CSE 564 - Visualization

H1-B Visa Petitions 2011-2016

Visualizations based on data based on H1-B visas from 2011 to 2016

Hemant Pandey – 110828730
Rakshit Gautam - 111168264

Context

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 H1-B visa is reserved for some categories and is provided to others on a lottery system. Well, this is one of the most important part of economy of United States and hence a visual analytics is much needed to get all the facts and figures.

Problem Statement

In the recent years, there has been a lot of news and controversies regarding the change of immigration policies and the H1-B Visas rule, especially after the new government came into power. Visual analytics of the data about the applicants of H1-B visa, their income and region would surely help to gain deeper insights about the following information:

- Which Employers send most number of H-1B visa applications?
- What is the Percentage share out of the 85,000 visa cap for the Employers with most applications?

- Most common Job Titles and their Relative wages to Industry standard as well as absolute?

- Comparison of Number of applications and Wages for Data Scientist, Data Engineer and Software Engineers.

- Are the jobs concentrated in few specific regions?

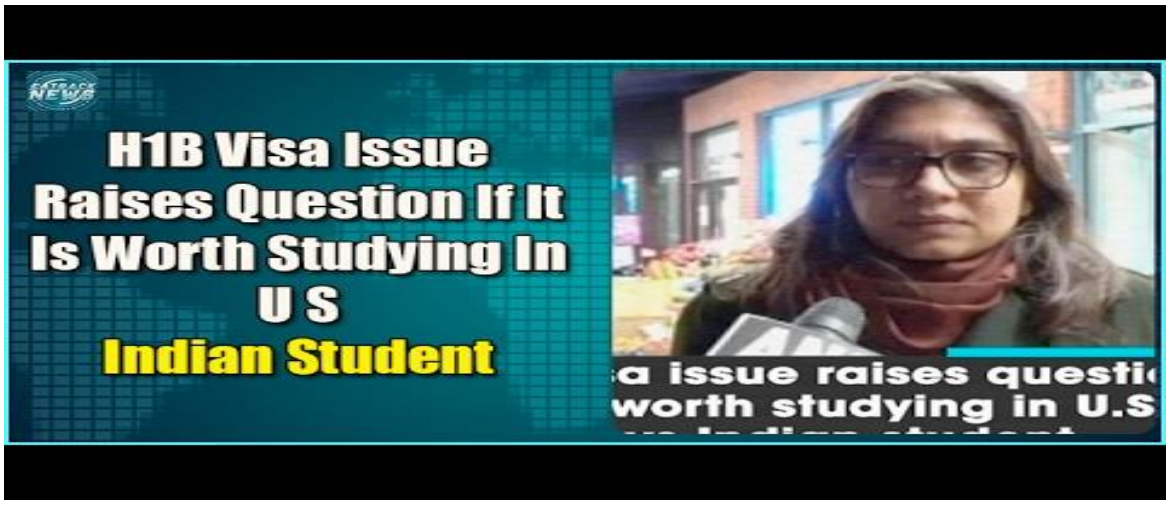
- What percentage of visas are certified, declined and withdrawn?

- What share of H1-B visas does Big 4 (Google, Facebook, Amazon and Microsoft) employers get?
- What percentage of the certified H1-B employees work in major cities like San Francisco, New York and Seattle.

- Which industry has maximum H1-B employees and who effects the US economy.

- What are your chances of getting a H1-B visa if you ae a software engineer working for Google Inc. in Mountain View, California.

After the republicans came to power, President Donald Trump has raised many anomalies regarding the H1-B rules and this has led to many international students thinking twice before planning for abroad studies.



Above are some important questions, the answers to which you would like to know to predict the chances based on the history, popular employers, profession and locations. Creating visualizations based on the data and extracting important information helpful to others is the ultimate goal.

Dataset

The Office of Foreign Labor Certification (OFLC) generates program data that is useful information about the immigration programs including the H1-B visa. However, the raw data available is messy and might not be suitable for rapid analysis. A set of data transformations were performed making the data more accessible for quick exploration. The dataset has been derived from the Kaggle website.

The columns in the dataset include:

1. **CASE_STATUS**: Status associated with the last significant event or decision. Valid values include “Certified,” “Certified-Withdrawn,” Denied,” and “Withdrawn”.
2. **EMPLOYER_NAME**: Name of employer submitting labor condition application.
3. **SOC_NAME**: Occupational name associated with the SOC_CODE. SOC_CODE is the occupational code associated with the job being requested for temporary labor condition, as classified by the Standard Occupational Classification (SOC) System.
4. **JOB_TITLE**: Title of the job
5. **FULL_TIME_POSITION**: Y = Full Time Position; N = Part Time Position
6. **PREVAILING_WAGE**: Prevailing Wage for the job being requested for temporary labor condition. The wage is listed at annual scale in USD. The prevailing wage for a job position is defined as the average wage paid to similarly employed workers in the requested occupation in the area of intended employment. The prevailing wage is based on the employer’s minimum requirements for the position.
7. **YEAR**: Year in which the H-1B visa petition was filed
8. **WORKSITE**: City and State information of the foreign worker's intended area of employment
9. **lon**: longitude of the Worksite
10. **lat**: latitude of the Worksite

Technologies

Python, D3 JavaScript, Pandas, HTML/CSS and Bootstrap.

Data Preprocessing

To gather more specifics, we will split the 'worksite' attribute into separate 'city' and 'state' columns to gather specific information about the major states employing and the major cities within a particular state. Similarly, we have given a thought to calculate haversine distance from the 'latitude' and 'longitude' attributes and find the proximity between different employers. Also, determining the industry from the 'Job_Title' is a feasible option and can help us to get the visualization based on specific industries and get information about how a specific industry contributes to US economy.

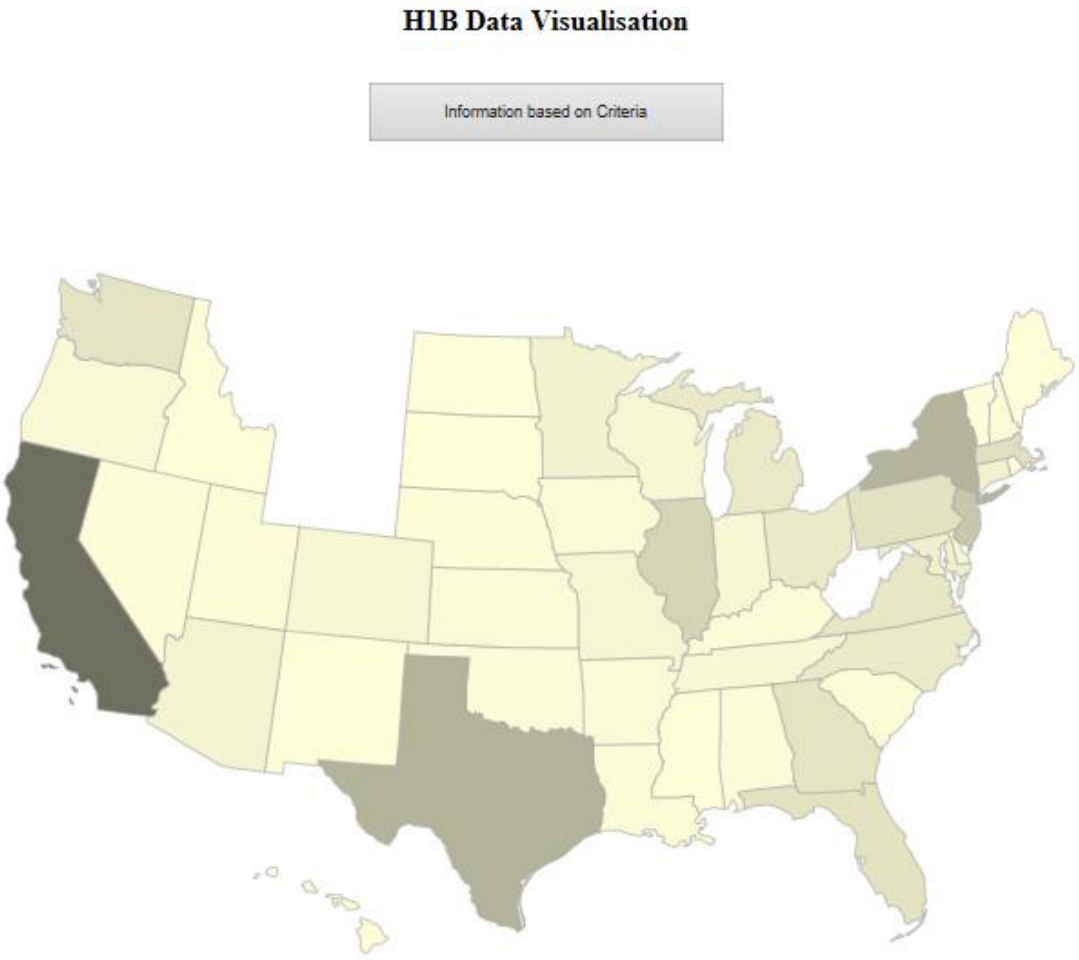
All the sampling and data processing is included in *data_processing.py*. The database has been transformed into capital letters and all the rows having NAN or incomplete values have been removed. Different csv files are generated using Pandas data frames to extract different information

Now the goal is to visualize the data by using information derived from the the operations on dataset using the python script.

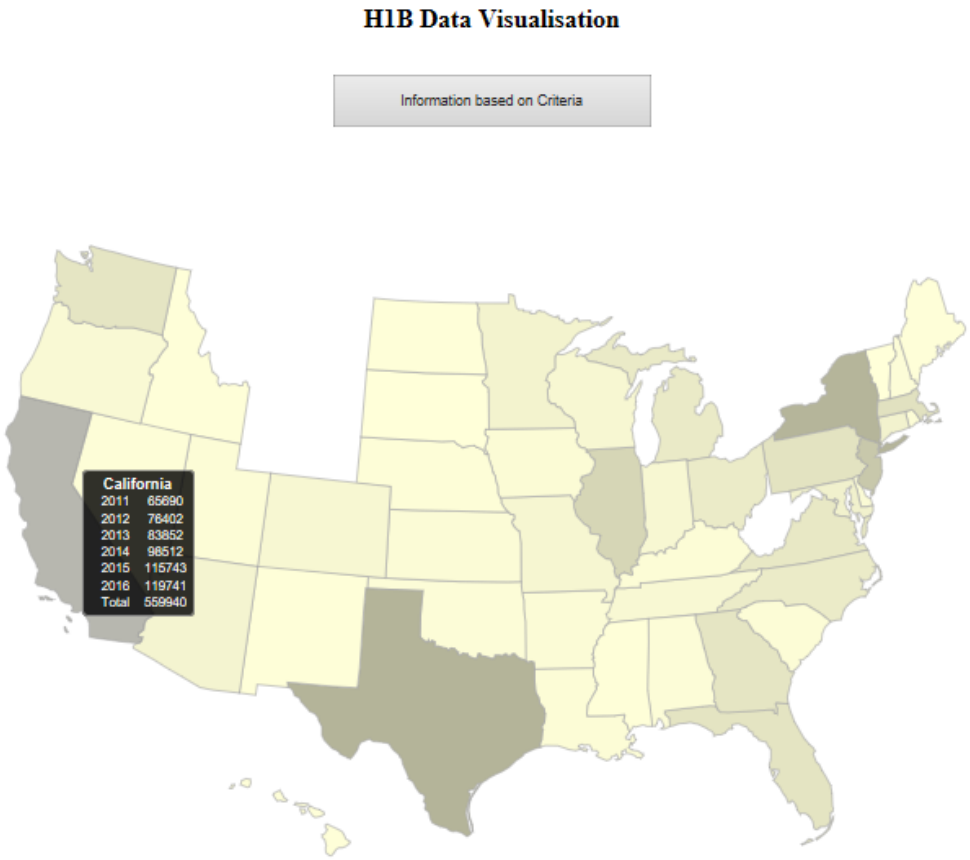
Visualization

The most basic category by which the information can be categorized and visualized in an effective manner is by categorizing on basis of WORKSITE, specifically the STATE. Various information like 'Which state in US has highest H1B vis holders?' or 'Which state in US has highest average wage for people on H1B Visa'?

We begin with a US map which is in form of a heat map in accordance with the number of applicants which applied for the H1B visa. California being the hub of techies can be clearly seen as the state favorite to H1B applicants. **The darker the color in the map, more is the number of visa applicants.** The top 3 states can be easily identified as **California, Texas** and **New York.**

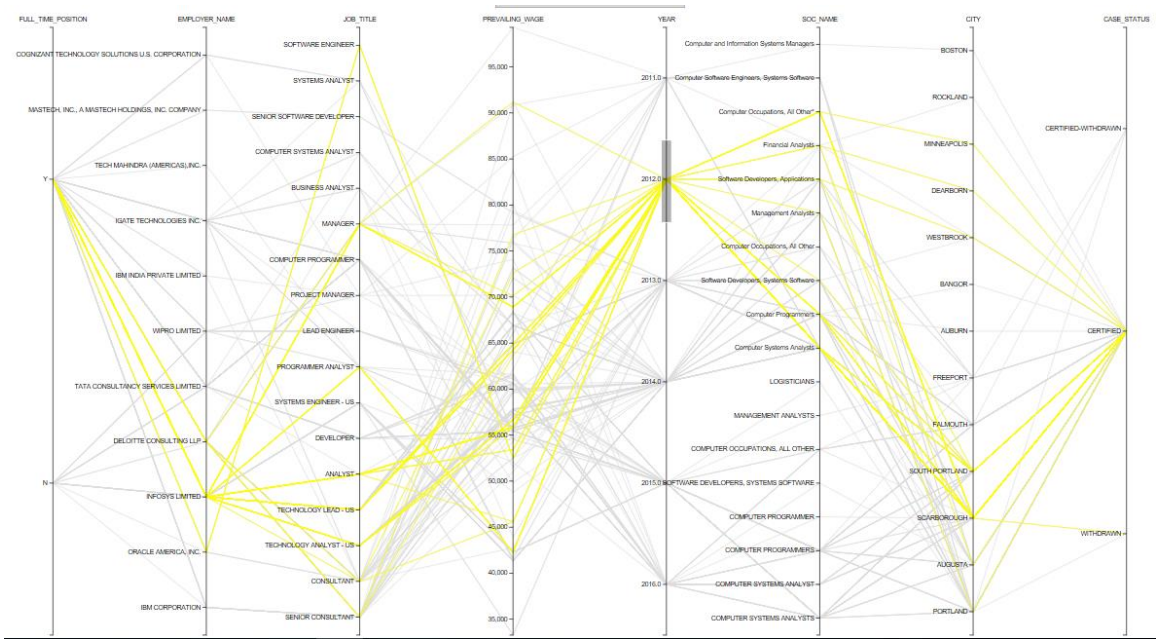


On hovering the desired state on the map, it gives the count of applications for the last five years along with the total count for the given state. The colors have been interpolated in such a way that the darker regions specifies the regions with more applicant counts and the lighter region specifies the areas with less number of applicants.



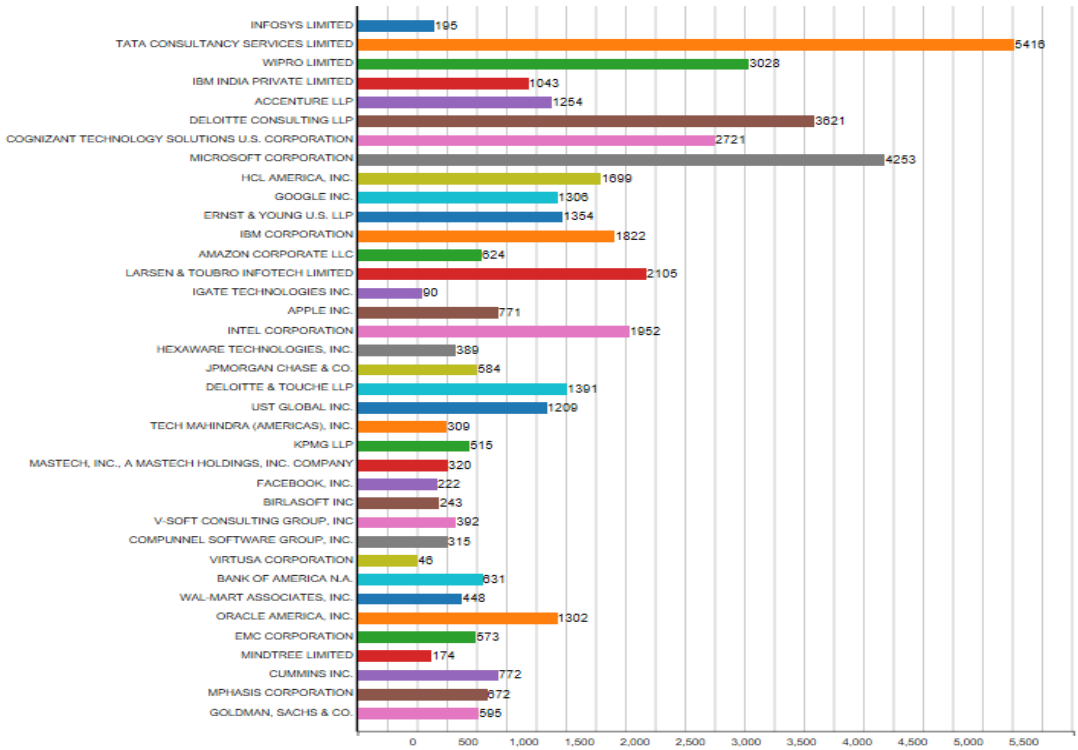
Now, the goal is to get all the details for the given state on clicking the state in the map. The best way to show the details is by **using parallel co-ordinates plot** which shows the interconnection of all the attributes of the selected state. The details of interconnection of the attributes of the selected state is highlighted i.e. all the employers, applicants, status, wage details, cities and all other details are highlighted. This might seem very chaotic since the number of employers are very large for some states. The visualization is made more efficient and **interactive by implementing brushing,** allowing the user to analyze the data points for a chosen range of attribute values. User can easily analyze data points in case of categorical data (for eg. See all the applications for Employee Amazon in California State)

The entire information for all the entities related to the selected state can be clearly represented in an interconnected manner using the parallel coordinates plot. The following plot shows the data points for state *Maine.* The visualization is shown for top 30 employers, cities and job titles. There is a csv generated for each state using the python script and the parallel coordinate plot is plotted using the data of csv for the respective state.

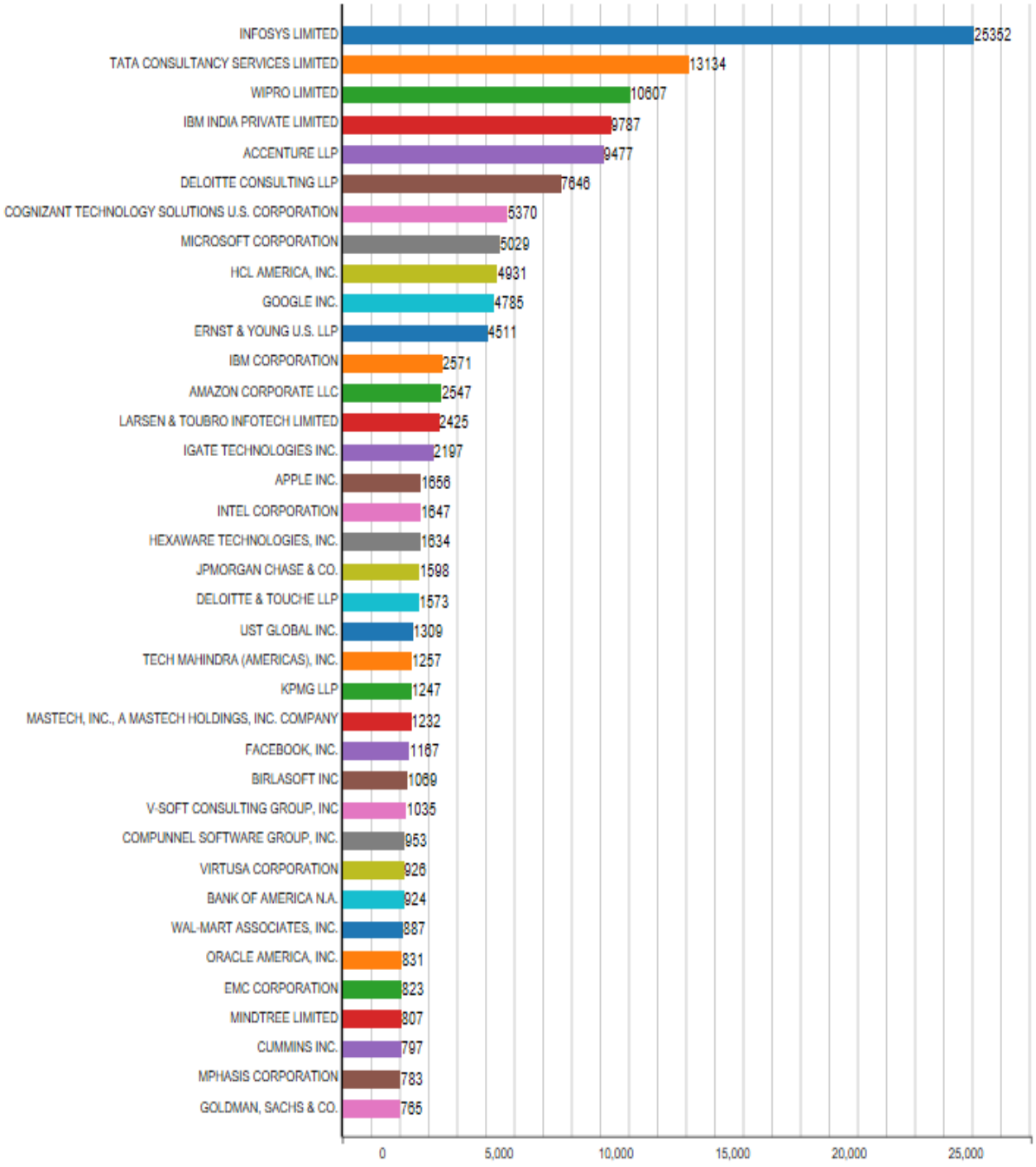


Some popular categories using which information can be extracted: **Employer**, **Job Title** and **State.** The year wise information can be visualized using bar charts which are invoked upon clicking the 'Information based on Criteria' button. Below are the part of screenshots:

Employer vs H1B_application count (Year 2011)

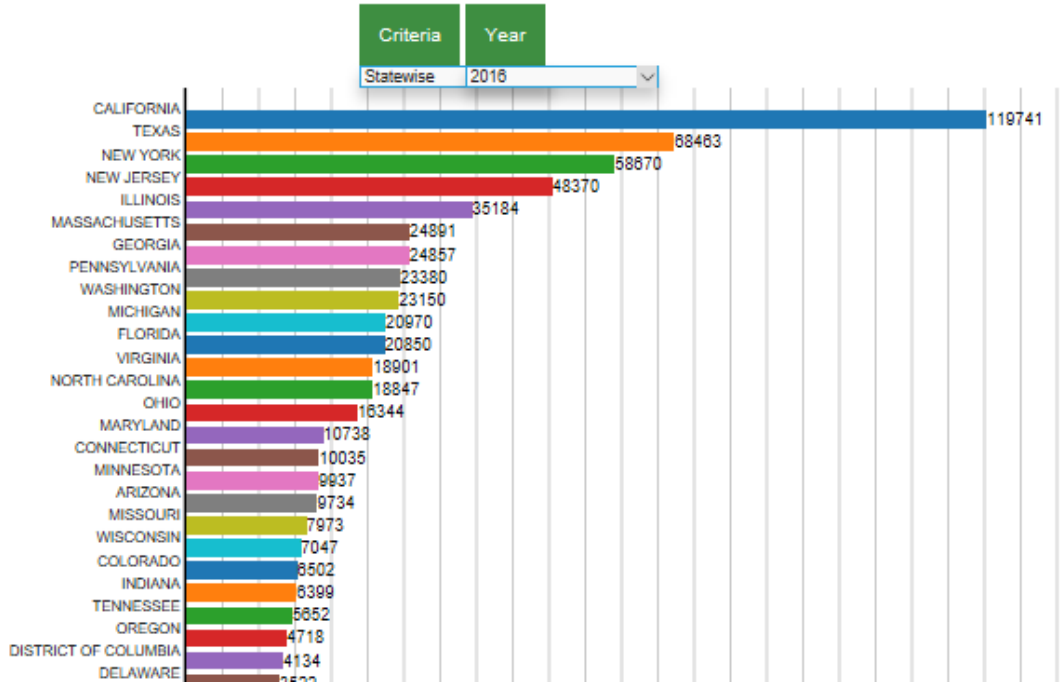


Employer vs H1B_application count (Year 2016)

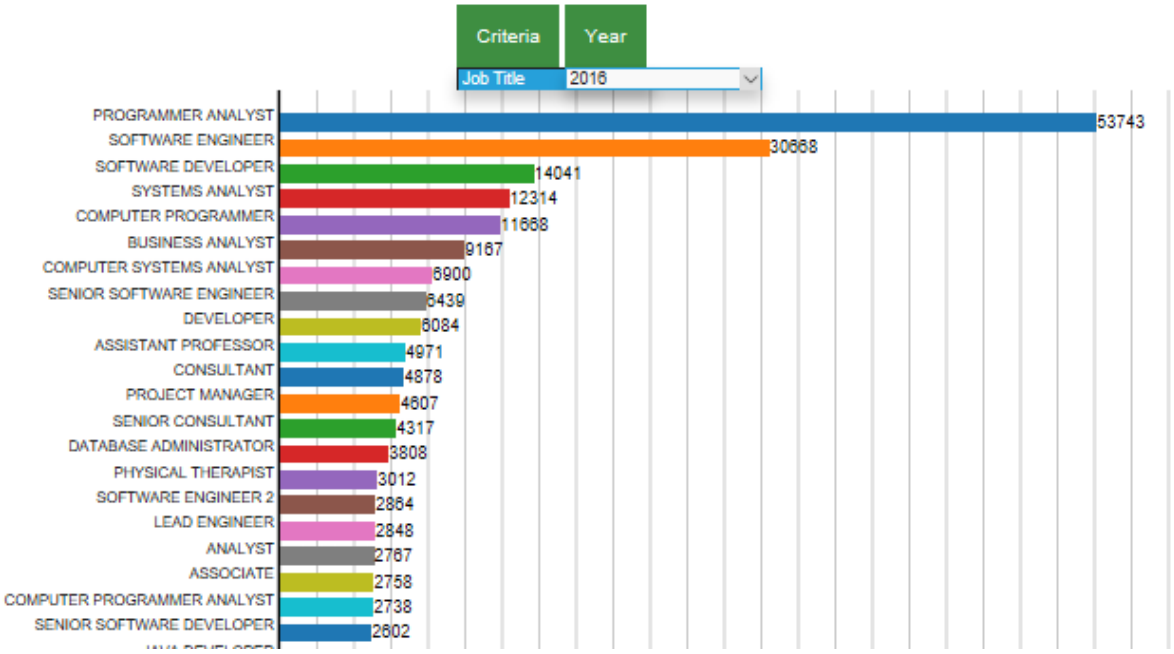


Notice, how Infosys has emerged from 2011 to 2016.

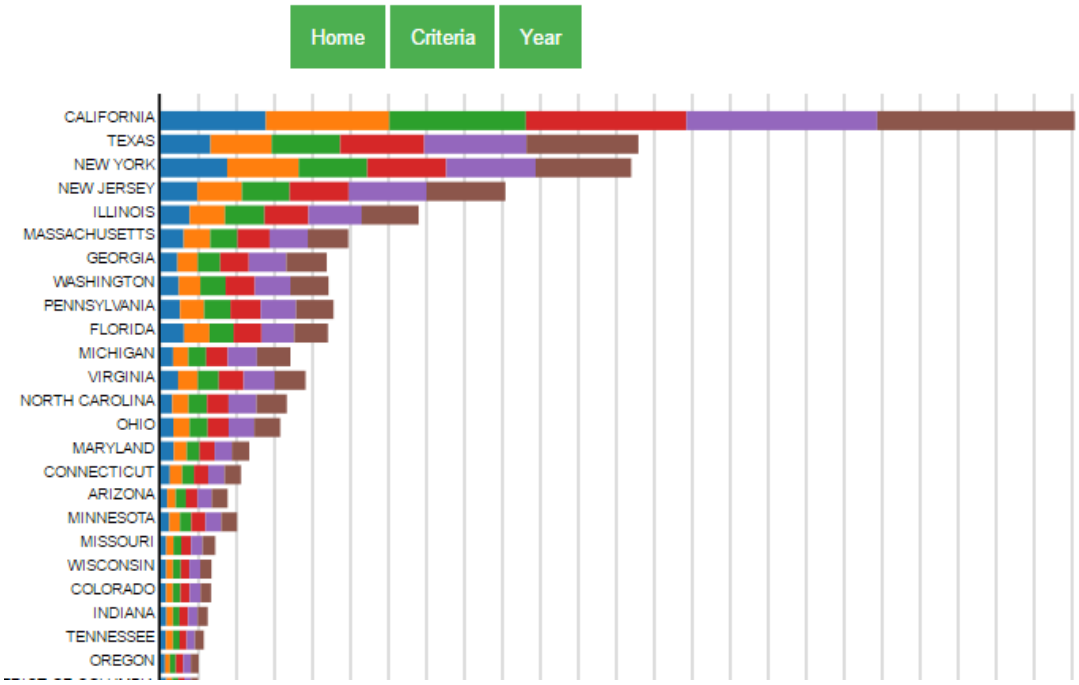
State vs H1B_application count (Year 2016)



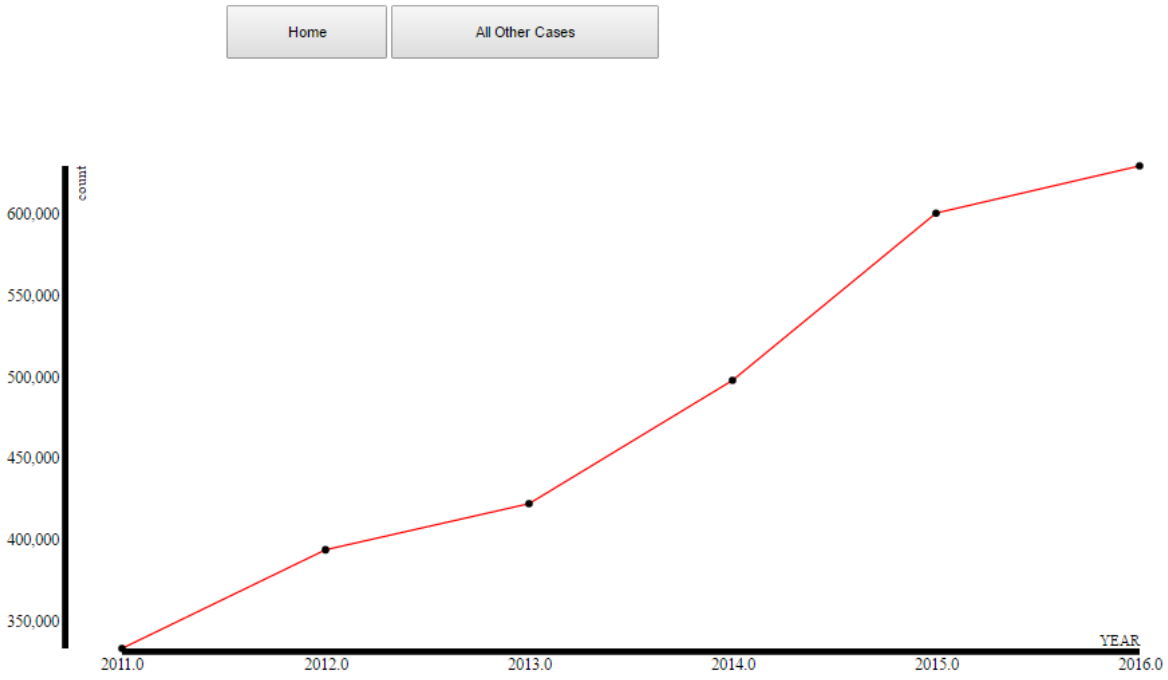
Job Title vs H1B_application count (Year 2016)



State vs H1B_application count (Last 5 years using different colors)

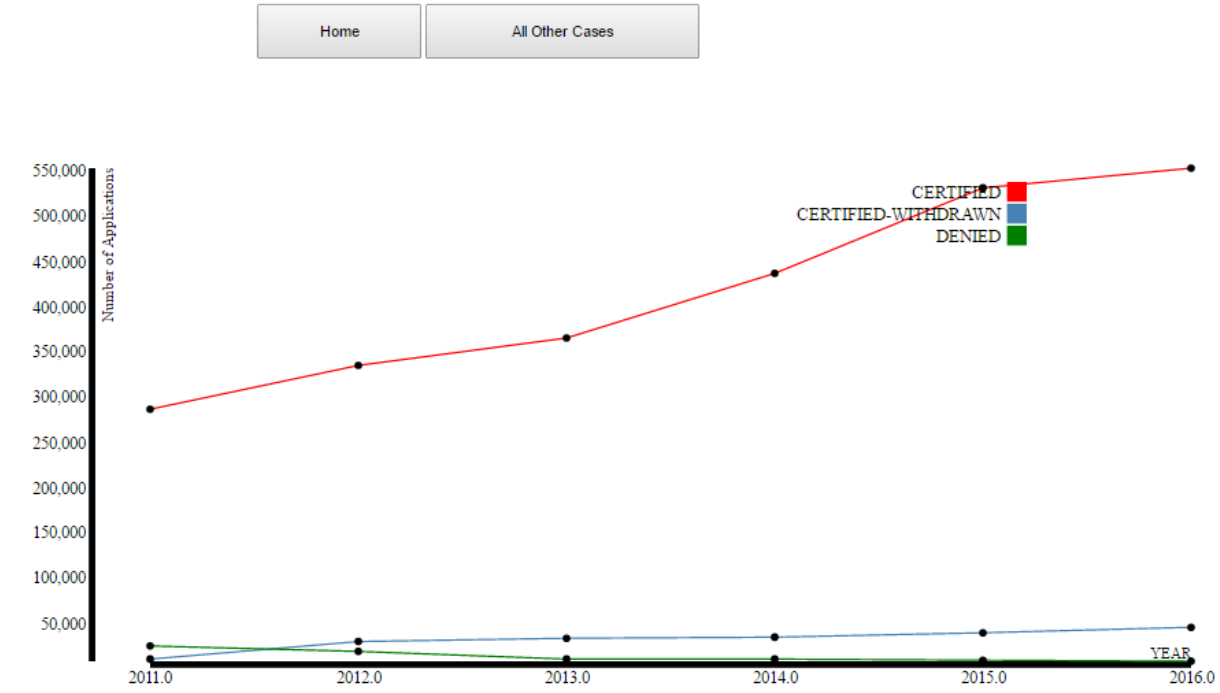


Information based on total number of applicants



See that the number of applicants have been rising every year. The plots above gave an evidence of the inferences provided in the report below.

Information based on total number of applicants (Certified, Denied and Withdrawn cases)



Code Snippets

```
function parallel(filename){

    filename = "data/" + filename;
    document.getElementById("bar_chart").innerHTML= '';

    var margin = {top: 30, right:30, bottom: 10, left: 30},
    width = 2000 - margin.left - margin.right,
    height = 1920 - margin.top - margin.bottom;

    var x = d3.scale.ordinal().rangePoints([0, width], 1),
        y = {},
        dragging = {};

    var line = d3.svg.line(),
        axis = d3.svg.axis().orient("left"),
        background,
        foreground;

    var svg = d3.select("#bar_chart").append("svg")
        .attr("width", width + margin.left + margin.right)
        .attr("height", height + margin.top + margin.bottom)
      .append("g")
        .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

    d3.csv(filename, function(error, cars) {

      // Extract the list of dimensions and create a scale for each.
      // x.domain(dimensions = d3.keys(
      //     cars[0]).filter(function(d) {
        //     return d != "" && (y[d] = d3.scale.linear()
        //         .domain(d3.extent(cars, function(p) { return +p[d]; }))
        //         .range([height, 0]));
        //   })
      // );

      x.domain(dimensions = d3.keys(cars[0]).filter(function(d) {
        return ((d == 'PREVAILING_WAGE' || d =='YEAR') && (y[d] = d3.scale.linear()
            .domain(d3.extent(cars, function(p) { return +p[d]; }))
            .range([height, 0]))) || (d!="" && (y[d] = d3.scale.ordinal()
            .domain(d3.map(cars, function(p){return p[d];}).keys())
            .rangePoints([height, 0],1)));
```

```
    <fieldset>
    <legend><b><font color="Blue"> Random Person (Visualisation Project 2)</font></b></legend>
    <center>
    <br>
    <br>

<div class="dropdown">
  <button class="dropbtn">Criteria</button>
  <div class="dropdown-content">
  <select onchange="chooseVariable1(this.value)" class="dropdown-content"  autocomplete="off">
      <option value="appcount_employer_name_yearwise.csv,EMPLOYER_NAME">Employer</option>
      <option value="appcount_state_yearwise.csv,STATE">Statewise</option>
      <option value="appcount_job_title_yearwise.csv,JOB_TITLE">Job Title</option>
</select>
  </div>
</div>
<div class="dropdown">
  <button class="dropbtn">Year</button>
  <div class="dropdown-content">
  <select onchange="chooseVariable2(this.value)" class="dropdown-content"  autocomplete="off">
      <option value="2011">2011</option>
      <option value="2012">2012</option>
      <option value="2013">2013</option>
       <option value="2014">2014</option>
      <option value="2015">2015</option>
      <option value="2016">2016</option>
</select>
  </div>
</div>
```

```
def createYearwiseCount(key, data_frame_original):
    df = pd.DataFrame({'count' : data_frame_original.groupby( [ key, "YEAR"] ).size()}).reset_index()
    df = df.sort_values(['YEAR','count',key], ascending=[False,False,True])
    temp = df.head(50)
    temp['2016'] = temp['count']

    x = df[key].isin(temp[key].tolist())
    y = (df['YEAR']==2015)
    m1 = df.loc[x & y]
    m1['2015'] = m1['count']
    m1 = m1[[key,'2015']]
    # print m1.head()

    y = (df['YEAR']==2014)
    m2 = df.loc[x & y]
    m2['2014'] = m2['count']
    m2 = m2[[key,'2014']]
    # print m2.head()

    y = (df['YEAR']==2013)
    m3 = df.loc[x & y]
    m3['2013'] = m3['count']
    m3 = m3[[key,'2013']]
    # print m3.head()

    y = (df['YEAR']==2012)
    m4 = df.loc[x & y]
    m4['2012'] = m4['count']
    m4 = m4[[key,'2012']]
    # print m4.head()

    y = (df['YEAR']==2011)
    m5 = df.loc[x & y]
    m5['2011'] = m5['count']
    m5 = m5[[key,'2011']]
    # print m5.head()

    temp = pd.merge(temp, m1, on=key)
    temp = pd.merge(temp, m2, on=key)
```

```
{id:"NM",n:"New Mexico",d:"M288.15255,424.01315L287.37714,419.26505L296.02092,419.79045L326.19268,422.73635L353.46084,424.42624L355.67611,
{id:"KS",n:"Kansas",d:"M507.88059,324.38028L495.26233,324.58471L449.17324,324.12748L404.61576,322.06985L379.98602,320.81244L383.87981,256.
{id:"NE",n:"Nebraska",d:"M486.09787,240.70058L489.32848,247.72049L489.19985,250.02301L492.65907,255.51689L495.37836,258.66923L490.32888,25
{id:"SD",n:"South Dakota",d:"M476.44687,204.02465L476.39942,203.44378L473.50371,198.59834L475.36394,193.88623L476.85667,187.99969L474.0748
{id:"ND",n:"North Dakota",d:"M475.30528,128.91846L474.69037,120.48479L473.01342,113.66887L471.12193,100.64465L470.66469,89.657624L468.9252
{id:"WY",n:"Wyoming",d:"M360.37668,143.27587L253.63408,129.81881L239.5506,218.27684L352.81521,231.86233L360.37668,143.27587Z"},
{id:"MT",n:"Montana",d:"M369.20952,56.969133L338.5352,54.1613L309.27465,50.60477L280.01411,46.563258L247.68201,41.228463L229.25272,37.8335
{id:"CO",n:"Colarado",d:"M380.03242,320.96457L384.93566,234.63961L271.5471,221.99565L259.33328,309.93481L380.03242,320.96457Z"},
{id:"ID",n:"Idaho",d:"M148.47881,176.48395L157.24968,141.26323L158.62142,137.03371L161.13626,131.08953L159.87884,128.8033L157.36398,128.91
{id:"UT",n:"Utah",d:"M259.49836,310.10509L175.74933,298.23284L196.33694,185.69149L243.11725,194.43663L241.63245,205.06705L239.32083,218.23
{id:"AZ",n:"Arizona",d:"M144.9112,382.62909L142.28419,384.78742L141.96087,386.24237L142.44585,387.21233L161.36012,397.88192L173.48466,405.
{id:"NV",n:"Nevada",d:"M196.39273,185.57552L172.75382,314.39827L170.92158,314.74742L169.34882,317.1536L166.97588,317.16429L165.50393,314.4
{id:"OR",n:"Oregon",d:"M148.72184,175.53153L157.57154,140.73002L158.62233,136.5005L160.9767,130.87727L160.36119,129.71439L157.84633,129.66
{id:"WA",n:"Washington",d:"M102.07324,7.6117734L106.43807,9.0667177L116.1377,11.814946L124.7057,13.754871L144.7516,19.412988L167.70739,25.
{id:"CA",n:"California",d:"M144.69443,382.19813L148.63451,381.70951L150.12055,379.69807L150.66509,376.75698L147.11357,376.16686L146.5994,3
];
var uStates={};

uStates.draw = function(id, data, toolTip){
    function mouseOver(d){
        d3.select("#tooltip").transition().duration(200).style("opacity", .9);

        d3.select("#tooltip").html(toolTip(d.n, data[d.id]))
            .style("left", (d3.event.pageX) + "px")
            .style("top", (d3.event.pageY - 28) + "px");
    }

    function mouseOut(){
        d3.select("#tooltip").transition().duration(500).style("opacity", 0);
    }

    d3.select(id).selectAll(".state")
        .data(uStatePaths).enter().append("path").attr("class","state").attr("d",function(d){ return d.d;})
        .style("fill" function(d){ return data[d.id].color; })
```

```
    <script src = "src/bar_chart.js"></script>
    <script src = "src/parallel.js"></script>
    <script src = "src/scree_plot.js"></script>
    <script src = "src/scree_plot_multi.js"></script>

  </head>
  <body onload="plot_scree('year_applicants.csv')">

    <style>
    .button1 {
    background-color: white;
    color: black;
    border: 2px solid #4CAF50; /* Green */
}
    .dropbtn {
    background-color: #4CAF50;
    color: white;
    padding: 16px;
    font-size: 16px;
    border: none;
    cursor: pointer;
}
        </style>



  <fieldset>
  <legend><b><font color="Blue"></font></b></legend>
  <center>
```

You can follow the README for steps on how to run.

Inferences

1. Infosys Limited was much behind Tata Consultancy Services in 2011 but surpassed it year by year and now the applicant count of Infosys is almost twice of TCS and it is currently the highest H1B provider company. **The monopoly of these Indian tech giants** is one of the reason that the number of Indians in United States are increasing every year.

2. From 2011-2016, the states have varied in the H1B application counts but the relative position haven't changed much. **California**, **Texas** and **New York** are the top 3 states with maximum H1-B application count. The reason being the mentioned states are the hub of IT and financial forms which employs the most.

3. **Jobs related to the computers** have been the most popular choices for the H1B applicants, the top ones being the programmer analyst, software engineer, business analyst and others**. In past few years, there is also a hike in the number of H1 Visas issued to assistant professors in the US universities.**

4. The **number of H1B applicants has increased drastically** from 2011 to 2016 especially from the companies belonging to India like Infosys, TCS, Wipro etc.

5. In the last five years, the number of certified visas have increased drastically and the percentage of visas which are denied have been decreasing every year. **This rapid increase every year is one main reason because of which strict immigration laws especially for H1B visas are being enforced.** This can be easily seen using the line charts.

6. We have also noted many interesting observations for different states**.** On brushing the wage attribute, we see that **most of the high paying applicants were denied visas for the state of Texas and also the wage for H1B employees in Texas is much less than that of employees in California and New York.** The reason could be that Texas is a tax-free states and the employees tend to give less income to employers since their expenses may not be the same as in California and New York.

7. By looking at the bar chart, on comparing the H1B applicant count **for the Big 4 companies (Google, Microsoft, Amazon and Facebook), Microsoft employs the maximum and Facebook employs minimum H1B employees.** However, all the 4 companies have increased their H1B employees count up to 4 times from 2011-2016. The reason could be the increasing number of people engaged in CS/IT.

8. **The central US is not much of an eye catcher region when it comes to H1B employees.** Apart from Texas, all the states receive very few applications. The reason being not much of industries are there in the central US.

9. We also tried to perform some **hypothesis** testing using our dataset. If a person is applying for H1B in San Francisco, California with a wage of less than 1,20,000 $, then we can say with good confidence that his/her visa will be approved.

10. Using the parallel plot for different states, we can also see which cities are the most active ones and attractive to the H1B employees. In California, San Francisco and Palo Alto are two most popular cities, the reason being their location in bay area which is the heart of Silicon Valley. For Washington, Seattle and Bellevue steals the show. For Texas, Dallas, Houston and Austin are the popular ones.

11. Since our dataset was much larger and incorporated the data for last five years, to gain insightful information from such a large data, parallel plot was the most effective visualization plot. The inferences we derived are very interesting and also have a strong reasoning with them.

References

1.) Dataset (obtained from Kaggle public datasets)
https://www.kaggle.com/nsharan/h-1b-visa

2.) H1-B visa wiki
https://en.wikipedia.org/wiki/H-1B_visa

3.) H1-B application process
http://www.immi-usa.com/h1b- application-process- step-by- step-guide/

4.) Parallel Plot
https://bl.ocks.org/mbostock/1341021