SALARY PREDICTION USING UNSUPERVISED MACHINE LEARNING

Statistical Learning, Deep Learning and Artificial Intelligence by Prof. Silvia Salini

Tejaswini Yadav Nakka - 06651A

A.A. 2022-2023

Nowadays, salary has become the most crucial aspect of employee compensation and a vital component of organizational strategy. Companies are increasingly adopting the practice of benchmarking to gather data on tech labor wages in other organizations with similar roles. This information is used to design competitive pay scales, compare with industry rivals, and ensure fair compensation practices. Moreover, job seekers can benefit from understanding the salary ranges for potential careers, enabling them to focus on high-paying industries and align their skills and aspirations accordingly. However, the research landscape lacks comprehensive studies on the implementation of machine learning models for predicting salary in tech jobs, especially considering the uncertainties surrounding the key factors influencing data science job salaries. Therefore, this project aims to develop a model that accurately predicts salary in tech jobs, while also identifying the primary features that contribute to earning a better wage in the role of a data scientist.

Contents

1 Introduction

This project focuses on developing a machine learning model to predict salary in tech jobs and identify key features influencing data scientist salaries. The study employs benchmarking techniques to gather wage data from similar roles in other organizations. By utilizing various algorithms like multiple linear regression, random forest, and XGBoost, the model aims to accurately predict salary. The analysis also explores the relationship between factors such as age, company size and average salary through exploratory data analysis. The project provides valuable insights for job seekers and organizations seeking to design competitive pay scales and enhance employee remuneration strategies.

2 Methodology

- Data Cleaning and Preparation:
  - Select relevant columns using the dplyr::select function.
  - Clean the data for age, average salary, job title, company size, and revenue.
- Exploratory Data Analysis (EDA):
  - Visualize relationships between variables: Age vs. Average Salary.
  - Compare Salary distribution across company sizes.
  - Compare salary distribution for different tech job titles.
- Identify Key Skills:

  - Analyze and visualize the prevalence of skills like Python, R, Spark, AWS, and Excel in different tech job positions.
- Machine Learning Model Building:
  - Split the data into training and testing datasets.
  - Build and train the following machine learning models:
  - Multiple Linear Regression.
  - Random Forest.
  - XGBoost.
- Model Evaluation:

  - Evaluate the performance of each model using metrics such as Root Mean Squared Error (RMSE) and R-squared.
- Results and Insights:
  - Present the evaluation metrics for each model.

– Provide insights into the accuracy and performance of the models.

- Recommendations:

  - Offer valuable knowledge and recommendations for individuals seeking data science roles and organizations designing salary structures.

3 Data Collection & Data Description

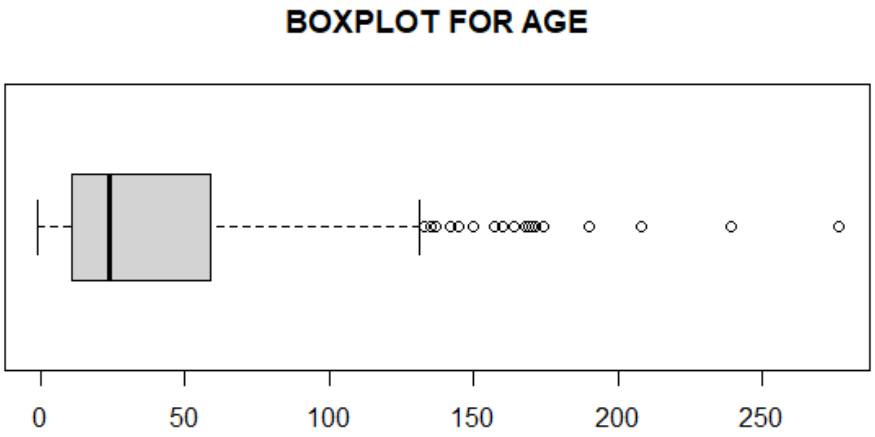
The dataset used for this research on salary prediction consists of 742 entries and was sourced from Kaggle. It is a collection of tech job positions extracted from glassdoor.com in 2017. The dataset serves the purpose of predicting salaries based on various features such as age, company size, and more. Among the available columns, one specifically labeled as "job title" represents the tech job positions. To streamline the analysis, only 12 features were selected as the primary focus of this project.

```
> #Cleaning the data
> df <- df %>%
+   dplyr::select(1,9,12,14,20,24:30)
> colnames(df)
 [1] "X"          "Size"       "Industry"   "Revenue"    "avg_salary"
 [6] "age"        "python_yn"  "R_yn"       "spark"      "aws"
[11] "excel"      "job_simp"
> str(df)
'data.frame':   742 obs. of  12 variables:
 $ X         : int  0 1 2 3 4 5 6 7 8 9 ...
 $ Size      : chr  "501 to 1000 employees" "10000+ employees" "501 to 1
000 employees" "1001 to 5000 employees" ...
 $ Industry  : chr  "Aerospace & Defense" "Health Care Services & Hospit
als" "Security Services" "Energy" ...
 $ Revenue   : chr  "$50 to $100 million (USD)" "$2 to $5 billion (USD)"
"$100 to $500 million (USD)" "$500 million to $1 billion (USD)" ...
 $ avg_salary: num  72 87.5 85 76.5 114.5 ...
 $ age       : int  47 36 10 55 22 20 12 15 6 11 ...
 $ python_yn : int  1 1 1 1 1 1 0 1 0 1 ...
 $ R_yn      : int  0 0 0 0 0 0 0 0 0 0 ...
 $ spark     : int  0 0 1 0 0 0 0 1 0 1 ...
 $ aws       : int  0 0 0 0 0 1 0 1 0 0 ...
 $ excel     : int  1 0 1 0 1 1 1 1 0 0 ...
 $ job_simp  : chr  "data scientist" "data scientist" "data scientist"
"data scientist" ...
```

Figure 1: Considering necessary columns

4 Data Cleaning

4.1 Age




```
> #Checking for people who do no belong to the working age
> min_age <- sum(df$age < 18)
> min_age
[1] 0
> max_age <- sum(df$age > 70)
> max_age
[1] 0
> # To replace <18 outliers with IQR Q1 (30) & replace >70 outliers with
IQR Q3(52)
> getWorkingAge <- df$age[df$age <66 & df$age > 23 & !is.na(df$age)]
> quantile(getWorkingAge)
  0%  25%  50%  75% 100%
  24   30   30   52   62
> df$age[df$age<18] <- 30
> df$age[df$age>70] <- 52
> hist(df$age,xlab="age",main="Age")
```

Figure 2: Boxplot for age

We are assuming that the age for the working is between 18-70. But we can see that there are some outliers. So when we checked for the outliers. we found that the people who are below the age 18 are a total of 290 and the people who are above the age 70 are 156. To replace these outliers, we created a subset of the "age" variable (getWorkingAge) by selecting ages that are between 23 and 66, excluding missing values (!is.na(df$age)).

Next, the quantile function is applied to the getWorkingAge subset. The quantile function calculates various quantiles of a dataset, which in this case, is used to determine the quartiles (Q1, Q2, and Q3) of the working age group.

By obtaining the quartiles, we replaced the outliers with more representative values. Specifically, replacing ages less than 18 with the first quartile (Q1) value of 30, and ages greater than 70 with the third quartile (Q3) value of 52.

This approach aims to handle extreme age values that might be considered unrealistic or erroneous by replacing them with more reasonable values based on the distribution of the working age group.

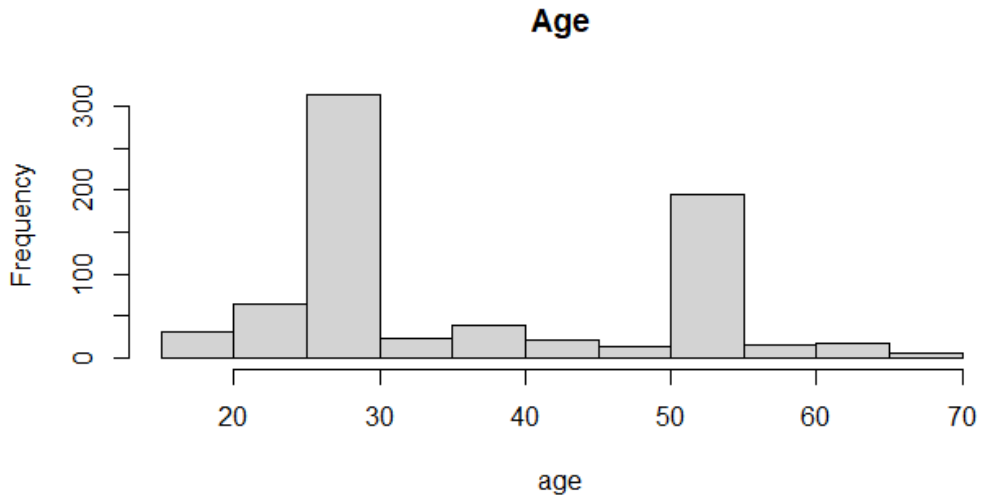


Figure 3: Histogram for age

4.2 Average Salary

The average salary is in decimal point. So we multiplied it by 1000. This multiplication is performed to scale the salary values. Scaling avoid very small decimal values and make the values more manageable and interpretable.

4.3 Job Title Simplified

This helps replacing any occurences of NA with "Other Tech Jobs". By replacing these missing values with a specific string like "Other tech jobs", we can ensure consistency and

clarity in the data.

4.4 Size of the company

By categorizing the company sizes, we create a new variable that groups companies into meaningful size categories. This categorization can be helpful in analyzing and comparing various factors across different company sizes, such as salary distribution or skills required. It allows us to gain insights into how company size might impact these factors and draw meaningful conclusions from the data.

Here's the breakdown of the categorization:

Companies with 1 to 50 employees or 51 to 200 employees are categorized as "Small".

Companies with 201 to 500 employees or 501 to 1000 employees are categorized as "Medium".

Companies with 1001 to 5000 employees, 5001 to 10000 employees, or 10000+ employees are categorized as "Large".

4.5 Revenue of the company

By categorizing the revenue, we create a new variable that groups businesses into meaningful revenue categories.
Here's the breakdown of the categorization:

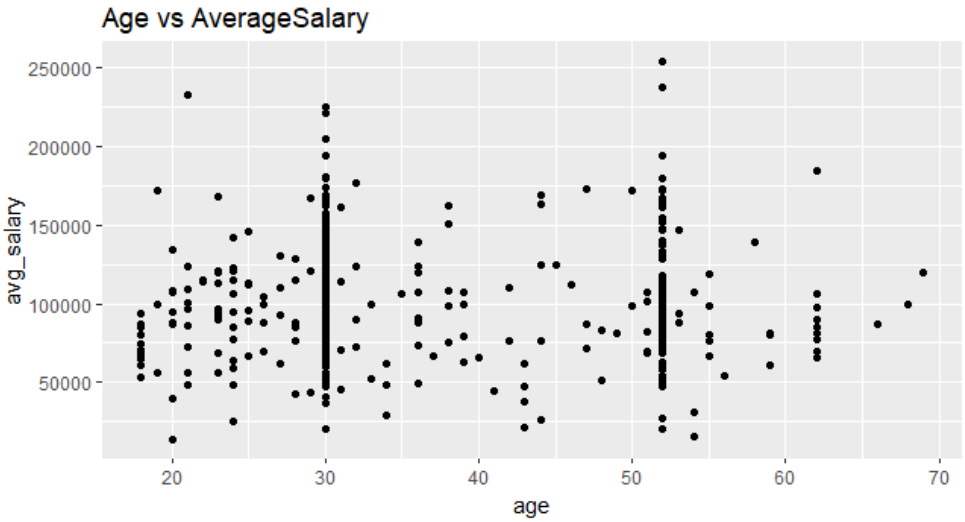
- Businesses with revenue less than $1 million (USD) are categorized as "micro-business".
- Businesses with revenue between $1 million and $10 million (USD) are categorized as "small-business".
- Businesses with revenue between $10 million and $50 million (USD) are categorized as "medium-business".
- Businesses with revenue $50 million and above (USD) are categorized as "large-business".

4.6 Industry

The data frame is subsetted to exclude rows where the 'Industry' column has a value of "-1". This step removes instances where the 'Industry' information is missing or invalid.

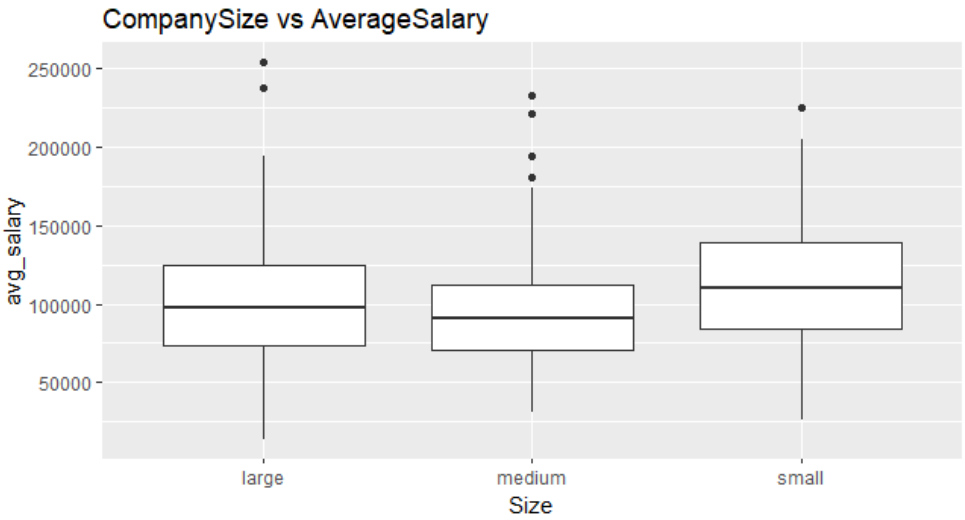
5 Exploratory Data Analysis & Feature Engineering:

5.1 Relationship between age and average Salary



The scatter plot illustrates a random scattering of points, suggesting the absence of any close relationship between age and average salary. The distribution of points appears to be scattered in across the plot, without any noticeable pattern or trend. The absence of a clear association in this particular dataset indicates that age alone may not be a determining factor in explaining variations in average salary.

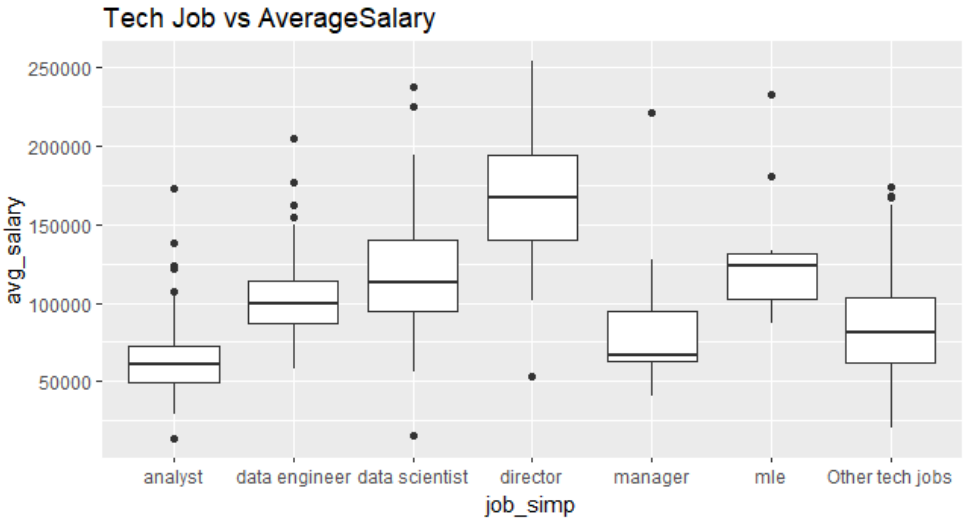
5.2 Relationship between Company and average Salary



Upon examining the three batches of data, it is evident that they exhibit a similar overall distribution. The interquartile ranges, as represented by the lengths of the boxes in the box plots, show comparable variability among the different company sizes. However, there are

noticeable differences in the median average salary across the company sizes. The median average salary of small-sized companies is higher compared to both large-sized and medium-sized companies.
Furthermore, it is worth noting the presence of outliers within the large and medium-sized company categories. These outliers indicate that certain data points within these categories deviate significantly from the majority of the data. These outliers warrant further investigation and analysis to determine the underlying reasons for their deviation.

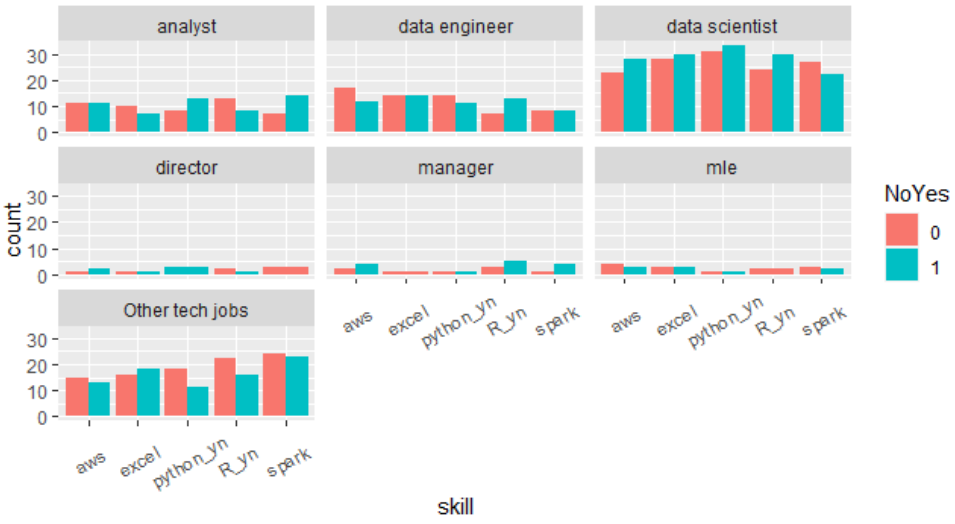
5.3 Salary distribution for every job



In terms of median average salary, the director position stands out with a higher value compared to roles such as MLE, data scientist, data engineer, other tech jobs, manager, and analyst. Among the mentioned positions, the box plots for analyst, data engineer, manager, and MLE appear relatively short, suggesting similar average salaries within these roles. However, the box plots for manager and MLE show uneven sections, indicating greater variability in average salaries across different parts of the scale. Notably, the upper whisker in the MLE box plot suggests more variation in average salaries within the least positive quartile group compared to the most positive quartile group. Outliers are also present across all positions, indicating the presence of data values that significantly deviate from the norm and warrant further examination.

5.4 Job skills needed for different positions

From the figure , we can see that the 3 skills that most needed across the tech jobs would be spark, excel and python_yn.



6 Predicting the Salary

6.1 feature engineering

```
> #Salary Prediction
> #Feature Selection and Normalization
> sal_df<-
+   dplyr::select(df,-c("X","Industry"))
> norm_minmax <- function(x){
+   (x- min(x)) /(max(x)-min(x))
+ }
> sal_df[sapply(sal_df, is.numeric)] <- lapply(sal_df[sapply(sal_df, is.
numeric)],norm_minmax)
> sal_df[sapply(sal_df, is.character)] <- lapply(sal_df[sapply(sal_df, i
s.character)],as.factor)
> str(sal_df)
'data.frame':   732 obs. of  10 variables:
 $ Size       : Factor w/ 3 levels "large","medium",..: 2 1 2 1 3 2 2 2 1
3 ...
 $ Revenue    : Factor w/ 4 levels "large-business",..: 2 1 1 1 2 1 1 2 1
1 ...
 $ avg_salary: num  0.243 0.308 0.297 0.262 0.42 ...
 $ age        : num  0.5686 0.3529 0.2353 0.7255 0.0784 ...
 $ python_yn : num  1 1 1 1 1 1 0 1 0 1 ...
 $ R_yn       : num  0 0 0 0 0 0 0 0 0 0 ...
 $ spark      : num  0 0 1 0 0 0 0 1 0 1 ...
 $ aws        : num  0 0 0 0 0 1 0 1 0 0 ...
 $ excel      : num  1 0 1 0 1 1 1 1 0 0 ...
 $ job_simp   : Factor w/ 7 levels "analyst","data engineer",..: 3 3 3 3
3 3 3 3 7 3 ...
```

6.2 Data Preprocessing

We have created a dataframe called sal_df by by selecting columns from the original data frame, excluding the "X" and "Industry" columns.we applied norm_minmax function to all the numerical columns. so as to normalise all the numeric columns and we converted all the

categorical columns to factors

6.3 Splitting the data

This splitting of the data into training and testing sets allows for model training on the training set and evaluation on the testing set to assess the performance of the model on unseen data.

7 Model Building

After preprocessing the data, the processed data is utilized to obtain accurate results by applying multiple algorithms. A model, in this context, refers to a set of algorithms that aids in identifying relationships within a dataset. An effective model has the capability to predict precise outcomes by uncovering insightful patterns in the data. We will commence with the straightforward linear models, which provide a foundational understanding of the data's behavior. As we gain insights from these models, we will gradually transition to more complex models such as RandomForest and XGBoost. we will build

Linear Regression

Random Forest

XGBoost

7.1 Evaluation Metrics for Regression

The purpose of the model is not complete with the evaluation of models performance. That's why we need an evaluation metric to evaluation our model. Since this is the regression problem, we can evaluate our models using any one of the following evaluation metrics:

- Mean Absolute Error(MAE) is the mean of the absolute value of the errors

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- Mean Square Error (MSE) is the mean of the square error

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

- Root Mean Square Error (MSE) is the mean of the square error

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

7.2 Linear regression

Linear regression algorithm tries to predict the results by plotting the graph between an independent variable and a dependent variable that are derived from the dataset. It is a general statistical analysis mechanism used to build machine learning models. The general equation for linear regression is Z = a + bE Where, Z is the dependent variable and E is independent variable

7.3 Random Forest

Random Forest Algorithm is used to incorporate predictions from multiple decision trees into a single model. This algorithm uses bagging mechanism to create a forest of decision trees. It the incorporates the predictions from multiple decision trees to give very accurate predictions.

The RandomForest algorithm has two steps involved,

- Random forest formation.
- Predict by Random forest classifier generated.

RandomForest is a tree based bootstrapping algorithm wherein a certain no. of weak learners (decision trees) are combined to make a powerful prediction model. For every individual learner, a random sample of rows and a few randomly chosen variables are used to build a decision tree model. Final prediction can be a function of all the predictions made by the individual learners. In case of regression problem, the final prediction can be mean of all the predictions.

7.4 XGBoost

XGBoost is a fast and efficient algorithm and has been used to by the winners of many data science competetions.XGBoost works only with numeric variables.There are many tuning parameters in XGBoost which can be broadly classified into General Parameters, Booster Parameters and Task Parameters.

- General parameters refers to which booster we are using to do boosting. The commonly used are tree or linear model
- Booster parameters depends on which booster you have chosen
- Learning Task parameters that decides on the learning scenario, for example, regression tasks may use different parameters with ranking tasks.

g

	mse	mae	rmse	r2
Multiple Linear Regression	0.07404735	0.2313050	0.2721164	0.05609743
random forest	0.03145445	0.1343582	0.1773540	0.40260665
XGBoost	0.02900874	0.1227350	0.1703195	0.44814521

- To determine the best model, we typically look for the lowest values of evaluation metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE), and the highest value of the coefficient of determination (R-squared).
- Based on the provided evaluation metrics, the models can be ranked as follows:
  - XGBoost: It has the lowest values of MSE, MAE, and RMSE, and the highest R-squared value among the three models. Therefore, it can be considered as the best model based on these metrics.
  - Random Forest: It has relatively higher values of MSE, MAE, and RMSE compared to XGBoost, and a lower R-squared value.
  - Multiple Linear Regression: It has the highest values of MSE, MAE, and RMSE, and the lowest R-squared value among the three models. Hence, based on the provided evaluation metrics, XGBoost is the best model for predicting salary.

8 Conclusion

In this project, we performed exploratory data analysis on a dataset related to tech job positions. We cleaned the data by addressing outliers, imputing missing values, and categorizing variables. We explored the relationships between age and average salary, salary and company size, and salary distribution for different tech job positions. Additionally, we analyzed the job skills required for tech job positions. Finally, we built and evaluated multiple linear regression, random forest, and XGBoost models to predict salaries based on various features.

The objective of this framework is to predict the future salaries from given data of the previous year's using machine Learning techniques. In this paper, we have discussed how different machine learning models are built using different algorithms like Linear regression, Random forest regressor, and XG booster algorithms. These algorithms have been applied to predict the final result of salaries. We have addressed in detail about how the noisy data is been removed and the algorithms used to predict the result. Based on the evaluation metrics by different models we conclude that the random forest approach and XG Booster approach are best models. However, the 4 models does not perform well which the max R2 score is only 44% from XGBoost This may due to several limitation of dataset: The small size of dataset, is insufficient to carry out the accurate salary prediction. There are insufficient features which upon added can increase the prediction.

9 Appendix

```
install.packages("stringr")
install.packages("dplyr")
install.packages("mice")
install.packages("ggplot2")
install.packages("caret")
install.packages("randomForest")
install.packages("xgboost")


library("stringr")
library("dplyr")
library("mice")
library("ggplot2")
library("caret")
library("randomForest")
library("xgboost")


#Loading the data
df <- read.csv("C:/Users/Tejaswini␣yadav/Desktop/Supervised␣learning/eda_data.csv")
head(df)
str(df)
colnames(df)
#Cleaning the data
df <- df %>%
  dplyr::select(1,9,12,14,20,24:30)
summary(df)
colnames(df)
str(df)
head(df)

#Data cleaning(Age)
#Assuming People who are of age (18-70) are the working age group
boxplot(df$age, horizontal= T,main="BOXPLOT␣FOR␣AGE")

#Checking for people who do no belong to the working age
min_age <- sum(df$age < 18)
min_age
max_age <- sum(df$age > 70)
max_age

# To replace <18 outliers with IQR Q1 (30) & replace >70 outliers with IQR Q3(52)
getWorkingAge <- df$age[df$age <66 & df$age > 23 & !is.na(df$age)]
quantile(getWorkingAge)

df$age[df$age<18] <- 30
df$age[df$age>70] <- 52
```

```r
hist(df$age,xlab="age",main="Age")

#Data Cleaning(Average Salary)
head(df$avg_salary)
df$avg_salary <- df$avg_salary*1000
head(df$avg_salary)
#Data Cleaning(Job Title Simplified)
# Replace NA with other tech jobs
df$job_simp <- str_replace_all(df$job_simp, "\\bna\\b", "Other tech jobs")

#Data Cleaning(Categorizing number of employees in the size of company)
#1 to 200 employees -    Small
#201 to 1000 employees -    Medium
#1001 and above -    Large

df$Size <- ifelse(df$Size %in% c("1 to 50 employees", "51 to 200 employees"), "small",
                  ifelse(df$Size %in% c("201 to 500 employees", "501 to 1000 employees"),
                         ifelse(df$Size %in% c("1001 to 5000 employees", "5001 to 10000 e


#Data Cleaning(Revenue categorized by size of the business)

#Less than $1 million (USD) -   micro -business
#$1 million to 10 million (USD) -   small -business
#$10 million to 50 million (USD) -   medium -business
#$50 million and above (USD) -   large -business


df$Revenue<- ifelse(df$Revenue == "Less than $1 million (USD)", "micro-business",
                    ifelse(df$Revenue %in% c("$1 to $5 million (USD)", "$5 to $10 million
                           ifelse(df$Revenue %in% c("$10 to $25 million (USD)","$25 to $5
                                  ifelse(df$Revenue %in% c("$100 to $500 million (USD)",



#Data Cleaning - Impute Missing Value Revenue(195) and Size(2) by MICE. polyreg method wa

df <- df %>%
  mutate(
    Size = as.factor(Size),
    Revenue = as.factor(Revenue)
  )

init = mice(df, maxit = 0)

meth = init$method
predM = init$predictorMatrix
meth[c("Size")] = "polyreg"
meth[c("Revenue")] = "polyreg"
```

```
imp_rev <- mice(df, m=5, method= meth, predictorMatrix = predM, maxit = 10, seed = 20)

df <- complete(imp_rev)

#Data Cleaning(Industry)
df <- subset(df, Industry != "-1")


#Exploratory Data Analysis

#Relationship between age and average salary
ggplot(data = df, mapping = aes(x = age, y = avg_salary ))+labs(title="Age vs Average Salar

#comparing salary and company size
ggplot(data = df, aes(x = Size, y = avg_salary))+labs(title="Company Size vs Average Salary


#comparing salary distribution for each tech job
ggplot(data = df, aes(x = job_simp, y = avg_salary)) +labs(title="Tech Job vs Average Salar




#job skills needed for tech job positions
skill <- c("python_yn", "R_yn","spark","aws","excel")
NoYes <- c("1","0")
df2 <- df %>%
  dplyr::select('python_yn':'job_simp')
df2_new <- cbind(df2, skill)
df2_new2 <- cbind(df2_new, NoYes)

ggplot(df2_new2) + geom_bar(aes(x=skill, fill=NoYes),position = "dodge") + facet_wrap(~jo


#Salary Prediction
#Feature Selection and Normalization
sal_df<-
  dplyr::select(df,-c("X","Industry"))
norm_minmax <- function(x){
  (x- min(x)) /(max(x)-min(x))
}
sal_df[sapply(sal_df, is.numeric)] <- lapply(sal_df[sapply(sal_df, is.numeric)],norm_minm
sal_df[sapply(sal_df, is.character)] <- lapply(sal_df[sapply(sal_df, is.character)],as.fa
str(sal_df)


#Splitting the data into train 20% and test 80%
set.seed(20)
```

```
indice<-caret::createDataPartition(y=sal_df$avg_salary,p=0.8,list=FALSE)
sal_train<-sal_df[indice,]
sal_test<-sal_df[-indice,]

sal_train_x = sal_train[, -4]
sal_train_y = sal_train[,4]
sal_test_x = sal_test[,-4]
sal_test_y = sal_test[,4]

#Multiple Linear Regression model
# Create Multiple Regression Model
set.seed(20)
slmModel=lm(avg_salary~.,data=sal_train)
summary(slmModel)

# Predict Salary
pred_sal_slm<-predict(slmModel,sal_test_x)

# Evaluate Model
mse <- mean((sal_test_y - pred_sal_slm)^2)
mae <- MAE(sal_test_y, pred_sal_slm)
rmse <-RMSE(sal_test_y,pred_sal_slm)
r2 <- R2(sal_test_y, pred_sal_slm)
model_metrics_lm <- cbind(mse,mae,rmse,r2)
row.names(model_metrics_lm)<-"Multiple␣Linear␣Regression"
model_metrics_lm
overall<-rbind(overall,model_metrics_lm)


#Random Forest
#Create Random Forest Model
set.seed(20)
RFModel = randomForest(x = sal_train_x,
                       y = sal_train_y,
                       ntree = 500)

#Predict Salary
pred_sal_RF<-predict(RFModel,sal_test_x)
#Evaluate Model
mse = mean((sal_test_y - pred_sal_RF)^2)
mae = MAE(sal_test_y , pred_sal_RF)
rmse =RMSE(sal_test_y ,pred_sal_RF)
r2 = R2(sal_test_y, pred_sal_RF)
model_metrics_RF <- cbind(mse,mae,rmse,r2)
row.names(model_metrics_RF)<-"random␣forest"
model_metrics_RF
overall<-rbind(model_metrics_lm,model_metrics_RF)
```

```
#XGBoost
set.seed(20)
xgb_train = xgb.DMatrix(data = data.matrix(sal_train_x), label = sal_train_y)
xgb_test = xgb.DMatrix(data = data.matrix(sal_test_x), label = sal_test_y)
xgboostModel = xgboost(data = xgb_train, max.depth = 3, nrounds = 100, verbose = 0)

#Predict Salary
pred_sal_xgb = predict(xgboostModel, xgb_test)
#Evaluate Model
mse = mean((sal_test_y - pred_sal_xgb)^2)
mae = MAE(sal_test_y , pred_sal_xgb)
rmse =RMSE(sal_test_y ,pred_sal_xgb)
r2 = R2(sal_test_y, pred_sal_xgb)
model_metrics_xgb <- cbind(mse,mae,rmse,r2)
row.names(model_metrics_xgb)<-"XGBoost"
model_metrics_xgb
overall<-rbind(overall,model_metrics_xgb)
#Evaluation of regression models
overall
```