In [0]:
```
%matplotlib inline
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from datetime import timedelta
import datetime as dt
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [16, 10]
import seaborn as sns
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
from sklearn.cluster import MiniBatchKMeans
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from scipy import stats
from scipy.stats import norm, skew #for some statistics
from sklearn.cluster import KMeans
from sklearn.cluster import KMeans
from sklearn.ensemble import RandomForestRegressor,  GradientBoostingRe
gressor
from sklearn.kernel_ridge import KernelRidge
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import KFold, cross_val_score, train_test_
split
from sklearn.metrics import mean_squared_error
import random as rnd
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import train_test_split
import seaborn as sns  #advanced visualization library
import requests, zipfile, io
import warnings
from  datetime import datetime
warnings.filterwarnings('ignore')
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
%config InlineBackend.figure_format = 'retina' #set 'png' here when wor
king on notebook
%matplotlib inline
```

In [0]:
```
train = pd.read_csv('drive/My Drive/TAXI/train.csv')
```

In [0]:
```
train.head(5)
```

Out[0]:

	id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	pickup_longitude
0	id2875421	2	2016-03-14 17:24:55	2016-03-14 17:32:30	1	-73.982155
1	id2377394	1	2016-06-12 00:43:35	2016-06-12 00:54:38	1	-73.980415
2	id3858529	2	2016-01-19 11:35:24	2016-01-19 12:10:48	1	-73.979027
3	id3504673	2	2016-04-06 19:32:31	2016-04-06 19:39:40	1	-74.010040
4	id2181028	2	2016-03-26 13:30:55	2016-03-26 13:38:10	1	-73.973053

In [0]:
```
train = train[['pickup_datetime','dropoff_datetime','passenger_count',
'pickup_longitude','pickup_latitude','dropoff_longitude','dropoff_latit
ude','trip_duration']]
```

In [0]:
```
# outliers
train =train[ train['trip_duration']<5000]
```

In [0]:
```
# outliers
train=train[(train.passenger_count>=0) &(train.passenger_count<=8)]
train= train[(train.pickup_longitude>=-74.03) & (train.pickup_longitude
<=-73.77)]
train= train[(train.pickup_latitude>=40.63) & (train.pickup_latitude<=4
0.85)]
train= train[(train.dropoff_longitude>=-74.03) & (train.dropoff_longitu
de<=-73.77)]
train= train[(train.dropoff_latitude>=40.63) & (train.dropoff_latitude<
=40.85)]
train = train.dropna(how = 'any', axis = 'rows')
```

Feature Extraction

from Date

In [0]:
```
train['pickup_datetime']=pd.to_datetime(train['pickup_datetime'],format
="%Y-%m-%d %H:%M:%S")
train.loc[:, 'pickup_weekday'] = train['pickup_datetime'].dt.weekday
train.loc[:, 'pickup_hour_weekofyear'] = train['pickup_datetime'].dt.we
ekofyear
train.loc[:, 'pickup_hour'] = train['pickup_datetime'].dt.hour
train.loc[:, 'pickup_minute'] = train['pickup_datetime'].dt.minute
train.loc[:, 'pickup_dt'] = (train['pickup_datetime'] - train['pickup_d
atetime'].min()).dt.total_seconds()
train.loc[:, 'pickup_week_hour'] = train['pickup_weekday'] * 24 + train
['pickup_hour']
```

Distance feature

- We use PCA to transform longitude and latitude coordinates. In this case it is not about dimension reduction since we transform 2D-> 2D. The rotation could help for decision tree splits.

In [0]:
```
coords = np.vstack((train[['pickup_latitude', 'pickup_longitude']].valu
es,
                    train[['dropoff_latitude', 'dropoff_longitude']].va
lues,
                    ))

pca = PCA().fit(coords)
train['pickup_pca0'] = pca.transform(train[['pickup_latitude', 'pickup_
longitude']])[:, 0]
train['pickup_pca1'] = pca.transform(train[['pickup_latitude', 'pickup_
longitude']])[:, 1]
train['dropoff_pca0'] = pca.transform(train[['dropoff_latitude', 'dropo
ff_longitude']])[:, 0]
train['dropoff_pca1'] = pca.transform(train[['dropoff_latitude', 'dropo
ff_longitude']])[:, 1]
```

- Distance

Let's calculate the distance (km) between pickup and dropoff points. Currently Haversine is used, geopy has another heuristics (vincenty() or great_circle()) if you prefer. The cabs are not flying and we are in New York so we could check the Manhattan (L1) distance too :)

pd.DataFrame.apply() would be too slow so the haversine function is rewritten to handle arrays. We extraxt the middle of the path as a feature as well.

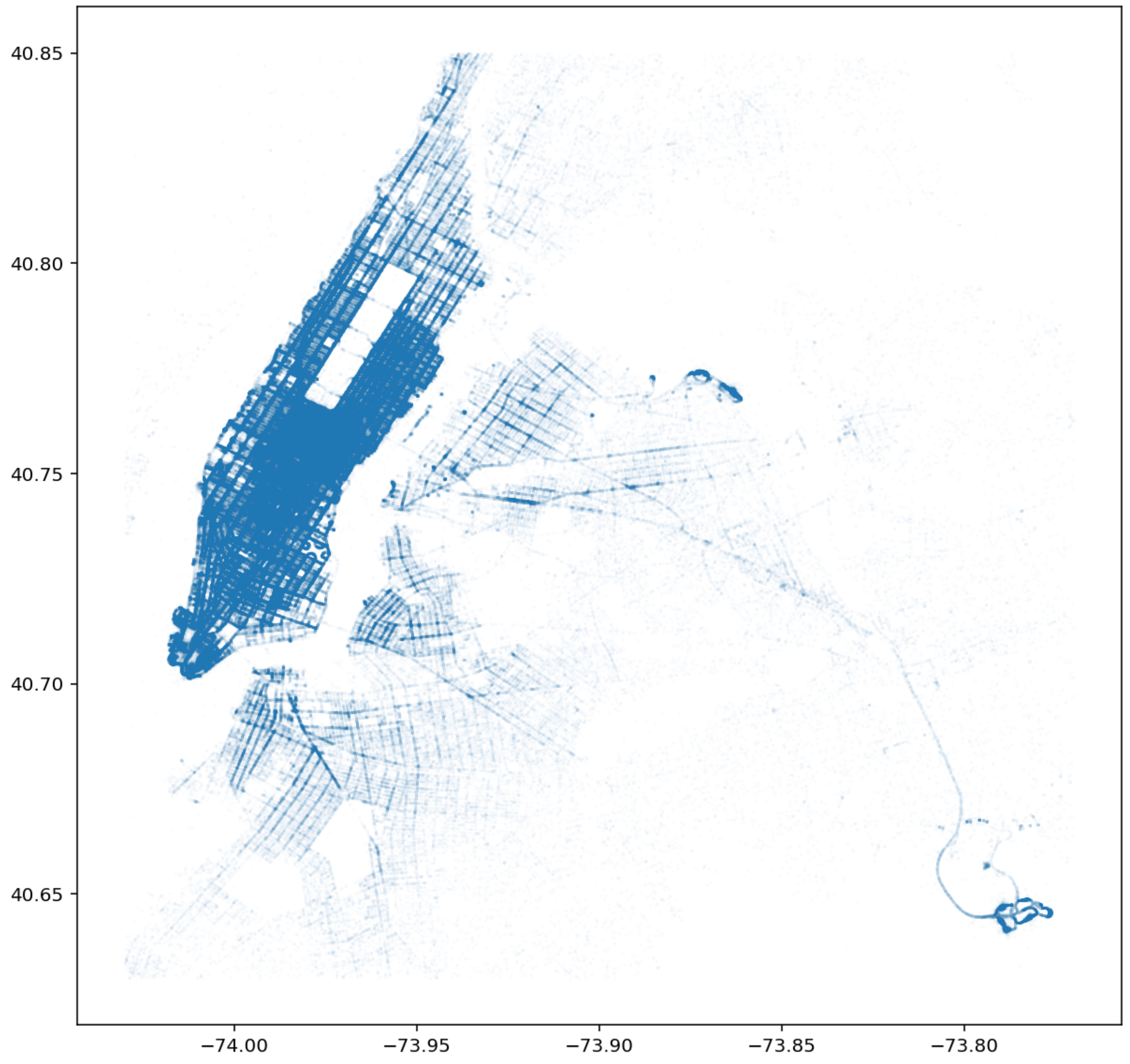
In [0]:
```
def haversine_array(lat1, lng1, lat2, lng2):
    lat1, lng1, lat2, lng2 = map(np.radians, (lat1, lng1, lat2, lng2))
    AVG_EARTH_RADIUS = 6371  # in km
    lat = lat2 - lat1
    lng = lng2 - lng1
    d = np.sin(lat * 0.5) ** 2 + np.cos(lat1) * np.cos(lat2) * np.sin(l
ng * 0.5) ** 2
    h = 2 * AVG_EARTH_RADIUS * np.arcsin(np.sqrt(d))
    return h

def dummy_manhattan_distance(lat1, lng1, lat2, lng2):
    a = haversine_array(lat1, lng1, lat1, lng2)
    b = haversine_array(lat1, lng1, lat2, lng1)
    return a + b

def bearing_array(lat1, lng1, lat2, lng2):
    AVG_EARTH_RADIUS = 6371  # in km
    lng_delta_rad = np.radians(lng2 - lng1)
    lat1, lng1, lat2, lng2 = map(np.radians, (lat1, lng1, lat2, lng2))
    y = np.sin(lng_delta_rad) * np.cos(lat2)
    x = np.cos(lat1) * np.sin(lat2) - np.sin(lat1) * np.cos(lat2) * np.
cos(lng_delta_rad)
    return np.degrees(np.arctan2(y, x))

train.loc[:, 'distance_haversine'] = haversine_array(train['pickup_lati
tude'].values, train['pickup_longitude'].values, train['dropoff_latitud
e'].values, train['dropoff_longitude'].values)
train.loc[:, 'distance_dummy_manhattan'] = dummy_manhattan_distance(tra
in['pickup_latitude'].values, train['pickup_longitude'].values, train[
'dropoff_latitude'].values, train['dropoff_longitude'].values)
train.loc[:, 'direction'] = bearing_array(train['pickup_latitude'].valu
es, train['pickup_longitude'].values, train['dropoff_latitude'].values,
 train['dropoff_longitude'].values)
train.loc[:, 'pca_manhattan'] = np.abs(train['dropoff_pca1'] - train['p
ickup_pca1']) + np.abs(train['dropoff_pca0'] - train['pickup_pca0'])
train.loc[:, 'center_latitude'] = (train['pickup_latitude'].values + tr
ain['dropoff_latitude'].values) / 2
train.loc[:, 'center_longitude'] = (train['pickup_longitude'].values +
train['dropoff_longitude'].values) / 2
```

In [0]:
```
train["diff_lat"]=abs(train.pickup_latitude-train.dropoff_latitude)
train["diff_long"]=abs(train.pickup_longitude-train.dropoff_longitude)
```

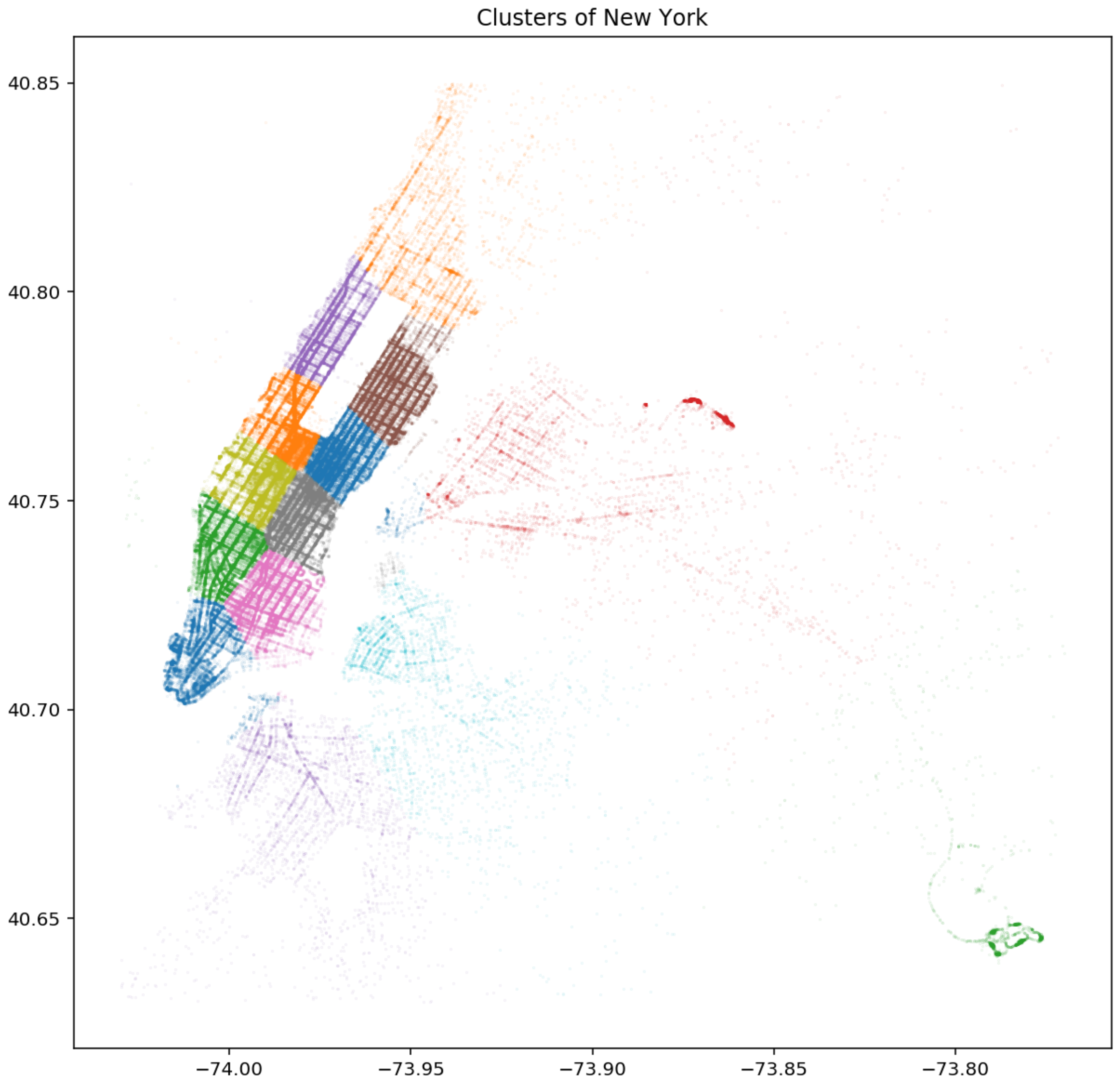
clustring the city

Let's cluster New York City based on the pick-up and drop-off points of each taxi ride

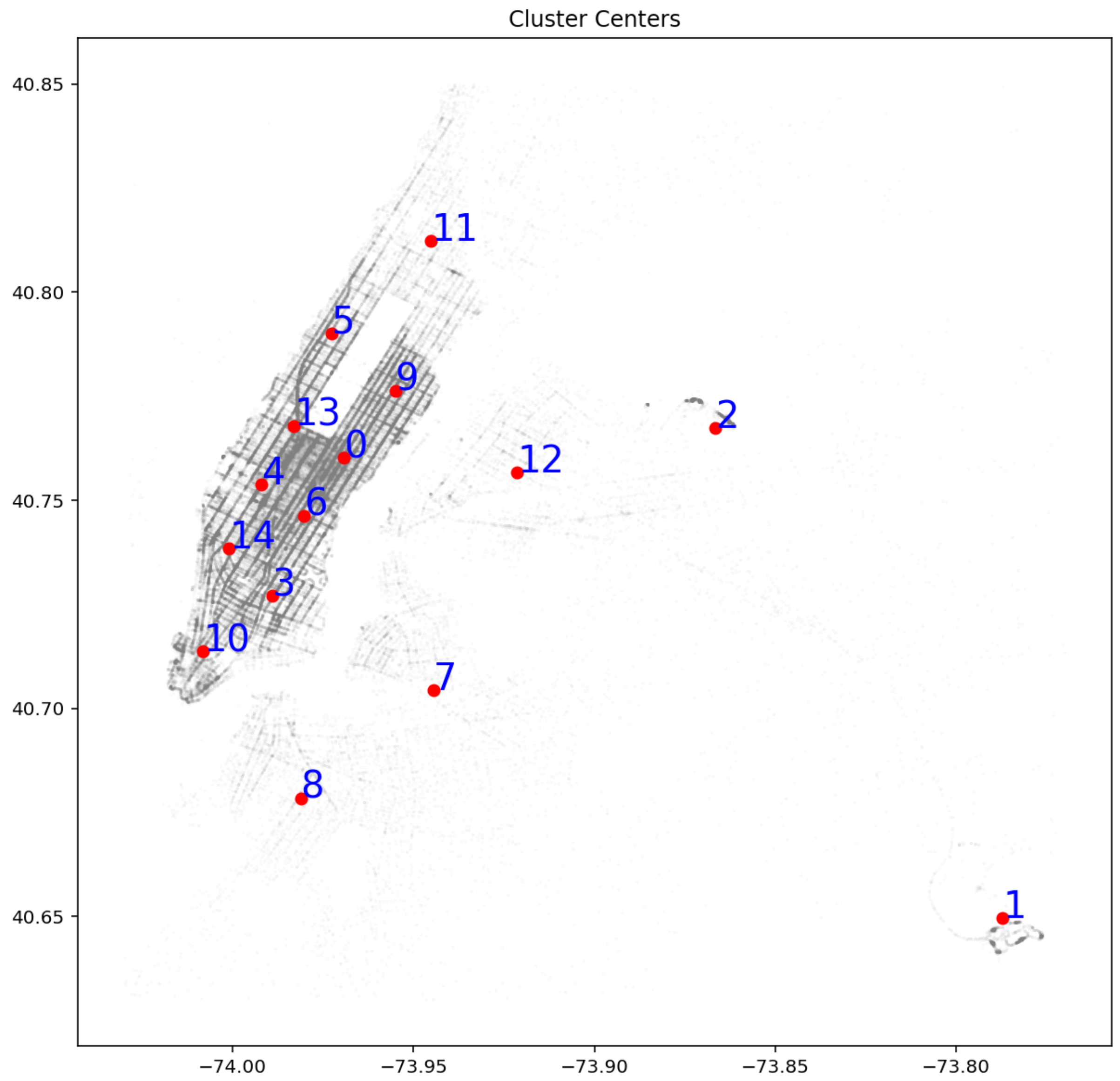
In [0]:
```
longitude = list(train.pickup_longitude) + list(train.dropoff_longitude
)
latitude = list(train.pickup_latitude) + list(train.dropoff_latitude)
plt.figure(figsize = (10,10))
plt.plot(longitude,latitude,'.', alpha = 0.4, markersize = 0.05)
plt.show()
```



```
In [0]: longitude = list(train.pickup_longitude) + list(train.dropoff_longitude
        )
        latitude = list(train.pickup_latitude) + list(train.dropoff_latitude)
        loc_df = pd.DataFrame()
        loc_df['longitude'] = longitude
        loc_df['latitude'] = latitude
        kmeans = KMeans(n_clusters=15, random_state=2, n_init = 10).fit(loc_df)
        loc_df['label'] = kmeans.labels_

        loc_df = loc_df.sample(200000)
        plt.figure(figsize = (10,10))
        for label in loc_df.label.unique():
            plt.plot(loc_df.longitude[loc_df.label == label],loc_df.latitude[lo
        c_df.label == label],'.', alpha = 0.3, markersize = 0.3)

        plt.title('Clusters of New York')
        plt.show()
```



```
In [0]: fig,ax = plt.subplots(figsize = (10,10))
        for label in loc_df.label.unique():
            ax.plot(loc_df.longitude[loc_df.label == label],loc_df.latitude[loc
        _df.label == label],'.', alpha = 0.4, markersize = 0.1, color = 'gray')
            ax.plot(kmeans.cluster_centers_[label,0],kmeans.cluster_centers_[la
        bel,1],'o', color = 'r')
            ax.annotate(label, (kmeans.cluster_centers_[label,0],kmeans.cluster
        _centers_[label,1]), color = 'b', fontsize = 20)
        ax.set_title('Cluster Centers')
        plt.show()
```

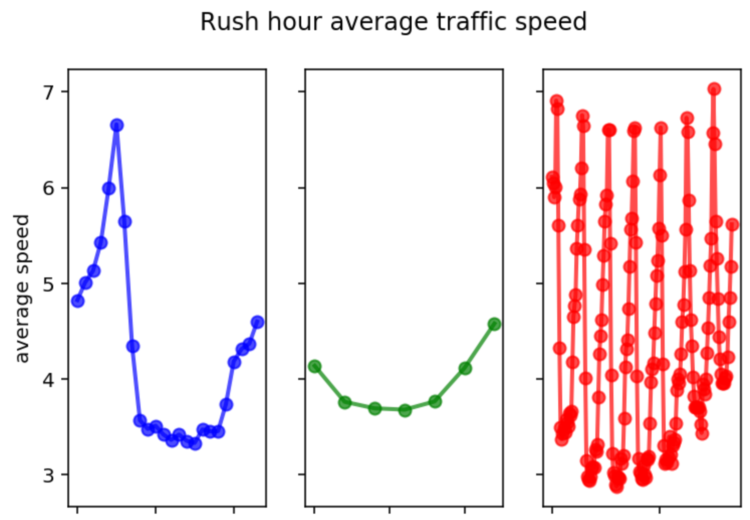


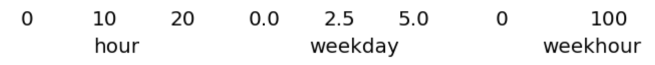
In [0]:
```
train['pickup_cluster'] = kmeans.predict(train[['pickup_longitude','pic
kup_latitude']])
```

```
train['dropoff_cluster'] = kmeans.predict(train[['dropoff_longitude','d
ropoff_latitude']])
```

Speed

In [0]:
```
train.loc[:, 'avg_speed_h'] = 1000 * train['distance_haversine'] / trai
n['trip_duration']
train.loc[:, 'avg_speed_m'] = 1000 * train['distance_dummy_manhattan']
/ train['trip_duration']
fig, ax = plt.subplots(ncols=3, sharey=True)
ax[0].plot(train.groupby('pickup_hour').mean()['avg_speed_h'], 'bo-', l
w=2, alpha=0.7)
ax[1].plot(train.groupby('pickup_weekday').mean()['avg_speed_h'], 'go-'
, lw=2, alpha=0.7)
ax[2].plot(train.groupby('pickup_week_hour').mean()['avg_speed_h'], 'ro
-', lw=2, alpha=0.7)
ax[0].set_xlabel('hour')
ax[1].set_xlabel('weekday')
ax[2].set_xlabel('weekhour')
ax[0].set_ylabel('average speed')
fig.suptitle('Rush hour average traffic speed')
plt.show()
```





Mean speed per cluster an time

In [0]:
```
new = train.groupby(['pickup_cluster','dropoff_cluster'])['avg_speed_h'
].agg({'mean_clus':'mean'})
train = pd.merge(train,new, how = 'left',on=['pickup_cluster','dropoff_
cluster'])
```

In [0]:
```
new = train.groupby('pickup_hour')['avg_speed_h'].agg({'mean_hour':'mea
n'})
train = pd.merge(train,new, how = 'left',on='pickup_hour')
```

In [0]:
```
new = train.groupby('pickup_weekday')['avg_speed_h'].agg({'mean_weekda
y':'mean'})
train = pd.merge(train,new, how = 'left',on='pickup_weekday')
```

In [0]:
```
new = train.groupby('pickup_week_hour')['avg_speed_h'].agg({'mean_weekd
ay_hour':'mean'})
train = pd.merge(train,new, how = 'left',on='pickup_week_hour')
```

Modeling

In [0]:
```
X_all = train.drop(['pickup_datetime','dropoff_datetime','avg_speed_h',
'avg_speed_m','trip_duration'],1)
Y = train["trip_duration"]
```

In [0]:
```
from sklearn.model_selection import KFold, cross_val_score, train_test_
split
from sklearn.preprocessing import RobustScaler, StandardScaler, MinMaxS
caler
```

```
X_train, val_X, Y_train, val_y = train_test_split(X_all, Y,random_state
=1)
```

In [0]:
```
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import ElasticNet, Lasso,  BayesianRidge, Las
soLarsIC
from sklearn.ensemble import RandomForestRegressor,  GradientBoostingRe
gressor
from sklearn.kernel_ridge import KernelRidge
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import RobustScaler
from sklearn.base import BaseEstimator, TransformerMixin, RegressorMixi
n, clone
from sklearn.model_selection import KFold, cross_val_score, train_test_
split
from sklearn.metrics import mean_squared_error
import xgboost as xgb
import lightgbm as lgb
```

In [0]:
```
lightgbm = lgb.LGBMRegressor(objective='regression',
                                       num_leaves=12,
                                       learning_rate=0.2,
                                       n_estimators=10000,
                                       max_bin=200,
                                       bagging_fraction=0.85,
                                       bagging_freq=5,
                                       bagging_seed=7,
                                       feature_fraction=0.2,
                                       feature_fraction_seed=7,
                                       verbose=-1,
                                       min_data_in_leaf=2,
                                       min_sum_hessian_in_leaf=11,
                                       tree_method='gpu_hist' ,
                                       boosting_type = 'gbdt',
                                       metric = 'rmse'
                                       )
```

In [0]:
```
eval_set = [(X_train, Y_train), (val_X, val_y)]
```

```
lightgbm.fit(X_train,Y_train,eval_set=eval_set,
        verbose=50, early_stopping_rounds=100)
preds = lightgbm.predict(val_X)
val_mae2 =np.sqrt(mean_squared_error( val_y,preds))
print('valid_error',val_mae2)
preds = lightgbm.predict(X_train)
val_mae2 =np.sqrt(mean_squared_error( Y_train, preds))
print('train_error',val_mae2)
```

```
Training until validation scores don't improve for 100 rounds.
[50]	training's rmse: 303.421	valid_1's rmse: 303.934
[100]	training's rmse: 289.467	valid_1's rmse: 290.384
[150]	training's rmse: 283.219	valid_1's rmse: 284.471
[200]	training's rmse: 279.498	valid_1's rmse: 280.871
[250]	training's rmse: 276.886	valid_1's rmse: 278.571
[300]	training's rmse: 274.388	valid_1's rmse: 276.27
[350]	training's rmse: 272.159	valid_1's rmse: 274.2
[400]	training's rmse: 270.898	valid_1's rmse: 273.214
[450]	training's rmse: 269.479	valid_1's rmse: 272.054
[500]	training's rmse: 268.171	valid_1's rmse: 270.938
[550]	training's rmse: 267.051	valid_1's rmse: 270.028
[600]	training's rmse: 266.127	valid_1's rmse: 269.363
[650]	training's rmse: 265.2	valid_1's rmse: 268.774
[700]	training's rmse: 264.383	valid_1's rmse: 268.236
[750]	training's rmse: 263.532	valid_1's rmse: 267.624
[800]	training's rmse: 262.787	valid_1's rmse: 267.095
[850]	training's rmse: 262.104	valid_1's rmse: 266.615
[900]	training's rmse: 261.392	valid_1's rmse: 266.099
[950]	training's rmse: 260.741	valid_1's rmse: 265.703
[1000]	training's rmse: 259.985	valid_1's rmse: 265.158
[1050]	training's rmse: 259.318	valid_1's rmse: 264.685
[1100]	training's rmse: 258.737	valid_1's rmse: 264.315
[1150]	training's rmse: 258.152	valid_1's rmse: 263.911
[1200]	training's rmse: 257.561	valid_1's rmse: 263.539
[1250]	training's rmse: 257.078	valid_1's rmse: 263.291
[1300]	training's rmse: 256.588	valid_1's rmse: 263.05
[1350]	training's rmse: 256.097	valid_1's rmse: 262.743
[1400]	training's rmse: 255.657	valid_1's rmse: 262.496
[1450]	training's rmse: 255.14	valid_1's rmse: 262.198
[1500]	training's rmse: 254.676	valid_1's rmse: 262.006
```

```
[1550]	training's rmse: 254.192	valid_1's rmse: 261.731
[1600]	training's rmse: 253.758	valid_1's rmse: 261.521
[1650]	training's rmse: 253.358	valid_1's rmse: 261.345
[1700]	training's rmse: 252.891	valid_1's rmse: 261.069
[1750]	training's rmse: 252.468	valid_1's rmse: 260.888
[1800]	training's rmse: 252.065	valid_1's rmse: 260.683
[1850]	training's rmse: 251.682	valid_1's rmse: 260.54
[1900]	training's rmse: 251.29	valid_1's rmse: 260.367
[1950]	training's rmse: 250.945	valid_1's rmse: 260.26
[2000]	training's rmse: 250.604	valid_1's rmse: 260.141
[2050]	training's rmse: 250.234	valid_1's rmse: 259.975
[2100]	training's rmse: 249.879	valid_1's rmse: 259.846
[2150]	training's rmse: 249.542	valid_1's rmse: 259.691
[2200]	training's rmse: 249.201	valid_1's rmse: 259.572
[2250]	training's rmse: 248.861	valid_1's rmse: 259.456
[2300]	training's rmse: 248.487	valid_1's rmse: 259.282
[2350]	training's rmse: 248.165	valid_1's rmse: 259.162
[2400]	training's rmse: 247.83	valid_1's rmse: 259.039
[2450]	training's rmse: 247.463	valid_1's rmse: 258.883
[2500]	training's rmse: 247.135	valid_1's rmse: 258.745
[2550]	training's rmse: 246.877	valid_1's rmse: 258.667
[2600]	training's rmse: 246.577	valid_1's rmse: 258.571
[2650]	training's rmse: 246.284	valid_1's rmse: 258.507
[2700]	training's rmse: 246.019	valid_1's rmse: 258.403
[2750]	training's rmse: 245.744	valid_1's rmse: 258.319
[2800]	training's rmse: 245.454	valid_1's rmse: 258.228
[2850]	training's rmse: 245.18	valid_1's rmse: 258.123
[2900]	training's rmse: 244.9	valid_1's rmse: 258.064
[2950]	training's rmse: 244.601	valid_1's rmse: 257.92
[3000]	training's rmse: 244.348	valid_1's rmse: 257.824
[3050]	training's rmse: 244.048	valid_1's rmse: 257.721
[3100]	training's rmse: 243.746	valid_1's rmse: 257.603
[3150]	training's rmse: 243.476	valid_1's rmse: 257.54
[3200]	training's rmse: 243.194	valid_1's rmse: 257.501
[3250]	training's rmse: 242.907	valid_1's rmse: 257.421
[3300]	training's rmse: 242.657	valid_1's rmse: 257.391
[3350]	training's rmse: 242.388	valid_1's rmse: 257.297
[3400]	training's rmse: 242.156	valid_1's rmse: 257.235
[3450]	training's rmse: 241.863	valid_1's rmse: 257.143
```

```
[3500]	training's rmse: 241.623	valid_1's rmse: 257.082
[3550]	training's rmse: 241.365	valid_1's rmse: 256.991
[3600]	training's rmse: 241.1	valid_1's rmse: 256.935
[3650]	training's rmse: 240.86	valid_1's rmse: 256.873
[3700]	training's rmse: 240.596	valid_1's rmse: 256.817
[3750]	training's rmse: 240.361	valid_1's rmse: 256.771
[3800]	training's rmse: 240.117	valid_1's rmse: 256.737
[3850]	training's rmse: 239.862	valid_1's rmse: 256.657
[3900]	training's rmse: 239.608	valid_1's rmse: 256.577
[3950]	training's rmse: 239.374	valid_1's rmse: 256.523
[4000]	training's rmse: 239.164	valid_1's rmse: 256.481
[4050]	training's rmse: 238.943	valid_1's rmse: 256.427
[4100]	training's rmse: 238.756	valid_1's rmse: 256.395
[4150]	training's rmse: 238.523	valid_1's rmse: 256.334
[4200]	training's rmse: 238.302	valid_1's rmse: 256.308
[4250]	training's rmse: 238.087	valid_1's rmse: 256.271
[4300]	training's rmse: 237.876	valid_1's rmse: 256.226
[4350]	training's rmse: 237.647	valid_1's rmse: 256.143
[4400]	training's rmse: 237.457	valid_1's rmse: 256.108
[4450]	training's rmse: 237.24	valid_1's rmse: 256.036
[4500]	training's rmse: 237.04	valid_1's rmse: 255.998
[4550]	training's rmse: 236.816	valid_1's rmse: 255.948
[4600]	training's rmse: 236.598	valid_1's rmse: 255.9
[4650]	training's rmse: 236.433	valid_1's rmse: 255.899
[4700]	training's rmse: 236.212	valid_1's rmse: 255.844
[4750]	training's rmse: 236.012	valid_1's rmse: 255.797
[4800]	training's rmse: 235.806	valid_1's rmse: 255.749
[4850]	training's rmse: 235.619	valid_1's rmse: 255.718
[4900]	training's rmse: 235.419	valid_1's rmse: 255.656
[4950]	training's rmse: 235.226	valid_1's rmse: 255.615
[5000]	training's rmse: 234.973	valid_1's rmse: 255.547
[5050]	training's rmse: 234.784	valid_1's rmse: 255.543
[5100]	training's rmse: 234.583	valid_1's rmse: 255.51
[5150]	training's rmse: 234.408	valid_1's rmse: 255.475
[5200]	training's rmse: 234.24	valid_1's rmse: 255.452
[5250]	training's rmse: 234.068	valid_1's rmse: 255.421
[5300]	training's rmse: 233.873	valid_1's rmse: 255.385
[5350]	training's rmse: 233.685	valid_1's rmse: 255.362
[5400]	training's rmse: 233.519	valid_1's rmse: 255.343
```

```
[5450]	training's rmse: 233.313	valid_1's rmse: 255.278
[5500]	training's rmse: 233.138	valid_1's rmse: 255.278
[5550]	training's rmse: 232.952	valid_1's rmse: 255.242
[5600]	training's rmse: 232.769	valid_1's rmse: 255.235
[5650]	training's rmse: 232.483	valid_1's rmse: 255.102
[5700]	training's rmse: 232.296	valid_1's rmse: 255.077
[5750]	training's rmse: 232.146	valid_1's rmse: 255.064
[5800]	training's rmse: 231.982	valid_1's rmse: 255.041
[5850]	training's rmse: 231.81	valid_1's rmse: 255.002
[5900]	training's rmse: 231.615	valid_1's rmse: 255.001
[5950]	training's rmse: 231.444	valid_1's rmse: 254.976
[6000]	training's rmse: 231.246	valid_1's rmse: 254.932
[6050]	training's rmse: 231.074	valid_1's rmse: 254.898
[6100]	training's rmse: 230.895	valid_1's rmse: 254.857
[6150]	training's rmse: 230.725	valid_1's rmse: 254.85
[6200]	training's rmse: 230.55	valid_1's rmse: 254.834
[6250]	training's rmse: 230.396	valid_1's rmse: 254.81
[6300]	training's rmse: 230.229	valid_1's rmse: 254.815
[6350]	training's rmse: 230.064	valid_1's rmse: 254.8
[6400]	training's rmse: 229.917	valid_1's rmse: 254.799
[6450]	training's rmse: 229.612	valid_1's rmse: 254.585
[6500]	training's rmse: 229.441	valid_1's rmse: 254.572
[6550]	training's rmse: 229.257	valid_1's rmse: 254.547
[6600]	training's rmse: 229.083	valid_1's rmse: 254.521
[6650]	training's rmse: 228.92	valid_1's rmse: 254.482
[6700]	training's rmse: 228.759	valid_1's rmse: 254.451
[6750]	training's rmse: 228.583	valid_1's rmse: 254.425
[6800]	training's rmse: 228.394	valid_1's rmse: 254.375
[6850]	training's rmse: 228.219	valid_1's rmse: 254.37
[6900]	training's rmse: 228.051	valid_1's rmse: 254.349
[6950]	training's rmse: 227.89	valid_1's rmse: 254.329
[7000]	training's rmse: 227.734	valid_1's rmse: 254.319
[7050]	training's rmse: 227.587	valid_1's rmse: 254.297
[7100]	training's rmse: 227.428	valid_1's rmse: 254.298
[7150]	training's rmse: 227.278	valid_1's rmse: 254.288
[7200]	training's rmse: 227.118	valid_1's rmse: 254.275
[7250]	training's rmse: 226.957	valid_1's rmse: 254.264
[7300]	training's rmse: 226.805	valid_1's rmse: 254.261
[7350]	training's rmse: 226.644	valid_1's rmse: 254.233
```

```
[7400]  training's rmse: 226.479        valid_1's rmse: 254.211
[7450]  training's rmse: 226.332        valid_1's rmse: 254.213
[7500]  training's rmse: 226.171        valid_1's rmse: 254.186
[7550]  training's rmse: 226.033        valid_1's rmse: 254.168
[7600]  training's rmse: 225.877        valid_1's rmse: 254.159
[7650]  training's rmse: 225.706        valid_1's rmse: 254.122
[7700]  training's rmse: 225.564        valid_1's rmse: 254.108
[7750]  training's rmse: 225.413        valid_1's rmse: 254.088
[7800]  training's rmse: 225.269        valid_1's rmse: 254.08
[7850]  training's rmse: 225.108        valid_1's rmse: 254.043
[7900]  training's rmse: 224.96 valid_1's rmse: 254.048
[7950]  training's rmse: 224.826        valid_1's rmse: 254.043
Early stopping, best iteration is:
[7861]  training's rmse: 225.071        valid_1's rmse: 254.038
valid_error 254.0377180117667
train_error 225.07125865888986
```

In [0]:
```
xgboost =xgb.XGBRegressor(colsample_bytree=0.4603, gamma=0.0468,
                             learning_rate=0.1, max_depth=6,
                             min_child_weight=1.7817, n_estimators=1000
00,
                             reg_alpha=0.4640, reg_lambda=0.8571,
                             subsample=0.5213, silent=1,
                             random_state =7, nthread = -1,tree_method=
'gpu_hist' )
```

In [0]:
```
eval_set = [(X_train.values, Y_train), (val_X.values, val_y)]

xgboost.fit(X_train.values,Y_train,eval_set=eval_set,
         verbose=100, early_stopping_rounds=100)

preds = xgboost.predict(X_train.values)
val_mae2 =np.sqrt(mean_squared_error(Y_train, preds))
print('train_error',val_mae2)
preds = xgboost.predict(val_X.values)
val_mae2 =np.sqrt(mean_squared_error( val_y, preds))
print('vaild_error',val_mae2)
```

```
---------------------------------------------------------------------------
XGBoostError                              Traceback (most recent call last)
<ipython-input-28-5fd006817ab8> in <module>()
      2 
      3 xgboost.fit(X_train.values,Y_train,eval_set=eval_set,
----> 4             verbose=100, early_stopping_rounds=100)
      5 
      6 preds = xgboost.predict(X_train.values)

/usr/local/lib/python3.6/dist-packages/xgboost/sklearn.py in fit(self, X, y, sample_weight, eval_set, eval_metric, early_stopping_rounds, verbose, xgb_model, sample_weight_eval_set, callbacks)
    394                               evals_result=evals_result, obj=obj, feval=feval,
    395                               verbose_eval=verbose, xgb_model=xgb_model,
--> 396                               callbacks=callbacks)
    397 
    398         if evals_result:

/usr/local/lib/python3.6/dist-packages/xgboost/training.py in train(params, dtrain, num_boost_round, evals, obj, feval, maximize, early_stopping_rounds, evals_result, verbose_eval, xgb_model, callbacks, learning_rates)
    214                            evals=evals,
    215                            obj=obj, feval=feval,
--> 216                            xgb_model=xgb_model, callbacks=callbacks)
    217 
    218 

/usr/local/lib/python3.6/dist-packages/xgboost/training.py in _train_internal(params, dtrain, num_boost_round, evals, obj, feval, xgb_model, callbacks)
     72         # Skip the first update if it is a recovery step.
     73         if version % 2 == 0:
---> 74             bst.update(dtrain, i, obj)
```

```
     75                 bst.save_rabit_checkpoint()
     76                 version += 1

/usr/local/lib/python3.6/dist-packages/xgboost/core.py in update(self,
 dtrain, iteration, fobj)
   1107         if fobj is None:
   1108             _check_call(_LIB.XGBoosterUpdateOneIter(self.handl
e, ctypes.c_int(iteration),
-> 1109                                                     dtrain.hand
le))
   1110         else:
   1111             pred = self.predict(dtrain)

/usr/local/lib/python3.6/dist-packages/xgboost/core.py in _check_call(r
et)
    174     """
    175     if ret != 0:
--> 176         raise XGBoostError(py_str(_LIB.XGBGetLastError()))
    177
    178

XGBoostError: [12:38:59] /workspace/src/tree/updater_gpu_hist.cu:1407:
 Exception in gpu_hist: NCCL failure :unhandled cuda error /workspace/s
rc/tree/../common/device_helpers.cuh(896)

Stack trace:
  [bt] (0) /usr/local/lib/python3.6/dist-packages/xgboost/./lib/libxgbo
ost.so(dmlc::LogMessageFatal::~LogMessageFatal()+0x24) [0x7f45dbe6fcb4]
  [bt] (1) /usr/local/lib/python3.6/dist-packages/xgboost/./lib/libxgbo
ost.so(xgboost::tree::GPUHistMakerSpecialised<xgboost::detail::Gradient
PairInternal<double> >::Update(xgboost::HostDeviceVector<xgboost::detai
l::GradientPairInternal<float> >*, xgboost::DMatrix*, std::vector<xgboo
st::RegTree*, std::allocator<xgboost::RegTree*> > const&)+0x1270) [0x7f
45dc0ab7f0]
  [bt] (2) /usr/local/lib/python3.6/dist-packages/xgboost/./lib/libxgbo
ost.so(xgboost::gbm::GBTree::BoostNewTrees(xgboost::HostDeviceVector<xg
boost::detail::GradientPairInternal<float> >*, xgboost::DMatrix*, int,
 std::vector<std::unique_ptr<xgboost::RegTree, std::default_delete<xgbo
ost::RegTree> >, std::allocator<std::unique_ptr<xgboost::RegTree, std::
```

```
default_delete<xgboost::RegTree> > > >*)+0xa81) [0x7f45dbef5791]
  [bt] (3) /usr/local/lib/python3.6/dist-packages/xgboost/./lib/libxgbo
ost.so(xgboost::gbm::GBTree::DoBoost(xgboost::DMatrix*, xgboost::HostDe
viceVector<xgboost::detail::GradientPairInternal<float> >*, xgboost::Ob
jFunction*)+0xd65) [0x7f45dbef6c95]
  [bt] (4) /usr/local/lib/python3.6/dist-packages/xgboost/./lib/libxgbo
ost.so(xgboost::LearnerImpl::UpdateOneIter(int, xgboost::DMatrix*)+0x39
6) [0x7f45dbf09556]
  [bt] (5) /usr/local/lib/python3.6/dist-packages/xgboost/./lib/libxgbo
ost.so(XGBoosterUpdateOneIter+0x35) [0x7f45dbe6caa5]
  [bt] (6) /usr/lib/x86_64-linux-gnu/libffi.so.6(ffi_call_unix64+0x4c)
 [0x7f4610d09dae]
  [bt] (7) /usr/lib/x86_64-linux-gnu/libffi.so.6(ffi_call+0x22f) [0x7f4
610d0971f]
  [bt] (8) /usr/lib/python3.6/lib-dynload/_ctypes.cpython-36m-x86_64-li
nux-gnu.so(_ctypes_callproc+0x2b4) [0x7f4610f1d5c4]
```

In [0]:

```
from keras import optimizers
from keras.utils import plot_model
from keras.models import Sequential, Model
from keras.layers.convolutional import Conv1D, MaxPooling1D
from keras.layers import Dense, LSTM, RepeatVector, TimeDistributed, Fl
atten
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
```

In [0]:

```
epochs = 20
batch = 256
lr = 0.0003
adam = optimizers.Adam(lr)
model_mlp = Sequential()
model_mlp.add(Dense(1024, activation='relu', input_dim=X_train.shape[1
]))
model_mlp.add(Dense(512, activation='relu'))
model_mlp.add(Dense(256, activation='relu'))
model_mlp.add(Dense(128, activation='relu'))
model_mlp.add(Dense(56, activation='relu'))
model_mlp.add(Dense(1))
model_mlp.compile(loss='mse', optimizer=adam)
model_mlp.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_7 (Dense)              (None, 1024)              30720     
_________________________________________________________________
dense_8 (Dense)              (None, 512)               524800    
_________________________________________________________________
dense_9 (Dense)              (None, 256)               131328    
_________________________________________________________________
dense_10 (Dense)             (None, 128)               32896     
_________________________________________________________________
dense_11 (Dense)             (None, 56)                7224      
_________________________________________________________________
dense_12 (Dense)             (None, 1)                 57        
=================================================================
Total params: 727,025
Trainable params: 727,025
Non-trainable params: 0
_________________________________________________________________
```

In [0]:
```
!pip install catboost
```

```
Collecting catboost
  Downloading https://files.pythonhosted.org/packages/ca/ae/aaff63662f7
f5d2af7ec8d61a6f39e78ada9348e5df4f43e665ecc4bea10/catboost-0.21-cp36-no
ne-manylinux1_x86_64.whl (64.0MB)
     |████████████████████████████████| 64.0MB 59kB/s 
Requirement already satisfied: pandas>=0.24.0 in /usr/local/lib/python
3.6/dist-packages (from catboost) (0.25.3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.6/d
ist-packages (from catboost) (3.1.3)
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-pac
kages (from catboost) (1.12.0)
Requirement already satisfied: graphviz in /usr/local/lib/python3.6/dis
```

```
t-packages (from catboost) (0.10.1)
Requirement already satisfied: numpy>=1.16.0 in /usr/local/lib/python3.
6/dist-packages (from catboost) (1.17.5)
Requirement already satisfied: plotly in /usr/local/lib/python3.6/dist-
packages (from catboost) (4.4.1)
Requirement already satisfied: scipy in /usr/local/lib/python3.6/dist-p
ackages (from catboost) (1.4.1)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.
6/dist-packages (from pandas>=0.24.0->catboost) (2018.9)
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/li
b/python3.6/dist-packages (from pandas>=0.24.0->catboost) (2.6.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.
6/dist-packages (from matplotlib->catboost) (0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1
in /usr/local/lib/python3.6/dist-packages (from matplotlib->catboost)
(2.4.6)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/pyth
on3.6/dist-packages (from matplotlib->catboost) (1.1.0)
Requirement already satisfied: retrying>=1.3.3 in /usr/local/lib/python
3.6/dist-packages (from plotly->catboost) (1.3.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/d
ist-packages (from kiwisolver>=1.0.1->matplotlib->catboost) (45.1.0)
Installing collected packages: catboost
Successfully installed catboost-0.21
```

In [0]:

```python
from catboost import CatBoostRegressor
model_cat= CatBoostRegressor(learning_rate=0.03,iterations=1500,depth=1
0 ,verbose = 100

                                       )
eval_set = [(X_train, Y_train), (val_X, val_y)]

model_cat.fit(X_train.values ,Y_train)

preds = model_cat.predict(X_train.values)
val_mae2 =np.sqrt(mean_squared_error( Y_train, preds))
print('train_error',val_mae2)
preds = model_cat.predict(val_X.values)
```

```
val_mae2 =np.sqrt(mean_squared_error( val_y, preds))
print('vaild_error',val_mae2)
```

```
0:      learn: 609.8788004      total: 838ms    remaining: 20m 56s
100:    learn: 303.6896114      total: 1m 19s   remaining: 18m 19s
200:    learn: 286.5177276      total: 2m 37s   remaining: 16m 58s
300:    learn: 278.3525816      total: 3m 55s   remaining: 15m 39s
400:    learn: 272.9706923      total: 5m 14s   remaining: 14m 21s
500:    learn: 268.9150448      total: 6m 33s   remaining: 13m 4s
600:    learn: 265.5572195      total: 7m 52s   remaining: 11m 46s
700:    learn: 262.9148895      total: 9m 11s   remaining: 10m 29s
800:    learn: 260.7549750      total: 10m 30s  remaining: 9m 10s
900:    learn: 258.7712970      total: 11m 50s  remaining: 7m 52s
1000:   learn: 257.0866014      total: 13m 8s   remaining: 6m 33s
1100:   learn: 255.5272372      total: 14m 26s  remaining: 5m 13s
1200:   learn: 254.0033580      total: 15m 42s  remaining: 3m 54s
1300:   learn: 252.5841054      total: 16m 59s  remaining: 2m 35s
1400:   learn: 251.3420703      total: 18m 15s  remaining: 1m 17s
1499:   learn: 250.2308867      total: 19m 30s  remaining: 0us
train_error 250.23088669167072
vaild_error 257.4810369927262
```

In [0]:
```
def ensemble_pred(X) :
  return 0.3 * xgboost.predict(X.values)+0.2*lightgbm.predict(X)+0.5*mo
del_cat.predict(X.values)
```

save model

In [0]:
```
import torch
model_save_name = 'trip_xgb.pt'
path = F"drive/My Drive/IOT Project/{model_save_name}"
torch.save(xgboost, path)
model_save_name = 'trip_lgb.pt'
path = F"drive/My Drive/IOT Project/{model_save_name}"
torch.save(lightgbm, path)
model_save_name = 'trip_cat.pt'
```

```
path = F"drive/My Drive/IOT Project/{model_save_name}"
torch.save(model_cat, path)
```

load the models

In [0]:
```
preds = ensemble_pred(val_X)
val_X['preds'] = preds
val_X['trip_duration'] = val_y
```

In [0]:
```
l = val_X.index.to_list()
```

In [0]:
```
!pip install celluloid
```

```
Requirement already satisfied: celluloid in /usr/local/lib/python3.6/di
st-packages (0.2.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.6/d
ist-packages (from celluloid) (3.1.3)
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.6/
dist-packages (from matplotlib->celluloid) (1.17.5)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.
6/dist-packages (from matplotlib->celluloid) (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/p
ython3.6/dist-packages (from matplotlib->celluloid) (2.6.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1
in /usr/local/lib/python3.6/dist-packages (from matplotlib->celluloid)
(2.4.6)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/pyth
on3.6/dist-packages (from matplotlib->celluloid) (1.1.0)
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-pac
kages (from cycler>=0.10->matplotlib->celluloid) (1.12.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/d
ist-packages (from kiwisolver>=1.0.1->matplotlib->celluloid) (45.1.0)
```

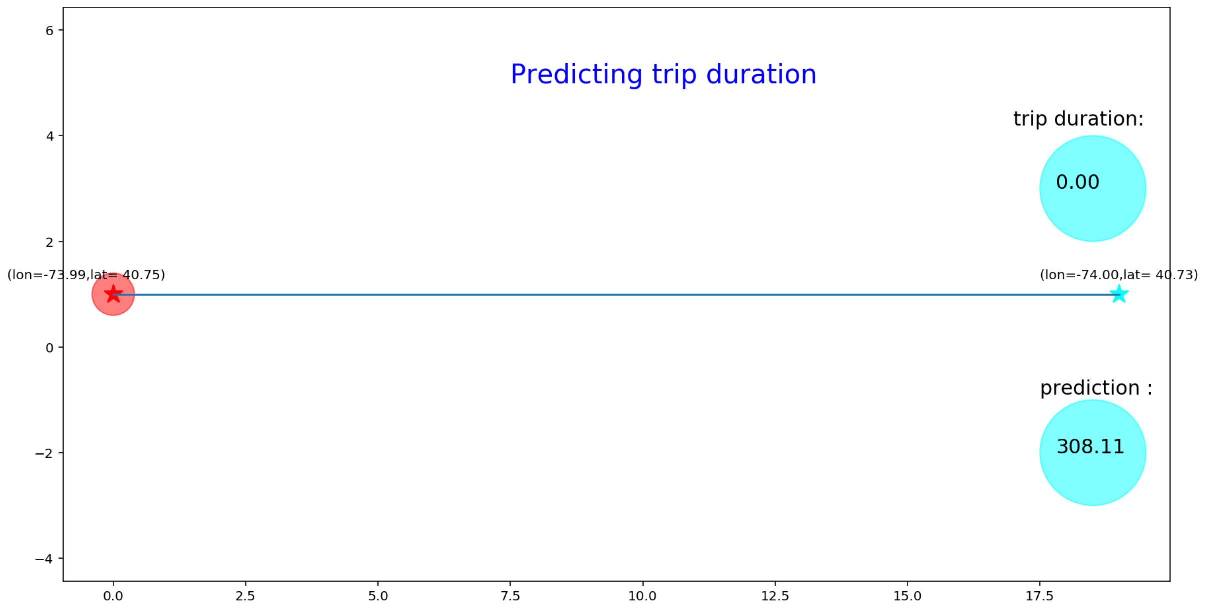
In [0]:
```
example = val_X.loc[522462]
```

In [0]:
```
example
```

Out[0]:
```
passenger_count               1.000000e+00
pickup_longitude             -7.399151e+01
pickup_latitude               4.074979e+01
dropoff_longitude            -7.400244e+01
dropoff_latitude              4.073362e+01
pickup_weekday                0.000000e+00
pickup_hour_weekofyear        7.000000e+00
pickup_hour                   2.200000e+01
pickup_minute                 2.000000e+00
pickup_dt                     3.967357e+06
pickup_week_hour              2.200000e+01
pickup_pca0                  -1.755646e-02
pickup_pca1                   2.630611e-03
dropoff_pca0                 -2.754136e-02
dropoff_pca1                  1.940308e-02
distance_haversine            2.020263e+00
distance_dummy_manhattan      2.719041e+00
direction                    -1.528724e+02
pca_manhattan                 2.675737e-02
center_latitude               4.074170e+01
center_longitude             -7.399697e+01
diff_lat                      1.617050e-02
diff_long                     1.093292e-02
pickup_cluster                4.000000e+00
dropoff_cluster               1.400000e+01
mean_clus                     4.000970e+00
mean_hour                     4.367874e+00
mean_weekday                  4.136401e+00
mean_weekday_hour             4.877751e+00
preds                         3.081132e+02
trip_duration                 2.760000e+02
Name: 522462, dtype: float64
```

In [0]:
```
from matplotlib import animation
from celluloid import Camera
from matplotlib.animation import FuncAnimation
x = [1 for i in range(0,20)]
fig, ax = plt.subplots(figsize=(15,8))
```

```
circle=plt.Circle((0,1),0.4,color='r', alpha=0.5)
ax.add_artist(circle)
ax.plot(0,1,'*',markersize=15,color='r')
ax.plot(19,1,'*',markersize=15,color='#00ffff')
ax.plot(x)
ax.text(-2,1.3,'(lon=%.2f'%example['pickup_longitude']+',lat= %.2f)' %e
xample['pickup_latitude'],{'color': 'k', 'fontsize': 10})
ax.text(17.5,1.3,'(lon=%.2f'%example['dropoff_longitude']+',lat= %.2f)'
 %example['dropoff_latitude'],{'color': 'k', 'fontsize': 10})
ax.text(17,4.2,'trip duration:',{'color': 'k', 'fontsize': 15})
ax.text(17.5,-0.9,'prediction :',{'color': 'k', 'fontsize': 15})
ax.text(7.5,5,'Predicting trip duration',{'color': 'b', 'fontsize': 20
})
circle1=plt.Circle((18.5,3),1,color='#00ffff', alpha=0.5)
ax.add_artist(circle1)
circle2=plt.Circle((18.5,-2),1,color='#00ffff', alpha=0.5)
ax.add_artist(circle2)
ax.set_aspect('equal', adjustable='datalim')
text1 = ax.text(17.8,-2,'%.2f'%example['preds'],{'color': 'k', 'fontsiz
e': 15})
text2 = ax.text(17.8,3,'0',{'color': 'k', 'fontsize': 15})
n = len(np.arange(0,19,0.01))
l = np.linspace(0,example['trip_duration'],n)

def animate(i):
  circle.set_center((i,1))

  text2.set_text('%.2f'%l[int(i*100)])

animation = FuncAnimation(fig,func=animate,frames = np.arange(0,19,0.01
),interval=10 )
```



In [0]:
```
animation.save('first_ani.mp4', fps=100, extra_args=['-vcodec', 'libx26
4'])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call l
ast)
<ipython-input-1-9c1315a3749b> in <module>()
----> 1 animation.save('first_ani.mp4', fps=100, extra_args=['-vcodec',
 'libx264'])

NameError: name 'animation' is not defined
```

In [0]: