MARKET SEGMENTATION AND ANALYSIS OF

MACHINE LEARNING JOB MARKET IN INDIA

UNDER THE INTERNSHIP

AT FEYNNLABS

SUBMITTED BY:

Team Leader: Akash Jyoti Borah

Members: Pranil Savale

FeyNN Labs

EXPERIMENT WITH YOUR KNOWLEDGE

Abstract

Segmentation is the process of isolating possible target groups in order to identify which ones would provide the best return on investment for your marketing efforts. Segmentation is determined by certain factors such as an individual's age, income, interests, and habits. When you segment different markets, you discover more about their underlying beliefs and what will eventually draw them to your brand. The basic unit of analysis for grouping consumers is the distinction between traditional market segmentation methods and the jobs-based segmentation methodology. The basic unit of analysis for traditional segmentation is the qualities of the clients themselves. A task that clients are attempting to complete is the fundamental unit of analysis for jobs-based segmentation. A market is traditionally characterised by the product and service categories specified by solution providers. According to Jobs Theory, a market is an aggregate of all available options, both provider and non-provider, that customers consider as being capable of meeting their demands in terms of getting a job done. Job segmentation provides a firm with a substantial competitive edge because it allows them to predict the value that consumers want—even before customers are aware of certain demands. A corporation may swiftly and efficiently improve existing offers and develop new ones that meet client demands better than competitor alternatives at the lowest feasible cost.

Introduction

Machine Learning employment have increased tremendously in recent years, with a high demand in the industry. As machine learning makes significant inroads into numerous businesses, so does the demand for machine learning engineers.

Machine learning (ML) is a subset of Artificial Intelligence (AI). AI is present everywhere, from gaming stations to the management of complicated data at work. Computer engineers and scientists are working hard to instil intelligent behaviour in machines, allowing them to think and respond to real-world events. AI is progressing from being a research issue to being in the early phases of industry implementation. Google and Facebook have made significant investments in Artificial Intelligence and Machine Learning and are already incorporating it into their businesses. However, this is only the beginning; over the next several years, we may see AI gradually creep into one device after another.

Simply expressed, the objective of AI is to make computers/computer programmes clever enough to mimic the behaviour of the human mind. ML is the study of creating and implementing algorithms that can learn from previous situations. If a certain behaviour has occurred in the past, you can forecast whether or not it will occur again. That is, if there are no previous examples, there can be no forecast.

ML can be used to address difficult problems such as credit card fraud detection, self-driving automobiles, and facial detection and identification. ML employs complicated algorithms that run over enormous data sets indefinitely, evaluating patterns in data and allowing computers to adapt to circumstances for which they have not been expressly trained. Machines learn from past in order to deliver consistent outcomes. To forecast reasonable outcomes, ML algorithms employ Computer Science and Statistics.

JOB MARKET OVERVIEW

When it comes to work prospects, the scope of Machine Learning in India and other areas of the world is great in comparison to other professional disciplines. According to a Monster research, big data analytics and AI/ML would be the most in-demand talents in India in 2022. According to a Monster analysis, with fast tech adoption across industries and completely tech-enabled sectors like as IT and BFSI, the role of AI/ML will only rise in 2022. The average yearly income of an entry-level AI engineer in India is about 8 lakhs, which is much more than the average salary of any other engineering graduate. The compensation of an AI engineer at a high level might reach 50 lakhs.

A fresher can acquire a machine learning job if he or she possesses the necessary abilities. To have a successful career in the machine learning environment, newcomers must prepare how they will perform effectively and collaborate closely with others who have extensive expertise in the same sector.

To Start Career in ML/ AI Field, following skills are needed (may be acquired):

- Statistical Skill
- Mathematical skills and Probability Programming skills
- Advanced Signal Processing Techniques
- Distributed Computing
- Work on projects

Problem Statement

Finding Companies most probable to hire an **ML Engineer/Data Analyst Applicant** in respect to his/her skillset.

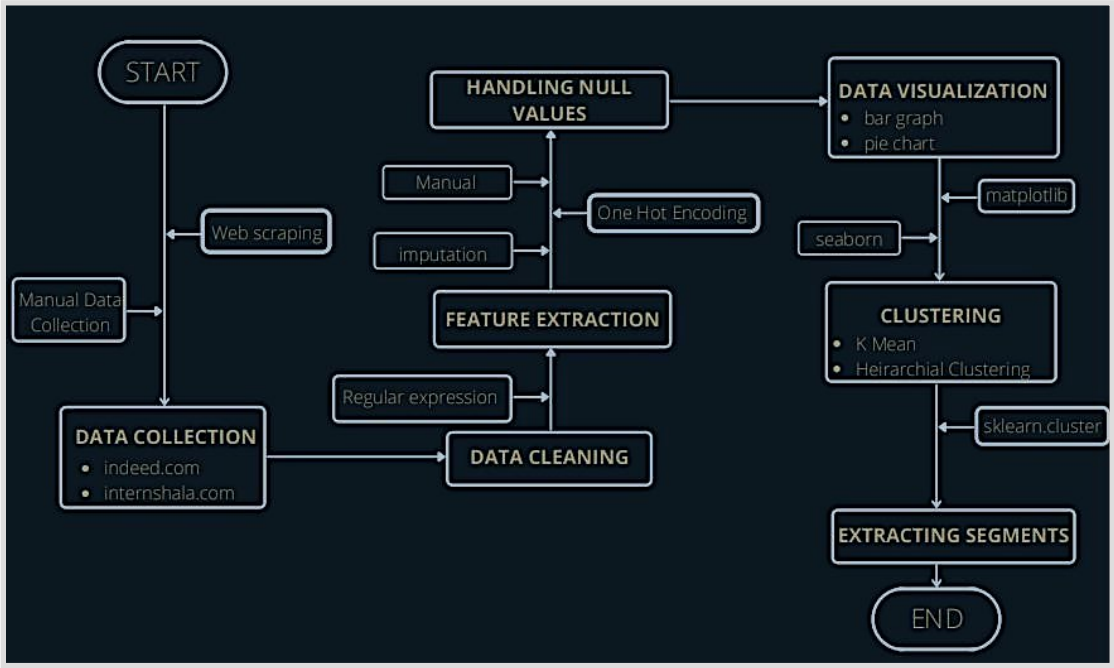
Data Collection/Scraping based on

1. Geography

2. Company’s field of work
3. Company size
4. Upcoming vacancies in respect to company’s growth (IPO/Funding etc.)
5. Machine Learning/Data Analysis Skills currently most demanded in the market in respect to
    i) Experience required,
    ii) Time required to acquire the skill,
    iii) Vacancies open
    iv) Salary etc.

We have to analyze Machine Learning Job Market in India with respect to the given problem statement using Segmentation analysis and outline the segments most optimal to apply or prepare for Machine Learning Jobs.

SEGMENTS: Apart from Geographic, Demographic, Psychographic, Behavioral segments, teams can consider different CATEGORY of Segments for the Segmentation Tasks, based on AVAILABILITY OF DATA.



Break Down of the Problem Statement using Fermi Estimation

HISTORY AND BACKGROUND

Enrico Fermi (1901-1954), an Italian physicist best known for his contributions to nuclear physics and the development of quantum theory, is responsible for the eponymous Fermi issue. Fermi received the Nobel Prize in Physics in 1938 for his work on the nuclear process. Fermi was famous for being able to construct a numerical estimate based on information that appeared to be insufficient to produce a quantitative conclusion. His method of consecutive approximations "zeroed in" on solutions by determining that the value in issue was unquestionably more than a specific number and less than another amount. He would go through an issue in this manner until he had a quantifiable response within certain boundaries.

Fermi questions are now a fixture of high school scientific contests and may be included in many college science textbooks. Popular Fermi questions include: How many gallons of gasoline are used in France each year? How many hot dogs will be consumed during big league baseball games in a single season?

A CLASSIC EXAMPLE

Let's begin by solving a classic Fermi question: How many piano tuners are listed in the San Francisco phonebook? At first glance it may not seem possible to arrive at a solution without simply guessing, but by making reasonable assumptions, one can make surprisingly accurate estimates. Let's see how it might be done:

• estimate the population of San Francisco at two million;
• assume that families own pianos;
• estimate five persons per family which yields 400,000 families in San Francisco;
• estimate that one out of ten families have pianos, which yields 40,000 pianos in the city;
• estimate that pianos are tuned once per year which yields 40,000 pianos tuned each year;
• estimate that each tuner can tune four pianos per day, 200 days per year which yields 800 pianos per tuner per year;
or 40,000 pianos/800 tuned per tuner = 50 piano tuners.

Checking online, we find that 46 piano tuners are listed for San Francisco (yellowpages.com, 2009). At any estimation step above, we could have chosen a different number, e.g., we could have estimated the population to be 1 million, but this would still yield a reasonably accurate

estimate. In fact, the more assumptions and estimates made, the more estimation errors tend to cancel. Interestingly, researchers have found that the average of two guesses is more accurate than either guess alone (Vul & Pashler, 2008). So, using more than one estimator and averaging their estimates should lead to greater accuracy.

Applying Fermi Estimation to Business Problems

When neither time nor resources are available for standard evaluations, it is frequently necessary to create rapid approximations. This is especially true during the concept stage of product development, when even a rough estimate might help save unnecessary expenses. The Fermi issue, with which the scientific and technical communities have long been familiar, is a good place to start when learning about order of magnitude estimation. Although there are various worked-out answers to Fermi issues, there is no systematic way to solve them. A quick estimate of market size, costs, or technical feasibility may be required. A computation of this type eliminates specifics, concentrates solely on main elements, and seeks an approximation within an order of magnitude (a power of 10) of the real result. Whether or not to move on to the next stage of development can then be based on an informed prediction of the product's potential. The Fermi question is a rapid estimating procedure that the scientific and engineering communities have long been familiar with for order of magnitude estimation.

The goal of order of magnitude estimation is to provide an estimate that is as least as accurate as the true answer, rounded to the nearest power of ten. In other words, if the exact answer is 40 million, an estimate as low as 10 million or as high as 100 million would be within the acceptable range. A Fermi question is one that can be addressed through an informed estimating technique that depends on making fair assumptions rather than studying related knowledge. "How many retail malls are there in the United States?" is a common Fermi inquiry. Fermi problems demand numerous phases of estimation, and the estimator must begin by identifying and estimating the data values or components required to calculate the desired response.

In the instance of pizza sales in New Jersey, important criteria may include: the state's population; the average pizza consumption per person per week; the number of pizza restaurants in the state, and so on. Errors tend to cancel one other out throughout the factor estimation process, with some too high estimates being offset by extremely low estimations. After estimating these factors, the desired answer can often be calculated with surprising accuracy, i.e., within ten percent of the correct answer.

Until recently, business instructors who wanted to teach their students how to utilise Fermi questions found it difficult to obtain published questions about the business environment. Fermi issues and their solutions frequently involved physics, chemistry, biology, and other "hard science." However, this may be changing as consulting firms and corporations such as Google and Microsoft are asking applicants to try to answer Fermi questions as part of the job interview process, with the goal of identifying creative thinkers; thus, it appears that the time has come for a greater emphasis on the application of the Fermi technique to business.

FERMI ESTIMATION IN JOB MARKET

The job market is the market in which companies look for workers and workers look for work. The job market is not so much a geographical location as it is a concept that depicts the competitiveness and interplay of many labour forces. It is sometimes referred to as the labour market. The job market has seen significant turmoil in the previous two years, owing mostly to the pandemic, and the situation is unlikely to improve in 2022.

Because the employment market is so dynamic, it's more crucial than ever to keep up with significant developments and trends. That is true not only if you are a recruiter or an employer; if you are a jobseeker, staying on top of things is your best chance of securing the best position.

A fermi estimate can help us understand this dynamic market by asking the following questions:

- What are the top skills companies are looking for?

- What is the most desired experience level in the industry?
- What are the companies that are actively offering jobs in this field?
- What are the locations that have more openings?
- What is the avg salary offered by a particular type of job in industry?

These breakups are helpful for making any decision by an individual who is willing to join a particular industry or an organization to understand how much competition they may face if they enter a particular industry.

DATA COLLECTION

The method of gathering, measuring, and evaluating correct insights for study using established approved techniques is characterised as data collection. On the basis of the facts gathered, a researcher might assess their hypothesis. In most situations, regardless of the subject of study, data gathering is the first and most significant stage. Depending on the information requested, the approach to data gathering differs for different topics of research. To begin, while collecting data, we should emphasise that empirical data serves as the foundation for both commonsense and data-driven job segmentation since it gives a solid foundation to work from. This information allows us to construct task segments as well as provide a comprehensive image and explanation of these segments. Although data is a valuable asset for every organization, it does not serve any purpose until analyzed or processed to get the desired results.

- Data Collection is one of the very crucial and difficult steps of data analysis.
- This part can be performed in various ways. For instance, from Kaggle, website or industry.
- Here, the data is scrapped from a website called Naukri.com. It will be later used to perform market segmentation.

- This report is focused on analyzing Machine Learning jobs in Indian market and draw valuable insights. The data is extracted from leading website Naukri.com for the year 2022 till March, **IT CONSISTS OF 3878 RECORDS**.

  We also did a bit of manual data collection and added many features in our dataset.
  By performing web scraping we have stored the data into 5 dictionaries which are **“roles”, “companies”, “locations”, “experience** “, and **“skills”.**

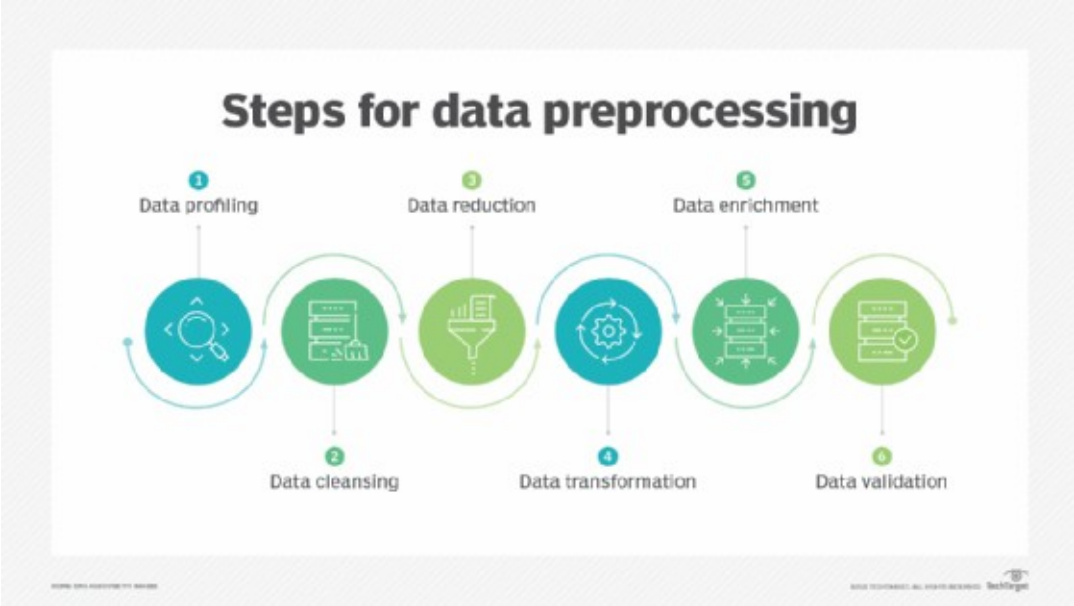
- Finally, once this is extracted from the website it is converted into the following data frame.

The code that we have used to obtain these columns is following:

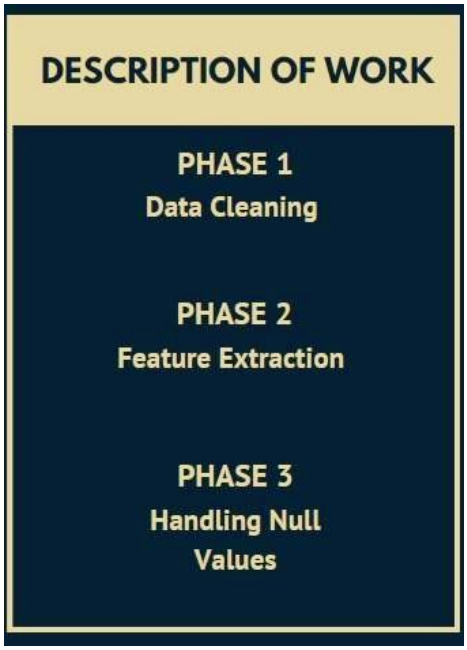
```
#create an instance of browser
driver = webdriver.Chrome(ChromeDriverManager().install())

#creating a dictionary for storing the information after scraping
jobs={"roles":[],
     "companies":[],
     "locations":[],
     "experience":[],
     "skills":[]}

#we will iterate over first 50 pages; each page contains 20 results
#for each job we will scrape the role,company, location, experience, key skills.
for i in range(50):
    driver.get("https://www.naukri.com/data-scientist-jobs-{}".format(i))
    time.sleep(3)
    lst=driver.find_elements_by_css_selector(".jobTuple.bgWhite.br4.mb-8")

    for job in lst:
        driver.implicitly_wait(10)
        role=job.find_element_by_css_selector("a.title.fw500.ellipsis").text
        company=job.find_element_by_css_selector("a.subTitle.ellipsis.fleft").text
        location=job.find_element_by_css_selector(".fleft.grey-text.br2.placeHolderLi.location").text
        exp=job.find_element_by_css_selector(".fleft.grey-text.br2.placeHolderLi.experience").text
        skills=job.find_element_by_css_selector(".tags.has-description").text
        jobs["roles"].append(role)
        jobs["companies"].append(company)
        jobs["locations"].append(location)
        jobs["experience"].append(exp)
        jobs["skills"].append(skills)

data = pd.DataFrame(jobs)
data.to_csv('scrapped.csv')
```

roles	companies	locations	experience	skills
Data Scientist - Lead / Architect - Looking Fo...	Wipro	'Kochi', ' Kolkata', ' Pune', ' Gurgaon', ' Ch...	5-10	'Data Science', 'SAS', 'Python', 'IT Skills', ...
Urgent Requirement || Data Scientist || Noida	HCL	'Noida', 'Delhi'	3-8	'IT Skills', 'Python', 'Machine Learning', 'NL...
Global Tax Automation & Operations - Data Scie...	Dell	'Bangalore'	3-5	'Artificial Intelligence', 'Data Science', 'Da...
Data Analyst / Data Scientist / Business Analy...	GABA Consultancy services	'Noida', ' New Delhi', 'Delhi'	0-0	'O2C', 'fresher data analyst', 'Power Bi', 'Ba...
Technical Architect/ Data Scientist	DMI Innovations Pvt. Ltd	'Noida', ' Pune', ' Chennai', 'Bangalore'	8-13	'IT Skills', 'Python', 'Software Development',...

DATA PRE-PROCESSING



When it comes to building an ML model, the first step is data preparation, which marks the start of the process. We did not use any publicly available datasets on the internet, but instead web scraped data from employment portals. The issue with such a real-world dataset is that it is often inadequate, inconsistent, and lacking in some properties, which we subsequently attempted to add manually. Preprocessing assisted us in cleaning the format, organising the raw data and making it ready to proceed for the segmentation operation.



A regular expression is a pattern in the input text that the regular expression engine seeks to match. A pattern is made up of one or more literals, operators, or structures of one or more characters. For the essential job positions in the Job Title feature, we used Regular Expression. This was done for consistency's sake. Other aspects of the job title were not neglected, but were instead included in new columns. For example, the job title Data Scientist (AWS speciality) was changed to Data Scientist, and we established a new field that specified AWS was required, which we used to clear the "skills" and "location" columns. It contains a large amount of data, which must be cleaned out before we can proceed.

We have cleaned the skills index as it has a lot of errors was there.

```python
print(data.loc[10,'skills'].find("\n"))
print(type(data.loc[10,'skills']))
```

Python

```
1
<class 'str'>
```

```python
for i in range(len(data)):
    k = str(data.loc[i,'skills'])
```

Python

```python
data["skill index"] = data["skills"].str.find("\n")
```

Python

```python
type(data.loc[10,"skill index"])
```

Python

```
numpy.float64
```

Cleaned the skills data again the expressions that was in web scrapped has been cleaned.

```python
for i in range(len(data)):
    k = data.loc[i,'skills']
    if type(k) == str and k.find('\n')!=1:
        lis = k.split('\n')
        lis = str(lis)
        data.loc[i,'skills'] = lis
```

Python

```python
remove_sqrbrct('skills')
```

Python

With the help of Regular expression, we remove extra strings and characters from the given data.

```python
def remove_sqrbrct(string):
    for i in range(len(data)):
        k = data.loc[i,string]
        k=str(k)
        data.loc[i,string] = k
    data[string] = data[string].str.replace("^\[|.\]|$","")
# data.head()
```

Python

```python
remove_sqrbrct('locations')
```

Python

Experience preprocessing

```python
for i in range(len(data)):
    k = data.loc[i,'experience']
    if type(k) == str:
        data.loc[i,'experience'] = k[:-3]
    elif type(k) == float:
        data.loc[i,'experience'] = None
```

Python

preprocessing locations

```python
k = data['locations']
locat = list(k)
filtered_locat = []
for i in locat:
    if type(i) == str:
        all_elem = re.split(r",",i)
        for k in range(len(all_elem)):
            if all_elem[k].find('/') != -1: # uses only 1st value if string has /
                x = all_elem[k].split('/')
                all_elem[k] = x[0]
        if all_elem[k].find('(') != -1: #removes string if it contains ()
            x = all_elem[k].split('(')
            all_elem[k] = x[0]
        if all_elem[k].find(' ') != -1:
            all_elem[k] = all_elem[k].replace(" ","") # removes spaces
        if all_elem[k].find("'") != -1:
            all_elem[k] = all_elem[k].replace("'","") # removes spaces
    elif type(i) == float:
        all_elem = None

    filtered_locat.append(all_elem)
dict = {'location': filtered_locat}
df = pd.DataFrame(dict)
# print(filtered_locat)
```

FEATURE EXTRACTION:

The process of translating raw data into numerical features that can be processed while keeping the information in the original data set is referred to as feature extraction. Many new features were extracted, such as Cloud Requirement, Linux OS Requirement, R or Python Requirements, and SQL Database Requirements. It produces better outcomes than merely applying machine learning to raw data.

One hot encoding method is to encode categorical data variables so that they may be fed into machine learning algorithms to enhance segmentation. One hot encoding is an important component of feature engineering (extraction) for machine learning. For data that has no link to each other, one hot encoding is useful. The order of integers is treated as a significant characteristic by machine learning algorithms. It makes more accurate predictions than single labels. While this is useful in some ordinal scenarios, some input data lacks ranking for category values, which can cause problems with:

Predictions And Poor Performance. That's when one hot encoding saves the day. One hot encoding makes our training data more useful and expressive, and it can be rescaled easily. By using numeric values, we more easily determine a probability for our values.

- Missing completely at random (MCAR)
- Not missing at random (NMAR)
- Missing at random (MAR)

Missing values can arise as a result of a variety of causes, including missing totally at random, missing at random, or missing not at random. All of this might be the consequence of a system failure during data collecting or human mistake during data pre-processing. However, it is critical to deal with missing values before analysing data since neglecting or removing missing values may result in biassed or incorrect analysis or incorrect segmentation. One solution to this problem is to remove the observations with missing data. The missing data might be imputed as a better technique. In other words, we must deduce those missing values from the current data.

```
for i in range(len(data)):
    k = data.loc[i,'skills']
    if type(k) == list:
        for l in range(len(k)):
            r = k[l].lower()
            if r in max_ten:
                k[l] = r
            else:
                k[l] = '_'
```

```
to_drop = []
for i  in range(len(data)):
    k = data.loc[i,'skills']
    if type(k) == float:
        to_drop.append(i)
```

```
data.drop(labels=to_drop,inplace=True)
x= data.iloc[:,3:]
```

Continuing Preprocessing Skills:

```
k = data.loc[1,'skills']
k = k.split("'")
to_remove = list(dict.fromkeys(k))
```

Pyth

```
new_rem = []
for i in range(len(to_remove)):
    if to_remove[i][0].isalnum() is False:
        new_rem.append(to_remove[i])
```

Pyth

Choosing the top skills from our data to target as per our problem statement throught the label skills.

```
to_work = copy.deepcopy(skills_dict)
max_ten = []
for i in range(15):
    k = max(to_work,key=to_work.get)
    max_ten.append(k)
    del to_work[k]
```

Python

```
# print(max_ten)
for i in range(2):
    max_ten.pop(0)
for i in range(len(max_ten)):
    max_ten[i] = max_ten[i].lower()
```

Python

```
print(max_ten)
```

Python

```
['python', 'machine learning', 'it skills', 'data science', 'machine learning', 'computer science', 'artificial intelligence', 'data science', 'r',
'data scientist', 'java', 'sql', 'big data']
```

Here, It is further Data analysis along with visualisation for the the data in order to compute and extract the numbers of cluster for our data.

DATA ANALYSIS AND CLUSTERING

Job Role Preprocessing:

```
df.skills.str.replace('data sciencist','data science')
```

```
0       'data science', 'python', 'it skills', 'artifi...
1              'it skills', 'python', 'machine learning'
2       'artificial intelligence', 'data science', 'it...
4       'it skills', 'python', 'data science', 'machin...
5       'it skills', 'python', 'data science', 'machin...
                              ...
4154           'it skills', 'python', 'machine learning'
4155    'it skills', 'python', 'data science', 'machin...
4156         'data science', 'machine learning', 'python'
4158                                                'python'
4159    'data science', 'machine learning', 'it skills...
Name: skills, Length: 3878, dtype: object
```

```
x = df.iloc[:,5].values
```

```
r = x[0].split(",")
```

```
print(len(r))
```

We use PCA, Created 2 datasets with and two PCA,

```
pca_encoded_data.to_csv('pca_skills.csv')
encoded_df.to_csv('encoded_skills_data.csv')
```

```
encoded_colum = []
for i in range(len(encoder_data)):
    temp = [0]*10
    for j in range(len(encoder_data[i])):
        temp[segment_dict[encoder_data[i][j]]] = 1
    encoded_colum.append(temp)
```

```
len(encoded_colum)
```

```
3878
```

```
df.to_csv('final_till_12_03.csv')
```

```
encoded_df = pd.DataFrame(columns=colum,data = encoded_colum)
```

```
pca = PCA()
principal_comp = pca.fit_transform(encoded_df)
```

Principal Component Analysis, or PCA, is a dimensionality-reduction approach that is frequently used to decrease the dimensionality of big data sets by reducing a large collection of variables into a smaller set that retains the majority of the information in the large set.

Obviously, reducing the number of variables in a data collection reduces accuracy, but the idea in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to study and display, and because machine learning algorithms can analyse data much more easily and quickly without having to deal with superfluous factors.

So to sum up, the idea of PCA is simple — reduce the number of variables of a data set, while preserving as much information as possible.


```
pca_encoded_data.to_csv('pca_skills.csv')
encoded_df.to_csv('encoded_skills_data.csv')
```

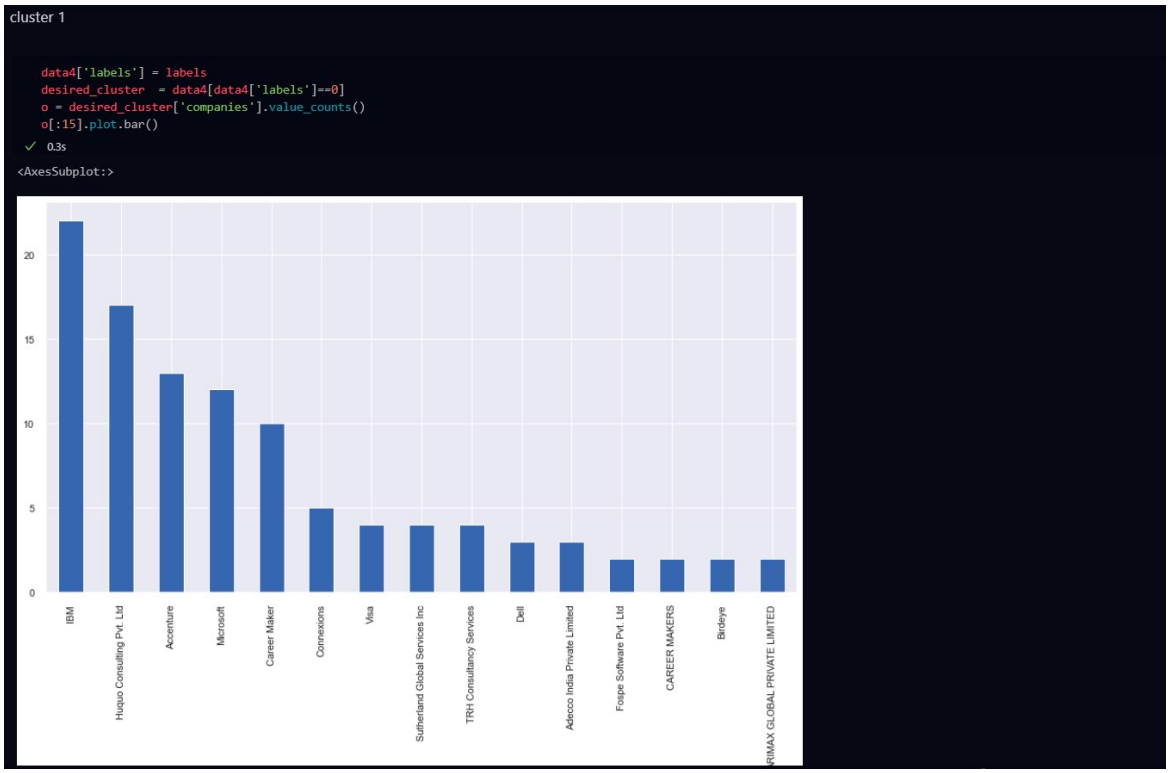
```
for i in range(len(data)):
    k = data.loc[i,'skills']
    if type(k) == list:
        for l in range(len(k)):
            r = k[l].lower()
            if r in max_ten:
                k[l] = r
            else:
                k[l] = '_'
```

```
to_drop = []
for i  in range(len(data)):
    k = data.loc[i,'skills']
    if type(k) == float:
        to_drop.append(i)
```

```
data.drop(labels=to_drop,inplace=True)
x= data.iloc[:,3:]
```

Now we create clusters. And visualising it

CLUSTER 1



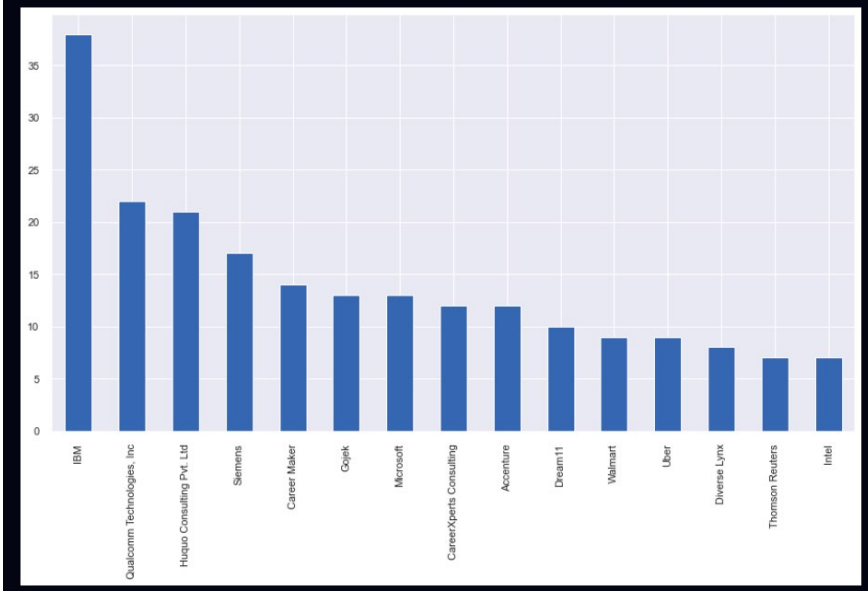
CLUSTER 2

cluster 2

```
desired_cluster  = data4[data4['labels']==1]
o = desired_cluster['companies'].value_counts()
o[:15].plot.bar()
```

✓ 0.4s

```
<AxesSubplot:>
```

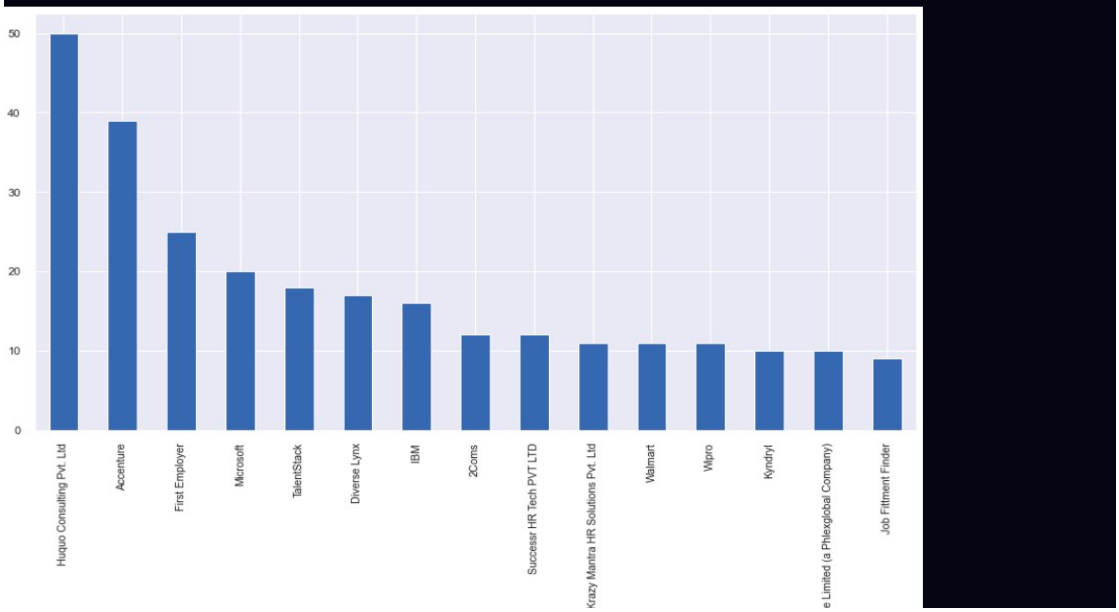


CLUSTER 3

```
desired_cluster  = data4[data4['labels']==2]
o = desired_cluster['companies'].value_counts()
o[:15].plot.bar()
```

✓ 0.3s

```
<AxesSubplot:>
```



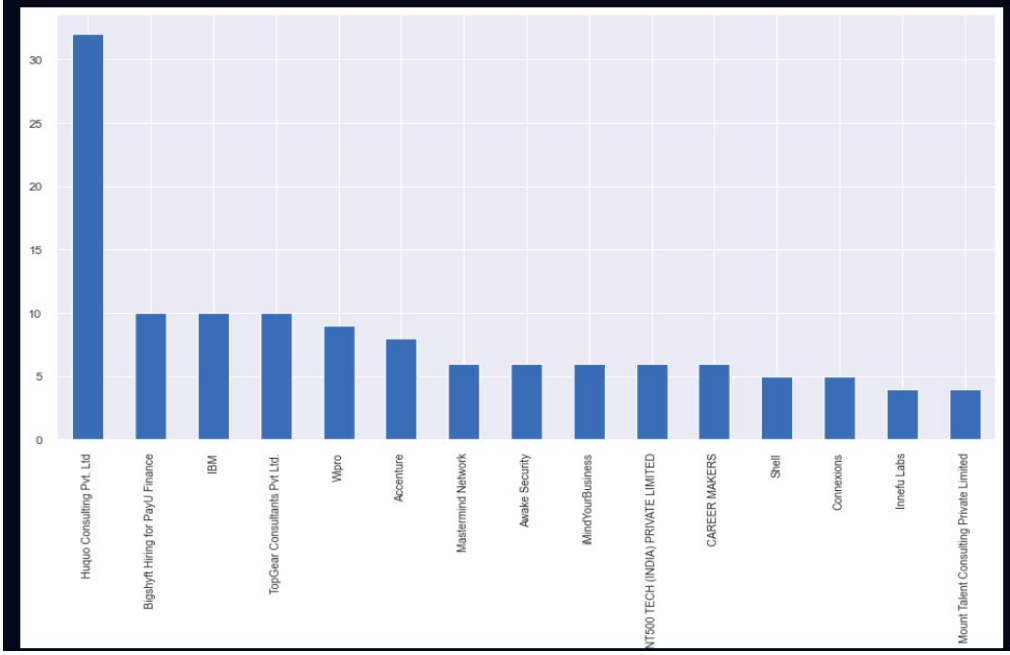
CLUSTER 4

cluster 4

```
desired_cluster  = data4[data4['labels']==3]
o = desired_cluster['companies'].value_counts()
o[:15].plot.bar()
```

✓ 0.4s

```
<AxesSubplot:>
```



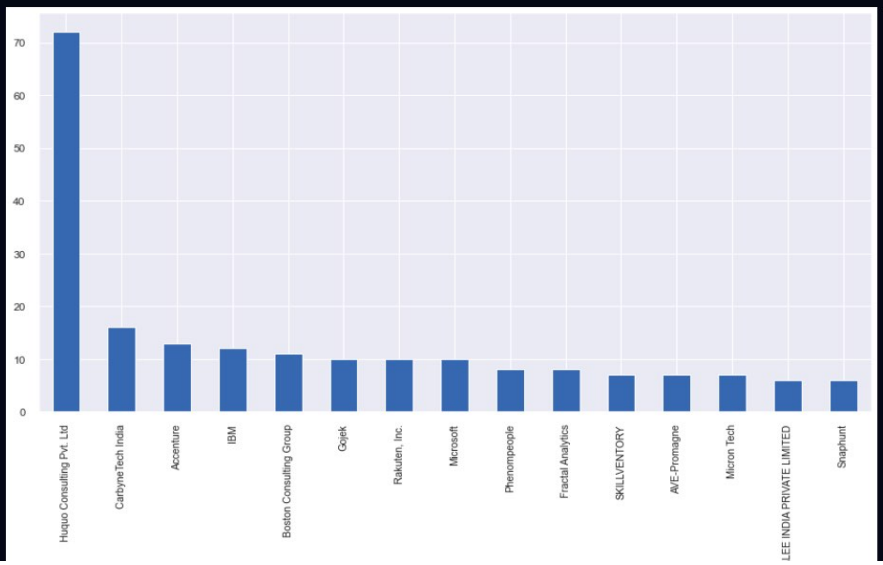
CLUSTER 5

cluster 5

```
desired_cluster  = data4[data4['labels']==4]
o = desired_cluster['companies'].value_counts()
o[:15].plot.bar()
```

✓ 0.5s

```
<AxesSubplot:>
```

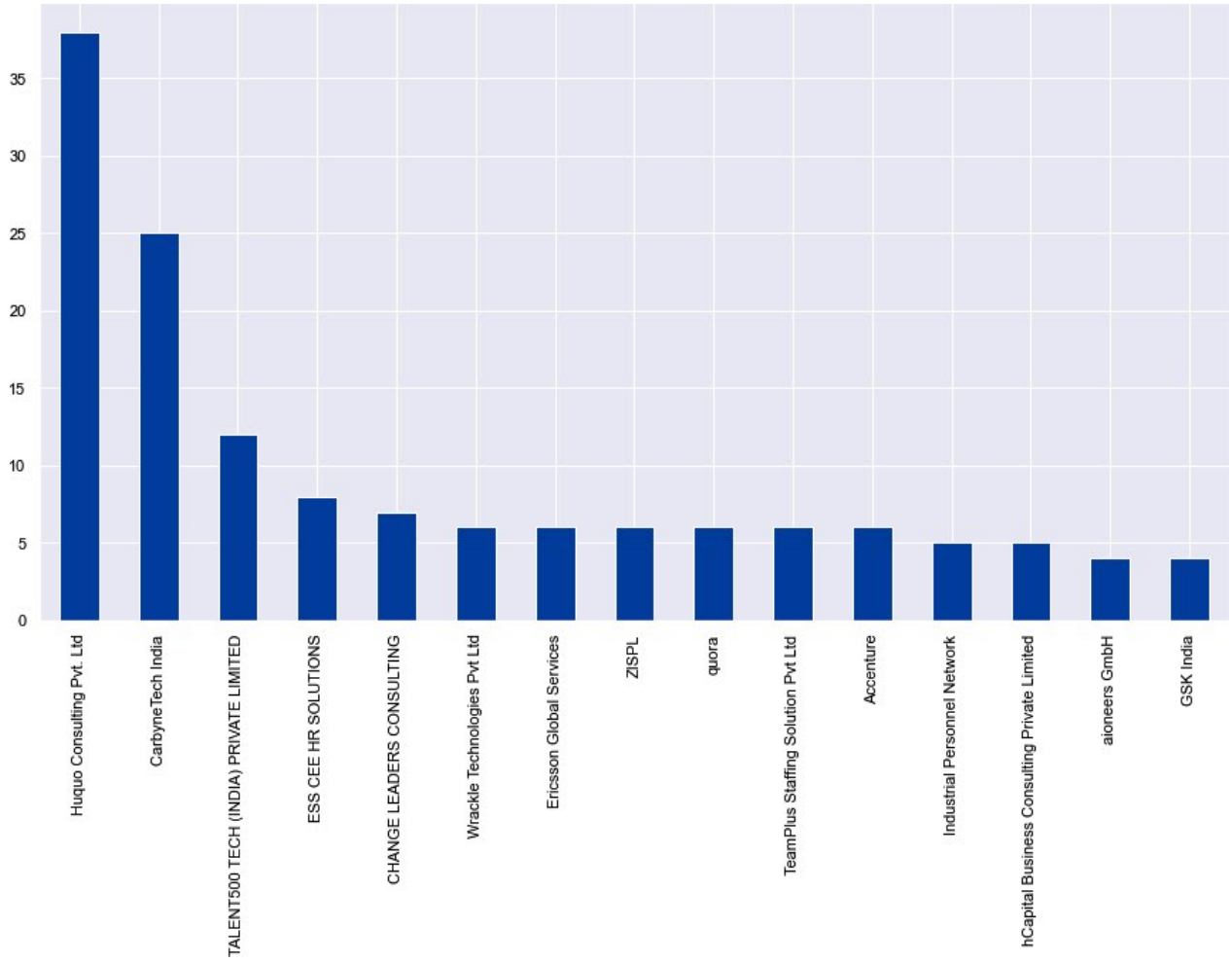


CLUSTER 6

cluster 6

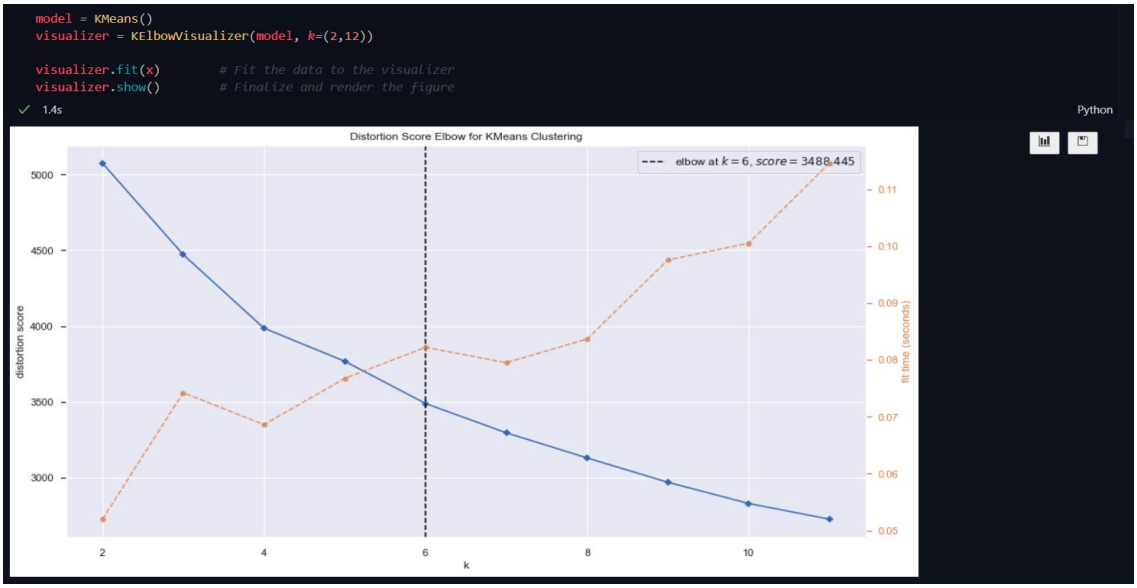
```
In [55]:
1 desired_cluster  = data4[data4['labels']==5]
2 o = desired_cluster['companies'].value_counts()
3 o[:15].plot.bar()
```

PROFILING SEGMENT



SEGMENT EXTRACTION

After preprocessing and producing PCA, we can now go on to building clusters. We used the Elbow approach to determine the best number of clusters, which indicated six clusters.

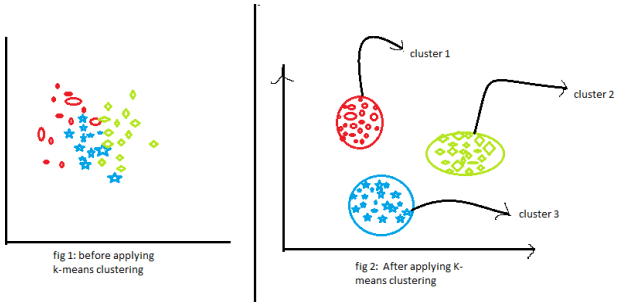


KMeans or KMeans Centroid:

A centroid is an imagined or actual point that represents the cluster's centre. In other words, the K-means algorithm finds k centroids and then assigns every data point to the closest cluster while keeping the centroids as small as feasible.

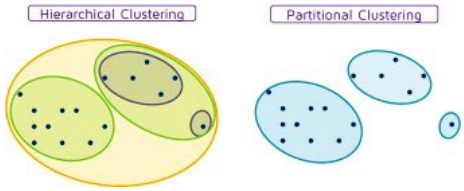
How the K-means algorithm works

To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids



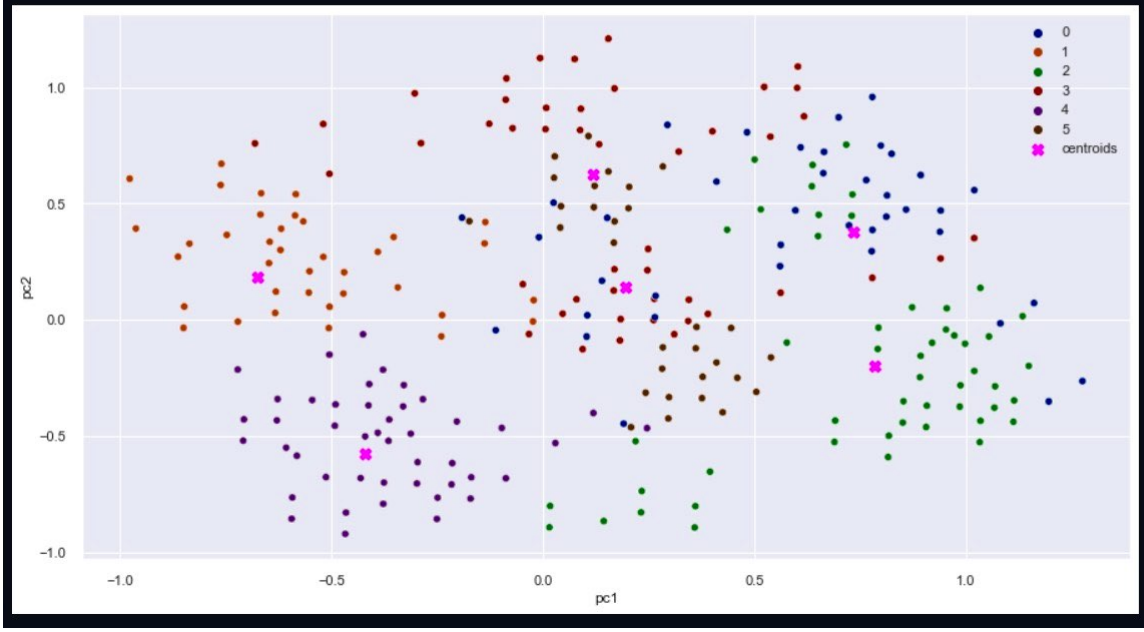
It halts creating and optimizing clusters when either:

1. The centroids have stabilized — there is no change in their values because the clustering has been successful.
2. The defined number of iterations has been achieved.




```
K_means = KMeans(n_clusters=6, n_init=10, verbose=0, random_state=0).fit(x_pca)
```
✓ 0.1s Python

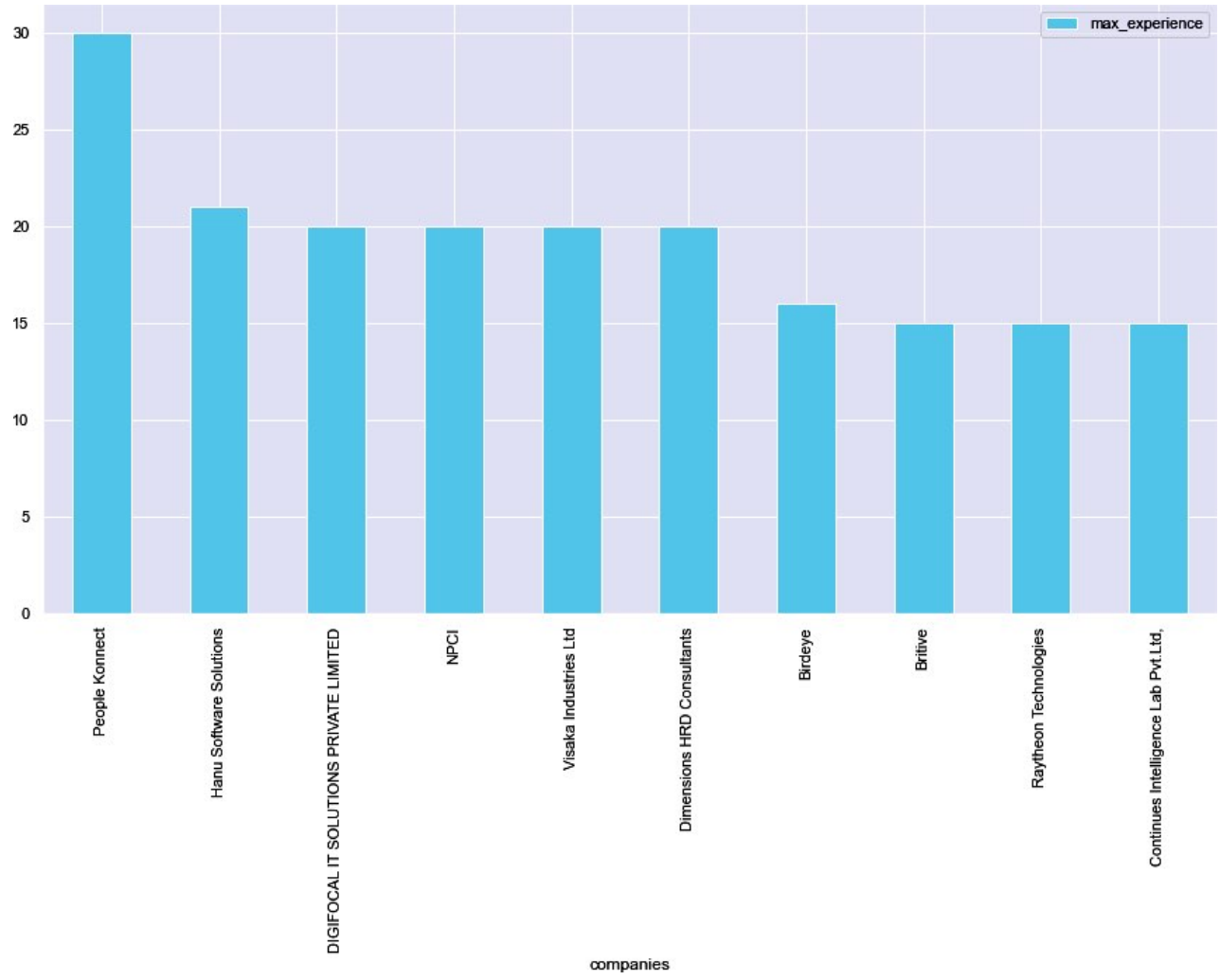
```
sns.set(rc = {'figure.figsize':(15,8)})
sns.scatterplot(data=x_pca_pd, x="pc1", y="pc2", hue=K_means.labels_,palette = "dark")
plt.scatter(K_means.cluster_centers_[:,0], K_means.cluster_centers_[:,1],
            marker="X",color='magenta', s=80, label="centroids")
plt.legend(loc='upper right')
plt.show()
```
✓ 0.5s



And after that we make segment profile

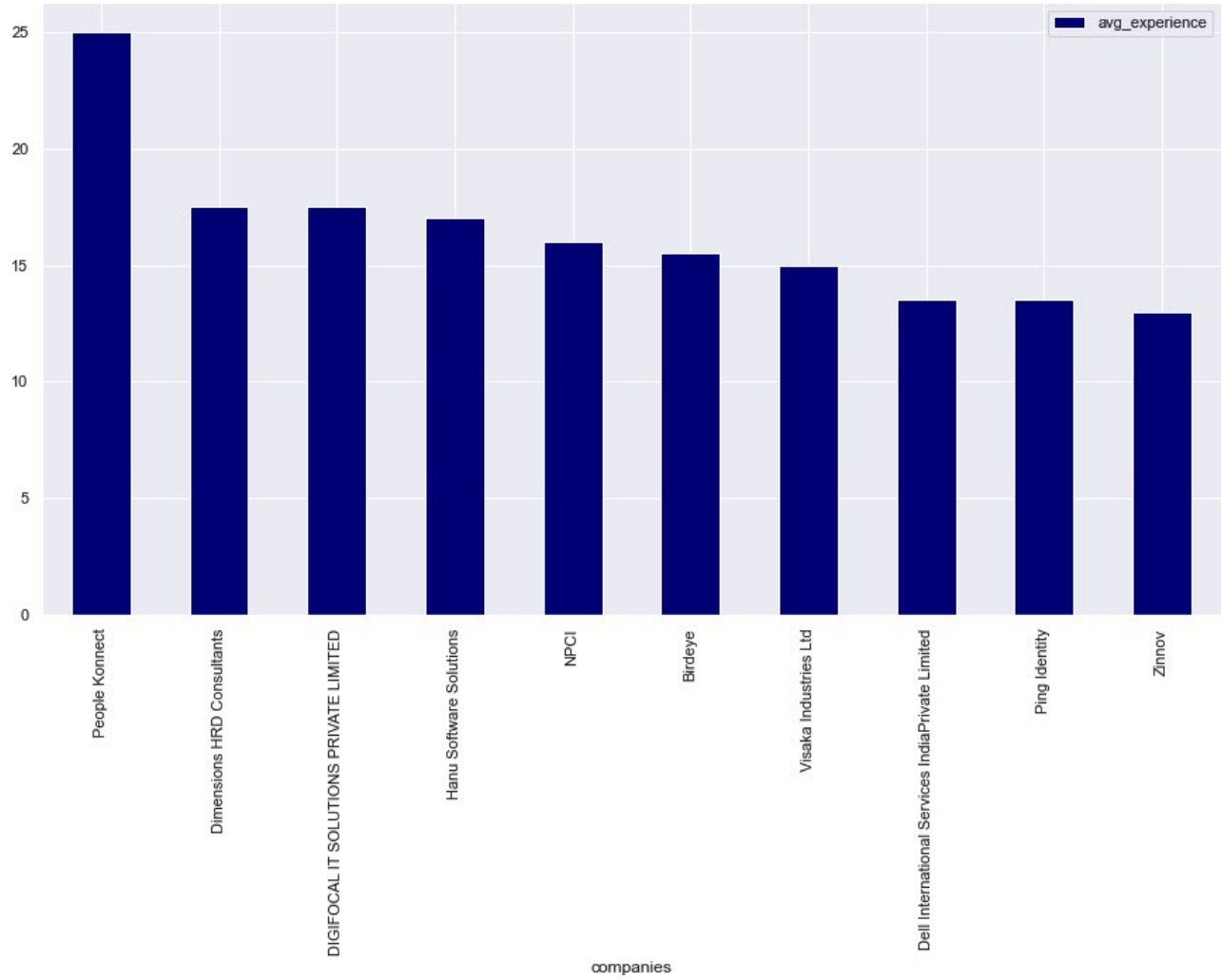
Before Segmentation,

```
1 data4[['max_experience','companies']].groupby(["companies"]).median().sort_values(by='max_experience',ascending=False).he
```




```
1 data4[['min_experience','companies']].groupby(["companies"]).median().sort_values(by='min_experience',ascending=False).he
```

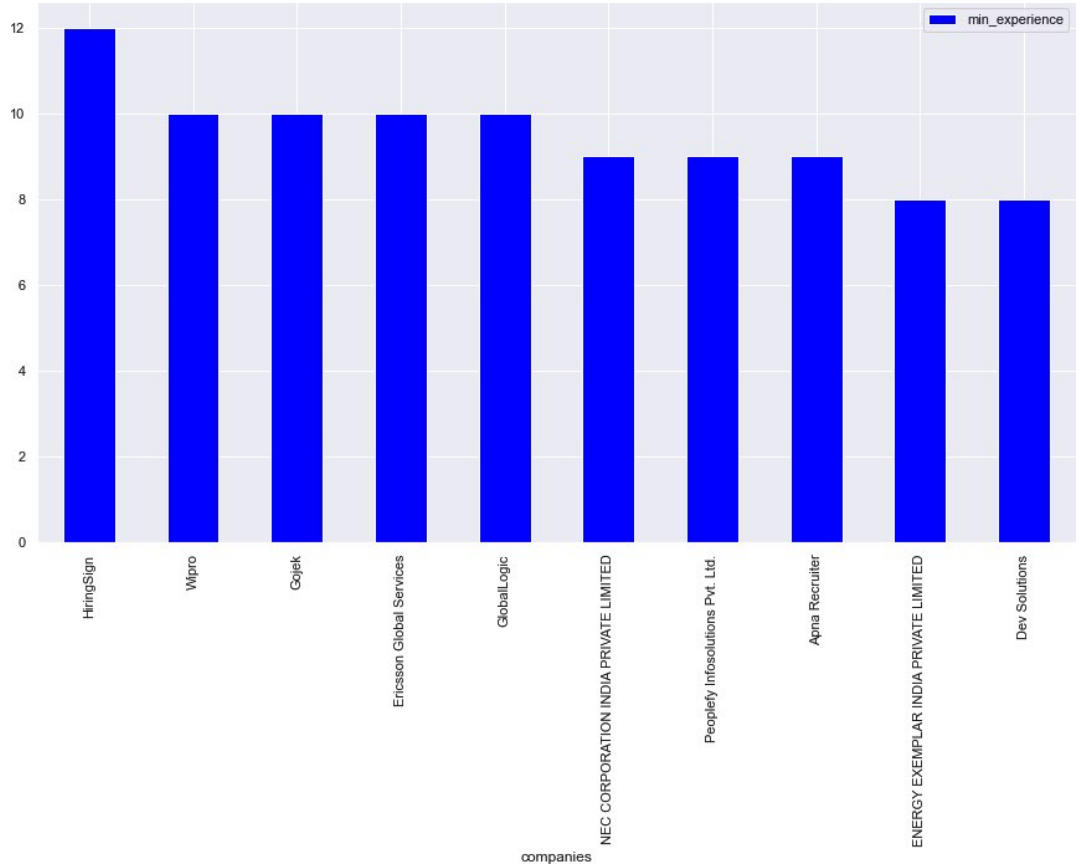
```
<AxesSubplot:xlabel='companies'>
```



After segmentation

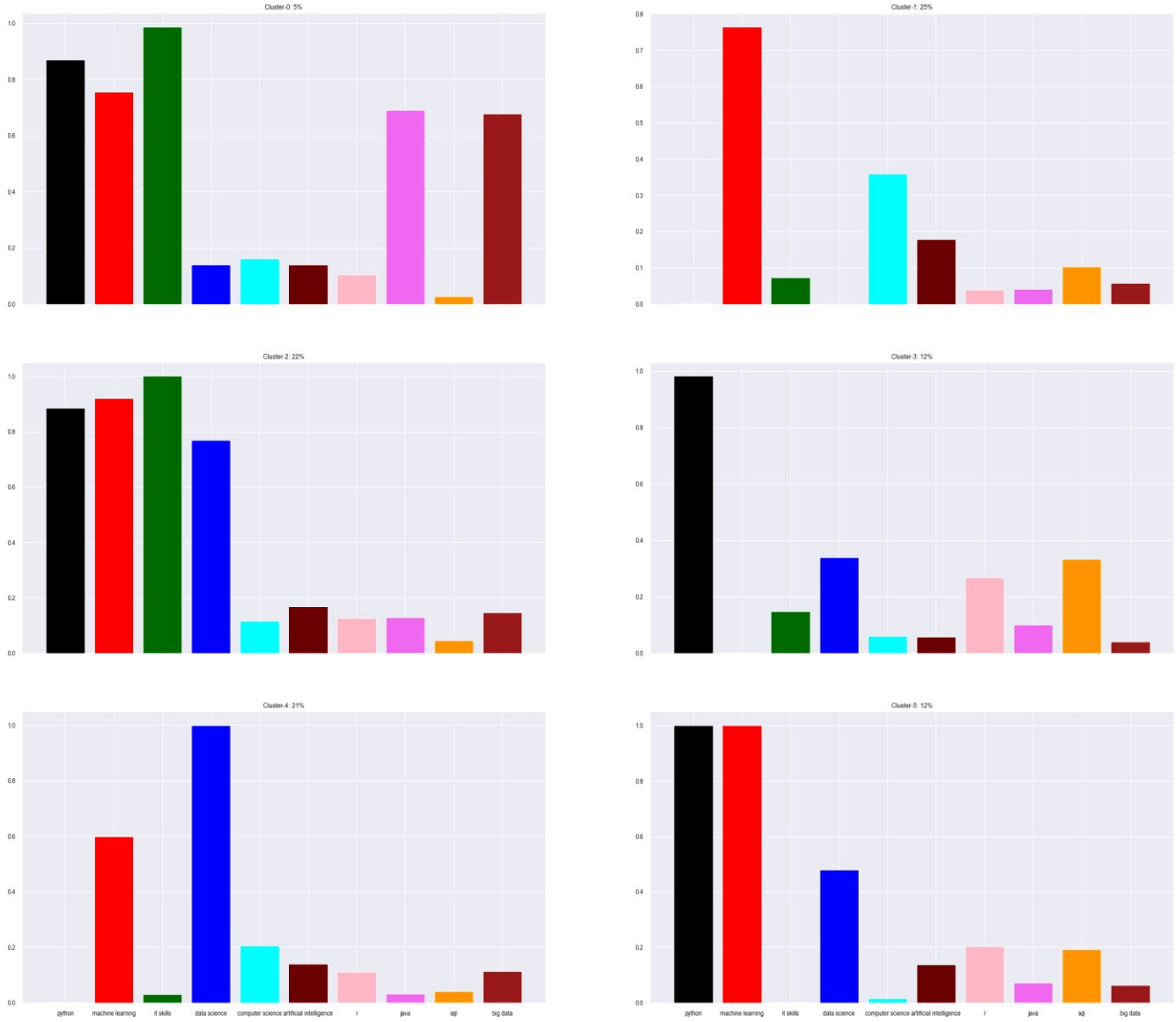
```
1 desired_cluster[['avg_experience','companies']].groupby(["companies"]).median().sort_values(by='avg_experience',ascending
```

```
<AxesSubplot:xlabel='companies'>
```



Profiling Segments:

```
fig, axs = plt.subplots(3, 2, sharex=True,figsize = (40,30))

colorr = ['black', 'red', 'green', 'blue', 'cyan','maroon','pink','violet','orange','brown','yellow']
axs[0, 0].bar(cluster_0_pd.columns,cluster_0_pd.mean(),color = colorr)
axs[0, 1].bar(cluster_1_pd.columns,cluster_1_pd.mean(),color =colorr)
axs[1, 0].bar(cluster_2_pd.columns,cluster_2_pd.mean(),color =colorr)
axs[1, 1].bar(cluster_3_pd.columns,cluster_3_pd.mean(),color =colorr)
axs[2, 0].bar(cluster_4_pd.columns,cluster_4_pd.mean(),color =colorr)
axs[2, 1].bar(cluster_5_pd.columns,cluster_5_pd.mean(),color =colorr)

axs[0, 0].title.set_text("Cluster-0: 5%")
axs[0, 1].title.set_text("Cluster-1: 25%")
axs[1, 0].title.set_text("Cluster-2: 22%")
axs[1, 1].title.set_text("Cluster-3: 12%")
axs[2, 0].title.set_text("Cluster-4: 21%")
axs[2, 1].title.set_text("Cluster-5: 12%")
# fig.tight_layout()
plt.show()
```



For the company's analysis based on experience demanded, it was observed that Wipro, GlobalLogic and Gojek didn't appear in top numbers before the segmentation and appeared after the segmentation was carried out for the minimum and average experience data.

Target segment in job market based on the experience in totality would be the top 10 companies as shown in the above plot.

DESCRIBING POTENTIAL SEGMENTS

Cluster-1: It contains companies which are more inclined towards hiring people with skills which are not oriented towards Data Analysis they prefer skills such as neural networking, computer vision and so on.

Cluster-2: It contains companies which prefer Python skills on Data Science and Machine Learning.

Cluster-4: It generally prefer Data science related skills and does not seem to prefer other languages it inclined towards more of an analysis and visualizing tasks.

The most popular skills demanded by the recruiters were found to be Python, Machine Learning, IT Skills, Data Science etc.

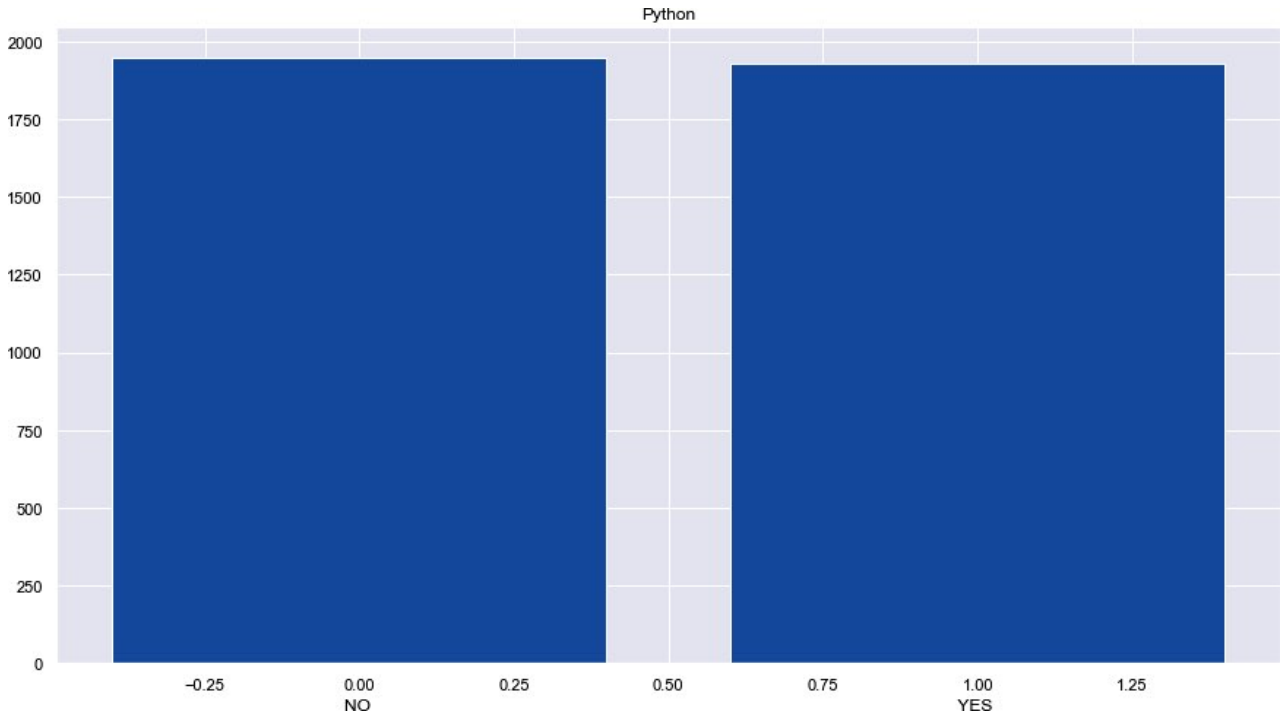
From our dataset we selected these numbers of particular skills and visualized them as follows.

Python

```
skills =list(encoded_data.columns)
skills_dict = dict.fromkeys(skills)
skill_count =[]
for i in range(10):
    skill_count.append(list(encoded_data.iloc[:,i].value_counts()))
incr = 0
for keys in skills_dict.keys():
    skills_dict[keys] = skill_count[incr]
    incr+=1
print(skills_dict)
```

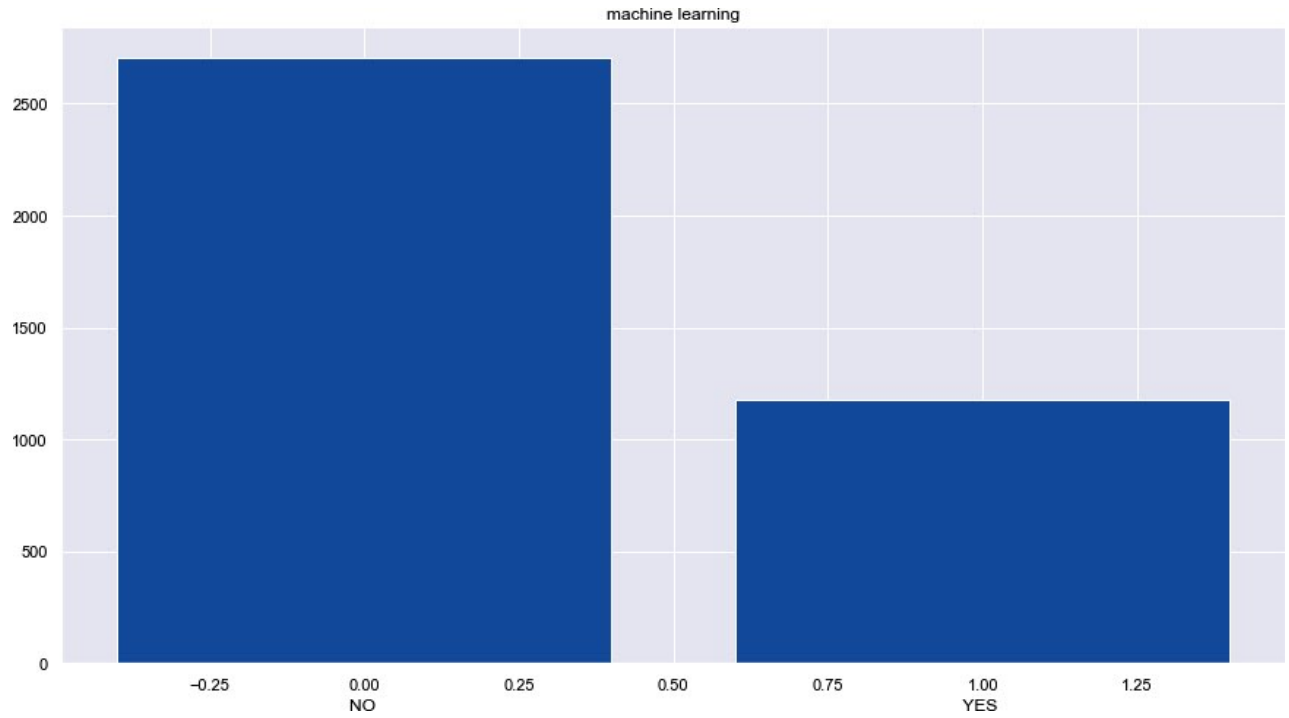
```
{'python': [1947, 1931], 'machine learning': [2704, 1174], 'it skills': [2618, 1260], 'data science': [1955, 1923], 'comput
er science': [3184, 694], 'artificial intelligence': [3313, 565], 'r': [3380, 498], 'java': [3455, 423], 'sql': [3436, 44
2], 'big data': [3395, 483]}
```

```
plt.bar(data='python',height=skills_dict['python'],x = [0,1])
plt.title('Python')
plt.xlabel('NO
plt.show()

```



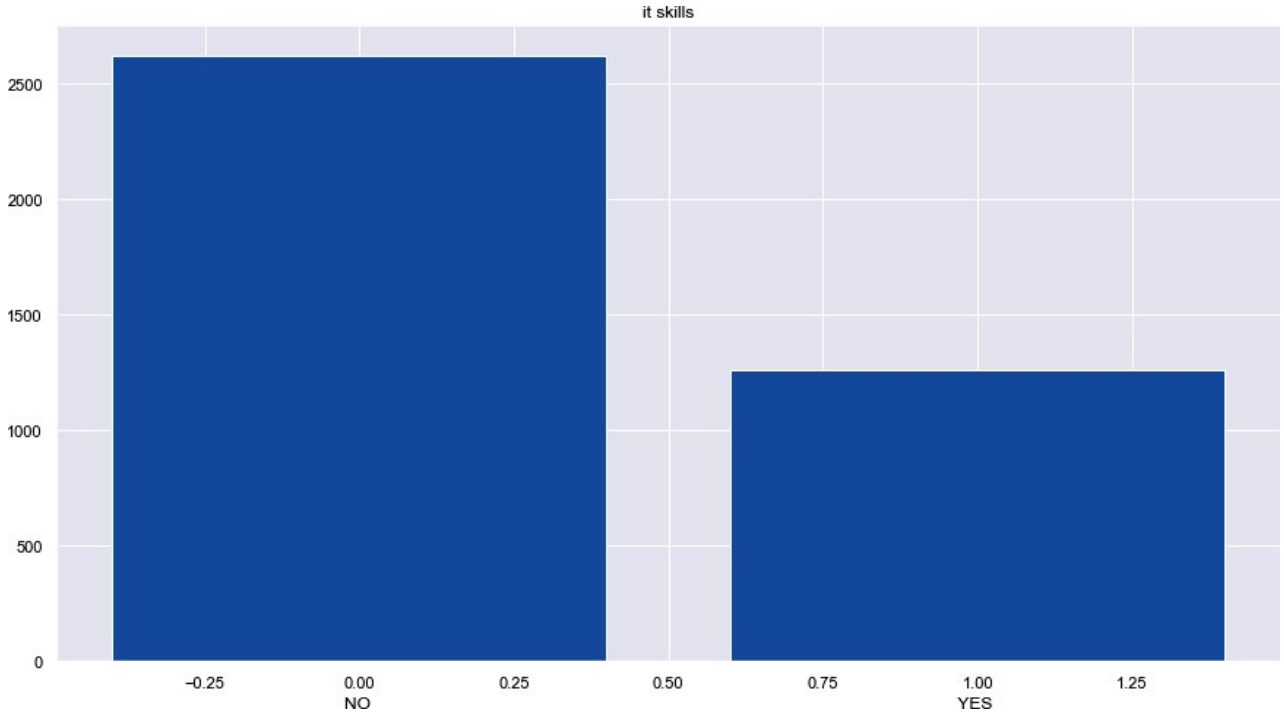
Machine Learning

```
plt.bar(data='machine learning',height=skills_dict['machine learning'],x = [0,1])
plt.title('machine learning')
plt.xlabel('NO
plt.show()
```



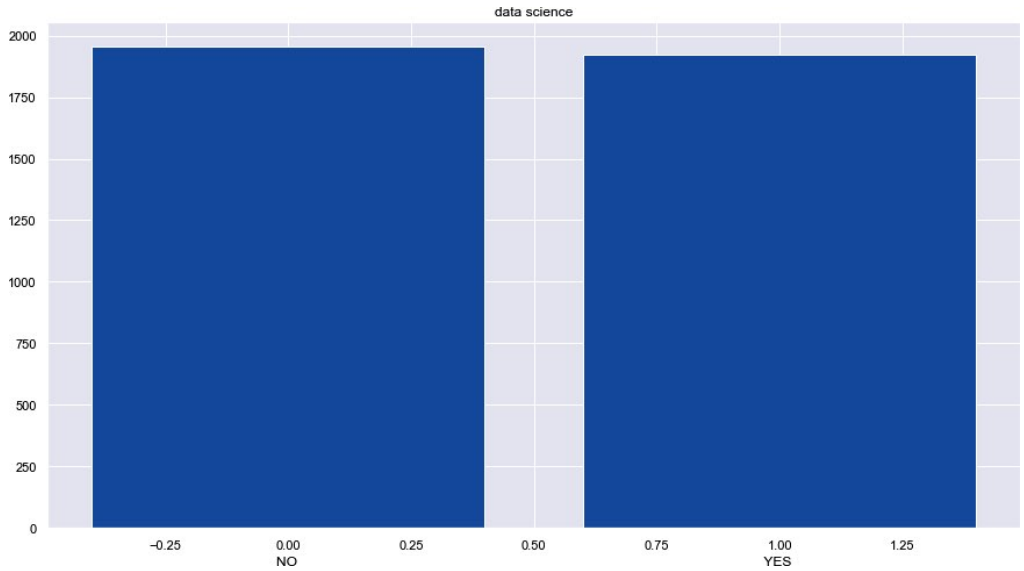
IT Skills

```
plt.bar(data='it skills',height=skills_dict['it skills'],x = [0,1])
plt.title('it skills')
plt.xlabel('NO
plt.show()

```



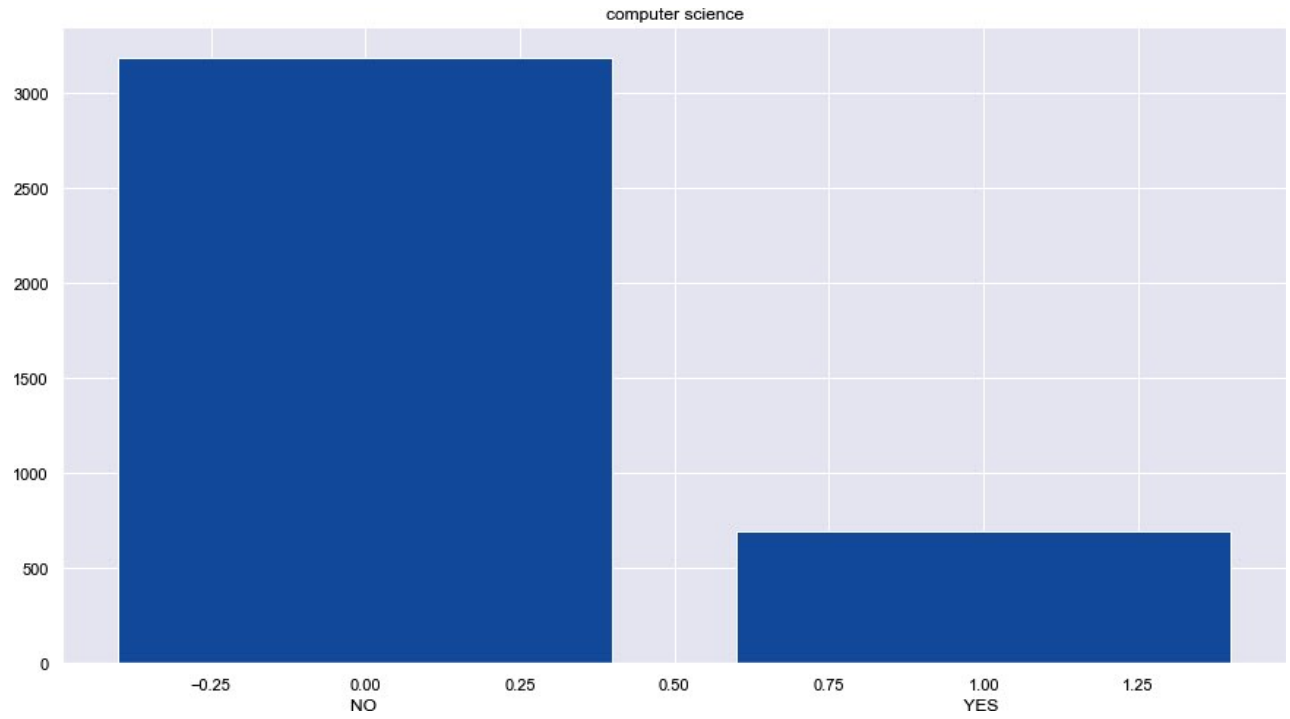
Data Science Skills

```
plt.bar(data='data science',height=skills_dict['data science'],x = [0,1])
plt.title('data science')
plt.xlabel('NO
plt.show()

```



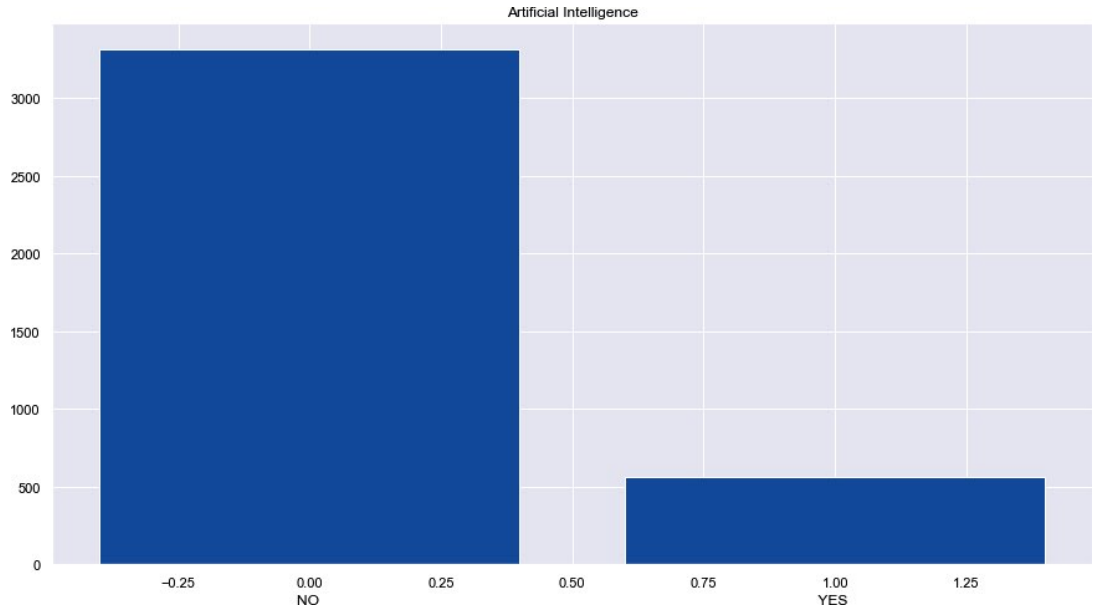
Computer Science

```
plt.bar(data='computer science',height=skills_dict['computer science'],x = [0,1])
plt.title('computer science')
plt.xlabel('NO
plt.show()
```



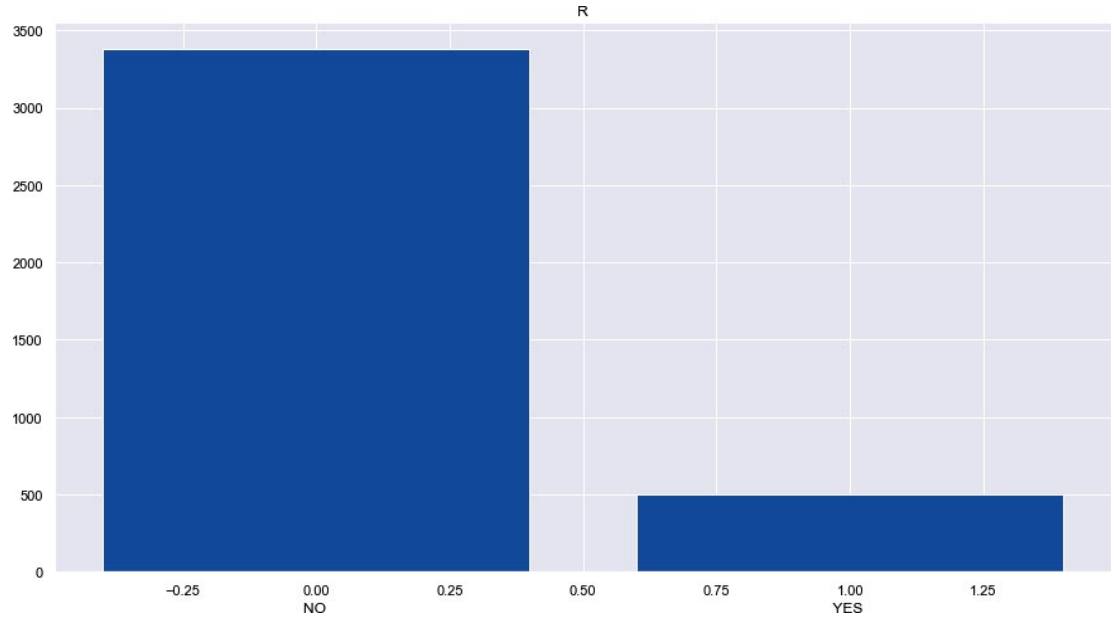
Artificial Intelligence

```
plt.bar(data='artificial intelligence',height=skills_dict['artificial intelligence'],x = [0,1])
plt.title('Artificial Intelligence')
plt.xlabel('NO
plt.show()
```



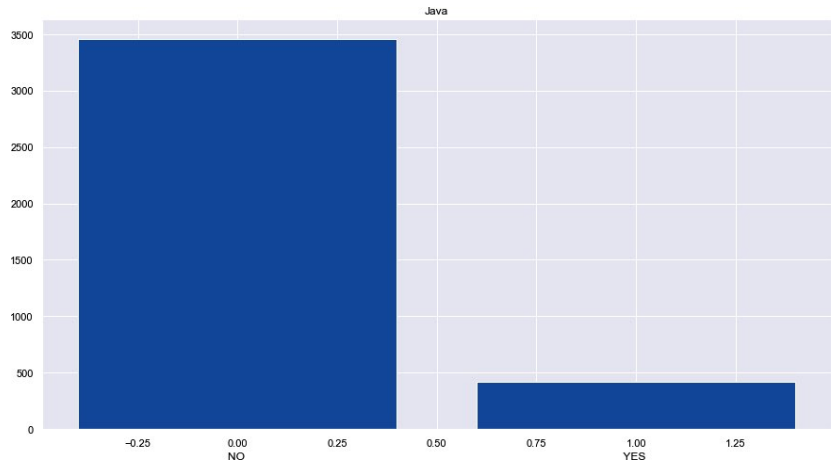
Language R

```
plt.bar(data='r',height=skills_dict['r'],x = [0,1])
plt.title('R')
plt.xlabel('NO
plt.show()
```



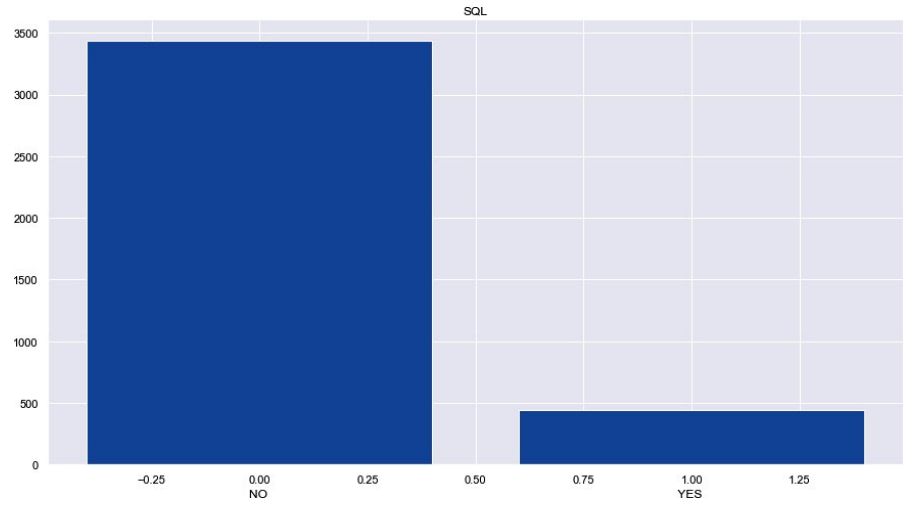
Java

```
plt.bar(data='java',height=skills_dict['java'],x = [0,1])
plt.title('Java')
plt.xlabel('NO
plt.show()

```



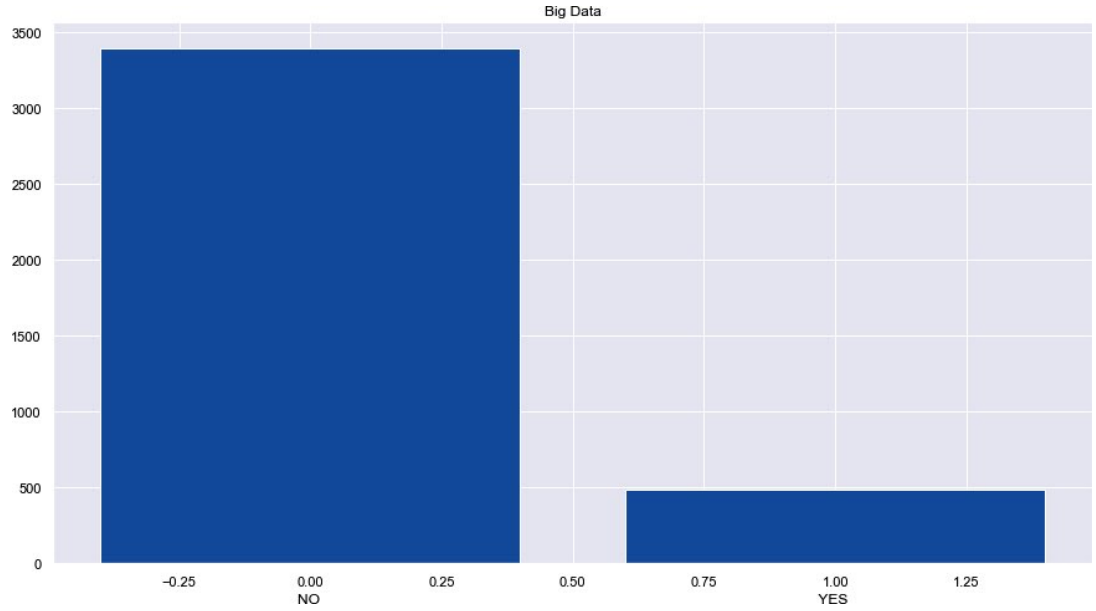
SQL

```
plt.bar(data='sql',height=skills_dict['sql'],x = [0,1])
plt.title('SQL')
plt.xlabel('NO
plt.show()


```



BIG DATA

```
1 plt.bar(data='big data',height=skills_dict['big data'],x = [0,1])
2 plt.title('Big Data')
3 plt.xlabel('NO
4 plt.show()
5
```



CONCLUSION

We conducted job market segmentation using scraped data from Naukri.com (3878) and discovered that Cluster-1 has organisations that are more likely to hire employees with talents that are not geared towards Data Analysis; they prefer skills such as neural networking, computer vision, and so on. Cluster-2 is made up of businesses that favour Python expertise in Data Science and Machine Learning. Cluster-4 It prefers data science-related abilities and does not appear to prefer other languages; it is more oriented toward analysis and visualisation jobs. Data Science and Machine Learning abilities, such as Python and SQL, are in high demand among recruiters.

We were also able to evaluate and clean the data scraped from Naukri.com and extract the strongest segments from the dataset based on all of the breakdown criteria given, such as Technographic Segments (Location), Demographic Segments (Experience), and Behavioral Segments (Skills). For the company's study based on experience requested, it was discovered that Wipro, GlobalLogic, and Gojek did not feature in the top numbers prior to segmentation but did emerge after segmentation for the minimum and average experience data.

Overall, the cluster analysis we conducted would be a wonderful tool for both individuals and employers to locate the needed experience and forthcoming expertise in the Data Science/ML job market, and it would make it easier for any tech company to find a suitable applicant.

REFERENCES

1. https://marutitech.com/artificial-intelligence-and-machine-learning/
2. https://builtin.com/data-science/step-step-explanation-principal-component-analysis
3. https://www.naukri.com/
4. https://selenium-python.readthedocs.io/