Predicting Flight Delays

Lighthouse Labs Mid-term Project

Ruslan and Christian

Objective

- The task of the project is to create a Machine Learning model that predicts flight delays one week in advance.
- This is a regression problem, in which we wish to predict the continuous variable "arrival delay" for each flight.
- We are supplied with a PostgreSQL Database hosted on an AWS server with around 15 million rows of information on US flights for 2018 and 2019.

Extract, Transform, Load

```python
def import_flights( dateFrom, dateTill, filename, chunksize=100000, carrier='%' ):
    '''
    Connects to the database and saves csv file.

    Args:
        carrier as string - if omitted wildcard is set by default
        dateFrom as string in format YYYY-mm-dd
        dateTill as string in format YYYY-mm-dd
        chunksize as integer is used in LIMIT and OFFSET parts of SQL to query database. Helps avoiding freezing during query.
        filename as string
    '''

    dateFrom = datetime.strptime(dateFrom, '%Y-%m-%d')
    dateTill = datetime.strptime(dateTill, '%Y-%m-%d')



    #Establishing connection
    conn = psycopg2.connect(
    host="",
    database="",
    user="",
    password="")

    #Refined list of columns
    list_of_columns = ['fl_date',  'mkt_carrier',
        'mkt_carrier_fl_num',  'tail_num',
        'op_carrier_fl_num', 'origin_airport_id', 'origin', 'origin_city_name',
        'dest_airport_id', 'dest', 'dest_city_name', 'crs_dep_time', 'dep_time',
        'dep_delay', 'taxi_out', 'wheels_off', 'wheels_on', 'taxi_in',
        'crs_arr_time', 'arr_time', 'arr_delay', 'cancelled',
        'cancellation_code', 'diverted', 'dup', 'crs_elapsed_time',
        'actual_elapsed_time', 'air_time', 'flights', 'distance',
        'carrier_delay', 'weather_delay', 'nas_delay', 'security_delay',
        'late_aircraft_delay', 'first_dep_time', 'total_add_gtime',
        'longest_add_gtime', 'no_name']

    #Concatenating columns into one string for SQL Query
    cols=""
    for col in list_of_columns:
        if cols:
            cols=cols + ", " + col
        else:
            cols=col

    #Iterating over data till nothing is returned

    offset = 0


    while True:

        sql="""SELECT """ + cols + """ FROM flights
        WHERE fl_date >='""" + dateFrom.strftime('%Y%m%d') + """' and fl_date <'""" + dateTill.strftime('%Y%m%d') + """'
            and mkt_carrier LIKE '"""  + carrier + """' ORDER BY fl_date LIMIT """ + str(chunksize) + """ OFFSET """ + str(offset) +""";"""

        #Quering data from database
        df = pd.read_sql(sql, conn)

        if df.shape[0] == 0:
            break #No more data. Quiting the loop.

        #Verifying file csv
        if path.exists(filename + ".csv"):
            #appending
            df.to_csv(filename + ".csv", mode='a', header=False, index=False)
        else:
            #creating file
            df.to_csv(filename + ".csv",  index=False)

        offset +=chunksize
    conn.close()
```

```python
import_flights('2019-01-01', '2019-12-31', 'All flights 2019', chunksize=200000, carrier='%')
```

Built python function to easily pull from the PostgreSQL Database

First Approach

- Our EDA did not find any real correlation between our target delay and our features.
- Decided to use a "Naive" first approach.
- Attempted Linear Regression on default settings, with *no* feature engineering.
- Baseline model to compare our efforts to: **Mk 0**.

```
metrics.r2_score(y_test,y_pred)

0.00863460962832796
```

Second Approach

Focusing on Feature Engineering

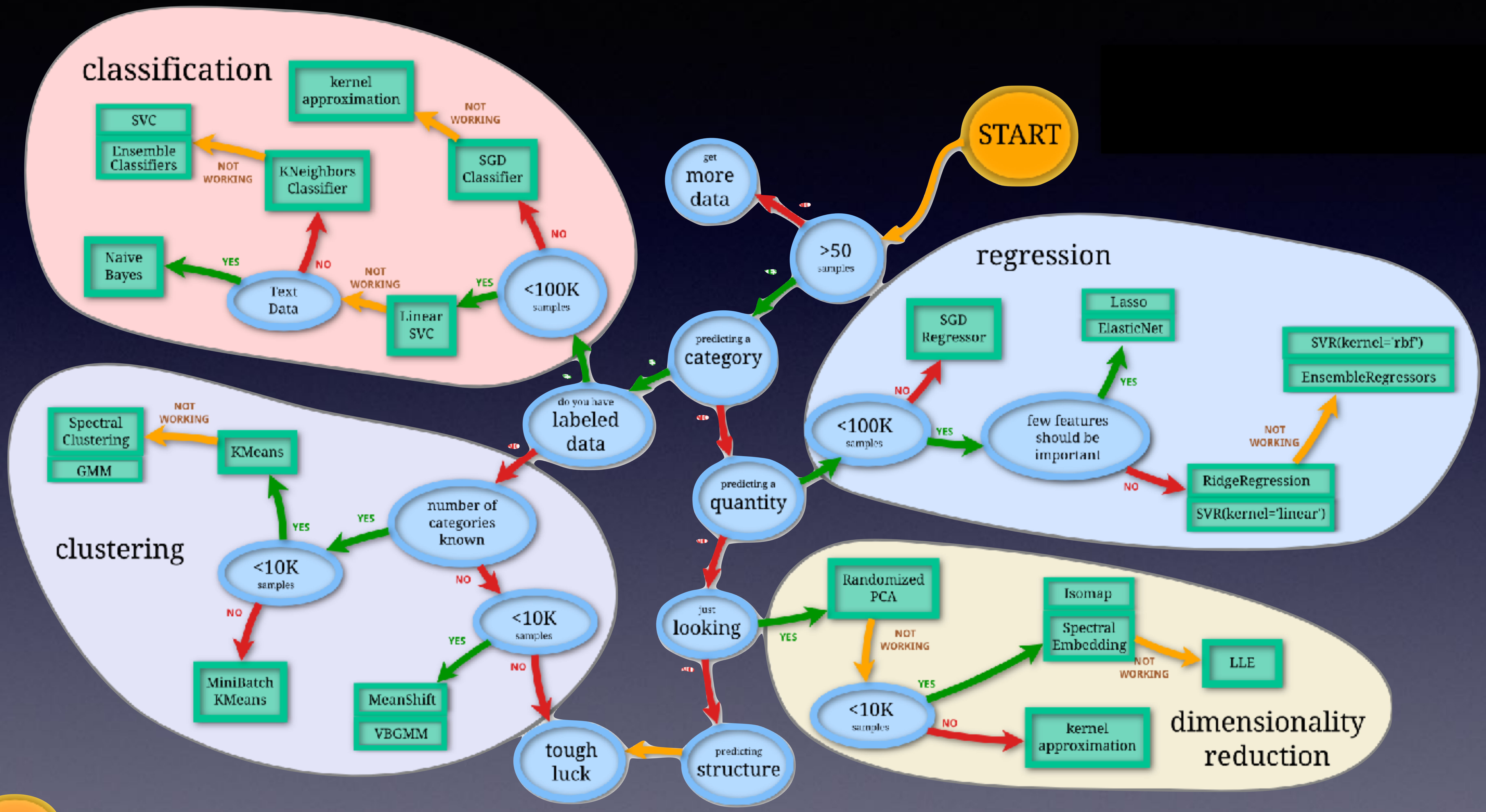
Mk I Model

- Taxi Rolling Average
- Traffic Rolling Average
- Month Dummies
- Weather API
- Ordinal Dates
- Departure Delay Rolling Average

Linear Regression

R2 score : 0.0900

Mk II Model Selection



Back

Mk II Model

- Taxi Rolling Average
- Traffic Rolling Average
- Month Dummies
- Weather API
- Ordinal Dates
- Departure Delay Rolling Average

Stochastic Gradient Descent

R2 score : 0.0892

Mk III Model

- Taxi Rolling Average
- Holidays
- Traffic Rolling Average
- Departure Delay
Per Tail Number (rolling average)
- Month Dummies
- Departure Delay per
Origin Airport ID (rolling average)
- Weather API
- Taxi out/in per Carrier
(Rolling average)
- Ordinal Dates
- Arrival delay per: airport, tail number, carrier
(Rolling average)
- Departure Delay
Rolling Average
- Polynomial Features
(Trial and Error)

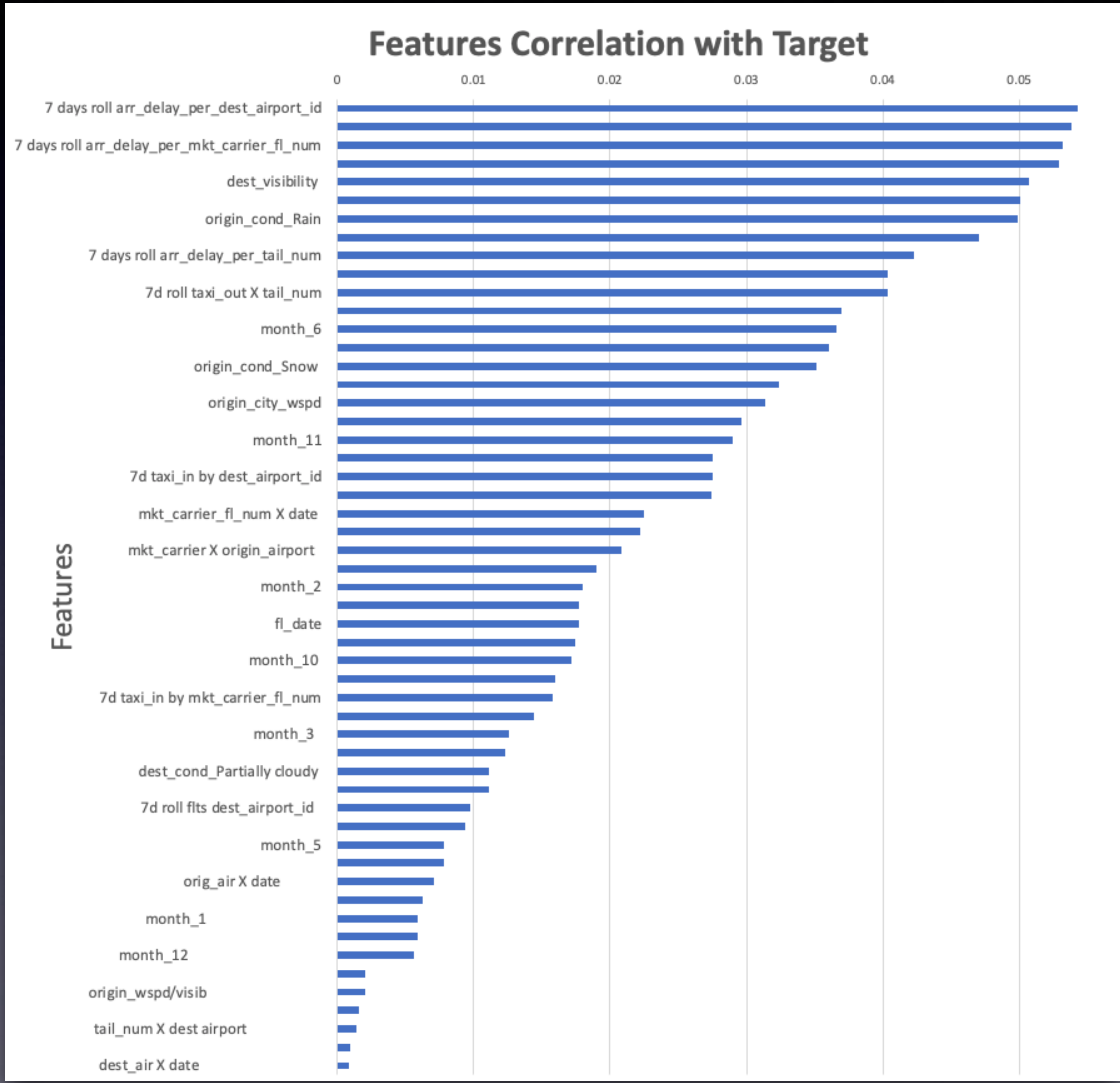
XGBoost
(Manual Tuning)

R2 score : 0.1112

Note: 7 Day rolling averages were offset 7 days back.

Feature Correlation

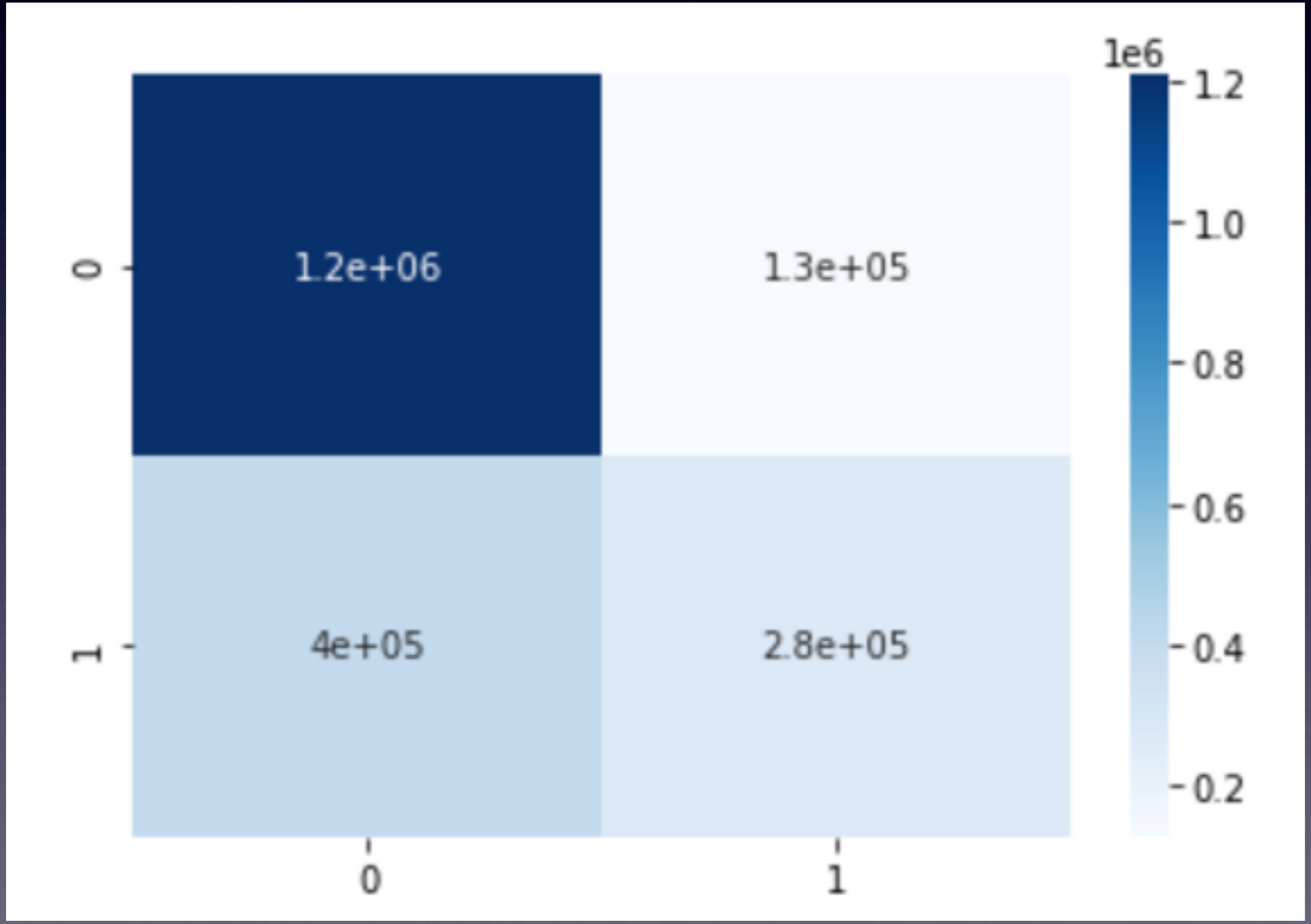
7 days roll arr_delay_per_dest_airport_id	0.05427
origin_visibility	0.05381
7 days roll arr_delay_per_mkt_carrier_fl_nu	0.05322
dest_cond_Rain	0.05288
dest_visibility	0.05071
7 days roll arr_delay_per_origin_airport_id	0.05002
origin_cond_Rain	0.04988
7 days roll dep_time	0.047
7 days roll arr_delay_per_tail_num	0.04225
7d taxi_out by origin_airport_id	0.04033
7d roll taxi_out X tail_num	0.04033
7 days roll dep_delay_per_tail_num	0.03694
month_6	0.03663
7d taxi_out by mkt_carrier_fl_num	0.03607
origin_cond_Snow	0.03516
dest_city_wspd	0.0324
origin_city_wspd	0.03135
month_9	0.02959
month_11	0.02901
7d roll taxi_in X tail_num	0.02756
7d taxi_in by dest_airport_id	0.02756
dest_cond_Snow	0.02745
mkt_carrier_fl_num X date	0.02245



Binary Classification (YES/NO)

- **ACCURACY: 0.7360**
- **PRECISION: 0.6833**
- **RECALL: 0.4082**

ACTUAL



PREDICTED