RESUME SCREEING.....

In [ ]:

```
import pandas as pd # for loading the data
import numpy as np # for arrays
import matplotlib.pyplot as plt # for visulaizing thr data
import seaborn as sns # for heatmaps
%matplotlib inline # for matplot

from google.colab import files # for google colab users used for import files
uploaded=files.upload()
```

Choose Files No file chosen

Upload widget is only available when the cell has been executed in the current browser session.
Please rerun this cell to enable.

```
Saving Resumesdata.csv to Resumesdata (1).csv
```

In [ ]:

```
df=pd.read_csv('Resumesdata.csv')
```

In [ ]:

```
df
```

Out[6]:

	Category	Resume
0	Data Science	Skills * Programming Languages: Python (pandas...
1	Data Science	Education Details \r\nMay 2013 to May 2017 B.E...
2	Data Science	Areas of Interest Deep Learning, Control Syste...
3	Data Science	Skills â□¢ R â□¢ Python â□¢ SAP HANA â□¢ Table...
4	Data Science	Education Details \r\n MCA YMCAUST, Faridab...
...	...	...
957	Testing	Computer Skills: â□¢ Proficient in MS office (...
958	Testing	â□□ Willingness to accept the challenges. â□□ ...
959	Testing	PERSONAL SKILLS â□¢ Quick learner, â□¢ Eagerne...
960	Testing	COMPUTER SKILLS & SOFTWARE KNOWLEDGE MS-Power ...
961	Testing	Skill Set OS Windows XP/7/8/8.1/10 Database MY...

962 rows × 2 columns

In [ ]:

```
df.head(5)
```

Out[7]:

	Category	Resume
0	Data Science	Skills * Programming Languages: Python (pandas...
1	Data Science	Education Details \r\nMay 2013 to May 2017 B.E...
2	Data Science	Areas of Interest Deep Learning, Control Syste...
3	Data Science	Skills â□¢ R â□¢ Python â□¢ SAP HANA â□¢ Table...
4	Data Science	Education Details \r\n MCA YMCAUST, Faridab...

Checking the unique values in categories

In [ ]:

```
print(df['Category'].unique())
print ('sum of unique categories:{}'.format(len(df['Category'].unique())))
```

```
['Data Science' 'HR' 'Advocate' 'Arts' 'Web Designing'
 'Mechanical Engineer' 'Sales' 'Health and fitness' 'Civil Engineer'
 'Java Developer' 'Business Analyst' 'SAP Developer' 'Automation Testing'
 'Electrical Engineering' 'Operations Manager' 'Python Developer'
 'DevOps Engineer' 'Network Security Engineer' 'PMO' 'Database' 'Hadoop'
 'ETL Developer' 'DotNet Developer' 'Blockchain' 'Testing']
sum of unique categories:25
```

In [ ]:

```
df['Category'].value_counts()
```

Out[9]:

```
Java Developer               84
Testing                      70
DevOps Engineer              55
Python Developer             48
Web Designing                45
HR                           44
Hadoop                       42
Blockchain                   40
ETL Developer                40
Operations Manager           40
Data Science                 40
Sales                        40
Mechanical Engineer          40
Arts                         36
Database                     33
Electrical Engineering       30
Health and fitness           30
PMO                          30
Business Analyst             28
DotNet Developer             28
Automation Testing           26
Network Security Engineer    25
SAP Developer                24
Civil Engineer               24
Advocate                     20
Name: Category, dtype: int64
```

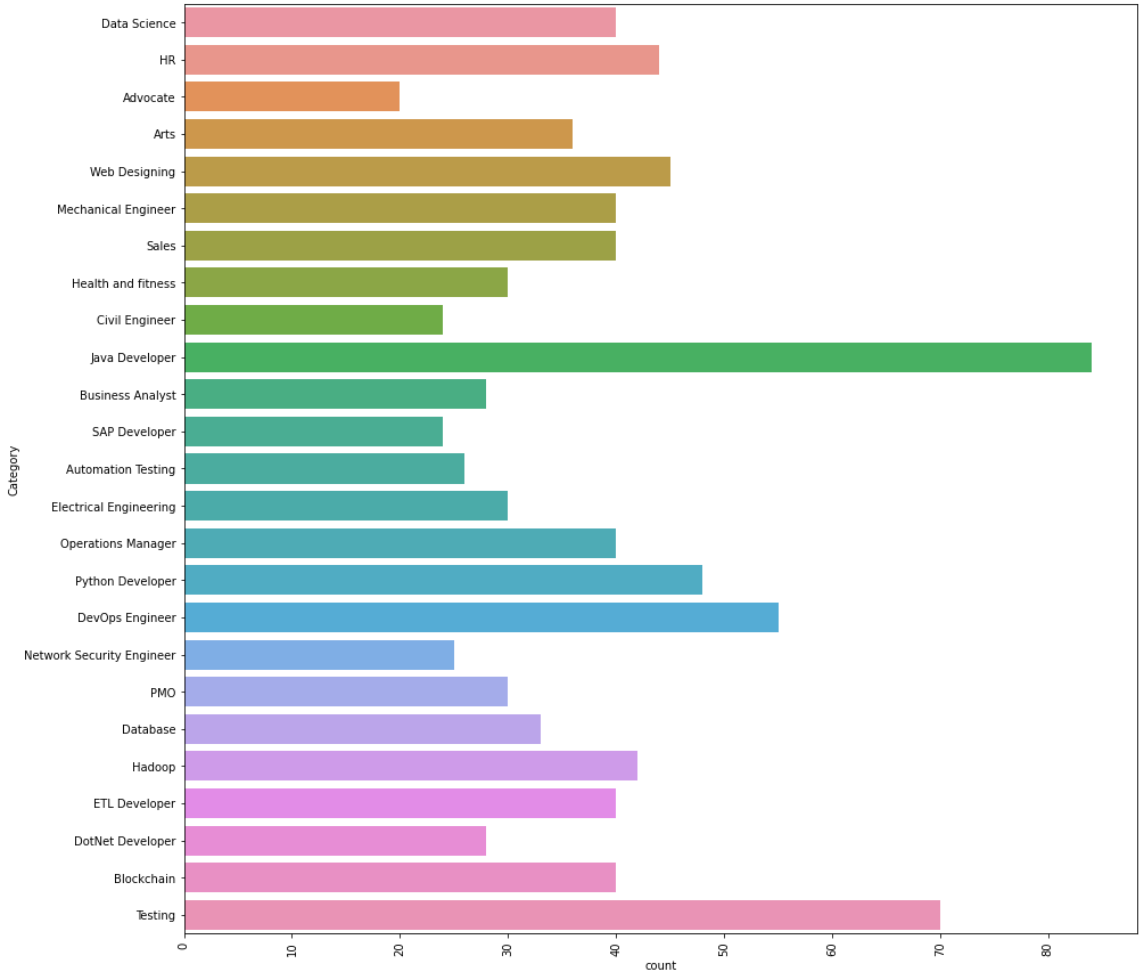
Visuailizing the category data

In [ ]:

```
plt.figure(figsize=(15,15))
plt.xticks(rotation=90)
sns.countplot(y="Category",data=df)
```

Out[10]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8234677a00>
```



Cleaning the resume column

Cleaning the data by remving spl charaters which are in below

In [ ]:

```
import re #to indicate spl charaters
def cleanResume(resumeText):
    resumeText = re.sub('http\S+\s*', ' ', resumeText)  # remove URLs
    resumeText = re.sub('RT|cc', ' ', resumeText)  # remove RT and cc
    resumeText = re.sub('#\S+', '', resumeText)  # remove hashtags
    resumeText = re.sub('@\S+', '  ', resumeText)  # remove mentions
    resumeText = re.sub('[%s]' % re.escape("""!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~"""), ' ',
    resumeText = re.sub(r'[^\x00-\x7f]',r' ', resumeText)
    resumeText = re.sub('\s+', ' ', resumeText)  # remove extra whitespace
    resumeText = re.sub('Ã¢Â€Â¢', ' ', resumeText)
    return resumeText

df['cleaned_resume'] = df.Resume.apply(lambda x: cleanResume(x))
df.head()
```

Out[11]:

	Category	Resume	cleaned_resume
0	Data Science	Skills * Programming Languages: Python (pandas...	Skills Programming Languages Python pandas num...
1	Data Science	Education Details \r\nMay 2013 to May 2017 B.E...	Education Details May 2013 to May 2017 B E UIT...
2	Data Science	Areas of Interest Deep Learning, Control Syste...	Areas of Interest Deep Learning Control System...
3	Data Science	Skills â□¢ R â□¢ Python â□¢ SAP HANA â□¢ Table...	Skills R Python SAP HANA Tableau SAP HANA SQL ...
4	Data Science	Education Details \r\n MCA YMCAUST, Faridab...	Education Details MCA YMCAUST Faridabad Haryan...

Plotting word cloud

Natural Language Took Kit (NTLK) used for anayzing for the huamn text and the stopwords used for to stop the words in list. by downloading the package punkt it is unspervised model used for train the unlabelled data.

In [ ]:

```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
from nltk.corpus import stopwords # to stop list words
import string
from wordcloud import WordCloud # to visualize words
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

In [ ]:

```
oneSetOfStopWords = set(stopwords.words('english')+['``',"''"])
totalWords =[]
Sentences = df['Resume'].values
cleanedSentences = ""
for i in range(0,160):
    cleanedText = cleanResume(Sentences[i])
    cleanedSentences += cleanedText
    requiredWords = nltk.word_tokenize(cleanedText)
    for word in requiredWords:
        if word not in oneSetOfStopWords and word not in string.punctuation:
            totalWords.append(word)

wordfreqdist = nltk.FreqDist(totalWords)
mostcommon = wordfreqdist.most_common(50)

wc = WordCloud().generate(cleanedSentences)
plt.figure(figsize=(15,15))
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.show()
```

Encoding the column category

In [ ]:

```
from sklearn.preprocessing import LabelEncoder # for labeling values

var_mod = ['Category']
le = LabelEncoder()
for i in var_mod:
    df[i] = le.fit_transform(df[i])

df.head()
```

Out[18]:

	Category	Resume	cleaned_resume
0	6	Skills * Programming Languages: Python (pandas...	Skills Programming Languages Python pandas num...
1	6	Education Details \r\nMay 2013 to May 2017 B.E...	Education Details May 2013 to May 2017 B E UIT...
2	6	Areas of Interest Deep Learning, Control Syste...	Areas of Interest Deep Learning Control System...
3	6	Skills â□¢ R â□¢ Python â□¢ SAP HANA â□¢ Table...	Skills R Python SAP HANA Tableau SAP HANA SQL ...
4	6	Education Details \r\n MCA YMCAUST, Faridab...	Education Details MCA YMCAUST Faridabad Haryan...

Vectoizing and spliting the dataset

Vectorization means that a function can be evaluated on a whole array of values at once instead of looping over individual entries. TfidfVectorizer uses an in-memory vocabulary (a python dict) to map the most frequent words to feature indices and hence compute a word occurrence frequency (sparse) matrix.

In [ ]:

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from scipy.sparse import hstack

requiredText = df['cleaned_resume'].values
requiredTarget = df['Category'].values

word_vectorizer = TfidfVectorizer(
    sublinear_tf=True,
    stop_words='english',
    max_features=1500)
word_vectorizer.fit(requiredText)
WordFeatures = word_vectorizer.transform(requiredText)

X_train,X_test,y_train,y_test = train_test_split(WordFeatures,requiredTarget,random_stat
```

Training the model

In [ ]:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.multiclass import OneVsRestClassifier
from sklearn import metrics
clf = OneVsRestClassifier(KNeighborsClassifier())
clf.fit(X_train, y_train)
prediction = clf.predict(X_test)
print('Accuracy of KNeighbors Classifier on training set: {:.2f}'.format(clf.score(X_tra
print('Accuracy of KNeighbors Classifier on test set: {:.2f}'.format(clf.score(X_test, y

print("\n Classification report for classifier %s:\n%s\n" % (clf, metrics.classification
```

```
Accuracy of KNeighbors Classifier on training set: 0.99
Accuracy of KNeighbors Classifier on test set: 0.99

 Classification report for classifier OneVsRestClassifier(estimator=KNeigh
borsClassifier()):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         4
           1       1.00      1.00      1.00         4
           2       1.00      0.75      0.86         8
           3       1.00      1.00      1.00        15
           4       0.91      1.00      0.95        10
           5       0.91      1.00      0.95        10
           6       1.00      1.00      1.00        14
           7       1.00      1.00      1.00        10
           8       1.00      0.87      0.93        15
           9       1.00      1.00      1.00        10
          10       1.00      1.00      1.00        11
          11       0.93      1.00      0.96        13
          12       1.00      1.00      1.00        12
          13       1.00      1.00      1.00        13
          14       1.00      1.00      1.00         9
          15       0.96      1.00      0.98        26
          16       1.00      1.00      1.00         9
          17       1.00      1.00      1.00         7
          18       1.00      1.00      1.00        11
          19       1.00      1.00      1.00         8
          20       1.00      1.00      1.00        13
          21       1.00      1.00      1.00         8
          22       1.00      1.00      1.00        15
          23       1.00      1.00      1.00        20
          24       1.00      1.00      1.00        14

    accuracy                           0.99       289
   macro avg       0.99      0.98      0.99       289
weighted avg       0.99      0.99      0.99       289
```

In [ ]: