Predicting NYC Taxi ETA

Liyan Nie
Estelle Danilo

Data

- 2016 NYC yellow taxi data (BigQuery)
- 2016 NYC weather data (BigQuery)

Data Cleaning

Missing values

Taxi data:

- Remove rows

	missing_ratio
dropoff_latitude	47.11865
dropoff_longitude	47.11865
pickup_latitude	47.11865
pickup_longitude	47.11865

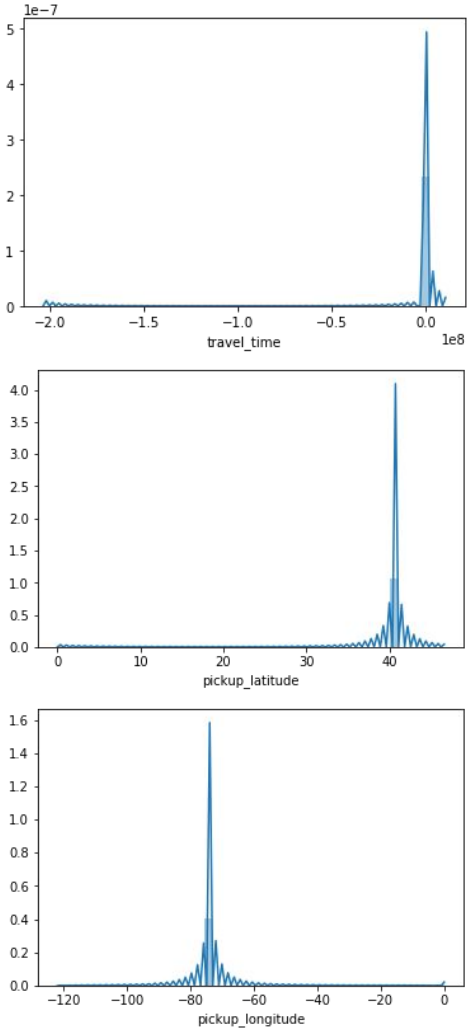
Weather data

- Remove missing rows
- Remove “sndp”

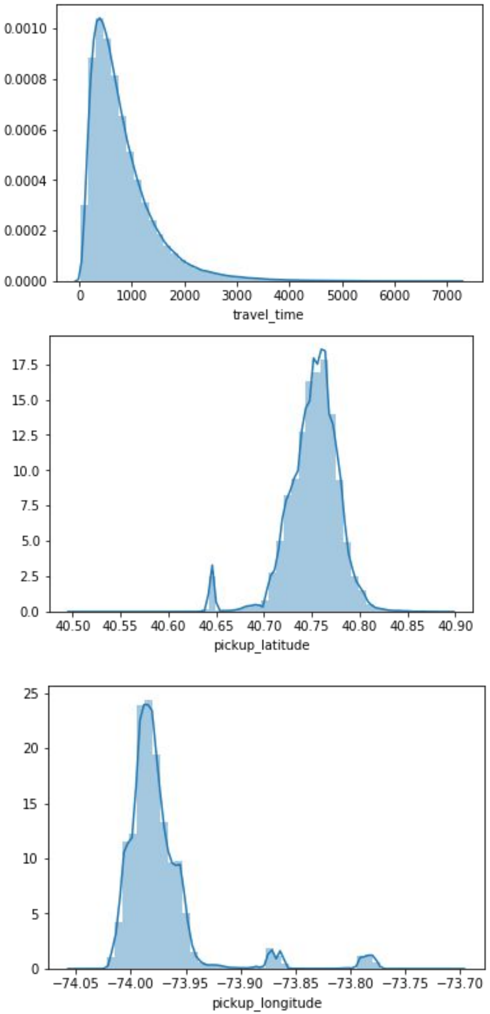
	missing_ratio
visib	2.30176
wdsp	2.86897
gust	28.1718
sndp	94.4242

Outliers

- Remove passenger_count == 0
- Remove rate_code == 99 (scale is 1-6 only)
- Only keep lat in [40.6, 40.9]
- Only keep long in [-74.05, -73.7]
- Only keep travel_time between 30s and 7200s (2 hours)



Remove outliers



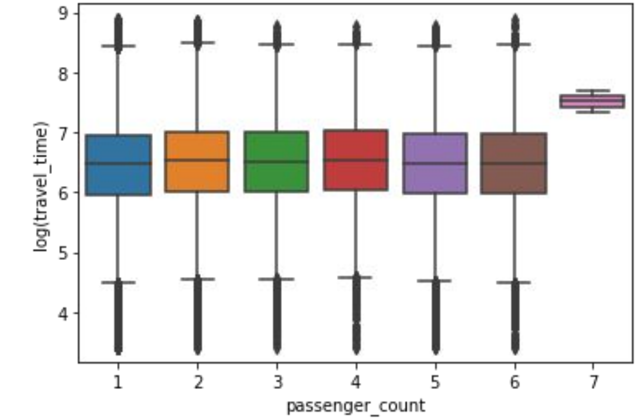
EDA & Identifying Features

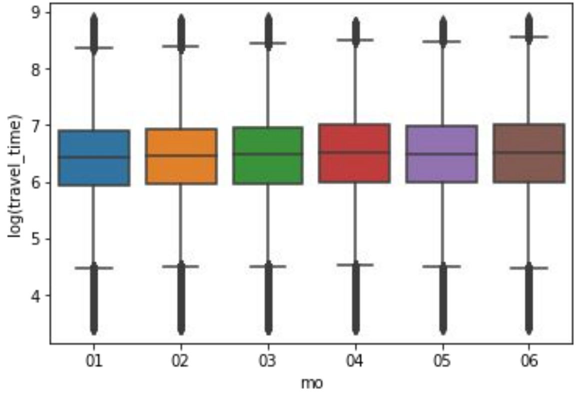
Information leakage

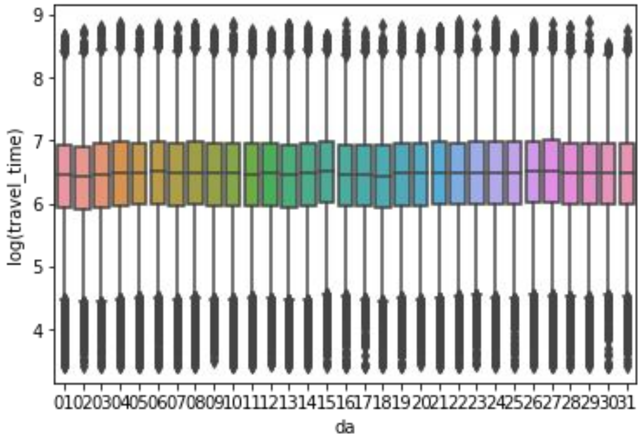
Remove:

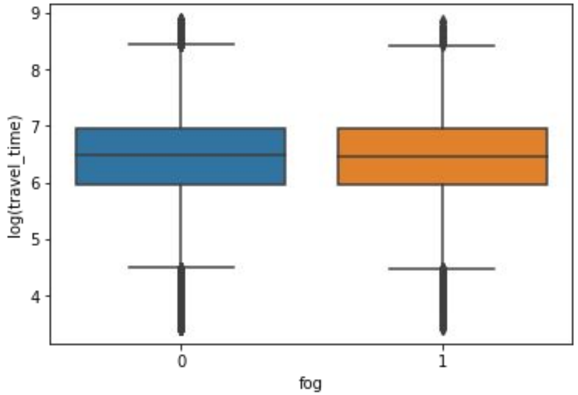
- Trip_distance
- Fare_amount
- Total_amount
- Payment_type
- Extra
- Mta_tax
- Tip_amount
- Tolls_amount
- imp_surcharge

Visualize response vs variables

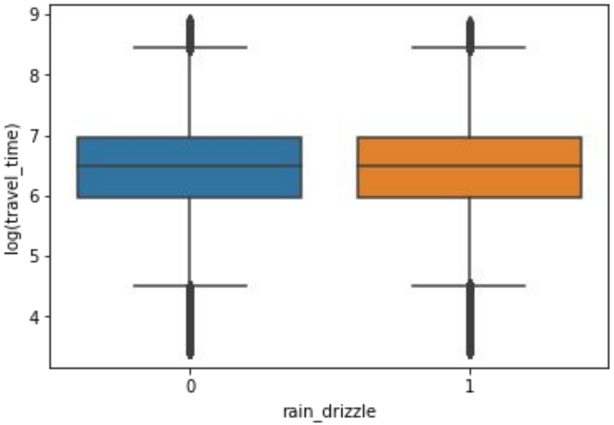


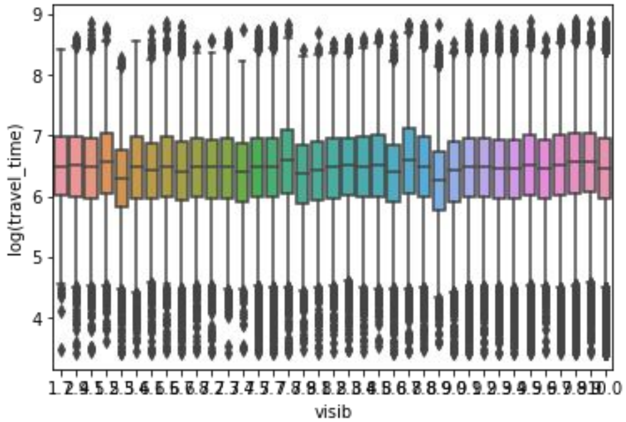


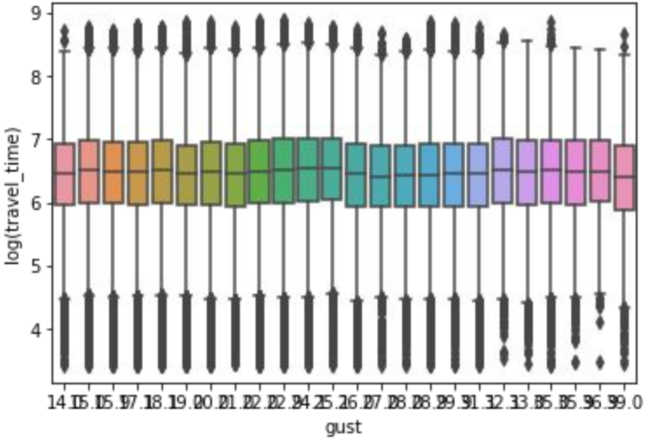


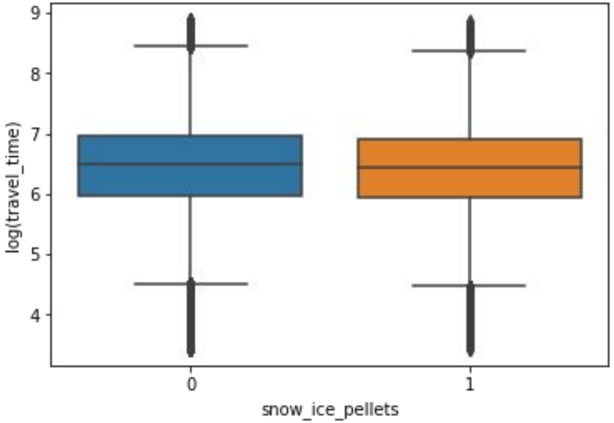


Visualize response vs variables



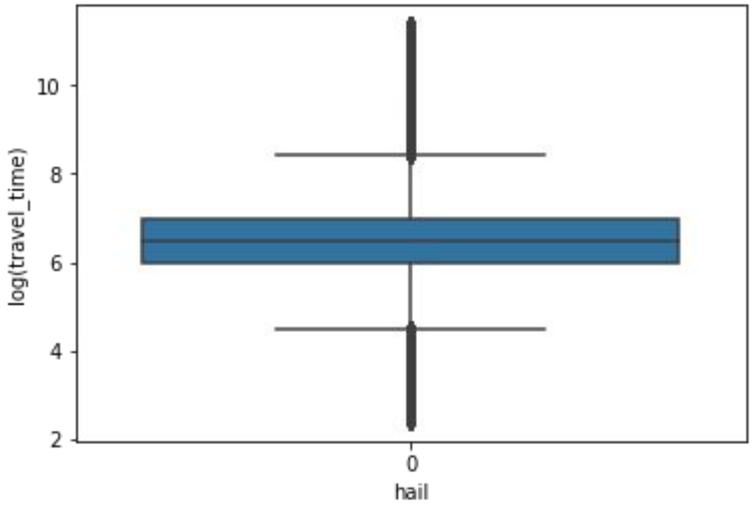


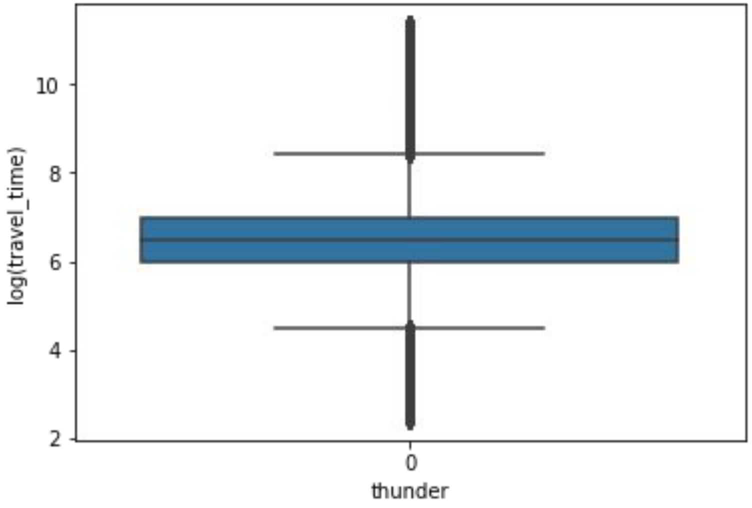




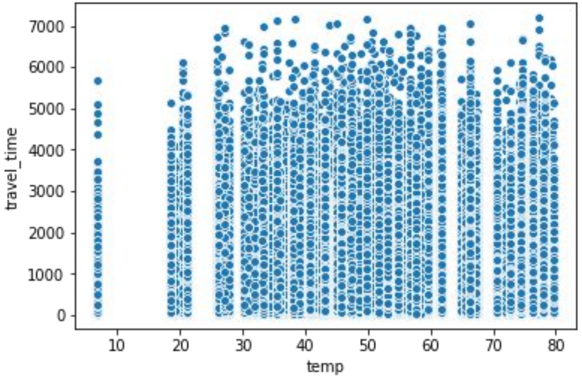
Visualize response vs variables

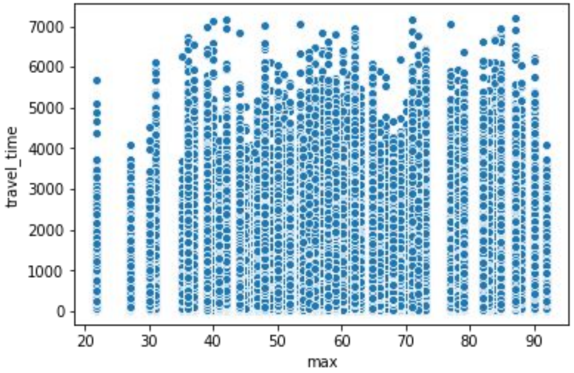
Remove

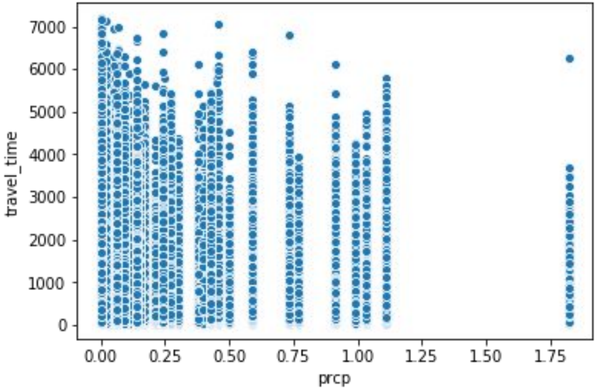


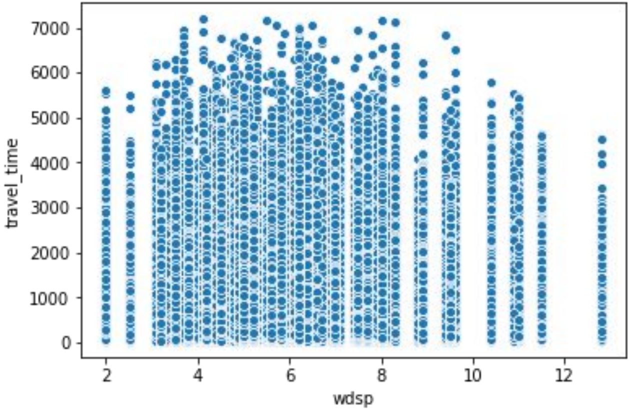


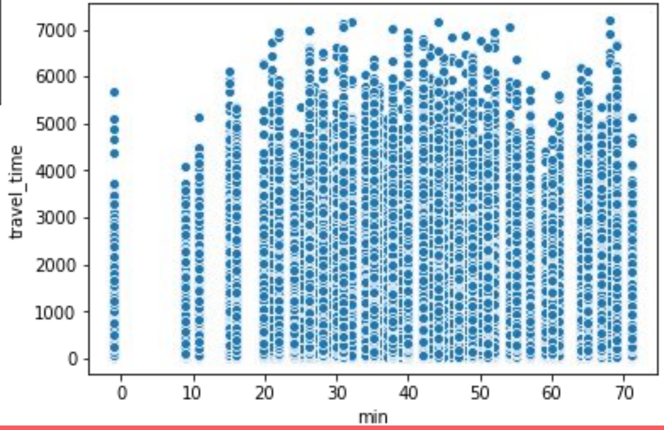
Visualize response vs variables











EDA

Preliminary conclusion:

- None of these features alone seem to have strong prediction power. Need further feature engineering
- But first, run a few models with minimal preprocessing/feature engineering to get a baseline idea

Baseline Modeling

Identifying Features

Using all possible **non-leaking** features, given no strong signal in EDA

Location

- pickup_latitude
- pickup_longitude
- dropoff_longitude
- dropoff_latitude

Time

- day_of_year
- month_of_year

Type of Taxi Trip

- vendor_id
- passenger_count
- rate_code
- store_and_fwd_flag
- payment_type

Features: Continuous vs. Categorical

Categorical (12 levels or less)

- vendor_id
- store_and_fwd_flag
- payment_type

- rate_code (**continuous by default**)
- passenger_count (continuous by default)
- month_of_year (continuous by default)

OneHotEncode (dropping first level)

Continuous

- day_of_year
- pickup_longitude
- pickup_latitude
- dropoff_longitude
- dropoff_latitude

Data Splitting

Goal: Find split by **pickup_datetime** from January to June 2016 (where lat-long exist) that creates a 80-20% train/test split

Method: Training = pickup_datetime > 2016-01-01 & < 2016-**05-25**
Test set = pickup_datetime >= 2016-**05-25** & < 2016-07-01

Result: training and test splitted into **X_train, X_test** (features) and **y_train, y_test** (travel_time)

Baseline Models:

1) Decision Tree

To prevent overfitting

2) Bagging

average out trees predictions

3) Random Forest

train on different samples of data

To split each node in tree:

All features considered

only **best split features** from a random subset of features

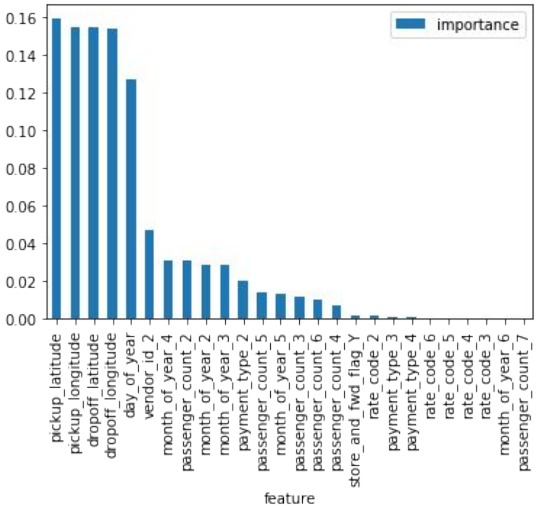
Baseline Model Performance

- Low 5-fold cross validated test performance
- **Bagging** least underperforming

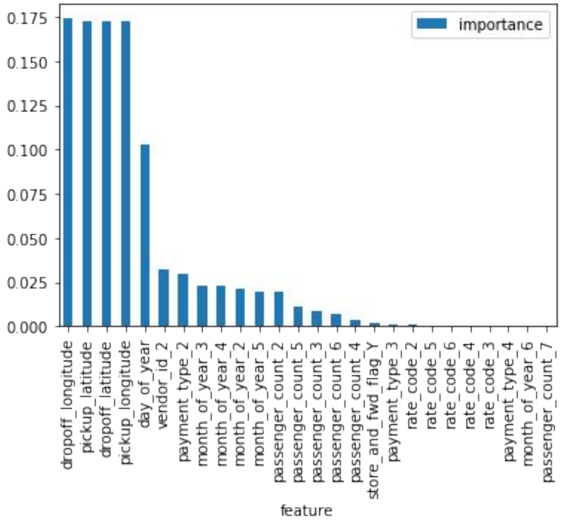
	Decision Tree	Bagging Classifier	Random Forest
R2	-16	-15	-52
RMSE	4141	3301	3987
RMSLE	1.07	1.28	1.18
Training Subsample Size	200,000	50,000	50,000
Testing Subsample Size	50,000	12,500	12,500

Baseline Feature Importance

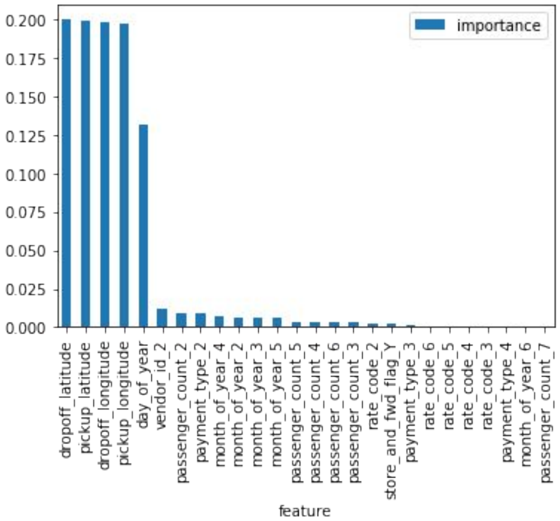
Decision Tree



Bagging Classifier



Random Forest



- Location features most important
- Similar feature and magnitude importance across models

GridSearched Baseline Models (5-fold)

Hyperparameter Tuning

1) Decision Tree

max_depth: [5,10],
min_samples_leaf: [10, 100],
min_samples_split : [50,200,500]

2) Bagging

n_estimators: [10, 50],
max_samples' : [0.1, 0.5]

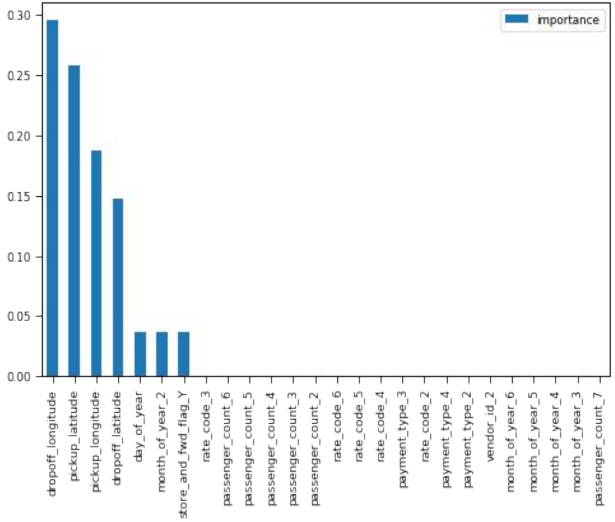
3) Random Forest

n_estimators : [100, 200],
max_depth : [5,10]

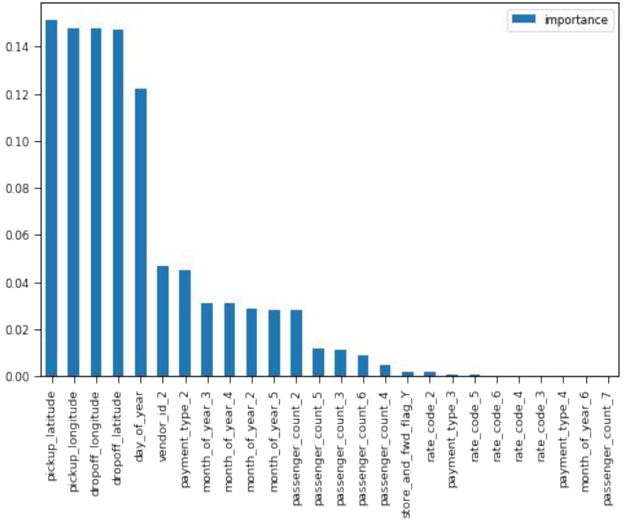
	Decision Tree	Bagging Classifier	Random Forest ✔
R2	-0.38	-1.58	-0.12
RMSE	3053	3484	2390
RMSLE	1	1.32	0.87

Feature Importance

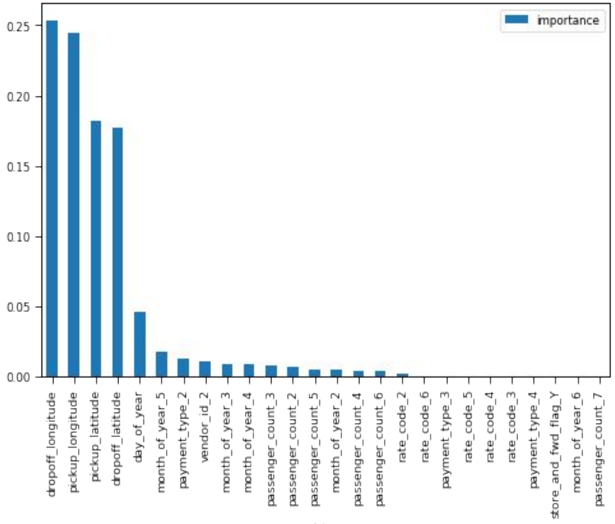
1) Decision Tree



2) Bagging



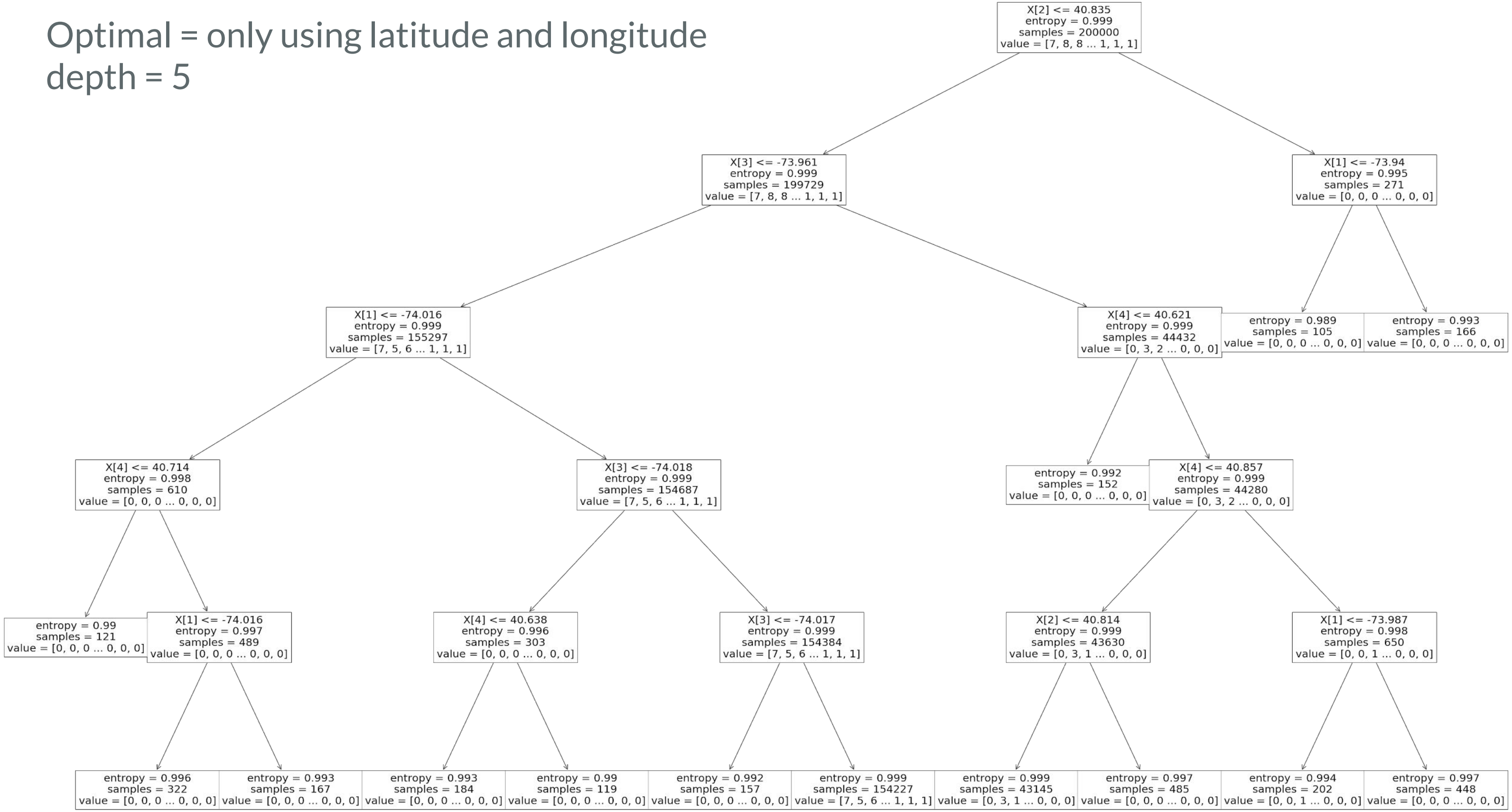
3) Random Forest



- Location features remain the most important
- More variance in magnitude in feature importances across models

GridSearched Decision Tree

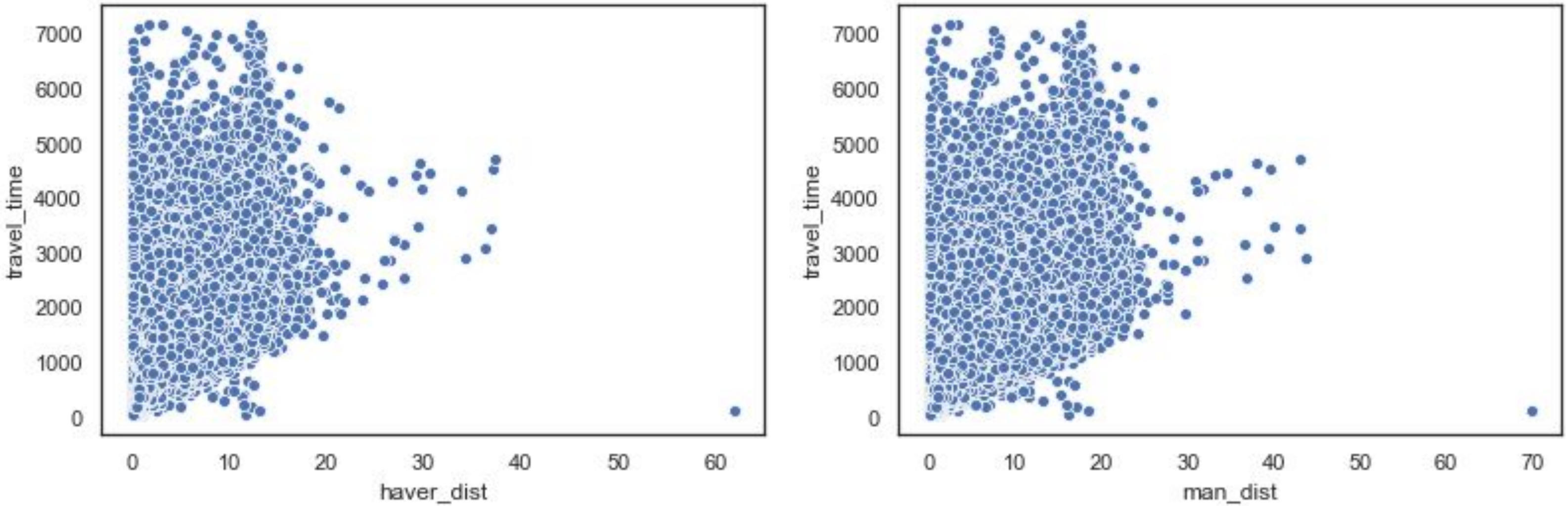
- Optimal = only using latitude and longitude
- depth = 5



Feature Engineering

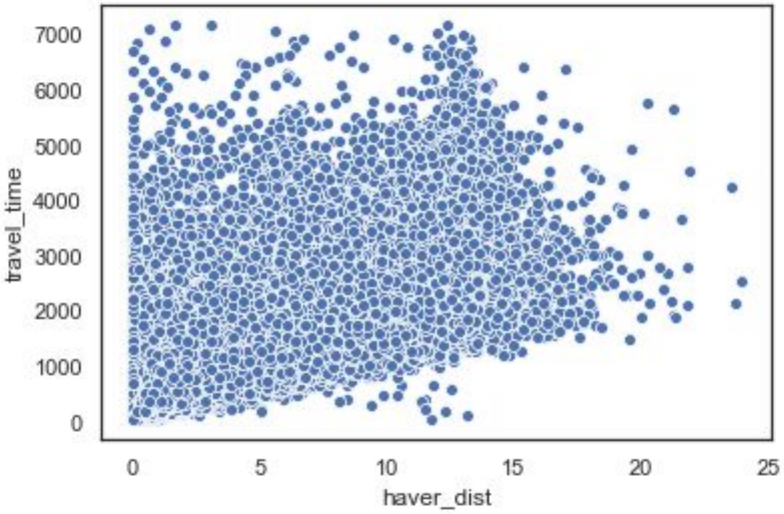
Distance

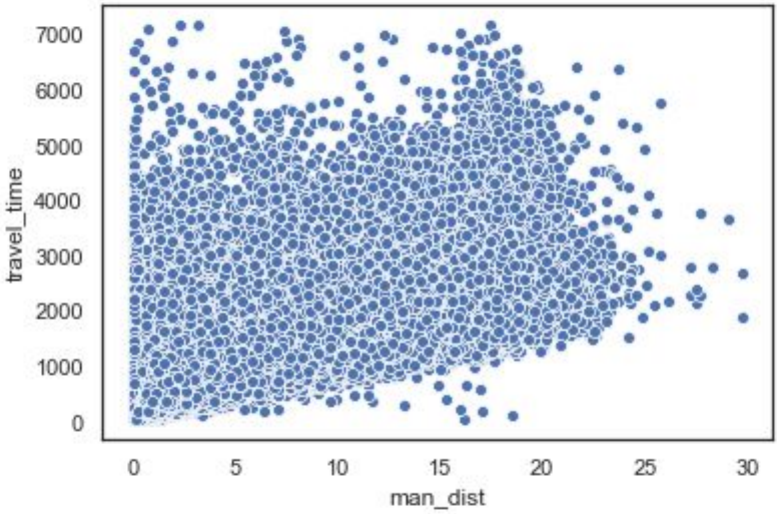
Potentially strong predictors!



Distance

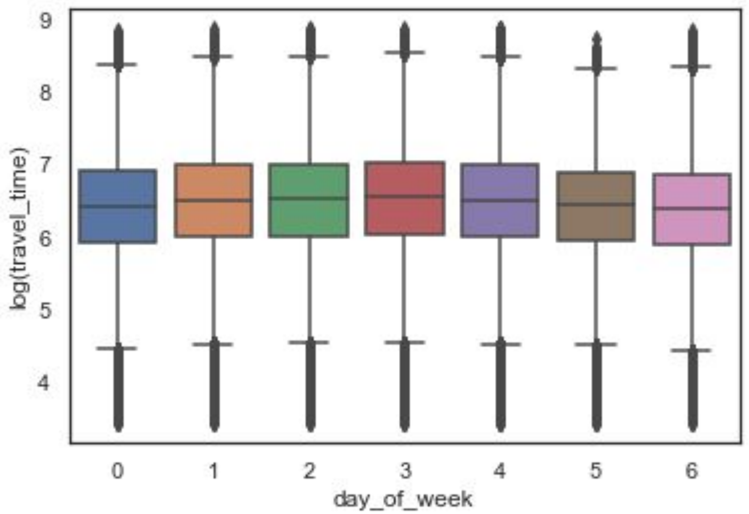
Further remove outliers: Haversine > 25, and Manhattan > 30

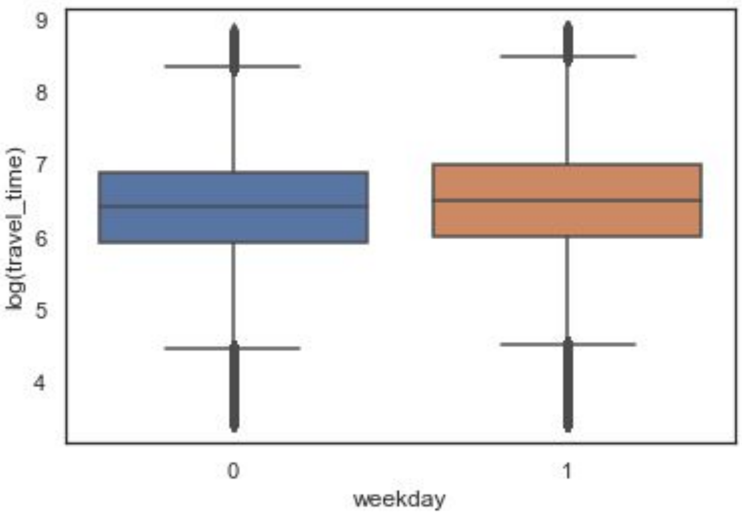




Day of week and weekday

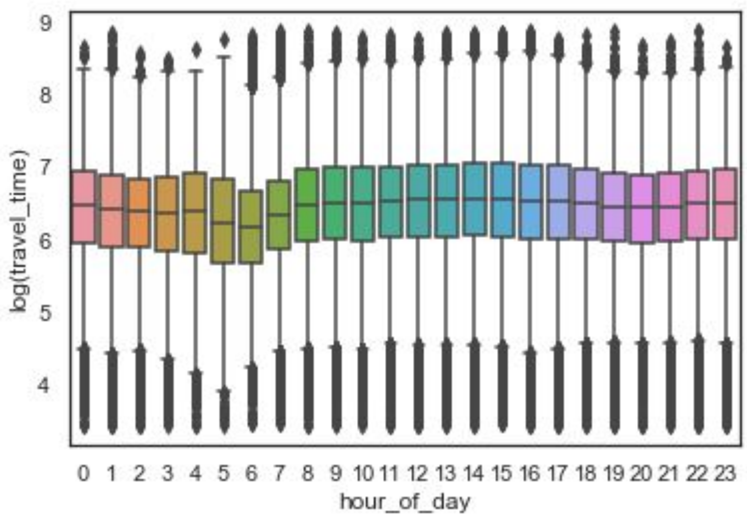
- Day_of_week: Mon to Fri seem to have longer travel time
- From day_of_week: weekday (1 or 0)

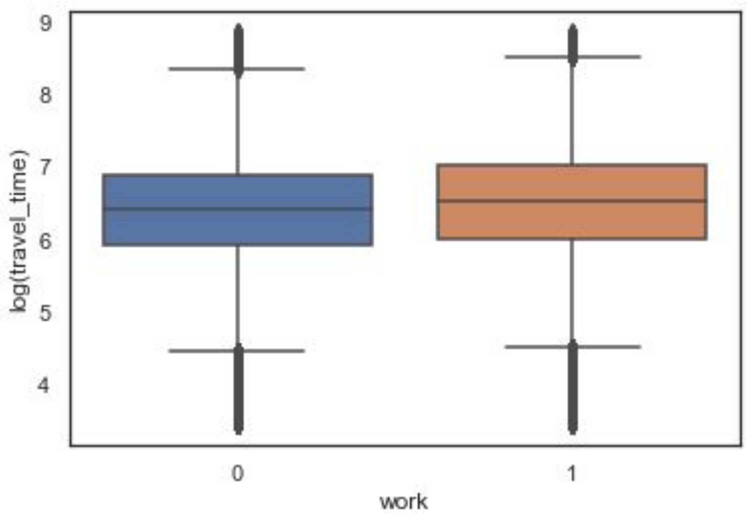




Hour of day and work

- Hour_of_day: 8 am - 6 pm seem to have longer travel time
- From hour_of_day: work (1 or 0)





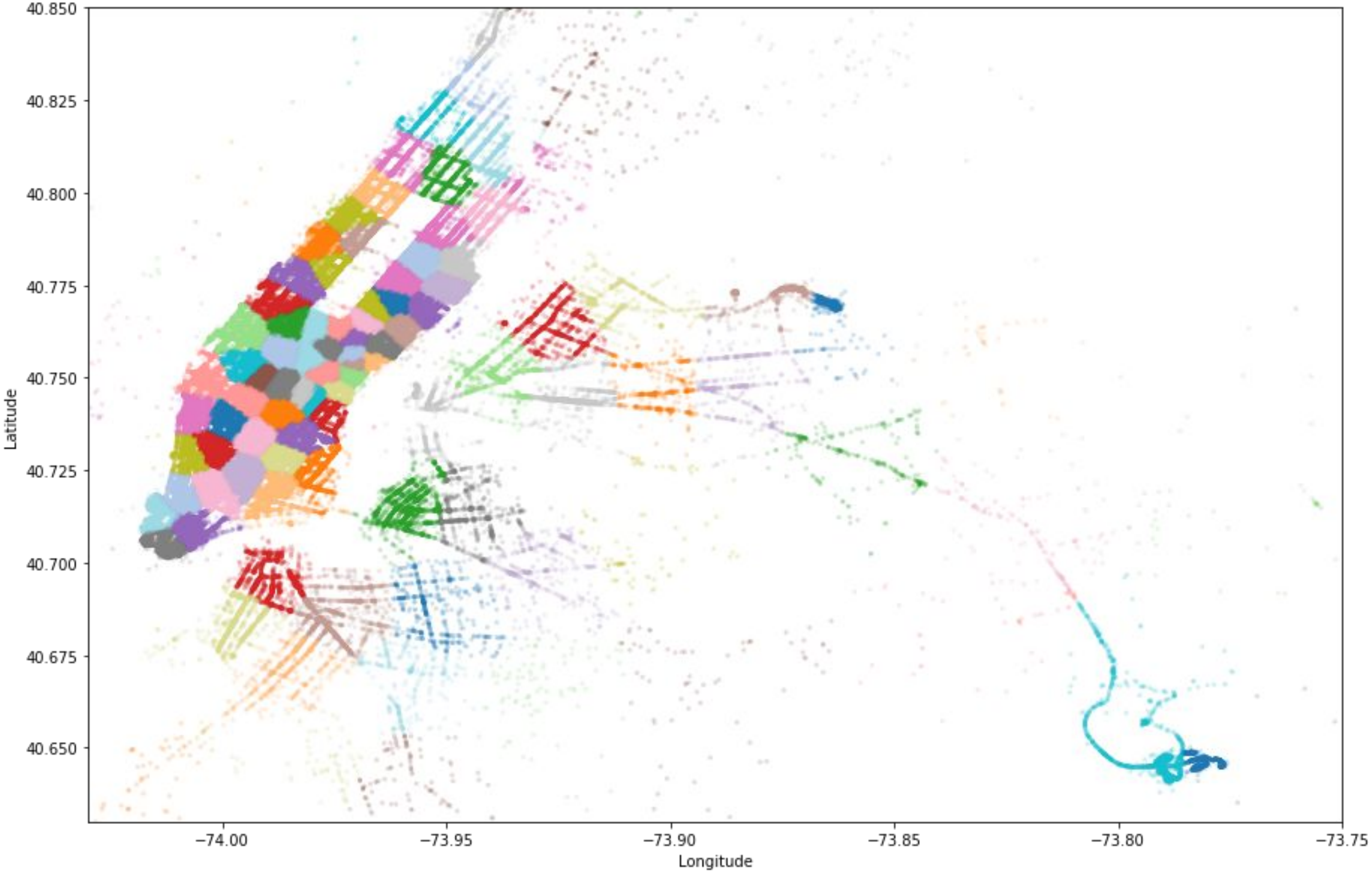
Clustering

- Center_latitude, center_longitude: middle point between pickup and dropoff. Use for later

Clustering all lat/lon pairs:

- K-Means: into 100 small neighborhoods using euclidean distance
- Create pickup_cluster, center_cluster and dropoff_cluster for each sample
- Can feed into linear models after one-hot-encoding

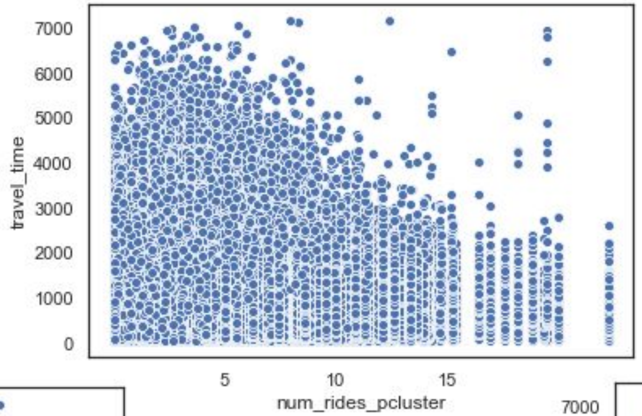
Clustering visualization



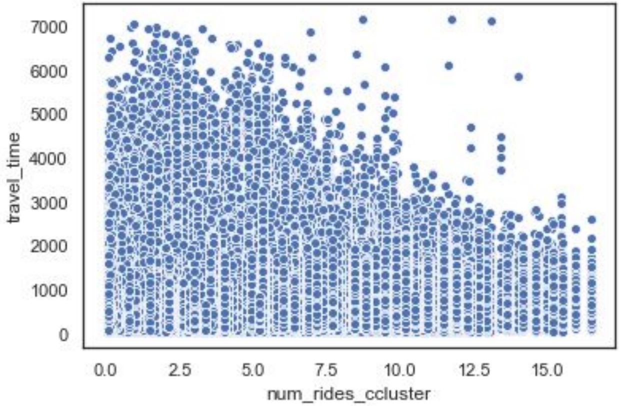
Traffic

- For each sample, average number of rides in that hour of that week day in its pickup cluster, center cluster and dropoff cluster separately.
- Can show number of yellow cabs in beginning, middle and end of the trip
- Potentially can indicate traffic?

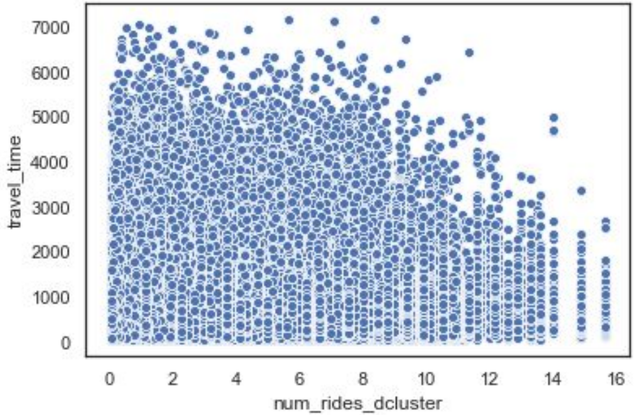
Traffic



7000
6000
5000
4000
3000
2000
1000
0
travel_time
5
10
15
num_rides_pcluster



7000
6000
5000
4000
3000
2000
1000
0
travel_time
0.0
2.5
5.0
7.5
10.0
12.5
15.0
num_rides_ccluster



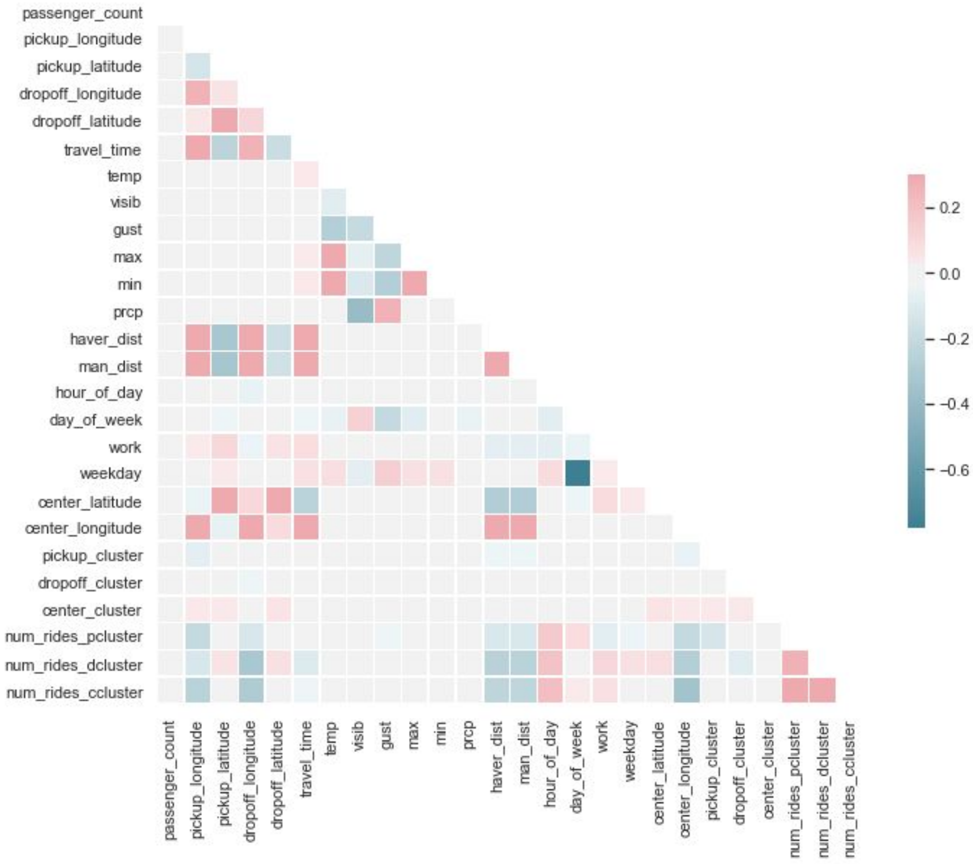
7000
6000
5000
4000
3000
2000
1000
0
travel_time
0
2
4
6
8
10
12
14
16
num_rides_dcluster

Traffic

Why the downward trend?

- Maybe when there are a lot of taxis picking up and dropping off riders in an area at a certain time, it means the traffic is actually moving instead of congested. Therefore the shorter travel time?

Correlations



Model Training
&
Model Selection

Preprocessing

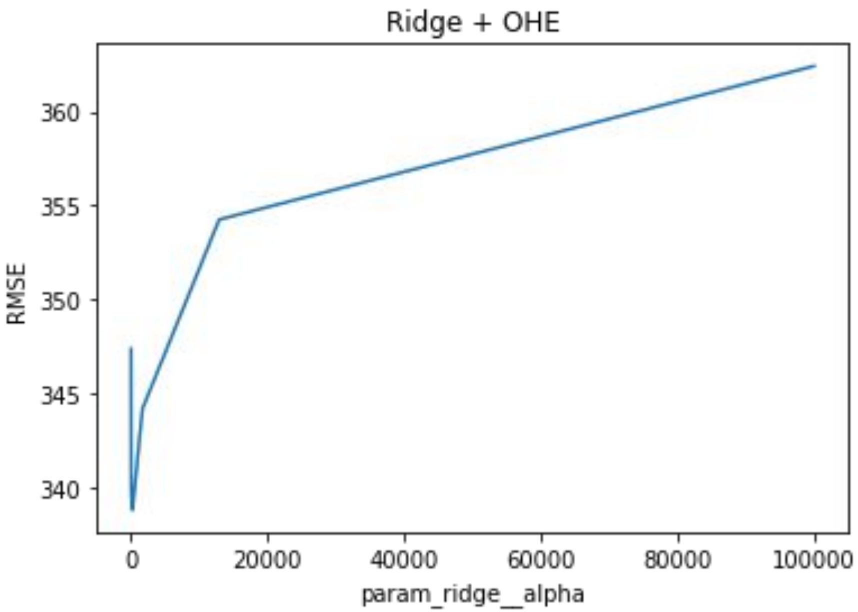
- **OneHotEncoder:** categorical features
- **TargetEncoder:** categorical features with > 15 categories , e.g. pickup_cluster. (For each cluster, what is the average travel_time?) Instead of 100 columns using OHE, it’s just one column
- **PowerTransformer**: make distributions of numerical features closer to normal
- **StandardScaler**: scale numerical features
- **ColumnTransformer:** do different preprocessing on different column types
- **Interaction features:** add interaction terms
- **Pipeline**: prevent test info leaking
- **GridSearchCV:** training and tune/select hyperparameters

Results

Model	**Test RMSE**	**Test R2**
Ridge (OHE)	458	0.56
Ridge (Target + OHE + Interaction)	446	0.58
Ridge (OHE + Target + Power + Scaling)	487	0.50
Lasso (OHE + Target + Power + Scaling)	493	0.49
XGBoost (OHE)	393	0.67
XGBoost (OHE + Target)	374	0.70
RF (OHE + Target)	380	0.69

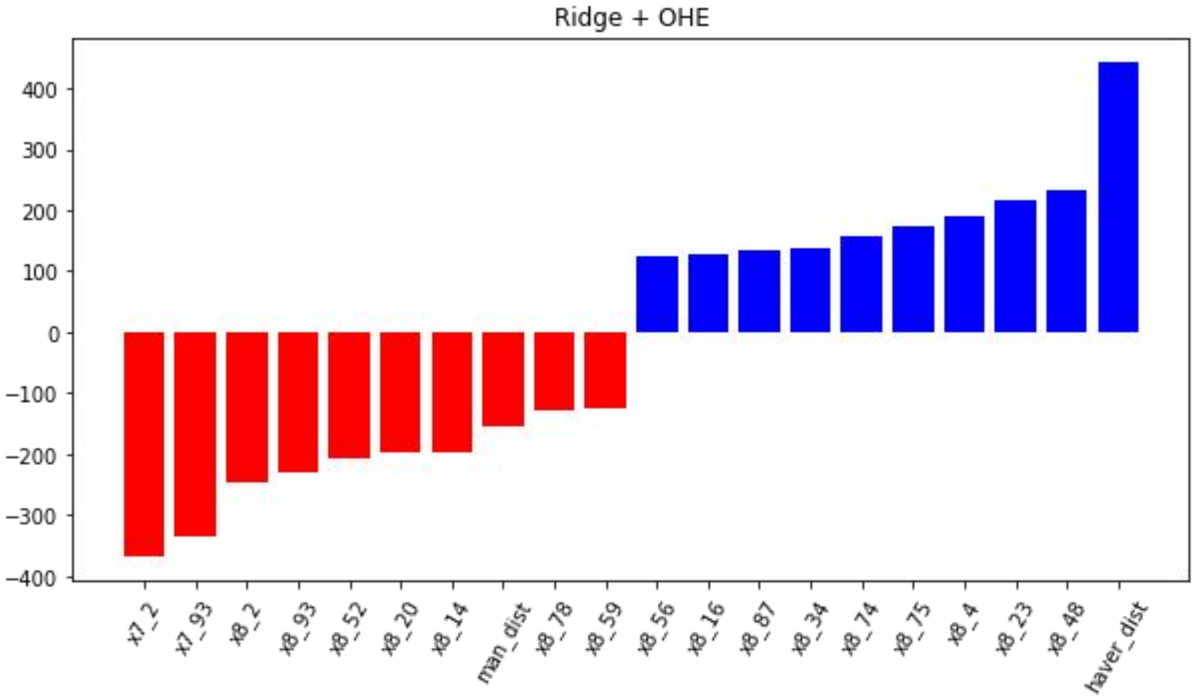
Model 1: Ridge + OHE

- Test RMSE = 458
- Test R2 = 0.56
- Best alpha = 215.44



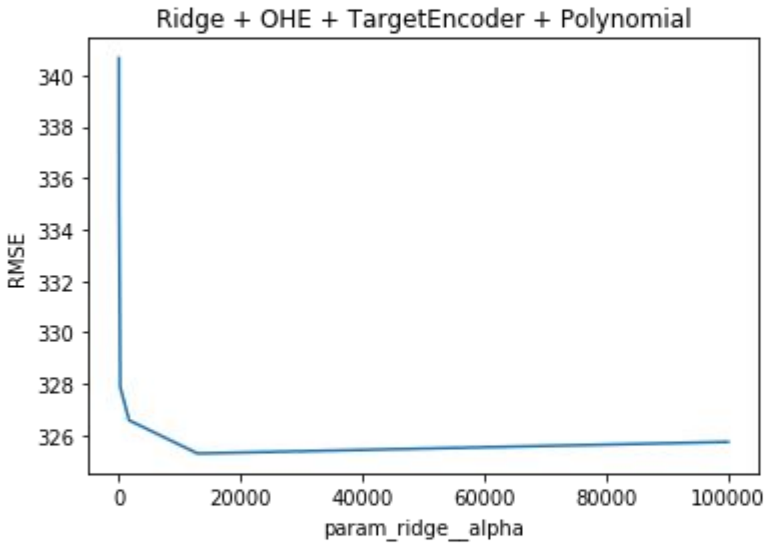
Model 1: Ridge + OHE

- Haversine
- Manhattan
- Pickup cluster
- Dropoff cluster



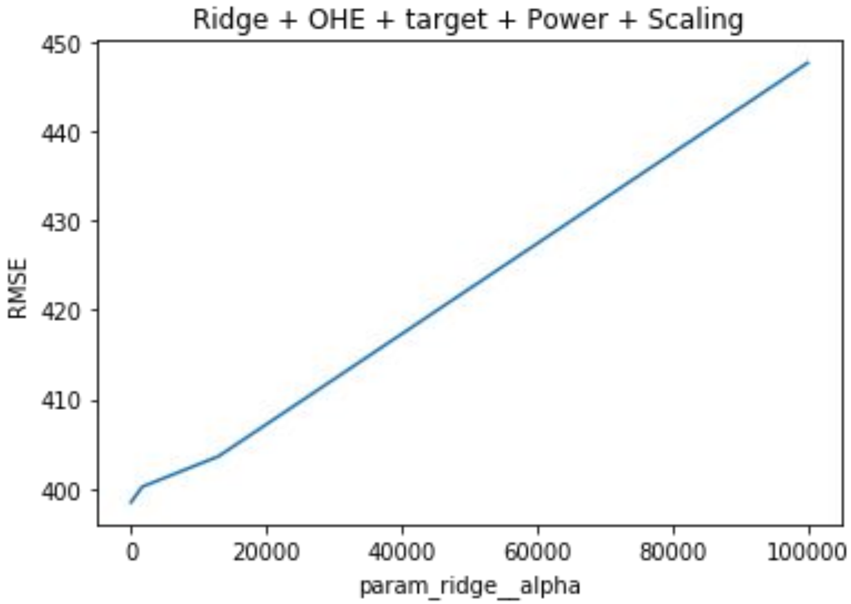
Model 2:
Ridge (OHE, TargetEncode, Interaction)

- Test RMSE = 446
- Test R2 = 0.58
- Best alpha = 10000
- **Best among linear models**
- **Good for interpretability**



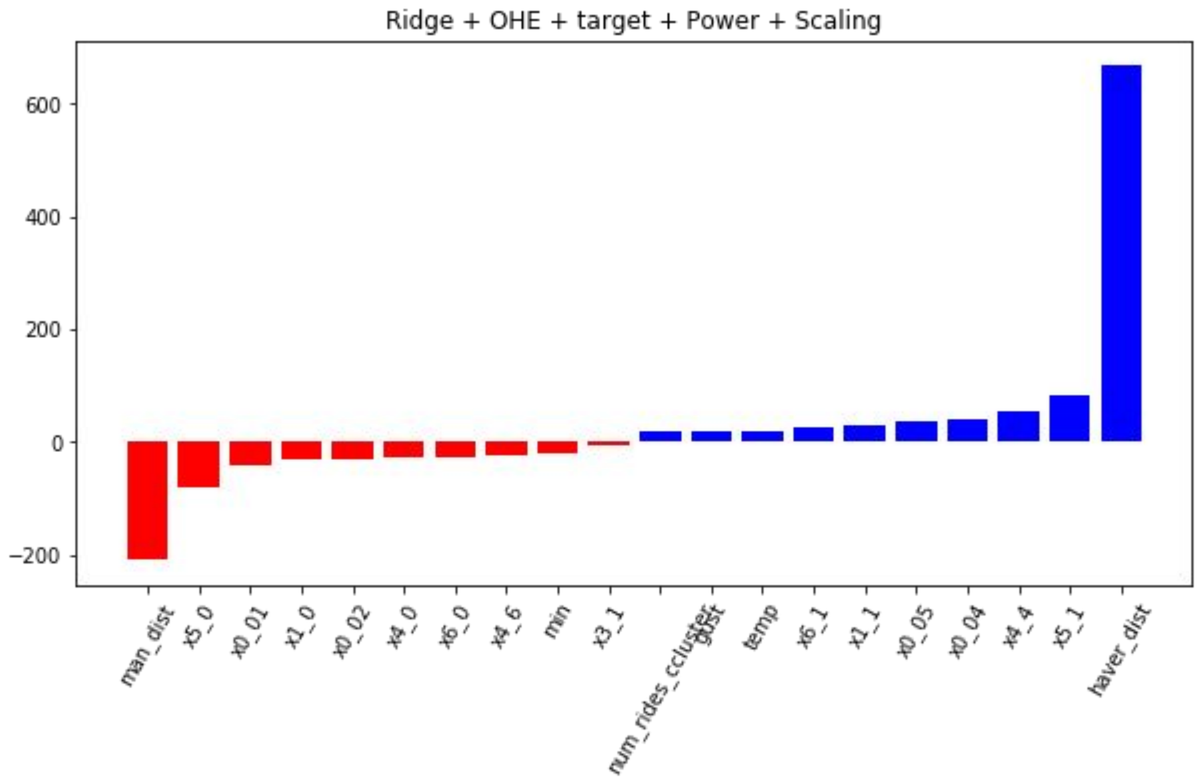
Model 3: Ridge (OHE, TargetEncode, PowerTransform, Scaling)

- Test RMSE = 487
- Test R2 = 0.50
- Best alpha = 27.83



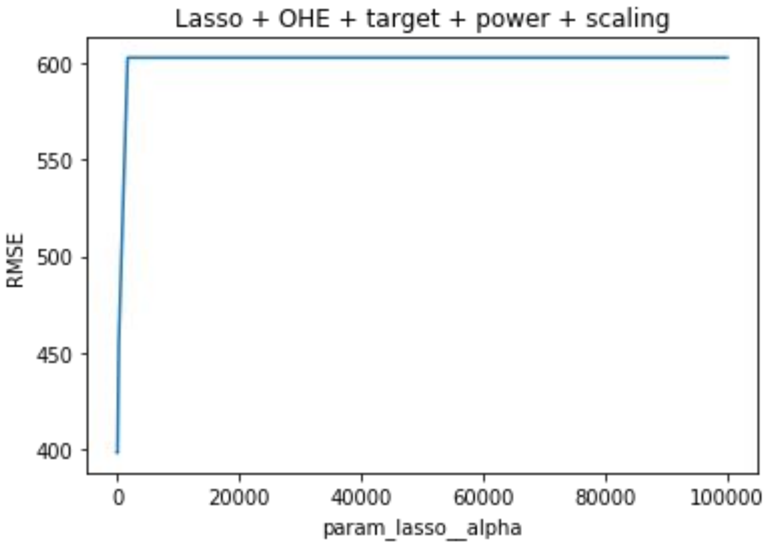
Model 3: Ridge (OHE, TargetEncode, PowerTransform, Scaling)

- Haversine
- Manhattan
- Work
- Month
- Fog
- Day of week



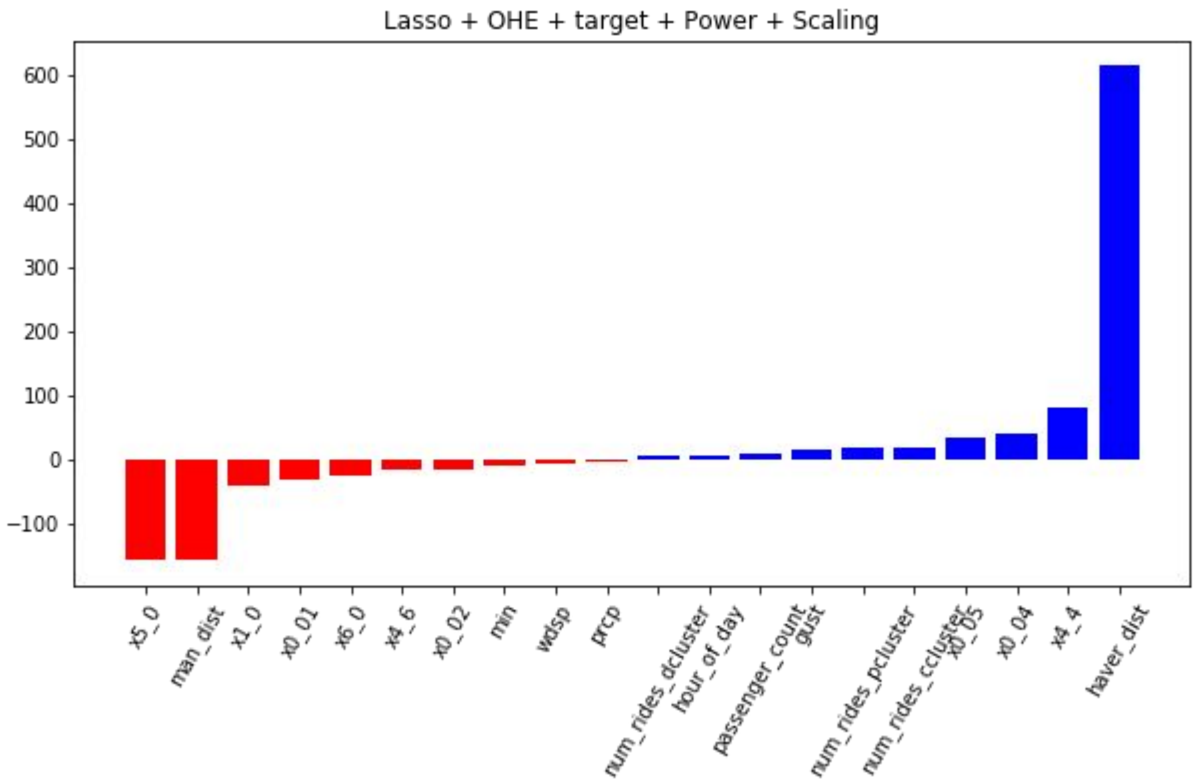
Model 4: Lasso (OHE, TargetEncode, PowerTransform, Scaling)

- Test RMSE = 493
- Test R2 = 0.49
- Best alpha = 0.46



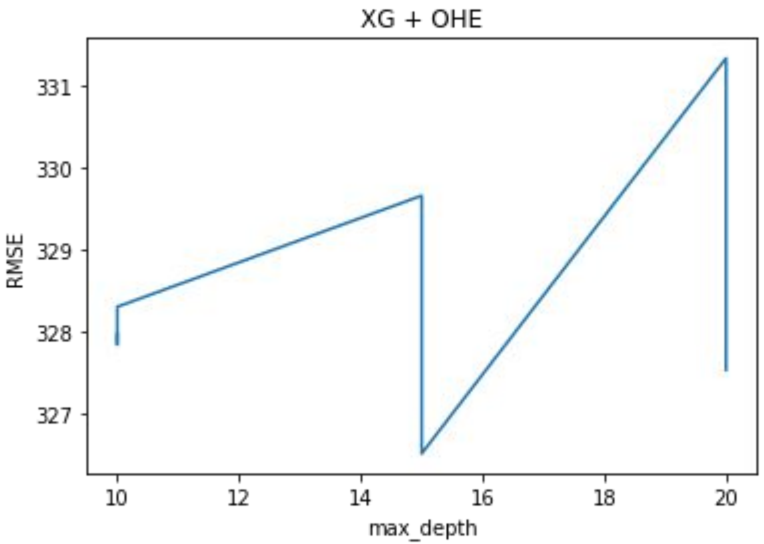
Model 4: Lasso (OHE, TargetEncode, PowerTransform, Scaling)

- Haversine
- Manhattan
- Month
- Fog
- Day of week
- Work
- Number of rides in center cluster



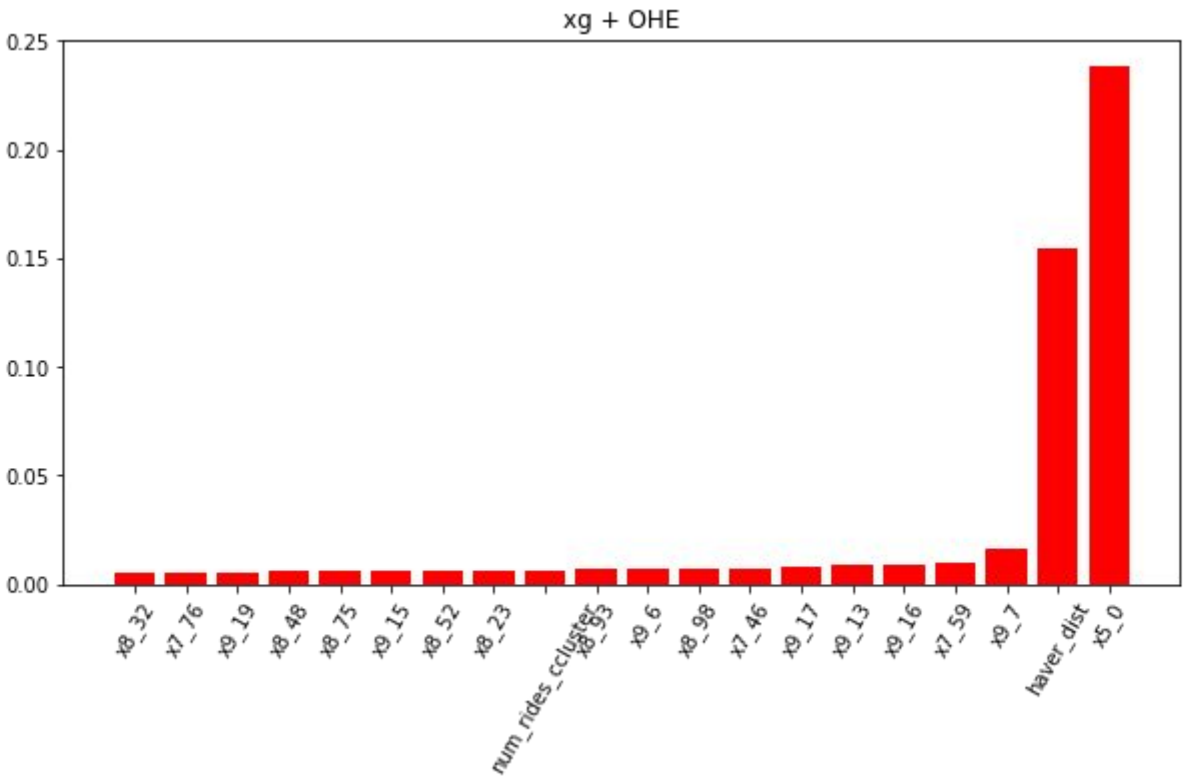
Model 5: XGBoost (OHE)

- Test RMSE = 393
- Test R2 = 0.67
- Best max_depth = 15



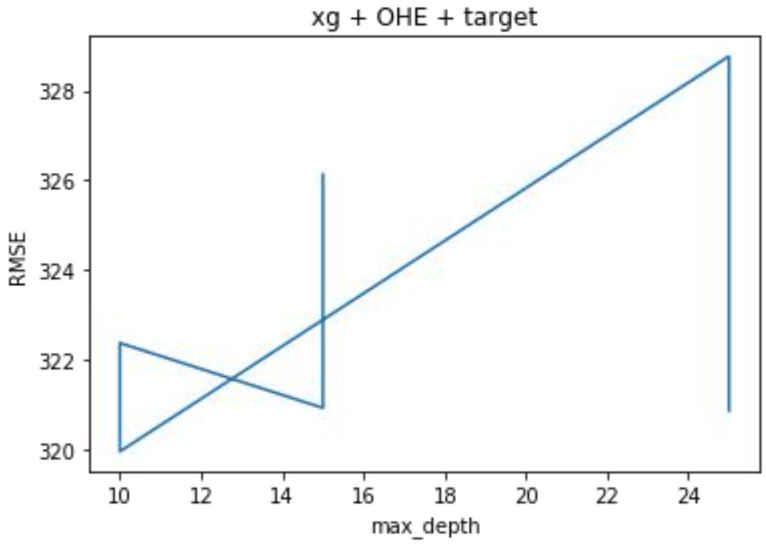
Model 5: XGBoost (OHE)

- Work
- Haversine
- Pickup cluster
- Dropoff cluster
- Hour of day



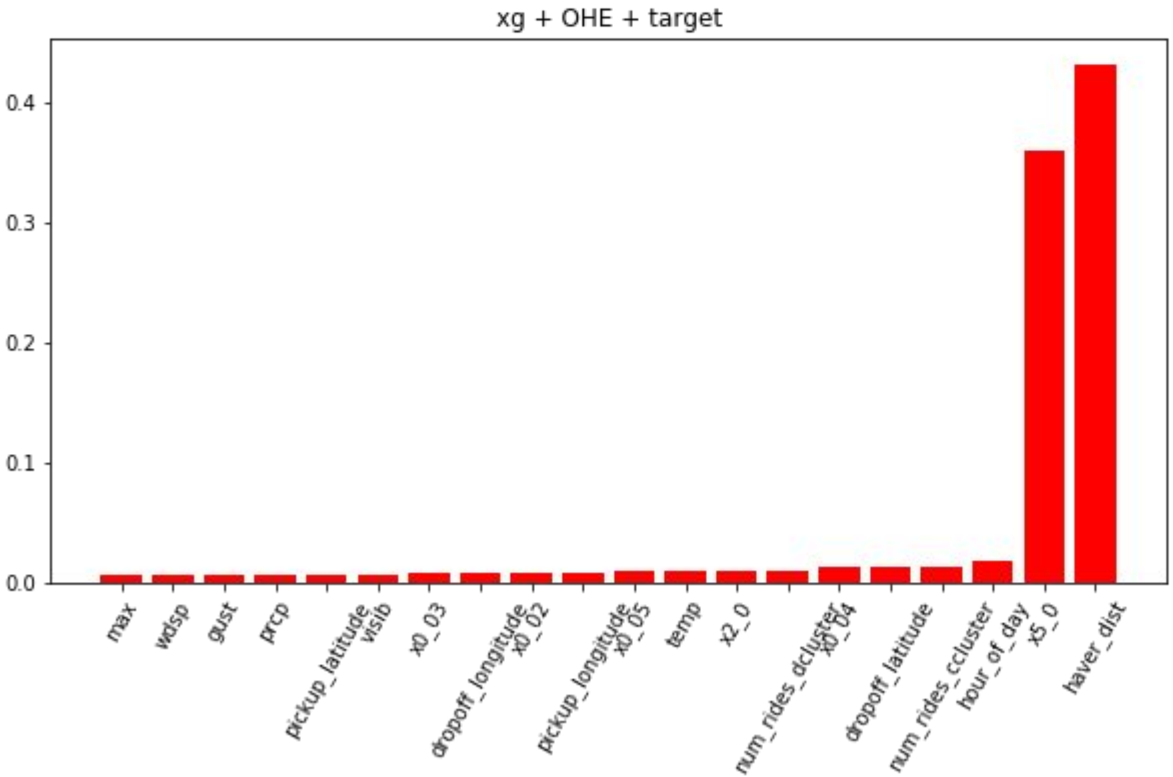
Model 6: XGBoost (OHE + TargetEncode)

- **Best for prediction**
- Test RMSE = 374
- Test R2 = 0.70
- Best max_depth = 10



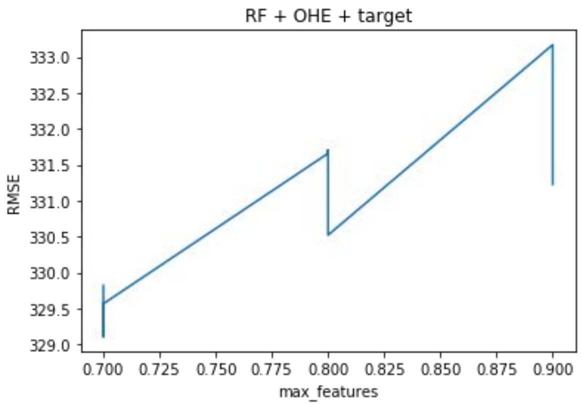
Model 6: XGBoost (OHE + TargetEncode)

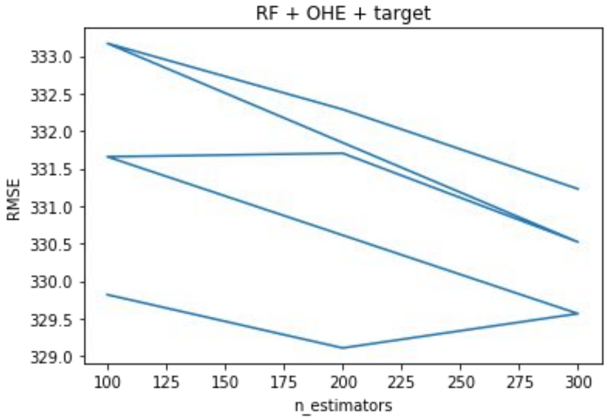
- Haversine
- Work
- Hour of day
- Number of rides in center cluster
- Dropoff latitude



Model 7:
Random Forest (OHE + TargetEncode)

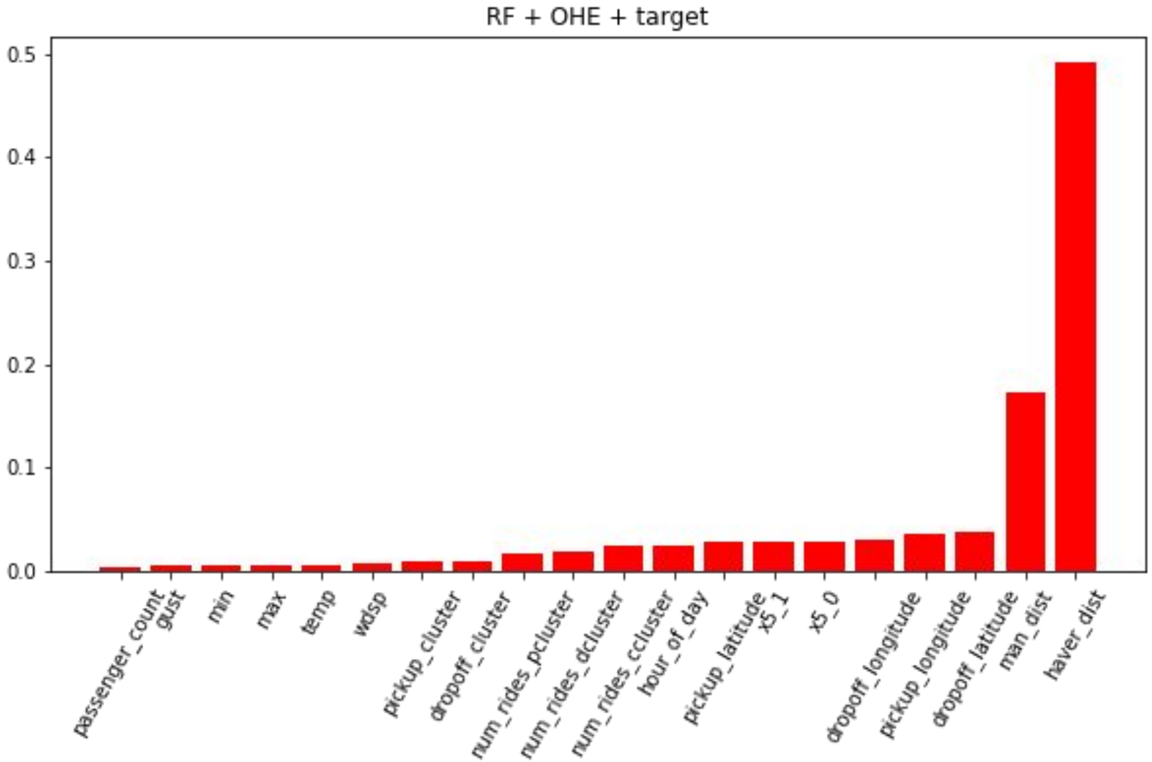
- Test RMSE = 380
- Test R2 = 0.69
- Best max_features = 0.7
- Best n_estimators = 200





Model 7:
Random Forest (OHE + TargetEncode)

- Haversine
- Manhattan
- Dropoff latitude
- Pickup latitude
- Pickup longitude
- Dropoff longitude



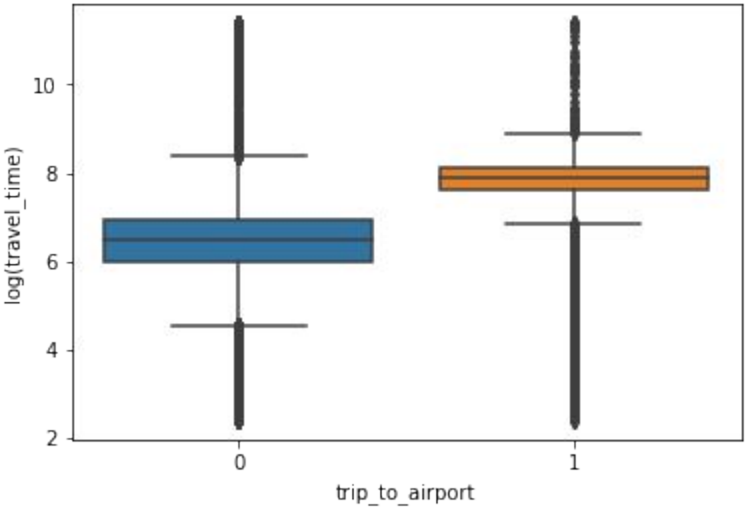
Conclusion

- Most important features: Haversine dist, Manhattan dist, work, dropoff cluster, pickup cluster
- Ensemble tree-based models > linear models
- Linear models and XGBoost are fast to train

Important lessons

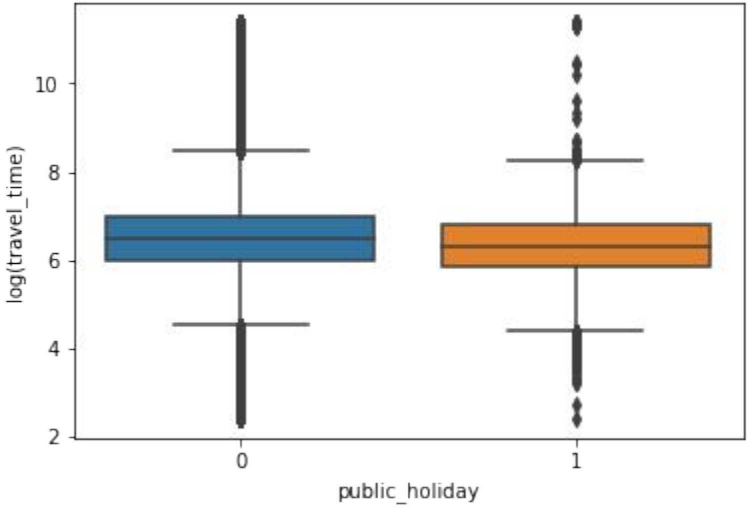
- Removing outliers: very important
- Even tree-based model that are supposedly robust to outliers were performing badly
- Always start with a simple model with simple preprocessing and variable transformations
- Then move on gradually to more complex models with more sophisticated preprocessing
- This way can have an idea of a baseline idea on how complex your model needs to be

Next Steps: Trip to Airport



Trip with rate_code == 2 or 3, to either JFK or Newark

Next Steps: NY Public Holiday



→ only from January to June

Federal & NY Public Holiday	Number of Trips in Dataset
New Year	5,023
Martin Luther King Jr. Day	5,087
President's Day	5,119
Memorial Day	3,817
Independence Day	0
Labor day	0
Columbus Day	0
Veterans Day	0
Thanksgiving Day	0
Christmas	0
All Public Holidays	**19,046**