HR Analytics: Prediction of Candidates Who Stay/Leave the Company and the City Development Index

Team 3: Xinping Yu, Ruchika Venkateswaran, Yigit Demiralp, Bosoo Kim, Muyan Xie

3/1/2021

BA810 Team Project

Introduction and Motivation

The dataset chosen by our group will be used in the field of “Human Resource Analytics”.Our primary goal is to utilize the dataset to predict whether a candidate training for a data science position will accept or reject a full time offer from the company. The dataset has been compiled by a data-related company that is aiming to reduce the time and cost spent by the human resources division to rain candidates who are potential full time employees. In addition, we will also discuss how these factors impact a candidate’s decision to stay and which is the most significant factor. We will also be using the most important features to predict the city development index. Generally, cities that are more developed provide a fertile ground for the development of science, technology, culture, and innovation. The prediction of the city development index can provide HR teams with deeper insight on whether they should train candidates who belong to cities with higher development indexes.

Impact of the Predictions

Enhanced candidate experience

- Better match of job seekers to roles
- More informative pre-hire communication

Efficient and effective recruitment

- Better prioritization of job requisitions
- Accelerated time-to-hire
- Identification of the most qualified candidates
- Minimizing the impact of employee turnover

Description of the Dataset

Our dataset has 19,158 observations and 14 features and approximately 8% of the dataset consists of missing values. The features of our dataset have been listed below

- enrollee_id : Unique ID for candidate
- city: City code
- city_ development _index : Development index of the city (scaled)
- gender: Gender
- relevent_experience: Relevant experience of candidate

- enrolled_university: Type of University course enrolled if any
- education_level: Education level of candidate
- major_discipline :Education major discipline of candidate
- experience: Candidate total experience in years
- company_size: No of employees in current employer's company
- company_type : Type of current employer
- Last new job: Difference in years between previous job and current job
- training_hours: training hours completed
- target: 0 - Not looking for job change, 1 - Looking for a job change

---

Loading the Data

```
## Uploading data

library(groupdata2)
library(data.table)
library(ggplot2)
library(ggthemes)
library(scales)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1
```

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

```
## The following objects are masked from 'package:Matrix':
##
##     expand, pack, unpack
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v tibble  3.0.5     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::between()    masks data.table::between()
## x readr::col_factor() masks scales::col_factor()
## x purrr::discard()    masks scales::discard()
## x tidyr::expand()     masks Matrix::expand()
## x dplyr::filter()     masks stats::filter()
## x dplyr::first()      masks data.table::first()
## x dplyr::lag()        masks stats::lag()
## x dplyr::last()       masks data.table::last()
## x tidyr::pack()       masks Matrix::pack()
## x purrr::transpose()  masks data.table::transpose()
## x tidyr::unpack()     masks Matrix::unpack()
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
library(RColorBrewer)
library(leaps)
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.0.4
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library(xgboost)
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```r
library(readr)
library(stringr)
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```r
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
library(rpart)
library(rpart.plot)
library(ggridges)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
##
## Attaching package: 'viridis'
```

```
## The following object is masked from 'package:scales':
##
##     viridis_pal
```

```
library(hrbrthemes)
```

```
## NOTE: Either Arial Narrow or Roboto Condensed fonts are required to use these themes.
```

```
##       Please use hrbrthemes::import_roboto_condensed() to install Roboto Condensed and
```

```
##       if Arial Narrow is not on your system, please see https://bit.ly/arialnarrow
```

```
library(ggridges)
library(forcats)
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(randomForestExplainer)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
theme_set(theme_bw())
```

```
df <- fread("C:/Users/ruchi/OneDrive/Documents/Ruchika/Boston University/BA810/Team Project/aug_
train.csv")
df <- as.data.table(df)
```

```
#viewing the head of dataset
head(df)
```

```
##    enrollee_id     city city_development_index gender     relevent_experience
## 1:        8949 city_103                  0.920   Male Has relevent experience
## 2:       29725  city_40                  0.776   Male  No relevent experience
## 3:       11561  city_21                  0.624         No relevent experience
## 4:       33241 city_115                  0.789         No relevent experience
## 5:         666 city_162                  0.767   Male Has relevent experience
## 6:       21651 city_176                  0.764        Has relevent experience
##    enrolled_university education_level major_discipline experience company_size
## 1:       no_enrollment        Graduate             STEM        >20
## 2:       no_enrollment        Graduate             STEM         15        50-99
## 3:    Full time course        Graduate             STEM          5
## 4:                            Graduate  Business Degree         <1
## 5:       no_enrollment         Masters             STEM        >20        50-99
## 6:    Part time course        Graduate             STEM         11
##      company_type last_new_job training_hours target
## 1:                           1             36      1
## 2:        Pvt Ltd           >4             47      0
## 3:                       never             83      0
## 4:        Pvt Ltd        never             52      1
## 5: Funded Startup            4              8      0
## 6:                           1             24      1
```

```
#structure of the dataset
str(df)
```

```
## Classes 'data.table' and 'data.frame':   19158 obs. of  14 variables:
##  $ enrollee_id           : int  8949 29725 11561 33241 666 21651 28806 402 27107 699 ...
##  $ city                  : chr  "city_103" "city_40" "city_21" "city_115" ...
##  $ city_development_index: num  0.92 0.776 0.624 0.789 0.767 0.764 0.92 0.762 0.92 0.92 ...
##  $ gender                : chr  "Male" "Male" "" "" ...
##  $ relevent_experience   : chr  "Has relevent experience" "No relevent experience" "No releve
nt experience" "No relevent experience" ...
##  $ enrolled_university   : chr  "no_enrollment" "no_enrollment" "Full time course" "" ...
##  $ education_level       : chr  "Graduate" "Graduate" "Graduate" "Graduate" ...
##  $ major_discipline      : chr  "STEM" "STEM" "STEM" "Business Degree" ...
##  $ experience            : chr  ">20" "15" "5" "<1" ...
##  $ company_size          : chr  "" "50-99" "" "" ...
##  $ company_type          : chr  "" "Pvt Ltd" "" "Pvt Ltd" ...
##  $ last_new_job          : chr  "1" ">4" "never" "never" ...
##  $ training_hours        : int  36 47 83 52 8 24 24 18 46 123 ...
##  $ target                : num  1 0 0 1 0 1 0 1 1 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
#summary to view data types and missing values
summary(df)
```

```
##   enrollee_id        city           city_development_index    gender
##  Min.   :    1   Length:19158       Min.   :0.4480         Length:19158
##  1st Qu.: 8554   Class :character   1st Qu.:0.7400         Class :character
##  Median :16983   Mode  :character   Median :0.9030         Mode  :character
##  Mean   :16875                      Mean   :0.8288
##  3rd Qu.:25170                      3rd Qu.:0.9200
##  Max.   :33380                      Max.   :0.9490
##  relevent_experience enrolled_university education_level    major_discipline
##  Length:19158        Length:19158        Length:19158       Length:19158
##  Class :character    Class :character    Class :character   Class :character
##  Mode  :character    Mode  :character    Mode  :character   Mode  :character
##
##
##
##   experience        company_size       company_type       last_new_job
##  Length:19158       Length:19158       Length:19158       Length:19158
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  training_hours       target
##  Min.   :  1.00   Min.   :0.0000
##  1st Qu.: 23.00   1st Qu.:0.0000
##  Median : 47.00   Median :0.0000
##  Mean   : 65.37   Mean   :0.2493
##  3rd Qu.: 88.00   3rd Qu.:0.0000
##  Max.   :336.00   Max.   :1.0000
```

Data Cleaning

```
#assigning missing values of 'last_new_job' to 0
#updating observations with 'never' to 0
levels(df$last_new_job) <- c(levels(df$last_new_job), 0)
df$last_new_job[df$last_new_job  == 'never']  <- 0
df$last_new_job[df$last_new_job == ""]<-0
table(df$last_new_job)
```

```
##
##   >4    0    1    2    3    4
## 3290 2875 8040 2900 1024 1029
```

```
## Drop "Primary School" under column - education_level
df <-df[!(education_level) %like% "Primary School"]
```

```
#Drop missing values for each column
df[df$enrollee_id == ''] = NA
df[df$city == ''] = NA
df[df$enrolled_university == ''] = NA
df[df$city_development_index == ''] = NA
df[df$gender == ''] = NA
df[df$relevent_experience == ''] = NA
df[df$education_level == ''] = NA
df[df$major_discipline == ''] = NA
df[df$experience == ''] = NA
df[df$company_size == ''] = NA
df[df$company_type == ''] = NA
df[df$last_new_job == ''] = NA
df[df$training_hours == ''] = NA
df[df$target == ''] = NA
```

```
#Experience for more than 4 becomes 5, never becomes 0
levels(df$last_new_job) <- c(levels(df$last_new_job), '5')
df$last_new_job[df$last_new_job == '>4'] <- '5'
unique(df$last_new_job)
```

```
## [1] NA  "5" "4" "1" "3" "2" "0"
```

```
#levels(df$last_new_job) <- c(levels(df$last_new_job), '0')
df$last_new_job[df$last_new_job == 'never'] <- '0'
```

```
#Change column to numeric
df$last_new_job <- as.numeric(as.character(df$last_new_job))
```

```
#Company size for less than 10 becomes 0-9,
levels(df$company_size) <- c(levels(df$company_size), '0-9')
df$company_size[df$company_size == '<10'] <- '0-9'
```

```
#Changing 'graduate' to 'undergraduate' since we already have data for Masters candidates
df$education_level[df$education_level == 'Graduate'] <- 'Undergraduate'
```

```
df$education_level = factor(df$education_level, levels=c('Undergraduate', 'Masters', 'Phd'))
```

```
#Experience for more than 20 becomes 21
levels(df$experience) <- c(levels(df$experience), '21')
df$experience[df$experience == '>20'] <- '21'
unique(df$experience)
```

```
##  [1] NA   "15" "21" "13" "7"  "5"  "16" "11" "<1" "18" "19" "12" "10" "9"  "2"
## [16] "6"  "4"  "14" "3"  "8"  "17" "20" "1"
```

```
#Experience for less than 1 becomes 0
levels(df$experience) <- c(levels(df$experience), '0')
df$experience[df$experience == '<1'] <- '0'
```

```
#Change column to numeric
df$experience <- as.numeric(as.character(df$experience))
```

```
#replace missing values in enrolled university with 'no enrollment'
df[is.na(enrolled_university), enrolled_university := 'no_enrollment']
```

```
#confirming that there are 0 observations with null values in enrolled_university
sum(is.na(df$enrolled_university))
```

```
## [1] 0
```

```
#confirming there are no missing values for education_level and major_discipline
df[!(is.na(df$education_level)) & !(is.na(df$major_discipline))]
```

```
##       enrollee_id     city city_development_index gender
##    1:       29725  city_40                  0.776   Male
##    2:         666 city_162                  0.767   Male
##    3:         402  city_46                  0.762   Male
##    4:       27107 city_103                  0.920   Male
##    5:       23853 city_103                  0.920   Male
##   ---
## 8973:       21319  city_21                  0.624   Male
## 8974:         251 city_103                  0.920   Male
## 8975:       32313 city_160                  0.920 Female
## 8976:       29754 city_103                  0.920 Female
## 8977:       24576 city_103                  0.920   Male
##           relevent_experience enrolled_university education_level
##    1:  No relevent experience       no_enrollment   Undergraduate
##    2: Has relevent experience       no_enrollment         Masters
##    3: Has relevent experience       no_enrollment   Undergraduate
##    4: Has relevent experience       no_enrollment   Undergraduate
##    5: Has relevent experience       no_enrollment   Undergraduate
##   ---
## 8973:  No relevent experience    Full time course   Undergraduate
## 8974: Has relevent experience       no_enrollment         Masters
## 8975: Has relevent experience       no_enrollment   Undergraduate
## 8976: Has relevent experience       no_enrollment   Undergraduate
## 8977: Has relevent experience       no_enrollment   Undergraduate
##       major_discipline experience company_size   company_type last_new_job
##    1:             STEM         15        50-99        Pvt Ltd            5
##    2:             STEM         21        50-99 Funded Startup            4
##    3:             STEM         13          0-9        Pvt Ltd            5
##    4:             STEM          7        50-99        Pvt Ltd            1
##    5:             STEM          5    5000-9999        Pvt Ltd            1
##   ---
## 8973:             STEM          1      100-500        Pvt Ltd            1
## 8974:             STEM          9        50-99        Pvt Ltd            1
## 8975:             STEM         10      100-500  Public Sector            3
## 8976:       Humanities          7        10/49 Funded Startup            1
## 8977:             STEM         21        50-99        Pvt Ltd            4
##       training_hours target
##    1:             47      0
##    2:              8      0
##    3:             18      1
##    4:             46      1
##    5:            108      0
##   ---
## 8973:             52      1
## 8974:             36      1
## 8975:             23      0
## 8976:             25      0
## 8977:             44      0
```

```
# Company-size:
# 1) Impute missing values to mode ("50-99" has the highest frequency)
a <- table(df$company_size)
# count the values
a
```

```
## 
##       0-9     10/49   100-500 1000-4999    10000+     50-99   500-999 5000-9999 
##       841       957      1817       931      1453      1992       593       393
```

```
df$company_size[is.na(df$company_size)] <- '50-99'
```

```
#confirming that there are no missing values in company size
unique(df$company_size)
```

```
## [1] "50-99"     "0-9"       "5000-9999" "1000-4999" "10/49"     "100-500"  
## [7] "10000+"    "500-999"
```

```
# 2) change '10/49' to '10-49'
levels(df$company_size) <- c(levels(df$company_size), '10-49')
df$company_size[df$company_size == '10/49'] <- '10-49'
unique(df$company_size)
```

```
## [1] "50-99"     "0-9"       "5000-9999" "1000-4999" "10-49"     "100-500"  
## [7] "10000+"    "500-999"
```

```
# 4) Company-type: drop missing values
df <- na.omit(df)
sum(is.na(df))
```

```
## [1] 0
```

```
str(df)
```

```
## Classes 'data.table' and 'data.frame':   8977 obs. of  14 variables:
##  $ enrollee_id           : int  29725 666 402 27107 23853 25619 6588 31972 19061 7041 ...
##  $ city                  : chr  "city_40" "city_162" "city_46" "city_103" ...
##  $ city_development_index: num  0.776 0.767 0.762 0.92 0.92 0.913 0.926 0.843 0.926 0.776 ...
##  $ gender                : chr  "Male" "Male" "Male" "Male" ...
##  $ relevent_experience   : chr  "No relevent experience" "Has relevent experience" "Has relev
ent experience" "Has relevent experience" ...
##  $ enrolled_university   : chr  "no_enrollment" "no_enrollment" "no_enrollment" "no_enrollmen
t" ...
##  $ education_level       : Factor w/ 3 levels "Undergraduate",..: 1 2 1 1 1 1 1 2 2 1 ...
##  $ major_discipline      : chr  "STEM" "STEM" "STEM" "STEM" ...
##  $ experience            : num  15 21 13 7 5 21 16 11 11 0 ...
##  $ company_size          : chr  "50-99" "50-99" "0-9" "50-99" ...
##   ..- attr(*, "levels")= chr [1:2] "0-9" "10-49"
##  $ company_type          : chr  "Pvt Ltd" "Funded Startup" "Pvt Ltd" "Pvt Ltd" ...
##  $ last_new_job          : num  5 4 5 1 1 3 5 1 2 1 ...
##  $ training_hours        : int  47 8 18 46 108 23 18 68 50 65 ...
##  $ target                : num  0 0 1 1 0 0 0 0 0 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
# drop column city and enrollee_id
 df <- subset(df, select = -c(enrollee_id,city))
```

Summary of Data Cleaning

We identified missing values and also removed variables which have no predictive power. A summary of the data cleaning process has been provided below:

Last New Job

Under “last_new_job”, each candidate belongs to a scale ranging from “0” to “4” or “never”. Observations belonging to the ‘never’ category indicate that these candidates do not have previous work experience. For this reason, we have updated observations with “never” to 0. We have also assigned the missing values of “last_new_job” to 0. There is an additional category “>4” under this feature. We have assigned candidates with experience for ‘>4’ as 5, and those who ‘never have experience’ have been assigned 0.

Education Level

We noticed that some candidates only have “primary school” education experience. Since most companies require candidates of at least 18 years of age, we have removed observations with candidates educated only upto primary school levels.

Company Size And Experience

We changed the value of “<10” to “0-9”, and “10/49” to “10-49” to provide a more meaningful understanding of the categories. We also replaced NA values under “company_size” with the mode (“50-99”) since it has the highest frequency. We also changed the value under the feature “experience” from “>20” to “21”, “<1” to “0”. We then proceed to convert the “experience” feature to a numeric variable that can easily be used in our models.

University Enrollments

Before dropping NA values, we change NA values under “enrolled_university” to “no enrollment”.

Dropping Null Values

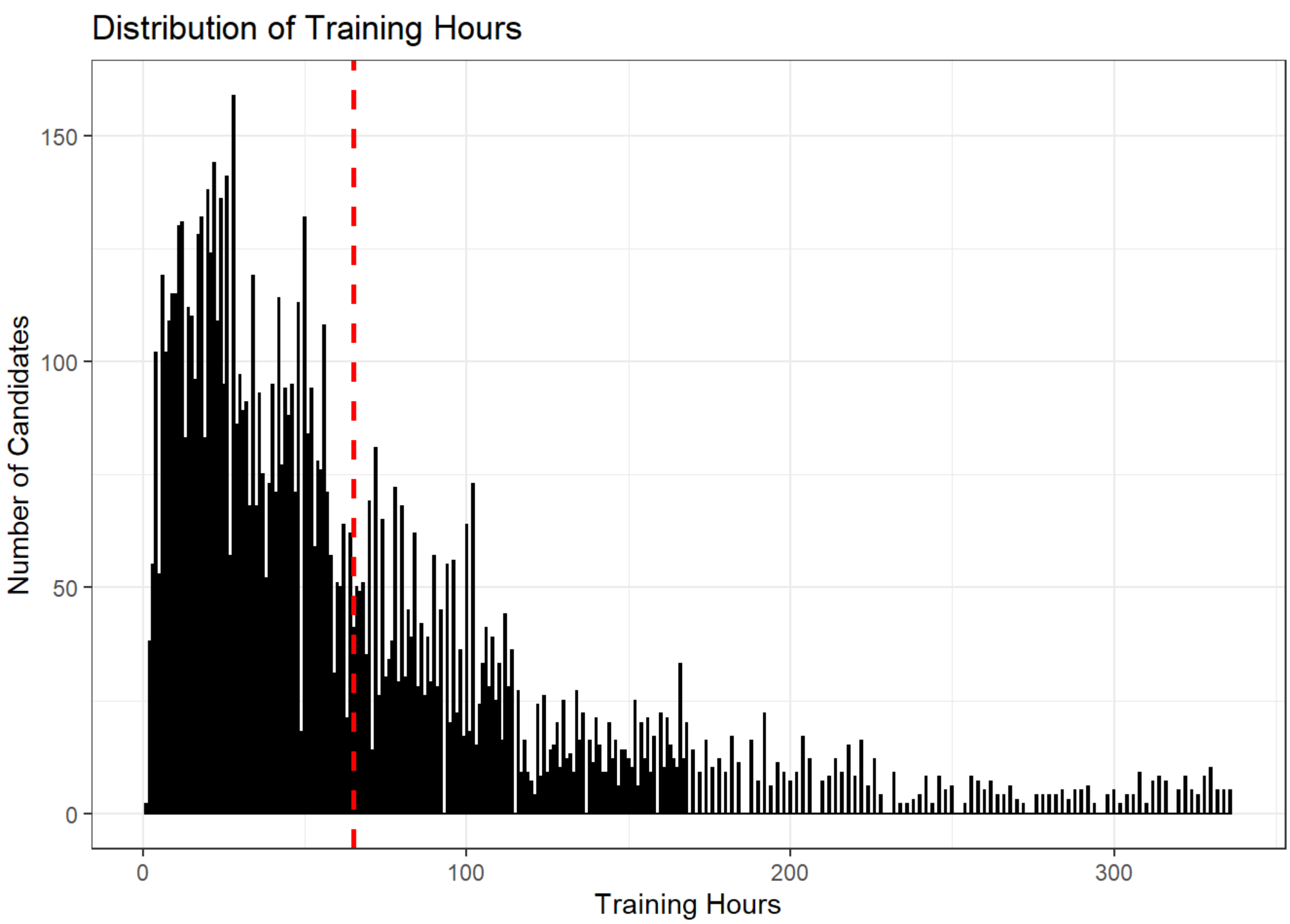
After cleaning and imputing missing values, our final step was to drop the remaining NA values under the features “company_size”, “enrolled_id”, “city”, “education_level”, and “major_discipline”.

Exploratory Data Analysis

Training Hours

The distribution below displays that the training hours is skewed to the right, with the mean number of training hours approximately around 60 hours.

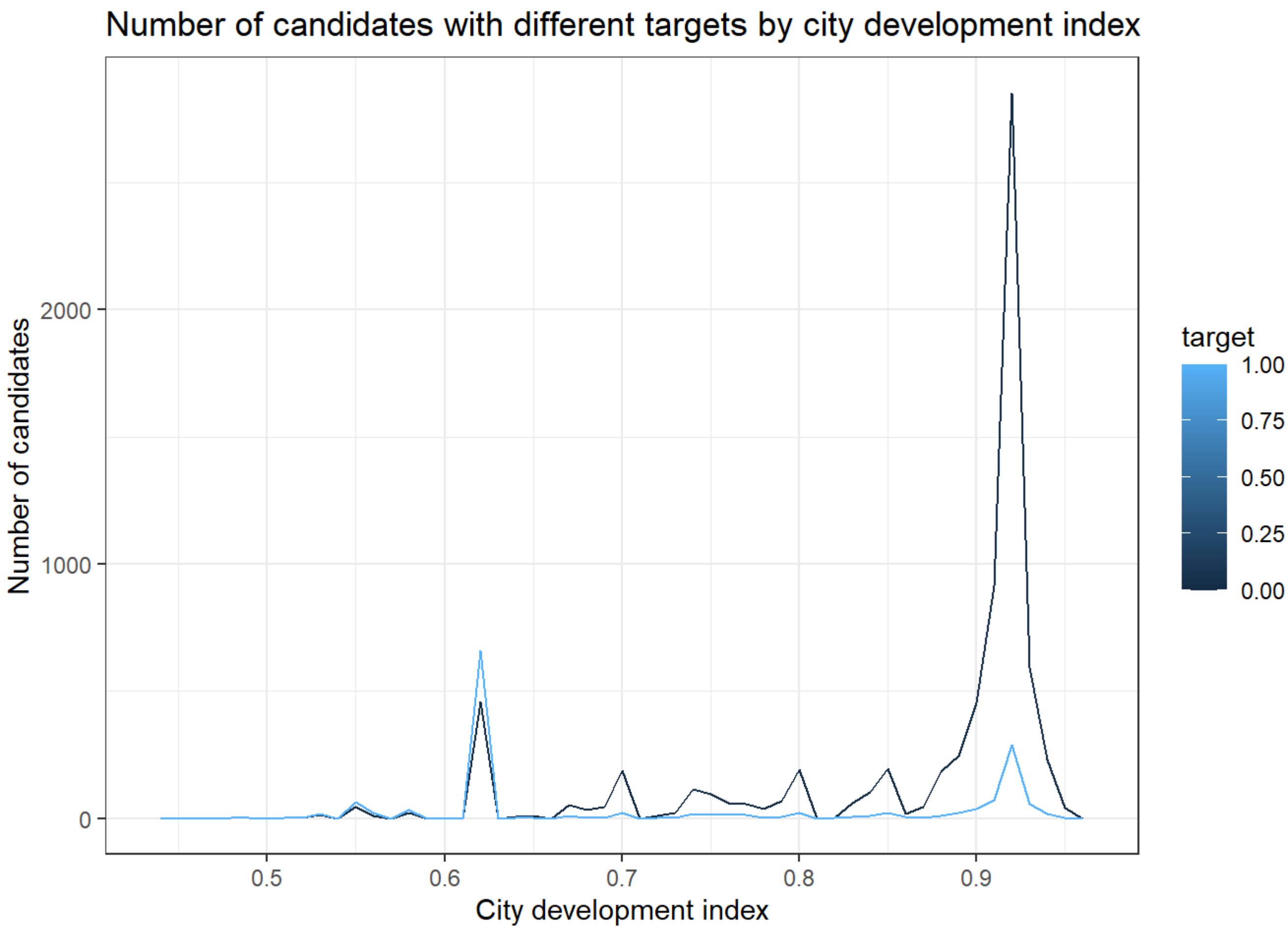
```
#plotting the distribution of training hours with the dotted line that marks the mean
ggplot(df, aes(x=training_hours)) +
  geom_histogram(binwidth=.5, colour="black", fill="white") +
  geom_vline(aes(xintercept=mean(training_hours, na.rm=T)),   # Ignore NA values for mean
             color="red", linetype="dashed", size=1) + ggtitle('Distribution of Training Hours')
+ ylab('Number of Candidates') + xlab('Training Hours')
```



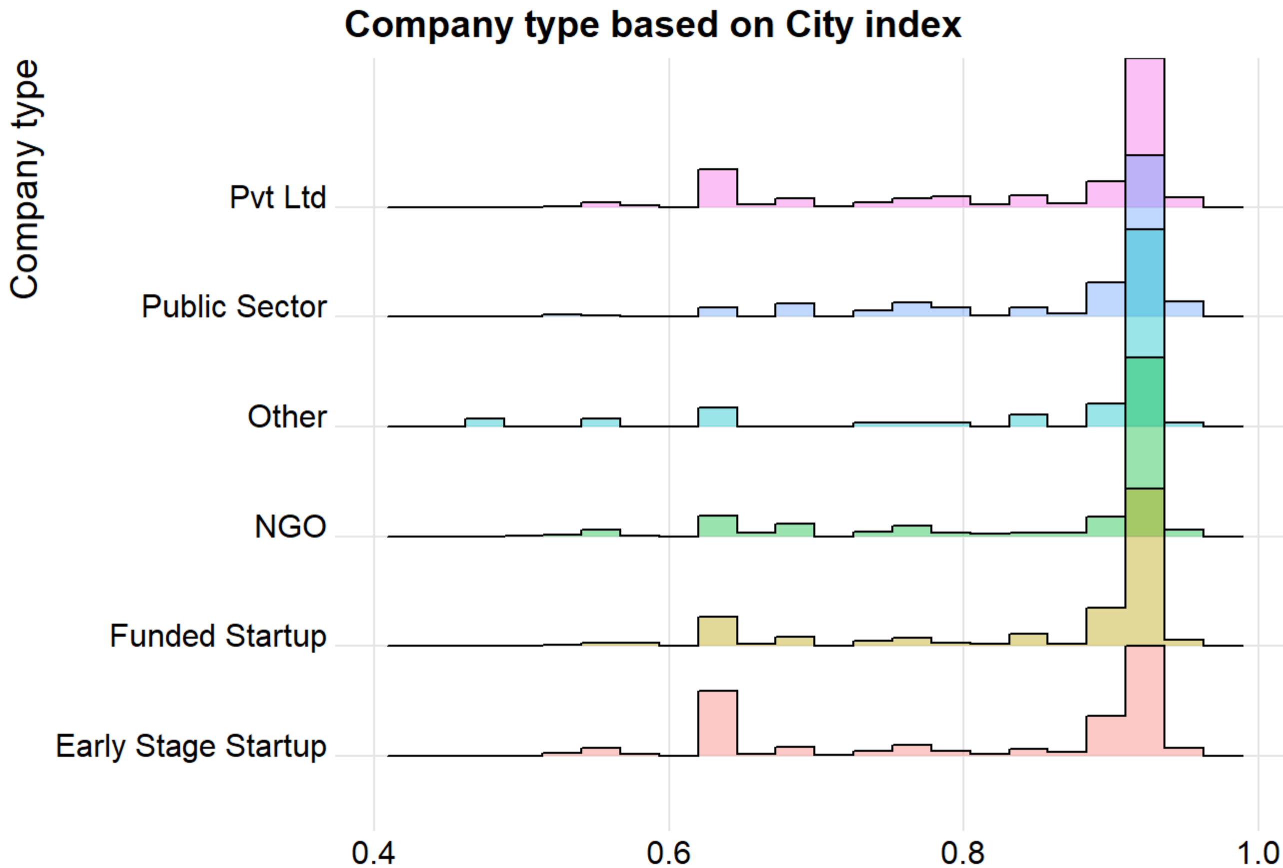
City Development Index

The charts below displays that candidates looking for job change belong to cities with lower development indexes. Candidates who belong to more developed cities are more likely to reject the job offer.

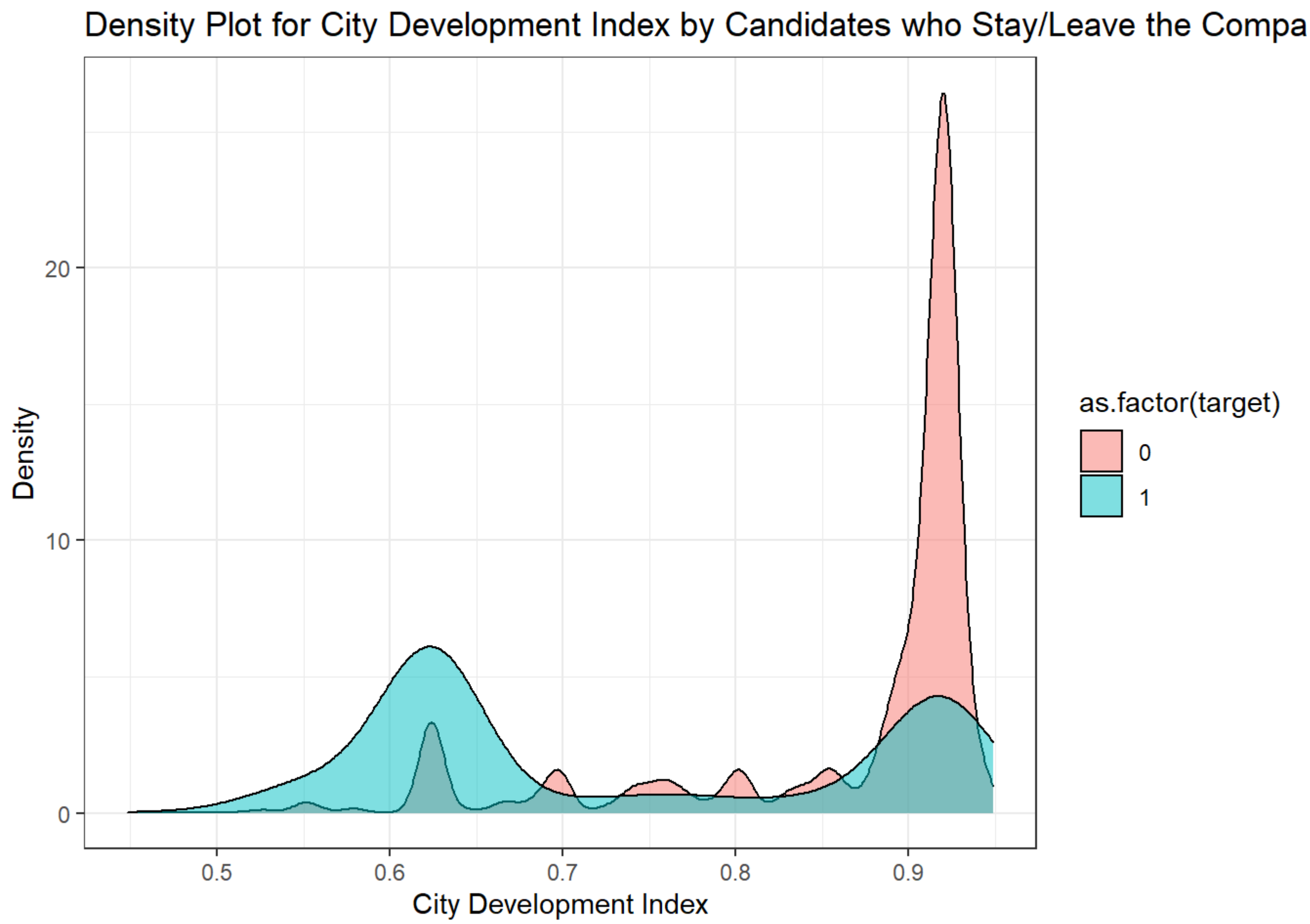
```
#discuss city development index, No change job(0,blue) vs will change job(1,black)
ggplot(data = df, mapping = aes(x = city_development_index)) +
  geom_freqpoly(mapping = aes(group = target,color = target), binwidth = .01)+
  ggtitle("Number of candidates with different targets by city development index") +
  ylab("Number of candidates") +
  xlab("City development index")
```




```
  ggplot(df, aes(y=company_type, x=city_development_index,  fill=company_type)) +
    geom_density_ridges(alpha=0.4, stat='binline', bins=20) +
    theme_ridges() +
    theme(
      legend.position='none',
      panel.spacing = unit(0.3, 'lines'),
      strip.text.x = element_text(size = 8)
    ) +
    labs(title = 'Company type based on City index') +
    xlab('') +
    ylab('Company type')
```



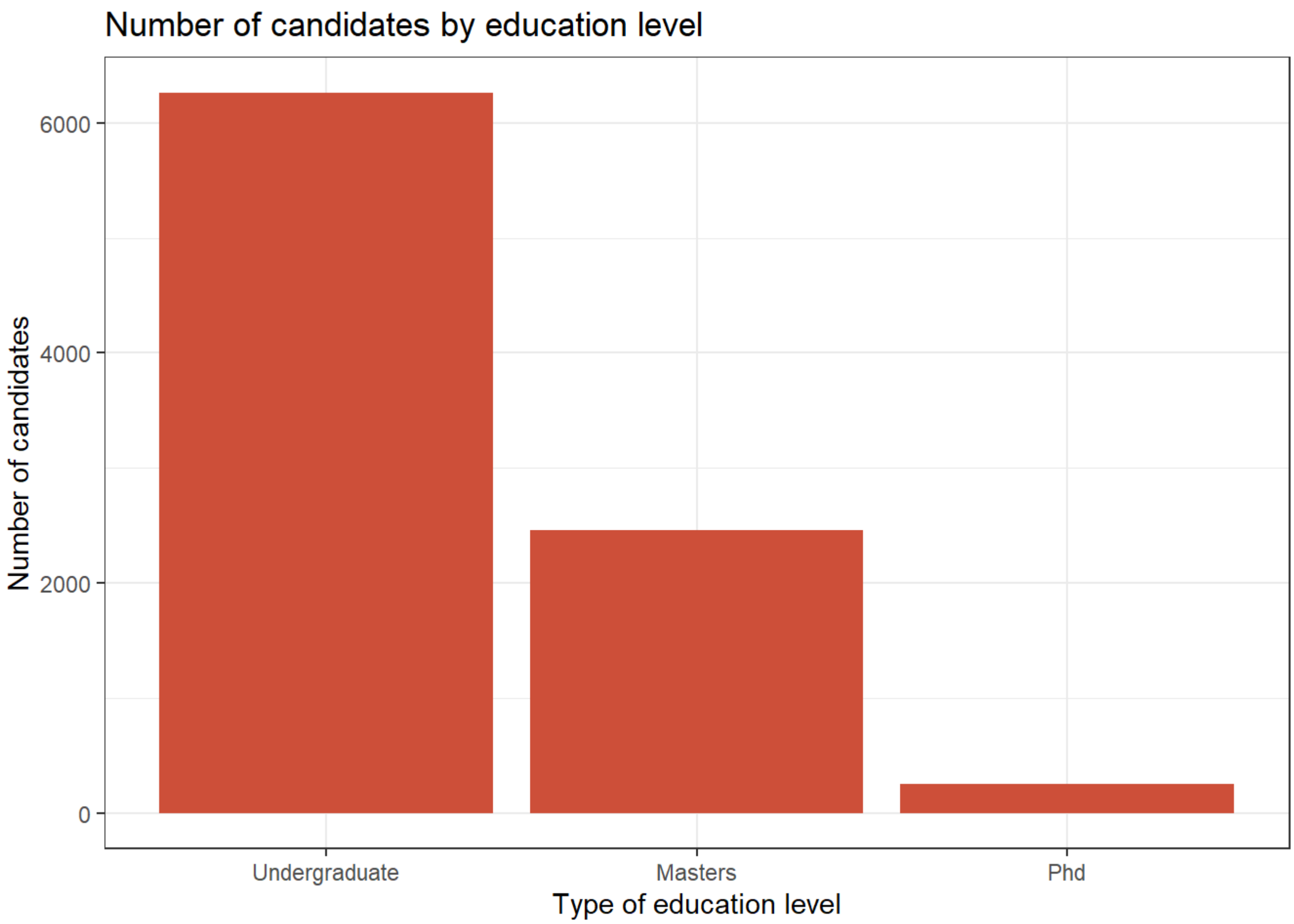

```
##city development index
ggplot(df, aes(city_development_index, fill = as.factor(target)))+
  geom_density(alpha = 0.5)+ ggtitle('Density Plot for City Development Index by Candidates who
 Stay/Leave the Company') + ylab('Density') + xlab('City Development Index')
```



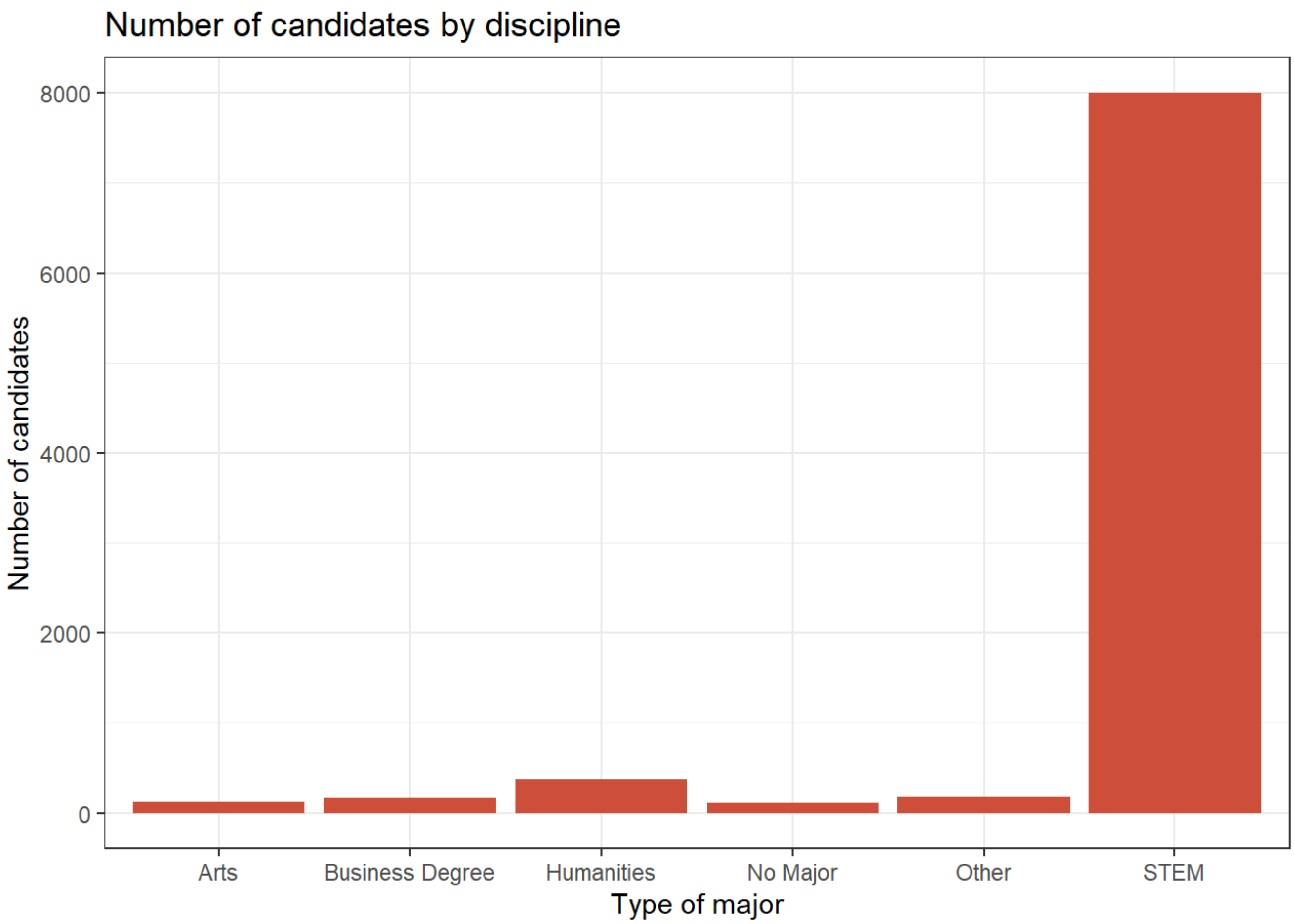
Education, Discpline Major and University Enrollments

The dataset consists of more number of students with graduate degrees, as compared to Masters and PhD degrees.Majority of the candidates have a STEM education background. The violin plot below displays that the range in the number of training hours is highest among STEM students, but generally similar ranges can be observed across all disciplines.Interestingly, candidates without any major have a smaller range in the number of training hours.

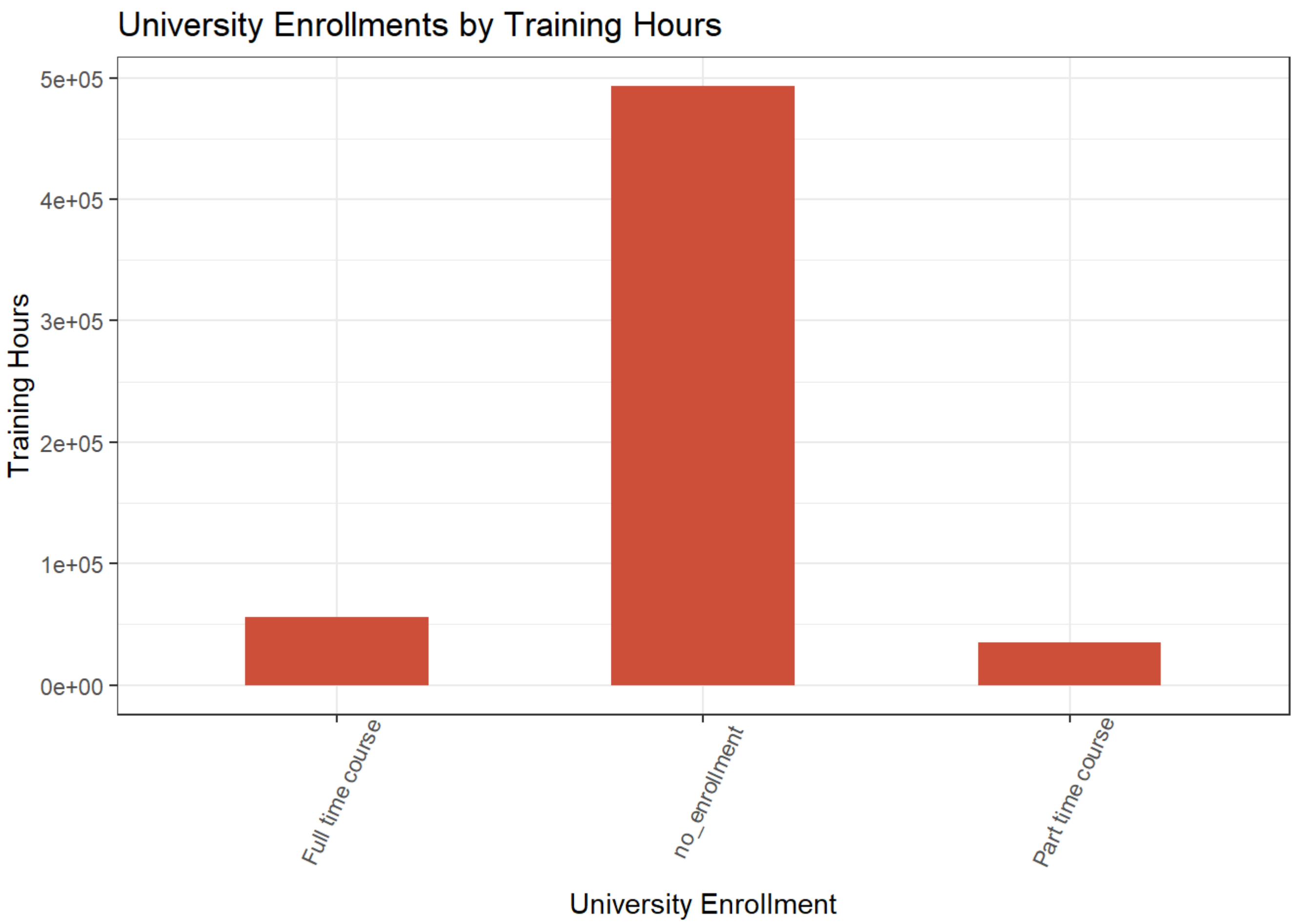
```
# count how many people in different education level where target equal to 1
phd <- df[df$education_level == 'Phd' & df$target == 1, .N ]
mas <- df[df$education_level == 'Masters' & df$target == 1, .N]
gra <- df[df$education_level == 'Undergraduate' & df$target == 1, .N]
# the percentage of people in different education level where target equal to 1
all_target <- df[df$target == 1, .N]
phd_p <- phd/all_target
mas_p <- mas/all_target
gra_p <- gra/all_target

# plot for education_level
ggplot(data = df) +
  geom_bar(mapping = aes(x = education_level), fill="tomato3") + ggtitle("Number of candidates b
y education level") + ylab("Number of candidates") + xlab("Type of education level")
```



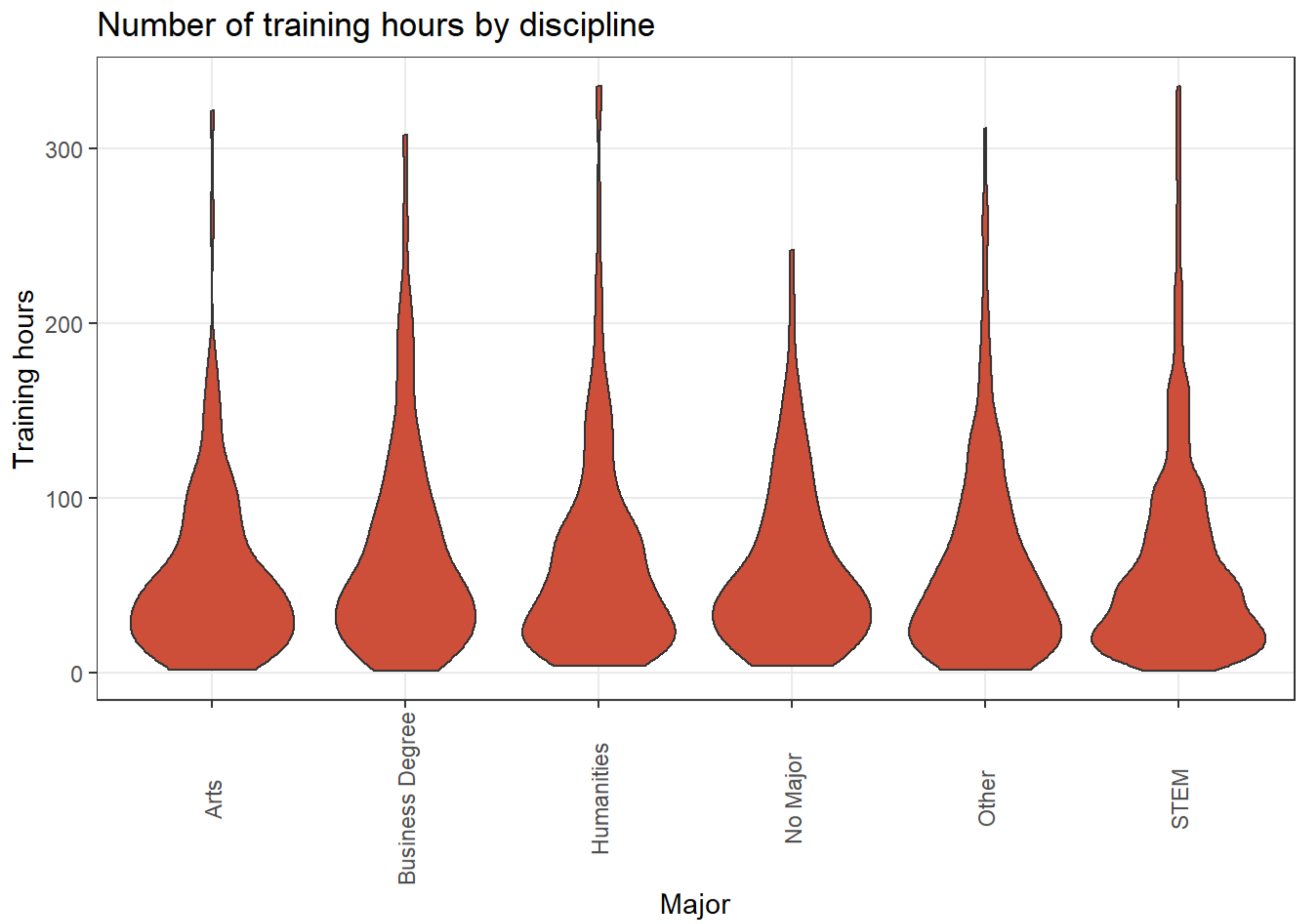

```
# count how many people in different major where target equal to 1
stem <- df[df$major_discipline == 'STEM' & df$target == 1, .N ]
hum <- df[df$major_discipline == 'Humanities' & df$target == 1, .N]
other <- df[df$major_discipline == 'Other' & df$target == 1, .N]
bus <- df[df$major_discipline == 'Business Degree' & df$target == 1, .N]
art <- df[df$major_discipline == 'Arts' & df$target == 1, .N]
# the percentage of people in different major where target equal to 1
stem_p <- stem/all_target
hum_p <- hum/all_target
other_p <- other/all_target
hum_p <- hum/all_target
art_p <- art/all_target
# plot of major discipline
ggplot(df, aes(df$major_discipline)) +
  geom_bar(fill="tomato3") + ggtitle("Number of candidates by discipline") + ylab("Number of can
didates") + xlab("Type of major")
```



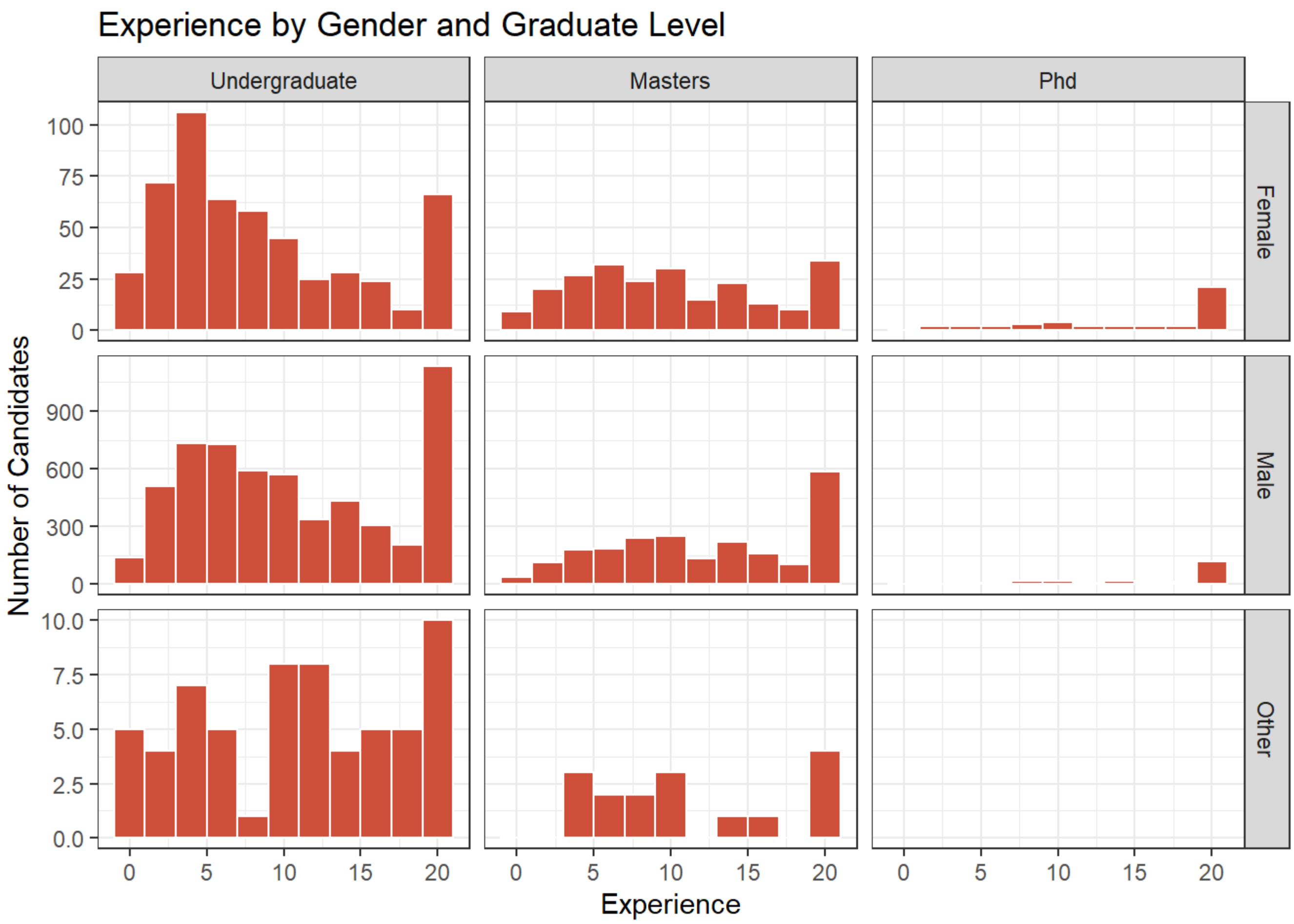

```
# University Enrollments by Training Hours
ggplot(df, aes(x=enrolled_university, y=training_hours)) +
  geom_bar(stat="identity", width=.5, fill="tomato3") +
  labs(title="University Enrollments by Training Hours") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6)) + xlab('University Enrollment') + ylab(
'Training Hours')
```



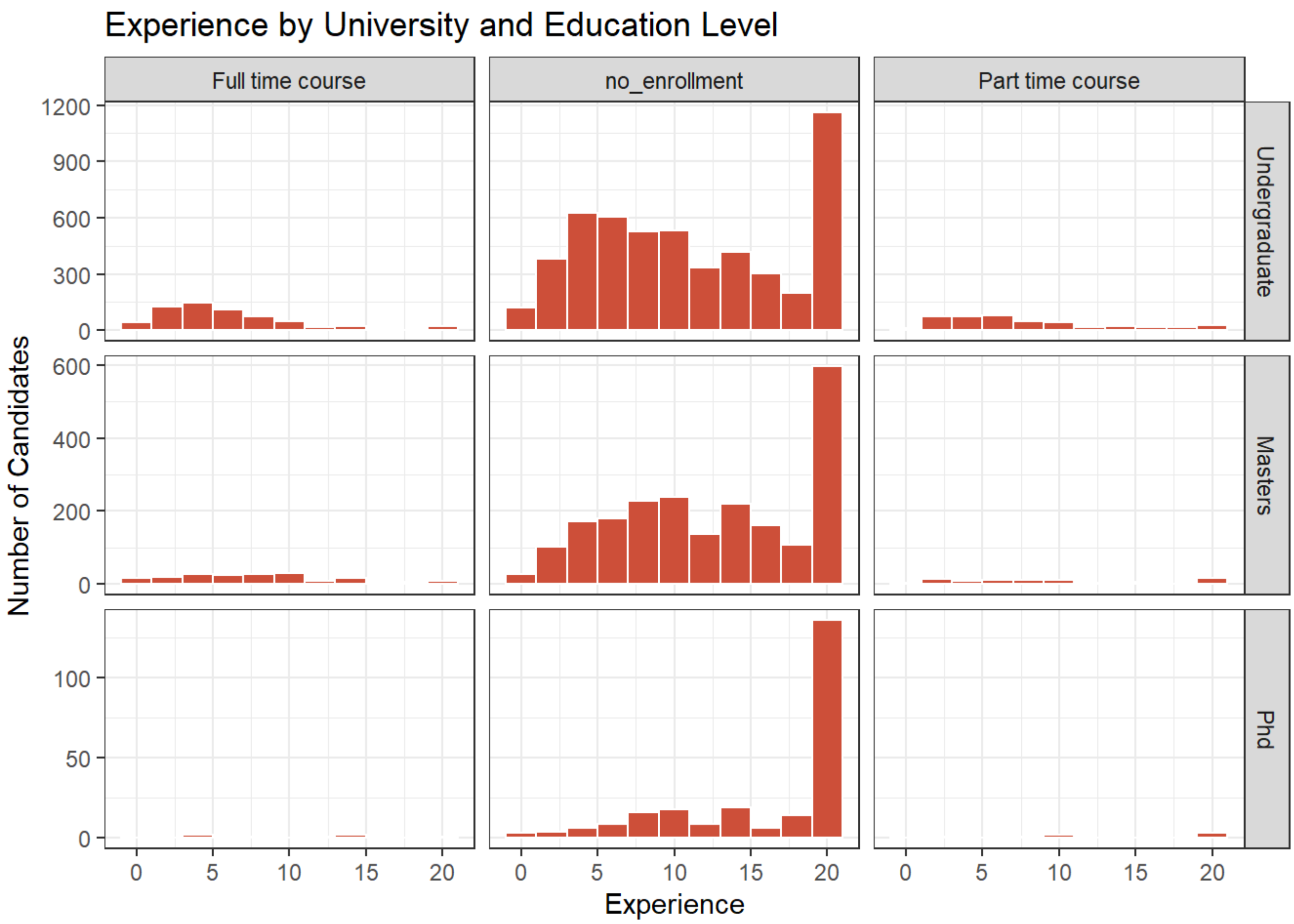

```
#Plot
g <- ggplot(df, aes(major_discipline, training_hours))
g + geom_violin(fill="tomato3") +
  labs(title="Number of training hours by discipline",
       x="Major",
       y="Training hours") +
  theme(axis.text.x = element_text(angle = 90, vjust=0.5),
        panel.grid.minor = element_blank())
```



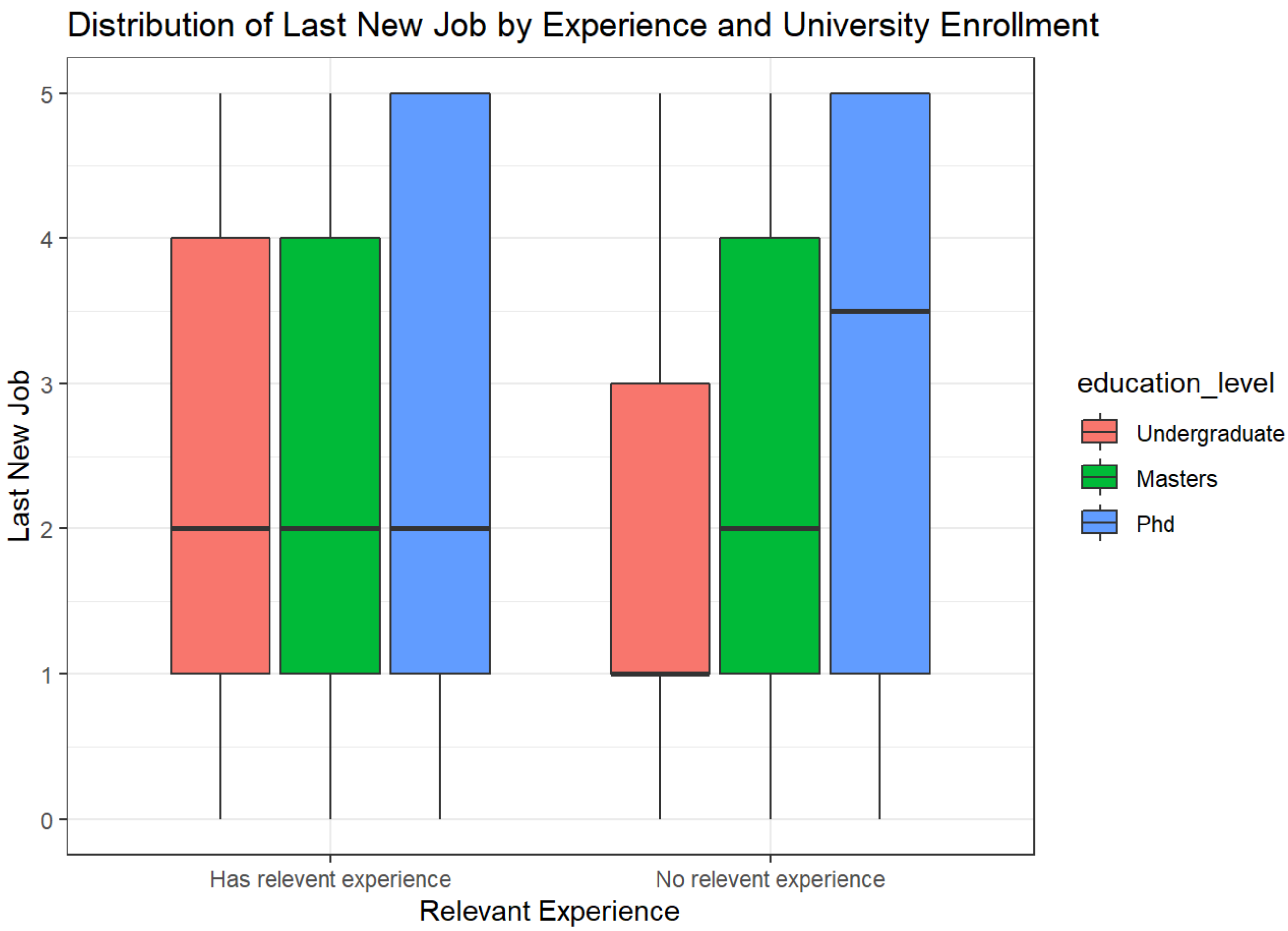

```
#defining new df 'hp' for plotting
hp <- ggplot(df, aes(x=experience)) + geom_histogram(binwidth=2,colour="white", fill="tomato3")
hp + facet_grid(gender ~ education_level, scales="free") + ggtitle('Experience by Gender and Gra
duate Level') + ylab('Number of Candidates') + xlab('Experience')
```




```
hp + facet_grid(education_level ~ enrolled_university, scales="free")+ ggtitle('Experience by Un
iversity and Education Level') + ylab('Number of Candidates') + xlab('Experience')
```



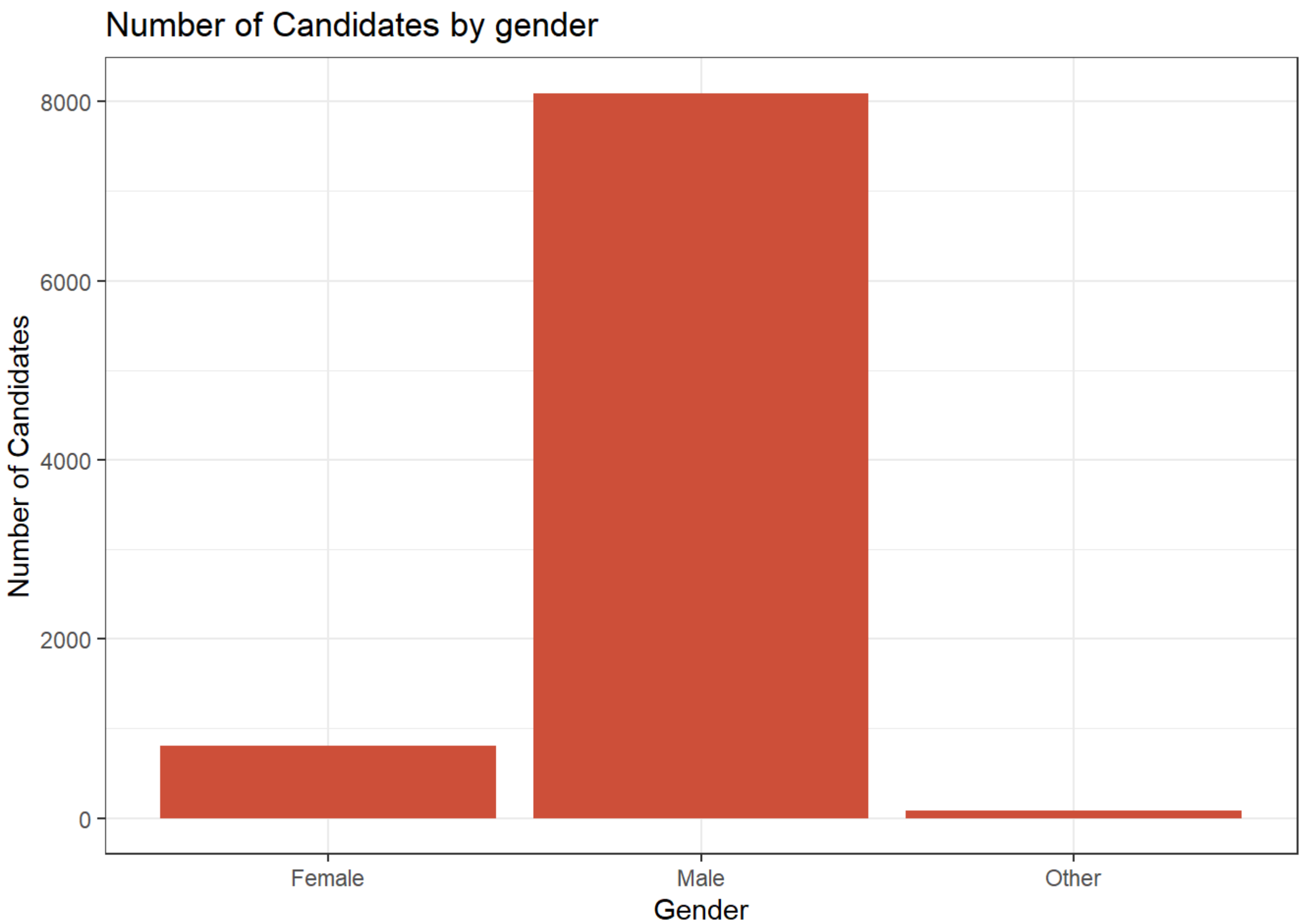

```
#examining outliers
ggplot(df, aes(x=relevent_experience, y=last_new_job, fill=education_level)) + geom_boxplot()+ g
gtitle('Distribution of Last New Job by Experience and University Enrollment') + ylab('Last New
 Job') + xlab('Relevant Experience')
```



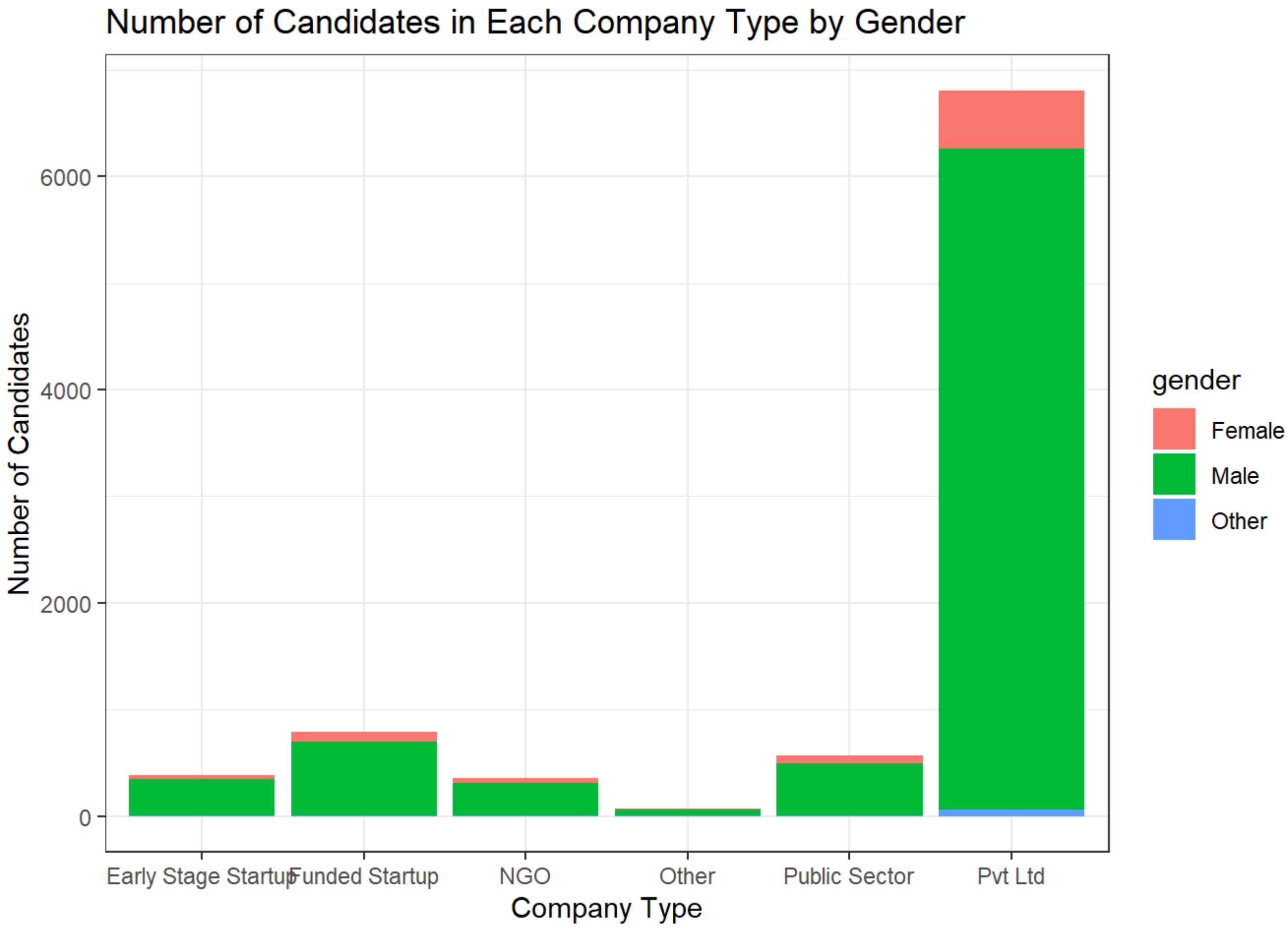
Work Experience, Company Type and Gender

The dataset has more number of males and this could be attributed to the fact that the company has not yet collected much data on candidates belonging to other genders.The dataset also consists of more number of candidates belonging to private companies with the size ranging from 50-99 employees. Interestingly, some candidates across across all genders have received training upto 300 hours.

```
# gender count
ggplot(data = df) +
  stat_count(mapping = aes(x = gender), fill="tomato3")+ ggtitle('Number of Candidates by gende
r') + ylab('Number of Candidates') + xlab('Gender')
```



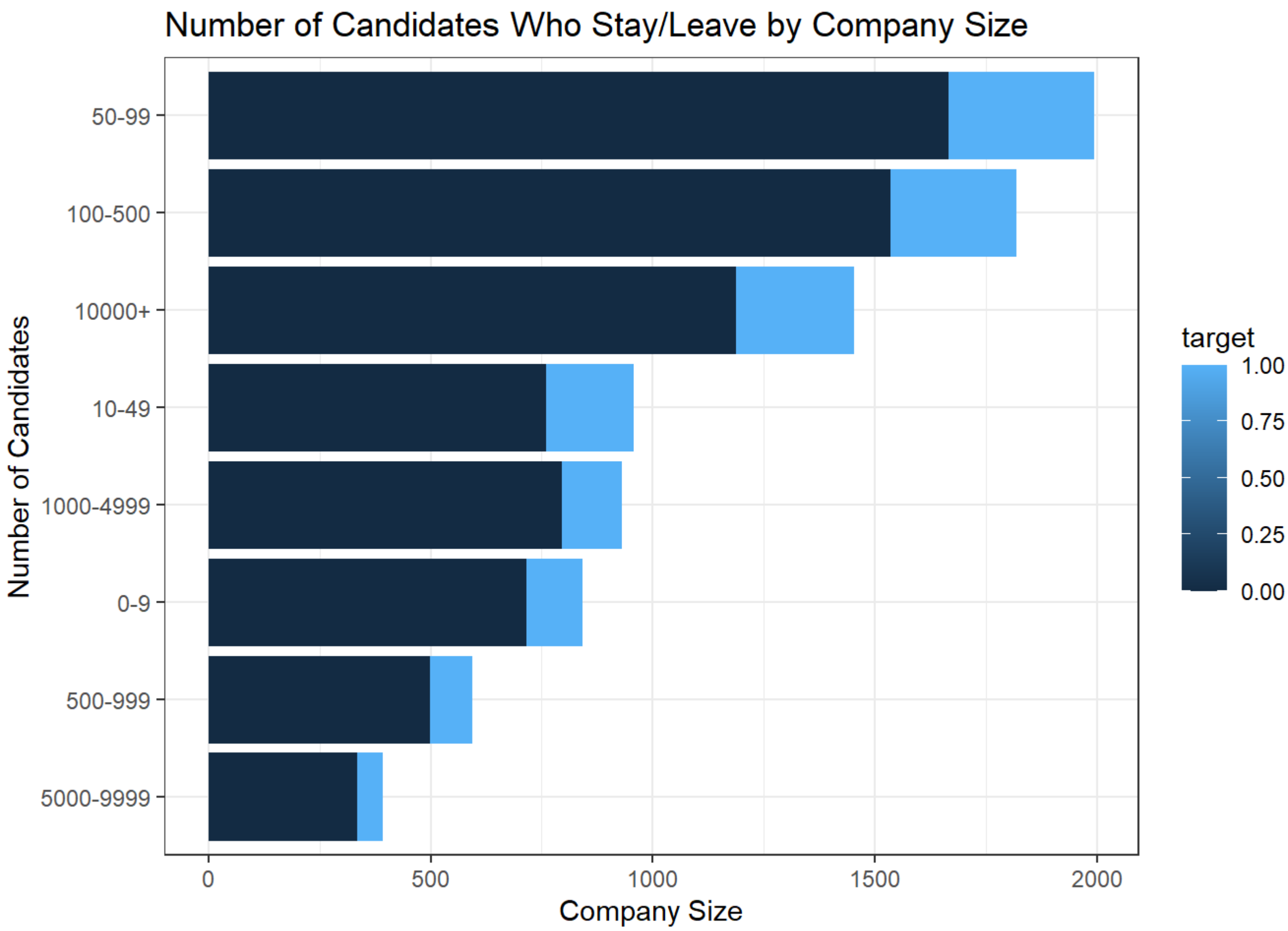

```
#gender count by company type
ggplot(data = df) + geom_bar(mapping = aes(x = company_type, fill = gender)) + ggtitle('Number o
f Candidates in Each Company Type by Gender') + ylab('Number of Candidates') + xlab('Company Typ
e')
```



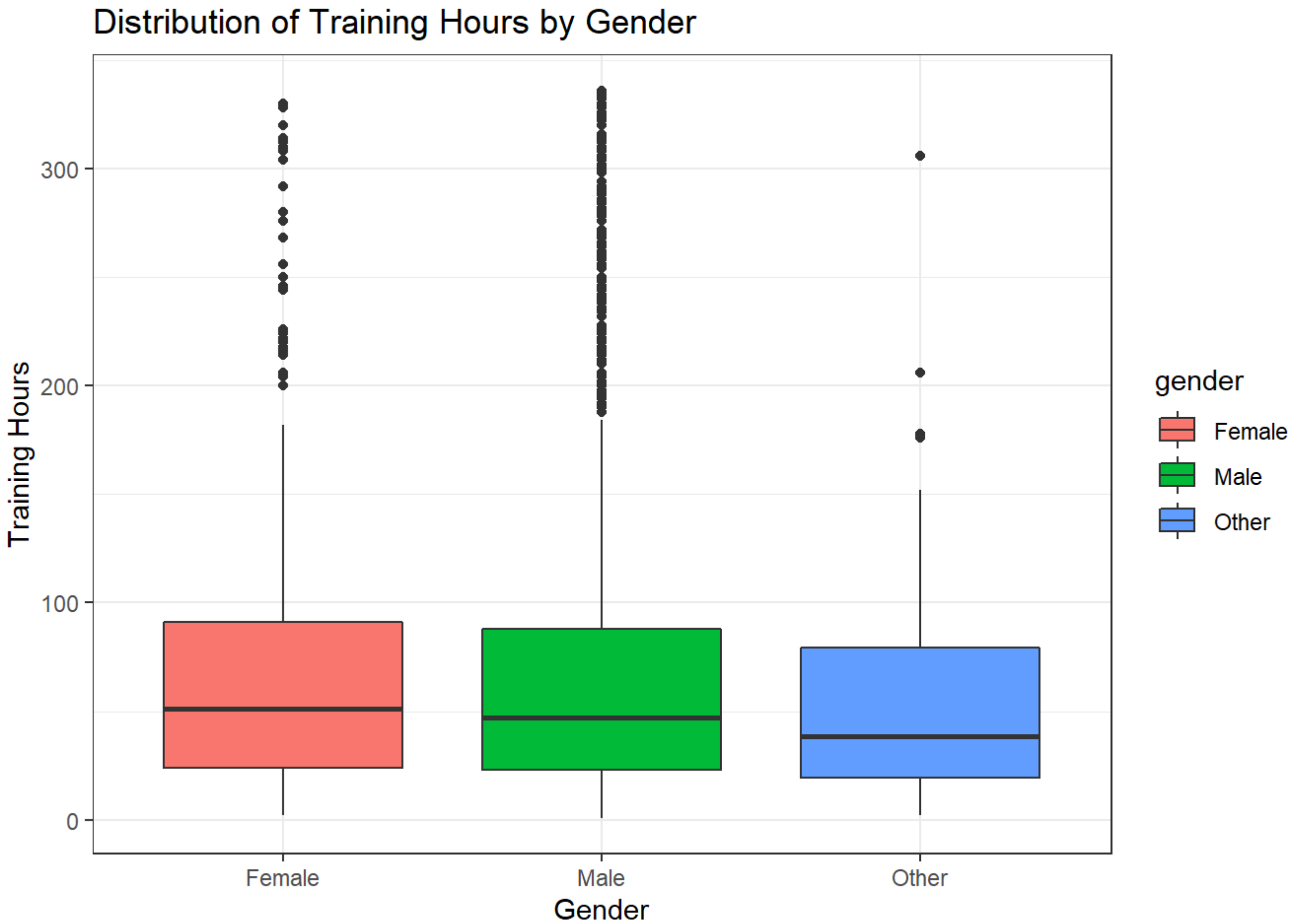
Most

of

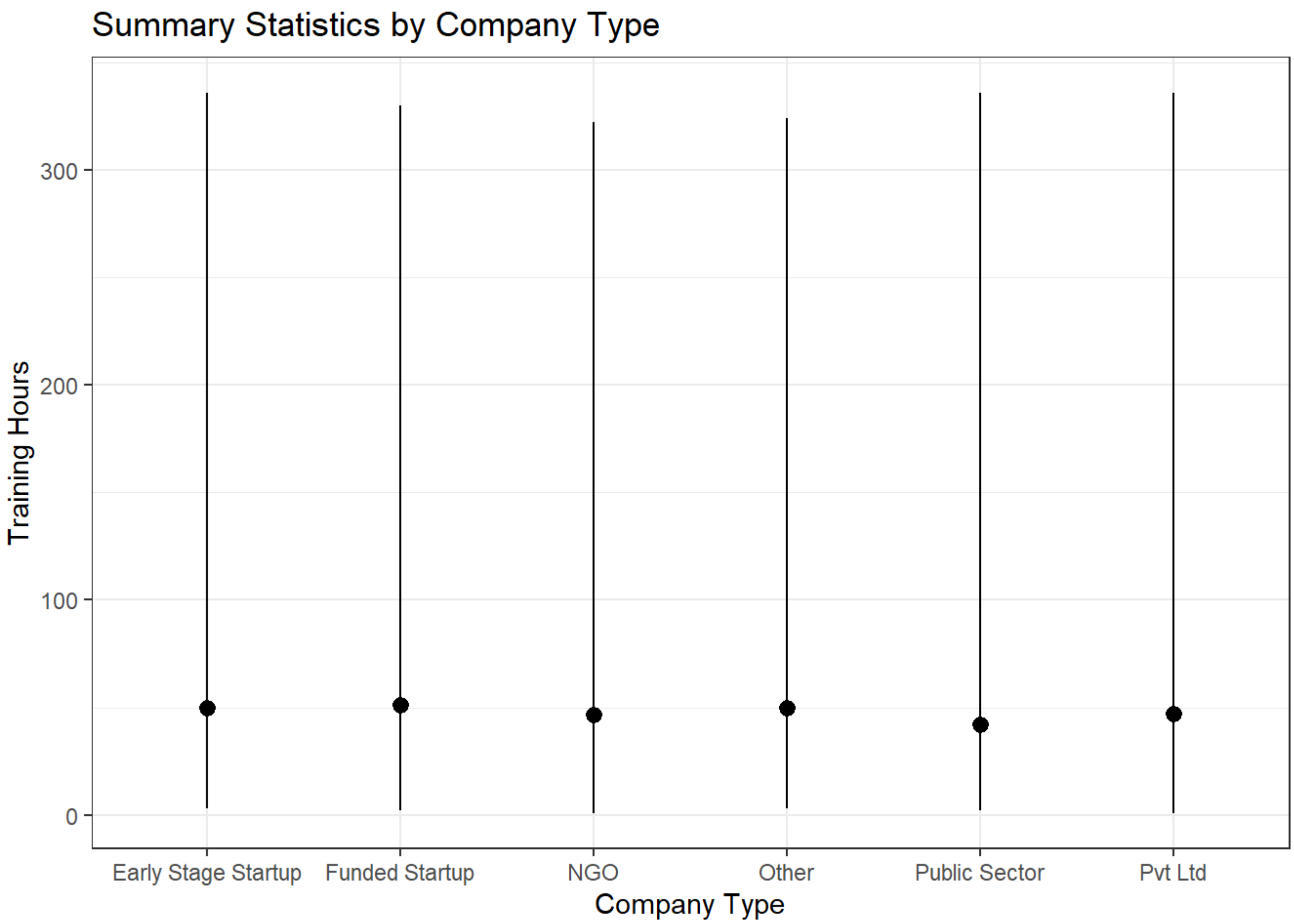
```
df %>%
  group_by(company_size) %>%
  count(target) %>%
  ggplot(aes(reorder(company_size, n), n, fill = target))+
  geom_col()+
  coord_flip() + ggtitle('Number of Candidates Who Stay/Leave by Company Size') + ylab('Company
 Size') + xlab('Number of Candidates')
```



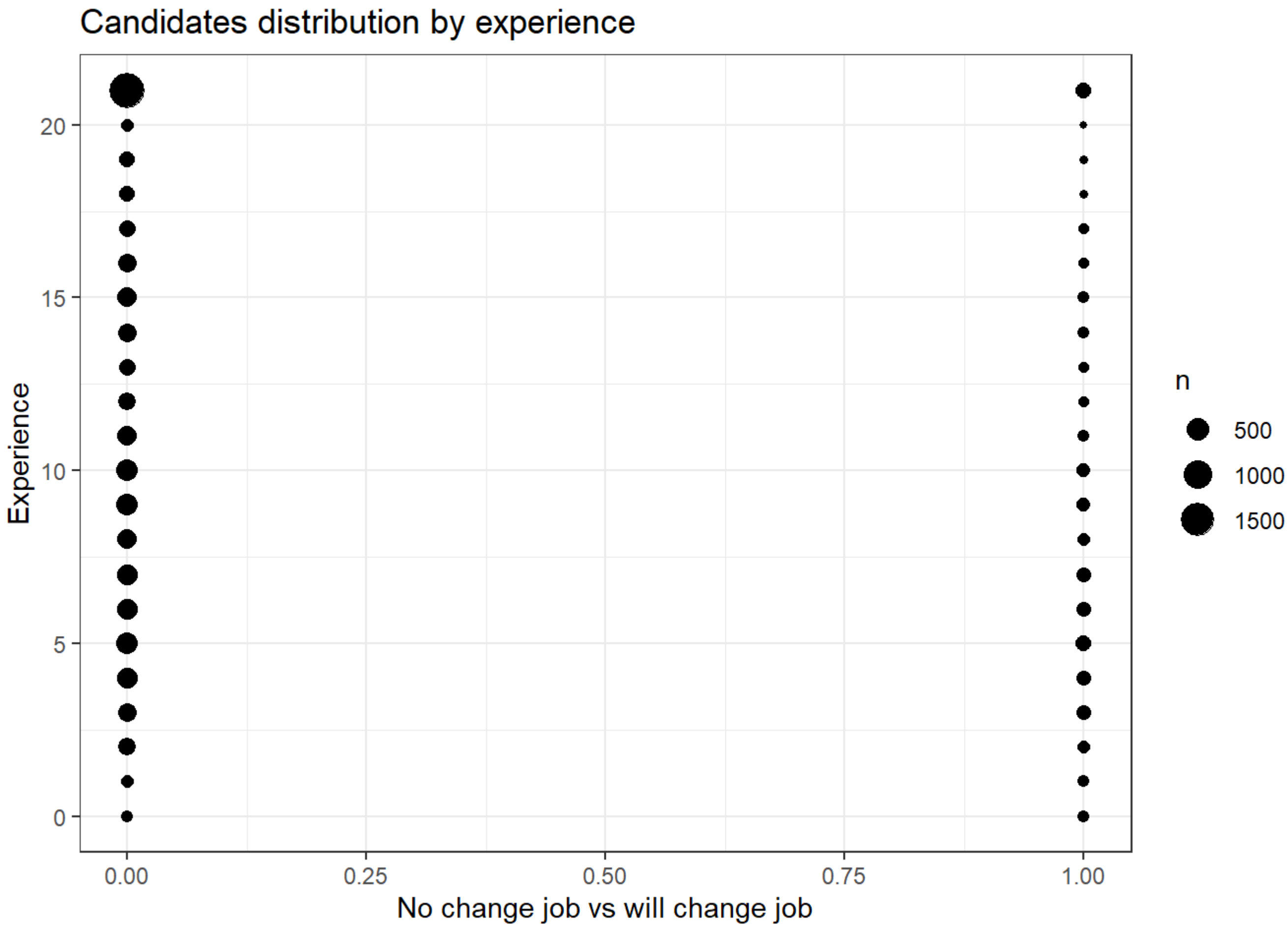

```
#examining outliers
ggplot(df, aes(x=gender, y=training_hours, fill=gender)) + geom_boxplot()+ ggtitle('Distribution of Training Hours by Gender') + ylab('Training Hours') + xlab('Gender')
```



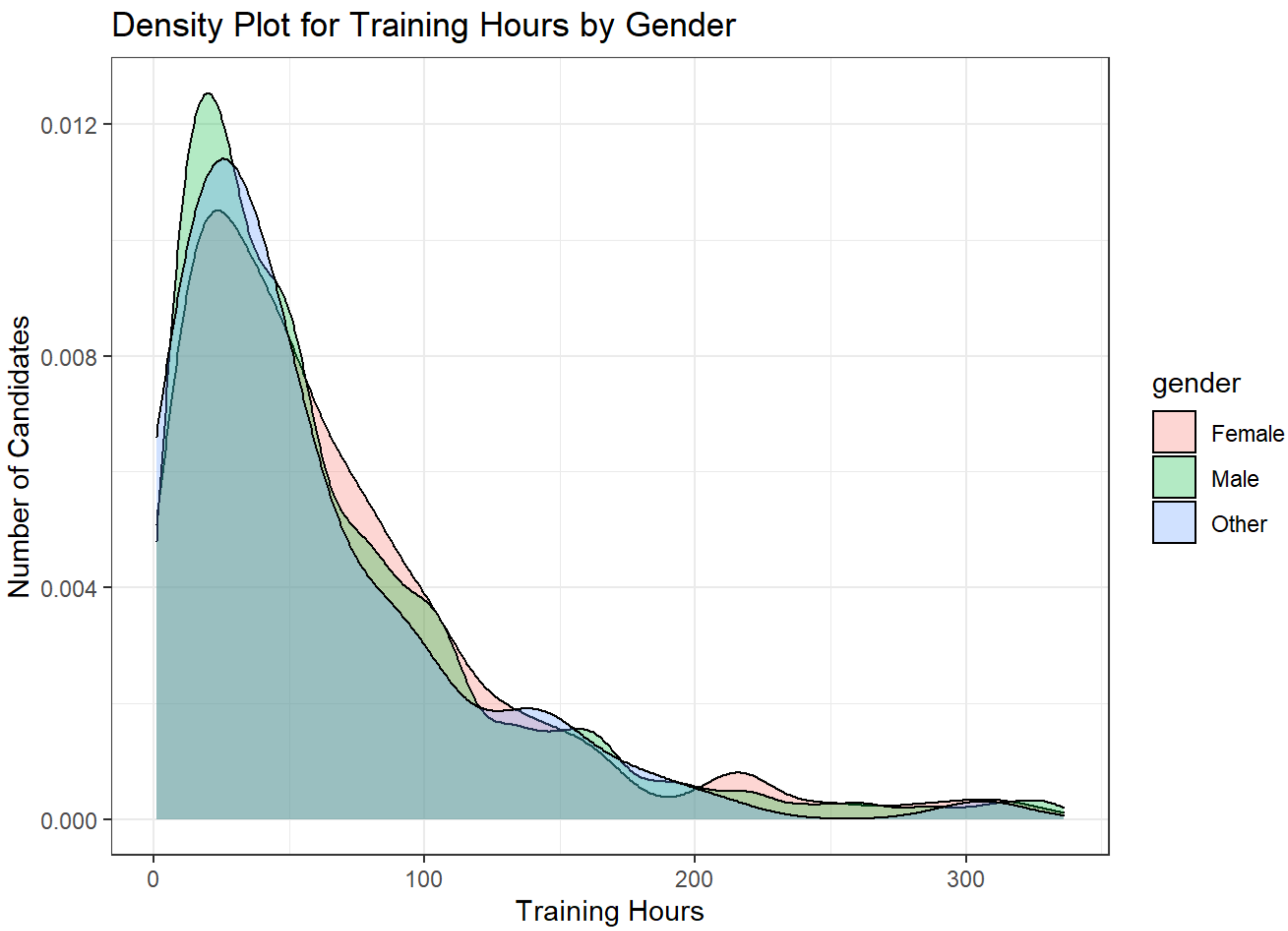

```
# min, max and median training hours
ggplot(data = df) +
  stat_summary(
    mapping = aes(x = company_type, y = training_hours),
    fun.min = min,
    fun.max = max,
    fun = median
  )+ ggtitle('Summary Statistics by Company Type') + ylab('Training Hours') + xlab('Company Typ
e')
```



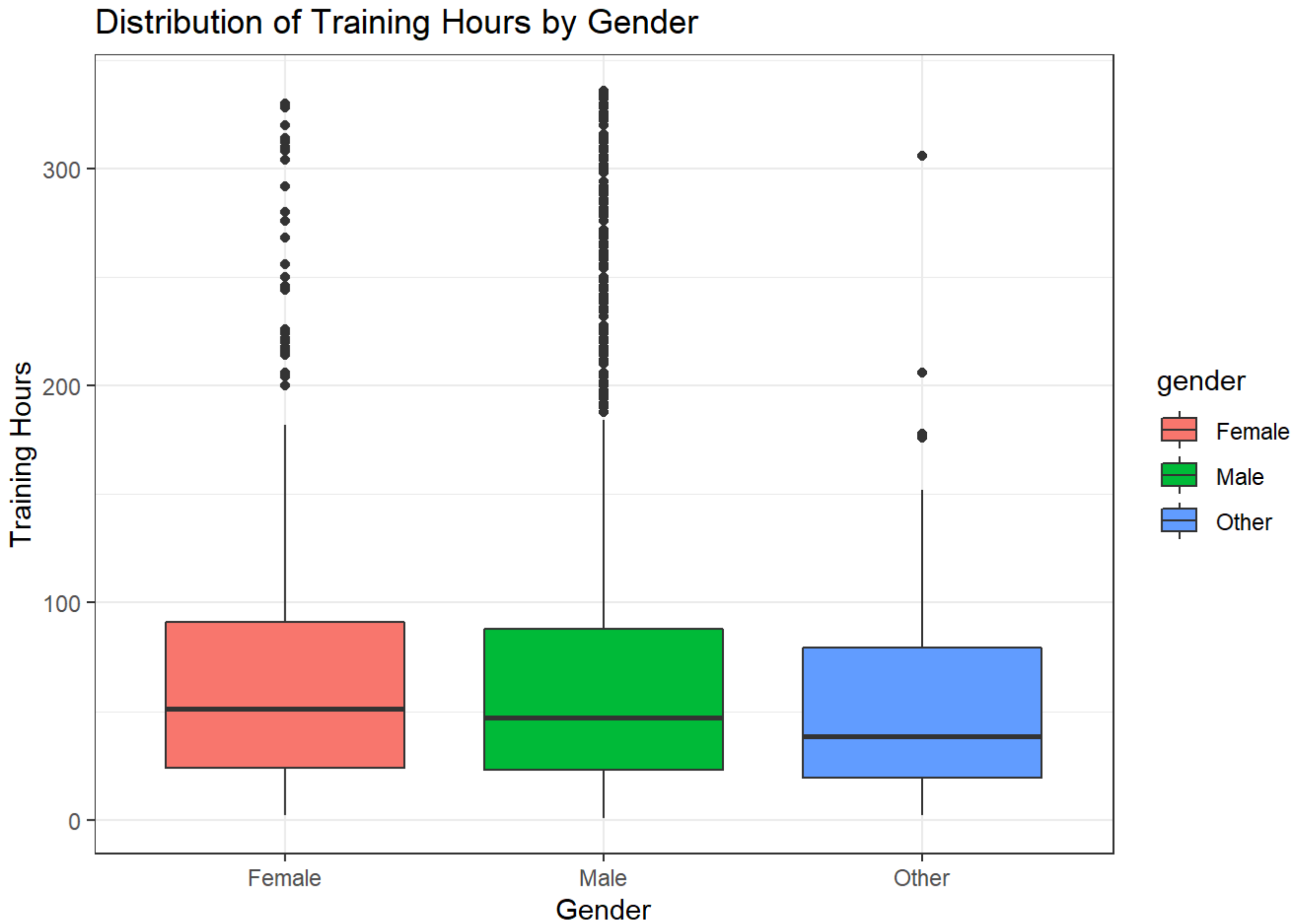

```
#experience regarding target
ggplot(data = df) +
  geom_count(mapping = aes(x = target, y = experience))+
  ggtitle("Candidates distribution by experience") +
  ylab("Experience") +
  xlab("No change job vs will change job")
```




```
# density plot for training hours by gender
ggplot(df, aes(x=training_hours, fill=gender)) + geom_density(alpha=.3)+ ggtitle('Density Plot f
or Training Hours by Gender') + ylab('Number of Candidates') + xlab('Training Hours')
```




```
#examining outliers
ggplot(df, aes(x=gender, y=training_hours, fill=gender)) + geom_boxplot() + ggtitle('Distributio
n of Training Hours by Gender') + ylab('Training Hours') + xlab('Gender')
```



***

Machine Learning (Model Building)

Classification Models to Predict Whether Candidates Will Stay or Leave the Company

```
# Drop dummy variable
dmy <- dummyVars(" ~ .", data = df, fullRank = T)
new_df <- data.frame(predict(dmy, newdata = df))
```

```
#splitting the data into test and train data
new_df$target = as.factor(new_df$target)
# Determine row to split on: split
split <- round(nrow(new_df) * 0.80)

# Create train
train_df <- new_df[1:split, ]

# Create test
test_df <- new_df[(split + 1):nrow(new_df), ]
```

Decision Tree

```
# creating the train control function for cross validation creating 5 folds. The same 5 folds wi
ll be used for all classification models
train_control <- trainControl(method = "cv",
                              number = 5)
model_dt <- train(target ~., data = train_df,
               method = "rpart",
               trControl = train_control)
print(model_dt)
```

```
## CART
##
## 7182 samples
##   28 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 5745, 5745, 5746, 5746, 5746
## Resampling results across tuning parameters:
##
##   cp           Accuracy   Kappa
##   0.003327787  0.8560291  0.4305738
##   0.008319468  0.8596490  0.4674190
##   0.166389351  0.8461432  0.2765865
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.008319468.
```

```
varImp(model_dt)
```

```
## rpart variable importance
##
##   only 20 most important variables shown (out of 28)
##
##                                          Overall
## city_development_index                   100.000
## experience                                13.933
## enrolled_universityno_enrollment           3.617
## last_new_job                               2.277
## major_disciplineSTEM                       1.397
## education_level.Phd                        0.000
## major_disciplineOther                      0.000
## company_typePvt.Ltd                        0.000
## enrolled_universityPart.time.course        0.000
## company_size50.99                          0.000
## company_typeNGO                            0.000
## company_size100.500                        0.000
## relevent_experienceNo.relevent.experience   0.000
## company_size10.49                          0.000
## major_disciplineBusiness.Degree            0.000
## company_size5000.9999                      0.000
## major_disciplineNo.Major                   0.000
## company_typeOther                          0.000
## genderMale                                 0.000
## company_typePublic.Sector                  0.000
```

```
# fit the model with test data and evaluate
class_pred <- predict(object = model_dt, newdata = test_df)
confusionMatrix(data = class_pred,reference = test_df$target, positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1392  128
##          1  117  158
##
##                Accuracy : 0.8635
##                  95% CI : (0.8468, 0.8791)
##     No Information Rate : 0.8407
##     P-Value [Acc > NIR] : 0.003933
##
##                   Kappa : 0.4824
##
##  Mcnemar's Test P-Value : 0.522903
##
##             Sensitivity : 0.55245
##             Specificity : 0.92247
##          Pos Pred Value : 0.57455
##          Neg Pred Value : 0.91579
##              Prevalence : 0.15933
##          Detection Rate : 0.08802
##    Detection Prevalence : 0.15320
##       Balanced Accuracy : 0.73746
##
##        'Positive' Class : 1
##
```

Random Forest

```
# build random forest
# apply 5-folds cross validation

model_rf <- randomForest(target ~., data = train_df,
               method = "rf",
               trControl = train_control,
               localImp = TRUE)
print(model_rf)
```

```
##
## Call:
##  randomForest(formula = target ~ ., data = train_df, method = "rf",      trControl = train_co
ntrol, localImp = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 14.7%
## Confusion matrix:
##      0   1 class.error
## 0 5633 347  0.05802676
## 1  709 493  0.58985025
```

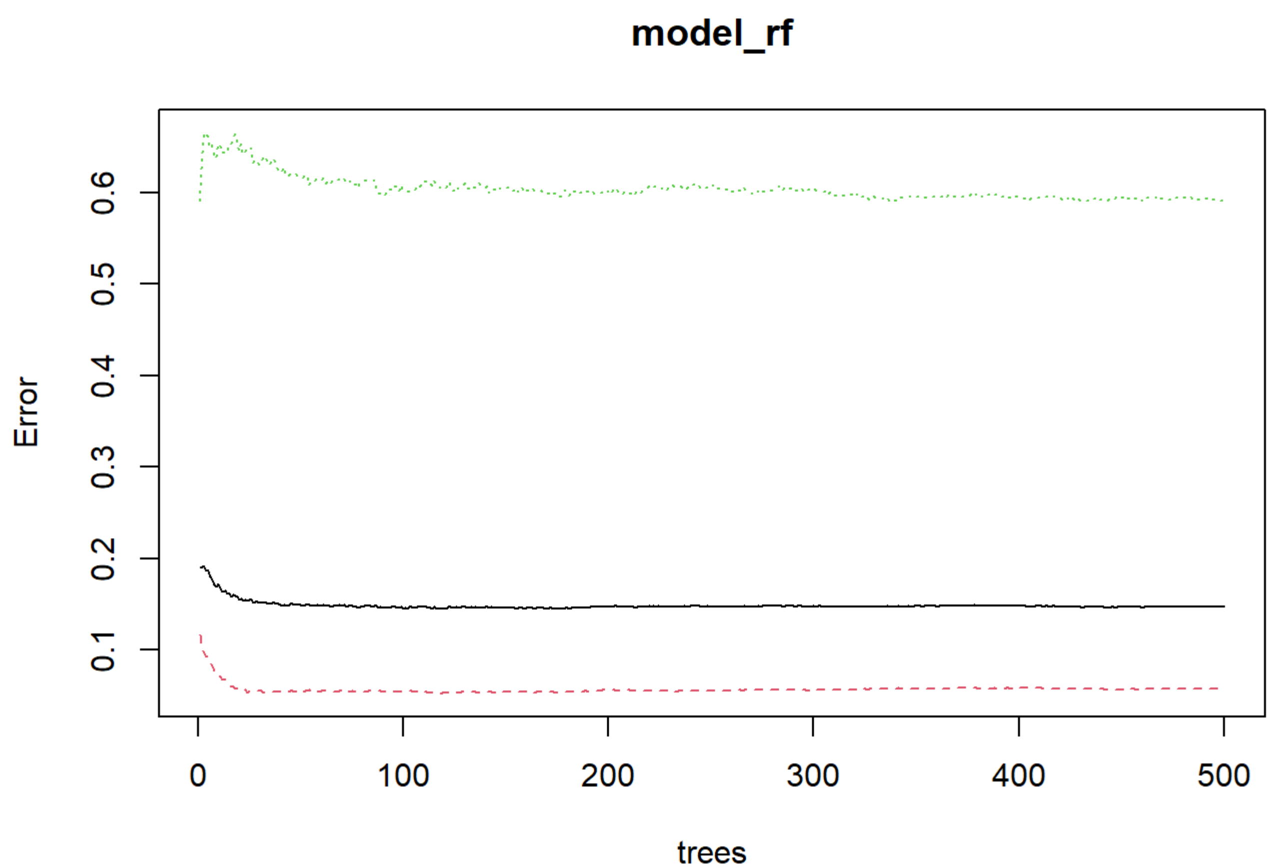
```
varImp(model_rf)
```

```
##                                                     0          1
## city_development_index                     97.6709017 97.6709017
## genderMale                                  2.8129644  2.8129644
## genderOther                                -0.1856656 -0.1856656
## relevent_experienceNo.relevent.experience  1.1188540  1.1188540
## enrolled_universityno_enrollment            4.6199493  4.6199493
## enrolled_universityPart.time.course         1.6981814  1.6981814
## education_level.Masters                     4.0112269  4.0112269
## education_level.Phd                         0.6877668  0.6877668
## major_disciplineBusiness.Degree            -2.5329340 -2.5329340
## major_disciplineHumanities                  1.8433167  1.8433167
## major_disciplineNo.Major                    2.6454497  2.6454497
## major_disciplineOther                       1.5592728  1.5592728
## major_disciplineSTEM                        1.0814137  1.0814137
## experience                                 13.1006733 13.1006733
## company_size10.49                          -1.9817724 -1.9817724
## company_size100.500                        -0.9085516 -0.9085516
## company_size1000.4999                       0.9075742  0.9075742
## company_size10000.                          1.1061541  1.1061541
## company_size50.99                          -1.0477985 -1.0477985
## company_size500.999                        -2.6244403 -2.6244403
## company_size5000.9999                       2.6051033  2.6051033
## company_typeFunded.Startup                  2.6699506  2.6699506
## company_typeNGO                             1.6131086  1.6131086
## company_typeOther                          -1.2869863 -1.2869863
## company_typePublic.Sector                   3.6446399  3.6446399
## company_typePvt.Ltd                         2.1915048  2.1915048
## last_new_job                                8.9971912  8.9971912
## training_hours                             -2.0616380 -2.0616380
```

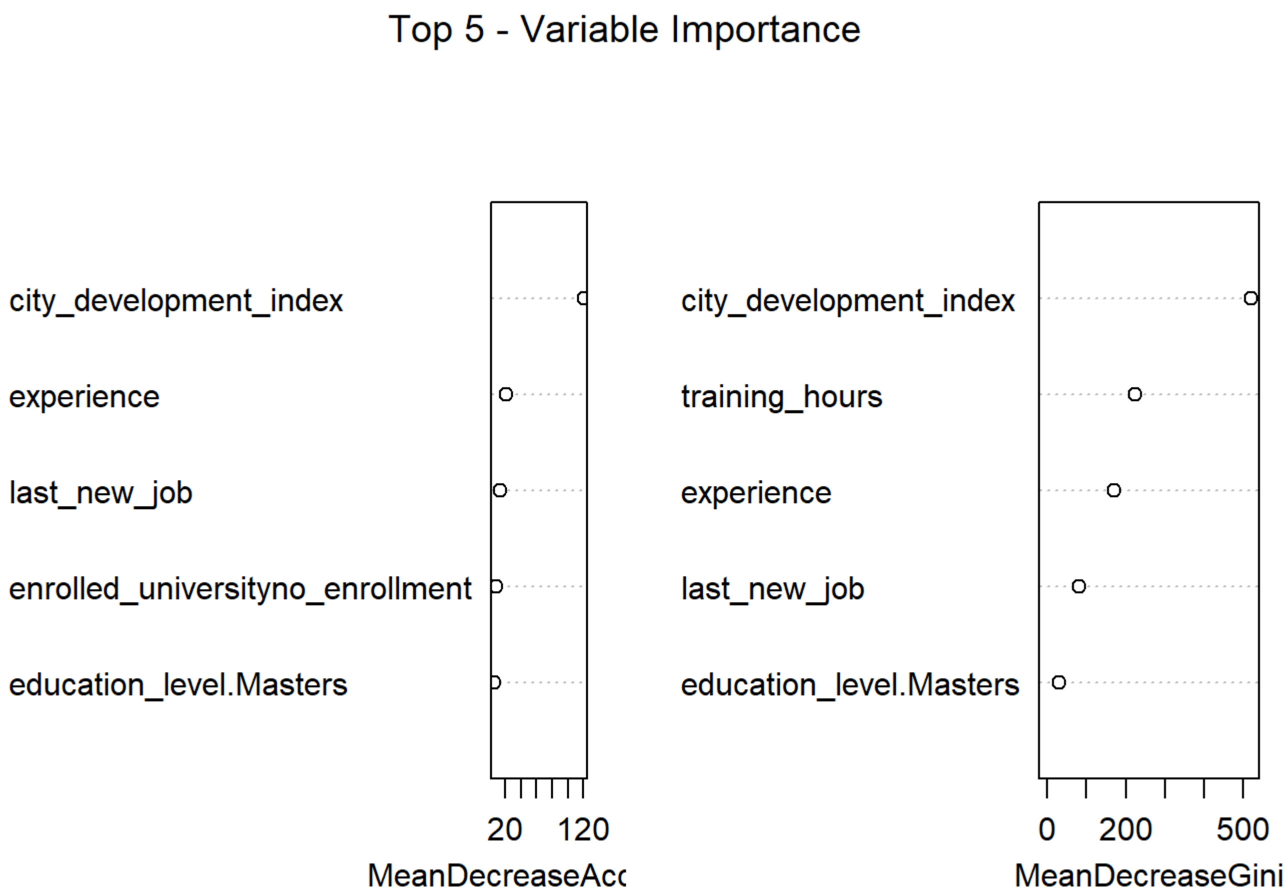
```
# fit the model with test data and evaluate
class_pred <- predict(object = model_rf, newdata = test_df)
confusionMatrix(data = class_pred,reference = test_df$target, positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1425  163
##          1   84  123
##
##                Accuracy : 0.8624
##                  95% CI : (0.8456, 0.878)
##     No Information Rate : 0.8407
##     P-Value [Acc > NIR] : 0.00581
##
##                   Kappa : 0.4216
##
##  Mcnemar's Test P-Value : 6.941e-07
##
##             Sensitivity : 0.43007
##             Specificity : 0.94433
##          Pos Pred Value : 0.59420
##          Neg Pred Value : 0.89736
##              Prevalence : 0.15933
##          Detection Rate : 0.06852
##    Detection Prevalence : 0.11532
##       Balanced Accuracy : 0.68720
##
##        'Positive' Class : 1
##
```

```
plot(model_rf)
```

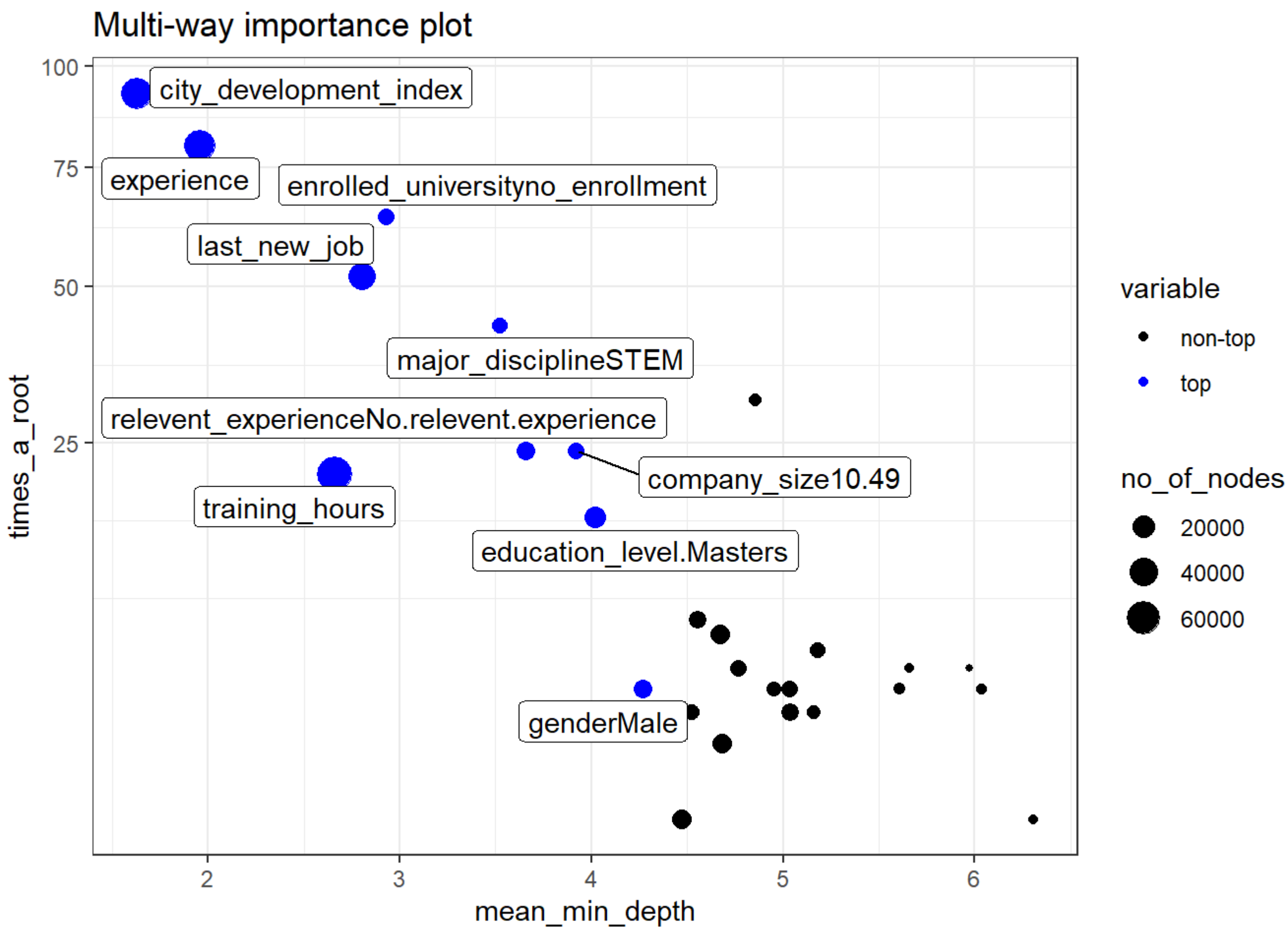



```
varImpPlot(model_rf,
           sort = T,
           n.var = 5,
           main = "Top 5 - Variable Importance")
```



Based on the variable importance plot of our random forest model, we can observe that city development index, experience, and last new job are the most important features in predicting if a candidate will accept the job offer.

```
importance_frame <- measure_importance(model_rf)
plot_multi_way_importance(importance_frame, size_measure = "no_of_nodes")
```

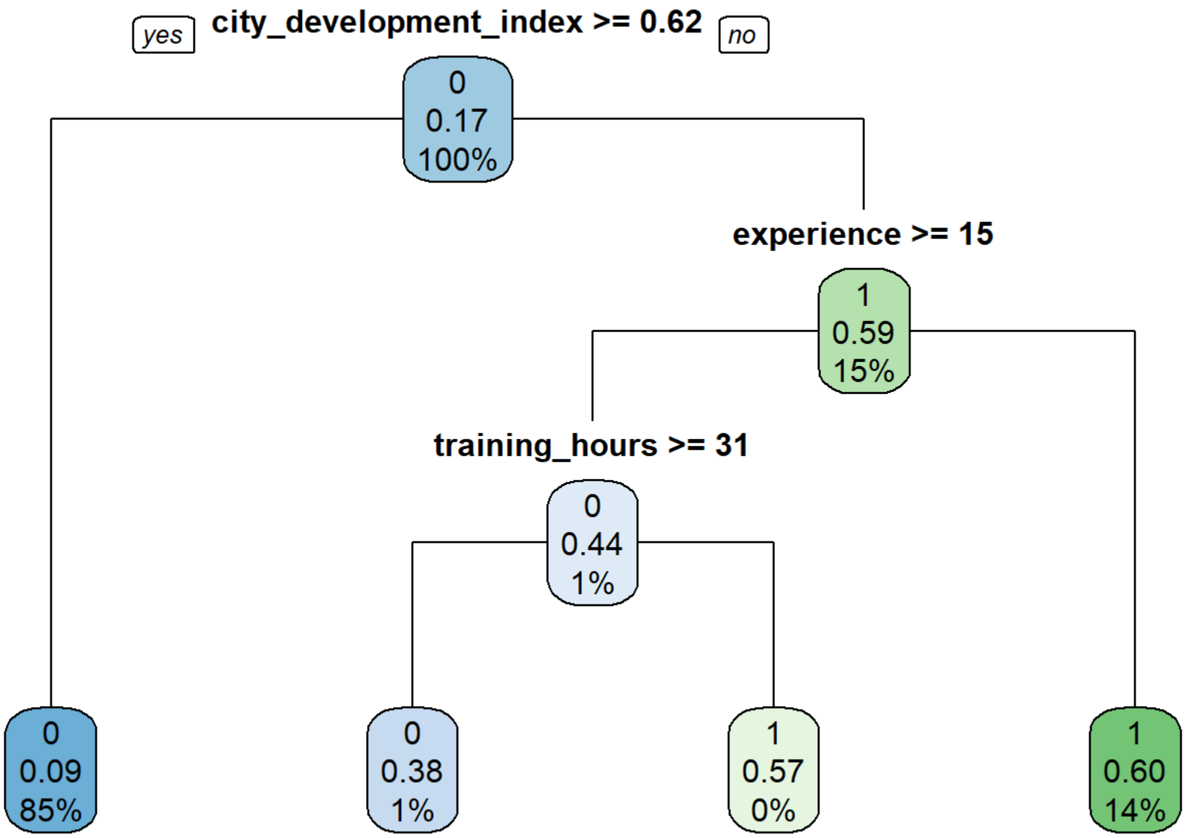


Decision Tree Using Rpart

```
model_rpart <- rpart(target ~., train_df,  control = rpart.control(cp = .0025))
```

```
par(xpd = TRUE)
plot(model_rpart, compress=TRUE)
text(model_rpart, use.n=TRUE)
```


```
rpart.plot(model_rpart, type = 1)
```



We can see from this overly simplified decision tree above that city development index, experience, and training hours are the top 3 variables that are important in predicting if a candidate will accept the job offer.

Gradient Boost Model

```
fit.btree <- gbm(target ~.,
data = train_df,
n.trees = 100,
distribution = 'multinomial',
cv.folds = 5,
interaction.depth = 2,
shrinkage = 0.001)
```

```
## Warning: Setting `distribution = "multinomial"` is ill-advised as it is
## currently broken. It exists only for backwards compatibility. Use at your own
## risk.
```

```
pred_gbm <- predict.gbm(fit.btree, test_df, n.trees = 100, type = 'response')
```

```
relative.influence(fit.btree)
```

```
## n.trees not given. Using 100 trees.
```

```
##                    city_development_index
##                              19212.894616
##                                genderMale
##                                  0.000000
##                               genderOther
##                                  0.000000
## relevent_experienceNo.relevent.experience
##                                  3.446264
##         enrolled_universityno_enrollment
##                                 36.587475
##      enrolled_universityPart.time.course
##                                  0.000000
##                    education_level.Masters
##                                  0.000000
##                        education_level.Phd
##                                  0.000000
##          major_disciplineBusiness.Degree
##                                  0.000000
##               major_disciplineHumanities
##                                  0.000000
##                 major_disciplineNo.Major
##                                  0.000000
##                    major_disciplineOther
##                                  0.000000
##                     major_disciplineSTEM
##                                  0.000000
##                                experience
##                                222.397381
##                       company_size10.49
##                                 13.175708
##                     company_size100.500
##                                  0.000000
##                   company_size1000.4999
##                                  0.000000
##                      company_size10000.
##                                  0.000000
##                       company_size50.99
##                                  0.000000
##                     company_size500.999
##                                  0.000000
##                   company_size5000.9999
##                                  2.712455
##              company_typeFunded.Startup
##                                  0.000000
##                         company_typeNGO
##                                  0.000000
##                       company_typeOther
##                                  0.000000
##               company_typePublic.Sector
##                                 18.285165
##                     company_typePvt.Ltd
##                                  0.000000
##                              last_new_job
```

```
##                              41.865644
##                         training_hours
##                              44.793497
```

The gradient boost model also identifies city development index as the top feature in predicting whether a candidate will accept or reject the job offer. Additional features that are important no university enrollment, experience, training hours, and last new job are most important.

```
gbm_labels = colnames(pred_gbm)[apply(pred_gbm, 1, which.max)]
```

```
confMat <- table(test_df$target,gbm_labels)
confMat
```

```
##    gbm_labels
##        0    1
##   0 1392  117
##   1  128  158
```

```
cm = confusionMatrix(test_df$target, as.factor(gbm_labels), positive = "1")
print(cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1392  117
##          1  128  158
##
##                Accuracy : 0.8635
##                  95% CI : (0.8468, 0.8791)
##     No Information Rate : 0.8468
##     P-Value [Acc > NIR] : 0.02528
##
##                   Kappa : 0.4824
##
##  Mcnemar's Test P-Value : 0.52290
##
##             Sensitivity : 0.57455
##             Specificity : 0.91579
##          Pos Pred Value : 0.55245
##          Neg Pred Value : 0.92247
##              Prevalence : 0.15320
##          Detection Rate : 0.08802
##    Detection Prevalence : 0.15933
##       Balanced Accuracy : 0.74517
##
##        'Positive' Class : 1
##
```

Logistic Regression

```
model_dt <- train(target ~., data = train_df,
               method = "glm",
               trControl = train_control,
               family=binomial(link=logit))
print(model_dt)
```

```
## Generalized Linear Model
##
## 7182 samples
##   28 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 5746, 5746, 5745, 5745, 5746
## Resampling results:
##
##   Accuracy   Kappa
##   0.8423828  0.2828831
```

```
varImp(model_dt)
```

```
## glm variable importance
##
##   only 20 most important variables shown (out of 28)
##
##                                           Overall
## city_development_index                    100.000
## experience                                 19.586
## company_size10000.                         11.671
## enrolled_universityno_enrollment           10.979
## company_size10.49                           9.534
## company_size1000.4999                       8.885
## enrolled_universityPart.time.course         8.882
## last_new_job                                7.865
## education_level.Phd                         6.746
## training_hours                              5.315
## company_size50.99                           5.070
## company_size100.500                         5.036
## company_typeOther                           4.236
## company_size5000.9999                       4.218
## relevent_experienceNo.relevent.experience   3.582
## genderOther                                 3.574
## company_typePublic.Sector                   3.304
## education_level.Masters                     2.510
## company_size500.999                         2.402
## major_disciplineBusiness.Degree             2.214
```

We also ran a logistic regression to identify most important features. Apart from city development index, experience, last new job and training hours, additional features identified as important predictors include company size (10,000, 10-49 and 1000-4999), candidates who enrolled in part time courses and those who did not enroll in university. Interestingly, Masters level and undergraduate level students have not been listed above and only the feature on PhD level candidates has been listed as an important feature.

```
logistic_labels <- predict(model_dt, test_df)
cm = confusionMatrix(test_df$target, as.factor(logistic_labels), positive = "1")
print(cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1442   67
##          1  204   82
##
##                Accuracy : 0.849
##                  95% CI : (0.8316, 0.8653)
##     No Information Rate : 0.917
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3007
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.55034
##             Specificity : 0.87606
##          Pos Pred Value : 0.28671
##          Neg Pred Value : 0.95560
##              Prevalence : 0.08301
##          Detection Rate : 0.04568
##    Detection Prevalence : 0.15933
##       Balanced Accuracy : 0.71320
##
##        'Positive' Class : 1
##
```

Regression Models to Predict City Development Index

For our regression model, we are only using experience, no university enrollments, last new job and STEM discipline variables to predict the city development index. This can be attributed to the variable importance results in our classification models that display these 4 features as the most important features across all classification models above.

```
keeps <- c("city_development_index", "experience","enrolled_universityno_enrollment","last_new_j
ob","major_disciplineSTEM")
new_df_th <- new_df[keeps]

# new_df_th <- scale(new_df_th)
# new_df_th <- as.data.frame(new_df_th)

split <- round(nrow(new_df_th) * 0.80)

# Create train and test data for regression models
train_df_th <- new_df_th[1:split, ]
test_df_th <- new_df_th[(split + 1):nrow(new_df_th), ]
```

Linear Regression

```
lmTemp = lm(city_development_index ~ ., data = train_df_th)
#Create a linear regression with a quadratic coefficient
summary(lmTemp)
```

```
##
## Call:
## lm(formula = city_development_index ~ ., data = train_df_th)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36419 -0.05343  0.03057  0.08120  0.16823
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      0.8059960  0.0053448 150.801  < 2e-16 ***
## experience                       0.0055170  0.0002204  25.027  < 2e-16 ***
## enrolled_universityno_enrollment  0.0138366  0.0036954   3.744 0.000182 ***
## last_new_job                     0.0024162  0.0008418   2.870 0.004111 **
## major_disciplineSTEM            -0.0486807  0.0041446 -11.745  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1085 on 7177 degrees of freedom
## Multiple R-squared:  0.1304, Adjusted R-squared:  0.1299
## F-statistic: 269.1 on 4 and 7177 DF,  p-value: < 2.2e-16
```

```
pred_train_lm <- predict(lmTemp, newdata = train_df_th)
mse_train.lm <- mean((pred_train_lm - train_df_th$city_development_index) ^ 2)
mse_train.lm
```

```
## [1] 0.01177169
```

```
pred_lm <- predict(lmTemp, newdata = test_df_th)
mse_test.lm <- mean((pred_lm - test_df_th$city_development_index) ^ 2)
mse_test.lm
```

```
## [1] 0.01170526
```

Ridge Regression

```
#set seed for reproducibality
set.seed(1)

# creating the train control function for cross validation creating 5 folds. The same 5 folds wi
ll be used for Ridge and Lasso regression
train_control <- trainControl(method = "cv",
                              number = 5)
model_ridge <- train(city_development_index ~., data = train_df_th,
               method = "ridge",
               trControl = train_control)

# printing model performance metrics
# along with other details
print(model_ridge)
```

```
## Ridge Regression
##
## 7182 samples
##    4 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 5746, 5747, 5745, 5745, 5745
## Resampling results across tuning parameters:
##
##   lambda  RMSE       Rsquared   MAE
##   0e+00   0.1085584  0.1302280  0.08782240
##   1e-04   0.1085584  0.1302281  0.08782232
##   1e-01   0.1085703  0.1300816  0.08774886
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was lambda = 1e-04.
```

```
varImp(model_ridge)
```

```
## loess r-squared variable importance
##
##                                  Overall
## experience                       100.000
## last_new_job                      13.906
## enrolled_universityno_enrollment   1.816
## major_disciplineSTEM               0.000
```

```
pred_ridge <- predict(model_ridge, newdata = test_df_th)

pred_train_ridge <- predict(model_ridge, newdata = train_df_th)
mse_train.ridge <- mean((pred_train_ridge - train_df_th$city_development_index) ^ 2)
print('Train')
```

```
## [1] "Train"
```

```
mse_train.ridge
```

```
## [1] 0.01177169
```

```
pred_ridge <- predict(model_ridge, newdata = test_df_th)
mse_test.ridge <- mean((pred_ridge - test_df_th$city_development_index) ^ 2)
print('Test')
```

```
## [1] "Test"
```

```
mse_test.ridge
```

```
## [1] 0.01170527
```

Lasso Regression

```
#set seed for reproducibality
set.seed(1)

model_lasso <- train(city_development_index ~., data = train_df_th,
               method = "lasso",
               trControl = train_control)
print(model_lasso)
```

```
## The lasso
##
## 7182 samples
##    4 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 5746, 5747, 5745, 5745, 5745
## Resampling results across tuning parameters:
##
##   fraction  RMSE       Rsquared   MAE
##   0.1       0.1144907  0.1106546  0.09550518
##   0.5       0.1100097  0.1195907  0.09028264
##   0.9       0.1086146  0.1298549  0.08796764
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was fraction = 0.9.
```

```
varImp(model_lasso)
```

```
## loess r-squared variable importance
##
##                                  Overall
## experience                       100.000
## last_new_job                      13.906
## enrolled_universityno_enrollment   1.816
## major_disciplineSTEM               0.000
```

```
pred_lasso <- predict(model_lasso, newdata = test_df_th)

pred_train_lasso <- predict(model_lasso, newdata = train_df_th)
mse_train.lasso <- mean((pred_train_lasso - train_df_th$city_development_index) ^ 2)
print('Train')
```

```
## [1] "Train"
```

```
mse_train.lasso
```

```
## [1] 0.01178403
```

```
pred_lasso <- predict(model_lasso, newdata = test_df_th)
mse_test.lasso <- mean((pred_lasso - test_df_th$city_development_index) ^ 2)
print('Test')
```

```
## [1] "Test"
```

```
mse_test.lasso
```

```
## [1] 0.01170773
```

To find out the meaningful information and build a better model, we applied cross validation and also used a varImp function to identify which variables have the most influence on the city development index.

Variable Importance:

The experience variable showed a score of 100, which means that this variable is extremely important or similar with our target “city development index”. Other three variables, last new job, enrolled university, major discipline STEM showed scores below 15, indicating that it is not as statistically significantly as experience in predicting the city development index.

MSE:

Both train and test MSE showed similar values for all linear, lasso, and ridge models, which means our model is very optimized with no evidence of overfitting or under fitting. We were also able to calculate very small values of MSEs. Here are training and test MSE values for all three regression models: * Linear: 0.01177169 / 0.01170526 * Lasso: 0.01178403 / 0.01170773 * Ridge: 0.01177169 / 0.01170527

Lambda & R-square:

The values of lambda and R-squares are very small (lambda is close to zero). Since the lambda value for both Lasso and Ridge regression is very small, we can conclude that the coefficients are not shrunk to a great extent and Since R square is very small, we can conclude that the coefficients are not shrunk to a great extent and all 4 features are important in predicting the city development index. However, we would have to re-engineer our features/conduct more research on identifying other features such as GDP, population, income levels, etc in order to improve the fit of our model (increase R square): * Linear: R^2 (0.1304) * Lasso: Lambda (0e+00), R^2(0.1299193) * Ridge: Lambda (0.1), R^2(0.1102777)

Conclusion

Chosen Model for Predicting Whether Candidates Will Stay or Leave the Company (Classification)

The criteria for us to make the choice of the model depends on multiple aspects of the result. In the evaluation process, we considered not only the accuracy, but also the sensitivity, specificity, and balanced accuracy. Finally, we picked the Gradient boosting model as it is the best model with highest accuracy. All the models we have run including decision tree, random forest, and Gradient boost have close accuracy (all around 86 percent). We next try to compare the sensitivity and specificity and find that the Gradient boosting model has better sensitivity as compared to the remaining 3 models. Our test dataset has 1795 observations, with a 90% accuracy of prediction of the target value 0, i.e. prediction of candidates not looking for any job change. The Gradient boosting model shows its advantage on sensitivity (57 percent) which means it has fewer false negative predictions. It has predicted more correct values of candidates looking for job change (158 over 275) as compared to other models and therefore, we have picked the Gradient boost model.

The variable importance and tree plots display that city development index is the most important feature across all models in predicting whether candidates will accept or reject the job, followed by years of experience. This implies that companies should also consider the city development index when predicting whether candidates will stay or leave the company. In our exploratory data analysis, we noticed that candidates belonging to more developed cities tend to reject job offers. Therefore, we find that it would be relevant to also use the most important variables to predict the city development index, which would help companies in guaging how much to invest in their employees. A higher development index and higher levels of work experience indicates that candidates will tend to reject the offer. This also helps companies to make the prediction for the cost of human force in the future.

Chosen Model for Predicting the City Development Index (Regression)

After we run three classification models, we find out that “city_development_index” has the highest impact on our target variable. Which means that a candidate with a high “city_development_index” value would be more likely to reject the offer.

We pick the best model on three values which are MSE/RMSE, MAE, and R^2. After fitting test data into Lasso, Ridge and Linear regression models, we pick linear regression as the best model. Linear regression gives us MSE with 0.01170526 which is the lowest and small MSE indicates a better model on unseen data. The residual standard error is 0.1085 and the R^2 is 0.1304 which is the highest. R^2 was based on correlation between actual and predicted value, as the value near 1 which indicates a better model. Also, MSE for both train and test are very close for Linear regression which means there is no overfitting or under fitting. Therefore, our best model for regression would be linear regression. The Ridge Regression model sets lambda to 0, which implies that it is actually an Ordinary Least Squares model, and the Lasso Regression model sets lambda to a value very close to 0, indicating that all features are important in predicting city development index. For this reason, we would recommend using a Linear Regression model, but also conducting research to identify additional features such as GDP, income of employees, population of the cities, etc to improve the prediction of the city development index.

We created a train dataset based on the target variable “city_development_index” to see which variable would impact “city_development_index” the most. We included the following four features (most important featurs from our classification models) in our new train and test data : “experience”, “enrolled_university_enrollment”, “last_new_job”, and “major_disciplineStem’’. By picking linear regression as our best model, we can easily see that”experience“,”enrolled_university_enrollment“, and”last_new_job" have positive relationships with the “city_development_index”. And as “enrolled_university_enrollment” increases, the target variable “city_development_index” increases more compared with the other two variables which all have a positive relationship with “city_development_index”.

Recommendations

Candidates who are more likely to accept the job offer include those who:

- Reside in low developed cities
- Have a discipline in STEM
- Do not have any degree from universities (candidates with no formal undergraduate or graduate degrees might be more willing to receive work experience)

Challenges Faced

- Imputation of values while cleaning data (examples include changing values with: greater than 20 years of experience as 21, less than 1 year of experience as 0)
- Regression variables used in the model might not be the only variables that are good predictors of the city development index. Unknown variables such as GDP, income and population levels might be better indicators

---

Future Work

- Additional research on variables that might be better predictors of city development index
- Checking if dimensionality reduction techniques are required to account for highly correlated variables