Scraping job posting data from Glassdoor and Predicting Salary using Machine Learning

Extracting salary information on data related jobs in order to predict the salaries for certain jobs based on the location, title, and job summary, etc. The project was a real test of data science skills, such as web-scraping, data pre-processing, data analysing, data visualisation, feature engineering, and building multiple regression model to predict the salary and comparing the result which performs best.

Contents

1 Data Collection

Scrapped data from Glassdoor.com, mainly focused on data related fields such as Data Analytics, Data Science/Data Engineering, Business Analytics, Database Management Data, etc.

This is how dataset looks like.

Job Title	Salary Estimate	Job Description	Rating	Company Name	Location	Headquarters	Size	Founded	Type of ownership	Industry	Sector	Revenu
Data Scientist	$53K-91K (Glassdoor est.)	Data Scientist\nLocation: Albuquerque, NM\nEdu...	3.8	Tecolote Research\n3.8	Albuquerque, NM	Goleta, CA	501 to 1000 employees	1973	Company - Private	Aerospace & Defense	Aerospace & Defense	$50 to $1(million (USI
Healthcare Data Scientist	$63K-112K (Glassdoor est.)	What You Will Do:\n\nI. General Summary\n\nThe...	3.4	University of Maryland Medical System\n3.4	Linthicum, MD	Baltimore, MD	10000+ employees	1984	Other Organization	Health Care Services & Hospitals	Health Care	$2 to $5 billi (USI
Data Scientist	$80K-90K (Glassdoor est.)	KnowBe4, Inc. is a high growth information sec...	4.8	KnowBe4\n4.8	Clearwater, FL	Clearwater, FL	501 to 1000 employees	2010	Company - Private	Security Services	Business Services	$100 to $5(million (USI
Data Scientist	$56K-97K (Glassdoor est.)	*Organization and Job ID**\nJob ID: 310709\n\n...	3.8	PNNL\n3.8	Richland, WA	Richland, WA	1001 to 5000 employees	1965	Government	Energy	Oil, Gas, Energy & Utilities	$500 million to 1 billi (USI
Data Scientist	$86K-143K (Glassdoor est.)	Data Scientist\nAffinity Solutions / Marketing...	2.9	Affinity Solutions\n2.9	New York, NY	New York, NY	51 to 200 employees	1998	Company - Private	Advertising & Marketing	Business Services	Unknown / No Applicab

2 Data Cleaning

This Phase is also referred as Data Wrangling. Data Scientists often hate this step because it consumes lot of time to clean the data and produce a good quality data. Here we will understand about data more deeply and prepare it for further analysis.(Lounge,2021)
For this data set we will need to spend good amount of time to clean the data and produce good quality data for the analysis purpose and to build a better machine learning model.

This is how our dataset looks like at the beginning.

	Unnamed: 0	Job Title	Salary Estimate	Job Description	Rating	Company Name	Location	Headquarters	Size	Founded	Type of ownership	Industry	Sec
0	0	Data Scientist	53K−91K (Glassdoor est.)	Data Scientist\nLocation: Albuquerque, NM\nEdu...	3.8	Tecolote Research\n3.8	Albuquerque, NM	Goleta, CA	501 to 1000 employees	1973	Company - Private	Aerospace & Defense	Aerospa & Defe
1	1	Healthcare Data Scientist	63K−112K (Glassdoor est.)	What You Will Do:\n\nI. General Summary\n\nThe...	3.4	University of Maryland Medical System\n3.4	Linthicum, MD	Baltimore, MD	10000+ employees	1984	Other Organization	Health Care Services & Hospitals	He C
2	2	Data Scientist	80K−90K (Glassdoor est.)	KnowBe4, Inc. is a high growth information sec...	4.8	KnowBe4\n4.8	Clearwater, FL	Clearwater, FL	501 to 1000 employees	2010	Company - Private	Security Services	Busin Servi
3	3	Data Scientist	56K−97K (Glassdoor est.)	*Organization and Job ID**\nJob ID: 310709\n\n...	3.8	PNNL\n3.8	Richland, WA	Richland, WA	1001 to 5000 employees	1965	Government	Energy	Oil, G Energ Utili
4	4	Data Scientist	86K−143K (Glassdoor est.)	Data Scientist\nAffinity Solutions / Marketing...	2.9	Affinity Solutions\n2.9	New York, NY	New York, NY	51 to 200 employees	1998	Company - Private	Advertising & Marketing	Busin Servi

1) From the above figure we can see that there is a column name “Unamed 0” which does not contain any information. We will drop the “Unamed 0” column. There are many other issues with this data which we will explore in this section.

	Job Title	Salary Estimate	Job Description	Rating	Company Name	Location	Headquarters	Size	Founded	Type of ownership	Industry
0	Data Scientist	53K−91K (Glassdoor est.)	Data Scientist\nLocation: Albuquerque, NM\nEdu...	3.8	Tecolote Research\n3.8	Albuquerque, NM	Goleta, CA	501 to 1000 employees	1973	Company - Private	Aerospace & Defense
1	Healthcare Data Scientist	63K−112K (Glassdoor est.)	What You Will Do:\n\nI. General Summary\n\nThe...	3.4	University of Maryland Medical System\n3.4	Linthicum, MD	Baltimore, MD	10000+ employees	1984	Other Organization	Health Care Services & Hospitals
2	Data Scientist	80K−90K (Glassdoor est.)	KnowBe4, Inc. is a high growth information sec...	4.8	KnowBe4\n4.8	Clearwater, FL	Clearwater, FL	501 to 1000 employees	2010	Company - Private	Security Services
3	Data Scientist	56K−97K (Glassdoor est.)	*Organization and Job ID**\nJob ID: 310709\n\n...	3.8	PNNL\n3.8	Richland, WA	Richland, WA	1001 to 5000 employees	1965	Government	Energy
4	Data Scientist	86K−143K (Glassdoor est.)	Data Scientist\nAffinity Solutions / Marketing...	2.9	Affinity Solutions\n2.9	New York, NY	New York, NY	51 to 200 employees	1998	Company - Private	Advertising & Marketing

2) We will remove all the rows where the Salary Estimate Feature is NaN which means the salary information is missing. The reason for removing the entire sample is that our target variable is the Salary Estimate and we cannot have any missing value for it to build a model.

```
# Ignoring the NaN values
df = df[df['Salary Estimate'] != '-1']
```

3) Now we have removed all the missing values but as you can see our salary estimate contains string, and that information are not useful in this case.

we have dollar symbol and (glassdoor est.) which are not useful information.

```
df["Salary Estimate"]
```

```
0       $53K-$91K (Glassdoor est.)
1      $63K-$112K (Glassdoor est.)
2       $80K-$90K (Glassdoor est.)
3       $56K-$97K (Glassdoor est.)
4      $86K-$143K (Glassdoor est.)
```

Using the below code we are removing the string (glassdoor est.)

```
# Removing  string from salary feautre
df["Salary Estimate"] = df["Salary Estimate"].apply(lambda x: x.split("(")[0])
```

Salary Estimate also contains some salary Per Hour and String like “Employer Provided Salary” and “K”. We need to remove all of this because they do not add any value. We only need a numeric value i.e., Salary information.

Salary Estimate
17–24 Per Hour
21–34 Per Hour
18–25 Per Hour

Salary Estimate
Employer Provided Salary: 150K–160K
Employer Provided Salary: 120K–145K
Employer Provided Salary: 200K–250K

Using the below code we are removing the unwanted string

```
# Removing  string from salary feautre
df["Salary Estimate"] = df["Salary Estimate"].str.replace("K","").str.replace("$","").str.replace("Employer","").str.replace("Pr
```

Now our salary estimate is looking much better. But there is a small problem, The salary contains a range of intervals, In order to solve this, we will take the average salary to simply the problem.

Salary Estimate
53-91
63-112
80-90
56-97
86-143

Code for getting the average salary.

```
# Taking average Salary
df["Salary Estimate"] = df["Salary Estimate"].apply(lambda x: (float(x.split("-")[0]) + float(x.split("-")[1]))/2 if len(str(x)
```

Now our Salary Estimate feature is looking perfect.

Salary Estimate
72.0
87.5
85.0
76.5
114.5

4) The location feature is in the following format. We will use only city code and remove the city name because both are the same information. And there might be some cases where the spelling of the city name might not be correct due to human error. And this will create 2 unique city names for the same city name which will cause problem for the model. So, we will use city code to be on a safer side.

Location
Albuquerque, NM
Linthicum, MD
Clearwater, FL
Richland, WA

5) Company Name contains some "\n" which is an html tag for the new line. It also contains rating information which is already there in the other column. We will need to remove both of this unwanted string.

Company Name
Tecolote Research\n3.8
University of Maryland Medical System\n3.4
KnowBe4\n4.8

Code for keeping only the company name.

```
df["Company Name"] = df["Company Name"].str.extract('([a-zA-Z ]+)')
```

6) Job titles are very different from company to company. Some might write healthcare Data Scientist or Sr Data Scientist.

Job Title
Data Scientist
Healthcare Data Scientist
Data Scientist
Data Scientist
Data Scientist
...
Sr Scientist, Immuno-Oncology - Oncology
Senior Data Engineer
Project Scientist - Auton Lab.

But we are just interested in the title whether it is data scientist or related to data science. And same goes for Machine Learning Engineer, Big data Engineer, Data Analyst, manager. In order to make them all fall into the same name related to

their domain. we will check if it contains a specific word for example if the title has scientist, then we will just name it to data scientist or if the title contains analyst or analytics then it will change to analyst, following are the names that we are going to use.

data scientist

analyst

analytics

ml

manager

python

big data

other: - If the job is not from any specified name then it'll be changed to "other"

By doing this we are making it easier for our model to easily identify the profession and there will be few professions to work with.

Using the below code, we are performing the above task defined.

```python
def title_encode(x):
    if 'data scientist' in x.lower():
        return 'data scientist'
    elif 'analyst' in x.lower():
        return 'analyst'
    elif 'analytics' in x.lower():
        return 'analyst'
    elif 'machine learning' in x.lower():
        return 'ml'
    elif 'manager' in x.lower():
        return 'manager'
    elif 'python' in x.lower():
        return 'python'
    elif 'big data' in x.lower():
        return "big data"
    else:
        return 'other'
```

3 Feature Engineering

In the phase of Feature Engineering, we use the domain knowledge to extract features from the raw data and we can even create new features. The features can be useful to make the model more robust. The most useful feature engineering approaches are used to change data into a format that a model can comprehend better, as well as to deal with data and pattern abnormalities. (Sivarajah, 2020)

1) We have described in the data cleaning step that the Salary estimate contains per hour information. But we want our model to know that it is for Per hour. To do this we are creating a new feature called hour when it is 1 then it means the salary is for Per Hour and 0 means the salary is for Per annum.

hour
0
0
0
0
0
...
0
0

Following code performs the above task

```
# Feature Creation
df["hour"] = df["Salary Estimate"].apply(lambda x: 1 if "hour" in x.lower() else 0)
```

2) We have a feature called “Founded” which tells when the company was founded. We are creating a new feature called year which tells how old the company is.

year
48
37
11
56
23

Following code performs the above task .

```
# Feature Creation
df["year"] = df["Founded"].apply(lambda x: 2021-x if x!=-1 else x)
```

3) We are creating head_state feature combining the job location state and the companies headquarter state, if the job location state and the companies headquarter state are same then the value is 1 else it is 0. In this way model have additional information whether the job is in headquarter state. Usually, the jobs which are in headquarter state have higher salary.

head_state
0
0
1
1
1

Following code performs the above task .

```
# Feature Creation
df['head_state'] = np.where(df['Headquarters']==df["Location"], 1, 0)
```

4) We have the job description feature using that feature we are creating new feature called jon_len which is the length of the job description. Maybe there is positive correlation between job_len and salary. As the salary increase the job_len might also increase.

Job_len
2517
4738
3427
3840
2708
...
6105
6093
3049
1606
3636

Following code performs the above task .

```
df["Job_len"] = df["Job Description"].apply(lambda x: len(x))
```

5) Again, using Job Description feature we are creating multiple new features python, aws, azure, visualization, analysis, analysis, and bi.
If the job description contains any of those words, then the value is set to 1 or else it is set to 0 for all the features.

python	aws	azure	visualization	analysis	excel	bi	j
1	0	0	1	1	1	1	
1	0	0	1	1	0	1	
1	0	0	1	1	1	1	

Following code performs the above task .

```
df['python'] = df["Job Description"].apply(lambda x: 1 if "python" in x.lower() else 0)
df['aws'] = df["Job Description"].apply(lambda x: 1 if "aws" in x.lower() else 0)
df['azure'] = df["Job Description"].apply(lambda x: 1 if "azure" in x.lower() else 0)
df['visualization'] = df["Job Description"].apply(lambda x: 1 if "visualization" in x.lower() else 0)
df['analysis'] = df["Job Description"].apply(lambda x: 1 if "analysis" in x.lower() else 0)
df['excel'] = df["Job Description"].apply(lambda x: 1 if "excel" in x.lower() else 0)
df['bi'] = df["Job Description"].apply(lambda x: 1 if "bi" in x.lower() else 0)
```

6) Using Job Title feature we are creating one addition feature called experience.

If the Job title contains keywords like senior or manager or sr or director or lead then it will be set to senior.

If the Job title contains keywords like junio or graduate or jr or intern then it will be set to junior.

Else it will be set to other.

This will help the model to tell whether the employee is senior, or junior, or inter

expeirence
other
other
other
other
other
...
senior
senior
other
senior

Following code performs the above task .

```
def exp(x):
    if "senior" in x.lower() or "manager" in x.lower() or "director" in x.lower() or "lead" in x.lower() or "sr" in x.lower():
        return "senior"
    elif "junior" in x.lower() or "graduate" in x.lower() or "intern" in x.lower() or "jr" in x.lower() :
        return "junior"
    else:
        return "other"
```

4 Exploratory Data Analysis

Exploratory Data Analysis is one of the important steps in the data science life cycle. In this step, we gain information about the data in depth. We can also find the independent variables which are important to predict the dependent variable. There are no rules in the Exploratory Data Analysis, we can explore the data as we like. Using various visualizations like scatter plot, bar plot, histogram, etc, we can explain the data.(Sanat,2019)

1) Let us start with the basic statistics like mean, median, mode standard deviation, and count.

```
df.describe()
```

	Salary Estimate	Rating	Founded	hour	year	head_state	Job_len	python	aws	azure	visualization	analysis
count	742.000000	742.000000	742.000000	742.000000	742.000000	742.000000	742.000000	742.000000	742.000000	742.000000	742.000000	742.000000
mean	101.434636	3.618868	1837.154987	0.032345	47.524259	0.557951	3828.919137	0.528302	0.237197	0.072776	0.257412	0.595687
std	37.546122	0.801210	497.183763	0.177034	53.839080	0.496965	1509.298887	0.499535	0.425651	0.259944	0.437503	0.491090
min	15.500000	-1.000000	-1.000000	0.000000	-1.000000	0.000000	407.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	73.500000	3.300000	1939.000000	0.000000	12.000000	0.000000	2761.250000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	97.500000	3.700000	1988.000000	0.000000	25.000000	1.000000	3699.500000	1.000000	0.000000	0.000000	0.000000	1.000000
75%	122.500000	4.000000	2007.000000	0.000000	60.000000	1.000000	4700.750000	1.000000	0.000000	0.000000	1.000000	1.000000
max	254.000000	5.000000	2019.000000	1.000000	277.000000	1.000000	9956.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Now Let us see all the Column Names: -

- Job Title
- Salary Estimate
- Job Description
- Rating
- Company Name
- Location
- Headquarters
- Size
- Founded
- Type of ownership
- Industry
- Sector
- Revenue
- Competitors
- hour
- year
- code
- head_state
- Job_len
- python
- aws
- azure
- visualization
- analysis
- excel
- bi
- job_focus
- experience

2) Count of Categorical Features: -

There are 14 Categorical Features in our dataset namely

- Revenue
- Industry
- experience
- Competitors
- Headquarters
- Type of ownership
- Company Name
- code
- Location
- Size,
- job Description
- job_focus
- Sector
- Job Title

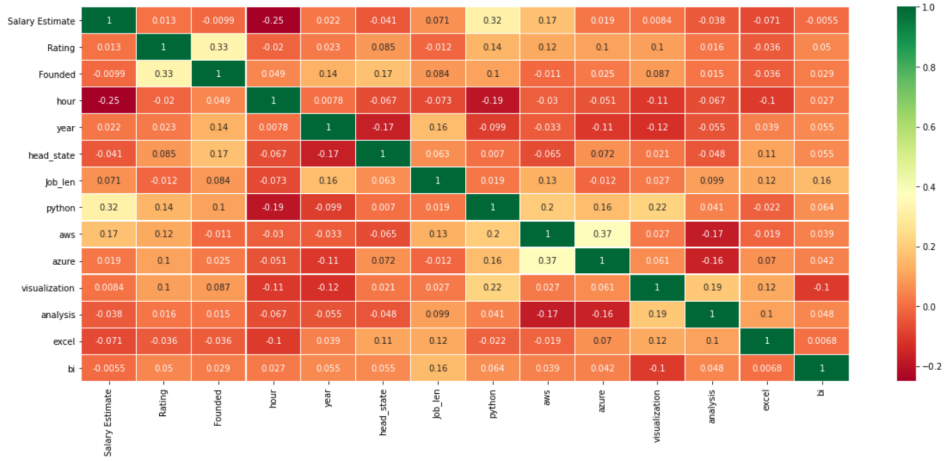
3) Count of Numerical Features: -

There are 14 Numerical Features in our dataset namely

- Salary Estimate
- Rating
- Founded
- hour
- year
- head_state
- Job_len
- python
- aws
- azure
- visualization
- analysis
- excel
- bi

4) Correlation Matrix

The following table shows us the relationship between all the features. Since we are using Linear models, it is important to see whether our data has multicollinearity problem.



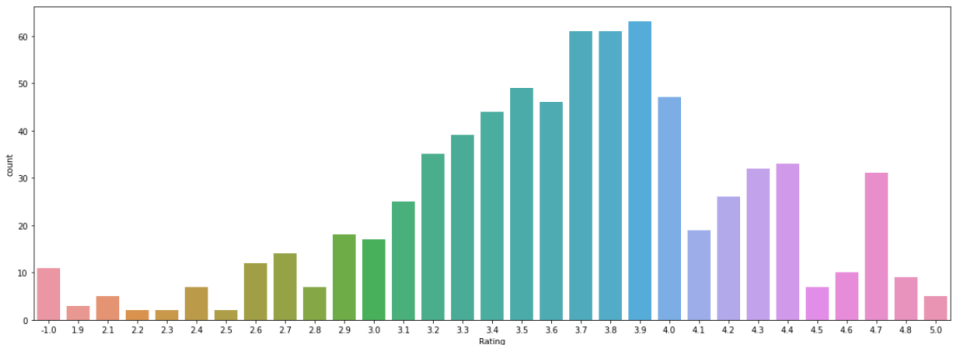
As we can see our data does not have any multicollinearity problem.

5) Exploring each feature in deep

i) **Job Description: -** Job Description has all the information that is required for the job, WordCloud would be the best choice in this case to see what are the most occurring skills that is common in all.

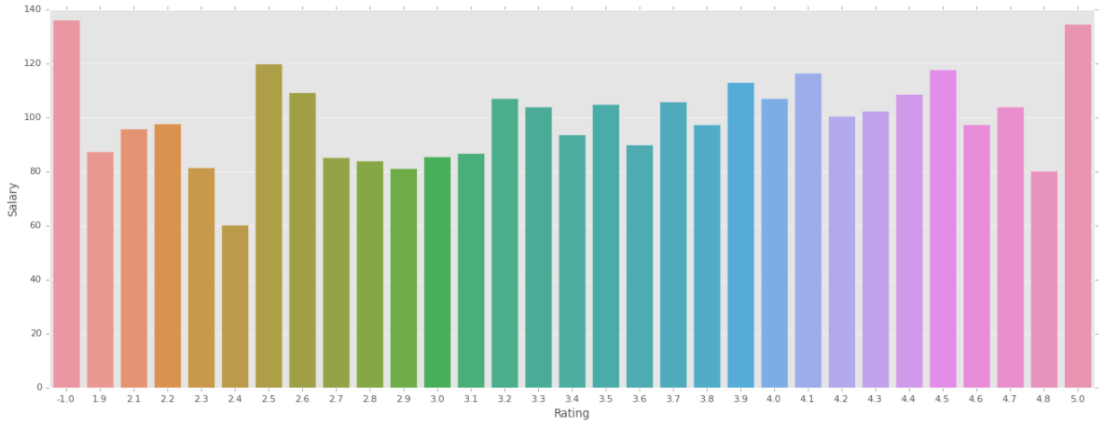
Observation: -
These are all the words that are most common in all. A person who is trying to make a CV can use this to find out what are the demanding skill that is required to be a data scientist. All Jobs related to Data Science require some sort of Experience.

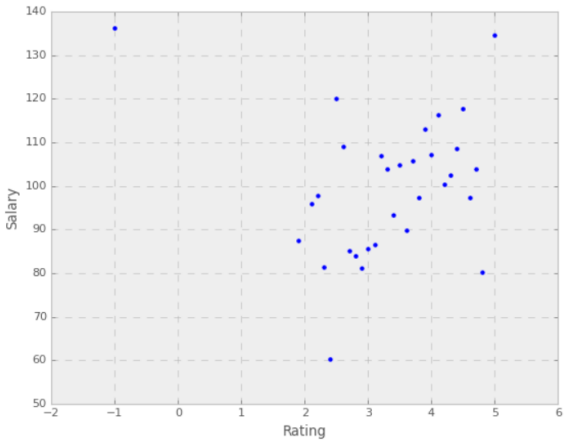
ii) **Rating: -** The rating is on a scale of 1 to 5, with -1 indicating that the data is missing.



Observation: -
Most of the ratings is in between 3.6 to 3.9
There is a very less missing value

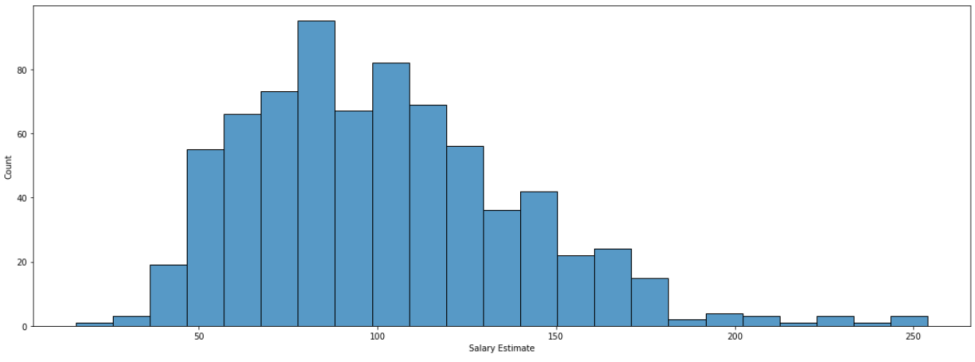
iii) **Salary and Rating: -** Taking mean Salary for all the unique rating and visualizing it with bar plot and scatter plot.





Observation: -
From the above two figures
We can see that as the rating increases the salary also Increases
This indicates a Positive Correlation between rating and Salary
Average salary for the ratings 5 company is high compared to other ratings.

iv) Salary: - Histogram for Salary Estimate



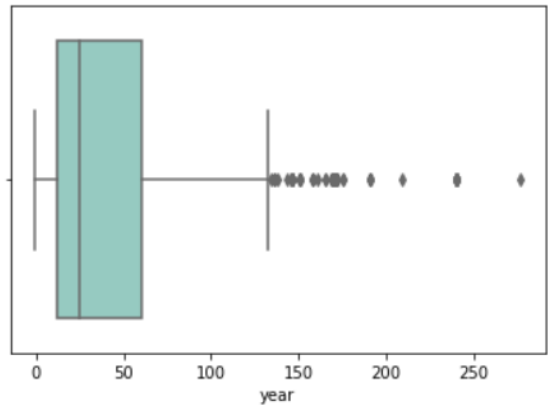
Observation: -

Well, the distribution of salary is kind of normal distribution.
The average salary is in the range of 80k – 105k
The median salary is 97.5k
There are some people who get salary above 200k which is very high.
Let us explore it in more depth, we will find the employers whose salary is above 200k

	Job Title	Type of ownership	Salary Estimate	experience
103	Senior Data Scientist	Company - Private	237.5	senior
176	Principal Data Scientist with over 10 years ex...	Company - Private	225.0	other
195	Lead Data Engineer	Company - Private	205.0	senior
266	Principal Data Scientist with over 10 years ex...	Company - Private	225.0	other
330	Lead Data Engineer	Company - Private	205.0	senior
354	Director II, Data Science - GRM Actuarial	Company - Private	254.0	senior
429	Principal Machine Learning Scientist	Subsidiary or Business Segment	232.5	other
476	Lead Data Engineer	Company - Private	205.0	senior
528	Director II, Data Science - GRM Actuarial	Company - Private	254.0	senior
613	Data Science Manager	Company - Private	221.5	senior
708	Director II, Data Science - GRM Actuarial	Company - Private	254.0	senior

From the above result we can figure out that all the employee who earn more than 200k are senior employees like Director or Head, and the highest salary is 254k and all the company are Private.

iv) Age: - Age define how old the company is.



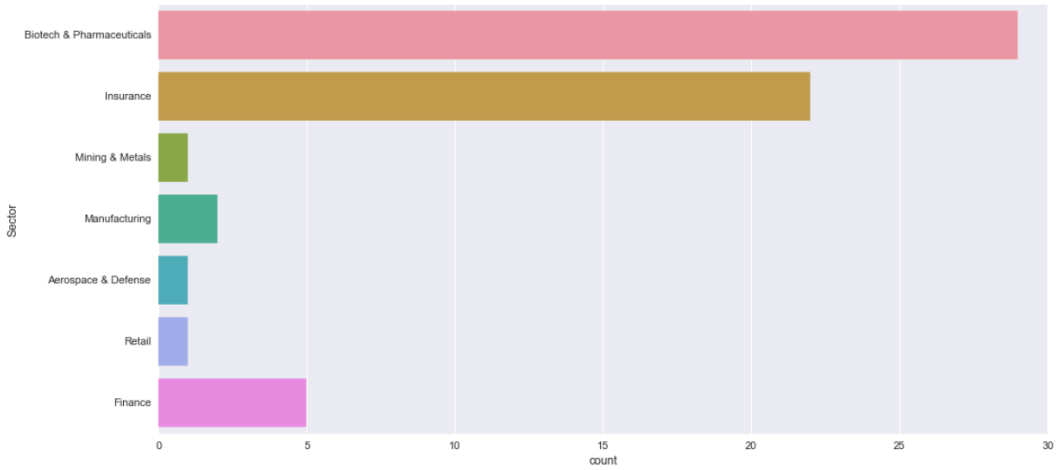
Observation: -
Many companies are less than 100 years old.

The below image shows the name of the companies which are more than 150 years old.

```
print(df[df["year"]>150]["Company Name"].unique())
```

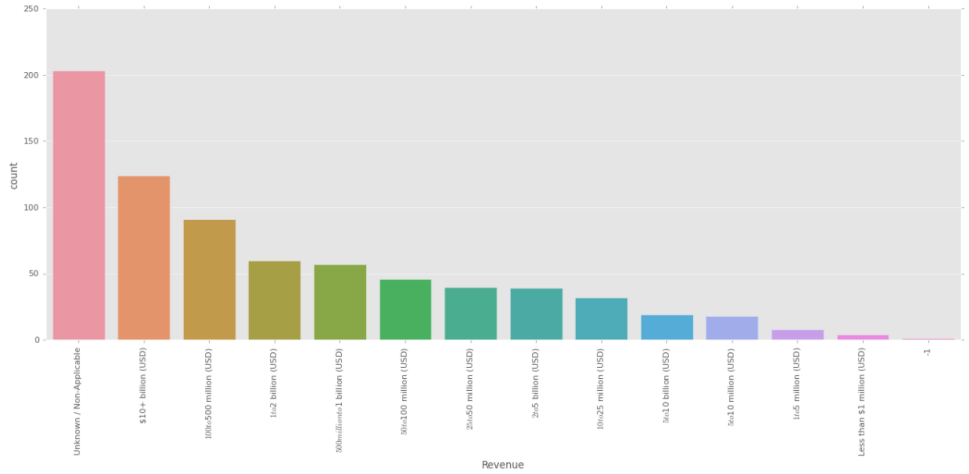
```
['Takeda Pharmaceuticals' 'Pfizer' 'The Hanover Insurance Group'
 'Sartorius' 'Swiss Re' 'Carmeuse' 'Church ' 'MassMutual'
 'BWX Technologies' 'Sotheby' 'Associated Banc' 'Santander' 'Citi' 'GSK']
```

The below image shows the number of Sector of the company which are more than 150 years old.



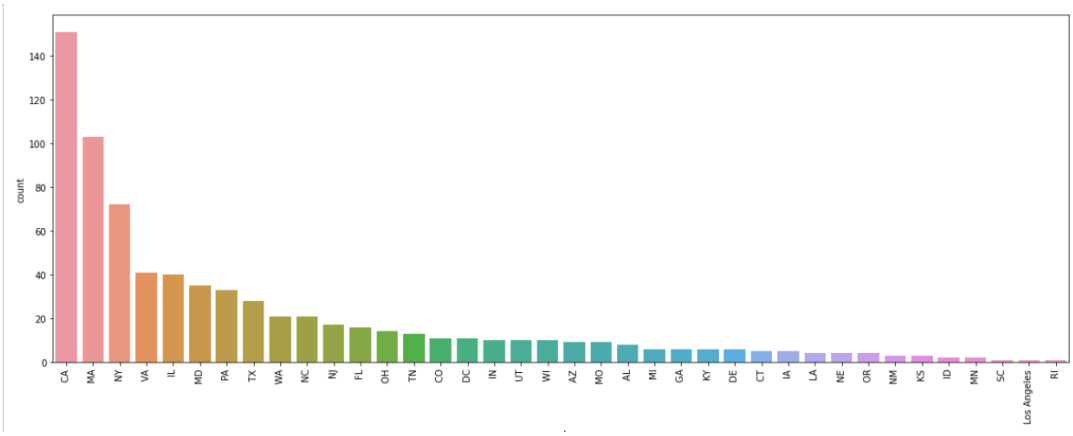
Biotech, Insurance, and finance sector companies are the oldest sector.

v) Revenue: - Profit of the Company



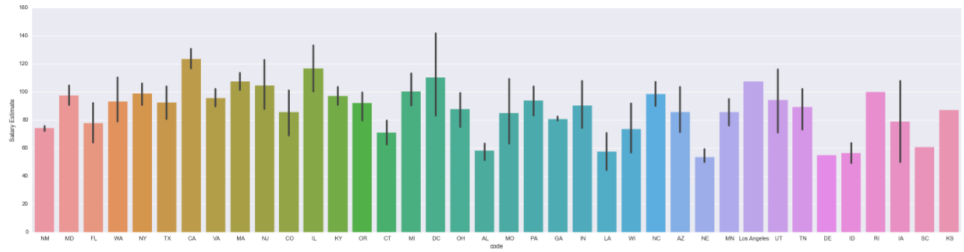
Observation: -
Most of the companies can generate the profit more than billion dollars.

vi) Job State: - Job location is where the position is available



Observation: -
There is large number of jobs available in California followed by Massachusetts, New York, etc.

vii) Job State and Salary: - Comparing average salary across all the states using a bar plot.



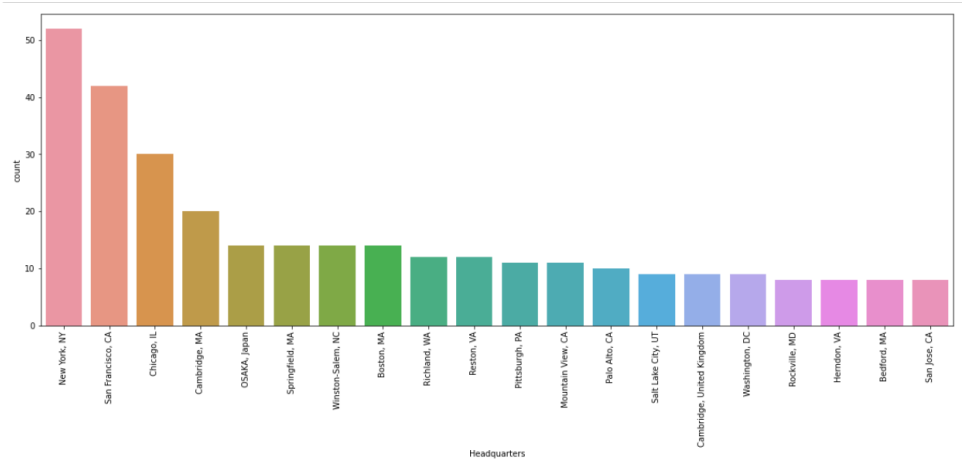
Here are the numbers in detail and in descending order of Salary

code	Salary Estimate
CA	123.619205
IL	116.662500
DC	110.181818
Los Angeles	107.500000
MA	107.412621
NJ	104.558824
MI	100.250000
RI	100.000000
NY	98.652778
NC	98.452381
MD	97.357143
KY	97.000000
VA	95.621951
UT	94.150000
PA	93.803030
WA	93.190476
TX	92.464286
OR	92.125000
IN	90.300000
TN	89.192308
OH	87.571429
KS	87.000000
AZ	85.666667

AZ	85.666667
CO	85.636364
MN	85.500000
MO	84.722222
GA	80.666667
IA	78.900000
FL	77.625000
NM	74.333333
WI	73.300000
CT	71.100000
SC	60.500000
AL	57.937500
LA	57.250000
ID	56.250000
DE	55.000000
NE	53.500000

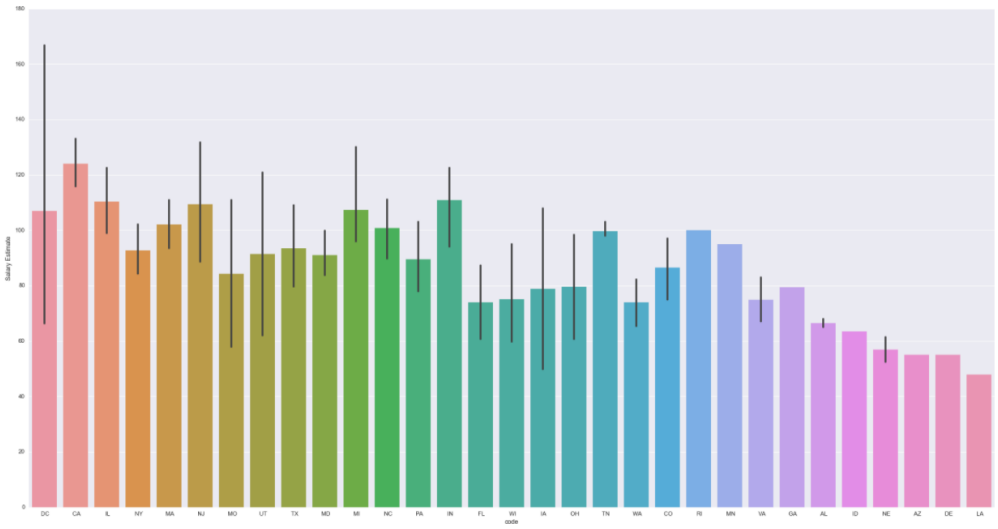
Observation: -
The number clearly tell us the average salary in California is highest followed by Illinois, Washington etc.According to Forbes list Aug 2021, San Francisco, California is the top IT city in the United States, which explains why there are so many job openings, and the salaries are so high.

viii) Headquarters: State count plot for the company's primary headquarters.



Observation: -
New York and California are home to many organizations. Some corporations have their headquarters in Cambridge, United Kingdom. This also explains why, because Cambridge is the United Kingdom's Information Technology capital.

ix) Headquarters and Salary: - Calculating the average salary based on location headquarters and visualizing it with the help of Bar plot.

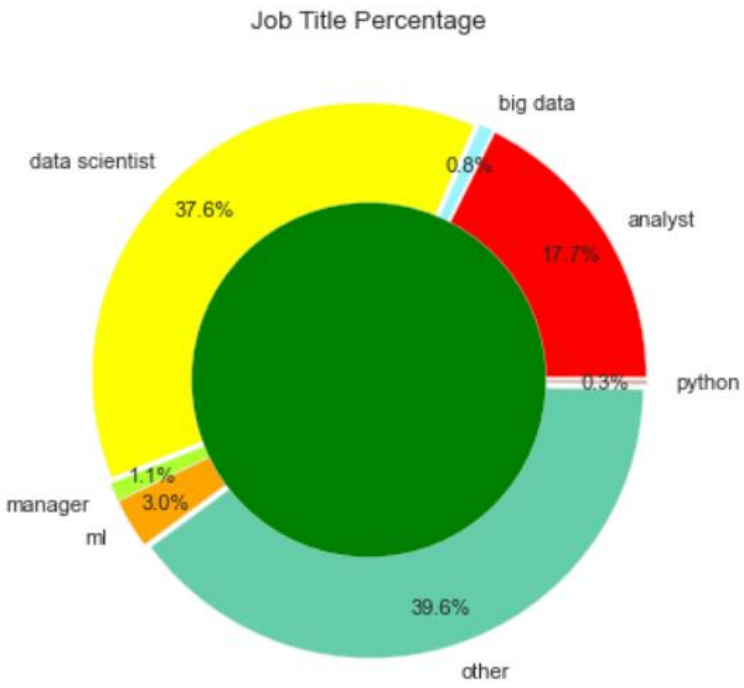


CA	124.136364
IN	110.900000
IL	110.333333
NJ	109.400000
MI	107.333333
DC	106.916667
MA	102.200000
NC	100.750000
RI	100.000000
TN	99.700000
MN	95.000000
TX	93.500000
NY	92.803922
UT	91.500000
MD	91.138889
PA	89.525000
CO	86.555556
MO	84.357143
OH	79.625000

GA	79.500000
IA	78.900000
WI	75.166667
VA	75.000000
FL	73.966667
WA	73.964286
AL	66.500000
ID	63.500000
NE	57.000000
AZ	55.000000
DE	55.000000
LA	48.000000

Employees who work at California headquarters earn significantly more than those in other states.

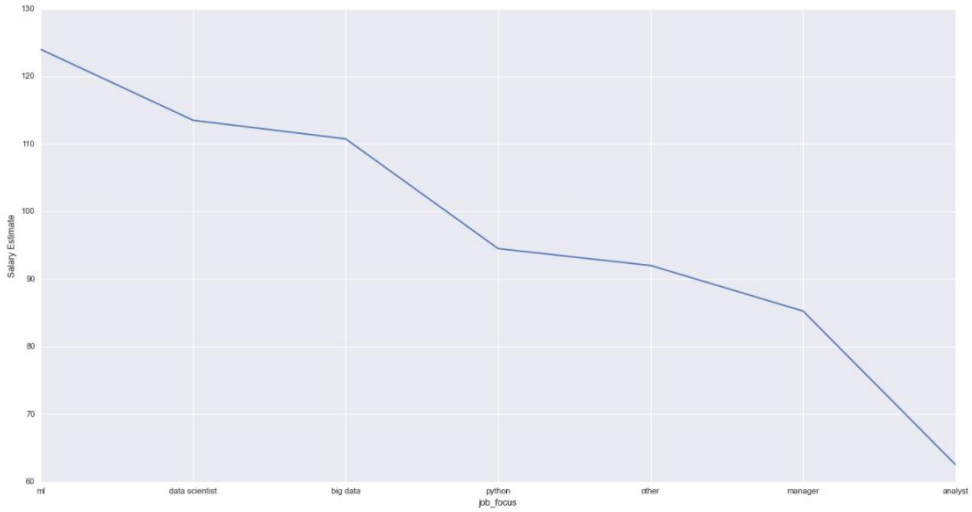
x) Job Focus: - Percentage of different job title

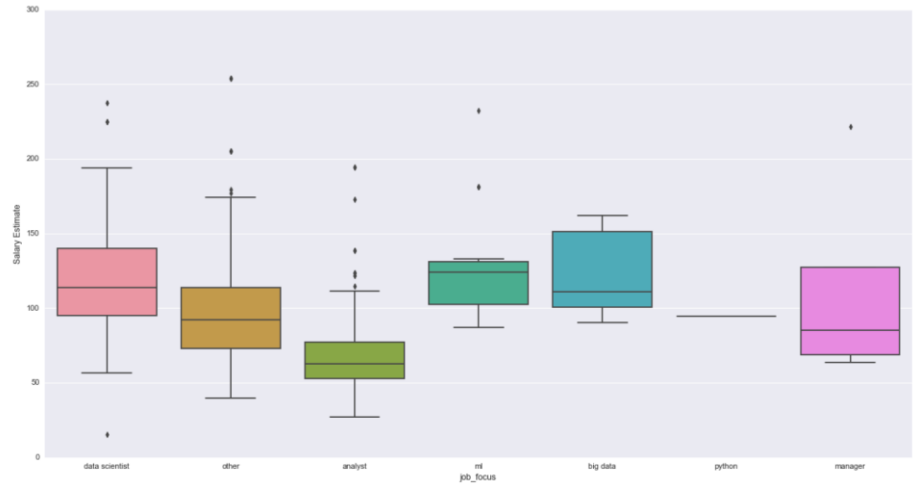


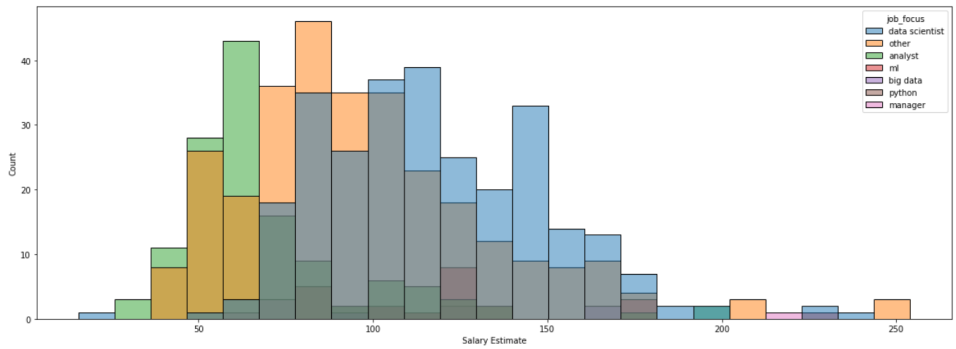
Observation: -
There is a huge demand for data scientists in the market.
Analysts are also required in the majority of organizations.
There is very little demand for python engineers.
Even big data demand is very low.

xi) Job Focus and Salary: - Calculating average salary for all this titles analyst, big data, data scientist, manager, ml, other, and python and visualizing it with boxplot.

job_focus	Salary Estimate
ml	124.00
data scientist	113.50
big data	110.75
python	94.50
other	92.00
manager	85.25
analyst	62.50



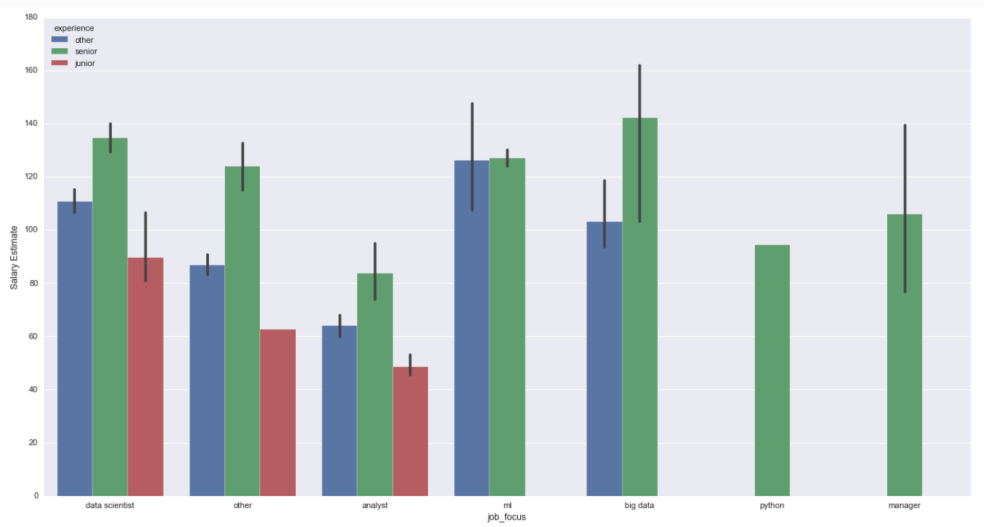




Observation: -
The income for a Machine Learning Engineer is the highest, as seen in the table above. Data Scientists are also well compensated. The fact that the managers are paid so little is unusual. Analyst is paid the least in the group.

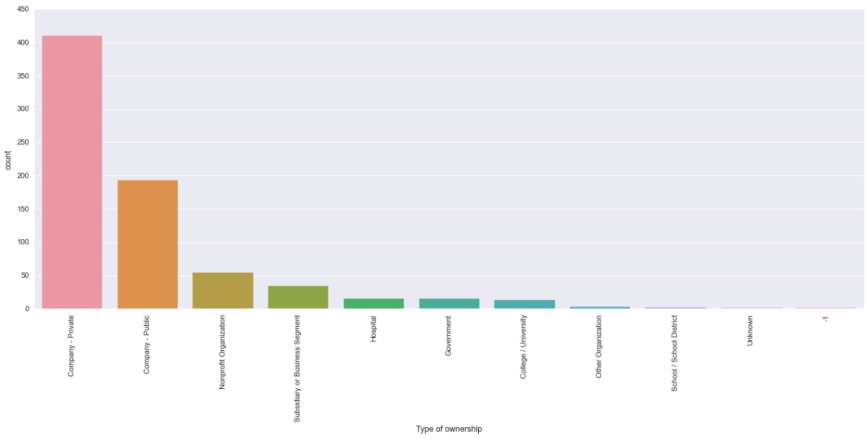
xii) Experience, Job Focus and Salary: - Calculating the average salary based on Job title and Experience.

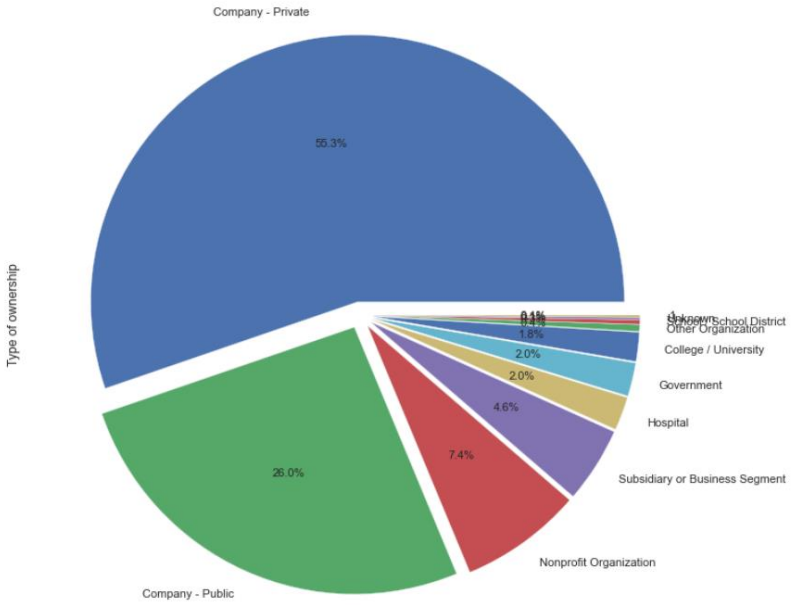
		Salary Estimate
job_focus	**experience**	
analyst	**junior**	48.600000
	other	63.882716
	senior	83.677778
big data	**other**	103.000000
	senior	142.333333
data scientist	**junior**	89.500000
	other	110.642487
	senior	134.674699
manager	**senior**	105.937500
ml	**other**	126.218750
	senior	127.000000
other	**junior**	62.500000
	other	86.745098
	senior	123.931818
python	**senior**	94.500000



Observation: -
Data scientist, other, and analyst profile clearly indicates that the senior employees earn more than the junior employee. Surprisingly senior employee in the field of big data earn the highest salary among all. Data Analyst field earns the lowest salary for both junior and senior among the group. Salary of junior data scientist is more than senior data analyst. Another intriguing finding is that managers are paid less than senior Data Scientists, Machine Learning experts, and big data analysts.

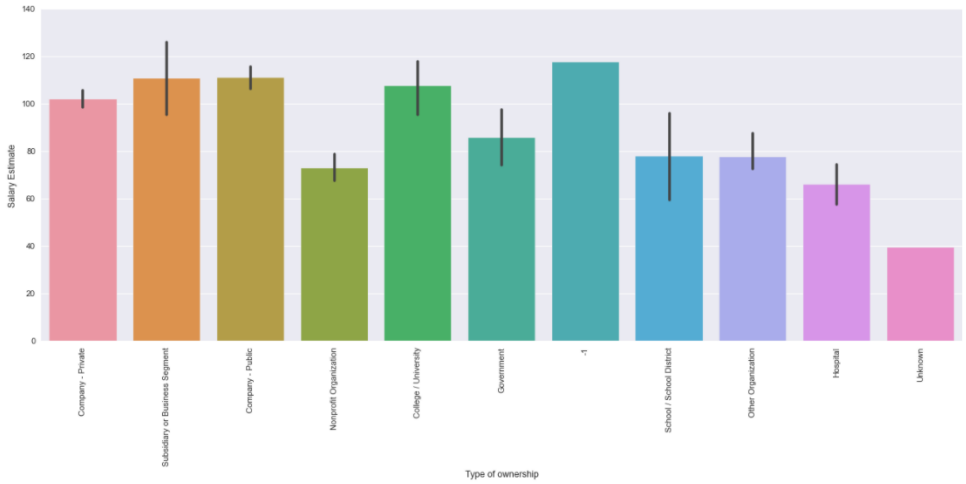
xi) Type of Ownership: - This identifies whether the company is private, public, or something else.





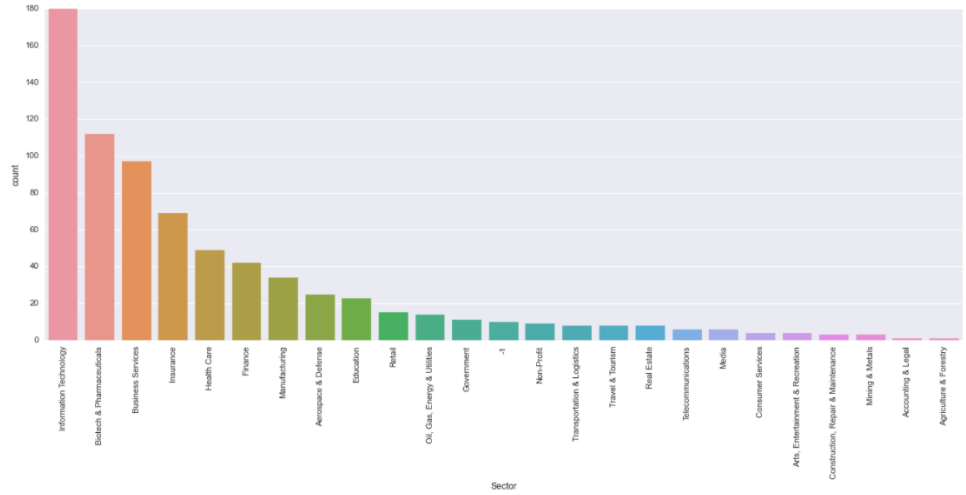
Observation: -
Majority of the companies are private followed by public and non-profit organization.

xiii) Type of Ownership and Salary: - Calculating the average salary for each type of ownership and visualising the result using bar plot.



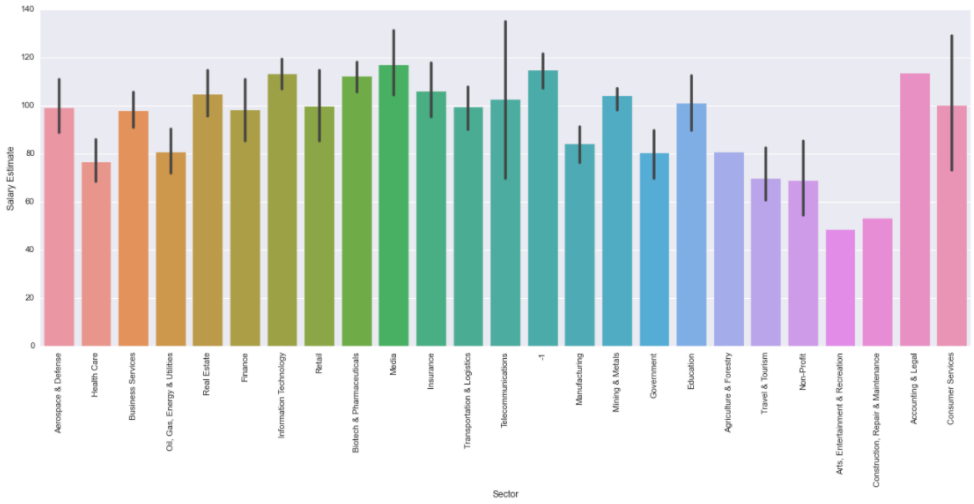
Observation: -
A corporation with a public ownership structure pays a higher wage than one with a private ownership structure. Even universities pay higher salaries than private businesses.

xiv) Sector: - Visualizing various Sector and total count of each sector.



Observation: -
Information Technology sector require a greater number of data science related domain employees. Sectors like Agriculture, Accounting, Construction, Repair, Entertainment etc, does not require data science related domain employees for their business.

xv) Sector and Salary: - Calculating average salary for all the sectors and visualizing the result.



Observation: - Almost all the sectors are getting paid evenly.
Media Sector is getting the highest average salary followed by Information Technology.

5 Model Building

Now we have our data pre-processed and we are done analysing our data. At this point we will take the cleaned data and build a machine learning model on it. We will take several different model and check how they are performing.

First, we will select the following feature to build our model

X = Rating, Location, Size, Founded, Type of ownership, Industry, Sector, Revenue, Competitors, hour, year, code, head_state, Job_len, python, aws, azure, visualization, analysis, excel, bi, job_focus, experience

y = Salary estimate

X is the independent variable
y is dependent variable

1 Encoding Categorical Variable

We have our X and y ready, but there is some problem we have lots of feature which are in string format and machine learning model only understand number
So we need to encode the categorical features into numbers. We will use one hot encoding to achieve this task.

Job Title	Salary Estimate	Job Description	Rating	Company Name	Location	Headquarters	Size	Founded	Type of ownership	...	Job_len	python	aws	azure	vis
Data Scientist	72.0	Data ScientistLocation: Albuquerque, NMEducati...	3.8	Tecolote Research	Albuquerque, NM	Goleta, CA	501 to 1000 employees	1973	Company - Private	...	2517	1	0	0	
Healthcare Data Scientist	87.5	What You Will Do:I. General SummaryThe Healthc...	3.4	University of Maryland Medical System	Linthicum, MD	Baltimore, MD	10000+ employees	1984	Other Organization	...	4738	1	0	0	
Data Scientist	85.0	KnowBe4, Inc. is a high growth information sec...	4.8	KnowBe	Clearwater, FL	Clearwater, FL	501 to 1000 employees	2010	Company - Private	...	3427	1	0	0	
Data Scientist	76.5	*Organization and Job ID**Job ID: 310709Direct...	3.8	PNNL	Richland, WA	Richland, WA	1001 to 5000 employees	1965	Government	...	3840	1	0	0	
Data Scientist	114.5	Data ScientistAffinity Solutions / Marketing C...	2.9	Affinity Solutions	New York, NY	New York, NY	51 to 200 employees	1998	Company - Private	...	2708	1	0	0	

After performing one hot encoding our data looks like following

Rating	hour	year	head_state	Job_len	python	aws	azure	visualization	analysis	...	job_focus_analyst	job_focus_big data	job_focus_data scientist	job_focus_man
3.8	0	48	0	2517	1	0	0	1	1	...	0	0	1	
3.4	0	37	0	4738	1	0	0	1	1	...	0	0	1	
4.8	0	11	1	3427	1	0	0	1	1	...	0	0	1	
3.8	0	56	1	3840	1	0	0	0	1	...	0	0	1	
2.9	0	23	1	2708	1	0	0	0	1	...	0	0	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
3.9	0	191	0	6105	0	1	0	0	0	...	0	0	0	
4.4	0	15	0	6093	1	1	0	0	1	...	0	0	0	
2.6	0	37	1	3049	0	0	0	0	0	...	0	0	0	

One hot encoding: - One approach of transforming data to prepare it for an algorithm and improve prediction is one hot encoding. With one-hot, each category value is converted into a new categorical column and given a binary value of 1 or 0. A binary vector is used to represent each integer value. The index is designated with a 1 and all of the values are zero. (Fawcett, 2021)

Now we have a data which is completely ready to build a machine learning model.

2 Splitting the data Train Set and Test Set

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

X_train and X_test contains independent Variable
y_train and y_test contains dependent Variable

Using X_train and y_train we will train our model and check it performance on X_test and y_test.

3 Linear Regression

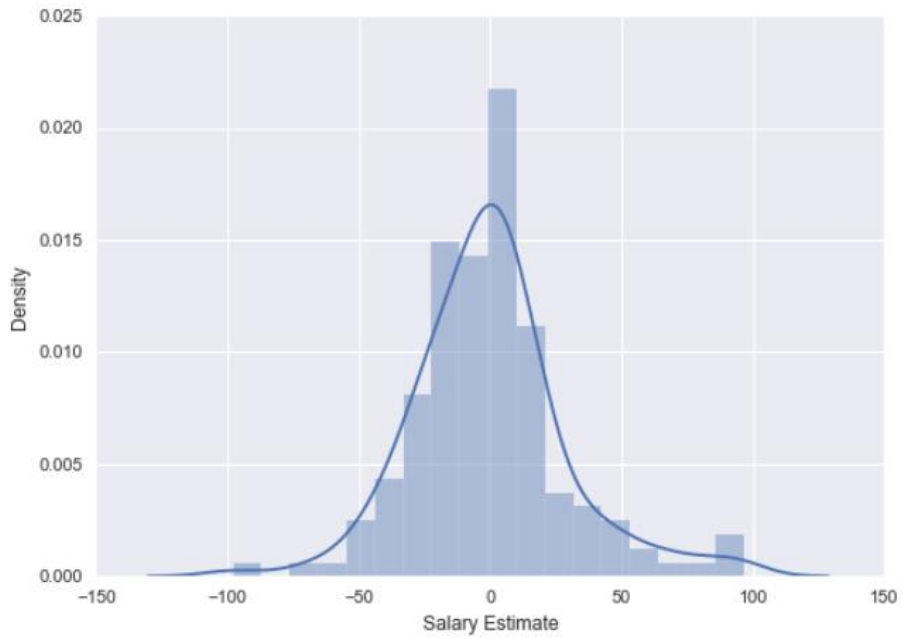
Linear Regression is a very simple model, and we will try to see how it performs on our data.

```
from sklearn import metrics
linear = LinearRegression()
linear.fit(X_train,y_train)
linear_predict = linear.predict(X_test)
algo.append(str(linear))
mae.append(metrics.mean_absolute_error(y_test, linear_predict))
print('MAE:', metrics.mean_absolute_error(y_test, linear_predict))
sns.distplot(y_test-linear_predict)
```

```
MAE: 20.649289446815185
```

Below figure shows the distribution of error (Prediction value – True value) and Mean Absolute Error

MAE: 20.649289446815185



The Mean Absolute Error is just 20.6492, which is extremely low, and the graph clearly illustrates that the result is in the shape of a bell curve. This is a good sign, and the fact that the majority of the values are close to zero indicates that our model has produced very few errors.

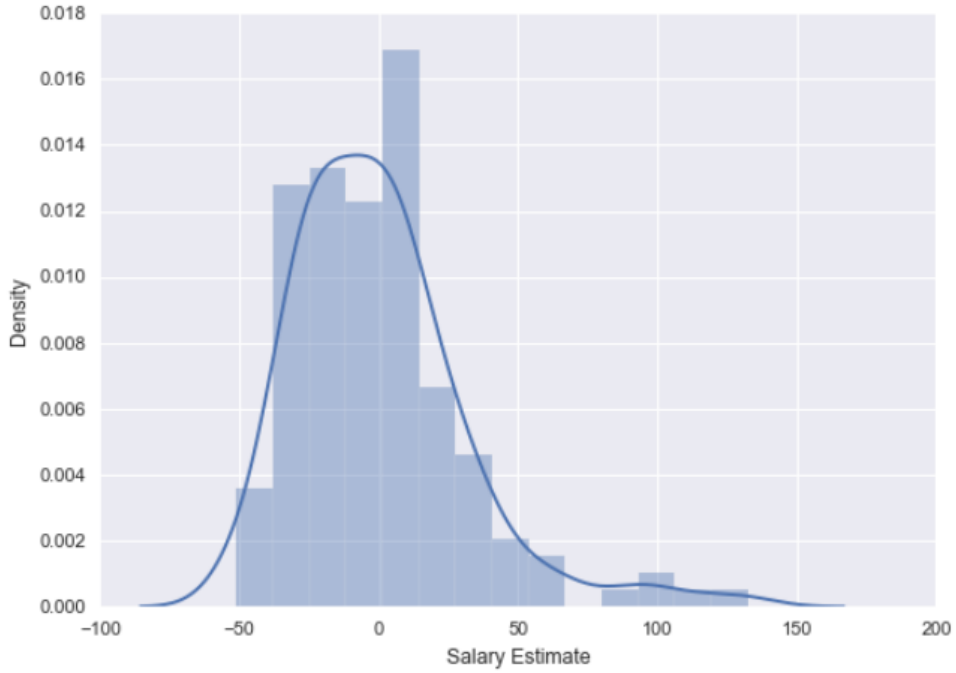
4 Lasso Regression

Lasso Regression is like a Linear Regression but with extra benefits like shrinkage.

```
linear = Lasso()
linear.fit(X_train,y_train)
linear_predict = linear.predict(X_test)
algo.append(str(linear))
mae.append(metrics.mean_absolute_error(y_test, linear_predict))
print('MAE:', metrics.mean_absolute_error(y_test, linear_predict))
sns.distplot(y_test-linear_predict)
```

Below figure shows the distribution of error (Prediction value – True value) and Mean Absolute Error.

```
MAE: 22.8835304367422
```



The Mean Absolute Error is just 22.8835, which is low, and the graph illustrates that the result is in the shape of a curve. In Lasso Regression there few high errors compared to Linear Regression

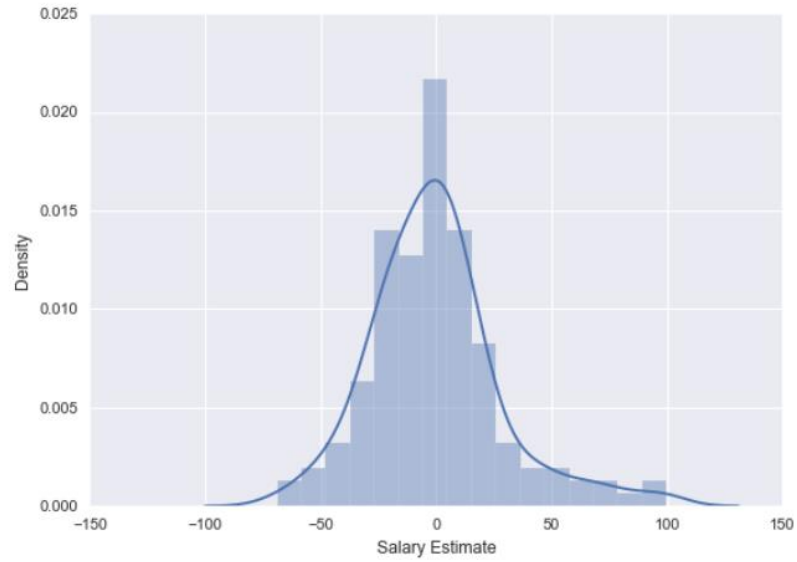
5 Ridge Regression

Ridge Regression is another type of regularized model.

```
linear = Ridge()
linear.fit(X_train,y_train)
linear_predict = linear.predict(X_test)
algo.append(str(linear))
mae.append(metrics.mean_absolute_error(y_test, linear_predict))
print('MAE:', metrics.mean_absolute_error(y_test, linear_predict))
sns.distplot(y_test-linear_predict)
plt.show()
```

Below figure shows the distribution of error (Prediction value – True value) and Mean Absolute Error

```
MAE: 20.341668405360302
```



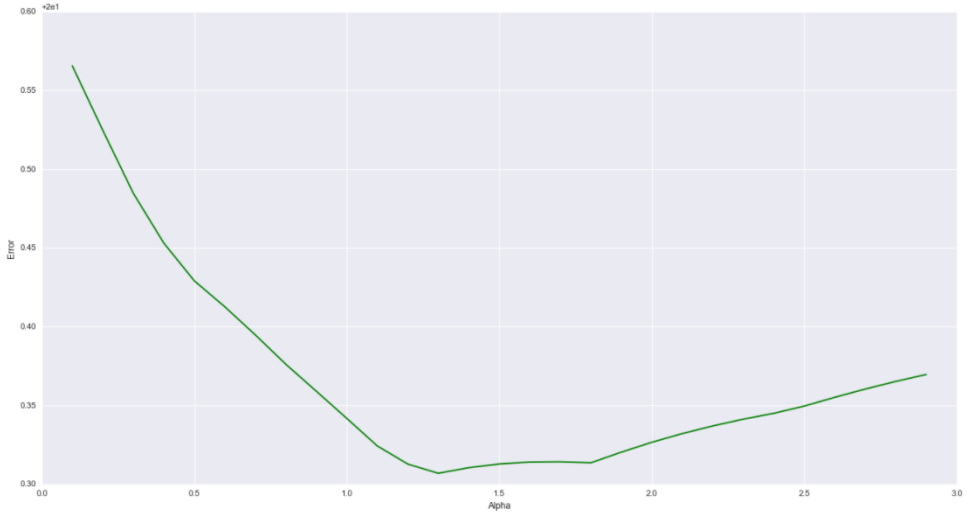
The Mean Absolute Error is just 20.3416, which is low, and the graph clearly illustrates that the result is in the shape of a bell curve. This is a good sign, and the fact that the majority of the values are close to zero indicates that our model has produced very few errors. Ridge Regression results are better than Linear Regression and Lasso Regression.

Ridge Regression with different alpha value

Alpha is hyperparameter and we need to select the value for it. Default value of alpha is 1. From 0.1 to 2.9 we will try different values for alpha and see the results.

```
alpha_list = []
diff_alpha_error = []
for i in range(1,30):
    i = i/10
    alpha_list.append(i)
    linear = Ridge(alpha=i)
    linear.fit(X_train,y_train)
    linear_predict = linear.predict(X_test)
    diff_alpha_error.append(metrics.mean_absolute_error(y_test, linear_predict))
```

Below figure shows the lineplot for different alpha on x-axis and the error on y-axis



The error is the least when alpha is 1.3. If the alpha is too small, the error rate is large. If the alpha is between 1 and 2, the error rate is low, but beyond 2, the error rate begins to rise.

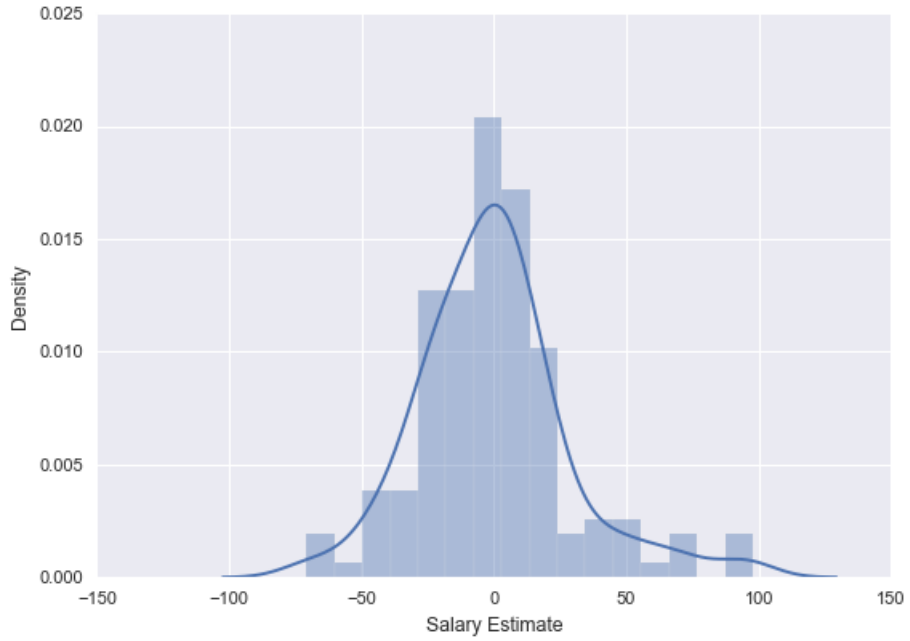
6 Kernel Ridge Regression

Kernel Ridge regression is useful when the data in non-linear.

```
linear = KernelRidge(alpha=0.5)
linear.fit(X_train,y_train)
linear_predict = linear.predict(X_test)
algo.append(str(linear))
mae.append(metrics.mean_absolute_error(y_test, linear_predict))
print('MAE:', metrics.mean_absolute_error(y_test, linear_predict))
sns.distplot(y_test-linear_predict)
```

```
MAE: 20.49164529946419
```

```
<AxesSubplot:xlabel='Salary Estimate', ylabel='Density'>
```



The Mean Absolute Error 20.4916 is lower than lasso Regression 22.88, But higher than Ridge Regression 20.34

7 Elastic Net

Elastic Net is the combination of Ridge Regression and Lasso Regression. (Michael Fuchs, 2019)

```
linear = ElasticNet(alpha=0.0005, l1_ratio=0.5)
linear.fit(X_train,y_train)
linear_predict = linear.predict(X_test)
algo.append(str(linear))
mae.append(metrics.mean_absolute_error(y_test, linear_predict))
print('MAE:', metrics.mean_absolute_error(y_test, linear_predict))
sns.distplot(y_test-linear_predict)
plt.show()
```

Below figure shows the distribution of error (Prediction value – True value) and Mean Absolute Error

```
MAE: 20.54060071763065
```

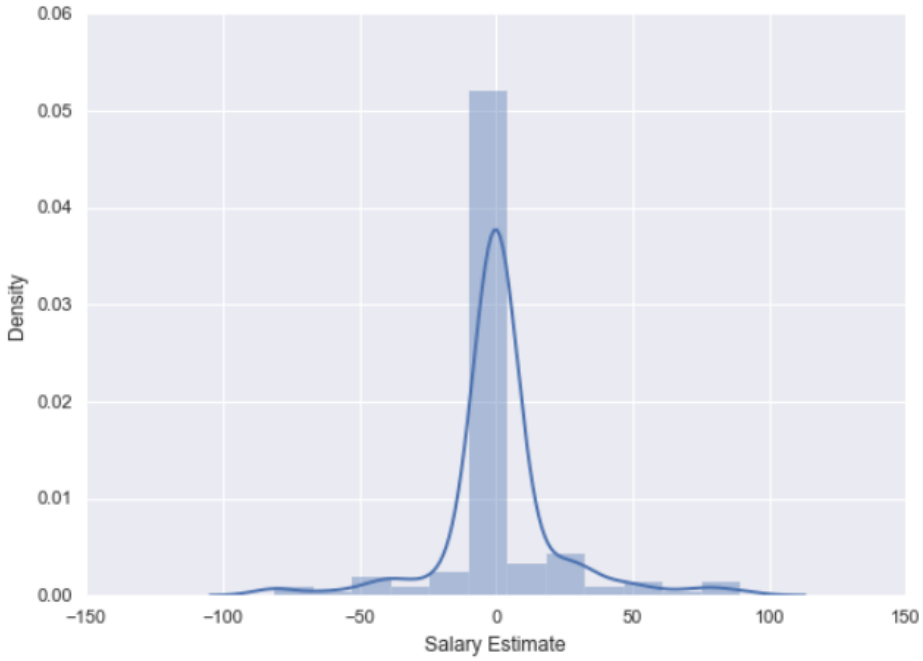
The Mean Absolute Error is just 20.5406, which is extremely low, and the graph clearly illustrates that the result is in the shape of a bell curve. This is a good sign, and the fact that the majority of the values are close to zero indicates that our model has produced very few errors.

8 Decision Tree

Decision Tree is a tree based model.

```
dt = DecisionTreeRegressor()
dt.fit(X_train,y_train)
dt_predict = dt.predict(X_test)
algo.append(str(dt))
mae.append(metrics.mean_absolute_error(y_test, dt_predict))
print('MAE:', metrics.mean_absolute_error(y_test, dt_predict))
sns.distplot(y_test-dt_predict)
plt.show()
```

```
MAE: 9.664429530201343
```



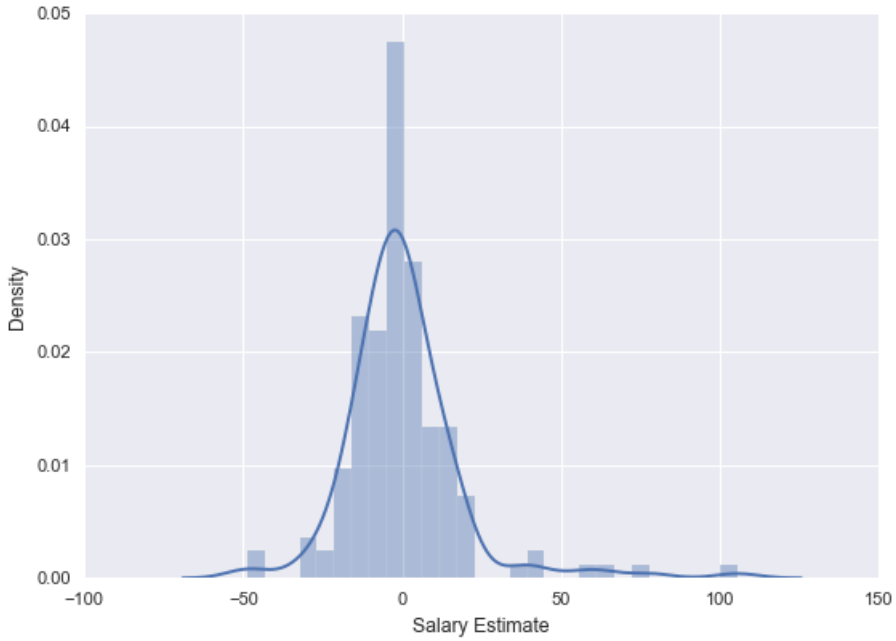
The Mean Absolute Error is just 9.66, which is extremely low till now , and the graph clearly illustrates that the result is in the shape of a bell curve. This is a good sign, and the fact that most of the values are close to zero indicates that our model has produced very few errors.

9 Random Forest

Random Forest is an Ensemble learning method.

```
dt = RandomForestRegressor()
dt.fit(X_train,y_train)
dt_predict = dt.predict(X_test)
algo.append(str(dt))
mae.append(metrics.mean_absolute_error(y_test, linear_predict))
print('MAE:', metrics.mean_absolute_error(y_test, dt_predict))
sns.distplot(y_test-dt_predict)
plt.show()
```

```
MAE: 11.33751677852349
```

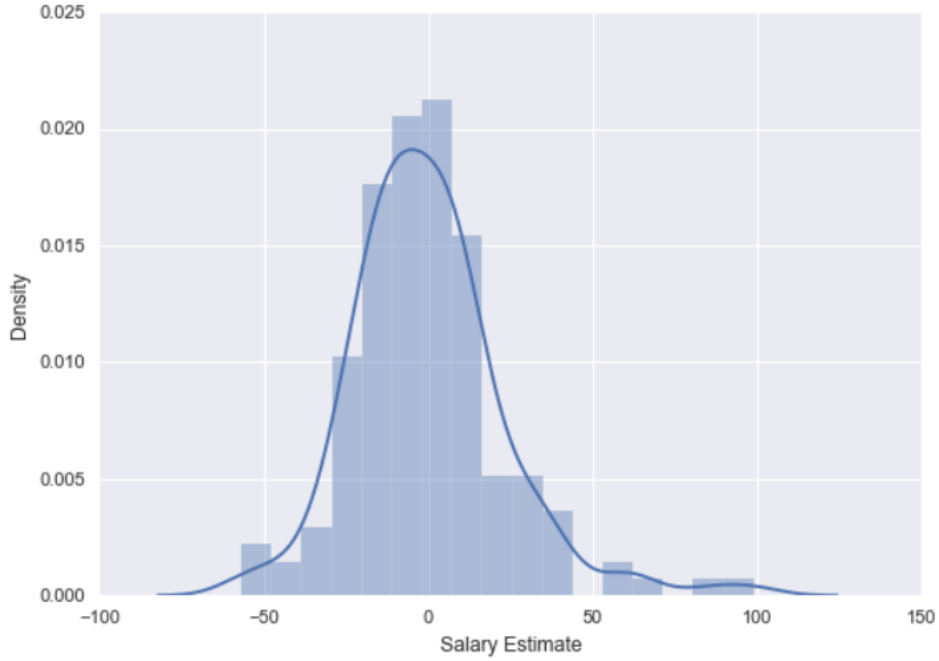


The Mean Absolute Error is just 11.3375, which is low, and the graph clearly shows that the result is in the shape of a bell curve

10 Gradient Boosting

```
from sklearn.ensemble import GradientBoostingRegressor
gdb = GradientBoostingRegressor()
gdb.fit(X_train,y_train)
gdb_predict = gdb.predict(X_test)
algo.append(str(gdb))
mae.append(metrics.mean_absolute_error(y_test, gdb_predict))
print('MAE:', metrics.mean_absolute_error(y_test, gdb_predict))
sns.distplot(y_test-gdb_predict)
plt.show()
```

```
MAE: 16.887021075545828
```

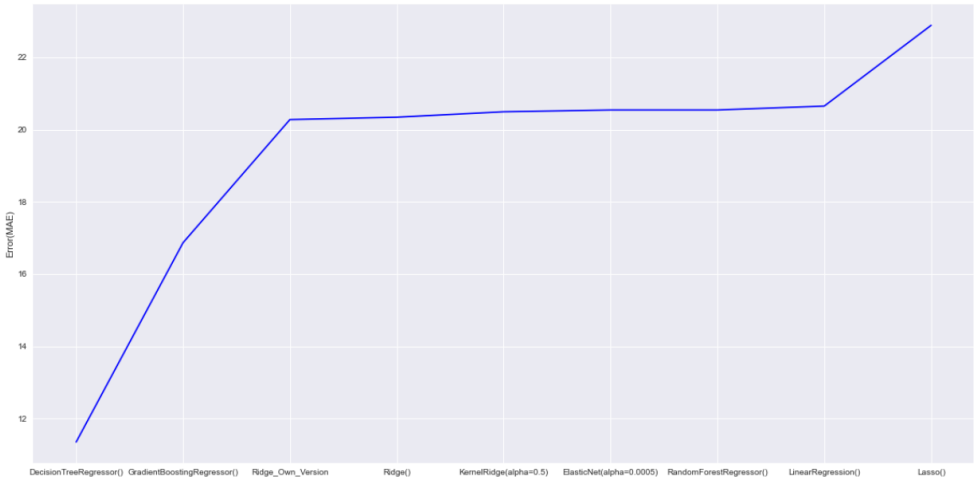


The Mean Absolute Error of Gradient Boosting is 16.88 which is better than linear models.

6 Visualizing the Results

After Trying multiple models now, it is time to compare all the models and check their performance.

Algorithm	MAE
DecisionTreeRegressor()	11.355705
GradientBoostingRegressor()	16.869047
Ridge_Own_Version	20.275655
Ridge()	20.341668
KernelRidge(alpha=0.5)	20.491645
ElasticNet(alpha=0.0005)	20.540601
RandomForestRegressor()	20.540601
LinearRegression()	20.649289
Lasso()	22.883530



The Results Clearly Shows that Decision Tree is the Best performing model for this dataset followed by Gradient Boosting and Ridge Regression. While the Lasso Regression Mean Absolute Error is the highest among all.