Assignment 1

H-1B Visa Petitions 2011-2016 Analysis and Prediction

Bhanuja Nagore | CSYE 7245, Northeastern University

Contents

H-1B Visa Petitions 2011-2016

Abstract

The H1B Visa enables foreign skilled workers to stay in the US for up to six years and apply for permanent residency. Because this visa permits a foreign national to have "dual intent", there is no conflict when an H1B temporary worker applies for a green card. It is important for the U.S. Department of Labor to identify which employers are sending the most number of H1B Visa applications and what is the percentage share of the annual 85,000 visa cap for the employers with most applications.
At the same time, figure out the most common job title, examine the data science jobs, and their relative wages for which the application is coming in larger frequency, the most popular job and its demand in future.
It will provide correlation between visa application attributes and the final status outcome. By performing a classification task, which takes in the information about applicants' prevailing wage, employer name, state the worksite belongs to, occupation code, full-time status and predicts the visa status as 'DENIED' or 'CERTIFIED'.
This study provides information about the employment gaps within the U.S., which are filled by foreign workers. This gives an idea as to which areas the U.S. government should emphasize to encourage domestic students to develop more local workforce in those job domains for the future.

Introduction

This study on the H1B Visa answers some of the questions that could be formed regarding H1B applications in the U.S. H1B Visa applications are increasing year by year and this study helps to identify the jobs that are being filled by the foreign workers. Also, Data Science is a new field since 2010, and this study provides information about the rising demand of the jobs related to the field of data science.
The source of the data set is the Office of Foreign Labor Certification (OFLC) – United States Department of Labor.
Some interesting questions that are answered in this study.
1) What has been the status of the petitioned applications in the time frame.?
2) Which employers send the most number of H-1B visa applications?
3) Most common job titles and their relative wages to the Industry standard as well as absolute?
4) What percentages of the job is full time and what percentage is part-time?
5) Which are the most common cities where the H1B employee lands in?
6) How have the Data Science related jobs has performed.
7) The demand of most popular job in the future.

H-1B Visa Data

The H-1B is an employment-based, non-immigrant visa category for temporary foreign workers in the United States. For a foreign national to apply for H1-B visa, an US employer must offer a job and

petition for H-1B visa with the US immigration department. This is the most common visa status applied for and held by international students once they complete college/ higher education (Masters, PhD) and work in a full-time position.
The Office of Foreign Labor Certification (OFLC) generates program data that is useful information about the immigration programs including the H1-B visa. The disclosure data updated annually is available at https://www.foreignlaborcert.doleta.gov/performancedata.cfm
In this project, I will analyze over 3 million records of H-1B petitions in the period 2011-2016.
Source of Data set: https://www.kaggle.com/nsharan/h-1b-visa

Dataset Description

The relevant columns include:

1) **EMPLOYER_NAME**: Name of employer submitting the H1-B application. Used in comparing salaries and number of applications of various employers.
2) **JOB_TITLE:** Title of the job using which we can filter specific job positions for e.g., Data Scientist, Data Engineer etc.
3)**PREVAILING_WAGE:** The prevailing wage for a job position is defined as the average wage paid to similarly employed workers in the requested occupation in intended employment. The prevailing wage is based on the employer's minimum requirements for the position
4) **WORKSITE_CITY, WORKSITE_STATE:** The foreign worker's intended area of employment. We will explore the relationship between prevailing wage for Data Scientist position across different locations.
5) **CASE_STATUS:** Status associated with the last significant event or decision. Valid values include "Certified," "Certified-Withdrawn," Denied," and "Withdrawn". This feature will help us analyze what share of the H-1B visa is taken by different employers/ job positions.
Other important columns include Unit of Pay and whether the Job position is a Full-Time position or a Part-Time position.

6) **YEAR:** 2016 and 2015

7) **FULLTIME:** 'Y' or 'N'

Code:

Methodology

Reading the input file:

In [2]:
```
#Reading the Input File
h1b=pd.read_csv("D:/Subjects/course7245/h1b_kaggle.csv/h1b_kaggle.csv")
h1b.head()
```

Out[2]:

	Unnamed: 0	CASE_STATUS	EMPLOYER_NAME	SOC_NAME	JOB_TITLE	FULL_TIME_POSITION	PREVAILING_WAGE
0	1	CERTIFIED-WITHDRAWN	UNIVERSITY OF MICHIGAN	BIOCHEMISTS AND BIOPHYSICISTS	POSTDOCTORAL RESEARCH FELLOW	N	36067.0
1	2	CERTIFIED-WITHDRAWN	GOODMAN NETWORKS, INC.	CHIEF EXECUTIVES	CHIEF OPERATING OFFICER	Y	242674.0
2	3	CERTIFIED-WITHDRAWN	PORTS AMERICA GROUP, INC.	CHIEF EXECUTIVES	CHIEF PROCESS OFFICER	Y	193066.0
3	4	CERTIFIED-WITHDRAWN	GATES CORPORATION, A WHOLLY-OWNED SUBSIDIARY O...	CHIEF EXECUTIVES	REGIONAL PRESIDEN, AMERICAS	Y	220314.0
4	5	WITHDRAWN	PEABODY INVESTMENTS CORP.	CHIEF EXECUTIVES	PRESIDENT MONGOLIA AND INDIA	Y	157518.4

Data Cleaning

Cleaning the null and inappropriate values

In [4]:
```
#Data cleaning
#Null values in the dataset.So checking how many non null values are there for each column
h1b[h1b.isnull()==False].count()    #not null values for each column
```

Out[4]:
```
Unnamed: 0            3002458
CASE_STATUS           3002445
EMPLOYER_NAME         3002399
SOC_NAME              2984724
JOB_TITLE             3002415
FULL_TIME_POSITION    3002443
PREVAILING_WAGE       3002373
YEAR                  3002445
WORKSITE              3002458
lon                   2895216
lat                   2895216
dtype: int64
```

In [5]:
```
h1b.isnull().any()  #checking for null values if present or not
```

Out[5]:
```
Unnamed: 0            False
CASE_STATUS            True
EMPLOYER_NAME          True
SOC_NAME               True
JOB_TITLE              True
FULL_TIME_POSITION     True
```

Count of null values in each column

```
In [6]: #Null values in ascending order
        h1b.isnull().sum().sort_values(ascending = False)
```

```
Out[6]: lat                  107242
        lon                  107242
        SOC_NAME              17734
        PREVAILING_WAGE          85
        EMPLOYER_NAME            59
        JOB_TITLE                43
        FULL_TIME_POSITION       15
        YEAR                     13
        CASE_STATUS              13
        WORKSITE                  0
        Unnamed: 0                0
        dtype: int64
```

```
In [7]: #drooping non appropriate values from the data and then analyzing the type of data
        h1b.drop(["Unnamed: 0"],inplace = True,axis = 1)
        h1b.head()
        h1b.info()
```

```
        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 3002458 entries, 0 to 3002457
        Data columns (total 10 columns):
        CASE_STATUS           object
```

```
Number of entries: 3002458
Number of missing data in each column:
CASE_STATUS                13
EMPLOYER_NAME              59
SOC_NAME                17734
JOB_TITLE                  43
FULL_TIME_POSITION         15
PREVAILING_WAGE            85
YEAR                       13
WORKSITE                    0
lon                    107242
lat                    107242
dtype: int64
```

Cleaned Data

```
#Removed bad data.and now its clean data named h1b
h1b.isnull().sum().sort_values(ascending = False)
```

```
lat                    0
lon                    0
WORKSITE               0
YEAR                   0
PREVAILING_WAGE        0
FULL_TIME_POSITION     0
JOB_TITLE              0
SOC_NAME               0
EMPLOYER_NAME          0
CASE_STATUS            0
dtype: int64
```

```
h1b.isna().sum().sort_values(ascending = False)
```

```
lat                    0
lon                    0
WORKSITE               0
YEAR                   0
PREVAILING_WAGE        0
FULL_TIME_POSITION     0
JOB_TITLE              0
SOC_NAME               0
```

no null values
h1b.isnull().any()

```
CASE_STATUS            False
EMPLOYER_NAME          False
SOC_NAME               False
JOB_TITLE              False
FULL_TIME_POSITION     False
PREVAILING_WAGE        False
YEAR                   False
WORKSITE               False
lon                    False
lat                    False
dtype: bool
```

Total number of rows in each column
h1b[h1b.isnull()==False].count()

```
CASE_STATUS            2877765
EMPLOYER_NAME          2877765
SOC_NAME               2877765
JOB_TITLE              2877765
FULL_TIME_POSITION     2877765
PREVAILING_WAGE        2877765
YEAR                   2877765
WORKSITE               2877765
lon                    2877765
lat                    2877765
dtype: int64
```

Exploratory analysis: Before building a model, it is important to explore the data. Every variable has to be analyzed and different statistics such as mean, maximum, minimum needs to be checked to handle the outliers.

Missing values should be investigated properly to check their significance.

```
In [21]: FULL_TIME_POSITION_cat=h1b.FULL_TIME_POSITION.unique()
         JOB_TITLE_cat

         # count of the values in dataset full time positions are there or not
         h1b.groupby('FULL_TIME_POSITION').count()
```

Out[21]:

FULL_TIME_POSITION	CASE_STATUS	EMPLOYER_NAME	SOC_NAME	JOB_TITLE	PREVAILING_WAGE	YEAR	WORKSITE
N	408044	408044	408044	408044	408044	408044	408044
Y	2469721	2469721	2469721	2469721	2469721	2469721	2469721

Grouped the dataset by Year

```
In [22]: YEAR_cat=h1b.YEAR.unique()
         YEAR_cat

         # count of the values in dataset
         h1b.groupby('YEAR').count()
```

Out[22]:

YEAR	CASE_STATUS	EMPLOYER_NAME	SOC_NAME	JOB_TITLE	FULL_TIME_POSITION	PREVAILING_WAGE	WORKSITE	
2011.0	333625	333625	333625	333625	333625	333625	333625	333
2012.0	394267	394267	394267	394267	394267	394267	394267	394
2013.0	422427	422427	422427	422427	422427	422427	422427	422
2014.0	498027	498027	498027	498027	498027	498027	498027	498
2015.0	600120	600120	600120	600120	600120	600120	600120	600
2016.0	629299	629299	629299	629299	629299	629299	629299	629

Statistics of Dataset and mean values of salary per year

```
[23]: # What are the summary statistics - describe the dataset
      h1b.PREVAILING_WAGE.describe()
```

```
[23]: count    2.877765e+06
      mean     1.451666e+05
      std      5.307856e+06
      min      0.000000e+00
      25%      5.460000e+04
      50%      6.512500e+04
      75%      8.151500e+04
      max      6.997607e+09
      Name: PREVAILING_WAGE, dtype: float64
```

```
[24]: # mean of the year with Wage in the dataset
      h1b.groupby('YEAR').mean().PREVAILING_WAGE
```

```
[24]: YEAR
      2011.0    194288.060074
      2012.0    176106.822690
      2013.0    194009.647824
      2014.0    181762.305063
      2015.0     91661.156577
      2016.0     89016.344110
      Name: PREVAILING_WAGE, dtype: float64
```

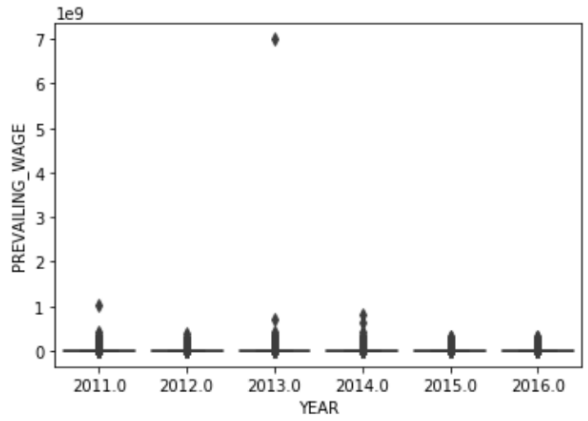
Median of the CASE_STATUS with Wage in the dataset
h1b.groupby('CASE_STATUS').median().PREVAILING_WAGE

```
CASE_STATUS
CERTIFIED                                                    65312.0
CERTIFIED-WITHDRAWN                                          64230.0
DENIED                                                       60216.0
INVALIDATED                                                 190195.0
PENDING QUALITY AND COMPLIANCE REVIEW - UNASSIGNED           75171.0
REJECTED                                                     57886.0
WITHDRAWN                                                    65674.0
Name: PREVAILING_WAGE, dtype: float64
```

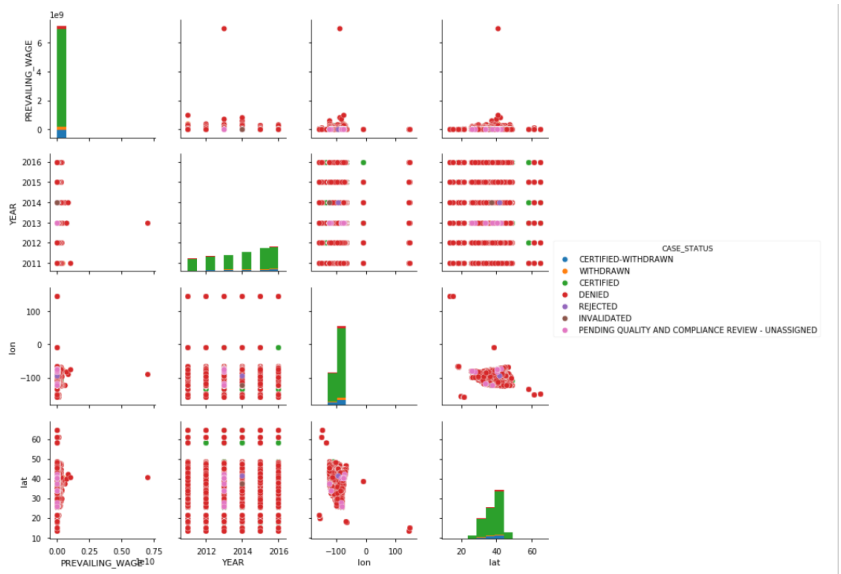
Boxplot for one group only and can be seen that there is outliers in the Prevailing_wage variable

```
sns.boxplot( x=h1b["YEAR"], y=h1b["PREVAILING_WAGE"] )
#sns.plt.show()
#It can be seen that the outlier is in year 2013
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1f300143358>
```



Quick plot of the data using seaborn



Top 20 Job titles with maximum Jobs and the top most is PROGRAMMER ANALYST

```
Job_volume = h1b['JOB_TITLE'].value_counts()[:20]
Job_volume
```

PROGRAMMER ANALYST	244274
SOFTWARE ENGINEER	119063
COMPUTER PROGRAMMER	68793
SYSTEMS ANALYST	60924
SOFTWARE DEVELOPER	41951
BUSINESS ANALYST	39134
COMPUTER SYSTEMS ANALYST	34138
TECHNOLOGY LEAD - US	28308
SENIOR SOFTWARE ENGINEER	26667
TECHNOLOGY ANALYST - US	26033
ASSISTANT PROFESSOR	24458
SENIOR CONSULTANT	24153
CONSULTANT	22643
PROJECT MANAGER	19046
DATABASE ADMINISTRATOR	16121
PHYSICAL THERAPIST	14239
RESEARCH ASSOCIATE	13419
COMPUTER PROGRAMMER ANALYST	13142
ACCOUNTANT	12934
DEVELOPER	12737

Top 20 Work Site with maximum Jobs and the top most is New York where maximum jobs are present

```
WORKSITE_volume = h1b['WORKSITE'].value_counts()[:20]
WORKSITE_volume
```

NEW YORK, NEW YORK	190864
HOUSTON, TEXAS	83385
SAN FRANCISCO, CALIFORNIA	62457
ATLANTA, GEORGIA	52008
CHICAGO, ILLINOIS	51167
SAN JOSE, CALIFORNIA	49582
SUNNYVALE, CALIFORNIA	34968
DALLAS, TEXAS	31509
BOSTON, MASSACHUSETTS	31336
REDMOND, WASHINGTON	30574
CHARLOTTE, NORTH CAROLINA	30176
IRVING, TEXAS	29316
MOUNTAIN VIEW, CALIFORNIA	29245
SAN DIEGO, CALIFORNIA	28656
SANTA CLARA, CALIFORNIA	27945
JERSEY CITY, NEW JERSEY	26822
SEATTLE, WASHINGTON	26745
AUSTIN, TEXAS	26695
LOS ANGELES, CALIFORNIA	26393
PHILADELPHIA, PENNSYLVANIA	24104

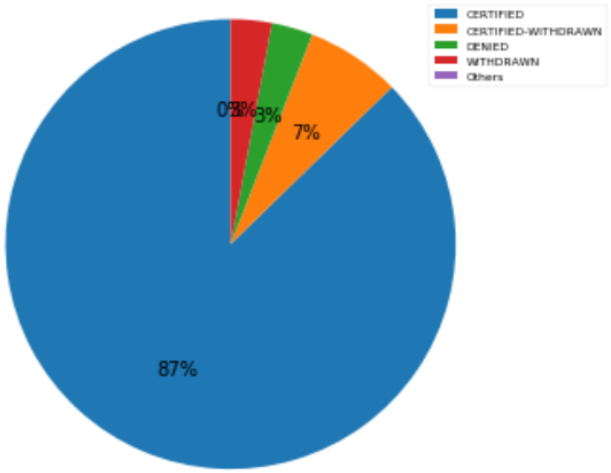
```
In [218]: #Analyze the H1Bs by the status of their visa applications
statusCount = h1b['CASE_STATUS'].value_counts()
statusTypes = statusCount.index.copy(deep=True)
statusTypes = statusTypes.values
statusTypes[4] = 'Others'
statusTypes = statusTypes[0:5]

statusValues = statusCount.copy()
statusValues = statusValues.values
statusValues[4] = np.sum(statusValues[4:7])
statusValues = statusValues[0:5]
print(statusTypes)
print(statusValues)

fig1 = plt.pie(statusValues, autopct='%1.0f%%', shadow=False, startangle=90)
plt.legend(statusTypes, loc="best", prop={'size':6})
plt.axis('equal')
plt.tight_layout()
plt.show()
```

It shows that there are 87% certified cases and 3% denied visa petition

```
['CERTIFIED' 'CERTIFIED-WITHDRAWN' 'DENIED' 'WITHDRAWN' 'Others']
[2512114  195721   85161   84752      17]
```



Correlation between H1bs per capita and voters. There is a correlation between the no of legal immigrants and the total population state wise.

In [50]:
```
#Correlation between H1bs per capita and voters. There is a correlation between the no of legal immigrants and the
#total population state wise.
H1bCity,H1bState = h1b['WORKSITE'].str.split(', ',1).str #Get the states values

#Putting everything in a dataframe
H1bStateCounts = H1bState.value_counts()
H1bStateCounts = H1bStateCounts.to_frame()
H1bStateCounts.columns = ['People']
H1bStateCounts = H1bStateCounts[H1bStateCounts.index!='NA']
H1bStateCounts = H1bStateCounts.sort_index()

#Importing population data from the 2010 census and merging it with the dataframe with the H1b values
PopulationInfo = [4779736, 710231, 6392017, 2915918, 37253956, 5029196,
           3574097, 897934, 601723, 18801310, 9687653, 1360301,
           1567582, 12830632, 6483802, 3046355, 2853118, 4339367,
           4533372, 1328361, 5773552, 6547629, 9883640, 5303925,
           2967297, 5988927, 989415, 1826341, 2700551, 1316470, 8791894,
           2059179, 19378102, 9535483, 672591, 11536504, 3751351, 3831074,
           12702379, 3725789, 1052567, 4625364, 814180, 6346105, 25145561,
           2763885, 625741, 8001024, 6724540, 1852994, 5686986, 563626]
H1bStateCounts['TotalPopulation'] = PopulationInfo


#Compute the H1bs per capita means the total number of immigrants with respect to the total population of the state
H1bStateCounts['H1bPerCapita'] = np.divide(H1bStateCounts['People'].values,H1bStateCounts['TotalPopulation'].values)
```

	People	TotalPopulation	H1bPerCapita
DISTRICT OF COLUMBIA	22408	601723	0.037240
NEW JERSEY	206876	8791894	0.023530
DELAWARE	17705	897934	0.019717
MASSACHUSETTS	112659	6547629	0.017206
WASHINGTON	100754	6724540	0.014983
CALIFORNIA	548539	37253956	0.014724
NEW YORK	282429	19378102	0.014575
CONNECTICUT	48174	3574097	0.013479
ILLINOIS	154843	12830632	0.012068
TEXAS	286704	25145561	0.011402
VIRGINIA	86948	8001024	0.010867
RHODE ISLAND	11316	1052567	0.010751
GEORGIA	99679	9687653	0.010289
MARYLAND	53145	5773552	0.009205

Model Building:

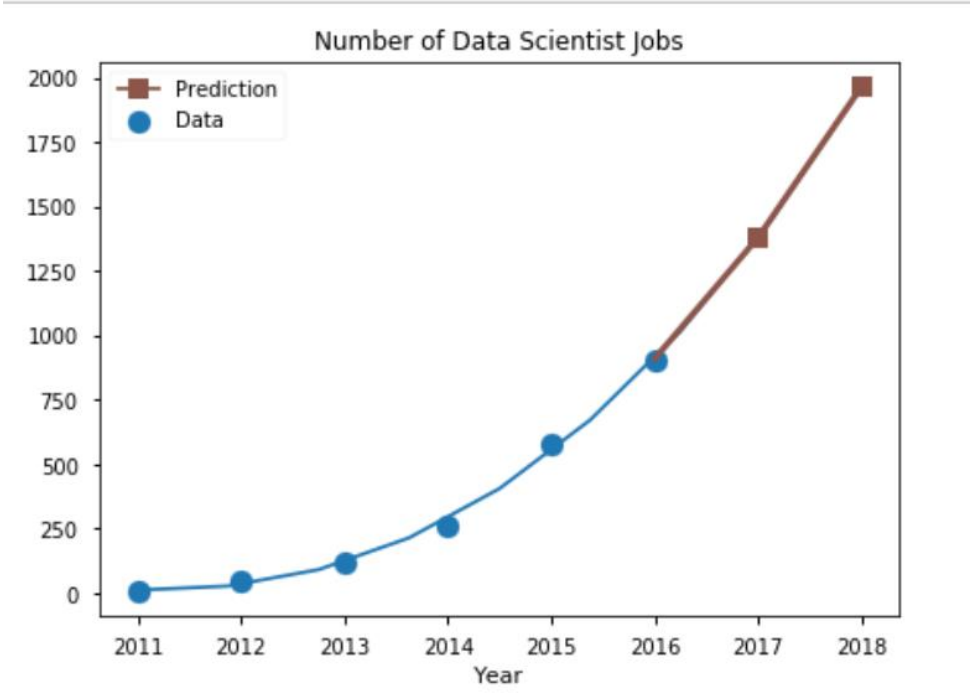
After performing the initial exploration, the data is divided into two groups of training and testing sets. If the model is built on full data, there can be a risk of over fitting that set of data, and the model might fail in the real world.

To reduce this over fitting, the data was divided into two sets of 60% training and 40% testing.

In [226]:
```
#The graph shows the increase in the data Science jobs from 2011 till 2016 and also prediction in 2017
#as the growing number of jobs are doubling every year hence in 2018 the prediction says that the data
#be in high demand hence people looking for growth and exciting opportunities should prepare them with
#required for DS
dsj = h1b[['JOB_TITLE','YEAR']][h1b['JOB_TITLE'] == "DATA SCIENTIST"].groupby('YEAR').count()['JOB_TITL
X = np.array(dsj.index)
Y = dsj.values
def func(x, a, b, c):
    return a*np.power(x-2011,b)+c

popt, pcov = curve_fit(func, X, Y)
X1 = np.linspace(2011,2018,9)
X2 = np.linspace(2016,2018,3)
X3 = np.linspace(2017,2018,2)
fig = plt.figure(figsize=(7,5))
plt.scatter(list(dsj.index), dsj.values, c='C0', marker='o', s=120, label='Data')
plt.plot(X1, func(X1,*popt), color='C0', label='')
plt.plot(X2, func(X2,*popt), color='C5', linewidth=3, marker='s', markersize=1, label='')
plt.plot(X3, func(X3,*popt), color='C5', marker='s', markersize=10, label='Prediction')
plt.legend()
plt.title('Number of Data Scientist Jobs')
plt.xlabel('Year')
plt.show()
```

The graph shows the increase in the data Science jobs from 2011 till 2016 and also prediction in 2017 and 2018 as the growing number of jobs are doubling every year hence in 2018 the prediction says that the data scientist jobs will be in high demand hence people looking for growth and exciting opportunities should prepare them with the skill sets
required for DS

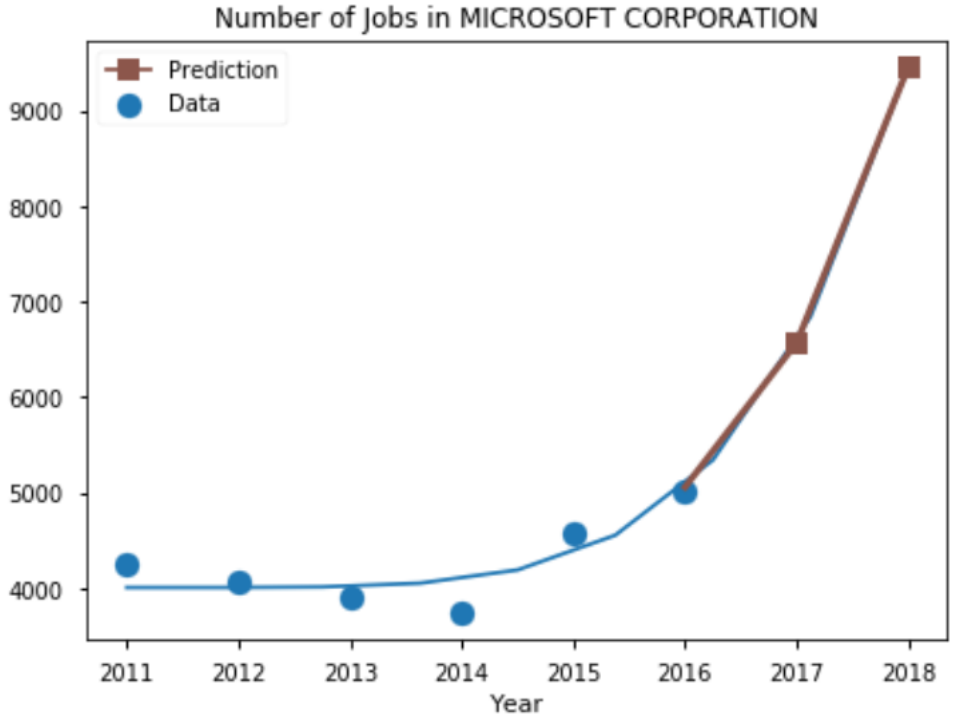


Data Scientist job is very popular and every company wants to hire such people with data science skill sets, so with some analysis and prediction on the number of Data Scientist jobs.
The regression used, where the feature is a power function of year. From figure below the number of Data scientist jobs is increasing rapidly (power of 2.32) and
expect to have about 1.5k and 2.1k Data Scientist jobs in 2017 and 2018 and this is from Data Scientist from other countries.

In [231]:
```
#The graph shows the increase in the jobs at Microsoft from 2011 till 2016 and also prediction in 2017
#as the growing number of jobs initially is not increasing that much as it is pretty straight in the gr
#be in high demand hence people looking for growth and exciting opportunities should prepare them with
#required for DS
dsj = h1b[['EMPLOYER_NAME','YEAR']][h1b['EMPLOYER_NAME'] == "MICROSOFT CORPORATION"].groupby('YEAR').co
X = np.array(dsj.index)
Y = dsj.values
def func(x, a, b, c):
    return a*np.power(x-2011,b)+c

popt, pcov = curve_fit(func, X, Y)
X1 = np.linspace(2011,2018,9)
X2 = np.linspace(2016,2018,3)
X3 = np.linspace(2017,2018,2)
fig = plt.figure(figsize=(7,5))
plt.scatter(list(dsj.index), dsj.values, c='C0', marker='o', s=120, label='Data')
plt.plot(X1, func(X1,*popt), color='C0', label='')
plt.plot(X2, func(X2,*popt), color='C5', linewidth=3, marker='s', markersize=1, label='')
plt.plot(X3, func(X3,*popt), color='C5', marker='s', markersize=10, label='Prediction')
plt.legend()
plt.title('Number of Jobs in MICROSOFT CORPORATION')
plt.xlabel('Year')
plt.show()
```

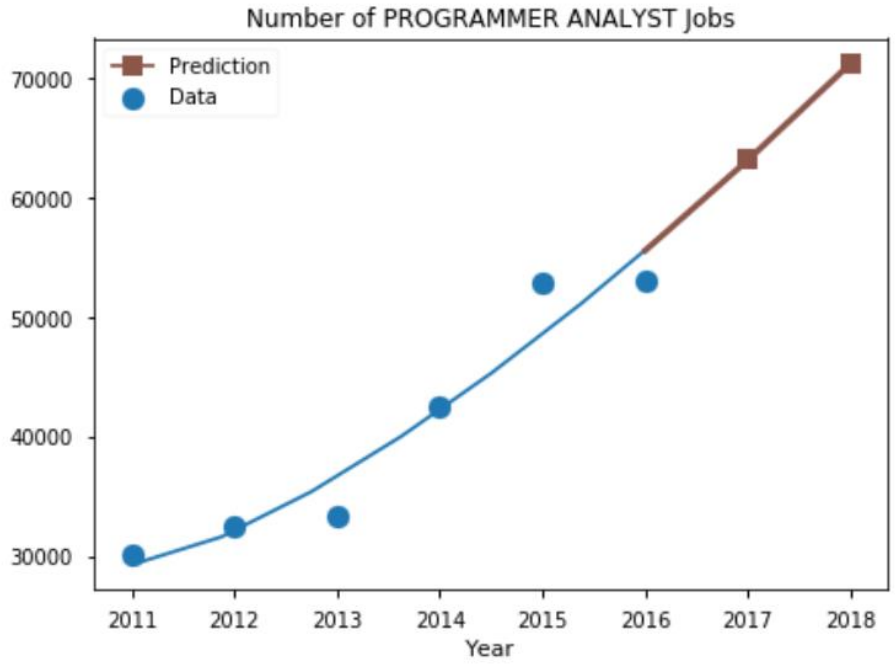
The graph shows the increase in the jobs at Microsoft from 2011 till 2016 and also prediction in 2017 and 2018 as the growing number of jobs initially is not increasing that much as it is pretty straight in the graph but are increasing rapidly in 2018 the prediction says that the data scientist jobs will be in high demand hence people looking for growth and exciting opportunities should prepare them with the skill sets required for DS



In [230]:
```
#The graph shows the increase in the Programmer Analyst jobs from 2011 till 2016 and also prediction in
#as the growing number of jobs are doubling every year hence in 2018 the prediction says that the progr
#be in high demand hence people looking for growth and exciting opportunities should prepare them with
#required for DS
dsj = h1b[['JOB_TITLE','YEAR']][h1b['JOB_TITLE'] == "PROGRAMMER ANALYST"].groupby('YEAR').count()['JOB_
X = np.array(dsj.index)
Y = dsj.values
def func(x, a, b, c):
    return a*np.power(x-2011,b)+c

popt, pcov = curve_fit(func, X, Y)
X1 = np.linspace(2011,2018,9)
X2 = np.linspace(2016,2018,3)
X3 = np.linspace(2017,2018,2)
fig = plt.figure(figsize=(7,5))
plt.scatter(list(dsj.index), dsj.values, c='C0', marker='o', s=120, label='Data')
plt.plot(X1, func(X1,*popt), color='C0', label='')
plt.plot(X2, func(X2,*popt), color='C5', linewidth=3, marker='s', markersize=1, label='')
plt.plot(X3, func(X3,*popt), color='C5', marker='s', markersize=10, label='Prediction')
plt.legend()
plt.title('Number of PROGRAMMER ANALYST Jobs')
plt.xlabel('Year')
plt.show()
```

The graph shows the increase in the Programmer Analyst jobs from 2011 till 2016 and also prediction in 2017 and 2018 as the growing number of jobs are doubling every year hence in 2018 the prediction says that the programmer jobs will be in high demand hence people looking for growth and exciting opportunities should prepare them with the skill sets required for Programmer



By building a model on 60% training data and testing on rest of 40% untouched data, the effectiveness of a model could be assessed.
In addition, the relationship between the variables needs to be tested. The relation between each independent variable and the response variable was checked such that the independent variables behaved in a similar way in a model as they behaved individually when run against the response variable.

K-Means Clustering

Unsupervised learning - K-Means Clustering to see where H1Bs are more prominent in terms of location and salary
Reading Latitude and Longitude data (taking out N/As)

In [64]:
```
#Unsupervised learning - K-Means Clustering to see where H1Bs are more  prominant in terms of location
#Reading Latitude and Longitude data (taking out N/As)
h1b
#Reading Latitude and Longitude data (taking out N/As)
H1bLatLong = h1b.loc[H1bState != 'NA']
H1bLatLong = H1bLatLong[['lon','lat']]
H1bLatLong = H1bLatLong.dropna();
H1bLong = H1bLatLong['lon'].values
H1bLat = H1bLatLong['lat'].values
```

In [65]:
```
#K-Means clustering to see where the applicants from hence Optimal K is 10
nClusters = 10
kmeans = KMeans(n_clusters=nClusters, random_state=0).fit(H1bLatLong.values)

kl = kmeans.labels_ #Getting the cluster for each case
H1bLatLong['Cluster'] = kl

H1bLatLong['State'] = H1bState
H1bLatLong = H1bLatLong.dropna()
H1bLatLong.sort_values('Cluster')
print(H1bLatLong[['Cluster','State']].drop_duplicates().sort_values('Cluster'))
#See which states a cluster cover
```

K-Means clustering to see where the applicants from hence Optimal K is 10

	Cluster	State
599040	0	NEW MEXICO
1	0	TEXAS
137994	0	ARIZONA
348	0	OKLAHOMA
349	0	ARKANSAS
865	0	KANSAS
112	0	LOUISIANA
16087	0	MISSOURI
9003	0	MISSISSIPPI
204	1	DELAWARE
109	1	NEW HAMPSHIRE
94	1	MASSACHUSETTS
70	1	PENNSYLVANIA
62	1	MARYLAND
44	1	NEW YORK
315	1	MAINE
300	1	CONNECTICUT
9	1	VIRGINIA
34	1	RHODE ISLAND
2	1	NEW JERSEY
82214	1	WEST VIRGINIA
30	1	DISTRICT OF COLUMBIA
15686	1	NORTH CAROLINA
1847	1	VERMONT

In [67]:
```
#Salary Data according to clusters
H1bSalary = H1bLatLong
H1bSalary['Salary'] = h1b['PREVAILING_WAGE']
H1bSalary['FT'] = h1b['FULL_TIME_POSITION']
H1bSalary['SOC'] = h1b['SOC_NAME']
```

In [68]:
```
salaryMin = np.zeros(nClusters)
salaryMax = np.zeros(nClusters)
salaryMean = np.zeros(nClusters)
salaryMedian = np.zeros(nClusters)
salaryStd = np.zeros(nClusters)

for k in range(0,nClusters):
    salaryClustered = H1bSalary.loc[H1bSalary['Cluster'] == k]
    salaryClustered = salaryClustered.loc[salaryClustered['FT'] == 'Y'] #We only consider full-time emp
    salaryClustered = salaryClustered.loc[salaryClustered['Salary'] <= 5000000] #We take out extreme ou
    salaryClustered = salaryClustered.loc[salaryClustered['Salary'] > 0] # We take out non-paid positio
    salaryClustered = salaryClustered.dropna()
    salaryMin[k] = np.min(salaryClustered['Salary'].values)
    salaryMax[k] = np.max(salaryClustered['Salary'].values)
    salaryMean[k] = np.mean(salaryClustered['Salary'].values)
    salaryMedian[k] = np.median(salaryClustered['Salary'].values)
    salaryStd[k] = np.std(salaryClustered['Salary'].values)
```

In [69]:
```
#Statistics
salaryStats = pd.DataFrame()
salaryStats['Mean'] = salaryMean.astype(int)
salaryStats['Median'] = salaryMedian.astype(int)
salaryStats['Std'] = salaryStd.astype(int)
salaryStats['Min'] = salaryMin.astype(int)
salaryStats['Max'] = salaryMax
salaryStats
```

Out[69]:

	Mean	Median	Std	Min	Max
0	68438	63981	34940	14000	4996888.0
1	72496	67700	35449	2012	4865100.0
2	90015	86611	31190	2000	4971600.0
3	65884	62234	30334	15080	4538560.0
4	64048	58157	53052	15080	4848792.0
5	71929	69160	39216	7799	4985760.0
6	69570	64230	38104	2011	4648800.0
7	80756	81307	27822	14851	2897400.0
8	65888	61942	32275	63	4729920.0
9	61037	55474	37268	16827	1040000.0

K-means clustering algorithm is to divide m points in N dimensions into K clusters, so that minimizing the mean square distance from each data point to its nearest center.
Thus, K-means clustering algorithm was utilized to separate the locations of H-1B applications, which is an ideal model to analyze the locations-related data.

Decision Tree

The Decision Trees were derived with techniques based on classification and regression trees (CART) and Chi-squared Automatic Induction, which are classifiers that predict class labels from data items. In our project, the Decision tress algorithm is utilized to predict whether the case of H-1B Visa application will be approved or not based on the previous data.

In [166]:
```
#-----------  data process  -----------
xSet1 = H1Info.loc[:,'WORKSITE']
WORKSITE_List = xSet1.drop_duplicates().values
X_input1 = []
for i in xSet1:
    X_input1.append(WORKSITE_List.tolist().index(i))

xSet2 = H1Info.loc[:,'EMPLOYER_NAME']
EMPLOYER_NAME_List = xSet2.drop_duplicates().values
X_input2 = []
for i in xSet2:
    X_input2.append(EMPLOYER_NAME_List.tolist().index(i))

xSet3 = H1Info.loc[:,'CASE_STATUS']
CASE_STATUS_List = xSet3.drop_duplicates().values
X_input3 = []
for i in xSet3:
    X_input3.append(CASE_STATUS_List.tolist().index(i))

#salary range
xSet4 = H1Info.loc[:,'PREVAILING_WAGE']
#define salary range
salaryRange = [20000.00,40000.00,60000.00,80000.00,100000.00,120000.00,140000.00,
               160000.00,180000.00,200000.00,400000.00,500000.00,1000000.00,2000000.00,float("inf")]
X_input4 = []
```

Creating a transpose matrix with the training data set and then randomly splitting the data into test and train data with 60% as training data set.

In [167]:
```
# -----------  decision Tree ALG  -----------
x_input = np.zeros([3,len(X_input4)])
x_input[0] = X_input1
x_input[1] = X_input2
x_input[2] = X_input4
x_matrix = x_input.transpose()
```

In [168]:
```
clf = tree.DecisionTreeClassifier()
clf = clf.fit(x_matrix, X_input3)
```

In [235]:
```
print("KfoldCrossVal mean score using Decision Tree is %s" %cross_val_score(clf,x_matrix,X_input3,cv=10
```

```
KfoldCrossVal mean score using Decision Tree is 0.7716521768929983
```

In [236]:
```
#dividing data to have a training and a testing set
X_train, X_test, y_train, y_test = train_test_split(x_matrix, X_input3, test_size= .4, random_state=0)

# Decision Tree cross validation


print("KfoldCrossVal mean score using Decision Tree is %s" %cross_val_score(clf,x_matrix,X_input3,cv=10
```

In [235]:
```
print("KfoldCrossVal mean score using Decision Tree is %s" %cross_val_score(clf,x_matrix,X_input3,cv=10
```

```
KfoldCrossVal mean score using Decision Tree is 0.7716521768929983
```

In [236]:
```
#dividing data to have a training and a testing set
X_train, X_test, y_train, y_test = train_test_split(x_matrix, X_input3, test_size= .4, random_state=0)

# Decision Tree cross validation


print("KfoldCrossVal mean score using Decision Tree is %s" %cross_val_score(clf,x_matrix,X_input3,cv=10



y_pred = sm.predict(X_test)
print("Accuracy score using Decision Tree is %s" %metrics.accuracy_score(y_test, y_pred))
```

```
KfoldCrossVal mean score using Decision Tree is 0.7717522857036031
Accuracy score using Decision Tree is 0.853952804703639
```

Random Forest Tree

In this analysis, the algorithm is utilized to predict whether the case of H-1B Visa application will be approved or not based on the previous data. This algorithm is implemented based on different attributes and could provide a prediction if some new attributes are given

In [241]:
```
# Random Forest classifier

RFm = RandomForestClassifier(random_state = 42,
                             criterion='gini',
                             n_estimators = 500)

#dividing data to have a training and a testing set
X_train, X_test, y_train, y_test = train_test_split(x_matrix, X_input3, test_size= .4, random_state=0)



# Random Forest metrics
sm = RFm.fit(X_train, y_train)

y_pred = sm.predict(X_test)
print("Accuracy score using Random Forest is %s" %metrics.accuracy_score(y_test, y_pred))
```

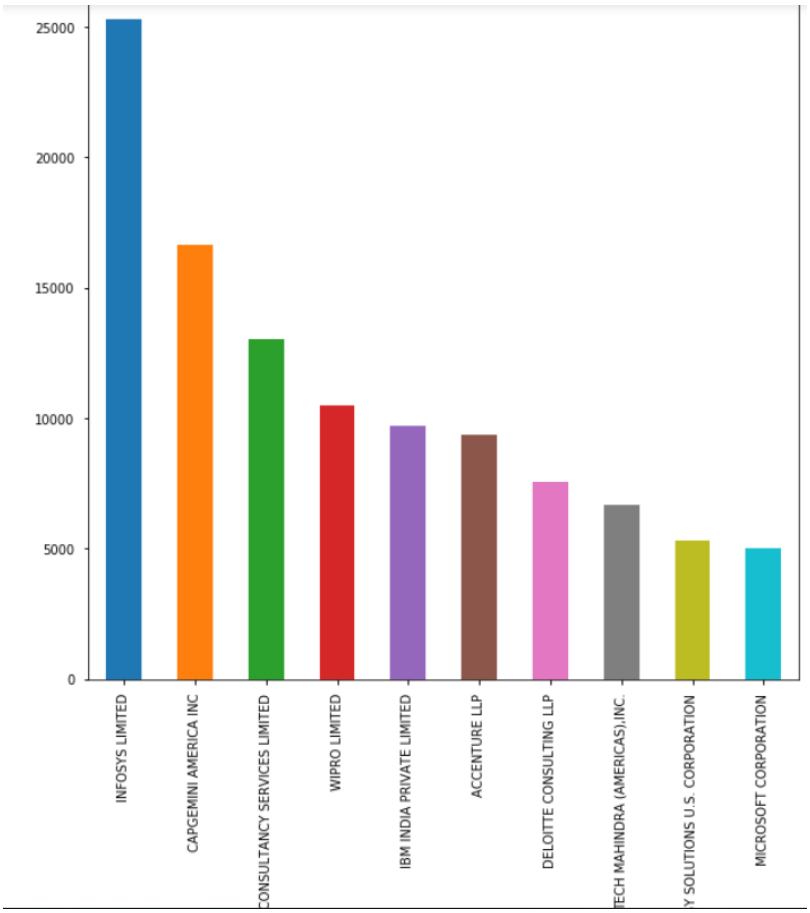
```
Accuracy score using Random Forest is 0.8724257111075798
```

Checking Performance Measures:

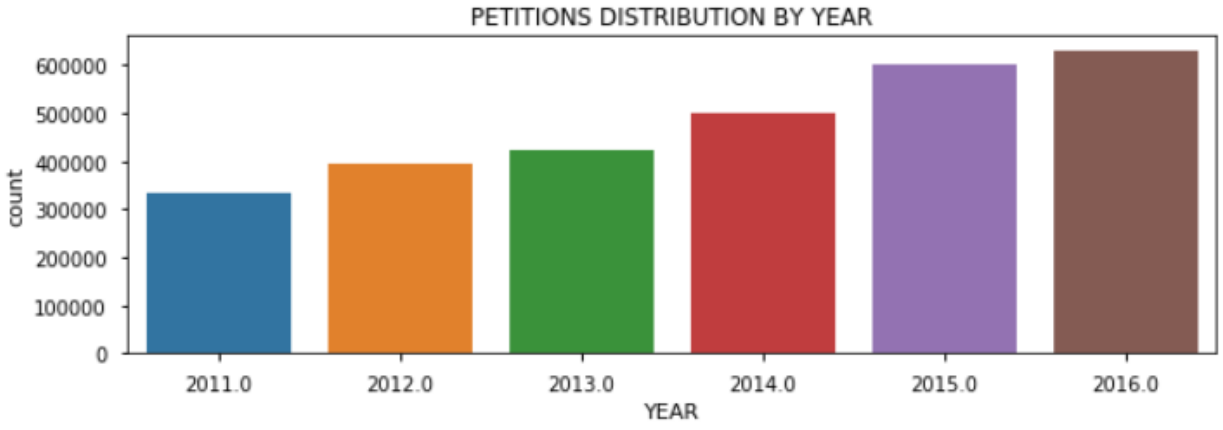
Various measures used to assess the performance of a model. Data split is done through 60% of training data and 40% of testing data. Different predictive learning techniques learn the data from the training set and build a model on it. It then implements the model on the testing set to check the performance of a model.

Results:

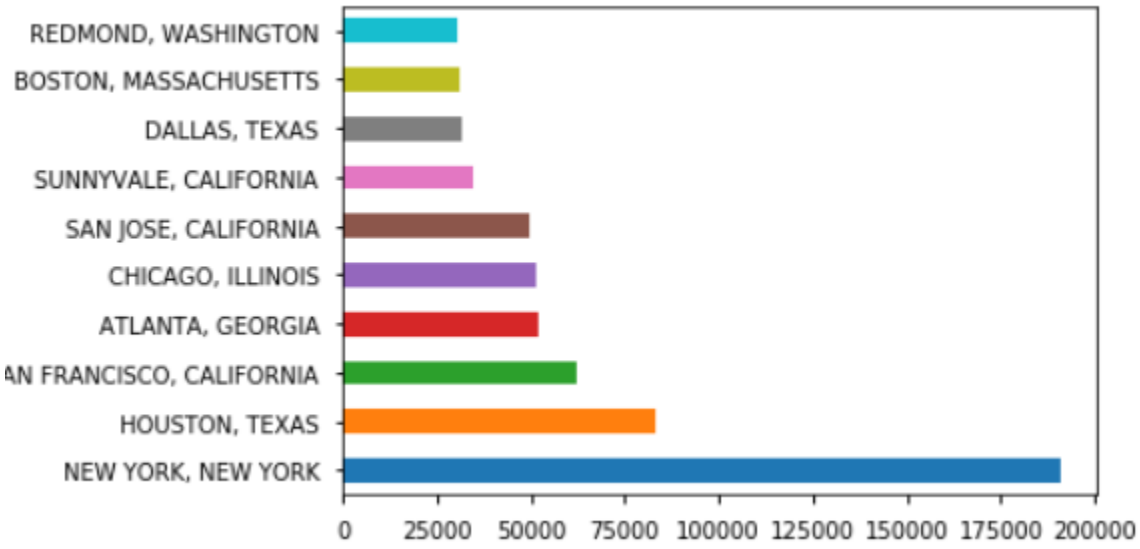
Top 10 Employers with highest H1B visa applications in 2016 - Infosys applied the highest visas in 2016



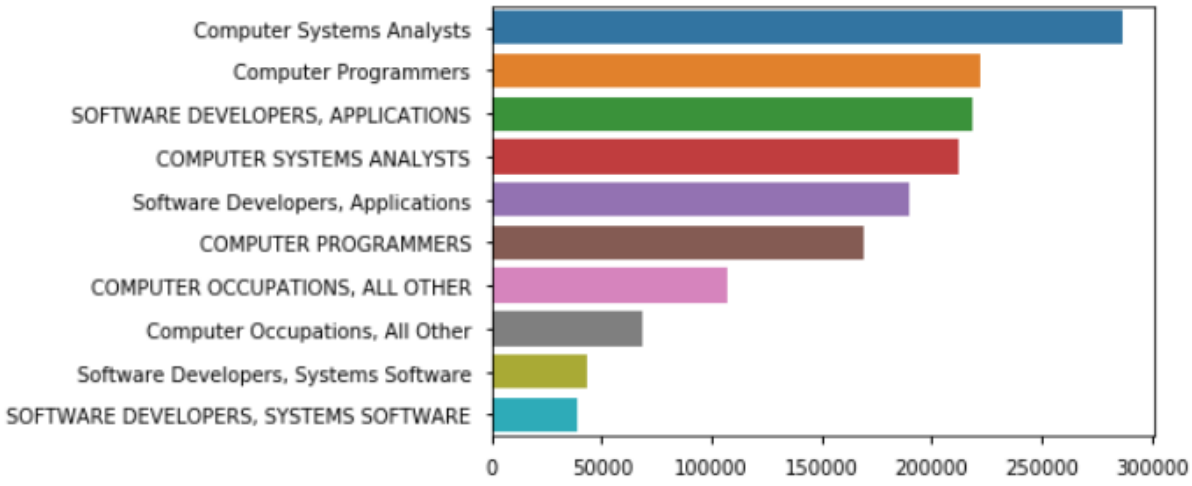
H1-B Visa petition applied by Years and with the graph its prominent that the visa petition is increasing every year

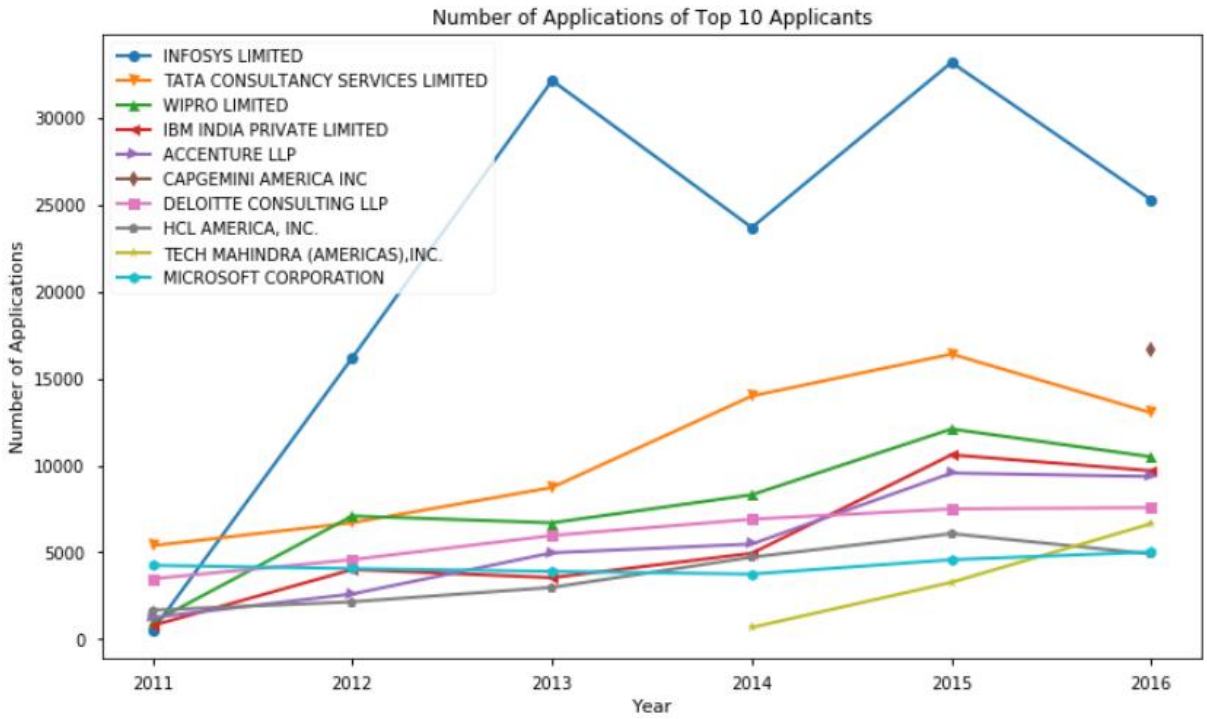


Top 10 cities as Worksites for H1-B Visa holders

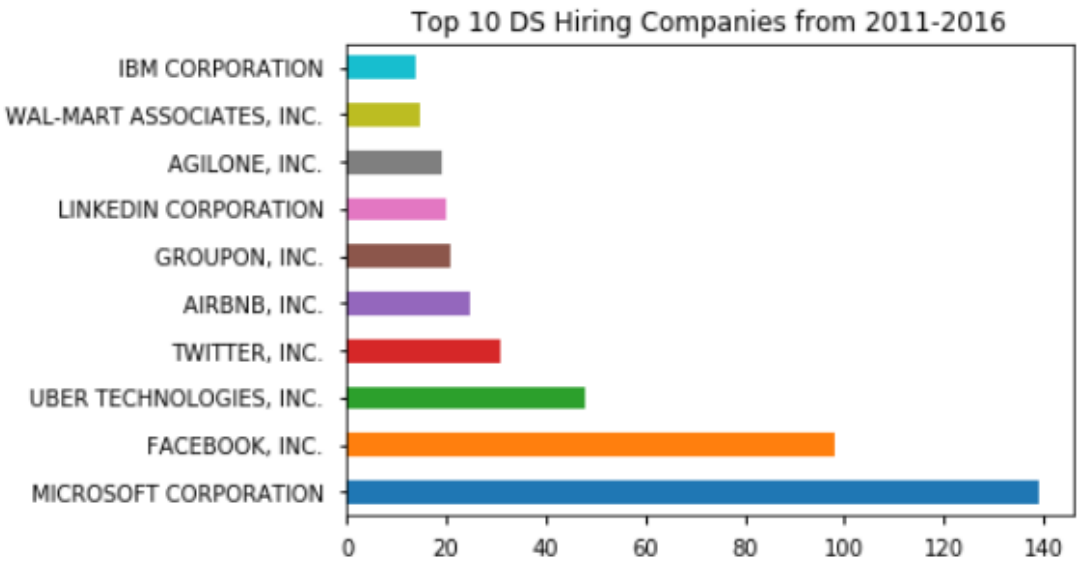


Computer systems Analysts has the highest number of visas





Companies who hire data scientist mostly and the location of such job opportunities
We can see that there are 2 new companies: CAPGEMINI and TECH MAHINDRA.
Analysis
INFOSYS shows a very rapid development, especially during period from 2011 to 2013, where it came from about zero to 32k applications!
TATA also shows a significant development.
Except for 2 new comers, we can see that all top EMPLOYERS have peaks of numbers of applications at the year of 2015 and a trend of decreasing to 2016. There could be a social or political event relating to this trend!
All the top applicants of visa applications are from India.

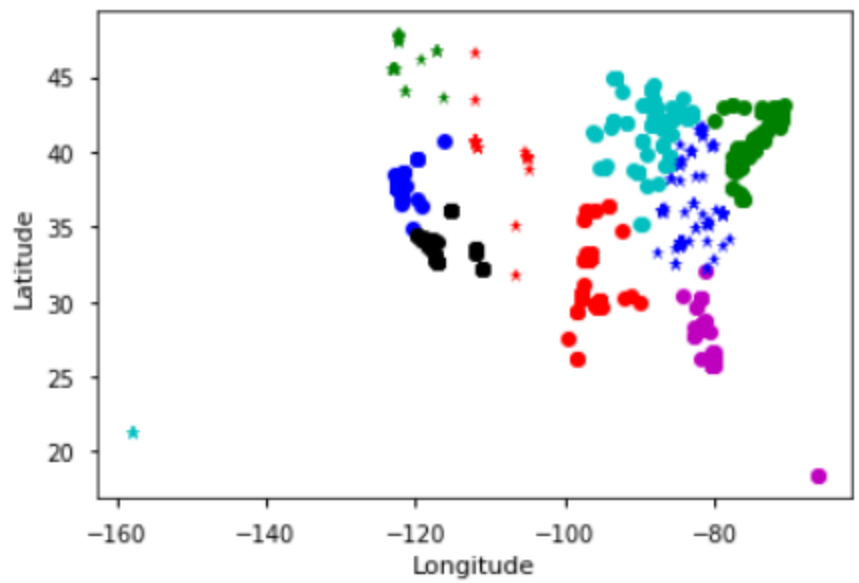


The analysis shows that Data Scientist hiring companies are IT and Tech mostly. Most imp highlight is that half of the applications for Data scientist from MICROSOFT CORPORATION in 2016.

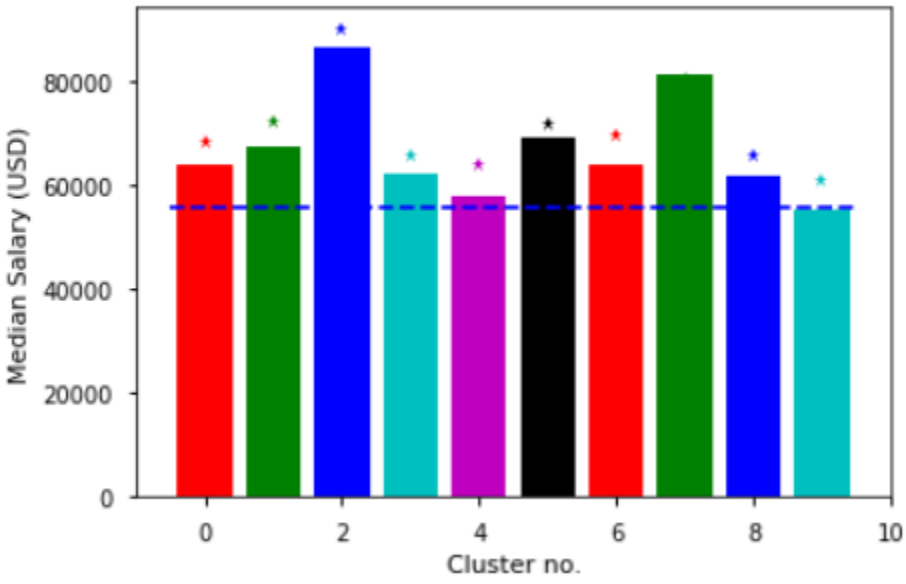
Also Walmart is in the top list for 2011 to 2016 timeframe.
Since all top IT and tech companies have offices in Silicon Valley, New York and Seattle area, top cities Data scientists work in are all in California, New York and Washington.

In this work, Forecasting, Clustering, Decision Tree and Random Forest Classifier were considered for determining the status of H1-B visa applications. Random Forest Classifier performed the best in terms of accuracy with 87%.
We inferred that the state of worksite, year of application, prevailing wages, employer name and

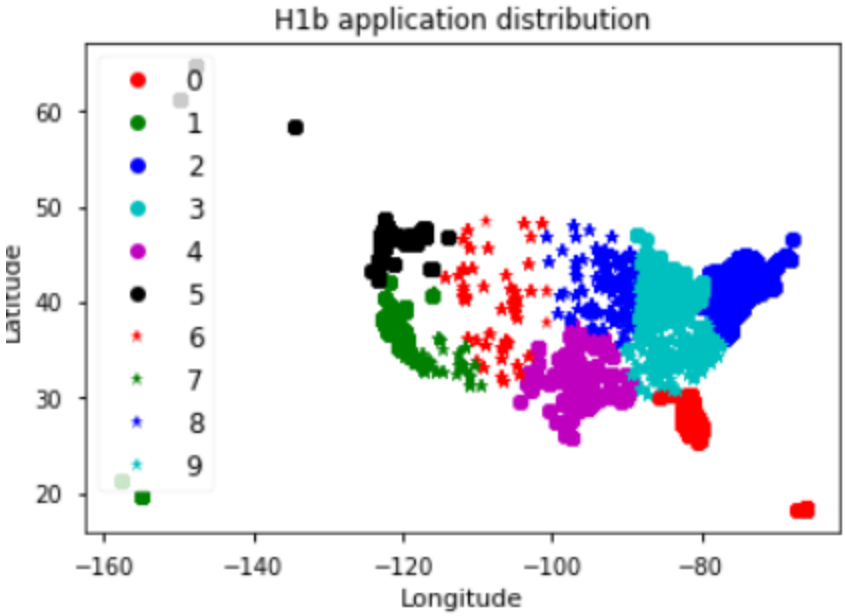
Unsupervised learning - K-Means Clustering to see where H1Bs are more prominant in terms of location and salary



Salary Data according to clusters



The same clustering algorithm is applied but is used for data year = 2016



The highest salary indicates for cluster 1 and the median salary for all the clusters in 2016 is approx. $82243

When applied the model

Decision Tree Accuracy is 85%

Random Forest is 87%

Will go ahead with Random Forest Classification model

soc-name play an important role in determining the case status of an H-1B application.

We observed that the most important feature to consider for our model is the acceptance ratio for the employer and the number of petitions filed by the employer. This clearly indicates the trends of H1- B visa filings which has a high correlation with the employer's acceptance rate.

Discussion:

- It is the technical jobs that are mostly filled by foreign workers. So emphasis should be driven on promoting coding in schools across U.S. such that more technical labor force is available for the jobs in the field. Only 40% of the schools in the United States teach computer programming.
- States like California, New York, Washington, New Jersey, Massachusetts, Illinois, and Texas are the prime location for foreign workers.
- Average Salary of Data Science related job in the current decade is higher than overall mean for all the jobs that lands to foreign workers.

Conclusion:

To conclude, in this project, I performed exploratory data analysis on the H-1B visa petition disclosure data for the period 2011-2016. I found that the employers with the most number of H-1B visa applications pay significantly lower wages compared to other employers for similar job positions offered by other employers. Also, I found that the Data Scientist position has experiences an exponential growth in terms of H-1B visa applications. Interestingly, the Data Scientist jobs are clustered in a few hotspots with San Francisco region having the highest number.

Top job titles from top 10 applications are mostly IT jobs. The most popular jobs in 2016 are "Technology Lead" and "Programmer Analyst" but the salaries are not top, they are in middle. The top offering salary is "Manager", which is reasonable.

References:

https://python-graph-gallery.com/
https://community.modeanalytics.com/python/tutorial/counting-and-plotting-in-python/
https://github.com/ryankarlos/H1-B-US-Visa-Petition-Data/blob/master/Code/Cytora_exercise_RyanNazareth.ipynb
https://www.analyticsvidhya.com/blog/category/machine-learning/
https://www.kaggle.com/gskhurana/h1b-visa-prediction/notebook
https://cseweb.ucsd.edu/classes/wi17/cse258-a/reports/a051.pdf