Table of Contents

Part 1 - What we could know about the Data Scientists?

1.0.A: List of columns which has null values and count of non-null values of the same columns

Out of 17 columns, 8 columns have null values. Below is the lift of those columns with the count of non-null (filled) values in those columns.

```
# List of columns which has NAs (Null values)
NA_list = df_demog.columns[df_demog.isnull().any()].tolist() ##Stackoverflow
NA_list
```

```
['TitleFit',
 'CurrentEmployerType',
 'MLToolNextYearSelect',
 'MLMethodNextYearSelect',
 'LanguageRecommendationSelect',
 'MajorSelect',
 'FirstTrainingSelect',
 'JobSatisfaction']
```

```
# Number of records with no NULL values for the columns with Null values
df_demog[NA_list].count() ##Stackoverflcounow
```

```
TitleFit                        4251
CurrentEmployerType             4275
MLToolNextYearSelect            4206
MLMethodNextYearSelect          4170
LanguageRecommendationSelect    4228
MajorSelect                     3952
FirstTrainingSelect             4324
JobSatisfaction                 4317
dtype: int64
```

1.0.B: Number of Data Scientists in the survey

Out of 4327 respondents, there are 1263 data scientists in the survey.

```
# Creating a dataframe with users whose current job title is Data Scientist
df_demog_ds = df_demog[df_demog.CurrentJobTitleSelect=='Data Scientist']## From Stackoverflow
print("Number of data scientists are",format(len(df_demog_ds)))
```

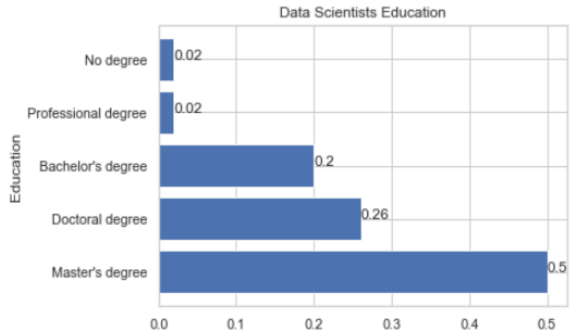
```
Number of data scientists are 1263
```

1.1: Type of formal education of Data Scientists

Overall 76% of the data scientists have doctoral and master's degree. Fifty percent of data scientists' respondents have master's degree. Thus, we can conclude that data scientists should at least have master's degree.

```
# replacing two values sentences to a value "No degree"
replace_values = {"I did not complete any formal education past high school":"No degree",
                  "Some college/university study without earning a bachelor's degree":"No degree"}
df_demog_ds = df_demog_ds.replace({"FormalEducation":replace_values})# Standardizing records with no degree
ds_edu = df_demog_ds['FormalEducation'].value_counts() # Frequency table of education level for data scientists (Stackoverflow)
ds_edu = pd.DataFrame({"Education":ds_edu.index, "Count":ds_edu.values})# Creating a dataframe with frequency table
ds_edu['Percentage'] = round(ds_edu.Count/ds_edu.Count.sum(),2)
print("Distribution of data scientists education\n",format(ds_edu))
plt.barh(ds_edu["Education"],ds_edu["Percentage"], align = "center")
plt.xlabel("Percentage")
plt.ylabel("Education")
plt.title("Data Scientists Education")
for d,c in zip(ds_edu['Percentage'],ds_edu['Education']):
    plt.text(d, c, str(d))
```

```
Distribution of data scientists education
              Education  Count  Percentage
0       Master's degree    635        0.50
1      Doctoral degree    326        0.26
2     Bachelor's degree    252        0.20
3  Professional degree     25        0.02
4             No degree     25        0.02
```



1.2.A: Salary of data scientists in Australian dollars

Maximum and median salary of data scientists in AUD is 860,017 and 89,429 respectively.

```
# Printing maximum salary in AUD
print("Maximum salary of data scientistis in AUD is {}"
       .format( round(df_demog_ds['compensationAUD'].max())))
```

```
Maximum salary of data scientistis in AUD is 860017
```

```
#Printing median salary in AUD
print("Median salary of data scientistis in AUD is {}"
       .format(round(df_demog_ds['compensationAUD'].median())))
```

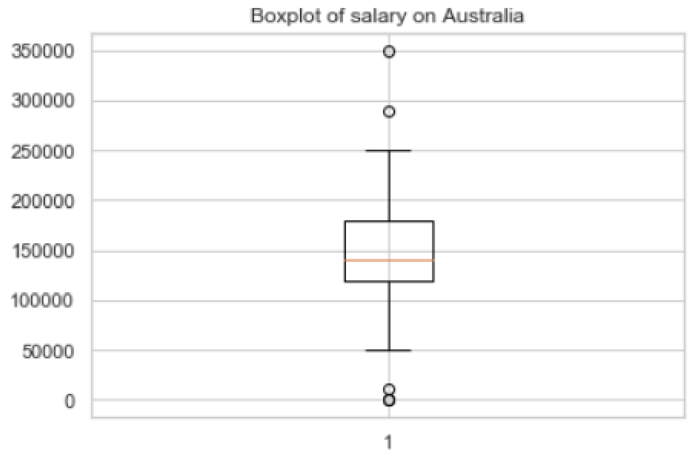
```
Median salary of data scientistis in AUD is 89429
```

1.2.B: Maximum and median salary of Data scientists from Australia

Maximum and median salary of data scientists in Australia is AUD 350,000 and AUD 140,000 respectively. From boxplot, we can infer that distribution is positively skewed with few outliers.

```python
#Creating dataframe for data scientists working in Australian
df_demog_ds_AUS = df_demog_ds[df_demog_ds['Country']=="Australia"]
plt.title("Boxplot of salary on Australia")
plt.boxplot(df_demog_ds_AUS['compensationAUD'])
plt.title("Boxplot of salary on Australia")
print(df_demog_ds_AUS.shape)
```

```
(29, 20)
```




```python
# Maximum and median salary in AUD of data scientists working in Australia
print("Maximum salary of Australian respondents in AUD is {}"
      .format( round(df_demog_ds_AUS['compensationAUD'].max()))) # Printing maximum salary in AUD
print("Median salary of Australian respondents  in AUD is {}"
      .format(round(df_demog_ds_AUS['compensationAUD'].median()))) #Printing median salary in AUD
```

```
Maximum salary of Australian respondents in AUD is 350000
Median salary of Australian respondents  in AUD is 140000
```

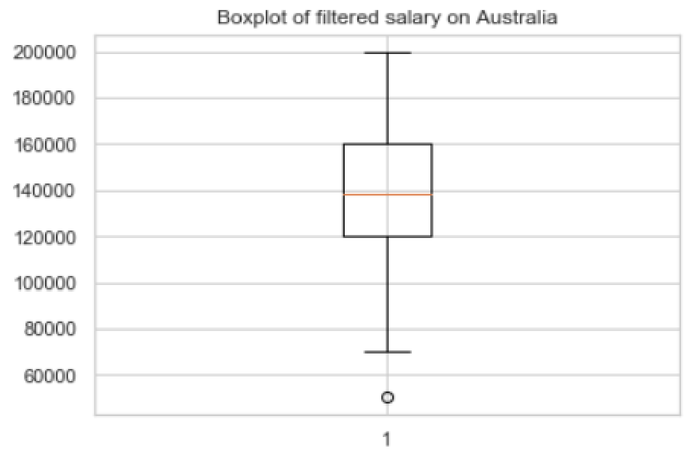
1.2.C: Salary of Australian Data science respondents after removing outliers

Now, maximum and median salary is AUD 200,000 and AUD 138,000 respectively. From boxplot, we can infer that now distribution is approximately symmetrical with one lower outlier at AUD 50,000.

```python
#Filtering data where Australian respondents has salary between 40,000 and 250,000
df_demog_ds_AUS_clean = df_demog_ds_AUS[(df_demog_ds_AUS['compensationAUD'] > 40000) &
                                        (df_demog_ds_AUS['compensationAUD'] < 250000)]
print("Maximum salary of filtered Australian respondents in AUD is {}"
      .format( round(df_demog_ds_AUS_clean['compensationAUD'].max()))) # Printing maximum salary in AUD
print("Median salary of filtered Australian respondents  in AUD is {}"
      .format(round(df_demog_ds_AUS_clean['compensationAUD'].median()))) #Printing median salary in AUD
print("Shape of filtered Australian respondents is {}". format(df_demog_ds_AUS_clean.shape))
```

```
Maximum salary of filtered Australian respondents in AUD is 200000
Median salary of filtered Australian respondents  in AUD is 138000
Shape of filtered Australian respondents is (20, 20)
```

```python
#Boxplot of Australian respondents salary after removing outliers
plt.title("Boxplot of filtered salary on Australia")
plt.boxplot(df_demog_ds_AUS_clean['compensationAUD'])# Outlier at compensation AUD 50,000
plt.show()
```

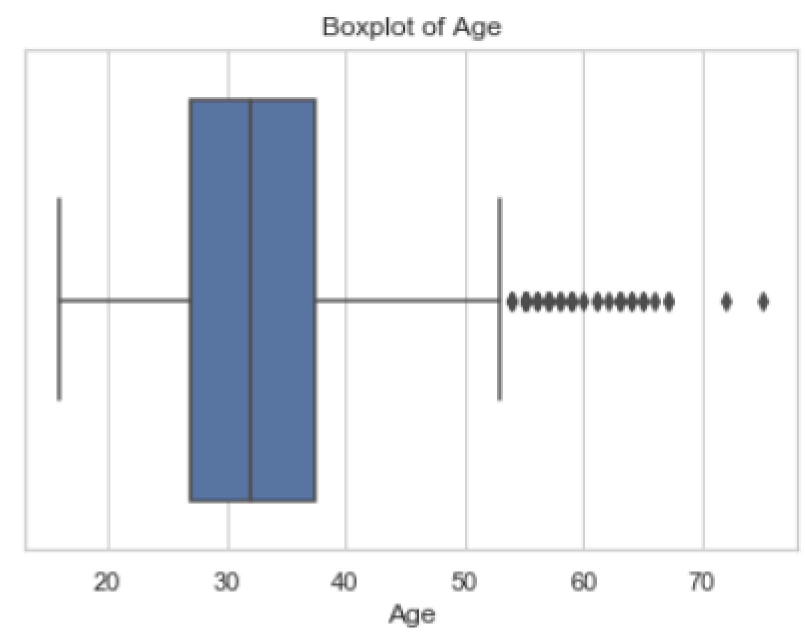


1.3: Exploring the data scientists Demographics

1.3.1: Age

From boxplot of age of data scientists, we can infer that distribution is positively skewed and has outliers at upper end of the distribution. Median age is 32 years with maximum age of 75 years and minimum of 16 years.

```python
#Boxplot of age of all data scientists
plt.title("Boxplot of Age")
ax = sns.boxplot(x=df_demog_ds["Age"])
```



1.3.A: Questions

1.3.A.1: Five number summary of age of data scientists

```
#Five number summary for all data scientists
#age including standard deviation and count
round(df_demog_ds["Age"].describe())
```

```
count    1263.0
mean       34.0
std         9.0
min        16.0
25%        27.0
50%        32.0
75%        38.0
max        75.0
Name: Age, dtype: float64
```

1.3.A.2: Mean age of all data scientists

```
# What is the mean age of all data scientists?
print("Mean age of all data scientists is"
      ,format(round(df_demog_ds['Age'].mean())))
```

```
Mean age of all data scientists is 34
```

1.3.A.3: Median age of all data scientists

```
# What is the median age of all data scientists?
print("Median age of all data scientists is",
      format(round(df_demog_ds['Age'].median())))
```

```
Median age of all data scientists is 32
```

1.3.A.3: Number of data scientists aged between 24 and 60

```
# Your code: How many data scientsits aged between 24 and 60
print("Number of data scientists aged between 24 and 60 are",
      format(((df_demog_ds['Age']>23) & (df_demog_ds['Age']<61)).sum()))
```

```
Number of data scientists aged between 24 and 60 are 1188
```

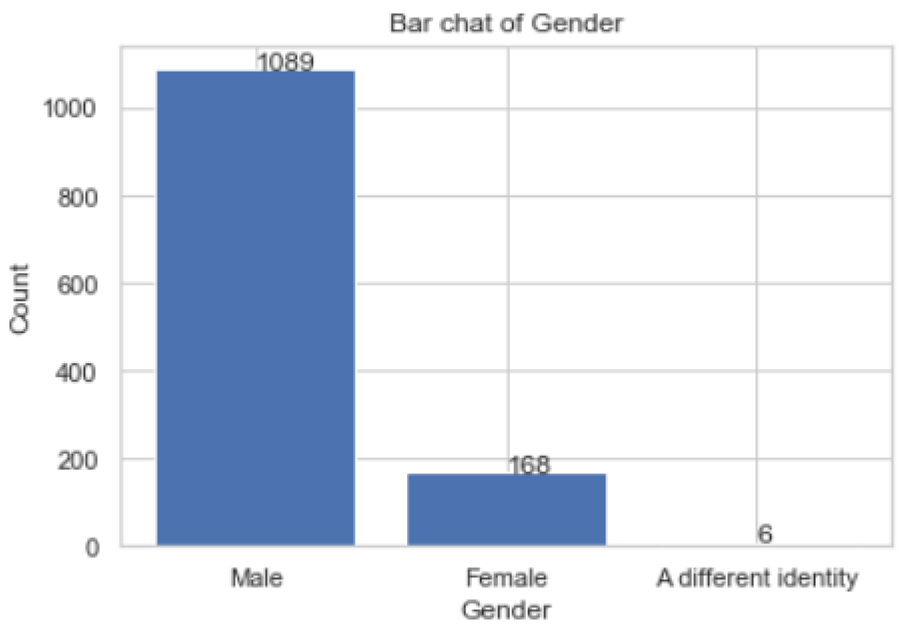
1.3.A.4: How many respondents under 18?

```
# Your Code: how many respondents under 18?
print("There is only",
      format((df_demog_ds['Age']<18).sum()),
      "data scientist aged under 18 years")
```

```
There is only 1 data scientist aged under 18 years
```

1.3.2: Gender distribution of data scientists using a bar chart

```
#Gender distribution of data scinetists
plt.title("Bar chat of Gender")
freq = df_demog_ds['GenderSelect'].value_counts()
plt.bar(freq.index, freq.values)
plt.xlabel('Gender')
plt.ylabel('Count')
for a,b in zip(freq.index,freq.values):
    plt.text(a, b, str(b))
```

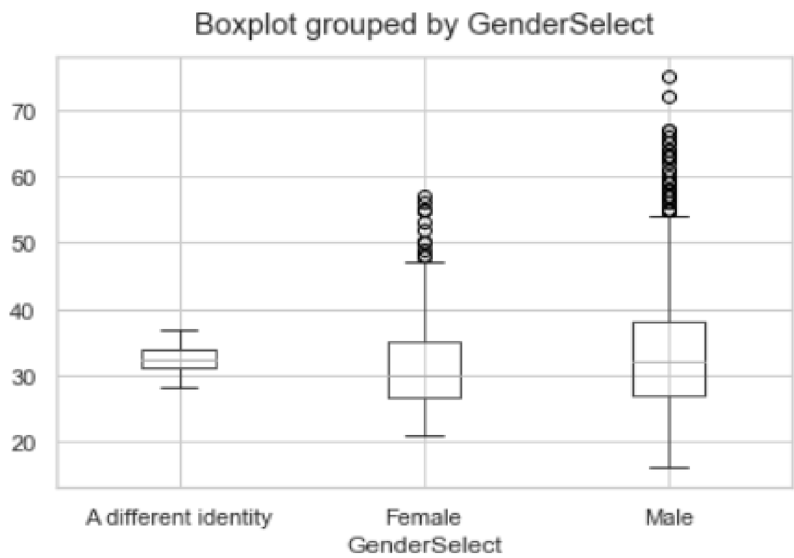


Box plot of age of all data scientists according to the gender:

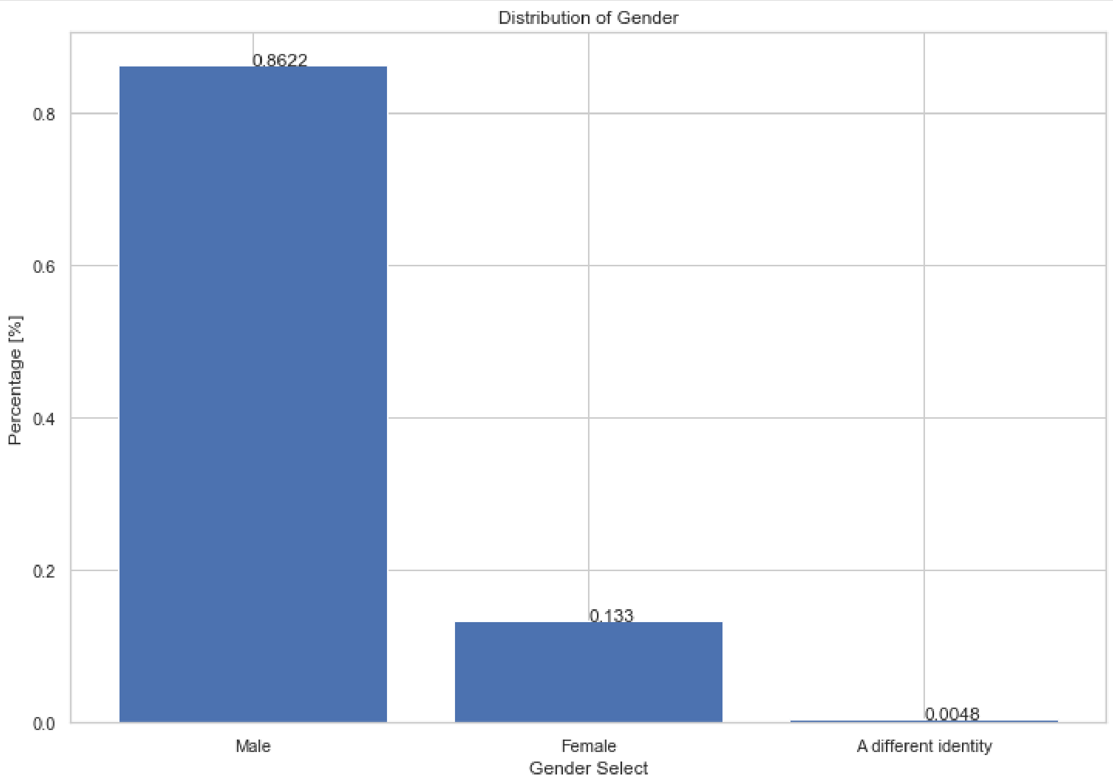
Distribution of gender of different kind is symmetrical whereas distribution of female and male data scientists is positively skewed with upper outliers.

```
df_demog_ds.boxplot('Age', by='GenderSelect')
plt.title("")
```

```
Text(0.5, 1.0, '')
```



1.3.B: Relative frequency bar chart of gender of data scientists

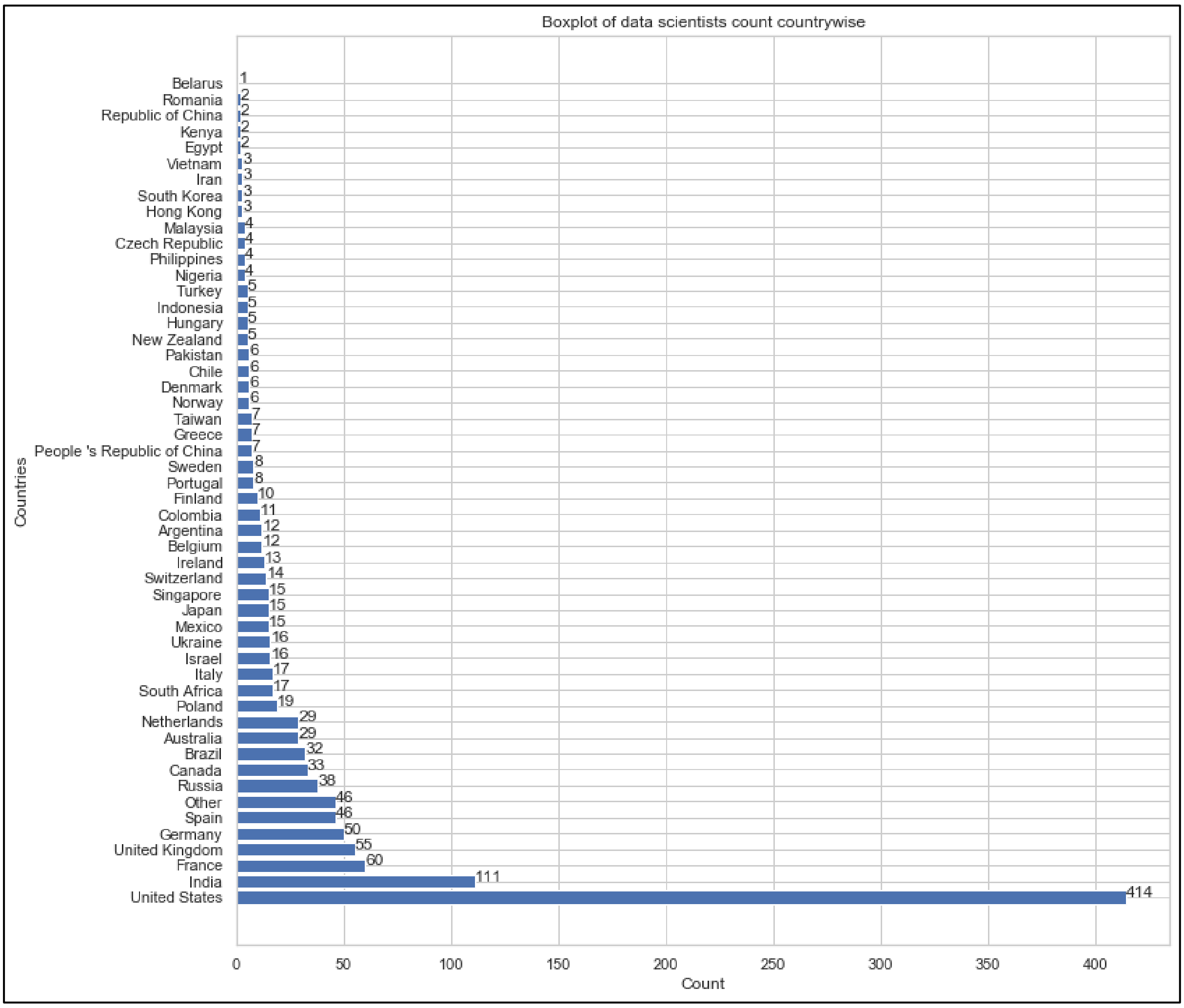


1.3.C: List of top 5 countries with data scientists

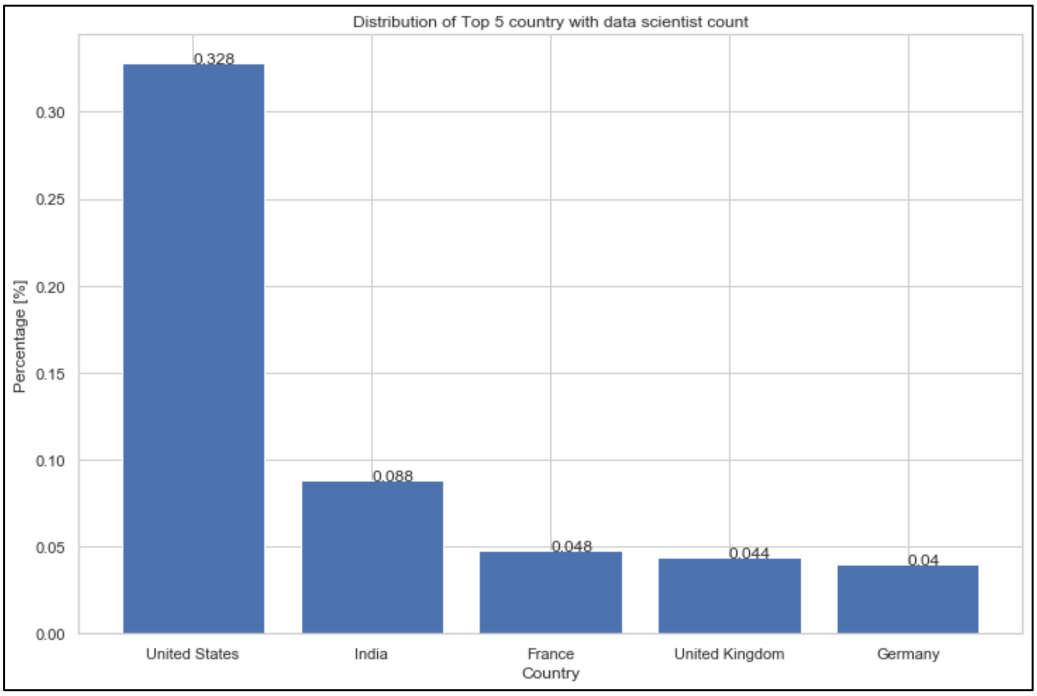
```
#Top 5 countries based on number of data scientists
df_country = df_demog_ds['Country'].value_counts()
print("Top 5 countries with data scientists are\n",
      format(df_country.nlargest(5)))
```

```
Top 5 countries with data scientists are
 United States      414
India              111
France              60
United Kingdom      55
Germany             50
Name: Country, dtype: int64
```

Suitable plot of frequency of data scientist's country wise. Bar plot is more suitable than count plot and box plot. Since the question did not specify if the plot should contain top countries or all, thus plot of all countries is plotted, as top 5 countries is plotted in next question.



1.3.D: Percentage bar chart of top 5 countries with data scientists based on all countries



1.3.E: Mean and median age for each gender for the United States, India, Australia, Pakistan

```
# Filtering data for countries United States, India, Australia, and Pakistan
df_countries_age_gen= df_demog_ds[df_demog_ds['Country']
                                  .isin(['United States', 'India', 'Australia','Pakistan'])]
avg_age = round(df_countries_age_gen
                .groupby(['Country', 'GenderSelect'])['Age'].mean())
med_age = round(df_countries_age_gen
                .groupby(['Country', 'GenderSelect'])['Age'].median())
print("Average gender wise age of data scientists in US, India, Australia, Pakistan is\n", format(avg_age))
print("\n Median gender wise age of data scientists in US, India, Australia, Pakistan is\n", format(med_age))
```

```
Average gender wise age of data scientists in US, India, Australia, Pakistan is
 Country        GenderSelect
Australia      Female                  33.0
               Male                    35.0
India          Female                  29.0
               Male                    30.0
Pakistan       Male                    32.0
United States  A different identity    31.0
               Female                  33.0
               Male                    36.0
Name: Age, dtype: float64

 Median gender wise age of data scientists in US, India, Australia, Pakistan is
 Country        GenderSelect
Australia      Female                  31
               Male                    34
India          Female                  27
               Male                    28
Pakistan       Male                    27
United States  A different identity    31
               Female                  31
               Male                    33
Name: Age, dtype: int64
```

Part 2 - Data Science Job Advertising Data

2.1.A: Extract the token and append them into the list 'token'.

```
lower = []
for item in df_text['job_description']:
    lower.append(item.lower())           # lowercase description

regex = RegexpTokenizer(r"\s+", gaps=True)
tokens = []
df_lower = pd.DataFrame(lower)
for index in range(len(lower)):
  for tokenized in (df_lower.apply(lambda row: regex.tokenize(row[index]),axis=0)):
     tokens.append(tokenized)

tokenizedList = [item for elem in tokens for item in elem]  #Creating list of words from tokens
```

2.1.B: List of words with frequency > 6000 after using stop words

```
stop_words = set(stopwords.words('english'))
tokenizedList_filtered = [token for token in tokenizedList
                        if token not in stop_words]           #Filtering words not in stop_words list
tokenizedList_filtered = pd.DataFrame(tokenizedList_filtered)  #Creating dataframe for the filtered words from stop_words
freq6000 = tokenizedList_filtered[0].value_counts()            #Creating frequency table from the filtered words from stop_words
freq6000 = freq6000.drop(freq6000[freq6000.index.isin(["-","/", "&"])].index)
#Dropping special characters from the frequency list
freq6000 = freq6000[freq6000.values>6000]                  #Filtering words with frequency greater than 6000
```

```
freq6000
```

```
data           114154
experience      51004
business        30610
work            26222
machine         19840
                ...
customer         6376
company          6372
quantitative     6261
big              6224
employment       6176
Name: 0, Length: 66, dtype: int64
```

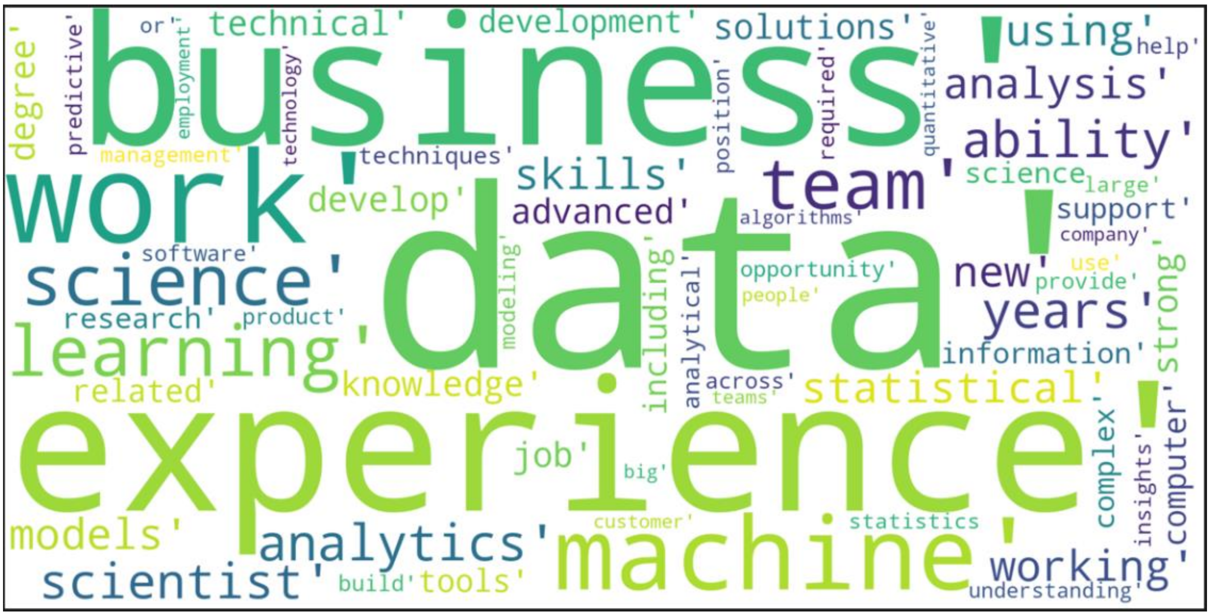
2.1.C: Top 10 high frequency words in 'freq6000'

```
# Top 10 high frequency words in 'freq6000'
freq6000_top_10 = pd.DataFrame(freq6000.nlargest(10))
freq6000_top_10
```

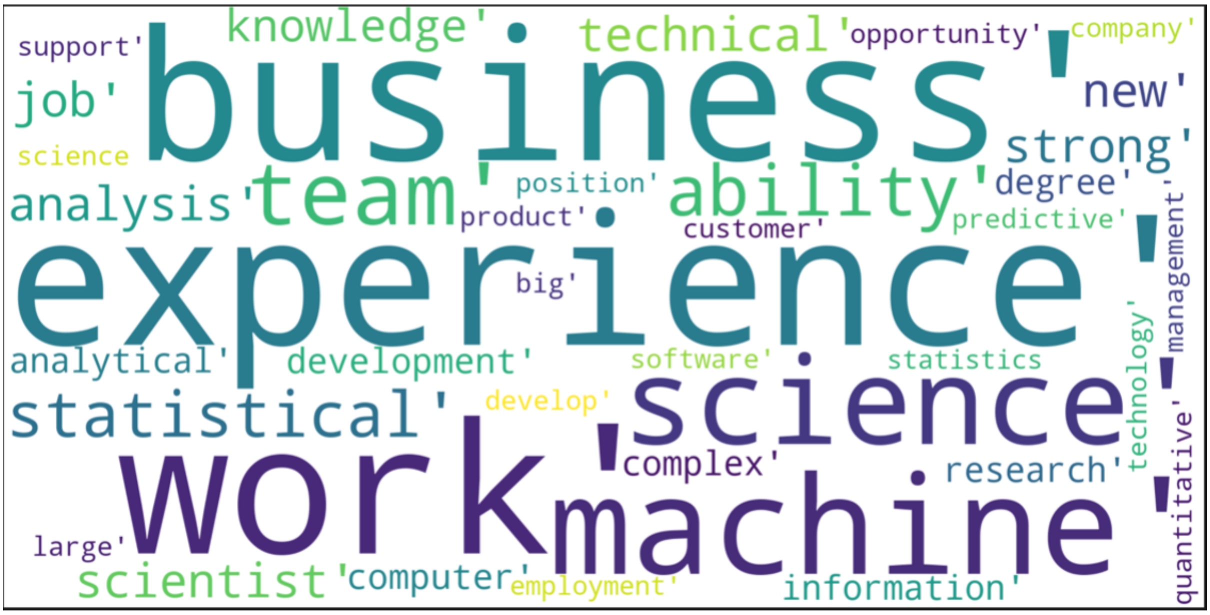
	0
data	114154
experience	51004
business	30610
work	26222
machine	19840
learning	19247
science	16973
team	16351
analytics	16218
ability	15021

2.1.D: Discovery from self-defined text analysis task

In this analysis, I have used word cloud on words with frequency greater than 6000. From this we can see, more important aspects of being a data scientist are analytical with business and data, machine learning, experience as data scientist, statistics knowledge, developing tools etc.



Secondly, on words after using stop words, I have tagged form of the word as per English grammar. Words tagged with base form of adjective, noun, verb, and adverb were filtered. Then words with frequency greater 6000 were filtered. This approach gives better important words which are essential to be a data scientist, like statistical analysis, machine learning, strong technical knowledge, business, research, experience etc. Thus, top 10 words with this approach is different than the previous approach.



References:

Stack Overflow. 2020. *Stack Overflow - Where Developers Learn, Share, & Build Careers*. [online] Available at: <https://stackoverflow.com/> [Accessed 17 April 2020].

2020. [online] Available at: <https://www.youtube.com/watch?v=AKcxEfz-EoI> [Accessed 17 April 2020].

Bartosz Mikulski. 2020. *Word Cloud From A Pandas Data Frame*. [online] Available at: <https://www.mikulskibartosz.name/word-cloud-from-a-pandas-data-frame/> [Accessed 17 April 2020].

Cs.nyu.edu. 2020. *Penn Part-Of-Speech Tags*. [online] Available at: <https://cs.nyu.edu/grishman/jet/guide/PennPOS.html> [Accessed 17 April 2020].