To Fly
or
Not to Fly

MEET THE TEAM

JANET
Storyteller

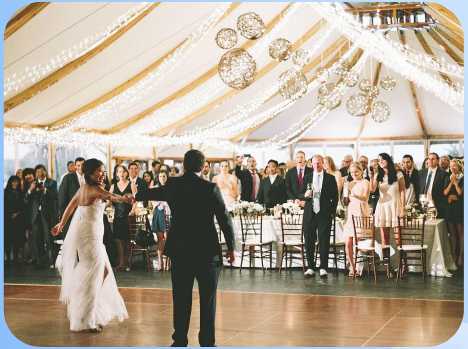
ERIC
Data Collector Guru

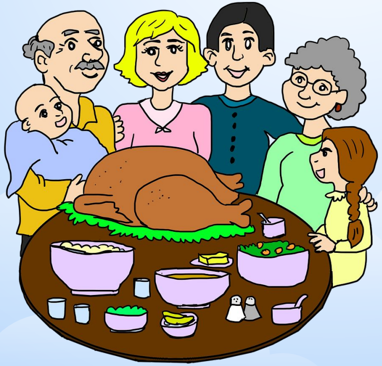
NEESHA
Data Cruncher

RYAN
Data Viz Whiz

The Story

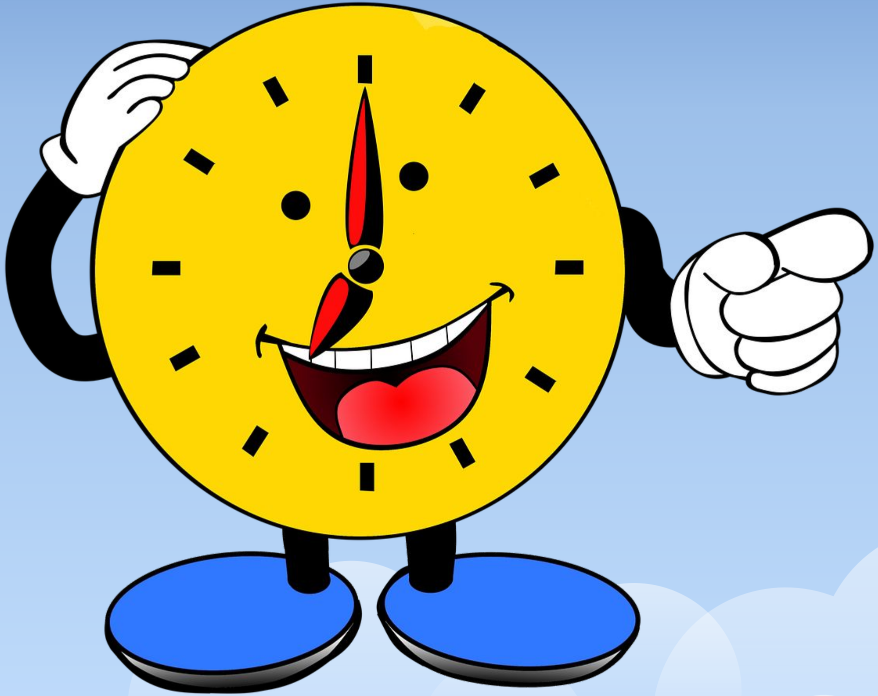
Delayed or canceled flights can ruin business trips, vacations, family events, and so much more.





$28 Billion

FAA/Nextor estimated the annual costs of delays in 2018



Questions:

1 How does weather impact flight cancellations?

2 Are certain weather events impact the decision to cancel more than other?

3 Are certain airlines more prone to cancel flights based on weather?

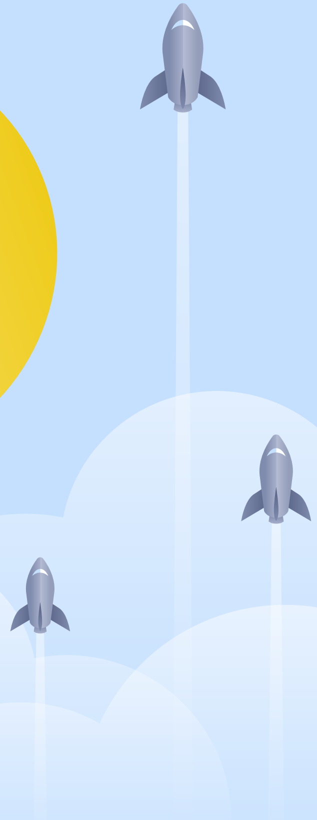
Tools Used

Dashboard
Google Slides

Data Analysis
Scikit-learn, Supervised learning, Python 3.7,

Data Collection
Python, Kaggle, Kaggle API, PostgreSQL, SQLAlchemy, Quick DBD Psycopg2, pgAdmin, Jupyter Notebook

Data Viz
Matplotlib, D3, Leaflet.

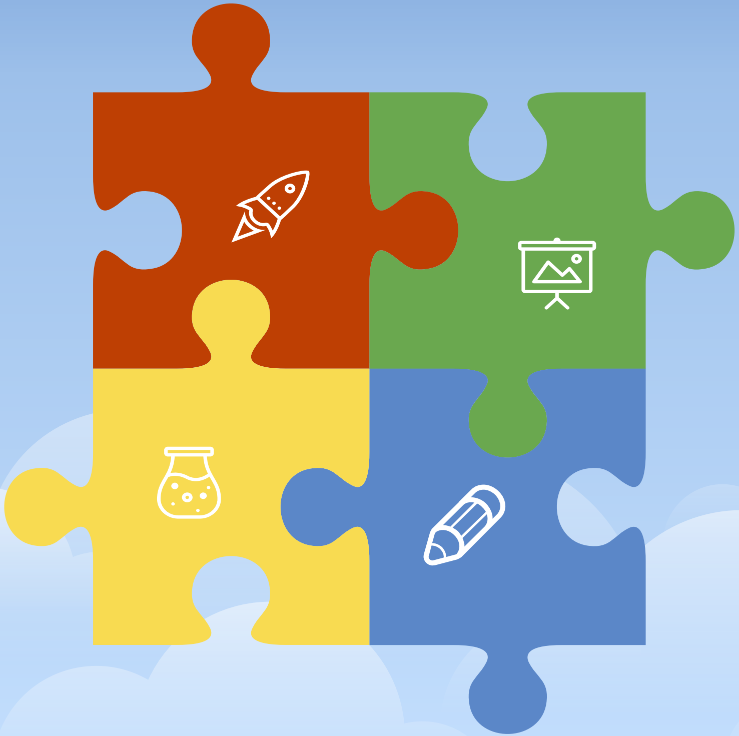


ERIC

Data Collector Guru

- Data Acquisition
- Preprocessing
- Database Storage
- Data Retrieval

Data Sources



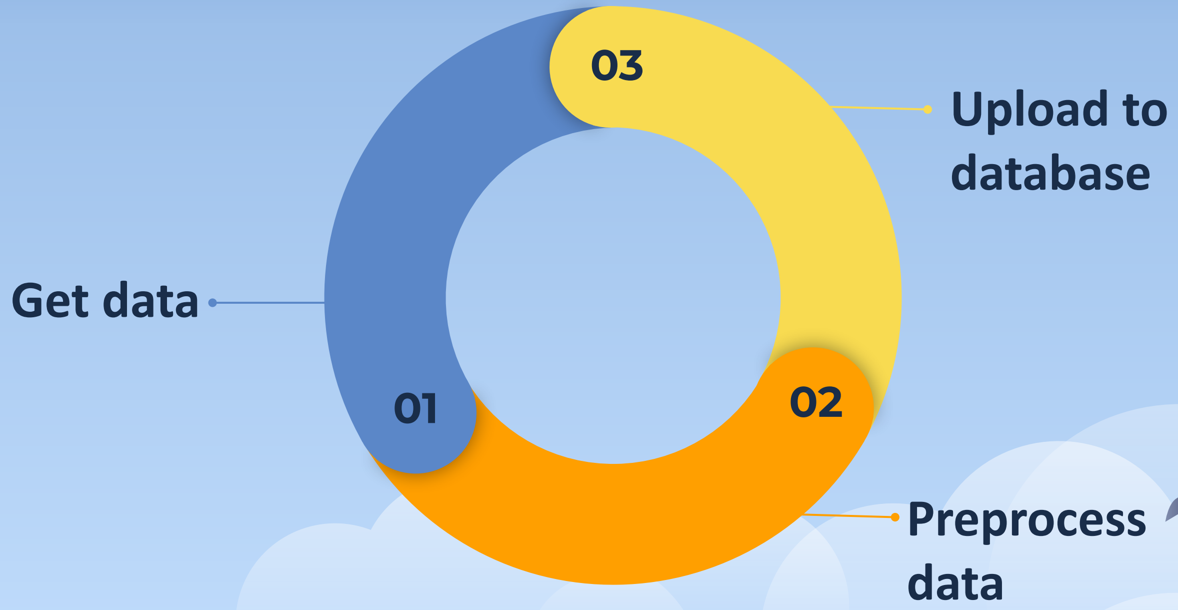
1. KAGGLE, Historical Flight Delay and Weather Data USA
   - United States Bureau of Transportation Statistics
   - National Oceanic and Atmospheric Administration
2. The Global Airport Database

Data Sources

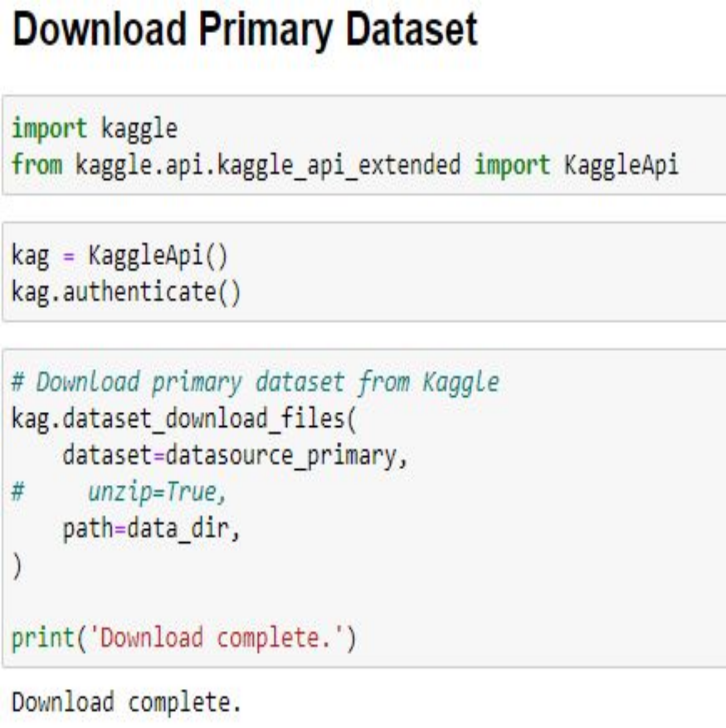
1. KAGGLE, Historical Flight Delay and Weather Data USA

2. The Global Airport Database

Extract-Transform-Load



Download Datasets



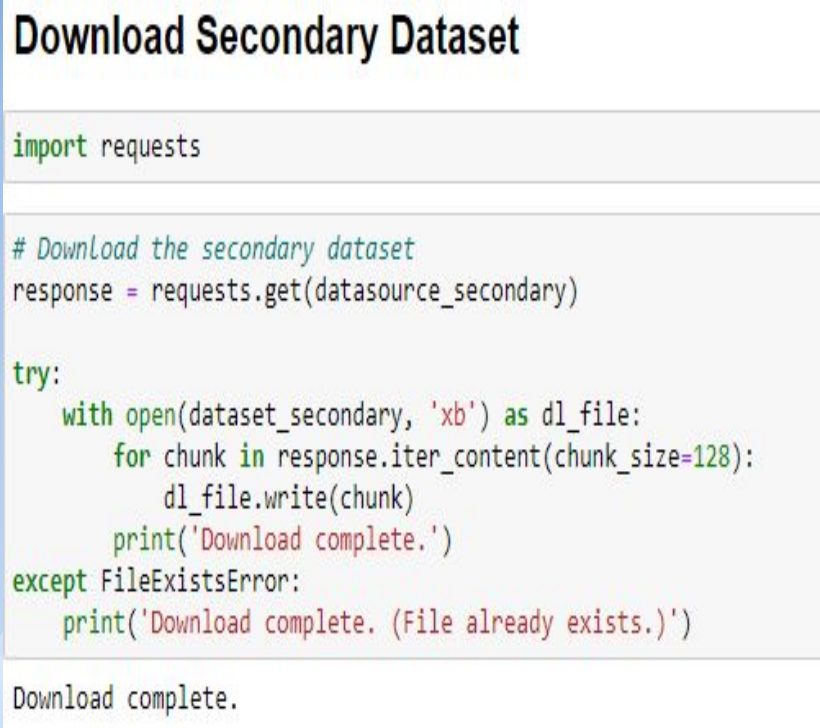
Download Primary Dataset

```
import kaggle
from kaggle.api.kaggle_api_extended import KaggleApi
```

```
kag = KaggleApi()
kag.authenticate()
```

```
# Download primary dataset from Kaggle
kag.dataset_download_files(
    dataset=datasource_primary,
#     unzip=True,
    path=data_dir,
)

print('Download complete.')
```

```
Download complete.
```



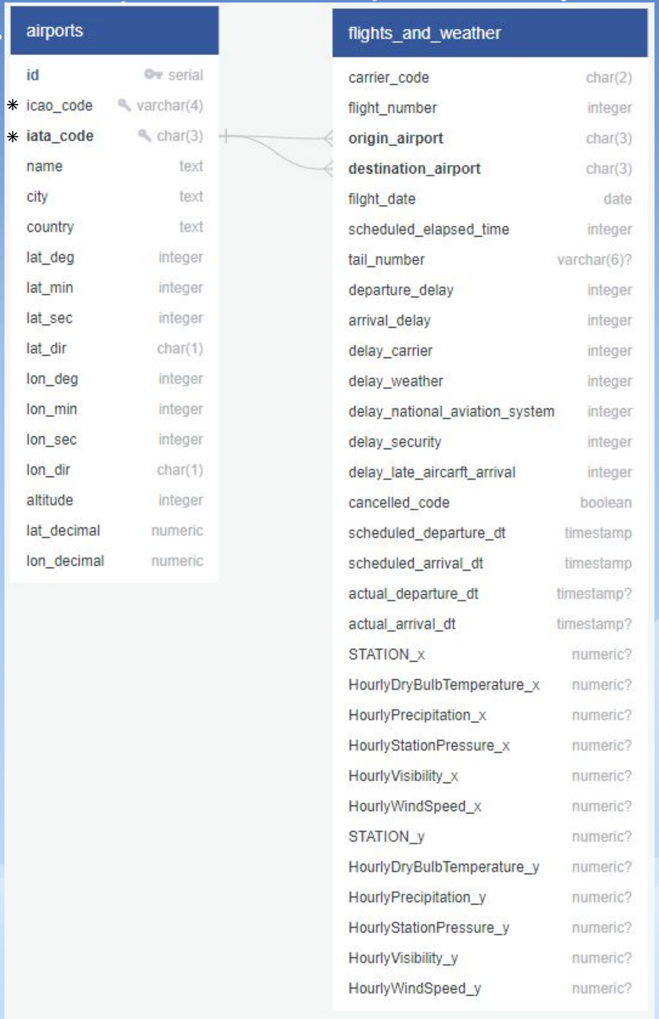
Download Secondary Dataset

```
import requests
```

```
# Download the secondary dataset
response = requests.get(datasource_secondary)

try:
    with open(dataset_secondary, 'xb') as dl_file:
        for chunk in response.iter_content(chunk_size=128):
            dl_file.write(chunk)
        print('Download complete.')
except FileExistsError:
    print('Download complete. (File already exists.)')
```

```
Download complete.
```

Preliminary Entity Relationship Diagram



airports

id	serial
* icao_code	varchar(4)
* iata_code	char(3)
name	text
city	text
country	text
lat_deg	integer
lat_min	integer
lat_sec	integer
lat_dir	char(1)
lon_deg	integer
lon_min	integer
lon_sec	integer
lon_dir	char(1)
altitude	integer
lat_decimal	numeric
lon_decimal	numeric

flights_and_weather

carrier_code	char(2)
flight_number	integer
origin_airport	char(3)
destination_airport	char(3)
filght_date	date
scheduled_elapsed_time	integer
tail_number	varchar(6)?
departure_delay	integer
arrival_delay	integer
delay_carrier	integer
delay_weather	integer
delay_national_aviation_system	integer
delay_security	integer
delay_late_aircarft_arrival	integer
cancelled_code	boolean
scheduled_departure_dt	timestamp
scheduled_arrival_dt	timestamp
actual_departure_dt	timestamp?
actual_arrival_dt	timestamp?
STATION_x	numeric?
HourlyDryBulbTemperature_x	numeric?
HourlyPrecipitation_x	numeric?
HourlyStationPressure_x	numeric?
HourlyVisibility_x	numeric?
HourlyWindSpeed_x	numeric?
STATION_y	numeric?
HourlyDryBulbTemperature_y	numeric?
HourlyPrecipitation_y	numeric?
HourlyStationPressure_y	numeric?
HourlyVisibility_y	numeric?
HourlyWindSpeed_y	numeric?

*** ICAO:** International Civil Aviation Organization
*** IATA:** International Air Transport Association

Airports Data

Flights and Weather Data

Preprocessing the tables

Airports Data

- General text conversion
- Column definitions: data types and unique values

Flights and Weather Data

- Irrelevant data: cancellation types
- Redundant data: date vs. year-month-day

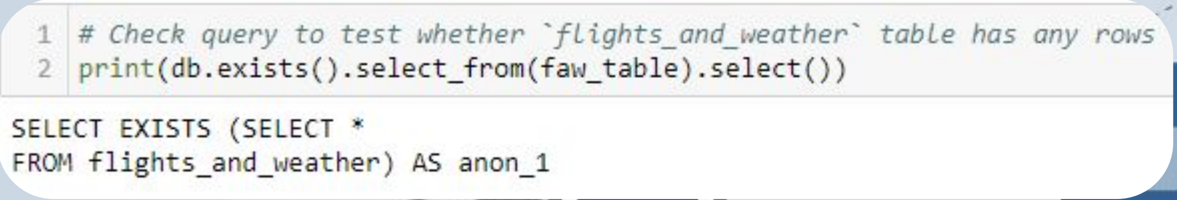
PostgreSQL Database Integration

Upload

More complications

Query

- Use object-relational mapping (ORM) to create SQL objects in SQLAlchemy (Python package)



```
# Check query to test whether `flights_and_weather` table has any rows
print(db.exists().select_from(faw_table).select())
```

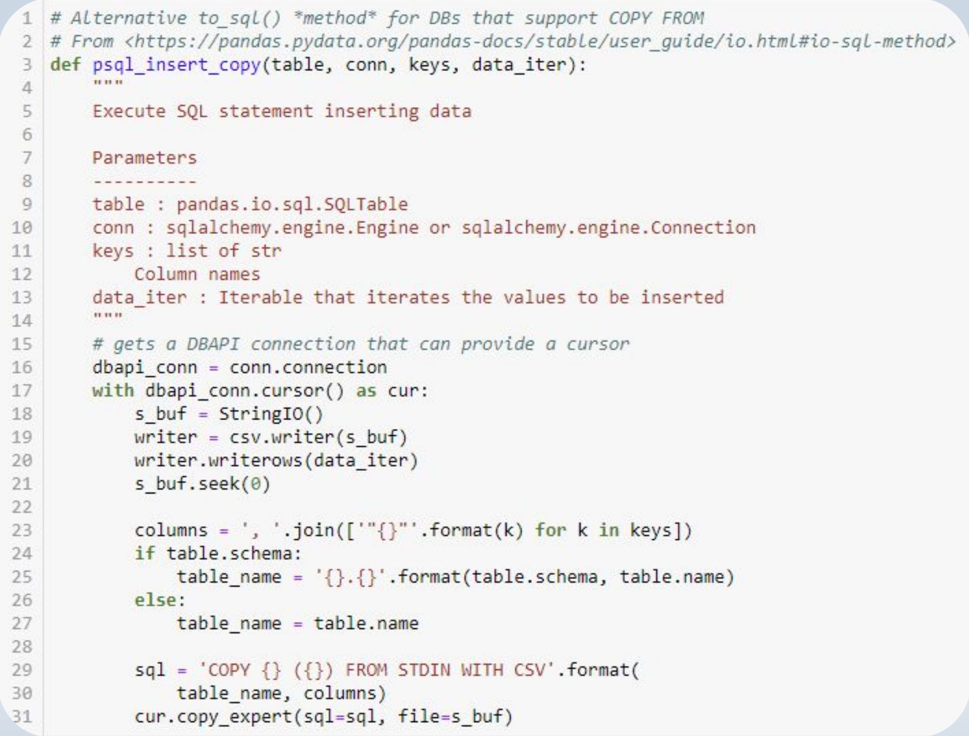
```
SELECT EXISTS (SELECT *
FROM flights_and_weather) AS anon_1
```

PostgreSQL Database Integration

Upload

- Use object-relational mapping (ORM) to create SQL objects in SQLAlchemy (Python package)
- Letter Case for Column Names: PostgreSQL columns are lowercase
- IATA code FOREIGN KEY CONSTRAINT
- Upload is slow: replace method with custom function!

Query

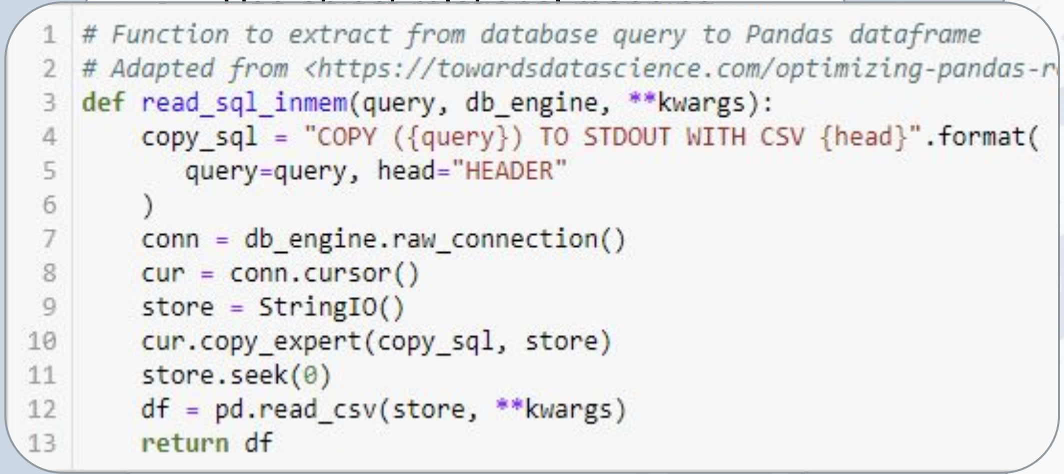


```python
# Alternative to_sql() *method* for DBs that support COPY FROM
# From <https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html#io-sql-method>
def psql_insert_copy(table, conn, keys, data_iter):
    """
    Execute SQL statement inserting data

    Parameters
    ----------
    table : pandas.io.sql.SQLTable
    conn : sqlalchemy.engine.Engine or sqlalchemy.engine.Connection
    keys : list of str
        Column names
    data_iter : Iterable that iterates the values to be inserted
    """
    # gets a DBAPI connection that can provide a cursor
    dbapi_conn = conn.connection
    with dbapi_conn.cursor() as cur:
        s_buf = StringIO()
        writer = csv.writer(s_buf)
        writer.writerows(data_iter)
        s_buf.seek(0)

        columns = ', '.join(['"{}"'.format(k) for k in keys])
        if table.schema:
            table_name = '{}.{}'.format(table.schema, table.name)
        else:
            table_name = table.name

        sql = 'COPY {} ({}) FROM STDIN WITH CSV'.format(
            table_name, columns)
        cur.copy_expert(sql=sql, file=s_buf)
```

PostgreSQL Database Integration

Upload

- Use object relational mapping



```
# Function to extract from database query to Pandas dataframe
# Adapted from <https://towardsdatascience.com/optimizing-pandas-r
def read_sql_inmem(query, db_engine, **kwargs):
    copy_sql = "COPY ({query}) TO STDOUT WITH CSV {head}".format(
       query=query, head="HEADER"
    )
    conn = db_engine.raw_connection()
    cur = conn.cursor()
    store = StringIO()
    cur.copy_expert(copy_sql, store)
    store.seek(0)
    df = pd.read_csv(store, **kwargs)
    return df
```

Query

- Needed different methods to retrieve the same data depending on how users were going to use the data: export to CSV or extract directly to Pandas dataframe
- Extraction is slow: once again, custom function to the rescue!

PostgreSQL Database Integration

Upload

- Use object-relational mapping (ORM) to create SQL objects in SQLAlchemy (Python package)
- Letter Case for Column Names: PostgreSQL columns are lowercase
- IATA code FOREIGN KEY CONSTRAINT
- Upload is slow: replace method with custom function!

Query

- Needed different methods to retrieve the same data depending on how users were going to use the data: export to CSV or extract directly to Pandas dataframe
- Extraction is slow: once again, custom function to the rescue!
- Pandas dataframe output required some post-processing due to how PostgreSQL outputs boolean values in CSVs (even virtual ones)

Final row count…

5,468,069

Whoa! That’s a lot of data!

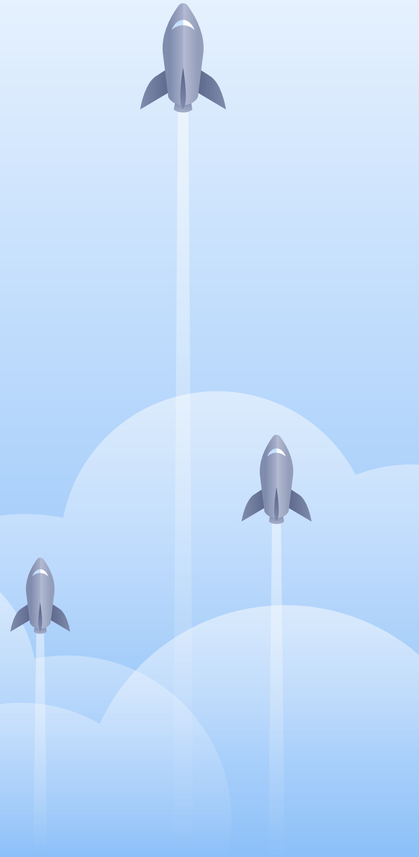
NEESHA

Data Cruncher

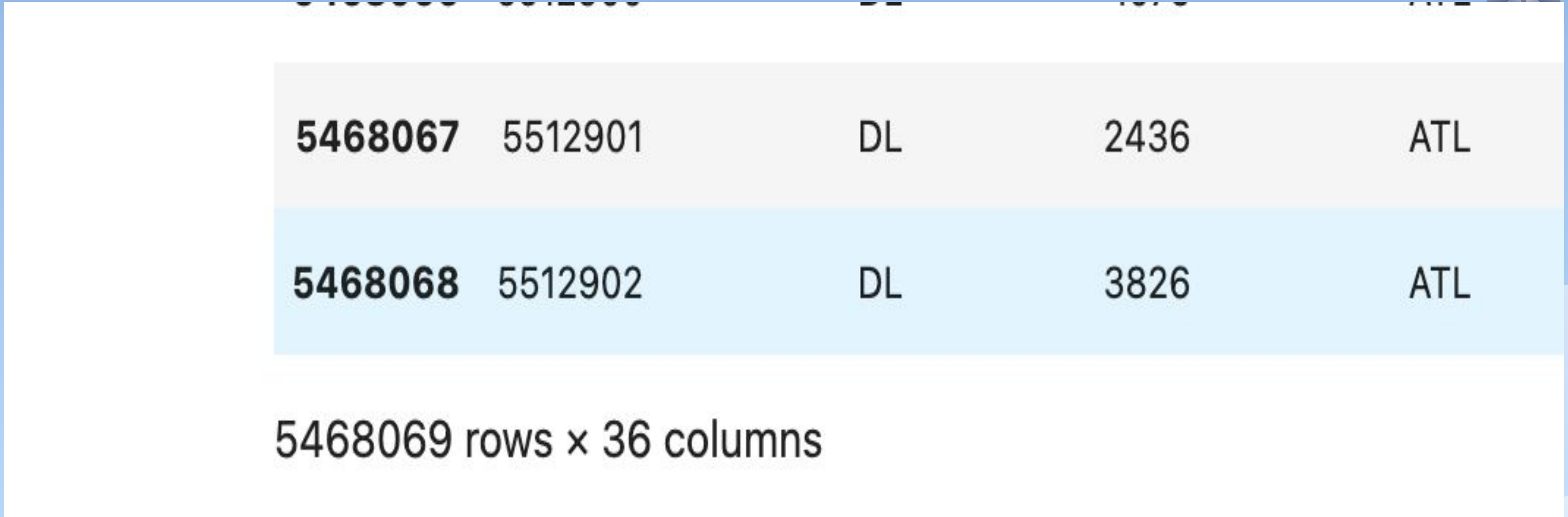
Description of the analysis phase of the project.

Machine Learning Model.

Analysis results.

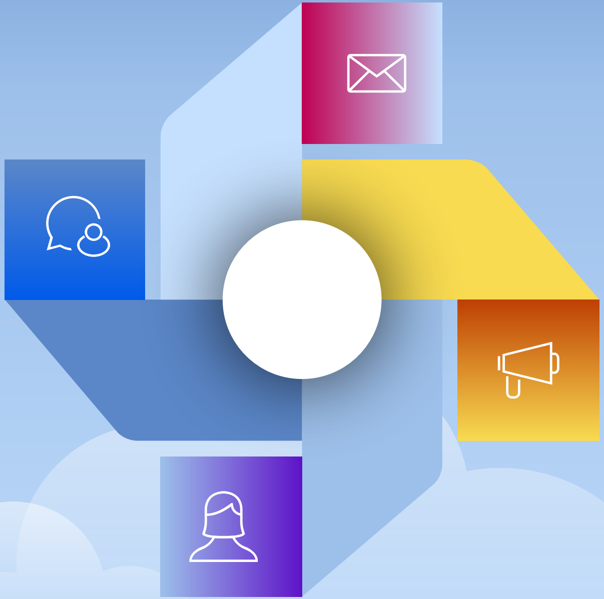


Available Dataset after merging



5468067	5512901	DL	2436	ATL
5468068	5512902	DL	3826	ATL

5468069 rows × 36 columns



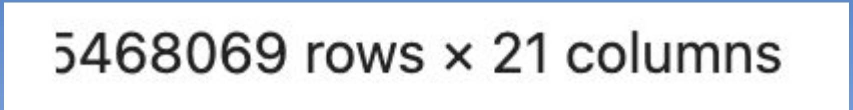
01 **Feature creation**

02 **Missing values**

03 **Label Encoding**

04 **Scaling the Data**

Features Selection:



5468069 rows × 21 columns

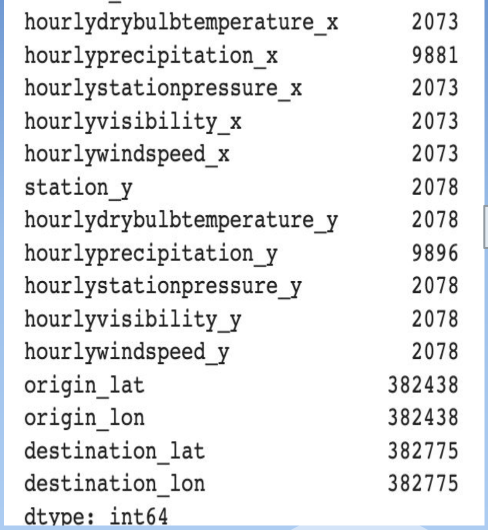
Flight and Delayed reasons other than weather conditions

Kept	**Dropped**
Origin_airport, origin_lat, origin_lon,	Id, carrier code
'Destination_airport, destination_lat, destination_lon	delay_national_aviation_system', 'delay_security
'departure_delay	delay_security
'arrival_delay'	delay_late_aircarft_arrival'

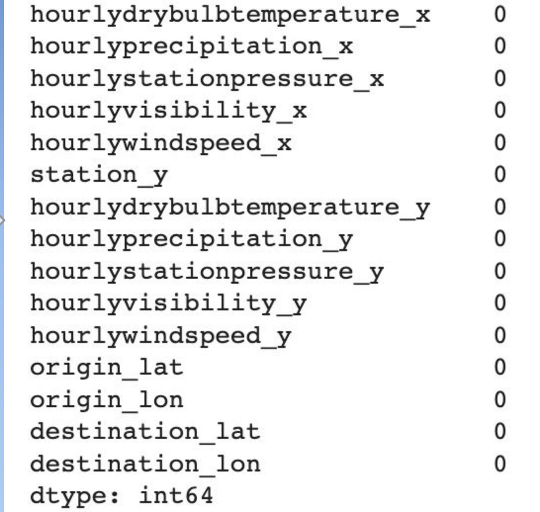
Weather Parameters

Kept	
Hourlydrybulbtemperature_x hourlyprecipitation_x hourlystationpressure_x hourlyvisibility_x hourlywindspeed_x	hourlydrybulbtemperature_y hourlyprecipitation_y hourlystationpressure_y hourlyvisibility_y hourlywindspeed_y

Removal of Missing Values

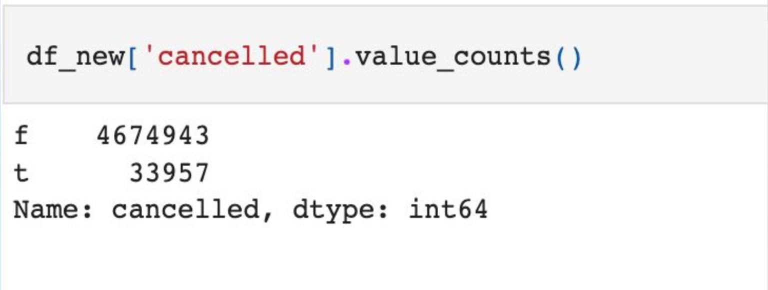


```
hourlydrybulbtemperature_x      2073
hourlyprecipitation_x           9881
hourlystationpressure_x         2073
hourlyvisibility_x              2073
hourlywindspeed_x               2073
station_y                       2078
hourlydrybulbtemperature_y      2078
hourlyprecipitation_y           9896
hourlystationpressure_y         2078
hourlyvisibility_y              2078
hourlywindspeed_y               2078
origin_lat                    382438
origin_lon                    382438
destination_lat               382775
destination_lon               382775
dtype: int64
```



```
hourlydrybulbtemperature_x     0
hourlyprecipitation_x          0
hourlystationpressure_x        0
hourlyvisibility_x             0
hourlywindspeed_x              0
station_y                      0
hourlydrybulbtemperature_y     0
hourlyprecipitation_y          0
hourlystationpressure_y        0
hourlyvisibility_y             0
hourlywindspeed_y              0
origin_lat                     0
origin_lon                     0
destination_lat                0
destination_lon                0
dtype: int64
```

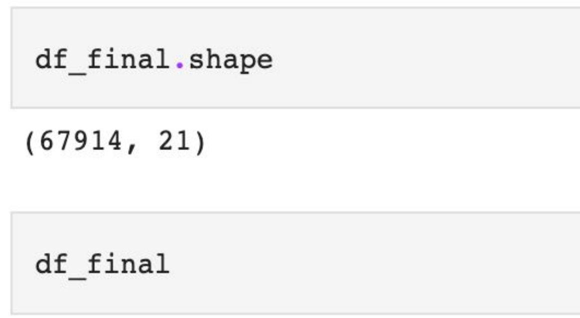
Handling imbalance in the data



```
df_new['cancelled'].value_counts()
```

```
f    4674943
t      33957
Name: cancelled, dtype: int64
```

After Undersampling

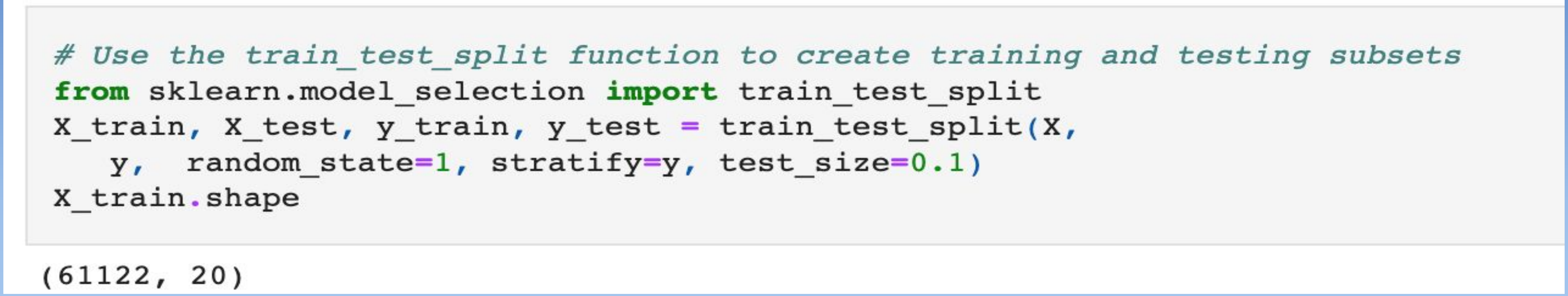


```
df_final.shape
```

```
(67914, 21)
```

```
df_final
```

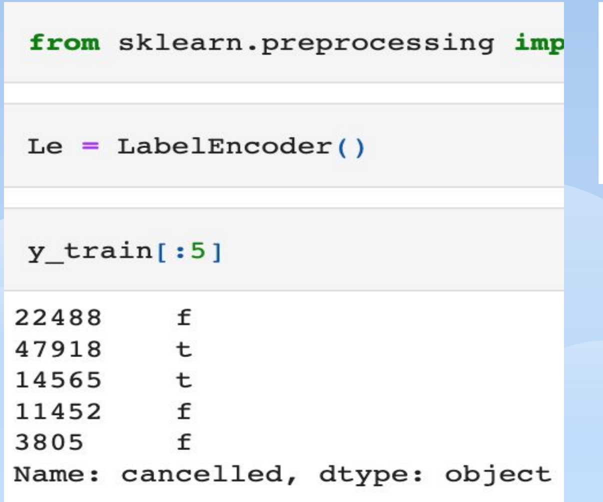
Splitting into training and test datasets



```
# Use the train_test_split function to create training and testing subsets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,
   y,  random_state=1, stratify=y, test_size=0.1)
X_train.shape
```

```
(61122, 20)
```

Label Encoding:

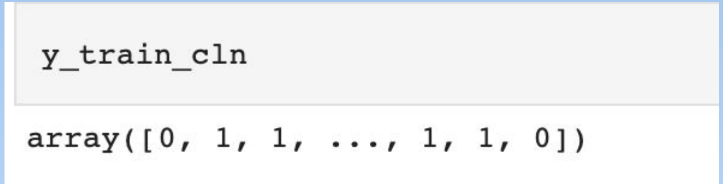


```
from sklearn.preprocessing imp
```

```
Le = LabelEncoder()
```

```
y_train[:5]
```

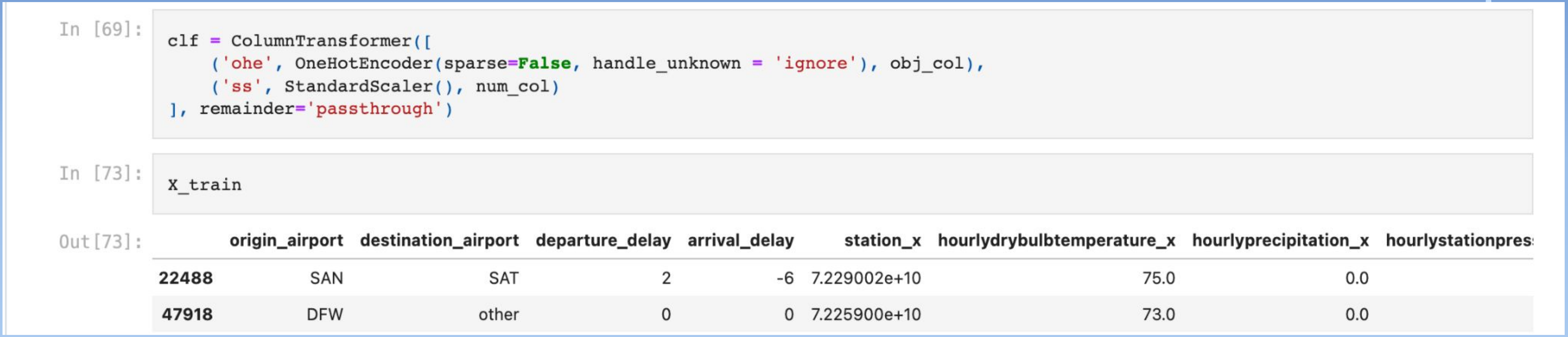
```
22488    f
47918    t
14565    t
11452    f
3805     f
Name: cancelled, dtype: object
```



```
y_train_cln
```

```
array([0, 1, 1, ..., 1, 1, 0])
```

Using Standard Scaler and Column Transformer



In [69]:

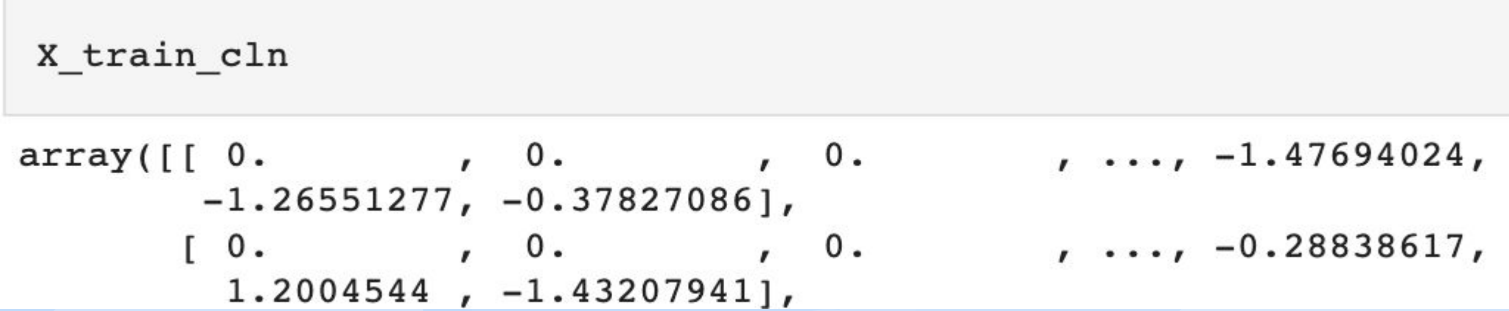
```
clf = ColumnTransformer([
    ('ohe', OneHotEncoder(sparse=False, handle_unknown = 'ignore'), obj_col),
    ('ss', StandardScaler(), num_col)
], remainder='passthrough')
```

In [73]:

```
X_train
```

Out[73]:

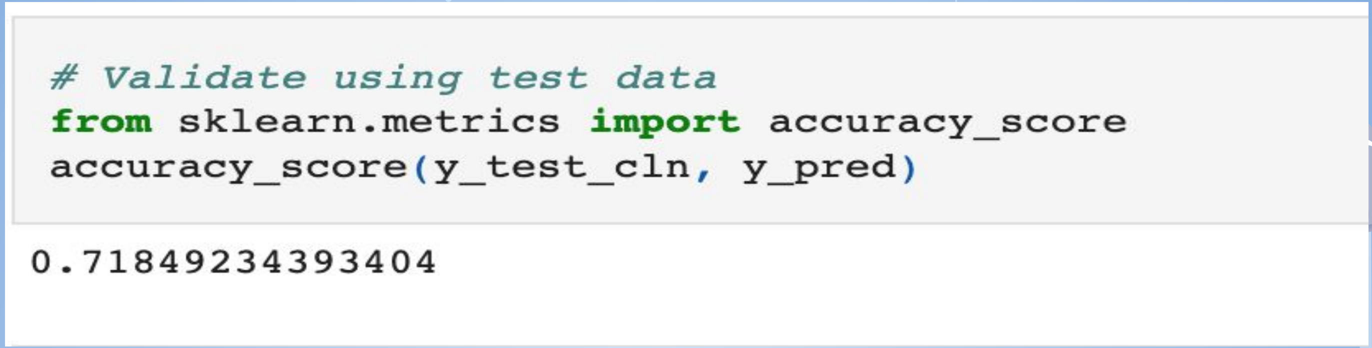
	origin_airport	destination_airport	departure_delay	arrival_delay	station_x	hourlydrybulbtemperature_x	hourlyprecipitation_x	hourlystationpres
22488	SAN	SAT	2	-6	7.229002e+10	75.0	0.0	
47918	DFW	other	0	0	7.225900e+10	73.0	0.0	



```
X_train_cln
```

```
array([[ 0.        ,  0.        ,  0.        , ..., -1.47694024,
        -1.26551277, -0.37827086],
       [ 0.        ,  0.        ,  0.        , ..., -0.28838617,
         1.2004544 , -1.43207941],
```

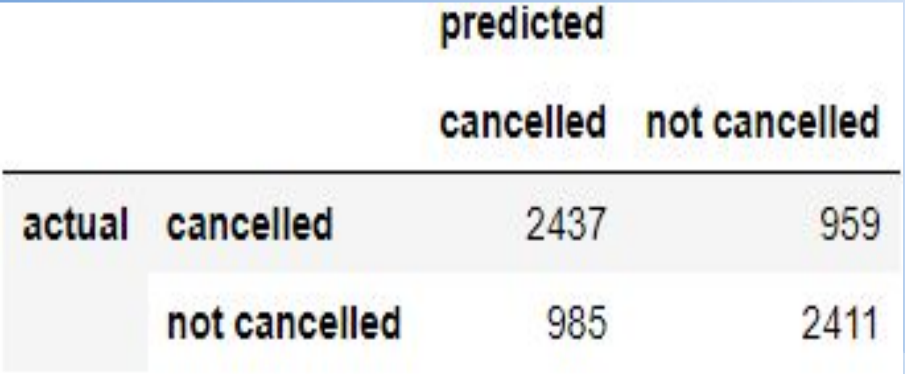
Logistic Regression



```
# Validate using test data
from sklearn.metrics import accuracy_score
accuracy_score(y_test_cln, y_pred)
```

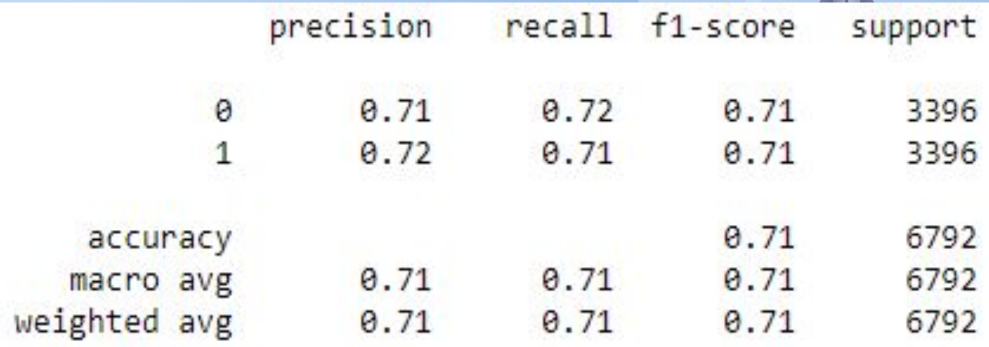
```
0.71849234393404
```

Confusion Matrix



		predicted	
		cancelled	not cancelled
actual	cancelled	2437	959
	not cancelled	985	2411

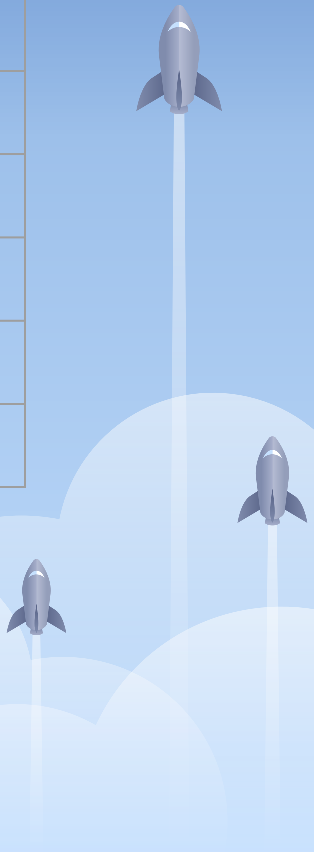
Classification Report



```
              precision    recall  f1-score   support

           0       0.71      0.72      0.71      3396
           1       0.72      0.71      0.71      3396

    accuracy                           0.71      6792
   macro avg       0.71      0.71      0.71      6792
weighted avg       0.71      0.71      0.71      6792
```

Machine-learning Trial Accuracy Results

Trial	Method	Accuracy
1	Logistic regression with undersampling	72%
2	Logistic regression with oversampling	71%
3	Random forest with undersampling	87%
4	Random forest with oversampling	99%
5	Random forest (raw)	99%



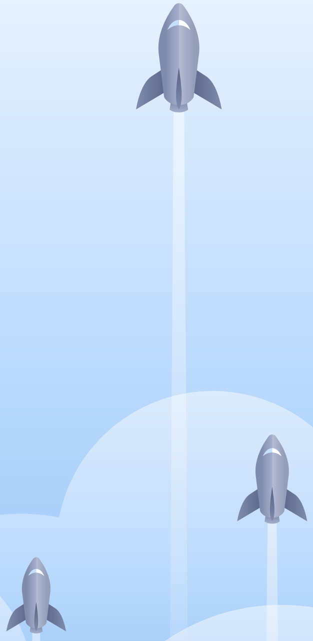
RYAN,

Data Viz Whiz

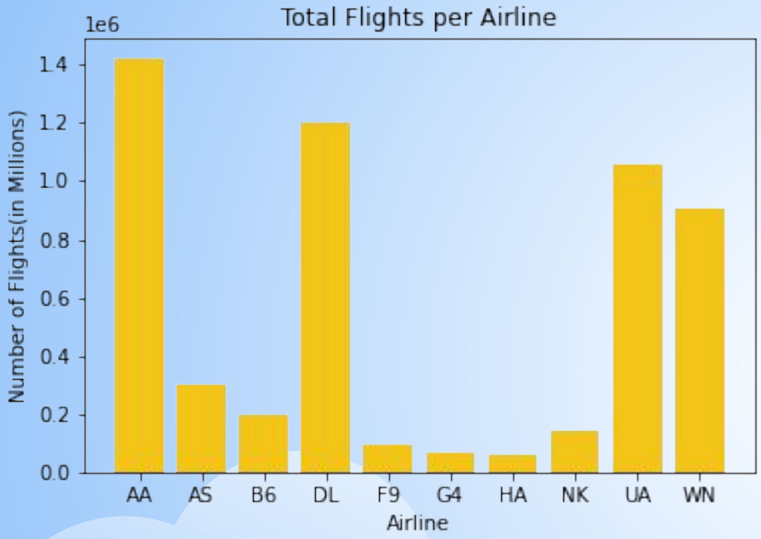
Visualization of analysis

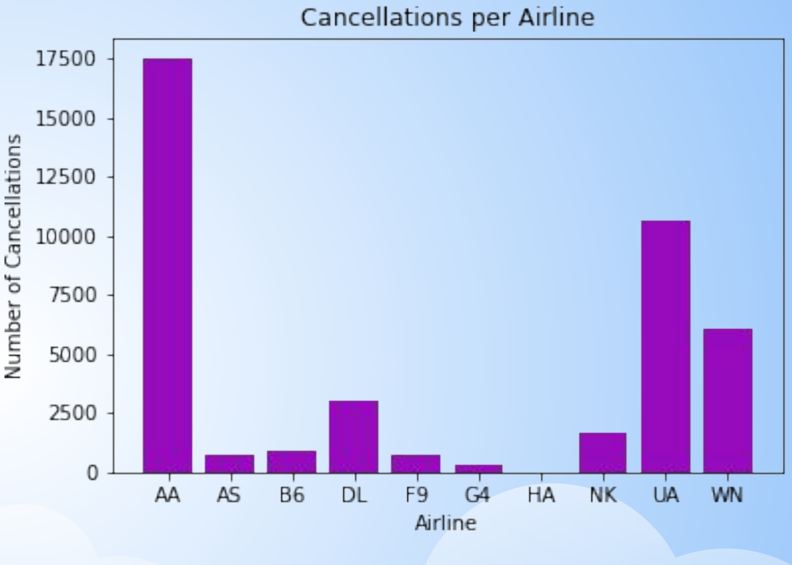
Recommendation for future analysis

Anything the team would have done differently



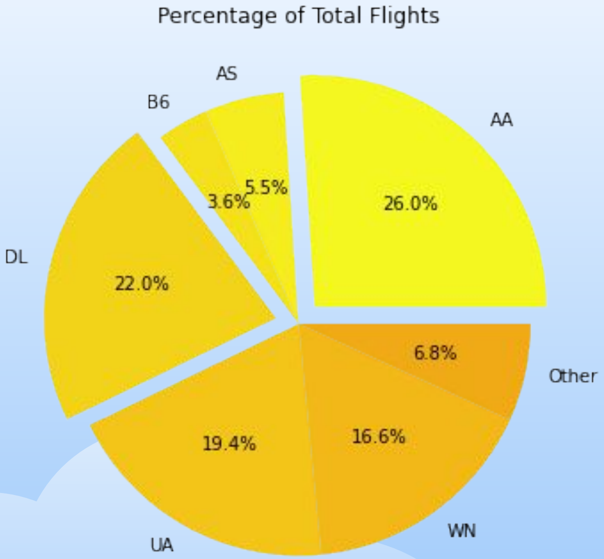
Airline Observations



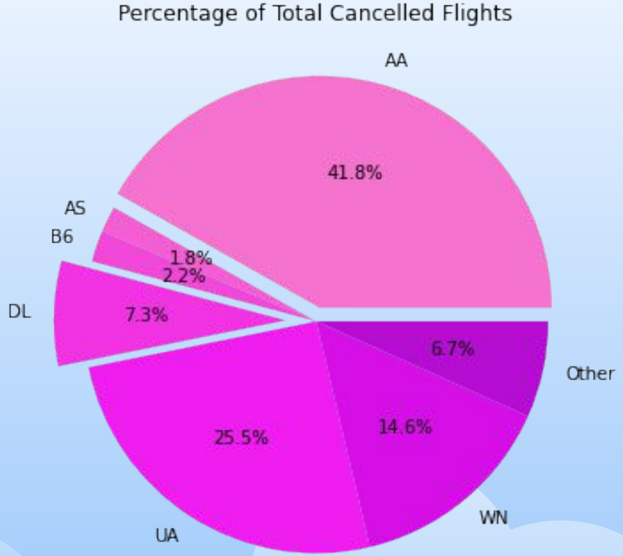


We noticed that although Delta Airlines (DL) is the second highest carrier by volume of total flights, they are fourth in number of cancellations.
We wanted to look into this further, so we looked at percentages of total flights vs. percentages of cancellations (next slide).

Airline Observations

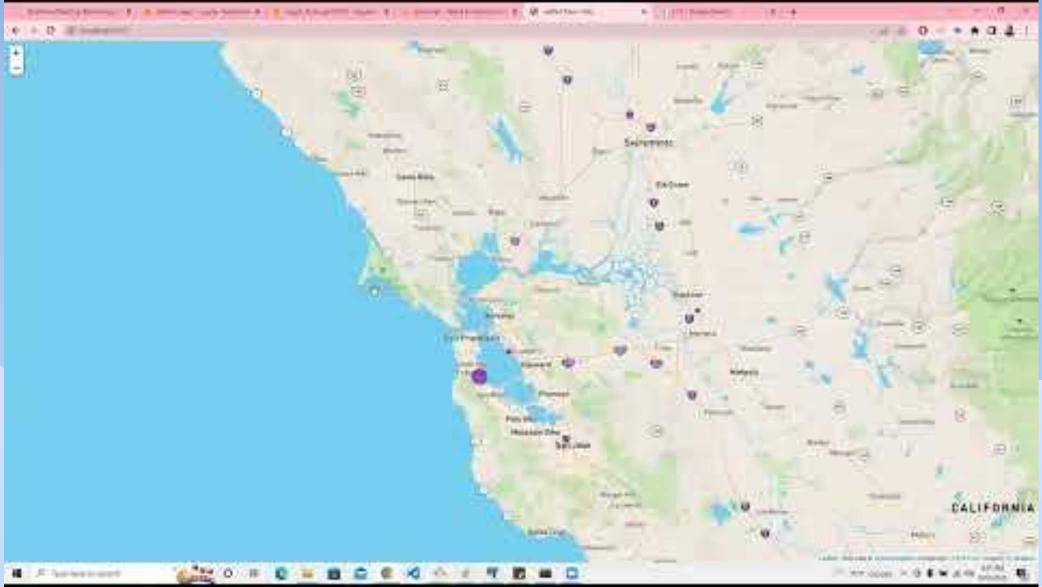


Percentage of Total Flights



Percentage of Total Cancelled Flights

American Airlines (AA) accounts for about 26% of total flight volume, but about 42% of cancellations due to weather, whereas Delta (DL) accounts for about 22% of total flights but only about 7% of cancelled flights due to weather.


Sacramento
Santa Rosa
San Francisco
Modesto
San Jose
CALIFORNIA

FINAL THOUGHTS

Recommendations for future analysis:

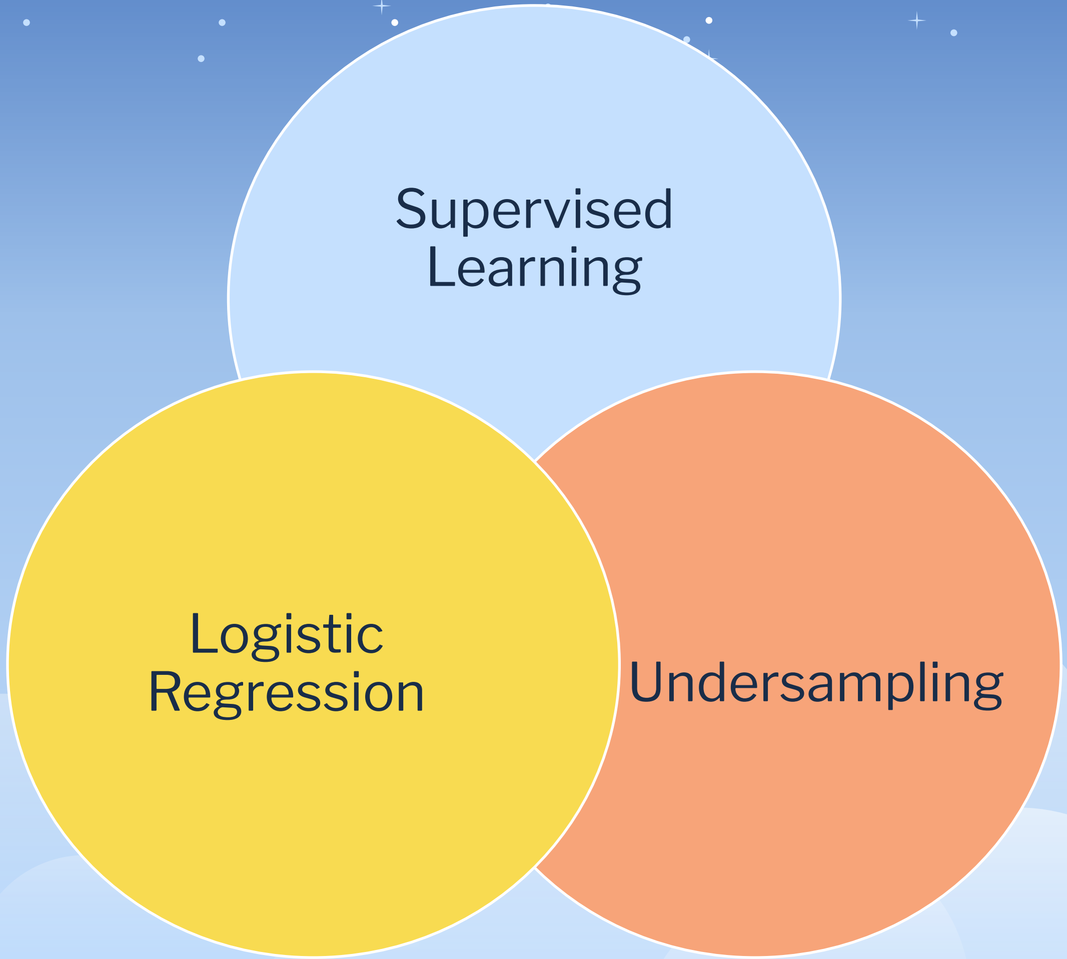
1. Use Random Forest to determine feature importance to analyze the results and determine if we have overfitted the model
2. Add additional functionality to the interactive map
3. Additional analysis to explore causes for higher cancellation rates in some airlines/airports
4. Source a cleaner dataset for airport data

QUESTIONS ?

Thank You!

BUSINESS
Economy of the European Union
25 great jobs for people who love to travel
Business Company
INVOICE

Machine Learning Model



Supervised Learning

Logistic Regression

Undersampling

Datasets and Airline Codes

Airports for which weather data was available.

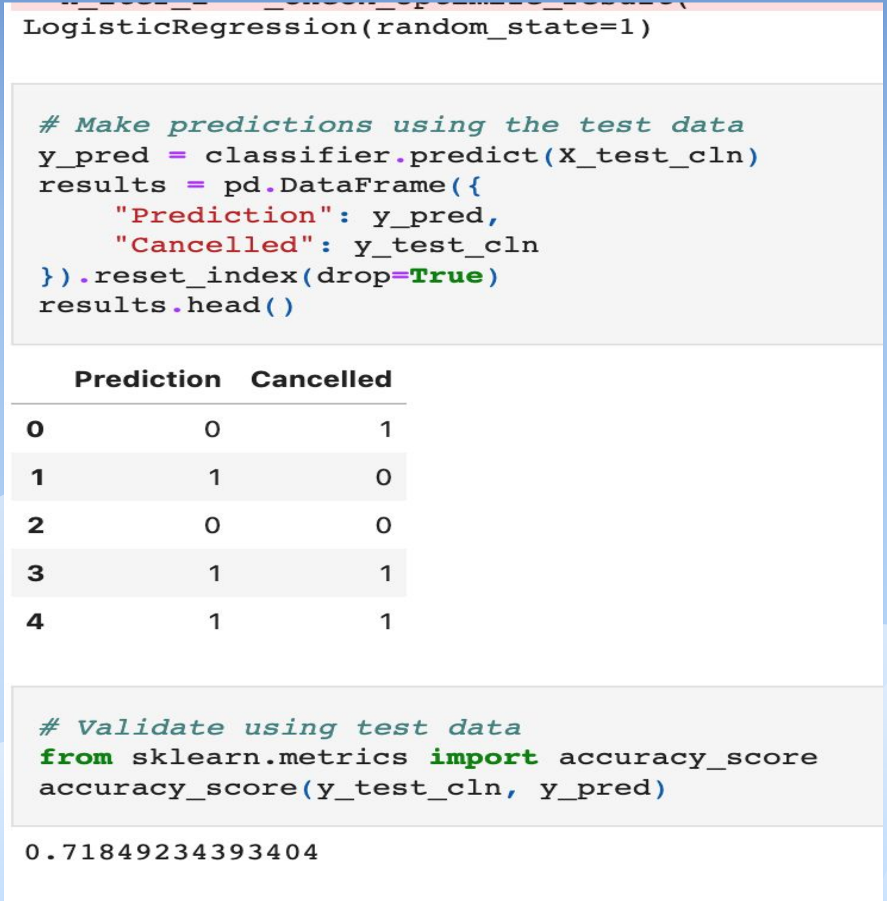
AA	American Airlines	G4	Allegiant Air
AS	Alaska Airlines Inc	HA	Hawaiian Airlines Inc
B6	JetBlue Airways	NK	Spirit Air Lines
DL	Delta Air Lines Inc	UA	United Air Lines Inc
F9	Frontier Airlines Inc	WN	Southwest Airlines Co

Cancellation Codes

A	Carrier Caused
B	Weather
C	National Aviation System
D	Security

Note: N is not on the list and represents "None" or "Not cancelled".

Using Binary Classification and Running Logistic Regression



```
LogisticRegression(random_state=1)
```

```python
# Make predictions using the test data
y_pred = classifier.predict(X_test_cln)
results = pd.DataFrame({
    "Prediction": y_pred,
    "Cancelled": y_test_cln
}).reset_index(drop=True)
results.head()
```

	Prediction	Cancelled
0	0	1
1	1	0
2	0	0
3	1	1
4	1	1

```python
# Validate using test data
from sklearn.metrics import accuracy_score
accuracy_score(y_test_cln, y_pred)
```

```
0.71849234393404
```