**REPORT ON SUMMER INTERNSHIP PROJECT**

**“POKEMON GO”**

--POKEMON PREDICTION USING

MACHINE LEARNING MODELS!

PREPARED BY:

**PRIYANSHU GUPTA**

**ACKNOWLEDGEMENT**

It is my pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behavior, and acts during the course of Data Science.

I express my sincere gratitude to Mr. Yogendra Sir & Mr. Sylvester Sir for their teaching, support, guidance and motivation provided to me during the internship program.

I also extend my sincere appreciation to Kunal Sir & Rohit Sir for their mentorship, valuable suggestions and precious time in accomplishing my project.

Lastly, I would like to thank the almighty and my friends with whom I shared my day-to-day experience and received lots of suggestions that improved my quality of work.

PRIYANSHU GUPTA

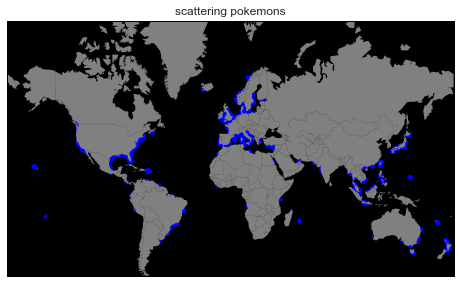
**ABSTRACT**

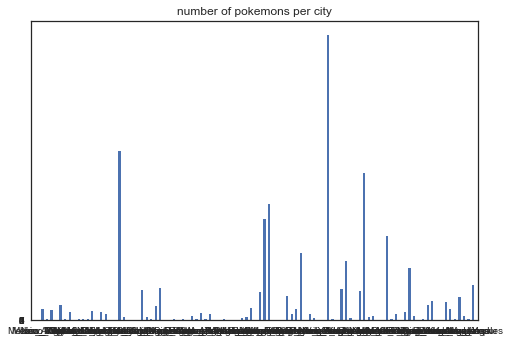
The first week of the internship period was spent in learning basic python coding and solving considerable number of code challenges. To get better acquainted with the design and coding techniques, various hands-on and quiz were employed. Through these codes, we got well acquainted with Anaconda design tools such as Spyder (IPython console, variable explorer), Jupyter and many more advanced tools.

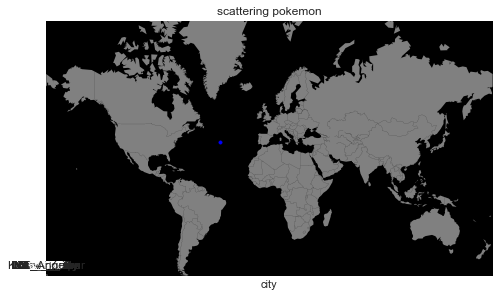
The second & third week of the very same were spent in learning various libraries involved in the Data Science. Data Preprocessing which included filling out missing data in the dataset, Encoding of categorical data, Scaling of data and many more followed by various imported Regression Algorithms including splitting of test&train data into it like Linear Regression, Logistic Regression. Following Regression, Classification methods were introduced which then followed by some outlier topics like Data scrapping, Apriori and many more.

After getting well-versed with all the algorithms, the project “POKEMON GO” was started. The code was written in Python (2.7 version) and implemented on Spyder tool. The project aims at predicting the location (Latitude&Longitude) of all the generation-1 Pokemon (151 in number) using well suited regression-classifier algorithms, given the data in .csv file format.

**LIST OF FIGURES**

****

****

****

**TABLE OF CONTENTS**

1. Acknowledgement……………………………………………………………………..3

2. Abstract……………………………………………………………………………………..4

3. List of figures……………………………………………………………………………..5

4. Introduction……………………………………………………………………………….7

5. Background theory/Motivation………………………………………………….8

6. Methodology……………………………………………………………………………..9

7. Result Analysis…………………………………………………………………………..10

8. Conclusion…………………………………………………………………………………11

9. Reference………………………………………………………………………………….12

10. ANNEXURE……………………………………………………………………………...13

**INTRODUCTION**

Theassigned project “POKEMON GO” on Data Science aims at finding the location of all the Pokemons individually of generation 1st (1-151 in counting), given the data in .csv file format containing all the basic functional details of Pokemon like city, continent Pokemon is hitting; day & date of appearance; weather at the time of Pokemon hit; Lattitude & Longitude corresponding to Pokemon; co-occurance with other functions and many more.

The project is categorized in five problem set; demanding plotting of Pokemon on world map, city having maximum number of Pokemon counts (10 most frequent cities) & plotting of same on Bar graph, classification and prediction of Pokemon as per continent-wise, mapping all the Pokemon names with their given ID’s, prediction of Pokemon (location prediction) respectively.

**BACKGROUND THEORY**

Data Science is the science of collecting all the past data and then searching for patterns in this data.

Machine Learning is the science of creating algorithms and program which learn on their own.

Pokemon Go, a location-based, augmented reality mobile game released by Niantic Inc. The players uses the GPS to locate, capture, and battle fictional creatures in a virtual setting. Like many millennials, I grew up playing Pokemon games on my Gameboy Color. So when this game was released, I was undoubtedly joined the rank of fellow Pokemon Go trainer, running on the streets of Jaipur City looking for the rarest Pokemon. Although the game was a huge success, there were many flaws that need to be addressed, and new features to be implemented. One of which was a way to locate and predict location of Pokemon. As per the assignment, I developed a pokemon prediction project to offer a potential solution for this problem –POKEMON GO.

**METHODOLOGY**

The project is divided into 5 problem set, each set demanding different algorithms to be employed.

Data Pre-processing part included label encoding of categorical data i.e., ‘names’ column (which was mapped as per fourth problem set into the dataset).

Regression method used is Linear Regression method as the fifth problem set (prediction location of pokemon), the latitude and longitude have linear dependency on names and appeared hour column. Hence employing Linear Regression algorithm, location of pokemon is predicted.

Classification of pokemon according to continents have been done (third problem set) using two different dataframes, namely id and continents and then latitude and longitude into it.

Data visualization included plotting of all the pokemons on the world map, plotting of all cities having pokemon frequency on the bar map, and lastly predicted location of next pokemon on the world map.

**RESULT ANALYSIS**

As previously mentioned in the background theory, Pokemon Go struggles with some flaws, and hence this project was aimed at dealing with all those flaws, as a result, project demands five different functions.

First one is, plotting of all the pokemon on the world map, which is done using Basemap function of Matplot library.

Second is, plotting cities with highest pokemon count on bar map and seaborn of matplotlib; Newyork comes out to be city with highest pokemon count.

Third is, classification of pokemon as per continents, which is done using 2 different dataframes; one containing id and continents and second predicting location into the same continent.

Fourth is, mapping of pokemon names with their corresponding ID, names have been fetched from given url (1st generation pokemon).

Fifth, the last one, is prediction of pokemon of your choice; linear regression have been employed in the process between highest impactful column (appeared hour) and pokemon names and then further using its result in predicting location (Latitude & Longitude).

**CONCLUSION**

The project “POKEMON GO –POKEMON PREDICTION USING MACHINE LEARNING MODELS” ended up acknowledging an individual the dependency between two data entities, how to visualize data on globe, how to categorize data and then finding result, and many more.

Using this project, rather than random searching for a pokemon, player can seek for the location corresponding the pokemon which the player want to catch and then reaching that predicted location directly, fetching the pokemon.

This project acknowledge us how to implement different data science- machine learning algorithms like Regression, Classification, Matplotlib and many more. As Python (2.7) is the most user interfacing language, hence all these algorithms were applied under python library storage.

**ANNEXURE**

**#importing basic libraries**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

**#fetching data into dataset**

dataset=pd.read\_csv('300k.csv')

ylong= dataset.iloc[:,2:3].values

ylat= dataset.iloc[:,1:2].values

**''' FIRST PROBLEM SET '''**

**#importing library for ploting on map**

from mpl\_toolkits.basemap import Basemap

**# Defining the projection, scale, the corners of the map, and the resolution.**

m = Basemap(projection='merc',llcrnrlat=-55,urcrnrlat=80,\

llcrnrlon=-180,urcrnrlon=180,lat\_ts=20,resolution='c')

**# Coloring the continents**

m.fillcontinents(color='grey',lake\_color='grey')

**#drawing boundary for countries**

m.drawcountries(linewidth=0.1)

**# fill in the oceans**

m.drawmapboundary(fill\_color='black')

**#scattering the pokemons**

m.scatter(ylong,ylat,latlon=True,marker='.',color='blue',zorder=1)

plt.title("scattering pokemons")

plt.show()

**''' SECOND PROBLEM SET '''**

city=dataset.iloc[:,21] #fetching city coloumn

from sklearn.preprocessing import LabelEncoder

le= LabelEncoder()

city\_encoded= le.fit\_transform(city)

**#calculating city frequency**

import collections

frequency= dict (collections.Counter(city\_encoded))

#print frequency

city\_names=dataset.iloc[:,21]

frequency\_value=list(frequency.values())

frequency\_key=list(frequency.keys())

city\_frequency = collections.Counter(city\_names)

print "The city having maximum number of pokemon and its frequency is:",city\_frequency.most\_common(1)

print "Cities having highest number of pokemon count are:",city\_frequency.most\_common(10)

**#plotting bar graph**

xlocations = np.array(range(len(frequency\_value)))+0.5

width = 0.5

plt.bar(xlocations, frequency\_value, width=width)

plt.yticks(range(0, 8))

plt.xticks(xlocations+ width/2, city\_names)

plt.xlim(0, xlocations[-1]+width\*2)

plt.title("number of pokemons per city")

plt.gca().get\_xaxis().tick\_bottom()

plt.gca().get\_yaxis().tick\_left()

plt.show()

q=zip(city\_names, city\_encoded)

import seaborn as sns

sns.axes\_style('white')

sns.set\_style('white')

colors = ['pink' if \_y >=0 else 'red' for \_y in frequency\_value]

ax = sns.barplot(city\_names, frequency\_value, palette=colors)

for n, (label, \_y) in enumerate(zip(city\_names, frequency\_value)):

ax.annotate(

s='{:.1f}'.format(abs(\_y)),

xy=(n, \_y),

ha='center',va='center',

xytext=(0,10),

textcoords='offset points',

color='white',

weight='bold'

)

ax.annotate(

s=label,

xy=(n, 0),

ha='center',va='center',

xytext=(0,10),

textcoords='offset points',

)

**# axes formatting**

ax.set\_yticks([])

ax.set\_xticks([])

sns.despine(ax=ax, bottom=True, left=True)

**''' THIRD PROBLEM SET '''**

df\_latitude=pd.DataFrame(dataset['latitude'][dataset['continent']=='Asia'])

df\_longitude=pd.DataFrame(dataset['longitude'][dataset['continent']=='Asia'])

df\_lat\_long=df\_latitude.join(df\_longitude)

df\_continent=pd.DataFrame(dataset['continent'])

df\_id\_asia=pd.DataFrame(dataset['pokemonId'][dataset['continent']=='Asia'])

from sklearn.linear\_model import LinearRegression

regressor=LinearRegression()

regressor.fit(df\_id\_asia,df\_lat\_long)

y\_pred=regressor.predict(99)

print "Your pokemon is at(classification by continents)",y\_pred

**'''FOURTH PROBLEM SET'''**

**#1st generation pokemon names(1-151)**

Names=['Bulbasaur','Ivysaur','Venusaur','Charmander','Charmeleon','Charizard','Squirtle','Wartortle','Blastoise','Caterpie','Metapod','Butterfree','Weedle','Kakuna','Beedrill','Pidgey','Pidgeotto','Pidgeot','Rattata','Raticate','Spearow','Fearow','Ekans','Arbok','Pikachu','Raichu','Sandshrew','Sandslash','Nidoran','Nidorina','Nidoqueen','Nidoran♂','Nidorino','Nidoking','Clefairy','Clefable','Vulpix','Ninetales','Jigglypuff','Wigglytuff','Zubat','Golbat','Oddish','Gloom','Vileplume','Paras','Parasect','Venonat','Venomoth','Diglett','Dugtrio','Meowth','Persian','Psyduck','Golduck','Mankey','Primeape','Growlithe','Arcanine','Poliwag','Poliwhirl','Poliwrath','Abra','Kadabra','Alakazam','Machop','Machoke','Machamp','Bellsprout','Weepinbell','Victreebel','Tentacool','Tentacruel','Geodude','Graveler','Golem','Ponyta','Rapidash','Slowpoke','Slowbro','Magnemite','Magneton','Farfetch','Doduo','Dodrio','Seel','Dewgong','Grimer','Muk','Shellder','Cloyster','Gastly','Haunter','Gengar','Onix','Drowzee','Hypno','Krabby','Kingler','Voltorb','Electrode','Exeggcute','Exeggutor','Cubone','Marowak','Hitmonlee','Hitmonchan','Lickitung','Koffing','Weezing','Rhyhorn','Rhydon','Chansey','Tangela','Kangaskhan','Horsea','Seadra','Goldeen','Seaking','Staryu','Starmie','Mr. Mime','Scyther','Jynx','Electabuzz','Magmar','Pinsir','Tauros','Magikarp','Gyarados','Lapras','Ditto','Eevee','Vaporeon','Jolteon','Flareon','Porygon','Omanyte','Omastar','Kabuto','Kabutops','Aerodactyl','Snorlax','Articuno','Zapdos','Moltres','Dratini','Dragonair','Dragonite','Mewtwo','Mew']

ID=[]

for i in range(1,152):

ID.append(i)

Pokemon\_configuration=zip(ID,Names)

Pokemon\_configuration= dict (Pokemon\_configuration)

dataset['names']=dataset['class'].map(Pokemon\_configuration)

**''' FIFTH PROBLEM SET '''**

corr=dataset.corr()

c=corr['longitude'].sort\_values(ascending=False)

**#correlation function showing that Appeared hour has most impact on longitude and latitude, hence it will also come under prediction**

Xnames=dataset.iloc[:,-1].values **#names column**

ylat= dataset.iloc[:,1:2].values **#longitude column**

ylong= dataset.iloc[:,2:3].values **#latitude column**

X=dataset.iloc[:,13:14].values  **#Appeared hour column**

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

Encoded\_names=le.fit\_transform(Xnames)

Encoded\_names=np.asarray(Encoded\_names)

Encoded\_names=Encoded\_names.reshape(len(Encoded\_names),1)

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

**#Regression function for Appeared hour and Encoded names**

reg= LinearRegression()

reg.fit(Encoded\_names,X)

pred1= reg.predict(Encoded\_names)

**#regression function for longitude**

reg\_long = LinearRegression()

reg\_long.fit(pred1,ylong)

ylong\_pred= reg\_long.predict(pred1)

**#regression function for lattitude**

reg\_lat = LinearRegression()

reg\_lat.fit(pred1,ylat)

ylat\_pred= reg\_lat.predict(pred1)

n=raw\_input("Enter name of pokemon whose location to be found(with proper casing):")

z=zip(Xnames,Encoded\_names)

r= dict (z) #Dictionary containing pokemon names and their encoded value

print "Latitude for",n,"is:",reg\_lat.predict(reg.predict(r[n]))

print "Longitude for",n,"is:",reg\_long.predict(reg.predict(r[n]))

**#importing library for ploting on map**

from mpl\_toolkits.basemap import Basemap

m = Basemap(projection='merc',llcrnrlat=-55,urcrnrlat=80,\

llcrnrlon=-180,urcrnrlon=180,lat\_ts=20,resolution='c')

**# Coloring the continents**

m.fillcontinents(color='grey',lake\_color='grey')

**#drawing boundary for countries**

m.drawcountries(linewidth=0.1)

**# fill in the oceans**

m.drawmapboundary(fill\_color='black')

**#scattering the pokemon**

m.scatter(reg\_long.predict(reg.predict(r[n])),reg\_lat.predict(reg.predict(r[n])),latlon=True,marker='.',color='blue',zorder=1)

plt.title("scattering pokemon")

plt.show()