CancellationCode	object	Specifies The Reason For Cancellation
DIVERTED	Diverted	float64	Diverted Flight Indicator (1=Yes)
CRS_ELAPSED_TIME	CRSElapsedTime	float64	CRS Elapsed Time of Flight, in Minutes
ELAPSED_TIME	ActualElapsedTime	float64	Elapsed Time of Flight, in Minutes
AIR_TIME	AirTime	float64	Flight Time, in Minutes
DISTANCE	Distance	float64	Distance between airports (miles)
DELAY_DUE_CARRIER	CarrierDelay	float64	Carrier Delay, in Minutes
DELAY_DUE_WEATHER	WeatherDelay	float64	Weather Delay, in Minutes
DELAY_DUE_NAS	NASDelay	float64	National Air System Delay, in Minutes
DELAY_DUE_SECURITY	SecurityDelay	float64	Security Delay, in Minutes
DELAY_DUE_LATE_AIRCRAFT	LateAircraftDelay	float64	Late Aircraft Delay, in Minutes

Architecture of data pipeline

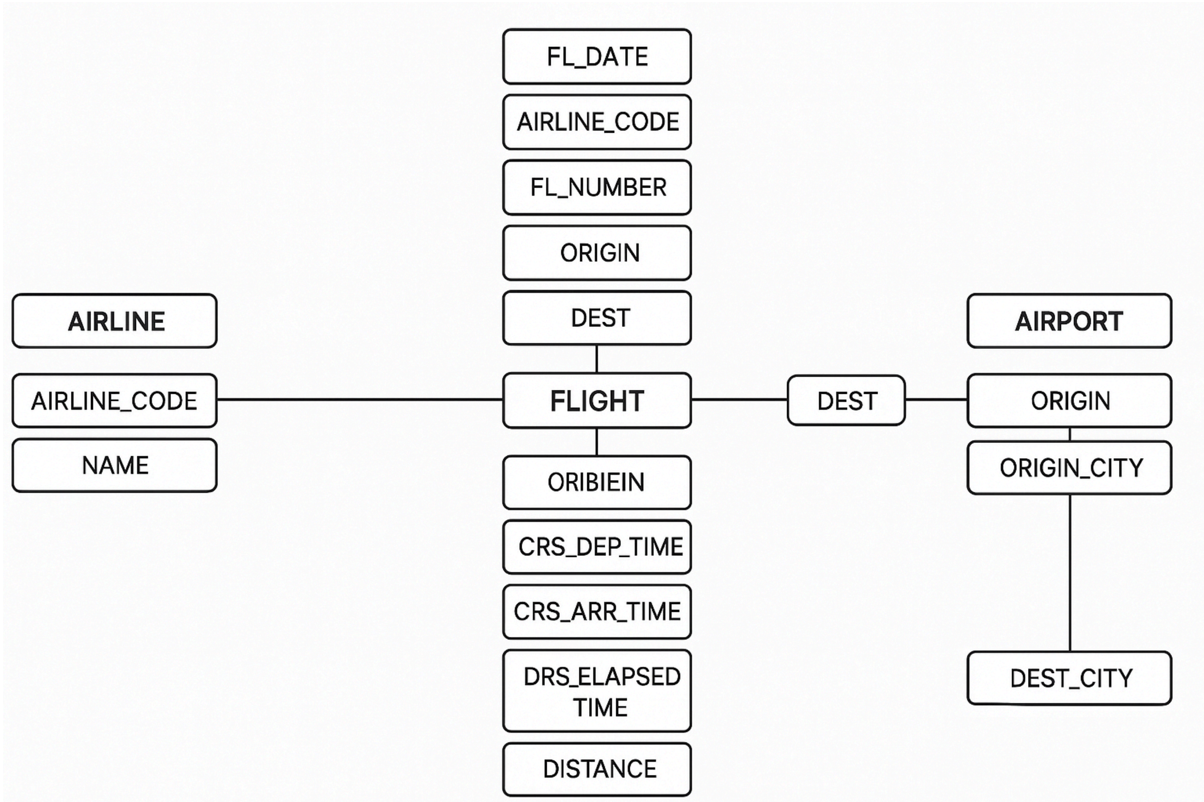
- Ingestion: Dataset downloaded using wget
- Storage: Uploaded to PostgreSQL
- Distributed Storage: Exported to HDFS as Parquet
- Data Optimization in Hive:
  Partitioned by ORIGIN
  Bucketed by FLIGHT_NUMBER

Stage	Input	Output

Data collection	Raw data in CSV	PostgreSQL database
Data transfer	PostgreSQL	HDFS Parquet file
Data optimization	HDFS	Hive table
EDA	Hive tables	Delay patterns (chart/insights)
Model building	Processed features (Spark)	Trained model (Random Forest and Binary tree)
Dashboard creation	Analysis results	Dashboard

Data preparation

ER diagram



Some samples from the database

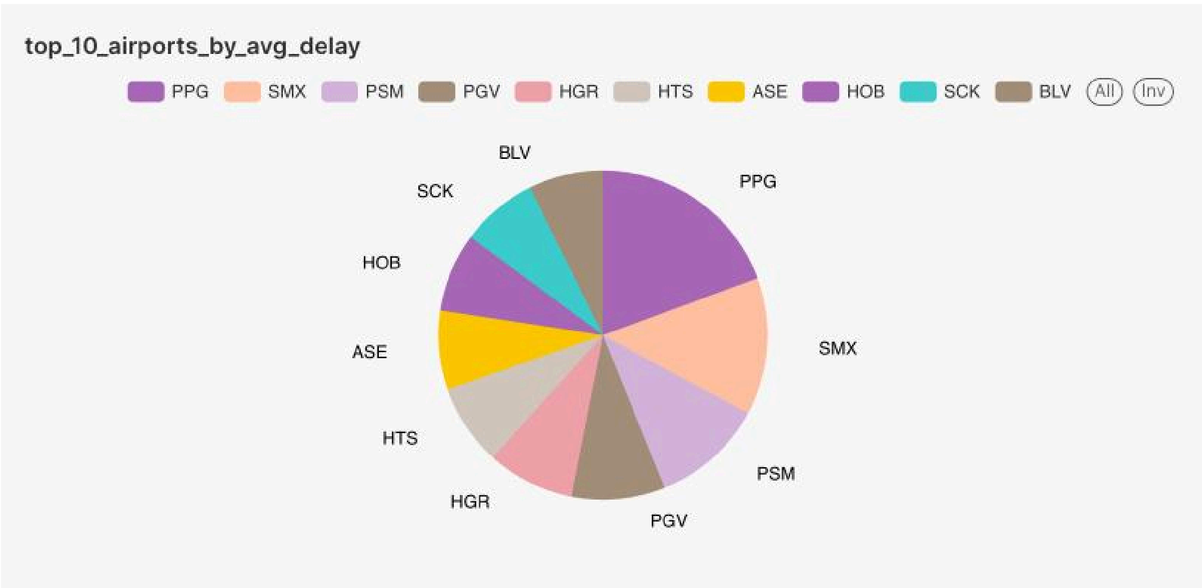
fl_date	airline_code	fl_number	origin	origin_city	dest	dest_city	crs_dep_time	crs_arr_time	crs_elapsed_time	distance
2019-01-01	9E	3303	MSP	Minneapolis, MN	CWA	Mosinee, WI	1355	1510	75	175
2019-01-01	9E	3281	MSP	Minneapolis, MN	CVG	Cincinnati, OH	1404	1709	125	596
2019-01-01	9E	3284	ATL	Atlanta, GA	FSM	Fort Smith, AR	1902	2005	123	579
2019-01-01	9E	3295	DTW	Detroit, MI	EWR	Newark, NJ	1230	1422	112	488
2019-01-01	9E	3302	ATL	Atlanta, GA	EYW	Key West, FL	1101	1253	112	646

Creating Hive tables and preparing the data for analysis

- External tables created over HDFS
- Partitioned by ORIGIN
- Bucketed by FLIGHT_NUMBER
- Cleaned missing values
- Filtered unrealistic values (negative delays)
- Created binary target (delayed vs not delayed)

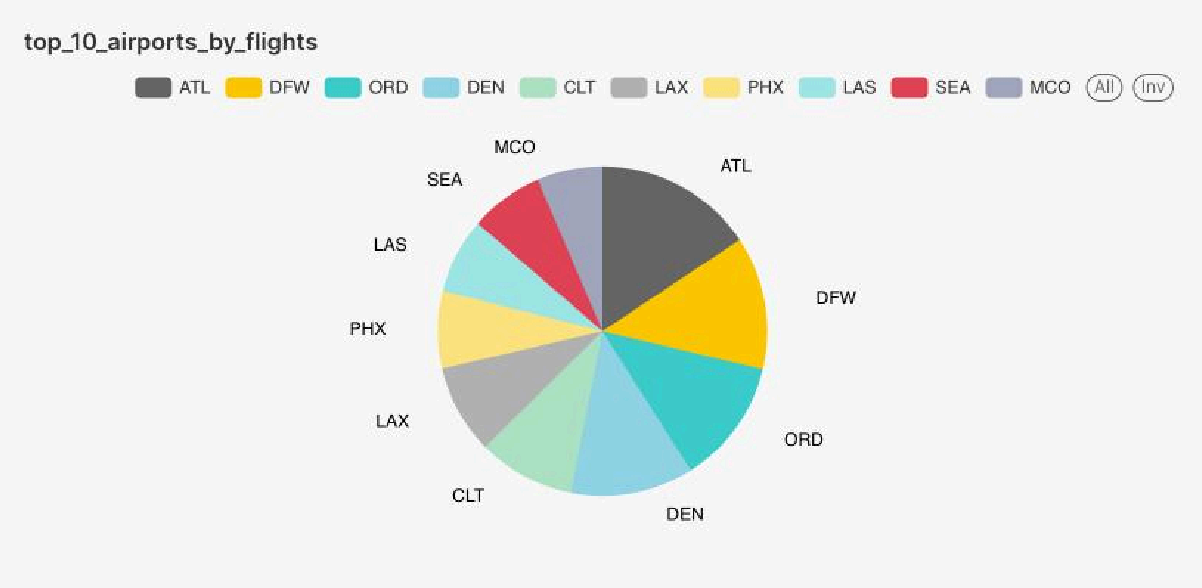
Data analysis

Charts and their interpretation

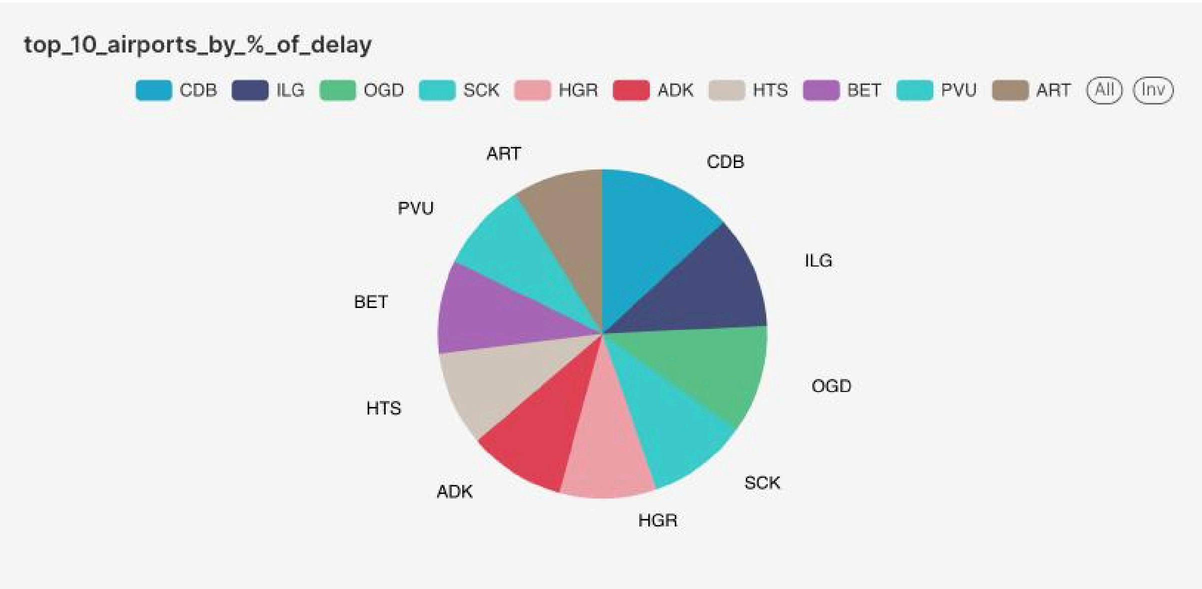


1. top 10 airports by average delay: limited runways/staff amplify delays even

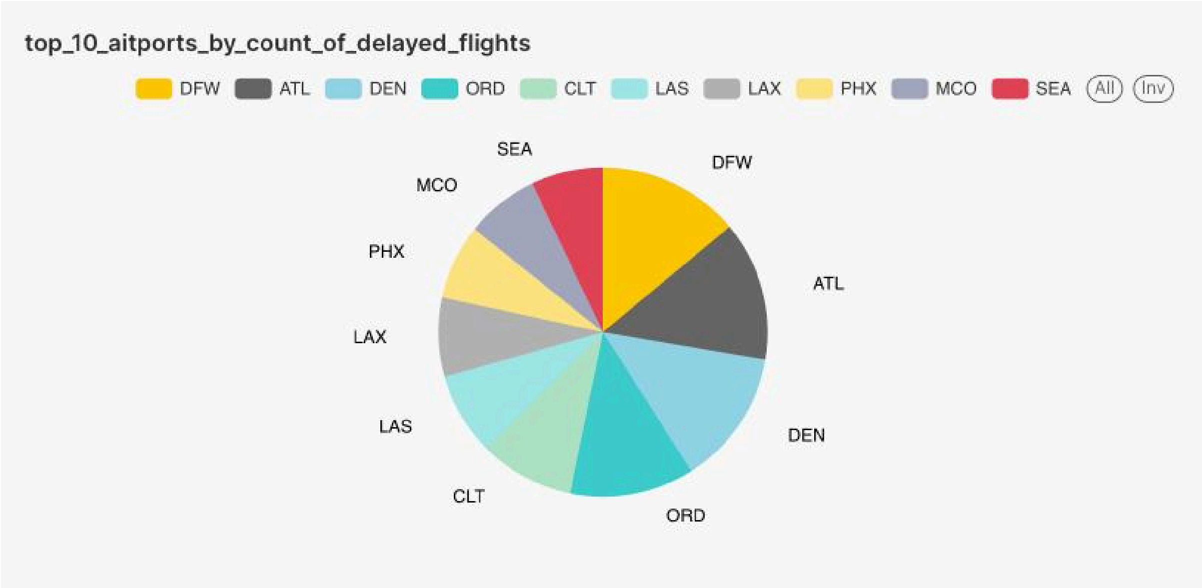
with fewer flights



2. top 10 airports by flights: larger airports manage delays better despite high traffic



3. top 10 airports by % of delay: smaller airports receive fewer recovery resources from airlines



4. top 10 airports by count of delayed flights: congestion is the primary cause for delays at major hubs

ML modelling

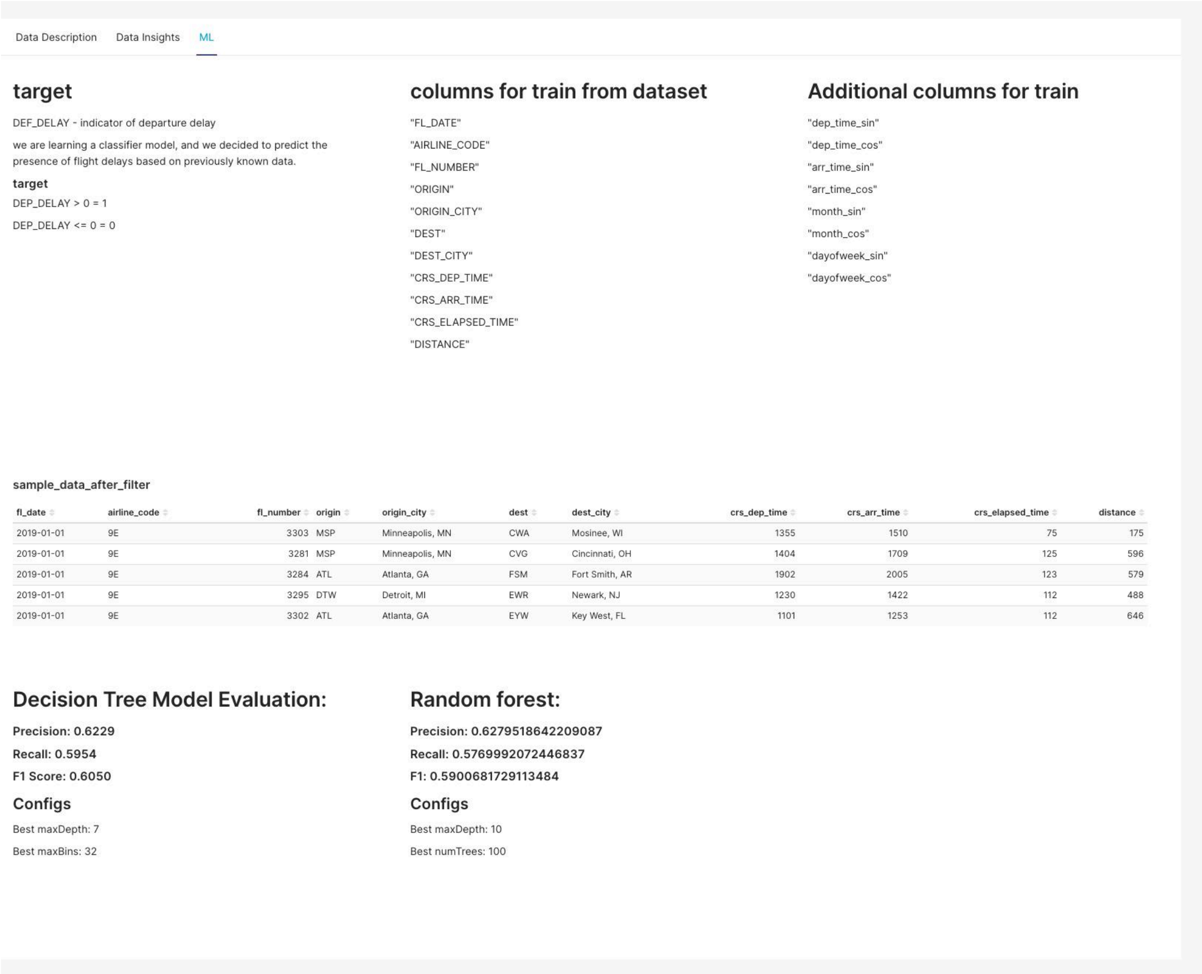
<u>Feature extraction and data preprocessing</u>

- We used cyclic encoding because we have the data connected with date(time, day of week and month)
- For the cyclic encoding we need to add new features sin and cos for each feature which connected with time, day of week or month
- Standardize the numerical features (date)
- Balance the number of flight with delays and without
- Split the data for train and test

<u>Training and fine-tuning</u>

- Tried 2 models for binary classification: RandomForest and Binary tree
- Used GridSearch for fine-tuning hyperparameters

<u>Evaluation</u>



Data Description Data Insights ML

target

DEF_DELAY - indicator of departure delay

we are learning a classifier model, and we decided to predict the presence of flight delays based on previously known data.

target

DEP_DELAY > 0 = 1

DEP_DELAY <= 0 = 0

columns for train from dataset

"FL_DATE"
"AIRLINE_CODE"
"FL_NUMBER"
"ORIGIN"
"ORIGIN_CITY"
"DEST"
"DEST_CITY"
"CRS_DEP_TIME"
"CRS_ARR_TIME"
"CRS_ELAPSED_TIME"
"DISTANCE"

Additional columns for train

"dep_time_sin"
"dep_time_cos"
"arr_time_sin"
"arr_time_cos"
"month_sin"
"month_cos"
"dayofweek_sin"
"dayofweek_cos"

sample_data_after_filter

fl_date	airline_code	fl_number	origin	origin_city	dest	dest_city	crs_dep_time	crs_arr_time	crs_elapsed_time	distance
2019-01-01	9E	3303	MSP	Minneapolis, MN	CWA	Mosinee, WI	1355	1510	75	175
2019-01-01	9E	3281	MSP	Minneapolis, MN	CVG	Cincinnati, OH	1404	1709	125	596
2019-01-01	9E	3284	ATL	Atlanta, GA	FSM	Fort Smith, AR	1902	2005	123	579
2019-01-01	9E	3295	DTW	Detroit, MI	EWR	Newark, NJ	1230	1422	112	488
2019-01-01	9E	3302	ATL	Atlanta, GA	EYW	Key West, FL	1101	1253	112	646

Decision Tree Model Evaluation:

Precision: 0.6229

Recall: 0.5954

F1 Score: 0.6050

Configs

Best maxDepth: 7

Best maxBins: 32

Random forest:

Precision: 0.6279518642209087

Recall: 0.5769992072446837

F1: 0.5900681729113484

Configs

Best maxDepth: 10

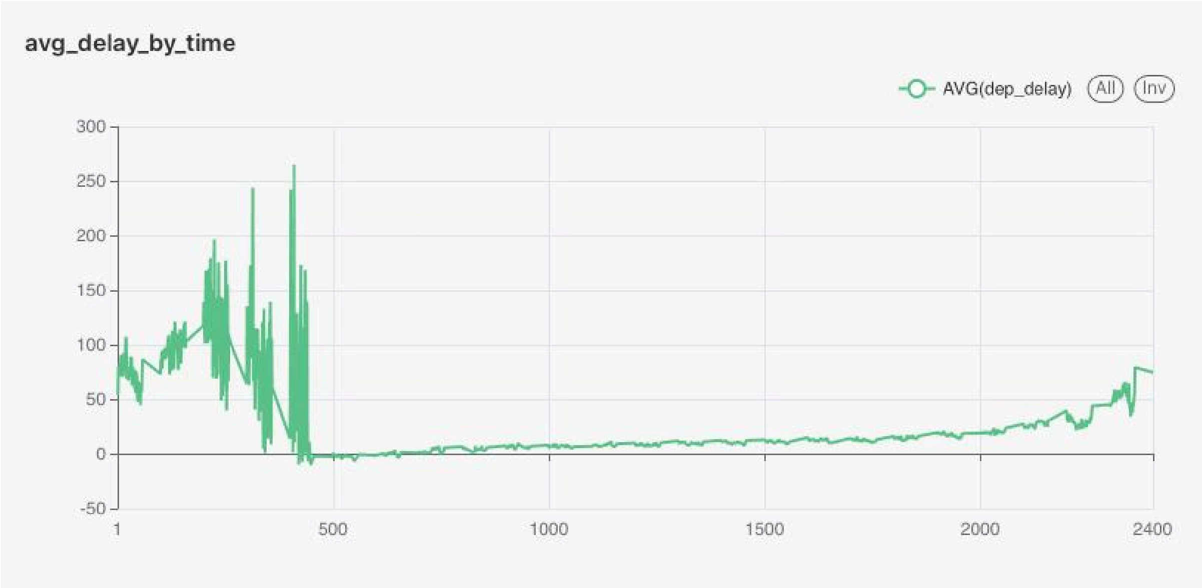
Best numTrees: 100

Data presentation

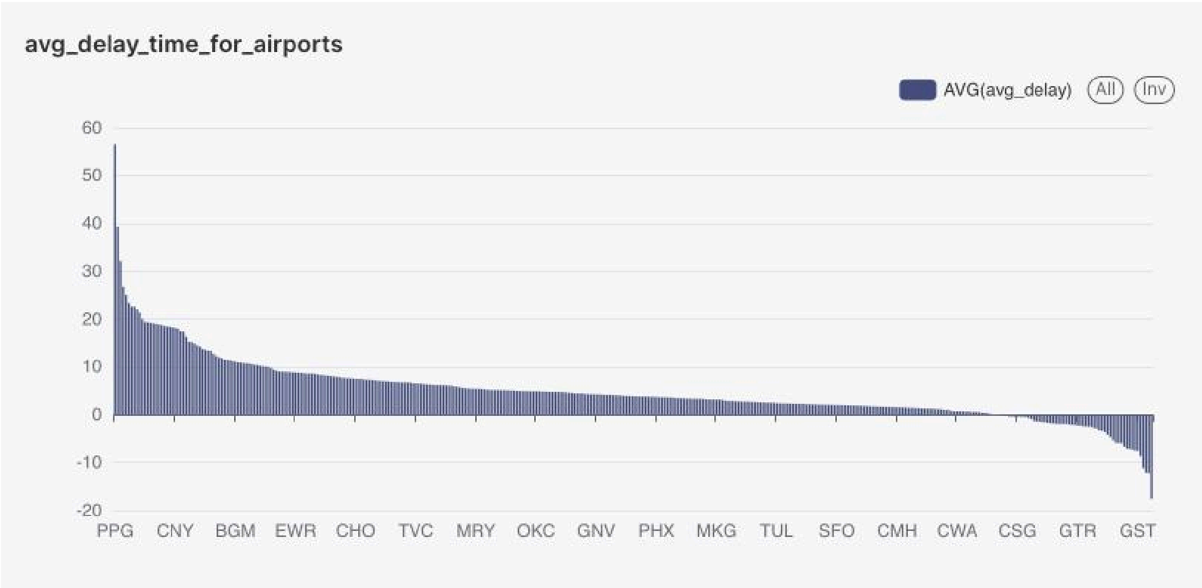
<u>Description of dashboard</u>

We create the dashboard for data and for the ml part separately (the dashboard of ml part you can find above)

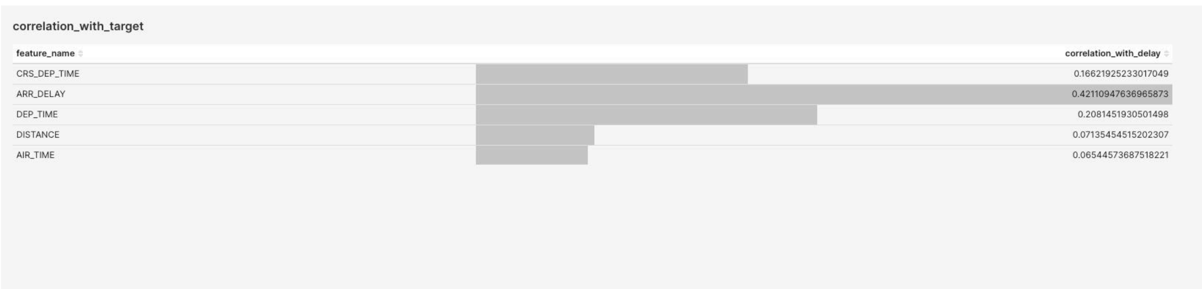
<u>Description of charts</u>



1. *avg_delay_by_time*: line chart plots the average departure delay (dep_delay) against the scheduled departure time (in 24-hour format from 0 to 2400)



2. *avg_delay_time_for_airports*: bar chart compares the average delay time by airport

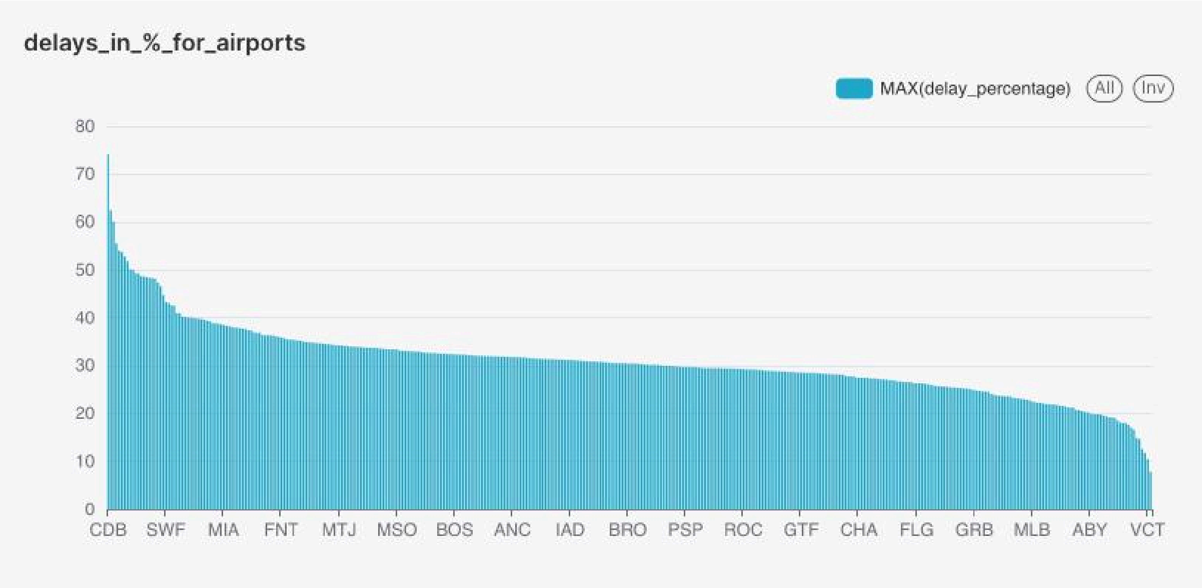


correlation_with_target

feature_name	correlation_with_delay
CRS_DEP_TIME	0.16621925233017049
ARR_DELAY	0.42110947636965873
DEP_TIME	0.2081451930501498
DISTANCE	0.07135454515202307
AIR_TIME	0.06544573687518221

3. *correlation_with_target*: correlation coefficients between various numerical features and target variable DEP_DELAY(departure delay)

4. *delays_for_airports*: the number of delayed flights for each airport (ORIGIN)



5. *delays_in_%_for_airports:* percentage of delayed flights for each airport (ORIGIN)

Findings

- ARR_DELAY has strong positive correlation with target = 0.421
- Delays are highly location-dependent
- Percentage-based delay analysis uncovers underperforming airports
- The most number of delays from 00:00 to ~05:00 so we can recommend not to plan flight on this time
- The most number of airport have ~20% of delays

Conclusion

We successfully built a scalable pipeline and trained ML models for predicting flight delays. We learned to handle Big Data workflows using Spark, Hive, and PostgreSQL. Also, we improve our skills in team work and performance optimization.

Reflection on own work

1. Challenges and difficulties:
    - Data size and optimization in Hive
    - Training time for large dataset
    - Handling imbalanced delay distribution
2. Recommendations:
    - Integrate weather APIs
    - Try to use deep learning (for example, LSTM)
    - Deploy model for users on web or mobile application
    - Not to schedule the trip form 00:00 to ~05:00 because of the lots of delays in this time

Project task	Task description	Ankudinova Anastasiia	Dmitry Enin	Alie Ablaeva	Egor Lebedev	Deliverables	Average hours spent
Data collection	Download and store dataset. Extract useful sample for project	40%	0%	0%	60%	scripts/data_collection.sh	4
Data transfer	Move to HDFS and convert to Parquet	30%	0%	0%	70%	sql/create_tables.sql scripts/build_projectdb.py scripts/ingest_hdfs.sh	4
Hive setup	Create and optimize the Hive tables	0%	80%	0%	20%	01_init_database.hql 02_optimize_tables.hql	6
EDA	Delay pattern analysis	0%	0%	100%	0%	sql/q1.hql sql/q2.hql sql/q3.hql sql/q4.hql	8

						scripts/stage2.sh	
Modelling	Train classifiers	0%	100%	0%	0%	notebooks/ml.ipynb scripts/ml.py scripts/stage3.sh	19
Dashboard	Create dashboards	40%	0%	60%	0%	charts	9
Report and slides	Compile and analyse results, write the report	70%	0%	0%	30%	pdf file and presentation	8