In [0]:
```
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/IOT Project/20140711.CSV')
```

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

Out[0]:

| | TripID | RouteID | StopID | StopName | dayBeginning | NumberOfBoardings |
|---|---|---|---|---|---|---|
| **0** | 23631 | 100 | 14156 | 181 Cross Rd | 2013-06-30 00:00:00 | 1 |
| **1** | 23631 | 100 | 14144 | 177 Cross Rd | 2013-06-30 00:00:00 | 1 |
| **2** | 23632 | 100 | 14132 | 175 Cross Rd | 2013-06-30 00:00:00 | 1 |
| **3** | 23633 | 100 | 12266 | Zone A Arndale Interchange | 2013-06-30 00:00:00 | 2 |
| **4** | 23633 | 100 | 14147 | 178 Cross Rd | 2013-06-30 00:00:00 | 1 |

In [0]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?
client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleuser
content.com&redirect_uri=urn%3aietf%3awg%3aoauth%3a2.0%3aoob&response_t
ype=code&scope=email%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdocs.t
est%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive%20https%3a%2f%2fw
ww.googleapis.com%2fauth%2fdrive.photos.readonly%20https%3a%2f%2fwww.go
ogleapis.com%2fauth%2fpeopleapi.readonly

Enter your authorization code:
··········
Mounted at /content/drive
```

## Creat train set

In [0]:
```
 new = train.groupby(['dayBeginning','StopID'])['NumberOfBoardings'].ag
g({'boardingperday':'sum'})
```

In [0]:
```
del train
```

In [0]:
```
new = new.reset_index()
```

In [0]:
```
new=new[new['boardingperday']<1000]
```

In [0]:
```
new["dayBeginning"]=pd.to_datetime(new["dayBeginning"])
```

In [0]:
```
a = new['StopID'].value_counts()
a = a[a==54]
```

In [0]:
```
l = a.index.to_list()
tr = new[new['StopID']==l[0]]
for i in l[1:] :
  b =  new[new['StopID']==i]
  tr = tr.append(b, ignore_index = True)
```

### Transform the data into a time series

In [0]:
```
def series_to_supervised(data, window=1, lag=1, dropnan=True):
    cols, names = list(), list()
    # Input sequence (t-n, ... t-1)
    for i in range(window, 0, -1):
        cols.append(data.shift(i))
        names += [('%s(t-%d)' % (col, i)) for col in data.columns]
    # Current timestep (t=0)
    cols.append(data)
    names += [('%s(t)' % (col)) for col in data.columns]
    # Target timestep (t=lag)
    cols.append(data.shift(-lag))
    names += [('%s(t+%d)' % (col, lag)) for col in data.columns]
    # Put it all together
    agg = pd.concat(cols, axis=1)
    agg.columns = names
    # Drop rows with NaN values
    if dropnan:
        agg.dropna(inplace=True)
    return agg
```

In [0]:
```
tr.head()
```

Out[0]:

| | dayBeginning | StopID | boardingperday |
|---|---|---|---|
| **0** | 2013-06-30 | 10235 | 12 |
| **1** | 2013-07-07 | 10235 | 11 |
| **2** | 2013-07-14 | 10235 | 8 |
| **3** | 2013-07-21 | 10235 | 21 |
| **4** | 2013-07-28 | 10235 | 24 |

In [0]:
```
def build_train( stop_id,lag_size = 1):
  window = 25
  lag = lag_size
  test = tr.copy()
  test =test[test['StopID']==stop_id]
  test =test.sort_values('dayBeginning')
  test = series_to_supervised(test.set_index(['dayBeginning','StopID'
]), window=window, lag=lag)

  return test
```

In [0]:
```
series = build_train(l[0]).reset_index()
for i in l[1:] :
  a=build_train(i).reset_index()
  series =series.append(a, ignore_index = True)
```

In [0]:
```
series=series.set_index(["dayBeginning",'StopID'],1)
```

In [0]:
```
series.head()
```

Out[0]:

| | | boardingperday(t-25) | boardingperday(t-24) | boardingperday(t-23) | boardingperday(t-22) |
|---|---|---|---|---|---|
| **dayBeginning** | **StopID** | | | | |
| **2013-12-22** | **10235** | 12.0 | 11.0 | 8.0 | 21.0 |
| **2013-12-29** | **10235** | 11.0 | 8.0 | 21.0 | 24.0 |

| dayBeginning | StopID | boardingperday(t-25) | boardingperday(t-24) | boardingperday(t-23) | boardingperday(t-22) |
|---|---|---|---|---|---|
| 2014-01-05 | 10235 | 8.0 | 21.0 | 24.0 | 20.0 |
| 2014-01-12 | 10235 | 21.0 | 24.0 | 20.0 | 13.0 |
| 2014-01-19 | 10235 | 24.0 | 20.0 | 13.0 | 21.0 |

## Modeling

In [0]:
```
# Label
labels_col = 'boardingperday(t+%d)' % 1
labels = series[labels_col]
series = series.drop(labels_col, axis=1)
```

In [0]:
```
X_train, X_valid, Y_train, Y_valid = train_test_split(series, labels.va
lues, test_size=0.4, random_state=0)
print('Train set shape', X_train.shape)
print('Validation set shape', X_valid.shape)
```

```
Train set shape (82034, 26)
Validation set shape (54690, 26)
```

## Modeling with Ligthgbm

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

model_lgb = lgb.LGBMRegressor(objective='regression',num_leaves=8,
                              learning_rate=0.1, n_estimators=4000,
                              max_bin = 55, bagging_fraction = 0.9,
                              bagging_freq = 5, feature_fraction = 0.43
19,
                              feature_fraction_seed=9, bagging_seed=15,
                              min_data_in_leaf =6,  tree_method='gpu_hi
st' , boosting_type = 'gbdt',
                              metric = 'rmse',min_sum_hessian_in_leaf =
 14)
model_lgb=model_lgb.fit(X_train, Y_train,eval_set=[(X_valid,Y_valid)],
        verbose=50, early_stopping_rounds=100)
preds = model_lgb.predict(X_train)
#np.sqrt(-cross_val_score(model_lgb, train, Y, cv=5, scoring="neg_mean_
squared_error")).mean()
val_mae2 =np.sqrt(mean_squared_error( Y_train, preds ))
print(val_mae2)
```

```
Training until validation scores don't improve for 100 rounds.
[50]    valid_0's rmse: 30.1862
[100]   valid_0's rmse: 27.9186
[150]   valid_0's rmse: 26.7257
[200]   valid_0's rmse: 25.7652
[250]   valid_0's rmse: 25.1483
[300]   valid_0's rmse: 24.7366
[350]   valid_0's rmse: 24.3856
[400]   valid_0's rmse: 24.1344
[450]   valid_0's rmse: 23.8947
[500]   valid_0's rmse: 23.7354
[550]   valid_0's rmse: 23.5512
[600]   valid_0's rmse: 23.419
[650]   valid_0's rmse: 23.2805
```

```
[700]   valid_0's rmse: 23.1851
[750]   valid_0's rmse: 23.1236
[800]   valid_0's rmse: 23.004
[850]   valid_0's rmse: 22.9244
[900]   valid_0's rmse: 22.8649
[950]   valid_0's rmse: 22.7743
[1000]  valid_0's rmse: 22.7197
[1050]  valid_0's rmse: 22.6754
[1100]  valid_0's rmse: 22.6375
[1150]  valid_0's rmse: 22.6128
[1200]  valid_0's rmse: 22.5809
[1250]  valid_0's rmse: 22.5567
[1300]  valid_0's rmse: 22.5393
[1350]  valid_0's rmse: 22.4898
[1400]  valid_0's rmse: 22.4579
[1450]  valid_0's rmse: 22.4375
[1500]  valid_0's rmse: 22.4046
[1550]  valid_0's rmse: 22.386
[1600]  valid_0's rmse: 22.3667
[1650]  valid_0's rmse: 22.342
[1700]  valid_0's rmse: 22.3276
[1750]  valid_0's rmse: 22.3021
[1800]  valid_0's rmse: 22.2754
[1850]  valid_0's rmse: 22.2702
[1900]  valid_0's rmse: 22.2439
[1950]  valid_0's rmse: 22.2326
[2000]  valid_0's rmse: 22.2193
[2050]  valid_0's rmse: 22.2083
[2100]  valid_0's rmse: 22.1811
[2150]  valid_0's rmse: 22.1779
[2200]  valid_0's rmse: 22.1645
[2250]  valid_0's rmse: 22.1646
[2300]  valid_0's rmse: 22.1437
[2350]  valid_0's rmse: 22.1422
[2400]  valid_0's rmse: 22.1407
[2450]  valid_0's rmse: 22.1391
[2500]  valid_0's rmse: 22.127
[2550]  valid_0's rmse: 22.1281
[2600]  valid_0's rmse: 22.125
[2650]  valid_0's rmse: 22.138
```

```
[2700]  valid_0's rmse: 22.1309
Early stopping, best iteration is:
[2604]  valid_0's rmse: 22.1219
16.3308079660714
```

Modeling with xgboost

In [0]:
```
model_xgb =xgb.XGBRegressor(colsample_bytree=0.4603, gamma=0.468,
                            learning_rate=0.005, max_depth=6,
                            min_child_weight=1.7817, n_estimators=1000
0,
                            reg_alpha=0.4640, reg_lambda=2.871,
                            subsample=0.5213, silent=1,
                            random_state =7, nthread = -1,
                              eval_metric='rmse',tree_method='gpu_his
t'
                           )

eval_set = [(X_train, Y_train), (X_valid, Y_valid)]

model_xgb=model_xgb.fit(X_train, Y_train,eval_set=eval_set,
        verbose=100, early_stopping_rounds=100)
```

```
[0]     validation_0-rmse:148.984       validation_1-rmse:147.641
Multiple eval metrics have been passed: 'validation_1-rmse' will be use
d for early stopping.

Will train until validation_1-rmse hasn't improved in 100 rounds.
[100]   validation_0-rmse:94.7234       validation_1-rmse:93.8491
[200]   validation_0-rmse:62.7299       validation_1-rmse:62.2698
[300]   validation_0-rmse:44.593        validation_1-rmse:44.5658
[400]   validation_0-rmse:34.7624       validation_1-rmse:35.1913
[500]   validation_0-rmse:29.6514       validation_1-rmse:30.481
[600]   validation_0-rmse:26.909        validation_1-rmse:28.0515
[700]   validation_0-rmse:25.3423       validation_1-rmse:26.7352
[800]   validation_0-rmse:24.3089       validation_1-rmse:25.8922
[900]   validation_0-rmse:23.5631       validation_1-rmse:25.3165
[1000]  validation_0-rmse:22.9885       validation_1-rmse:24.8897
[1100]  validation_0-rmse:22.5229       validation_1-rmse:24.5535
```

```
[1200]  validation_0-rmse:22.1433       validation_1-rmse:24.2906

[1300]  validation_0-rmse:21.8057       validation_1-rmse:24.0682
[1400]  validation_0-rmse:21.4959       validation_1-rmse:23.8647
[1500]  validation_0-rmse:21.2213       validation_1-rmse:23.6915
[1600]  validation_0-rmse:20.9658       validation_1-rmse:23.5274
[1700]  validation_0-rmse:20.7189       validation_1-rmse:23.3711
[1800]  validation_0-rmse:20.4899       validation_1-rmse:23.2255
[1900]  validation_0-rmse:20.267        validation_1-rmse:23.0831
[2000]  validation_0-rmse:20.061        validation_1-rmse:22.9634
[2100]  validation_0-rmse:19.8611       validation_1-rmse:22.849
[2200]  validation_0-rmse:19.67 validation_1-rmse:22.7387
[2300]  validation_0-rmse:19.4888       validation_1-rmse:22.6361
[2400]  validation_0-rmse:19.3196       validation_1-rmse:22.5393
[2500]  validation_0-rmse:19.1638       validation_1-rmse:22.4547
[2600]  validation_0-rmse:19.0212       validation_1-rmse:22.3774
[2700]  validation_0-rmse:18.8812       validation_1-rmse:22.3062
[2800]  validation_0-rmse:18.7391       validation_1-rmse:22.2342
[2900]  validation_0-rmse:18.6053       validation_1-rmse:22.1667
[3000]  validation_0-rmse:18.4648       validation_1-rmse:22.0939
[3100]  validation_0-rmse:18.3429       validation_1-rmse:22.0365
[3200]  validation_0-rmse:18.2257       validation_1-rmse:21.9825
[3300]  validation_0-rmse:18.1103       validation_1-rmse:21.9274
[3400]  validation_0-rmse:17.999        validation_1-rmse:21.8818
[3500]  validation_0-rmse:17.8802       validation_1-rmse:21.8257
[3600]  validation_0-rmse:17.7706       validation_1-rmse:21.7792
[3700]  validation_0-rmse:17.6655       validation_1-rmse:21.7357
[3800]  validation_0-rmse:17.5624       validation_1-rmse:21.6948
[3900]  validation_0-rmse:17.4621       validation_1-rmse:21.6518
[4000]  validation_0-rmse:17.3658       validation_1-rmse:21.6122
[4100]  validation_0-rmse:17.2709       validation_1-rmse:21.5746
[4200]  validation_0-rmse:17.1783       validation_1-rmse:21.5377
[4300]  validation_0-rmse:17.0839       validation_1-rmse:21.4991
[4400]  validation_0-rmse:16.9937       validation_1-rmse:21.4659
[4500]  validation_0-rmse:16.903        validation_1-rmse:21.4324
[4600]  validation_0-rmse:16.8142       validation_1-rmse:21.4002
[4700]  validation_0-rmse:16.7322       validation_1-rmse:21.3722
[4800]  validation_0-rmse:16.651        validation_1-rmse:21.3429
[4900]  validation_0-rmse:16.5727       validation_1-rmse:21.3178
[5000]  validation_0-rmse:16.4943       validation_1-rmse:21.2908
```

```
[5100]  validation_0-rmse:16.4133       validation_1-rmse:21.2625

[5200]  validation_0-rmse:16.3402       validation_1-rmse:21.241
[5300]  validation_0-rmse:16.2662       validation_1-rmse:21.2192
[5400]  validation_0-rmse:16.1914       validation_1-rmse:21.1948
[5500]  validation_0-rmse:16.1199       validation_1-rmse:21.1742
[5600]  validation_0-rmse:16.046        validation_1-rmse:21.1543
[5700]  validation_0-rmse:15.9749       validation_1-rmse:21.1352
[5800]  validation_0-rmse:15.908        validation_1-rmse:21.1173
[5900]  validation_0-rmse:15.8436       validation_1-rmse:21.0988
[6000]  validation_0-rmse:15.7804       validation_1-rmse:21.0825
[6100]  validation_0-rmse:15.7156       validation_1-rmse:21.0624
[6200]  validation_0-rmse:15.6532       validation_1-rmse:21.0465
[6300]  validation_0-rmse:15.5883       validation_1-rmse:21.0282
[6400]  validation_0-rmse:15.5302       validation_1-rmse:21.0153
[6500]  validation_0-rmse:15.468        validation_1-rmse:20.999
[6600]  validation_0-rmse:15.4036       validation_1-rmse:20.9825
[6700]  validation_0-rmse:15.34 validation_1-rmse:20.9656
[6800]  validation_0-rmse:15.2804       validation_1-rmse:20.951
[6900]  validation_0-rmse:15.2212       validation_1-rmse:20.9351
[7000]  validation_0-rmse:15.1659       validation_1-rmse:20.9219
[7100]  validation_0-rmse:15.109        validation_1-rmse:20.9077
[7200]  validation_0-rmse:15.0524       validation_1-rmse:20.8956
[7300]  validation_0-rmse:14.9967       validation_1-rmse:20.8817
[7400]  validation_0-rmse:14.9427       validation_1-rmse:20.8692
[7500]  validation_0-rmse:14.8881       validation_1-rmse:20.8565
[7600]  validation_0-rmse:14.8334       validation_1-rmse:20.8434
[7700]  validation_0-rmse:14.7775       validation_1-rmse:20.833
[7800]  validation_0-rmse:14.7221       validation_1-rmse:20.8203
[7900]  validation_0-rmse:14.6686       validation_1-rmse:20.8076
[8000]  validation_0-rmse:14.6173       validation_1-rmse:20.7967
[8100]  validation_0-rmse:14.5662       validation_1-rmse:20.785
[8200]  validation_0-rmse:14.5182       validation_1-rmse:20.7747
[8300]  validation_0-rmse:14.4688       validation_1-rmse:20.7632
[8400]  validation_0-rmse:14.4194       validation_1-rmse:20.7539
[8500]  validation_0-rmse:14.3687       validation_1-rmse:20.7435
[8600]  validation_0-rmse:14.3202       validation_1-rmse:20.7326
[8700]  validation_0-rmse:14.273        validation_1-rmse:20.7232
[8800]  validation_0-rmse:14.2259       validation_1-rmse:20.7146
[8900]  validation_0-rmse:14.1786       validation_1-rmse:20.7049
```

```
[9000]  validation_0-rmse:14.132        validation_1-rmse:20.6948

[9100]  validation_0-rmse:14.086        validation_1-rmse:20.6869
[9200]  validation_0-rmse:14.0406       validation_1-rmse:20.6781
[9300]  validation_0-rmse:13.9948       validation_1-rmse:20.6706
[9400]  validation_0-rmse:13.9504       validation_1-rmse:20.6628
[9500]  validation_0-rmse:13.9056       validation_1-rmse:20.6537
[9600]  validation_0-rmse:13.8627       validation_1-rmse:20.6464
[9700]  validation_0-rmse:13.8202       validation_1-rmse:20.6404
[9800]  validation_0-rmse:13.7771       validation_1-rmse:20.6336
[9900]  validation_0-rmse:13.735        validation_1-rmse:20.6269
[9999]  validation_0-rmse:13.6944       validation_1-rmse:20.6202
```

In [0]:
```
def ensemble_model(X) :
  return 0.9*model_xgb.predict(X)+0.1*model_lgb.predict(X)
```

## using the model

In [0]:
```
 tr['boardingperday']= tr['boardingperday'].astype('float')
```

In [0]:
```
l=[10502,10504,10508,10527,10530]
```

In [0]:
```
forcast = []
real = []
for j in l :
  bus_stop = tr[tr['StopID']==j]
  bus_stop = bus_stop.reset_index()
  forcasting = []
  realty=[]
  for i in range(22,28):
    val = ensemble_model(series_to_supervised(bus_stop.iloc[i:26+i].set
_index(['dayBeginning','StopID','index']), window=25, lag=0).drop('boar
dingperday(t+0)',1))
    forcasting.append(val[0])
    realty.append(bus_stop.iloc[26+i]['boardingperday'])
```

```
  forcast.append(forcasting)
  real.append(realty)
```

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

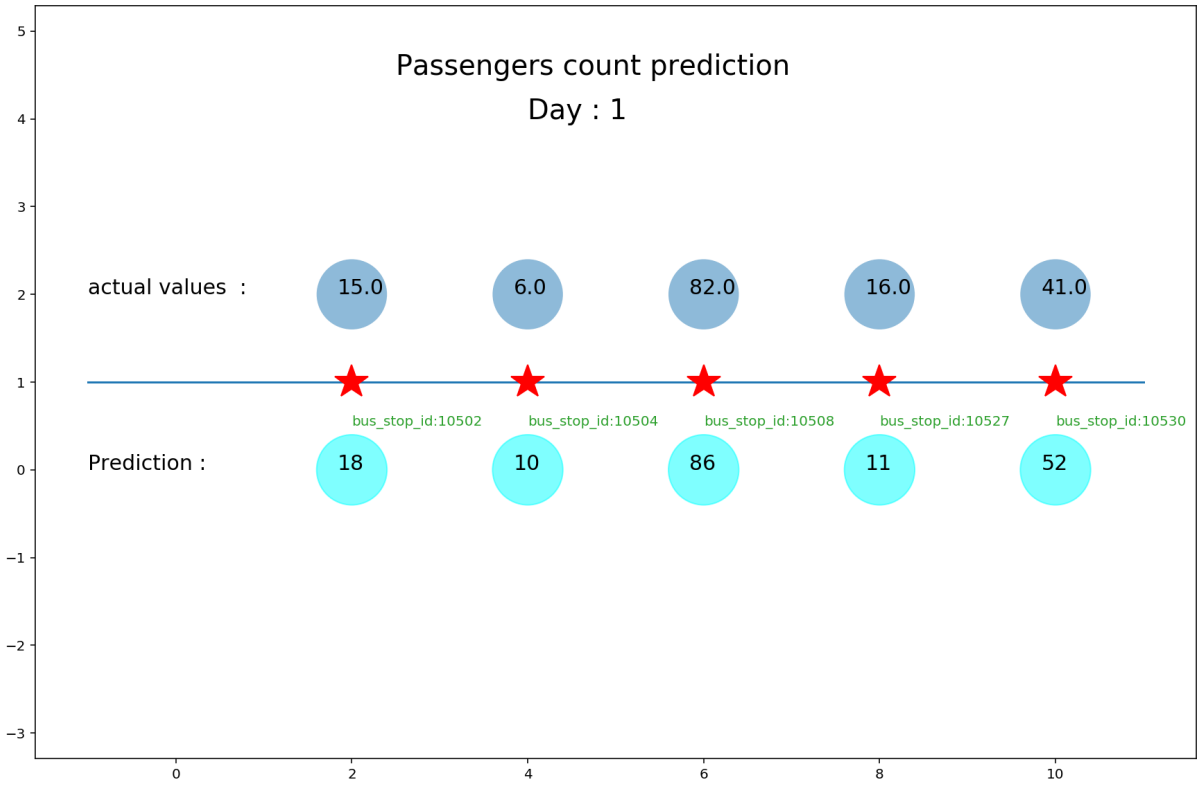
```
Requirement already satisfied: celluloid in /usr/local/lib/python3.6/di
st-packages (0.2.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.6/d
ist-packages (from celluloid) (3.1.3)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.
6/dist-packages (from matplotlib->celluloid) (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->celluloid)
(2.4.6)
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: kiwisolver>=1.0.1 in /usr/local/lib/pyth
on3.6/dist-packages (from matplotlib->celluloid) (1.1.0)
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.6/
dist-packages (from matplotlib->celluloid) (1.17.5)
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]:
```
from matplotlib import animation
from celluloid import Camera
from matplotlib.animation import FuncAnimation
y=[1 for i in range(-1,12)]
x = [i for i in range(-1,12)]
l=[10502,10504,10508,10527,10530]
fig, ax = plt.subplots(figsize=(15,10))

ax.plot(x,y)
ax.set_aspect('equal', adjustable='datalim')
for i in range(1,6):
  ide = str(l[i-1])
  circle=plt.Circle((i*2,2),0.4, alpha=0.5)
  ax.add_artist(circle)
```

```
  circle=plt.Circle((i*2,0),0.4,color='#00ffff', alpha=0.5)
  ax.add_artist(circle)
  ax.plot(i*2,1,'*',markersize=25,color='r')
  ax.text(i*2, 0.5, 'bus_stop_id:'+ide, {'color': 'C2', 'fontsize': 10
})
text11=ax.text((1-0.08)*2, 0, '', {'color': 'k', 'fontsize': 15})
text12=ax.text((1-0.08)*2, 2, '', {'color': 'k', 'fontsize': 15})
text21=ax.text((2-0.08)*2, 0, '', {'color': 'k', 'fontsize': 15})
text22=ax.text((2-0.08)*2, 2, '', {'color': 'k', 'fontsize': 15})
text31=ax.text((3-0.08)*2, 0, '', {'color': 'k', 'fontsize': 15})
text32=ax.text((3-0.08)*2, 2, '', {'color': 'k', 'fontsize': 15})
text41=ax.text((4-0.08)*2, 0, '', {'color': 'k', 'fontsize': 15})
text42=ax.text((4-0.08)*2, 2, '', {'color': 'k', 'fontsize': 15})
text51=ax.text((5-0.08)*2, 0, '', {'color': 'k', 'fontsize': 15})
text52=ax.text((5-0.08)*2, 2, '', {'color': 'k', 'fontsize': 15})
ax.text( -1,2,'actual values  :', {'color': 'k', 'fontsize': 15})
ax.text( -1,0,'Prediction : ', {'color': 'k', 'fontsize': 15})
text = ax.text(4,4,'',{'color': 'k', 'fontsize': 20})
ax.text(2.5,4.5,'Passengers count prediction',{'color': 'k', 'fontsize'
: 20})
def animate(i):
    if i%1 == 0 :
      i = int(i)+1
      text11.set_text(str(int(forcast[0][i])))
      text12.set_text(str(real[0][i]))
      text21.set_text(str(int(forcast[1][i])))
      text22.set_text(str(real[1][i]))
      text31.set_text(str(int(forcast[2][i])))
      text32.set_text(str(real[2][i]))
      text41.set_text(str(int(forcast[3][i])))
      text42.set_text(str(real[3][i]))
      text51.set_text(str(int(forcast[4][i])))
      text52.set_text(str(real[4][i]))
      text.set_text('Day : '+str(i))


animation = FuncAnimation(fig,func=animate,frames = np.arange(0,4,0.01
),interval=100 )
```




In [0]:
```
animation.save('first_ani.mp4', fps=25, extra_args=['-vcodec', 'libx264'])
```

In [0]:
```

```