STAT570 - Final Project

Data Scientist Job Posting Data

Miray Çınar & Ecenaz Tunç

Table of contents

Introduction

Glassdoor is an online platform where former or new employees can comment on companies and is also used for job search.

In their website, they define themselves as: "Glassdoor is one of the world's largest job and recruiting sites. We pride ourselves on helping people find a job and company they love; in fact, it's our mission. Our company was built on the foundation of increasing workplace transparency. With that in mind, we have developed numerous tools to help job seekers make more informed career decisions."

Data Description

Data is obtained from Kaggle, in which, the user claims that the data is obtained from Glassdoor.com by using web scrapping.

The variables in this data set are defined as follows:

Job Title: Title of the job posting

Salary Estimation: Salary range for that particular job

Job Description: This contains the full description of that job

Rating: Rating of that post

Company: Name of company

Location: Location of the company

Headquarter: Location of the headquater

Size: Total employee in that company

Type of ownership: Describes the company type i.e non-profit/public/private farm etc

Industry, Sector: Field applicant will work in

Revenue: Total revenue of the company

Import the Data

Let's start by importing the data. Fist, import the required libraries. If you don't already have them, you can use `install.packages()` function.

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.4.4     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.0
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```
library(readr)
library(ggplot2)

Uncleaned_DS_jobs <- read_csv("Uncleaned_DS_jobs.csv")
```

```
Rows: 672 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (12): Job Title, Salary Estimate, Job Description, Company Name, Locatio...
dbl  (3): index, Rating, Founded

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Let's start by investigating our dataset a little bit, by getting a glimpse and see the structure of the data:

```r
library(dplyr)
glimpse(Uncleaned_DS_jobs)
```

```
Rows: 672
Columns: 15
$ index               <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, ~
$ `Job Title`         <chr> "Sr Data Scientist", "Data Scientist", "Data Scien~
$ `Salary Estimate`   <chr> "$137K-$171K (Glassdoor est.)", "$137K-$171K (Glas~
$ `Job Description`   <chr> "Description\n\nThe Senior Data Scientist is respo~
$ Rating              <dbl> 3.1, 4.2, 3.8, 3.5, 2.9, 4.2, 3.9, 3.5, 4.4, 3.6, ~
$ `Company Name`      <chr> "Healthfirst\n3.1", "ManTech\n4.2", "Analysis Grou~
$ Location            <chr> "New York, NY", "Chantilly, VA", "Boston, MA", "Ne~
$ Headquarters        <chr> "New York, NY", "Herndon, VA", "Boston, MA", "Bad ~
$ Size                <chr> "1001 to 5000 employees", "5001 to 10000 employees~
$ Founded             <dbl> 1993, 1968, 1981, 2000, 1998, 2010, 1996, 1990, 19~
$ `Type of ownership` <chr> "Nonprofit Organization", "Company - Public", "Pri~
$ Industry            <chr> "Insurance Carriers", "Research & Development", "C~
$ Sector              <chr> "Insurance", "Business Services", "Business Servic~
$ Revenue             <chr> "Unknown / Non-Applicable", "$1 to $2 billion (USD~
$ Competitors         <chr> "EmblemHealth, UnitedHealth Group, Aetna", "-1", "~
```

And also take a quick summary:

```r
summary(Uncleaned_DS_jobs)
```

```
     index         Job Title         Salary Estimate    Job Description
 Min.   :  0.0   Length:672         Length:672         Length:672
 1st Qu.:167.8   Class :character   Class :character   Class :character
```

```
 Median :335.5   Mode  :character   Mode  :character   Mode  :character
 Mean   :335.5
 3rd Qu.:503.2
 Max.   :671.0
     Rating       Company Name         Location          Headquarters
 Min.   :-1.000   Length:672         Length:672         Length:672
 1st Qu.: 3.300   Class :character   Class :character   Class :character
 Median : 3.800   Mode  :character   Mode  :character   Mode  :character
 Mean   : 3.519
 3rd Qu.: 4.300
 Max.   : 5.000
     Size              Founded     Type of ownership    Industry
 Length:672         Min.   :  -1   Length:672         Length:672
 Class :character   1st Qu.:1918   Class :character   Class :character
 Mode  :character   Median :1995   Mode  :character   Mode  :character
                    Mean   :1636
                    3rd Qu.:2009
                    Max.   :2019
    Sector            Revenue          Competitors
 Length:672         Length:672         Length:672
 Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character
```

From both `glimpse()` and `summary()` outputs, we can see that, categorical variables are in character form. We will investigate them one by one later on.

But first, let's change the column names that have blank spaces so that it will be much easy to make the analyses later.

```
Uncleaned_DS_jobs <- Uncleaned_DS_jobs %>%
  rename(Job_Title = `Job Title`,
         Salary_Estimate = `Salary Estimate`,
         Job_Description = `Job Description`,
         Company_Name = `Company Name`,
         Type_of_Ownership =  `Type of ownership`)
```

Take a summary again to see the data:

```r
summary(Uncleaned_DS_jobs)
```

```
    index         Job_Title          Salary_Estimate    Job_Description   
 Min.   :  0.0   Length:672         Length:672         Length:672        
 1st Qu.:167.8   Class :character   Class :character   Class :character  
 Median :335.5   Mode  :character   Mode  :character   Mode  :character  
 Mean   :335.5                                                           
 3rd Qu.:503.2                                                           
 Max.   :671.0                                                           
     Rating        Company_Name         Location         Headquarters      
 Min.   :-1.000   Length:672         Length:672         Length:672        
 1st Qu.: 3.300   Class :character   Class :character   Class :character  
 Median : 3.800   Mode  :character   Mode  :character   Mode  :character  
 Mean   : 3.519                                                           
 3rd Qu.: 4.300                                                           
 Max.   : 5.000                                                           
     Size              Founded     Type_of_Ownership    Industry        
 Length:672         Min.   :  -1   Length:672         Length:672        
 Class :character   1st Qu.:1918   Class :character   Class :character  
 Mode  :character   Median :1995   Mode  :character   Mode  :character  
                    Mean   :1636                                        
                    3rd Qu.:2009                                        
                    Max.   :2019                                        
    Sector            Revenue          Competitors       
 Length:672         Length:672         Length:672        
 Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character  
```

From our summary, we can also see some strange values are present in the data. For instance there are some rows marked as "-1" in the Headquarters, Founded.

Investigating the Columns

Let's investigate the columns one by one:

- Index

Index column is not necessary for us, so we will remove it from our data set.

```r
Uncleaned_DS_jobs$index <- NULL
```

- Rating

We realized that from the summary, Rating has a minimum value as -1, but the rating should be between 0 to 5.

We need to fix that problem.

To fix this, first we need to look how many data are there with Rating = -1:

```r
sum(Uncleaned_DS_jobs$Rating == -1)
```

```
[1] 50
```

We have 50 values with Rating = -1. Rating variable should not be -1. So firstly for the rating variable we give change -1 to 0.

```r
Uncleaned_DS_jobs$Rating[Uncleaned_DS_jobs$Rating == -1] <- 0
```

Now lets check if it worked,

```r
summary(Uncleaned_DS_jobs$Rating)
```

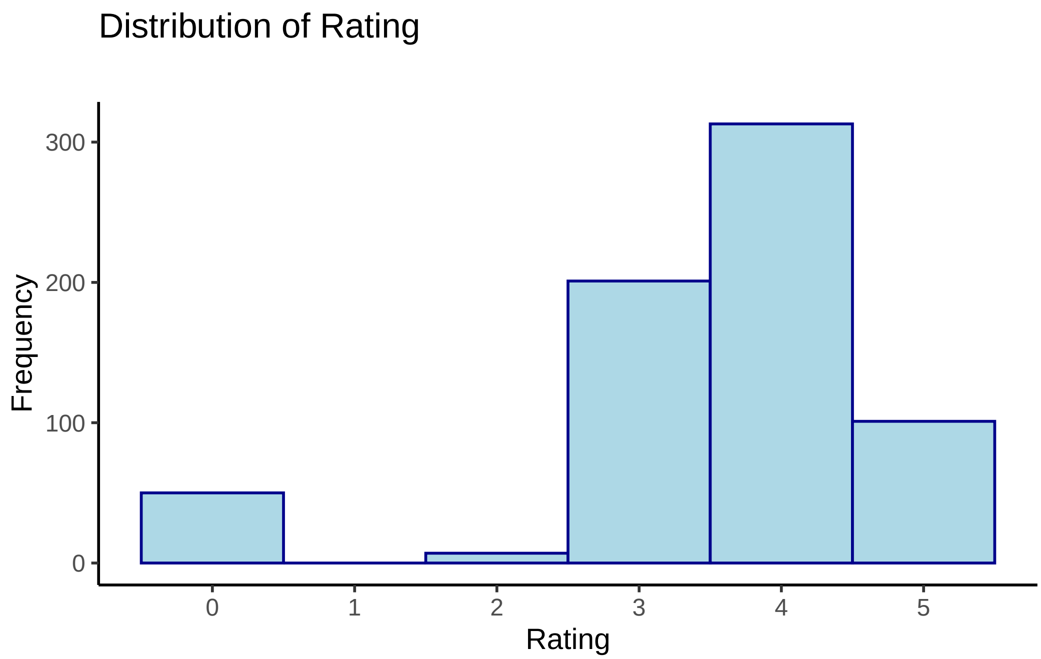
```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.000   3.300   3.800   3.593   4.300   5.000 
```

As can be seen from the summary of the rating we fix the -1 problem.

Histogram of the rating variable:

```r
intervals <- seq(0, 5, by = 1)

rating_histogram <- ggplot(Uncleaned_DS_jobs, aes(x = Rating)) +
  geom_histogram(binwidth = 1, boundary = 0.5, col = "darkblue", fill = "lightblue") +
  labs(
    title = "Distribution of Rating",
    x = "Rating",
    y = "Frequency",
    subtitle = ""
  ) +
```

```
  scale_x_continuous(breaks = intervals) +
  theme_classic()
rating_histogram
```



- Founded

After looking in a more detailed way, we realize that the foundation year of the companies have a value -1 also we need check for them:

```
sum(Uncleaned_DS_jobs$Founded == -1)
```

```
[1] 118
```

```
Uncleaned_DS_jobs$Founded[Uncleaned_DS_jobs$Founded == -1] <- "No information"
```

```
summary(as.factor(Uncleaned_DS_jobs$Founded), maxsum = 6)
```

```
No information           2012           2011           1996           1999
           118             34             25             22             22
       (Other)
           451
```

From the summary we also fix the problem for Founded.

- Industry

```
sum(Uncleaned_DS_jobs$Industry == -1)
```

```
[1] 71
```

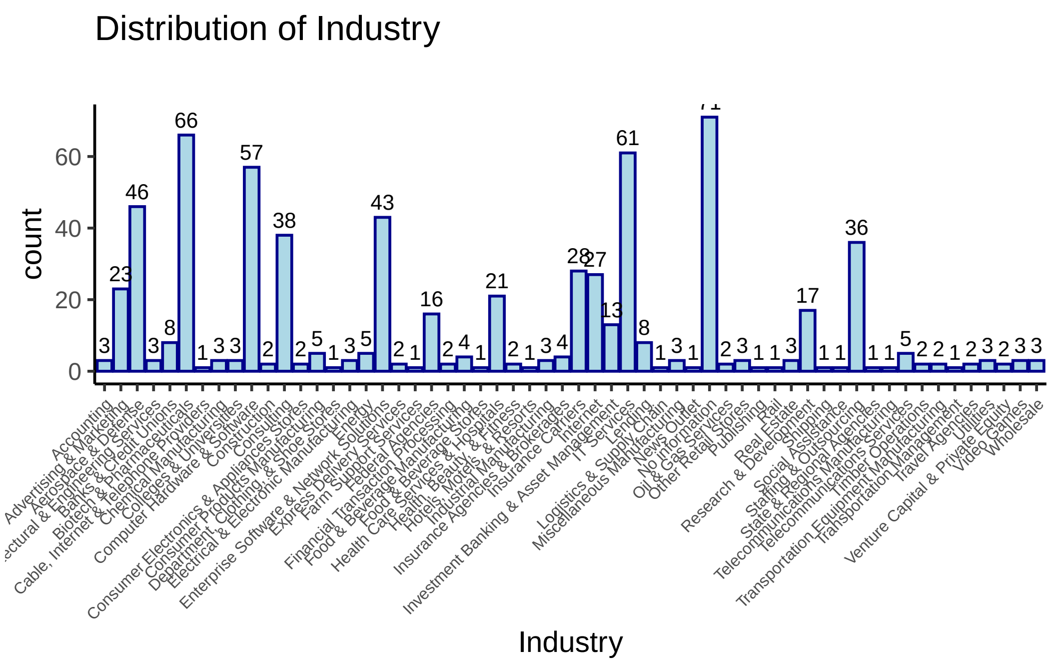
We see that industry has 71 -1 values so, again we assign those values to no information.

```
Uncleaned_DS_jobs$Industry[Uncleaned_DS_jobs$Industry == -1] <- "No information"
```

Histogram of the Industry Variable:

```
industry_plot<- ggplot(Uncleaned_DS_jobs, aes(x=Industry)) +
  labs(title = "Distribution of Industry", x = "Industry", subtitle = "") +
  geom_bar(colour="darkblue", fill="lightblue") +
  geom_text(stat='count', aes(label=..count..), vjust=-0.5,size=2.68) +
  theme_classic()+
   theme(axis.text.x = element_text(size = 6, angle = 45, hjust = 1))
industry_plot
```

```
Warning: The dot-dot notation (`..count..`) was deprecated in ggplot2 3.4.0.
i Please use `after_stat(count)` instead.
```



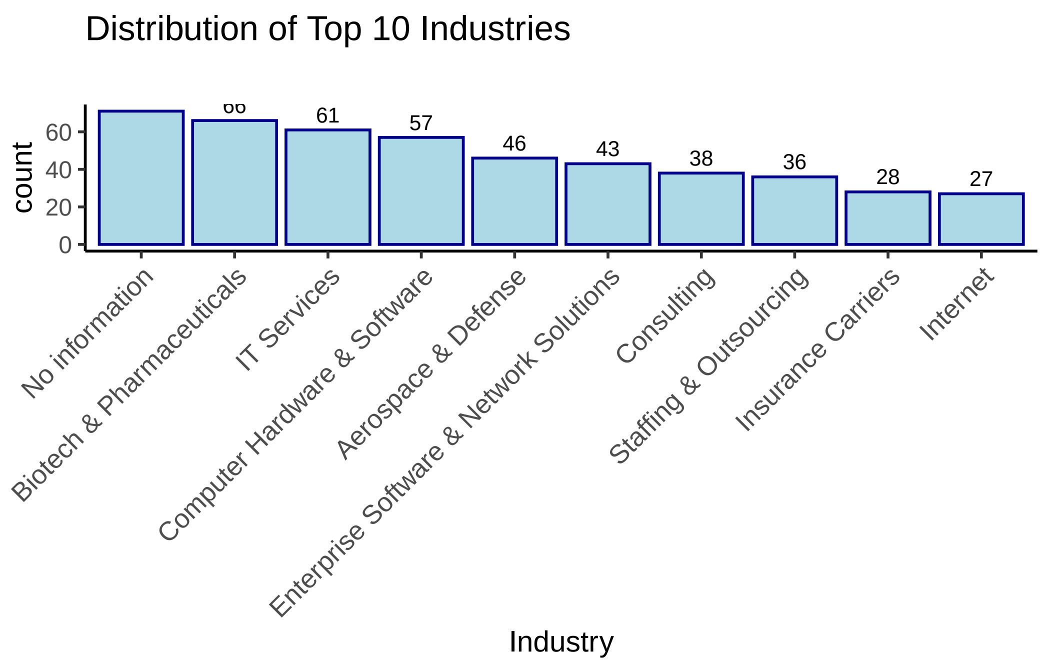
As can be seen from the graph it is hard to read the x axis names so to solve this problem, we picked the 10 industries that have the most frequencies in the data and draw a plot regarding these industries.

```
top10_industries <- Uncleaned_DS_jobs %>%
  group_by(Industry) %>%
  summarise(count = n()) %>%
  arrange(desc(count)) %>%
  top_n(10)
```

```
Selecting by count
```

```
# Reorder the levels of Industry based on frequency
plot_data <-  Uncleaned_DS_jobs
plot_data$Industry <- factor(plot_data$Industry, levels = top10_industries$Industry)

# Filter data to include only the top 10 industries
filtered_data <- plot_data %>%
  filter(Industry %in% top10_industries$Industry)

industry_plot_top10 <- ggplot(filtered_data, aes(x = Industry)) +
```

```r
  labs(title = "Distribution of Top 10 Industries", x = "Industry", subtitle = "") +
  geom_bar(colour = "darkblue", fill = "lightblue") +
  geom_text(stat = 'count', aes(label = ..count..), vjust = -0.5, size = 2.68) +
  theme_classic() +
  theme(axis.text.x = element_text(size = 10, angle = 45, hjust = 1))

industry_plot_top10
```



- Sector

Realizing that sector variable also has -1 values.

```r
sum(Uncleaned_DS_jobs$Sector == -1)
```

```
[1] 71
```

We change -1 values to no information for the sector variable.

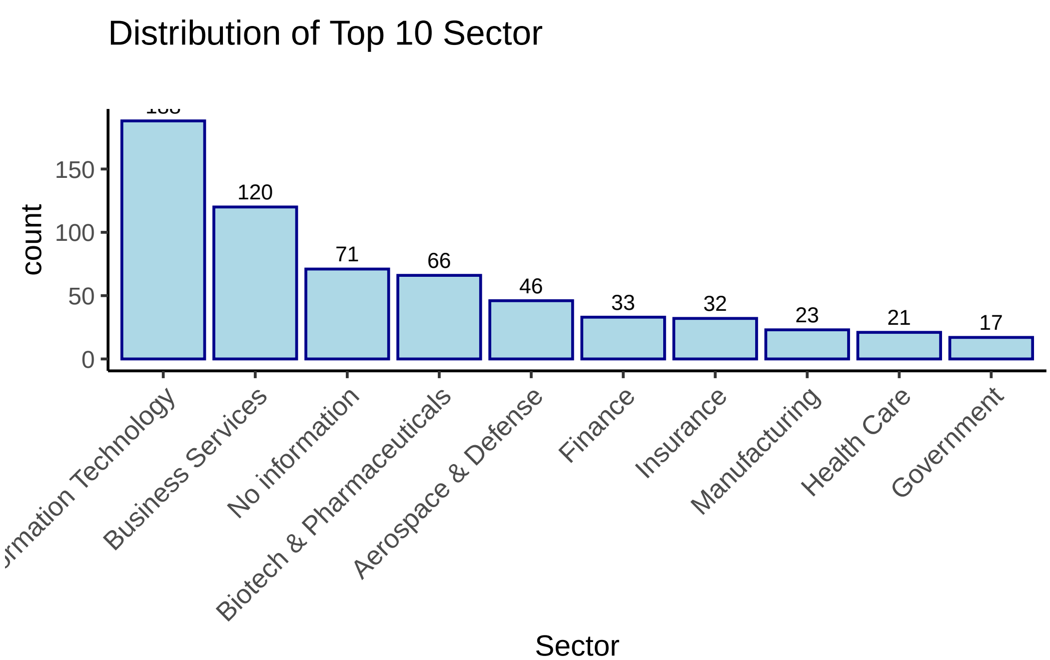
```r
Uncleaned_DS_jobs$Sector[Uncleaned_DS_jobs$Sector == -1] <- "No information"
```

Histogram of the Sector Variable:

```
top10_sector <- Uncleaned_DS_jobs %>%   group_by(Sector) %>%   summarise(count = n()) %>%
```

```
Selecting by count
```

```
# Reorder the levels of Sector based on frequency
plot_data <-  Uncleaned_DS_jobs
plot_data$Sector <- factor(plot_data$Sector, levels = top10_sector$Sector)
# Filter data to include only the top 10 sector
filtered_data <- plot_data %>%
  filter(Sector %in% top10_sector$Sector)
Sector_plot_top10 <- ggplot(filtered_data, aes(x = Sector)) +
  labs(title = "Distribution of Top 10 Sector", x = "Sector", subtitle = "") +
  geom_bar(colour = "darkblue", fill = "lightblue") +
  geom_text(stat = 'count', aes(label = ..count..), vjust = -0.5, size = 2.68) +
  theme_classic() +
  theme(axis.text.x = element_text(size = 10, angle = 45, hjust = 1))
Sector_plot_top10
```



- Revenue

In revenue column there are some -1 values

```
sum(Uncleaned_DS_jobs$Revenue == -1)
```

```
[1] 27
```

And this column has a value called Unknown / Non-Applicable

```
head(Uncleaned_DS_jobs$Revenue)
```

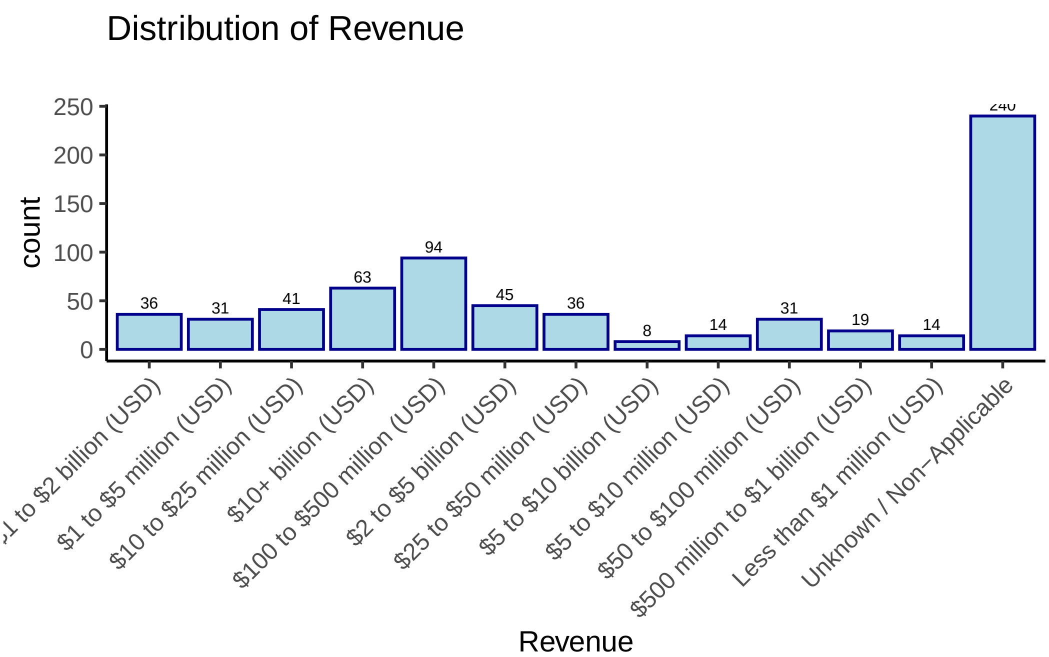
```
[1] "Unknown / Non-Applicable"   "$1 to $2 billion (USD)"
[3] "$100 to $500 million (USD)" "$100 to $500 million (USD)"
[5] "Unknown / Non-Applicable"   "Unknown / Non-Applicable"
```

We convert -1 to that value.

```
Uncleaned_DS_jobs$Revenue[Uncleaned_DS_jobs$Revenue == -1] <- "Unknown / Non-Applicable"
```

Histogram of the Revenue variable:

```
revenue_plot<- ggplot(Uncleaned_DS_jobs, aes(x=Revenue)) +
  labs(title = "Distribution of Revenue", x = "Revenue", subtitle = "") +
  geom_bar(colour="darkblue", fill="lightblue") +
  geom_text(stat='count', aes(label=..count..), vjust=-0.5,size=2.2) +
  theme_classic()+
   theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))
revenue_plot
```



- Company Name

When we check the Company Name variable, we see that it also has the Rating next to it:

```
head(Uncleaned_DS_jobs[, c("Company_Name", "Rating")])
```

```
# A tibble: 6 x 2
  Company_Name                Rating
  <chr>                        <dbl>
1 "Healthfirst\n3.1"             3.1
2 "ManTech\n4.2"                 4.2
3 "Analysis Group\n3.8"          3.8
4 "INFICON\n3.5"                 3.5
5 "Affinity Solutions\n2.9"      2.9
6 "HG Insights\n4.2"             4.2
```

We can separate them and get rid of the Rating variable inside of Company Name to clean this variable. We can do this by `str_replace()` function.

```
Uncleaned_DS_jobs$Company_Name <- str_replace(Uncleaned_DS_jobs$Company_Name, "\n[0-9.]+$"
```

Now we have cleaned the Company Name variable:

```r
head(Uncleaned_DS_jobs[, c("Company_Name", "Rating")])
```

```
# A tibble: 6 x 2
  Company_Name       Rating
  <chr>               <dbl>
1 Healthfirst           3.1
2 ManTech               4.2
3 Analysis Group        3.8
4 INFICON               3.5
5 Affinity Solutions    2.9
6 HG Insights           4.2
```

And we can see that how many of the Company Names:

```r
Uncleaned_DS_jobs |>
  count(Company_Name, sort = TRUE)
```

```
# A tibble: 432 x 2
   Company_Name                    n
   <chr>                       <int>
 1 Hatch Data Inc                 12
 2 Maxar Technologies             12
 3 Tempus Labs                    11
 4 AstraZeneca                    10
 5 Klaviyo                         8
 6 Autodesk                        7
 7 Phoenix Operations Group        7
 8 Novetta                         6
 9 Southwest Research Institute    6
10 MassMutual                      5
# i 422 more rows
```

We can see that the company with the most positions opened is "Hatch Data Inc" and "Maxar Technologies" with 12 positions opened.

Histogram of the Company name:

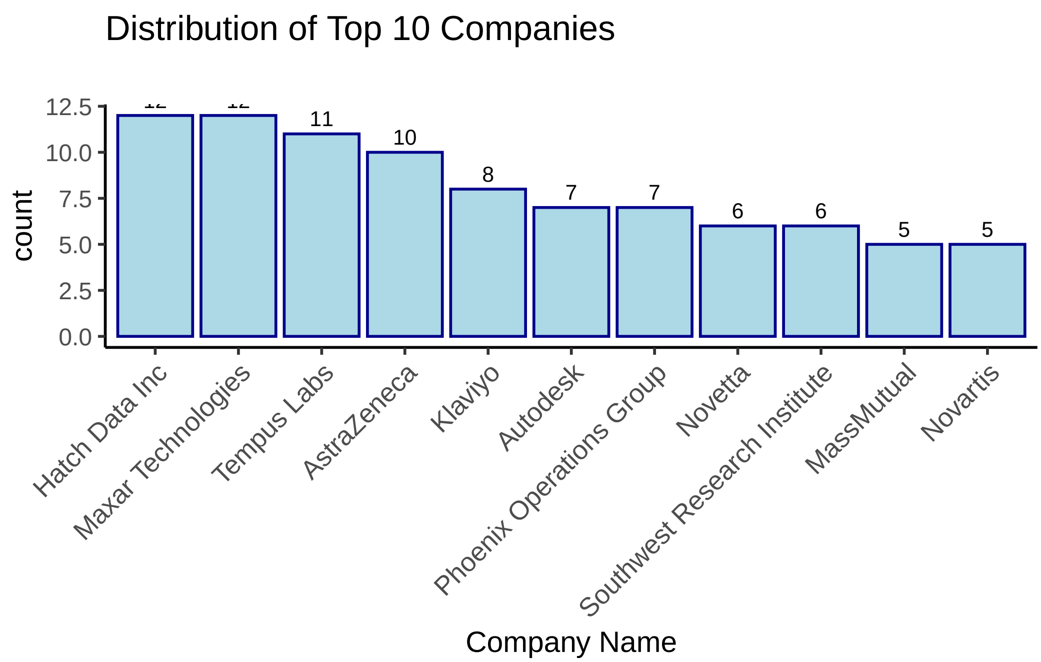
Same problem occurred here like industries so, to see the plot again the 10 company names with the highest frequencies.

Histogram of the company name:

```r
top10_companies <- Uncleaned_DS_jobs %>%
  group_by(Company_Name) %>%
  summarise(count = n()) %>%
  arrange(desc(count)) %>%
  top_n(10)
```

```
Selecting by count
```

```r
# Reorder the levels of Company names based on frequency
plot_data <-  Uncleaned_DS_jobs
plot_data$Company_Name <- factor(plot_data$Company_Name, levels = top10_companies$Company_

# Filter data to include only the top 10 industries
filtered_data <- plot_data %>%
  filter(Company_Name %in% top10_companies$Company_Name)

company_plot_top10 <- ggplot(filtered_data, aes(x = Company_Name)) +
  labs(title = "Distribution of Top 10 Companies", x = "Company Name", subtitle = "") +
  geom_bar(colour = "darkblue", fill = "lightblue") +
  geom_text(stat = 'count', aes(label = ..count..), vjust = -0.5, size = 2.68) +
  theme_classic() +
  theme(axis.text.x = element_text(size = 10, angle = 45, hjust = 1))

company_plot_top10
```



- Size

As can be seen from the summary that we have -1 for the Size. But we have unknown category for this variable.

```
summary(as.factor(Uncleaned_DS_jobs$Size))
```

```
                     -1        1 to 50 employees        10000+ employees
                     27                       86                      80
 1001 to 5000 employees    201 to 500 employees 5001 to 10000 employees
                    104                       85                      61
  501 to 1000 employees     51 to 200 employees                 Unknown
                     77                      135                      17
```

So we can assign "-1" to "Unknown" category for this variable:

```
Uncleaned_DS_jobs$Size[Uncleaned_DS_jobs$Size == -1] <- "Unknown"
```

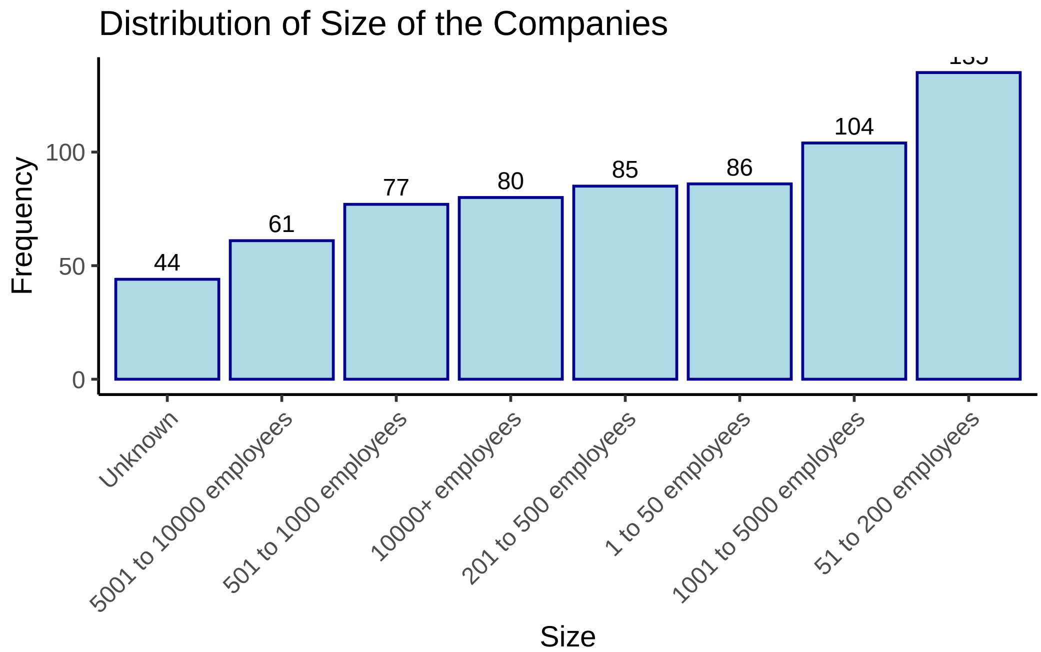
```
summary(as.factor(Uncleaned_DS_jobs$Size))
```

```
      1 to 50 employees        10000+ employees  1001 to 5000 employees
```

```
                     86                       80                      104
201 to 500 employees 5001 to 10000 employees   501 to 1000 employees 
                     85                       61                       77
 51 to 200 employees                  Unknown 
                    135                       44 
```

Histogram of the Size variable:

```
library(dplyr)

# Create a frequency table for Size
size_counts <- count(Uncleaned_DS_jobs, Size)

# Sort the data by count in ascending order
size_counts <- arrange(size_counts, desc(n))

# Create the plot
size_plot <- ggplot(size_counts, aes(x = reorder(Size, n), y = n)) +
  labs(title = "Distribution of Size of the Companies", x = "Size", y = "Frequency") +
  geom_col(colour = "darkblue", fill = "lightblue") +
  geom_text(aes(label = n), vjust = -0.5, size = 3) +
  theme_classic() +
  theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))

size_plot
```



- Competitors

There are -1 values in Competitors. We don't know their competitors' name so we can attribute them to no information

```
Uncleaned_DS_jobs$Competitors[Uncleaned_DS_jobs$Competitors == "-1"] <- "No information"
```

```
summary(as.factor(Uncleaned_DS_jobs$Competitors), maxsum = 6)
```

```
                                           No information
                                                      501
                             Roche, GlaxoSmithKline, Novartis
                                                       10
               Leidos, CACI International, Booz Allen Hamilton
                                                        6
Los Alamos National Laboratory, Battelle, SRI International
                                                        6
                           Battelle, General Atomics, SAIC
                                                        3
                                                  (Other)
                                                      146
```

- Location

Let's see the location variable first.

```
summary(as.factor(Uncleaned_DS_jobs$Location), maxsum = 6)
```

```
San Francisco, CA       New York, NY    Washington, DC        Boston, MA
               69                 50                26                24
      Chicago, IL            (Other)
               22                481
```

In the Location variable, we can see that they are written with the state which they are in. So we want to separate them. But,before that, in our data there are some problematic rows:

Some rows are too short. Let's see that columns:

```
Uncleaned_DS_jobs %>%
  filter(
    str_count(Location, ",\\s+") != 1
    ) %>%
  select(Location) %>% distinct_all()
```

```
# A tibble: 7 x 1
  Location
  <chr>
1 Remote
2 United States
3 Utah
4 New Jersey
5 Texas
6 Patuxent, Anne Arundel, MD
7 California
```

From this output, we can see that we have "Remote", "United States", locations that have the same names as their states; "Utah", "New Jersey", "Texas" and "California", and "Patuxent, Anne Arundel, MD" which is a region for the Anne Arundel county. So, we will add information for this columns firstly, then we will separate the Location and States. For this, we will use `str_replace()` function.

```
# Define replacements using case_when
Uncleaned_DS_jobs <- Uncleaned_DS_jobs %>%
  mutate(
    Location = case_when(
```

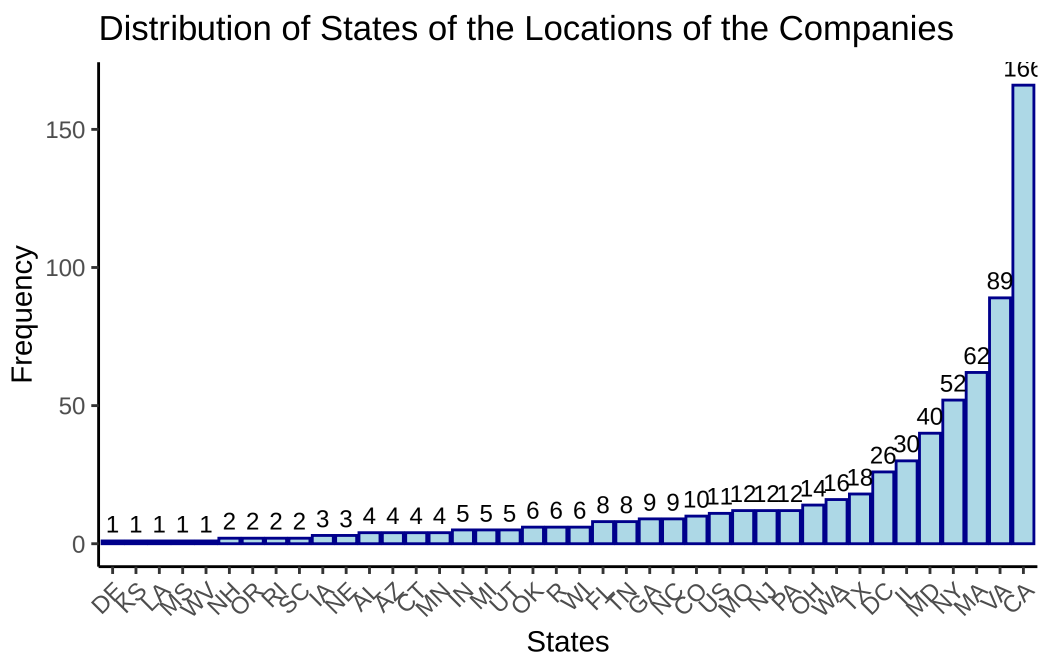
```
      Location == "Remote" ~ str_replace(Location, "Remote", "Remote, R"),
      Location == "United States" ~ str_replace(Location, "United States", "United States,
      Location == "Utah" ~ str_replace(Location, "Utah", "Utah, UT"),
      Location == "New Jersey" ~ str_replace(Location, "New Jersey", "New Jersey, NJ"),
      Location == "Texas" ~ str_replace(Location, "Texas", "Texas, TX"),
      Location == "California" ~ str_replace(Location, "California", "California, CA"),
      Location == "Patuxent, Anne Arundel, MD" ~ str_replace(Location, "Patuxent, Anne Aru
      TRUE ~ Location
    )
  )
```

Now we can separate them:

```
Uncleaned_DS_jobs <- Uncleaned_DS_jobs %>%
  separate(
    Location,
    into = c("Location", "Location_State"),
    sep = ",\\s+")
```

Let's visualize Location_State variable:

```
locationstates_counts <- count(Uncleaned_DS_jobs, Location_State)

# Sort the data by count in ascending order
locationstates_counts <- arrange(locationstates_counts, desc(n))

# Create the plot
locationstate_plot <- ggplot(locationstates_counts, aes(x = reorder(Location_State, n), y
  labs(title = "Distribution of States of the Locations of the Companies", x = "States", y
  geom_col(colour = "darkblue", fill = "lightblue") +
  geom_text(aes(label = n), vjust = -0.5, size = 3) +
  theme_classic() +
  theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))
locationstate_plot
```



- Headquarters

For the headquarters we have -1 values also,

```
sum(Uncleaned_DS_jobs$Headquarters == -1)
```

```
[1] 31
```

```
Uncleaned_DS_jobs$Headquarters[Uncleaned_DS_jobs$Headquarters == "-1"] <- "No information"
```

```
summary(as.factor(Uncleaned_DS_jobs$Headquarters), maxsum = 6)
```

```
   New York, NY    No information San Francisco, CA       Chicago, IL
             33                31                31                23
     Boston, MA           (Other)
             19               535
```

Similar to Location, we want to separate the states and the city. We will apply the similar approach to fix this column.

```r
Uncleaned_DS_jobs %>%
  filter(
    str_count(Headquarters, ",\\s+") != 1
    ) %>%
  select(Location) %>% distinct_all()
```

```
# A tibble: 14 x 1
   Location
   <chr>
 1 Hauppauge
 2 Reston
 3 New York
 4 Palo Alto
 5 San Francisco
 6 Brooklyn
 7 Sterling
 8 Chantilly
 9 Cambridge
10 Omaha
11 Fort Belvoir
12 Naperville
13 Redmond
14 Irwindale
```

We will assign the states for this rows too. and also to not get a warning regarding the NA values later, "No Information" is also going to be fixed:

```r
Uncleaned_DS_jobs <- Uncleaned_DS_jobs %>%
  mutate(
    Headquarters = case_when(
      Headquarters == "Hauppauge" ~ str_replace(Headquarters, "Hauppauge", "Hauppauge, NY"
      Headquarters == "Reston" ~ str_replace(Headquarters, "Reston", "Reston, VA"),
      Headquarters == "New York" ~ str_replace(Headquarters, "New York", "New York, NY"),
      Headquarters == "Palo Alto" ~ str_replace(Headquarters, "Palo Alto", "Palo Alto, CA"
      Headquarters == "San Francisco" ~ str_replace(Headquarters, "San Francisco", "San Fr
      Headquarters == "Brooklyn" ~ str_replace(Headquarters, "Brooklyn", "Brooklyn, NY"),
      Headquarters == "Sterling" ~ str_replace(Headquarters, "Sterling", "Sterling, IL"),
      Headquarters == "Chantilly" ~ str_replace(Headquarters, "Chantilly", "Chantilly, VA"
      Headquarters == "Cambridge" ~ str_replace(Headquarters, "Cambridge", "Cambridge, MA"
      Headquarters == "Fort Belvoir" ~ str_replace(Headquarters, "Fort Belvoir", "Fort Bel
      Headquarters == "Naperville" ~ str_replace(Headquarters, "Naperville", "Naperville,
```

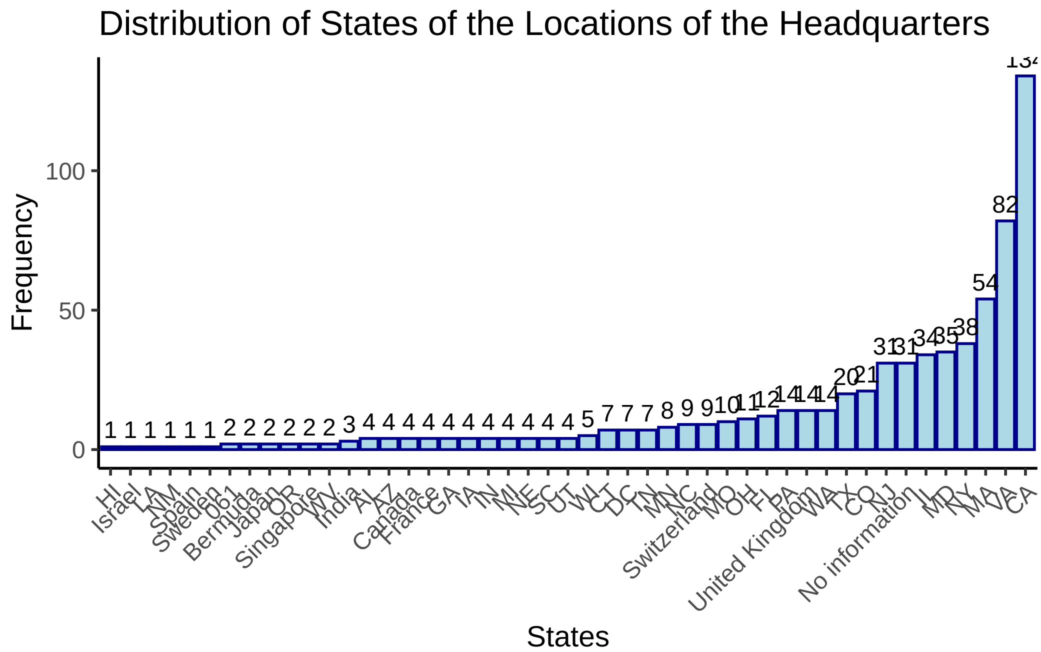
```
      Headquarters == "Redmond" ~ str_replace(Headquarters, "Redmond", "Redmond, WA"),
      Headquarters == "Irwindale" ~ str_replace(Headquarters, "Irwindale", "Irwindale, CA"
      Headquarters == "No information" ~ str_replace(Headquarters, "No information", "No i
      TRUE ~ Headquarters
    )
  )
```

And finally, we will separate the Headquarters and their states:

```
Uncleaned_DS_jobs <- Uncleaned_DS_jobs %>%
  separate(
    Headquarters,
    into = c("Headquarters", "Headquarters_State"),
    sep = ",\\s+")
```

Let's visualize Headquarters States:

```
hqstates_counts <- count(Uncleaned_DS_jobs, Headquarters_State)

# Sort the data by count in ascending order
hqstates_counts <- arrange(hqstates_counts, desc(n))

# Create the plot
hqstate_plot <- ggplot(hqstates_counts, aes(x = reorder(Headquarters_State, n), y = n)) +
  labs(title = "Distribution of States of the Locations of the Headquarters", x = "States"
  geom_col(colour = "darkblue", fill = "lightblue") +
  geom_text(aes(label = n), vjust = -0.5, size = 3) +
  theme_classic() +
  theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))
hqstate_plot
```



- Type Of Ownership

There are -1 values in the Type of Ownership also.

We check how many -1 values are in the variable.

```
sum(Uncleaned_DS_jobs$Type_of_Ownership == -1)
```

```
[1] 27
```

```
Uncleaned_DS_jobs$Type_of_Ownership[Uncleaned_DS_jobs$Type_of_Ownership == "-1"] <- "Unkno
```

Let's see the summary:

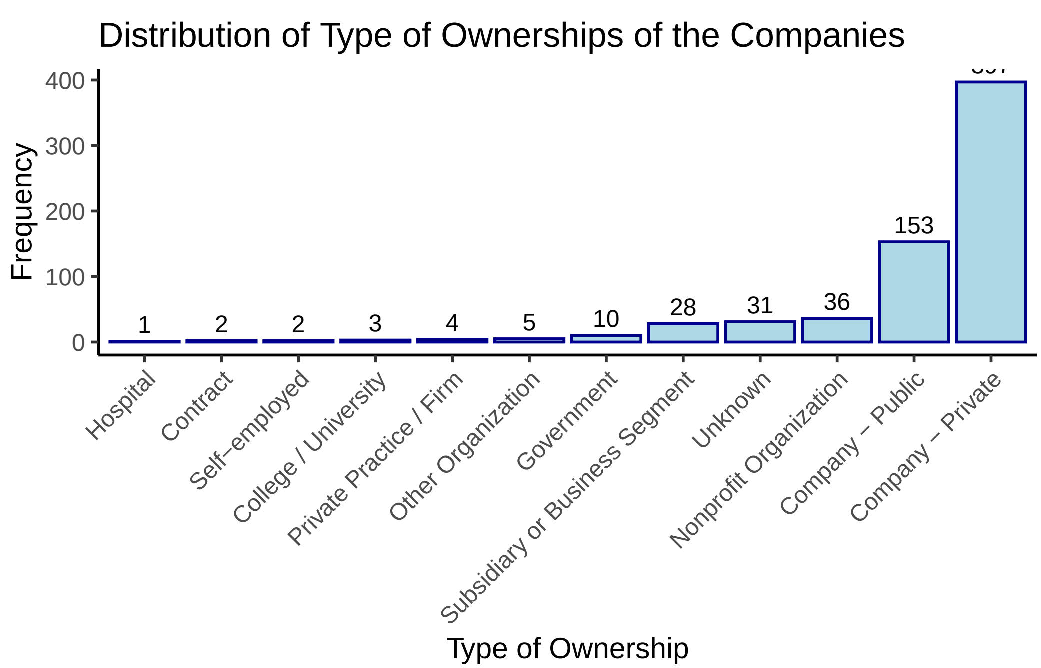
```
summary(as.factor(Uncleaned_DS_jobs$Type_of_Ownership))
```

```
        College / University              Company - Private
                           3                            397
            Company - Public                       Contract
                         153                              2
                  Government                       Hospital
                          10                              1
```

```
        Nonprofit Organization               Other Organization
                            36                                5
        Private Practice / Firm                   Self-employed
                             4                                2
Subsidiary or Business Segment                          Unknown
                            28                               31
```

Let's visualize type of ownerships of the companies:

```
Type_of_Ownership_counts <- count(Uncleaned_DS_jobs, Type_of_Ownership)

# Sort the data by count in ascending order
Type_of_Ownership_counts <- arrange(Type_of_Ownership_counts, desc(n))

# Create the plot
tow_plot <- ggplot(Type_of_Ownership_counts, aes(x = reorder(Type_of_Ownership, n), y = n)
  labs(title = "Distribution of Type of Ownerships of the Companies", x = "Type of Ownersh
  geom_col(colour = "darkblue", fill = "lightblue") +
  geom_text(aes(label = n), vjust = -0.5, size = 3) +
  theme_classic() +
  theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))
tow_plot
```



- Salary Estimation

For Salary Estimate column, let's see the unique values we have:

```
levels(as.factor(Uncleaned_DS_jobs$Salary_Estimate))
```

```
 [1] "$101K-$165K (Glassdoor est.)" "$105K-$167K (Glassdoor est.)"
 [3] "$110K-$163K (Glassdoor est.)" "$112K-$116K (Glassdoor est.)"
 [5] "$122K-$146K (Glassdoor est.)" "$124K-$198K (Glassdoor est.)"
 [7] "$128K-$201K (Glassdoor est.)" "$137K-$171K (Glassdoor est.)"
 [9] "$138K-$158K (Glassdoor est.)" "$141K-$225K (Glassdoor est.)"
[11] "$145K-$225K(Employer est.)"   "$212K-$331K (Glassdoor est.)"
[13] "$31K-$56K (Glassdoor est.)"   "$56K-$97K (Glassdoor est.)"
[15] "$66K-$112K (Glassdoor est.)"  "$69K-$116K (Glassdoor est.)"
[17] "$71K-$123K (Glassdoor est.)"  "$75K-$131K (Glassdoor est.)"
[19] "$79K-$106K (Glassdoor est.)"  "$79K-$131K (Glassdoor est.)"
[21] "$79K-$133K (Glassdoor est.)"  "$79K-$147K (Glassdoor est.)"
[23] "$80K-$132K (Glassdoor est.)"  "$87K-$141K (Glassdoor est.)"
[25] "$90K-$109K (Glassdoor est.)"  "$90K-$124K (Glassdoor est.)"
[27] "$91K-$150K (Glassdoor est.)"  "$92K-$155K (Glassdoor est.)"
[29] "$95K-$119K (Glassdoor est.)"  "$99K-$132K (Glassdoor est.)"
```

```
sum(is.na(as.factor(Uncleaned_DS_jobs$Salary_Estimate)))
```

```
[1] 0
```

From this output, we can see that we have common shape for the salary estimates with 0 NA values. We can separate this column into two separate columns for obtaining lower and upper limits for the salary estimates.

```
# Remove spaces in the column
Uncleaned_DS_jobs$Salary_Estimate_wo_spaces <- Uncleaned_DS_jobs$Salary_Estimate

Uncleaned_DS_jobs$Salary_Estimate_wo_spaces <- gsub(" ", "",Uncleaned_DS_jobs$Salary_Estim
# Display the updated data frame
head(Uncleaned_DS_jobs$Salary_Estimate_wo_spaces)
```

```
[1] "$137K-$171K(Glassdoorest.)" "$137K-$171K(Glassdoorest.)"
[3] "$137K-$171K(Glassdoorest.)" "$137K-$171K(Glassdoorest.)"
[5] "$137K-$171K(Glassdoorest.)" "$137K-$171K(Glassdoorest.)"
```

Now we have no blank space between the words.

Let's get rid of the parts at the end; Glassdoor est. and Employer est.

```r
Uncleaned_DS_jobs$Salary_Estimate_wo_spaces <-
  gsub("K\\(Glassdoorest.\\)",
       "",
       Uncleaned_DS_jobs$Salary_Estimate_wo_spaces)
```

```r
Uncleaned_DS_jobs$Salary_Estimate_wo_spaces <-
  gsub("K\\(Employerest.\\)",
       "",
       Uncleaned_DS_jobs$Salary_Estimate_wo_spaces)

head(Uncleaned_DS_jobs$Salary_Estimate_wo_spaces)
```

```
[1] "$137K-$171" "$137K-$171" "$137K-$171" "$137K-$171" "$137K-$171"
[6] "$137K-$171"
```

Let's see how we can select the numbers that are remaining in the rows: we can use [0-9]+ for this part:

```r
str_view(
  Uncleaned_DS_jobs$Salary_Estimate_wo_spaces,
  "[0-9]+")
```

```
 [1] | $<137>K-$<171>
 [2] | $<137>K-$<171>
 [3] | $<137>K-$<171>
 [4] | $<137>K-$<171>
 [5] | $<137>K-$<171>
 [6] | $<137>K-$<171>
 [7] | $<137>K-$<171>
 [8] | $<137>K-$<171>
 [9] | $<137>K-$<171>
[10] | $<137>K-$<171>
[11] | $<137>K-$<171>
[12] | $<137>K-$<171>
[13] | $<137>K-$<171>
[14] | $<137>K-$<171>
[15] | $<137>K-$<171>
```

```
[16] | $<137>K-$<171>
[17] | $<137>K-$<171>
[18] | $<137>K-$<171>
[19] | $<137>K-$<171>
[20] | $<137>K-$<171>
... and 652 more
```

```r
Uncleaned_DS_jobs <- Uncleaned_DS_jobs |>
  separate_wider_regex(
    Salary_Estimate_wo_spaces,
    patterns = c(
      "\\$",
      Low_Limit_For_Salary = "[0-9]+",
      "K-\\$",
      High_Limit_For_Salary = "[0-9]+"
      )
    )
```

By using `separate_wider_regex()` function, we defined the pattern in the data, and we got the new columns as Low_Limit_For_Salary and High_Limit_For_Salary as we wished.

```r
head(Uncleaned_DS_jobs[c("Salary_Estimate",
                         "Low_Limit_For_Salary",
                         "High_Limit_For_Salary")])
```

```
# A tibble: 6 x 3
  Salary_Estimate             Low_Limit_For_Salary High_Limit_For_Salary
  <chr>                       <chr>                <chr>
1 $137K-$171K (Glassdoor est.) 137                 171
2 $137K-$171K (Glassdoor est.) 137                 171
3 $137K-$171K (Glassdoor est.) 137                 171
4 $137K-$171K (Glassdoor est.) 137                 171
5 $137K-$171K (Glassdoor est.) 137                 171
6 $137K-$171K (Glassdoor est.) 137                 171
```

For not confusing the numbers later, lets multiply the low limit and high limit numbers with 1000 and make Salary Estimate factor.

```r
Uncleaned_DS_jobs <- Uncleaned_DS_jobs %>%
  mutate(
    Low_Limit_For_Salary = as.numeric(Low_Limit_For_Salary)*1000,
```

```
    High_Limit_For_Salary = as.numeric(High_Limit_For_Salary)*1000)
```

```
Uncleaned_DS_jobs$Salary_Estimate <- as.factor(Uncleaned_DS_jobs$Salary_Estimate)

head(Uncleaned_DS_jobs[c("Salary_Estimate",
                         "Low_Limit_For_Salary",
                         "High_Limit_For_Salary")])
```

```
# A tibble: 6 x 3
  Salary_Estimate             Low_Limit_For_Salary High_Limit_For_Salary
  <fct>                                      <dbl>                 <dbl>
1 $137K-$171K (Glassdoor est.)              137000                171000
2 $137K-$171K (Glassdoor est.)              137000                171000
3 $137K-$171K (Glassdoor est.)              137000                171000
4 $137K-$171K (Glassdoor est.)              137000                171000
5 $137K-$171K (Glassdoor est.)              137000                171000
6 $137K-$171K (Glassdoor est.)              137000                171000
```

- Job Title

For Job Title column, first let's examine it:

```
glimpse(
  as.factor(
    Uncleaned_DS_jobs$Job_Title))
```

```
 Factor w/ 172 levels "(Sr.) Data Scientist -",..: 156 50 50 50 50 50 65 50 165 50 ...
```

As we can see, we have 172 different levels for Job Titles. We can try to group them by searching common words.

```
head(
  levels(
    as.factor(
      Uncleaned_DS_jobs$Job_Title)))
```

```
[1] "(Sr.) Data Scientist -"
[2] "AI Data Scientist"
[3] "AI Ops Data Scientist"
```

```
[4] "AI/ML - Machine Learning Scientist, Siri Understanding"
[5] "Analytics - Business Assurance Data Analyst"
[6] "Analytics Manager"
```

But before that, we can see that some columns have "Senior", "Experience" words. By using this information, we can create a new column for seniority of the job.

By using `str_view()` function, first, let's see that columns;

```
str_view(
  Uncleaned_DS_jobs$Job_Title,
  regex("^Senior|^Sr|^Experience",
        multiline = TRUE))
```

```
  [1] | <Sr> Data Scientist
 [16] | <Experience>d Data Scientist
 [34] | <Senior> Research Statistician- Data Scientist
 [40] | <Senior> Analyst/Data Scientist
 [47] | <Senior> Data Scientist
 [57] | <Senior> Data Scientist
 [93] | <Senior> Data Scientist
 [99] | <Senior> Data Scientist
[104] | <Senior> Data Scientist
[107] | <Sr> Data Engineer (Sr BI Developer)
[122] | <Senior> Data Engineer
[123] | <Senior> Data Scientist
[126] | <Sr>. ML/Data Scientist - AI/NLP/Chatbot
[130] | <Sr>. ML/Data Scientist - AI/NLP/Chatbot
[132] | <Senior> Data Engineer
[137] | <Senior> Data Engineer
[143] | <Senior> Data Scientist
[154] | <Sr> Scientist - Extractables & Leachables
[156] | <Sr> Data Scientist
[158] | <Experience>d Data Scientist
... and 51 more
```

By using `str_detect()` function, we can detect the rows including "Senior", "Sr", "Experienced" words. This function returns TRUE if they exist, and returns FALSE if they don't exist.

By using `as.integer()` , we assign 1 to exists and 0 to nonexistent.

```
Uncleaned_DS_jobs$Senior_Position <- as.integer(
  str_detect(Uncleaned_DS_jobs$Job_Title,
             regex("^(Senior|Sr|Experienced)")
             ) )
```

Now that we defined the senior roles, we can assign the same titles to same jobs.

Let's start by Data Scientist. Let's find the columns including Data Scientist word.

```
str_view(Uncleaned_DS_jobs$Job_Title,
         regex(".*Data\\s+Scientist.*",
         multiline = TRUE))
```

```
 [1] | <Sr Data Scientist>
 [2] | <Data Scientist>
 [3] | <Data Scientist>
 [4] | <Data Scientist>
 [5] | <Data Scientist>
 [6] | <Data Scientist>
 [7] | <Data Scientist / Machine Learning Expert>
 [8] | <Data Scientist>
 [9] | <Staff Data Scientist - Analytics>
[10] | <Data Scientist>
[11] | <Data Scientist>
[12] | <Data Scientist>
[13] | <Data Scientist - Statistics, Early Career>
[15] | <Data Scientist>
[16] | <Experienced Data Scientist>
[17] | <Data Scientist - Contract>
[18] | <Data Scientist>
[21] | <Data Scientist>
[22] | <Data Scientist/Machine Learning>
[25] | <Data Scientist>
... and 435 more
```

Bu using `str_replace_all()` we can replace all the rows including Data Scientist word in some way, directly with "Data Scientist".

```
Uncleaned_DS_jobs$Job_Title <-
  str_replace_all(
    Uncleaned_DS_jobs$Job_Title,
    ".*Data\\s+Scientist.*",
```

```
    "Data Scientist")
```

Now let's get Data Analyst titles:

```
str_view(
  Uncleaned_DS_jobs$Job_Title,
  regex(".*Data\\s+Analyst.*",
        multiline = TRUE,
        ignore_case = TRUE))
```

```
 [19] | <Data Analyst II>
 [41] | <Data Analyst>
 [43] | <Data Analyst I>
 [51] | <Data Analyst>
 [55] | <E-Commerce Data Analyst>
 [61] | <Data Analyst>
 [65] | <Global Data Analyst>
 [74] | <Business Data Analyst>
 [76] | <Data Analyst>
 [87] | <Data Analyst>
[111] | <RFP Data Analyst>
[112] | <Data Analyst>
[118] | <Data Analyst/Engineer>
[139] | <Data Analyst>
[162] | <Say Business Data Analyst>
[164] | <Data Analyst>
[167] | <Senior Data Analyst>
[168] | <Senior Data Analyst>
[170] | <Sr Data Analyst>
[175] | <Data Analyst>
... and 27 more
```

```
Uncleaned_DS_jobs$Job_Title <-
  str_replace_all(
    Uncleaned_DS_jobs$Job_Title,
    ".*Data\\s+Analyst.*",
    "Data Analyst")
```

Now let's get Data Engineer titles:

```
str_view(
  Uncleaned_DS_jobs$Job_Title,
  regex(".*Data\\s+Engineer.*",
        multiline = TRUE,
        ignore_case = TRUE))
```

```
 [35] │ <Data Engineer>
 [54] │ <Jr. Data Engineer>
 [66] │ <Data Engineer>
 [71] │ <Data Engineer (Remote)>
 [77] │ <Data Engineer, Enterprise Analytics>
 [83] │ <Data Engineer>
[107] │ <Sr Data Engineer (Sr BI Developer)>
[108] │ <Data Engineer>
[114] │ <Data Engineer>
[120] │ <Data Engineer>
[122] │ <Senior Data Engineer>
[124] │ <Data Engineer>
[129] │ <Data Engineer>
[132] │ <Senior Data Engineer>
[133] │ <Data Engineer>
[137] │ <Senior Data Engineer>
[140] │ <Data Engineer>
[142] │ <Tableau Data Engineer 20-0117>
[153] │ <Data Engineer>
[169] │ <Data Engineer>
... and 27 more
```

```
Uncleaned_DS_jobs$Job_Title <-
  str_replace_all(
    Uncleaned_DS_jobs$Job_Title,
    ".*Data\\s+Engineer.*",
    "Data Engineer")
```

And Machine Learning Engineers:

```
str_view(
  Uncleaned_DS_jobs$Job_Title,
  regex(".*Machine\\s+Learning.*",
        multiline = TRUE,
        ignore_case = TRUE))
```

```
 [42] | <Machine Learning Engineer>
 [46] | <Computational Scientist, Machine Learning>
 [62] | <Machine Learning Engineer>
 [69] | <Data & Machine Learning Scientist>
 [89] | <Machine Learning Engineer>
 [92] | <Machine Learning Engineer>
[102] | <Machine Learning Engineer>
[135] | <Machine Learning Engineer>
[136] | <Machine Learning Engineer>
[145] | <Machine Learning Engineer>
[159] | <Machine Learning Engineer>
[172] | <Machine Learning Scientist - Bay Area, CA>
[176] | <Senior Data & Machine Learning Scientist>
[180] | <Machine Learning Engineer>
[191] | <Principal Machine Learning Scientist>
[203] | <Machine Learning Engineer>
[220] | <Senior Machine Learning Scientist - Bay Area, CA>
[229] | <Machine Learning Engineer>
[261] | <Principal Machine Learning Scientist>
[331] | <Machine Learning Engineer>
... and 16 more
```

```
Uncleaned_DS_jobs$Job_Title <-  str_replace_all(
  Uncleaned_DS_jobs$Job_Title,
  ".*Machine\\s+Learning.*",
  "Machine Learning Engineer")
```

Let's examine Managers this time:

```
str_view(
  Uncleaned_DS_jobs$Job_Title,
  regex(".*Analytics\\s+Manager.*|.*Data\\s+Science\\sManager.*|.*Director.*|.*Vice\\sPres
        multiline = TRUE,
        ignore_case = TRUE))
```

```
 [86] | <Data Science Manager, Payment Acceptance - USA>
[150] | <Analytics Manager>
[198] | <Principal Scientist/Associate Director, Quality Control and Analytical Technologies
[218] | <Analytics Manager - Data Mart>
[266] | <Director of Data Science>
[272] | <Manager / Lead, Data Science & Analytics>
```

```
[313] | <Principal Scientist/Associate Director, Quality Control and Analytical Technologies>
[332] | <Principal Data & Analytics Platform Engineer>
[343] | <VP, Data Science>
[381] | <Analytics Manager - Data Mart>
[470] | <VP, Data Science>
[523] | <Manager, Field Application Scientist, Southeast>
[564] | <Data Science Manager>
[581] | <Vice President, Biometrics and Clinical Data Management>
```

Let's replace them with "Data Science and Analytics Manager"

```
Uncleaned_DS_jobs$Job_Title <-
  str_replace_all(
    Uncleaned_DS_jobs$Job_Title,
    ".*Analytics\\s+Manager.*|.*Data\\s+Science\\sManager.*|.*Director.*|.*Vice\\sPresiden
    "Data Science and Analytics Manager")
```

Now, bu using `str_view()` function, we want to see all the jobs that have "Data" in it, but not "Data Analyst", "Data Scientist,"Data Engineer" or "Data Science and Analytics Manager" because we already took care of that titles.

```
str_view(
  Uncleaned_DS_jobs$Job_Title,
  regex("^(?!.*Data\\s+Analyst.*|.*Data\\s+Scientist.*|.*Data\\s+Science\\s+and\\s+Analyti
        ignore_case = TRUE))
```

```
 [14] | <Data Modeler>
 [24] | <Business Intelligence Analyst I- Data Insights>
 [56] | <Data Analytics Engineer>
 [97] | <Data Analytics Engineer>
[117] | <Software Engineer - Data Science>
[141] | <Data Integration and Modeling Engineer>
[187] | <Production Engineer - Statistics/Data Analysis>
[207] | <Data Science Instructor>
[214] | <Data Science Software Engineer>
[219] | <Data Modeler (Analytical Systems)>
[230] | <Equity Data Insights Analyst - Quantitative Analyst>
[257] | <Environmental Data Science>
[370] | <Data Science Software Engineer>
[382] | <Data Modeler (Analytical Systems)>
[388] | <IT Partner Digital Health Technology and Data Science>
[396] | <Data Solutions Engineer - Data Modeler>
```

```
[519] | <Data Science Software Engineer>
[540] | <Data Science Analyst>
[545] | <Data Modeler (Analytical Systems)>
[555] | <IT Partner Digital Health Technology and Data Science>
... and 3 more
```

We will save these as "Other Data Positions"

```
Uncleaned_DS_jobs$Job_Title <-
  str_replace_all(
    Uncleaned_DS_jobs$Job_Title,
     regex("(?!Data\\s+(Analyst|Scientist|Engineer|Science\\s+and\\s+Analytics\\s+Manager)
    "Other Data Positions" )
```

Finally, we will save all the jobs that are not include "Data" word in it and not "Machine Learning Engineer" into "Others" category because there are a lot of jobs with the titles like Scientist, Researcher etc.

```
Uncleaned_DS_jobs$Job_Title <-
  str_replace_all(
    Uncleaned_DS_jobs$Job_Title,
    regex("^(?!.*(Data|Machine\\s+Learning\\s+Engineer)).*$"),
    "Others")
```

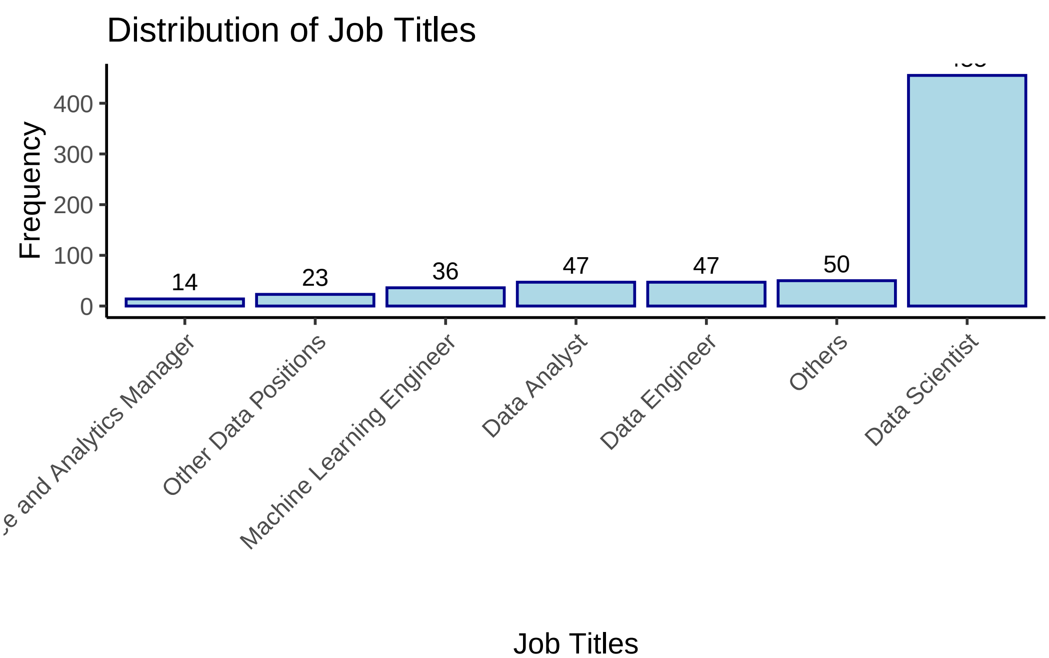
Finally, let's see our clean job titles:

```
Uncleaned_DS_jobs$Job_Title <- as.factor(Uncleaned_DS_jobs$Job_Title)

summary(Uncleaned_DS_jobs$Job_Title)
```

```
                    Data Analyst                    Data Engineer
                              47                               47
Data Science and Analytics Manager                   Data Scientist
                              14                              455
         Machine Learning Engineer             Other Data Positions
                              36                               23
                          Others
                              50
```

Let's visualize Job Titles:

```r
jt_counts <- count(Uncleaned_DS_jobs, Job_Title)

# Sort the data by count in ascending order
jt_counts <- arrange(jt_counts, desc(n))

jt_plot <- ggplot(jt_counts, aes(x = reorder(Job_Title, n), y = n)) +
   labs(title = "Distribution of Job Titles", x = "Job Titles", y = "Frequency") +
   geom_col(colour = "darkblue", fill = "lightblue") +
   geom_text(aes(label = n), vjust = -0.5, size = 3) +
   theme_classic() +
   theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))
jt_plot
```



- Job Description

When we look at the job description column,

We have so many different values but we can differentiate them into other columns like we can say that a job wants the skill SQL.

First, we need to look the job description column in a detailed way.

```r
head(Uncleaned_DS_jobs$Job_Description,1)
```

```
[1] "Description\n\nThe Senior Data Scientist is responsible for defining, building, and imp
```

We see some common requirements and common job descriptions for jobs.

For this we can separate the columns like SQL and we can say that this jobs wants an SQL bu using factor 1 or 0.

Let's start with SQL:

In this we should check if SQL is mentioned in the variable Job_Description

```
sql_mentioned <- function(description) {
  # We use tolower to match the SQL in the job description
  description <- tolower(description)

  # Check if SQL is mentioned
  if (grepl("\\bsql\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

Now we need to create a column called SQL, in this column we will see if SQL is a requirement in the job description or not.

```
Uncleaned_DS_jobs$sql_needed <- sapply(Uncleaned_DS_jobs$Job_Description, sql_mentioned)
```

```
Uncleaned_DS_jobs$sql_needed <- as.factor(Uncleaned_DS_jobs$sql_needed)
```

Now for Python we repeat the same process.

```
python_mentioned <- function(description) {
  # We use tolower to match the python in the job description
  description <- tolower(description)

  # Check if python is mentioned
  if (grepl("\\bpython\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```r
Uncleaned_DS_jobs$python_needed <- sapply(Uncleaned_DS_jobs$Job_Description, python_mentio
```

```r
Uncleaned_DS_jobs$python_needed <- as.factor(Uncleaned_DS_jobs$python_needed)
```

Now for Excel:

```r
excel_mentioned <- function(description) {
  # We use tolower to match the excel in the job description
  description <- tolower(description)

  # Check if excel is mentioned
  if (grepl("\\bexcel\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```r
Uncleaned_DS_jobs$excel_needed <- sapply(Uncleaned_DS_jobs$Job_Description, excel_mentione
```

```r
summary(Uncleaned_DS_jobs$excel_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.0000  0.0000  0.1161  0.0000  1.0000
```

```r
Uncleaned_DS_jobs$excel_needed <- as.factor(Uncleaned_DS_jobs$excel_needed)
```

For Hadoop:

```r
hadoop_mentioned <- function(description) {
  # We use tolower to match the hadoop in the job description
  description <- tolower(description)

  # Check if hadoop is mentioned
  if (grepl("\\bhadoop\\b", description)) {
    return(1)
  } else {
    return(0)
  }
```

```r
}
```

```r
Uncleaned_DS_jobs$hadoop_needed <- sapply(Uncleaned_DS_jobs$Job_Description, hadoop_mentio
```

```r
summary(Uncleaned_DS_jobs$hadoop_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.0000  0.0000  0.2128  0.0000  1.0000
```

```r
Uncleaned_DS_jobs$hadoop_needed<- as.factor(Uncleaned_DS_jobs$hadoop_needed)
```

For Spark:

```r
spark_mentioned <- function(description) {
  # We use tolower to match the spark in the job description
  description <- tolower(description)

  # Check if spark is mentioned
  if (grepl("\\bspark\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```r
Uncleaned_DS_jobs$spark_needed <- sapply(Uncleaned_DS_jobs$Job_Description, spark_mentione
```

```r
summary(Uncleaned_DS_jobs$spark_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.0000  0.0000  0.2664  1.0000  1.0000
```

```r
Uncleaned_DS_jobs$spark_needed <- as.factor(Uncleaned_DS_jobs$spark_needed)
```

For AWS:

```r
aws_mentioned <- function(description) {
  # We use tolower to match the AWS in the job description
  description <- tolower(description)

  # Check if AWS is mentioned
  if (grepl("\\baws\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```r
Uncleaned_DS_jobs$aws_needed <- sapply(Uncleaned_DS_jobs$Job_Description, aws_mentioned)
```

```r
summary(Uncleaned_DS_jobs$aws_needed)
```

```
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0000  0.0000  0.0000  0.2009  0.0000  1.0000
```

```r
Uncleaned_DS_jobs$aws_needed <- as.factor(Uncleaned_DS_jobs$aws_needed)
```

For Tableau:

```r
tableau_mentioned <- function(description) {
  # We use tolower to match the Tableau in the job description
  description <- tolower(description)

  # Check if Tableau is mentioned
  if (grepl("\\btableau\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```r
Uncleaned_DS_jobs$tableau_needed <- sapply(Uncleaned_DS_jobs$Job_Description, tableau_ment
```

```r
summary(Uncleaned_DS_jobs$tableau_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.000   0.000   0.000   0.183   0.000   1.000 
```

```
Uncleaned_DS_jobs$tableau_needed <- as.factor(Uncleaned_DS_jobs$tableau_needed)
```

For Big Data:

```
bigdata_mentioned <- function(description) {
  # We use tolower to match the Big data in the job description
  description <- tolower(description)

  # Check if Big data is mentioned
  if (grepl("\\bbig-data\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```
Uncleaned_DS_jobs$bigdata_needed <- sapply(Uncleaned_DS_jobs$Job_Description, bigdata_ment
```

```
summary(Uncleaned_DS_jobs$bigdata_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
0.00000 0.00000 0.00000 0.01042 0.00000 1.00000 
```

```
Uncleaned_DS_jobs$bigdata_needed<- as.factor(Uncleaned_DS_jobs$bigdata_needed)
```

For Numpy:

```
numpy_mentioned <- function(description) {
  # We use tolower to match the Numpy in the job description
  description <- tolower(description)

  # Check if Numpy is mentioned
  if (grepl("\\bnumpy\\b", description)) {
    return(1)
  } else {
    return(0)
  }
```

```r
}
```

```r
Uncleaned_DS_jobs$numpy_needed <- sapply(Uncleaned_DS_jobs$Job_Description, numpy_mentione
```

```r
summary(Uncleaned_DS_jobs$numpy_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
0.00000 0.00000 0.00000 0.08482 0.00000 1.00000 
```

```r
Uncleaned_DS_jobs$numpy_needed <- as.factor(Uncleaned_DS_jobs$numpy_needed)
```

For Machine Learning:

```r
ML_mentioned <- function(description) {
  # We use tolower to match the ML in the job description
  description <- tolower(description)

  # Check if ML is mentioned
  if (grepl("\\bmachine learning\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```r
Uncleaned_DS_jobs$ML_needed <- sapply(Uncleaned_DS_jobs$Job_Description, ML_mentioned)
```

```r
summary(Uncleaned_DS_jobs$ML_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.000   0.000   1.000   0.619   1.000   1.000 
```

```r
Uncleaned_DS_jobs$ML_needed<- as.factor(Uncleaned_DS_jobs$ML_needed)
```

For Deep Learning:

```r
DL_mentioned <- function(description) {
  # We use tolower to match the DL in the job description
  description <- tolower(description)

  # Check if DL is mentioned
  if (grepl("\\bdeep learning\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```r
Uncleaned_DS_jobs$DL_needed <- sapply(Uncleaned_DS_jobs$Job_Description, DL_mentioned)
```

```r
summary(Uncleaned_DS_jobs$DL_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.0000  0.0000  0.1429  0.0000  1.0000 
```

```r
Uncleaned_DS_jobs$DL_needed <- as.factor(Uncleaned_DS_jobs$DL_needed)
```

For Statistics:

```r
stat_mentioned <- function(description) {
  # We use tolower to match the statistics in the job description
  description <- tolower(description)

  # Check if statistics is mentioned
  if (grepl("\\bstatistics\\b", description)) {
    return(1)
  } else {
    return(0)
  }
}
```

```r
Uncleaned_DS_jobs$stat_needed <- sapply(Uncleaned_DS_jobs$Job_Description, stat_mentioned)
```

```r
summary(Uncleaned_DS_jobs$stat_needed)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.0000  0.0000  0.4926  1.0000  1.0000 
```

```r
Uncleaned_DS_jobs$stat_needed <- as.factor(Uncleaned_DS_jobs$stat_needed)
```

Now Let's check the new columns in our dataset:

```r
summary(Uncleaned_DS_jobs)
```

```
                              Job_Title                          Salary_Estimate
 Data Analyst                      : 47   $75K-$131K (Glassdoor est.) : 32  
 Data Engineer                     : 47   $79K-$131K (Glassdoor est.) : 32  
 Data Science and Analytics Manager: 14   $99K-$132K (Glassdoor est.) : 32  
 Data Scientist                    :455   $137K-$171K (Glassdoor est.): 30  
 Machine Learning Engineer         : 36   $90K-$109K (Glassdoor est.) : 30  
 Other Data Positions              : 23   $56K-$97K (Glassdoor est.)  : 22  
 Others                            : 50   (Other)                     :494  
 Job_Description        Rating      Company_Name         Location        
 Length:672         Min.   :0.000   Length:672         Length:672        
 Class :character   1st Qu.:3.300   Class :character   Class :character  
 Mode  :character   Median :3.800   Mode  :character   Mode  :character  
                    Mean   :3.593                                        
                    3rd Qu.:4.300                                        
                    Max.   :5.000                                        
                                                                         
 Location_State     Headquarters       Headquarters_State     Size          
 Length:672         Length:672         Length:672         Length:672        
 Class :character   Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character   Mode  :character  
                                                                            
                                                                            
                                                                            
                                                                            
   Founded          Type_of_Ownership    Industry            Sector         
 Length:672         Length:672         Length:672         Length:672        
 Class :character   Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character   Mode  :character  
```

```
  Revenue            Competitors          Low_Limit_For_Salary
 Length:672         Length:672          Min.   : 31000
 Class :character   Class :character    1st Qu.: 79000
 Mode  :character   Mode  :character    Median : 91000
                                        Mean   : 99196
                                        3rd Qu.:122000
                                        Max.   :212000

 High_Limit_For_Salary Senior_Position  sql_needed python_needed excel_needed
 Min.   : 56000        Min.   :0.0000   0:347      0:183         0:594
 1st Qu.:119000        1st Qu.:0.0000   1:325      1:489         1: 78
 Median :133000        Median :0.0000
 Mean   :148131        Mean   :0.1057
 3rd Qu.:165000        3rd Qu.:0.0000
 Max.   :331000        Max.   :1.0000

 hadoop_needed spark_needed aws_needed tableau_needed bigdata_needed
 0:529         0:493        0:537      0:549          0:665
 1:143         1:179        1:135      1:123          1:  7

 numpy_needed ML_needed DL_needed stat_needed
 0:615        0:256     0:576     0:341
 1: 57        1:416     1: 96     1:331
```

Visualizations

First, see our variables:

```r
glimpse(Uncleaned_DS_jobs)
```

```
Rows: 672
Columns: 31
```

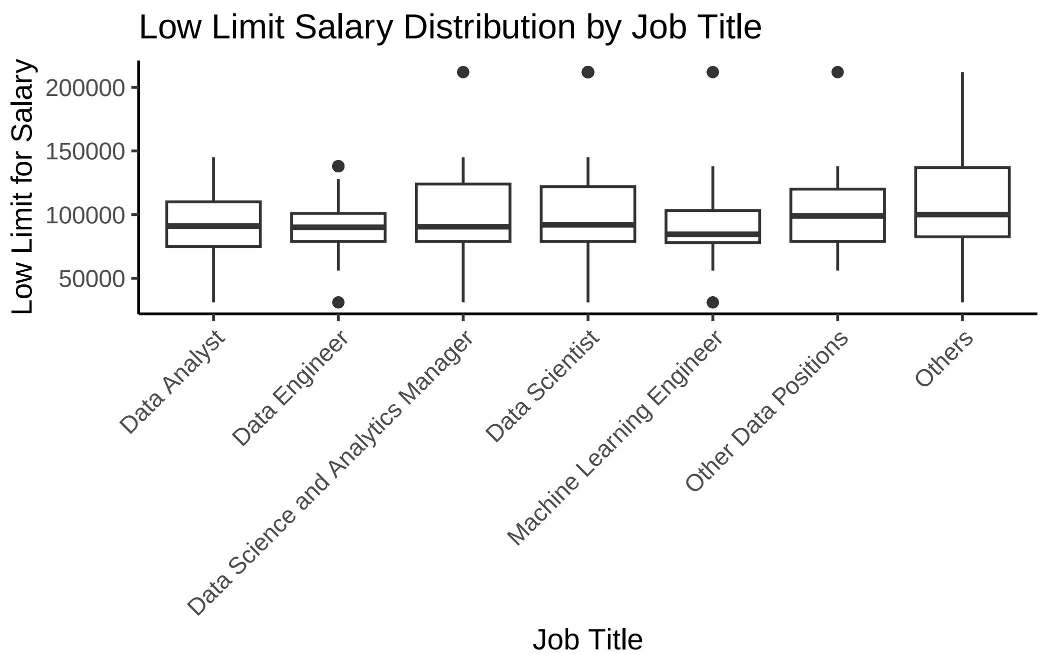
```
$ Job_Title             <fct> Data Scientist, Data Scientist, Data Scientist, ~
$ Salary_Estimate       <fct> $137K-$171K (Glassdoor est.), $137K-$171K (Glass~
$ Job_Description       <chr> "Description\n\nThe Senior Data Scientist is res~
$ Rating                <dbl> 3.1, 4.2, 3.8, 3.5, 2.9, 4.2, 3.9, 3.5, 4.4, 3.6~
$ Company_Name          <chr> "Healthfirst", "ManTech", "Analysis Group", "INF~
$ Location              <chr> "New York", "Chantilly", "Boston", "Newton", "Ne~
$ Location_State        <chr> "NY", "VA", "MA", "MA", "NY", "CA", "MA", "MA", ~
$ Headquarters          <chr> "New York", "Herndon", "Boston", "Bad Ragaz", "N~
$ Headquarters_State    <chr> "NY", "VA", "MA", "Switzerland", "NY", "CA", "Sw~
$ Size                  <chr> "1001 to 5000 employees", "5001 to 10000 employe~
$ Founded               <chr> "1993", "1968", "1981", "2000", "1998", "2010", ~
$ Type_of_Ownership     <chr> "Nonprofit Organization", "Company - Public", "P~
$ Industry              <chr> "Insurance Carriers", "Research & Development", ~
$ Sector                <chr> "Insurance", "Business Services", "Business Serv~
$ Revenue               <chr> "Unknown / Non-Applicable", "$1 to $2 billion (U~
$ Competitors           <chr> "EmblemHealth, UnitedHealth Group, Aetna", "No i~
$ Low_Limit_For_Salary  <dbl> 137000, 137000, 137000, 137000, 137000, 137000, ~
$ High_Limit_For_Salary <dbl> 171000, 171000, 171000, 171000, 171000, 171000, ~
$ Senior_Position       <int> 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, ~
$ sql_needed            <fct> 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, ~
$ python_needed         <fct> 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, ~
$ excel_needed          <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
$ hadoop_needed         <fct> 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, ~
$ spark_needed          <fct> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, ~
$ aws_needed            <fct> 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, ~
$ tableau_needed        <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, ~
$ bigdata_needed        <fct> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
$ numpy_needed          <fct> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, ~
$ ML_needed             <fct> 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, ~
$ DL_needed             <fct> 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, ~
$ stat_needed           <fct> 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, ~
```

- **Salary Distribution by Job Titles:** Let's display the distribution of salaries for different job titles; both for low limit estimate for salary and high limit estimate for salary:

- Low Limit Estimate for Salary

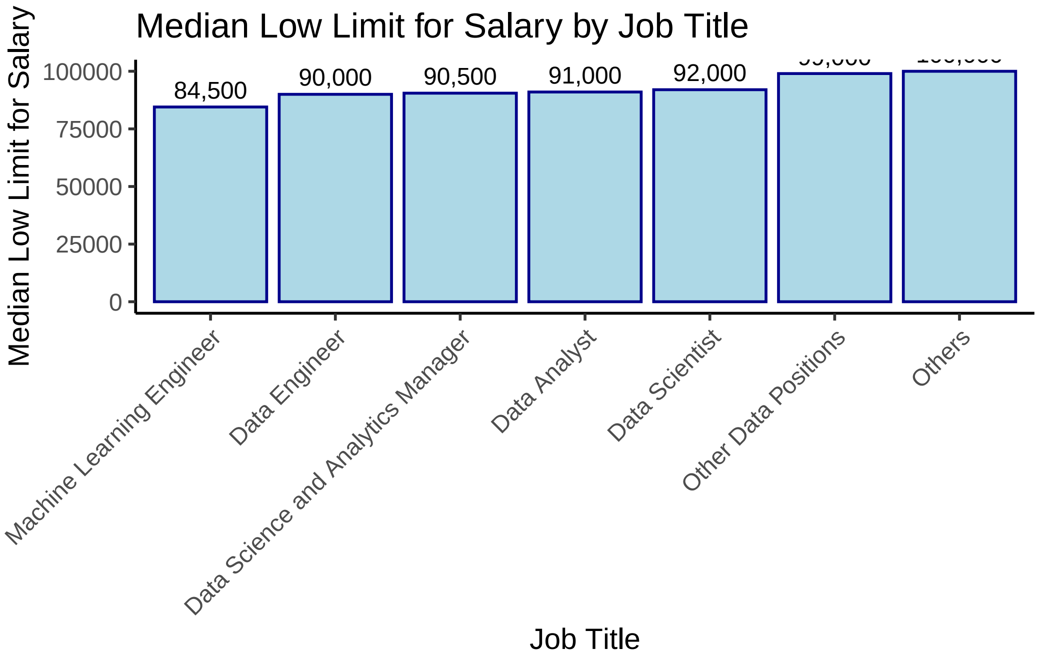
```
salary_distribution <- ggplot(Uncleaned_DS_jobs, aes(x = Job_Title, y = Low_Limit_For_Sala
  geom_boxplot() +
  labs(title = "Low Limit Salary Distribution by Job Title",
       x = "Job Title",
       y = "Low Limit for Salary") +
```

```
  theme_classic() +
  theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))

salary_distribution
```



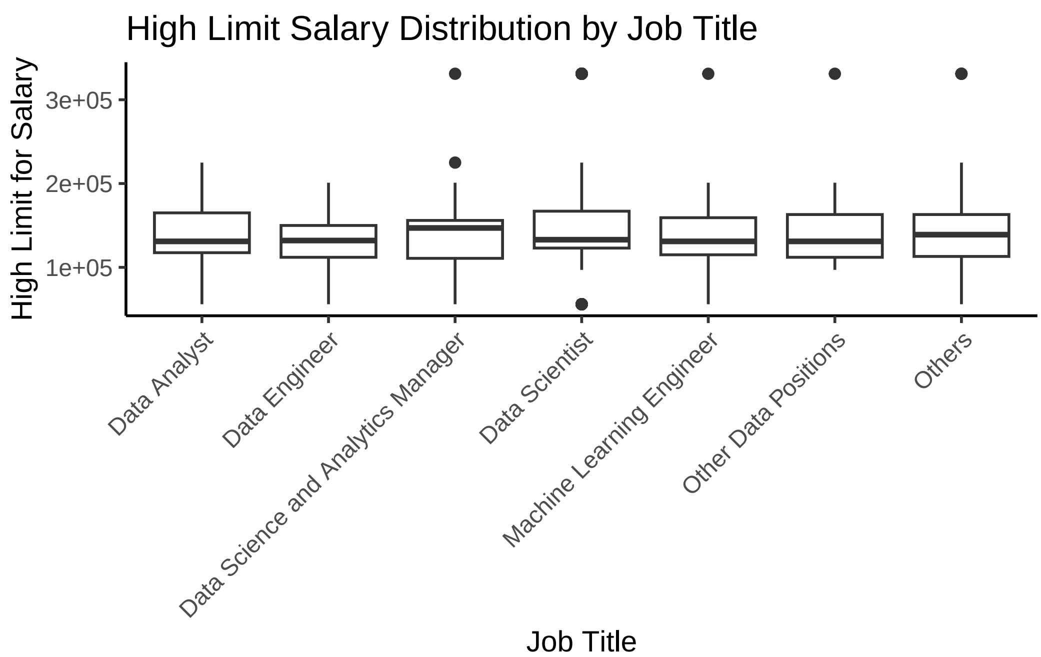

```
# Calculate median low salary limits for each job title
salary_median <- Uncleaned_DS_jobs %>%
  group_by(Job_Title) %>%
  summarise(median_salary = median(Low_Limit_For_Salary))

# Sort the data by median low salary in descending order
salary_median <- salary_median %>%
  arrange(desc(median_salary))


salary_distribution <- ggplot(salary_median,
                              aes(x = reorder(Job_Title,
                                              median_salary),
                                  y = median_salary)) +
  geom_bar(stat = "identity", fill = "lightblue", col = "darkblue") +
  geom_text(aes(label = scales::comma(median_salary)), vjust = -0.5, size = 3) +
```

```
  labs(title = "Median Low Limit for Salary by Job Title",
       x = "Job Title",
       y = "Median Low Limit for Salary") +
  theme_classic() +
  theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))

salary_distribution
```



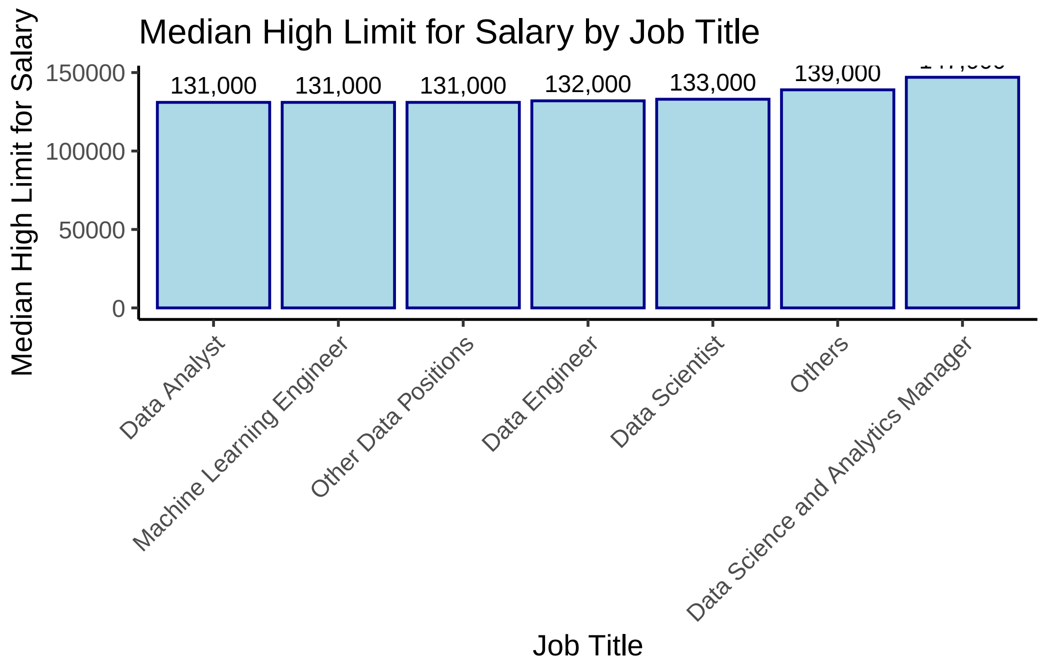
- High Limit Estimate for Salary

```
salary_distribution_high <- ggplot(Uncleaned_DS_jobs, aes(x = Job_Title, y = High_Limit_Fo
  geom_boxplot() +
  labs(title = "High Limit Salary Distribution by Job Title",
       x = "Job Title",
       y = "High Limit for Salary") +
  theme_classic() +
  theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))

salary_distribution_high
```




```r
# Calculate median low salary limits for each job title
salary_median <- Uncleaned_DS_jobs %>%
  group_by(Job_Title) %>%
  summarise(median_salary = median(High_Limit_For_Salary))

# Sort the data by median low salary in descending order
salary_median <- salary_median %>%
  arrange(desc(median_salary))


salary_distribution <- ggplot(salary_median,
                              aes(x = reorder(Job_Title,
                                              median_salary),
                                  y = median_salary)) +
  geom_bar(stat = "identity", fill = "lightblue", col = "darkblue") +
  geom_text(aes(label = scales::comma(median_salary)), vjust = -0.5, size = 3) +
  labs(title = "Median High Limit for Salary by Job Title",
       x = "Job Title",
       y = "Median High Limit for Salary") +
  theme_classic() +
  theme(axis.text.x = element_text(size = 9.2, angle = 45, hjust = 1))
```

```
salary_distribution
```

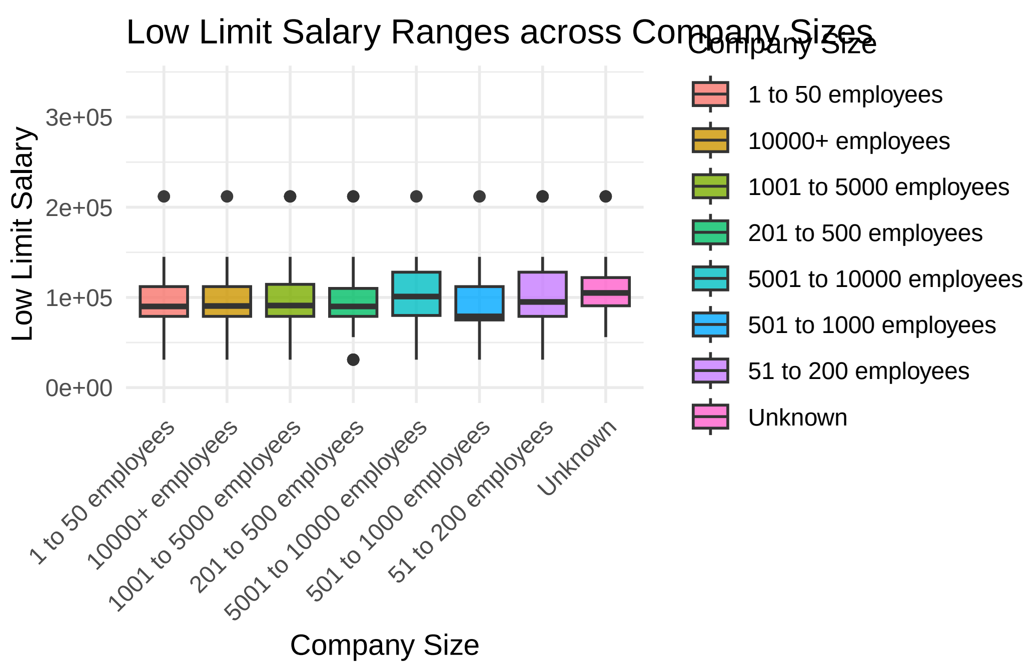


As can be seen from the graph that there is not a significant difference between the medians of the different job titles.

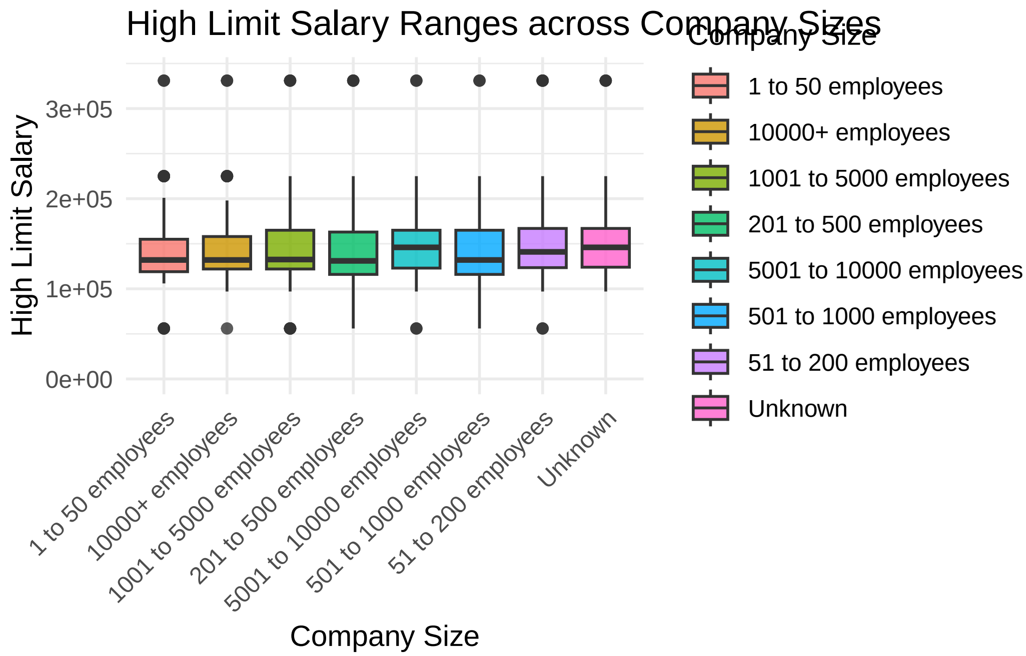
- **Company Size vs. Salaries:** To explore how salary ranges vary across different company sizes:

```
# Create a new variable with sorted factor levels
data_sorted <- transform(Uncleaned_DS_jobs, Size_Sorted = factor(Size, levels = sort(

# Create boxplot for Low_Limit_For_Salary with adjusted y-axis limits
plot_low <- ggplot(data_sorted, aes(x = Size_Sorted, y = Low_Limit_For_Salary, fill =
  geom_boxplot(alpha = 0.8) +
  labs(title = "Low Limit Salary Ranges across Company Sizes",
       x = "Company Size",
       y = "Low Limit Salary") +
  scale_fill_discrete(name = "Company Size") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  coord_cartesian(ylim = c(0, 340000))  # Set y-axis limits
```

```r
# Create boxplot for High_Limit_For_Salary with the same y-axis limits
plot_high <- ggplot(data_sorted, aes(x = Size_Sorted, y = High_Limit_For_Salary, fill
  geom_boxplot(alpha = 0.8) +
  labs(title = "High Limit Salary Ranges across Company Sizes",
       x = "Company Size",
       y = "High Limit Salary") +
  scale_fill_discrete(name = "Company Size") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  coord_cartesian(ylim = c(0, 340000))  # Set y-axis limits

plot_low
```




```r
plot_high
```



- **Job Titles vs. Senior Positions:** Visualize the proportion of senior positions against different job titles using a bar chart:

Firstly, we can see the distribution of Senior Position among the job titles:

```
Uncleaned_DS_jobs %>%
  select(Senior_Position, Job_Title)  %>%
  group_by(Job_Title, Senior_Position)  %>%
  count()
```

```
# A tibble: 12 x 3
# Groups:   Job_Title, Senior_Position [12]
   Job_Title                          Senior_Position     n
   <fct>                                        <int> <int>
 1 Data Analyst                                     0    37
 2 Data Analyst                                     1    10
 3 Data Engineer                                    0    41
 4 Data Engineer                                    1     6
 5 Data Science and Analytics Manager               0    14
 6 Data Scientist                                   0   413
 7 Data Scientist                                   1    42
 8 Machine Learning Engineer                        0    30
```

```
 9 Machine Learning Engineer                                   1      6
10 Other Data Positions                                        0     23
11 Others                                                      0     43
12 Others                                                      1      7
```

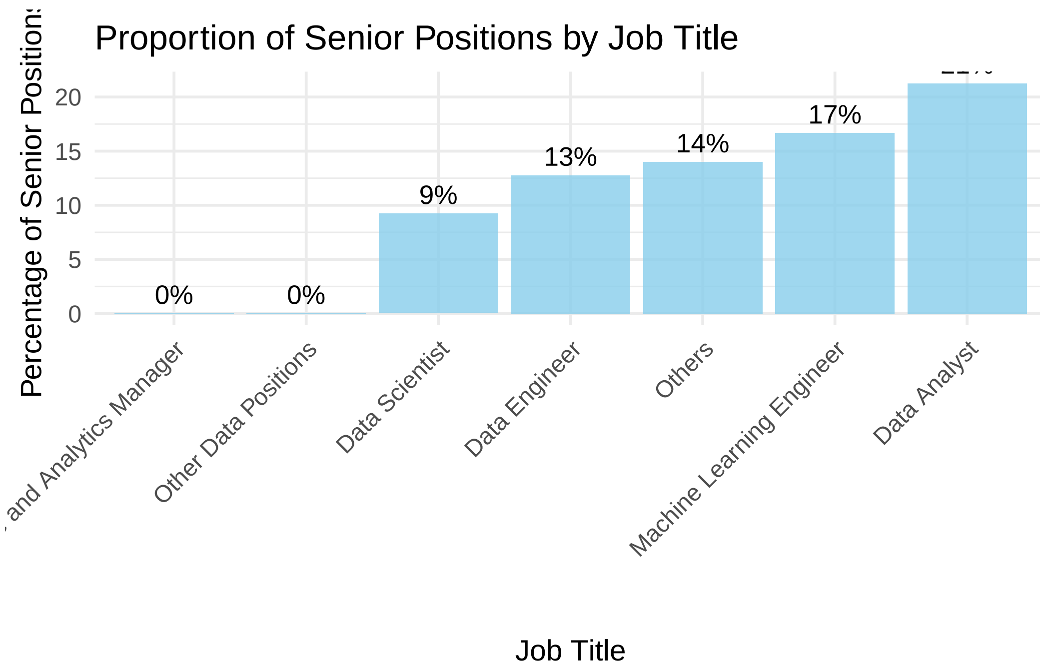
Then we will calculate the percentage of the senior positions for every title:

```
senior_proportion <- Uncleaned_DS_jobs %>%
  group_by(Job_Title) %>%
  summarise(Percentage_Senior = mean(Senior_Position) * 100) %>%
  arrange(desc(Percentage_Senior))
senior_proportion
```

```
# A tibble: 7 x 2
  Job_Title                           Percentage_Senior
  <fct>                                           <dbl>
1 Data Analyst                                    21.3
2 Machine Learning Engineer                       16.7
3 Others                                          14
4 Data Engineer                                   12.8
5 Data Scientist                                   9.23
6 Data Science and Analytics Manager               0
7 Other Data Positions                             0
```

Let's create bar graph:

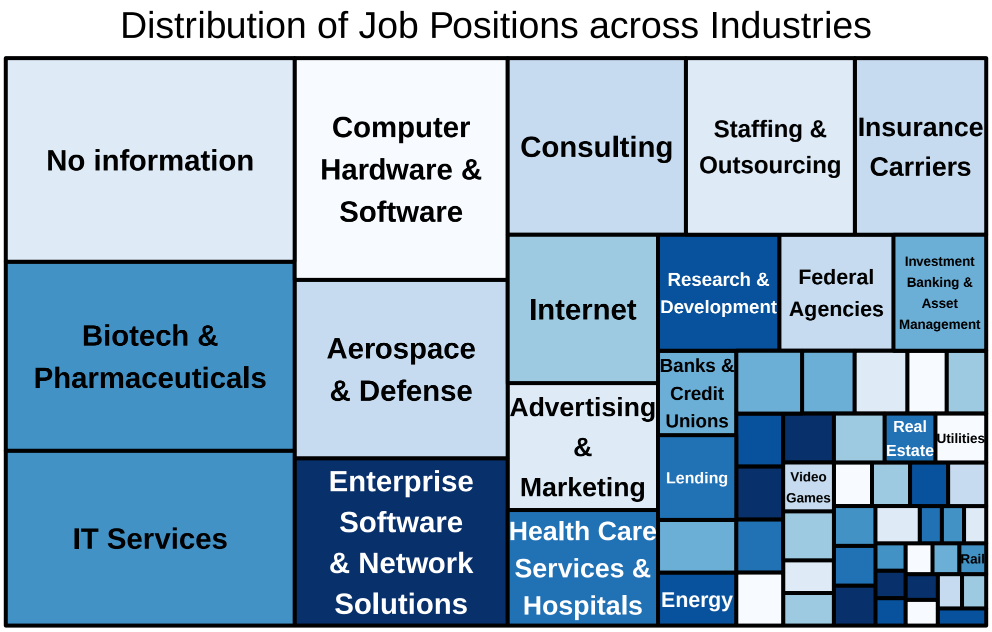
```
senior_plot <- ggplot(senior_proportion, aes(x = reorder(Job_Title, Percentage_Senior), y
  geom_bar(stat = "identity", fill = "skyblue", alpha = 0.8) +
  geom_text(aes(label = paste0(round(Percentage_Senior), "%")), vjust = -0.5, size = 3.5,
  labs(title = "Proportion of Senior Positions by Job Title",
       x = "Job Title",
       y = "Percentage of Senior Positions") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

senior_plot
```



We can see that the title that is searched for seniority is Data Analyst, however, only 21% of the Data Analyst positions are senior.

- **Industry Analysis:** Use a treemap to display the distribution of job positions across different industries.

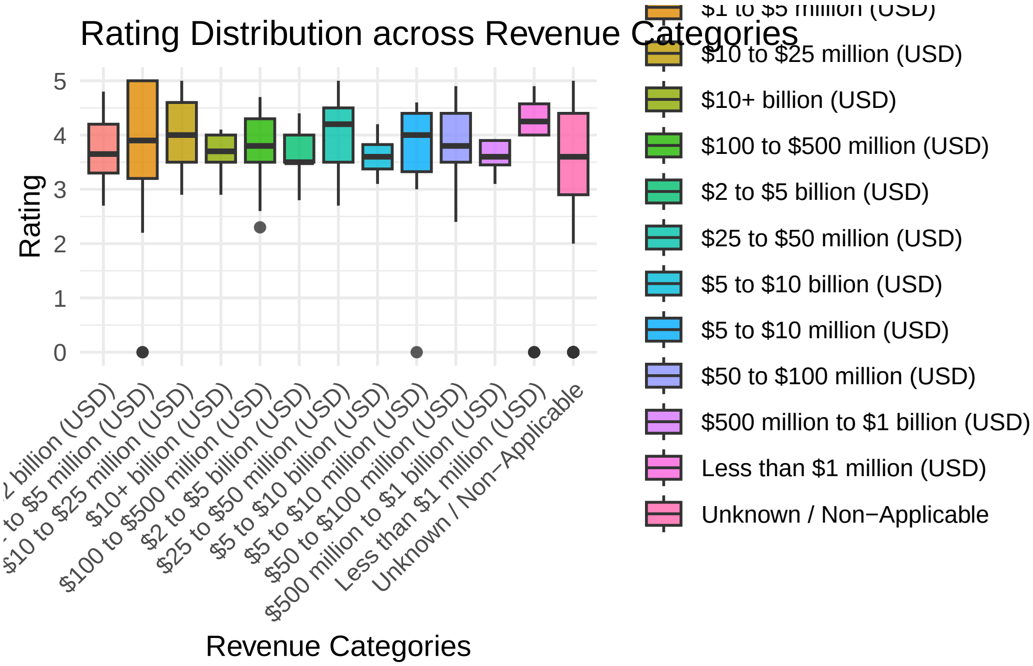
```r
library(treemap)

job_count_by_industry <- Uncleaned_DS_jobs %>%
  group_by(Industry) %>%
  summarise(Job_Count = n()) %>%
  arrange(desc(Job_Count))

# Create a treemap for job positions across different industries with custom theme
treemap_plot <- treemap(job_count_by_industry, index = "Industry", vSize = "Job_Count",
                        title = "Distribution of Job Positions across Industries",
                        palette = "Blues")
```



We can see that the industries that are hiring the most are: Biotech & Pharmaceuticals, IT Services, Computer Hardware & Software, Aerospace & Defense and so on.

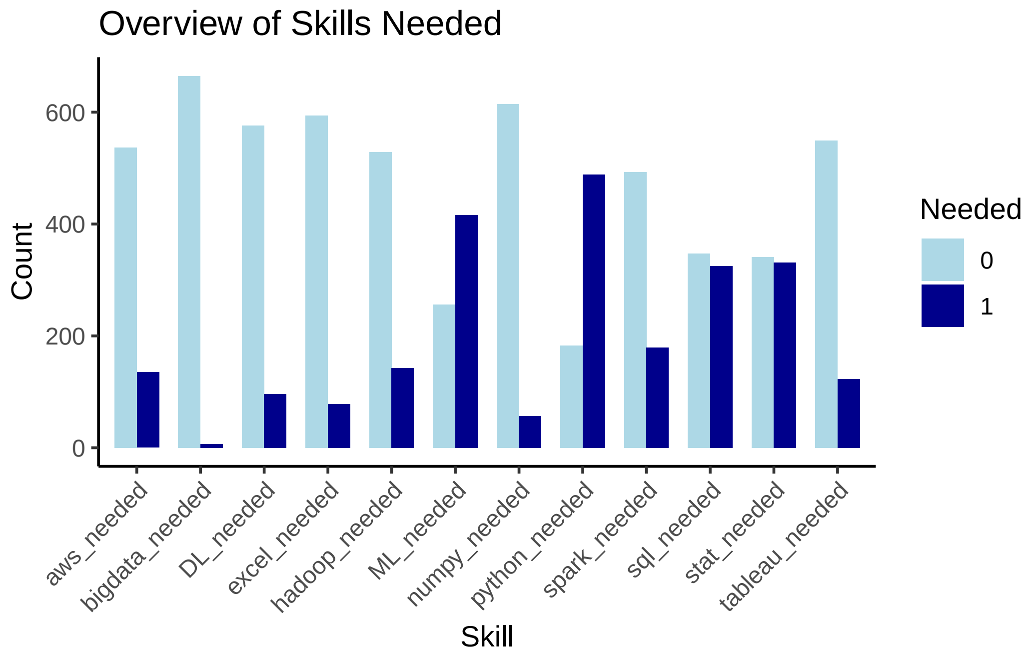
- Revenue vs. Ratings: Create a grouped boxplot to show the distribution of ratings for different revenue categories

```
boxplot <- ggplot(Uncleaned_DS_jobs,
                  aes(x = Revenue,
                      y = Rating,
                      fill = Revenue)) +
  geom_boxplot(alpha = 0.8) +
  labs(title = "Rating Distribution across Revenue Categories",
       x = "Revenue Categories",
       y = "Rating") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

boxplot
```



- Overview of Skills Needed in the Job Postings:

```
skills_df <- Uncleaned_DS_jobs[, c("sql_needed", "python_needed", "numpy_needed", "st
                    "aws_needed","tableau_needed","bigdata_needed","ML_needed","DL_ne

skills_long <- tidyr::gather(skills_df, key = "Skill", value = "Needed")


my_colors <- c("lightblue", "darkblue")

skills_long$Needed <- factor(skills_long$Needed, levels = c("0", "1"))

skills_plot <- ggplot(skills_long, aes(x = Skill, fill = Needed)) +
  geom_bar(position = "dodge", width = 0.7) +
  scale_fill_manual(values = my_colors) +
  labs(title = "Overview of Skills Needed",
       x = "Skill",
       y = "Count") +
  theme_classic() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

skills_plot
```



As can be seen from the graph that python and ML are most mentioned skills that are required in the job postings.

References

About us | glassdoor. (n.d.). https://www.glassdoor.com/about/

Barr, D., & DeBruine, L. (n.d.). Data Cleaning. https://rgup.gitlab.io/research_cycle/03_tidyr.html

Cotton, R. (2023, February 16). *Quarto cheat sheet (previously known as RMarkdown)*. DataCamp. https://www.datacamp.com/cheat-sheet/quarto-cheat-sheet-previously-known-as-r-markdown

GREP: Pattern matching and replacement. RDocumentation. (n.d.). https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/grep

Rahman, R. (n.d.). [Dataset] Data Science Job Posting on Glassdoor. Retrieved December 20, 2023,. https://www.kaggle.com/datasets/rashikrahmanpritom/data-science-job-posting-on-glassdoor/data

Wickham, H., Çetinkaya-Rundel, M., & Grolemund, G. (2023). *R for Data Science: Import, Tidy, transform, visualize, and model data.* O'Reilly Media, Inc.