In [ ]:

```
import speech_recognition as sr
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.svm import SVR
from xgboost import XGBRegressor
from pygeocoder import Geocoder
import pandas as pd
import numpy as np
import usaddress


record=sr.Recognizer()
addr=[]
with sr.Microphone() as source:
    print("Started Listening->>> Start speaking")
    audio_voice=record.listen(source)
    print("Listening Stopped")


try:
    addr.append(record.recognize_google(audio_voice))
    print("here is your audio to text",speech);

except:
    print("exception")
    pass;
```

```
address=usaddress.parse(addr)
address=[key[0] for key in address if key[1]!='Recipient']
address=' '.join(address)
print(address)

data = pd.read_csv('C:/Users/owner/Desktop/16.csv') #Enter proper path
data['zip']="XYZZ"
data=data.dropna(how='any')
# # data=data[0:800]

for row_index in data.index: #Enter your key
    results = Geocoder("A****************").reverse_geocode(data['Y'][row_inde
x], data['X'][row_index])
    data['zip'][row_index]=results.postal_code


final_dataframe = data[['Date', 'DayOfWeek', 'Time', 'zip']]
final_dataframe.loc[:, 'Date'] = pd.to_datetime(final_dataframe['Date'])

final_dataframe.loc[:, 'Hour'] = pd.to_datetime(final_dataframe['Time'])
final_dataframe.loc[:, 'Hour'] = final_dataframe.Hour.apply(lambda x: x.hour)
final_dataframe = final_dataframe[final_dataframe.Date.dt.month < 11]
final_dataframe = final_dataframe[['DayOfWeek', 'zip', 'Hour']]
final_dataframe.loc[:, 'Crimes'] = 1
grouped_dataframe = final_dataframe.groupby(['DayOfWeek', 'zip', 'Hour']).sum
().reset_index()
grouped_dataframe = grouped_dataframe[['Crimes', 'Hour', 'DayOfWeek', 'zip']]
```

```
grouped_dataframe = pd.get_dummies(grouped_dataframe)
x_train = grouped_dataframe.iloc[:, 1:]
y_train = grouped_dataframe.iloc[:, 0]


data = pd.read_csv('C:/Users/owner/Desktop/17.csv') #Enter proper path
data['zip']="XYZZ"
data=data.dropna(how='any')
data=data[0:20]
print(data.head())
for row_index in data.index:  #Enter your key
    results = Geocoder("A****************************").reverse_geocode(data
['Y'][row_index], data['X'][row_index])
    data['zip'][row_index]=results.postal_code


final_dataframe = data[['DayOfWeek', 'Time', 'zip']]


final_dataframe.loc[:, 'Hour'] = pd.to_datetime(final_dataframe['Time'])
final_dataframe.loc[:, 'Hour'] = final_dataframe.Hour.apply(lambda x: x.hour)
final_dataframe = final_dataframe[['DayOfWeek', 'zip', 'Hour']]
final_dataframe.loc[:, 'Crimes'] = 1
grouped_dataframe = final_dataframe.groupby(['DayOfWeek', 'zip', 'Hour']).coun
t().reset_index()

grouped_dataframe = grouped_dataframe[['Crimes', 'Hour', 'DayOfWeek', 'zip']]
grouped_dataframe = pd.get_dummies(grouped_dataframe)
x_test = grouped_dataframe.iloc[:, 1:]
y_test = grouped_dataframe.iloc[:, 0]
```

```
#linear Regression
linear_regression = LinearRegression()
linear_regression.fit(x_train,y_train)
linear_regression.score(x_test, y_test)


#Random Forest
rf = RandomForestRegressor()
rf.fit(x_train,y_train)
rf.score(x_test, y_test)


#KNN
knn = KNeighborsRegressor()
knn.fit(x_train,y_train)
knn.score(x_test, y_test)

#SVM
svm = SVR()
svm.fit(x_train,y_train)
svm.score(x_test, y_test)


#XGBoost
xgb = XGBRegressor()
xgb.fit(x_train,y_train)
xgb.score(x_test, y_test)


#MLP Regressor
mlp = MLPRegressor(hidden_layer_sizes = (100,100,100,100), random_state=444)
```