HR Analytics: Job Change for Data Scientists

Final Project

Amanda Niu, Runcong Wu, Soheil Bakhtiari, Peter Akioyamen

Group #8

Professor Camila de Souza

SS 4850G/9850: Advanced Data Analysis

A report submitted in partial fulfilment of the requirements for Statistical Sciences 4850G/9850

Table of Contents

1. Introduction

Data scientists are changing their jobs at a rapid pace; as a result, companies are wasting valuable resources and incurring extra costs (Jain, 2020). A company that is active in Big Data and Data Science is focused on hiring data scientists. The company offers an externally facing training course in data science for which many people sign-up. The course provides participants to learn extremely useful data science skills while also creating a pipeline to recruit talent into the company; at the end of the course, some candidates are given the opportunity to interview for a data science role and subsequently join the company if successful.

The company would like to assess which participants who complete the course are high potential candidates. That is, which people are candidates who would really like to work for the company after training or are actively looking for new employment. The ability to identify such candidates is of high importance to the company as it helps reduce the overall cost of training and the time it takes to onboard a hired data scientist. At the beginning of the course, participants provide specific information on demographics, education, and past experience to register. The company hopes to use this information to accomplish two tasks:

1. Develop a model that predicts whether a candidate is looking for a job change with a high degree of accuracy.
2. Better understand some of the factors that lead a person to leave their current job.

While the first desire is in line with the aforementioned statements, the second is derived from the need to understand what drives employee career decisions internally, so that human resources (HR) can improve as well.

With this in mind, the goal of this analysis is to analyze the data provided by the company and firstly construct a high-performing model which is able to classify candidates as looking for a job change or not looking for a job change. After this is done, we hope to interpret this model to provide some insight into the factors which drive employee career decisions more generally.

2. The Data

The data set used in this work is titled *HR Analytics: Job Change of Data Scientists* and was retrieved from Kaggle.com at the following link: Kaggle Data Set. Though the original task on Kaggle.com provides both a training and testing set individually, the testing set provided does not contain the labels for the target variable so we elect to use only the training data file in our analysis, referred to as the data from herein. The data has observations for 19158 candidates (rows) across 14 variables. The data has a total of 20733 missing values out of 268212 total values, equating to a missingness rate of approximately 7.73%. A description of the variables is as follows:

Table 1: List of variable descriptions in the data set.

Variable	Description
enrollee_id	Unique ID for candidate
city	City code
city_development_index	Development index of the city (scaled)
gender	Gender of candidate
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	Number 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

All data preprocessing and analyses were performed under R version 4.0.3. Various preprocessing steps are carried out to prepare the data to be analyzed. Observations containing missing values are dropped from the data set, resulting in a final data set with 8955 rows. The *enrollee_id* and *city* variables are dropped from the data set as *enrollee_id* is a candidate identifier and *city* has 123 unique values making it less valuable in analysis. Any candidate with more than 20 years of experience is given an experience value of 21 while candidates with less than 1 year of experience are given an experience value of 0; this allows for experience, measured in years, to be treated as a numerical variable in our analysis. Relevant variables were encoded as categorical for this work. The target variable is binary, with 0 indicating the candidate is not looking for a job change and 1 indicating that they are.

3. Methods

In order to better understand our data, we create several univariate and bivariate visualizations in an exploratory data analysis. To begin we consider the categorical predictors in the data set. For each relevant variable, we create a bar graph to see the count of the observations across the various levels of the categorical predictor. We also use histograms to assess the distribution of the data for all continuous variables. A log-transform is applied to each continuous variable to evaluate if any beneficial effect on the data distribution is achieved. Boxplots are created to view the distribution of continuous variables with respect to the target variable in the data set; this would highlight any large differences in the distribution of data based on the response.

Seven distinct models are used for classification including Logistic Regression, linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), a Classification Tree, Bagging, Random Forest, and Boosting. While LDA is used where a linear decision boundary between classes is required, QDA is used to find a non-linear boundary. Generally speaking, these classifiers should work better if the response classes are separable and the distribution of each class is normal. LDA assumes that the classes are normally distributed with class-specific mean (μ_i) and common covariance between classes ($\sum$). Conversely, QDA assumes class-specific mean (μ_i) and class-specific covariance ($\sum_i$). We consider both models since the covariance of the classes is not necessarily known. Logistic Regression makes no inherent assumptions about the distribution of the classes and is based on maximum likelihood estimation. The Hosmer-Lemeshow test is used to assess the goodness-of-fit of the logistic regression model. A backward selection is conducted based on AIC for variable selection and the likelihood ratio test is used for model comparison between the full model and the resulting model. A Classification Tree is used due to its high level of interpretability - though it is understood that Classification Trees suffer from high variance, which may limit the model’s ability to generalize on new data. Ensemble methods are also considered in this work, namely Random Forest, Bagging, and Boosting.

Logistic Regression: Firstly, we fit a model using all 11 features. Secondly, we perform the backwards feature selection algorithm with AIC as our criterion. Then, we select the best model using the Likelihood Ratio Test. The Hosmer-Lemeshow Test is used to determine the model’s goodness-of-fit. The model assumptions are evaluated by the deviance residual plot and half-normal plot.

Linear Discriminant Analysis (LDA): LDA is used to classify candidates' status (looking for a job change or not) by using linear combinations of the predictors. We used the features found by the best Logistic Regression model. Since the LDA model requires the predictors to follow approximately normal distributions, *training_hours* was log-transformed into an approximately normal variable.

Quadratic Discriminant Analysis (QDA): QDA creates non-linear decision boundaries to classify candidates' status (looking for a job change or not). We used the same features in the LDA model. *training_hours* was log-transformed to better fit the model assumption of using approximately normal predictors.

Tree-based Method: The random seed was set to 108 to ensure reproducible results for all tree models. All tree-based models are trained using the original predictors without any transformation since they are not sensitive to the range of variables.

Classification Tree: All 11 features are given to the tree, Gini index is used to find the best split.

Bagging: Bagging is a bootstrap-based approach that aims to reduce the variance in decision trees. It corrects the problem of overfitting by re-sampling the training sets with replacement. We decided to build 500 trees. Each split considers all 11 features.

Random Forest: Random forest is developed from the bagging tree and it further reduces the correlation of each tree by splitting based on a random sample of features. During this process, we can examine the importance of each feature in prediction power and remove the insignificant attributes to avoid overfitting. The model trained 500 trees with 3 variables randomly sampled as candidates at each split.

Boosting: Boosting can gradually reduce the error by fitting many weak trees sequentially. We decided to use a binary classification task because the response variable follows the Bernoulli distribution. The model is trained on 500 trees with a learning rate (shrinkage) of 0.3. We convert the response to a character type because the "gbm" R package needs a character type response instead of a factor type response.

A training set is created to fit all models considered in this work while a testing set is used to assess out-of-sample error and the ability for the models to generalize on new data. A data set specifically designated for testing helps reduce the bias in evaluating out-of-sample performance across the models. To assess model performance, we compute various metrics including sensitivity, specificity, and accuracy. We also construct the receiver operating characteristic (ROC) curve to evaluate the performance of the models as their decision threshold is varied from 0 to 1, and the area under the curve (AUC) as the metric to optimize. The model with the highest AUC is selected as the best model in this work. The method allows us to determine the optimal threshold value for the model chosen. A confusion matrix is created from the predictions the model makes on the testing data to provide a better understanding of the misclassifications being made by the classifier and the frequency at which they are made, given the optimal decision threshold. Finally, since our best model is determined to be boosting trees, we are able to evaluate feature importance. We use mean decrease in Gini index to determine relative variable importance; as Gini index is a measure of node impurity, important variables are those which increase node purity the most on average across the various trees in a model. Partial dependence plots are also created which show the marginal effect different features have on the predicted outcome of the boosting model.

4. Results

4.1 Exploratory Data Analysis

We begin by assessing the frequency of the different levels for each categorical predictor used in this work as well as the frequency of the target variables. A severe class imbalance with respect to the response variable is identified as a result of this which helps inform our model training and testing process. Since the target variable exhibits a class imbalance, rather than splitting the data into a traditional training and testing set, the train-test split is stratified based on the target variable to preserve the proportions of the classes in both splits. The training set will contain 80% of observations with a target value of 0 as well as 80% of observations with a target value of 1. Stratifying the splits preserves the distribution of the target variable across both data, ensuring that the training and testing sets are representative of the composition we expect in the population. The majority class is 0, which indicates a candidate is not looking for a job change.

The distribution of the city development index is negatively skewed with a large peak around the value of 1.0. Conversely, the distribution of candidate experience is positively skewed. The distribution shows a peak at 5 years of experience and a larger peak around 21 years - this second peak is likely a result of the data preprocessing that was done. Many candidates indicated that they had more than 20 years of experience without providing a specific value. It is likely that candidates at this level have similar profiles and skills due to their experience, so all of these observations were encoded as having 21 years of experience, allowing the predictor to be treated as numerical. Applying the natural logarithm to both of these variables has minimal effect on their distributions. The distribution of the number of hours candidates spent training is also visualized. The distribution shows a highly positive skew with a large peak near 0. The natural logarithm transformation has a significant impact on this predictor, making the resultant distribution approximately normal. Consequently, we will use the natural logarithm of training hours as a predictor in the logistic regression, LDA, and QDA models. The skewness of the city development index is more prominent in the collection of candidates who are not looking for a job change in comparison to those who are - it's possible that this may act as an important feature and predictor commonly used to differentiate candidates across models.

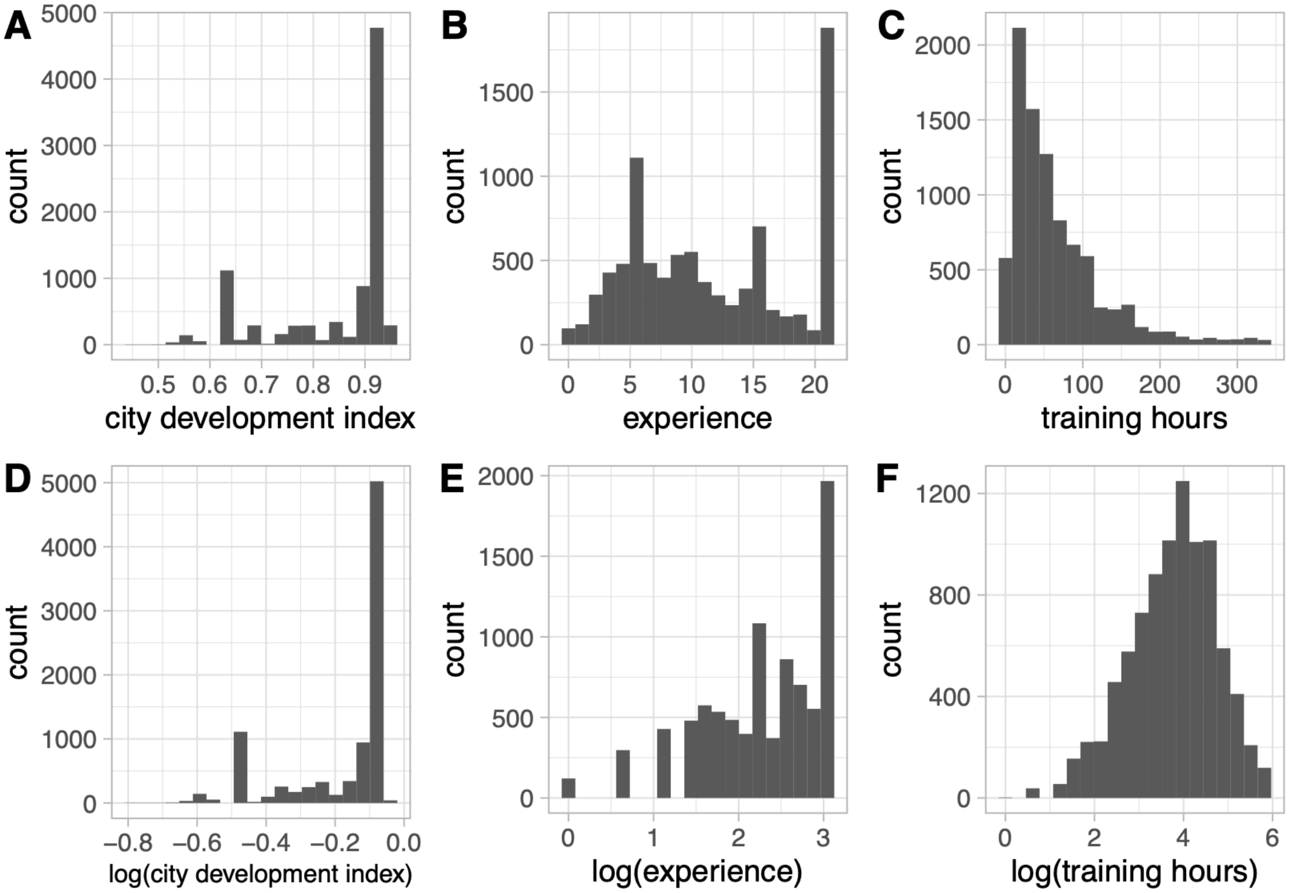


Figure 1: Distribution of continuous variables before and after the application of natural logarithm transform.

Then, we assessed the relationship between each continuous predictor and the target using boxplots shown in Figure 2. There are noticeable differences in the distribution of the *city_development_index* and *experience* based on the employees' decision (looking for a job change or not). It indicates that these 2 predictors may have a significant impact on the probability of employee leaving. Lower *city_development_index* results in a higher chance of employees leaving; likewise, less-experienced candidates have a greater chance of leaving their current job.

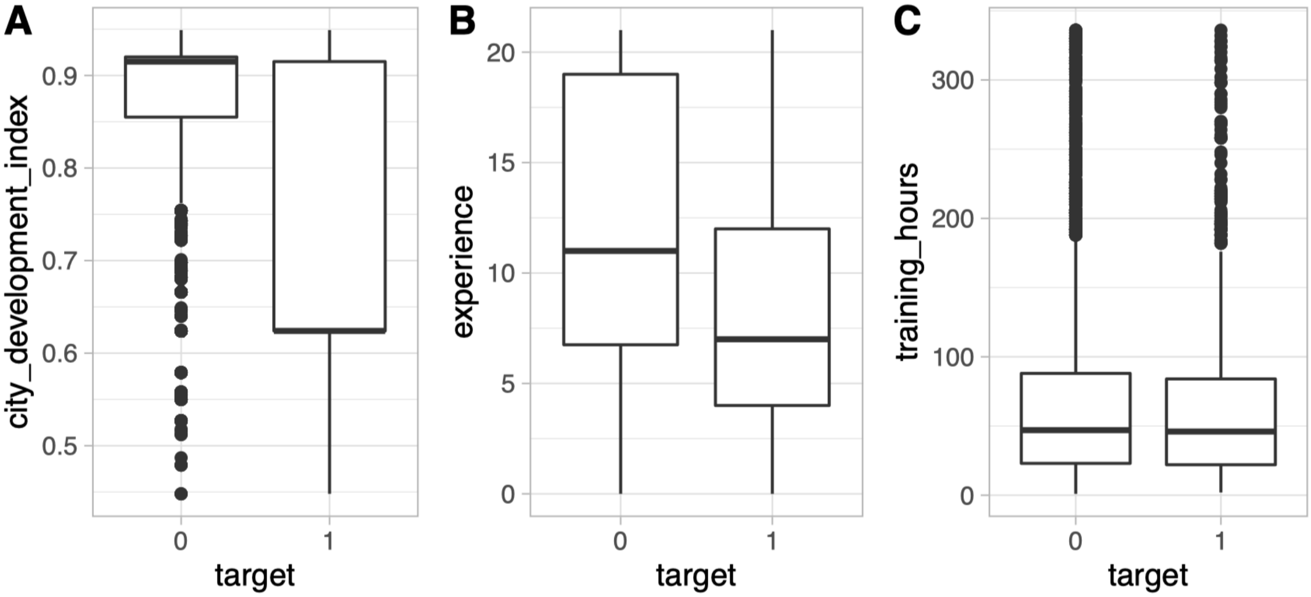


Figure 2: Side-by-side boxplots describe the relationship between each continuous predictor and the target.

4.2 Main Data Analysis

Logistic Regression Results: Our final model contains 7 predictors, including *city_development_index*, *relevant_experience*, *enrolled_university*, *experience*, *company_size*, *company_type*, and *training_hours*. Here are the steps on how we derive this model. After fitting the Logistic Regression model to all 11 features, we run the Hosmer-Lemeshow (HL) Test and find evidence that our model does not fit the data well with a p-value of $4.062 * 10^{-5}$. We consider 11 features to be too many for any parametric model and so we reduce the dimensions by the AIC stepwise algorithm and obtain a smaller model with 7 features. Then, we compare the smaller model with the larger model using the Likelihood Ratio Test. Since the p-value is 0.2066, which is larger than 0.05, we conclude that the smaller model selected by AIC is adequate. Next, we perform diagnostic tests to our best model: the deviance residual plot and half-normal plot. The deviance residual plot indicates an increasing pattern of deviance residuals as the linear predictor increases. The half-normal plot shows a few potential outliers. We could remove them from our training dataset but after running this scenario, we find that the model performs worse and does not add any value. The AUC for the best model is 0.7363 with high specificity and accuracy.

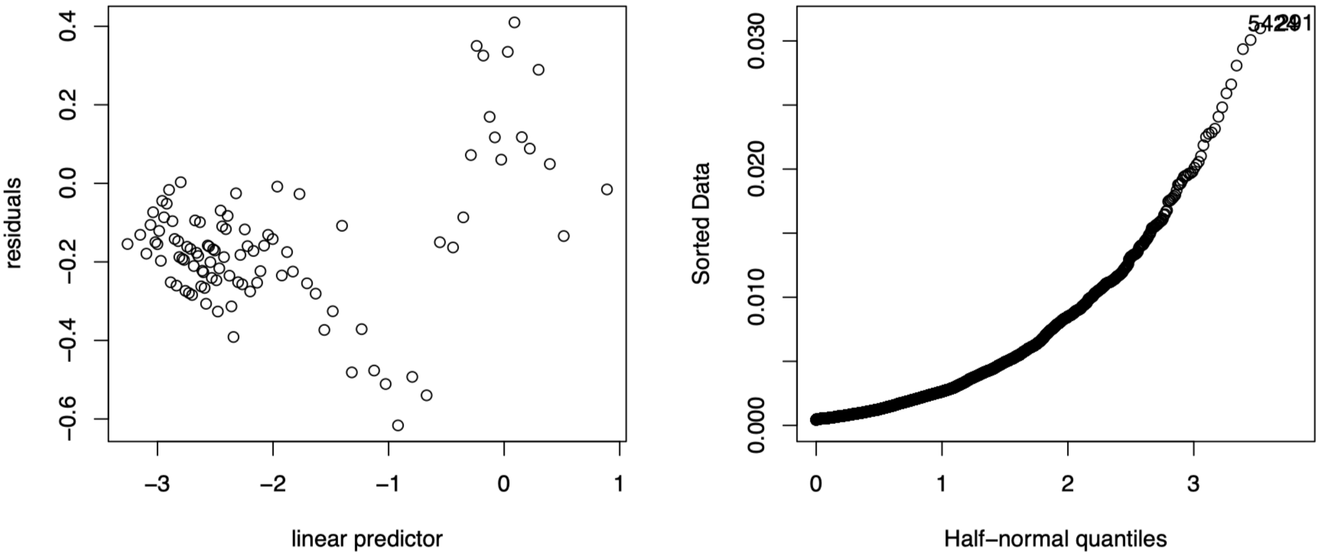


Figure 3: Model diagnostic plots. (a) Left: deviance residual plot (b) Right: half-normal plot

Linear Discriminant Analysis Results: *city_development_index*, *relevant_experience*, *enrolled_university*, *experience*, *company_size*, *company_type*, and *log(training_hours)* were used to fit the LDA model. The results come back with an AUC of 0.7363 which is competitive with our LR model and suggests that the data might have a linear decision boundary.

Quadratic Discriminant Analysis Results: *city_development_index*, *relevant_experience*, *enrolled_university*, *experience*, *company_size*, *company_type*, and *log(training_hours)* were used to fit the QDA model. The test AUC is 0.7115.

Classification Tree Results: The unpruned tree has only 1 split based on the predictor *city_development_index* with 2 terminal nodes, which indicates that *city_development_index* is the most important feature. There is no need to prune the tree. The classification tree is generated with a test AUC of 0.7105.

Bagging Results: The out-of-bag error rate is 14.7%. The classification error rate for the positive class is 61.6%, while only 5.39% of the negative classes are misclassified. The test AUC for the Bagging tree model is 0.7411.

Random Forest Results: The out-of-bag error rate is 14.36%. The classification error rate for the positive class is 60.5%, while only 5.20% of the negative classes are misclassified. The test AUC for the Random Forest model is 0.7431.

Boosting Results: The Boosting model performs the best compared to all other models with a test AUC of 0.7479.

Table 2: Evaluation criteria values for seven different models

Models	Threshold	Sensitivity	Specificity	Accuracy	AUC
Logistic Regression	0.3509	0.5203	0.9150	0.8497	0.7363
LDA	0.4369	0.5236	0.9137	0.8492	0.7363
QDA	0.2259	0.5743	0.8159	0.7760	0.7115
Classification Tree	0.3411	0.4966	0.9244	0.8536	0.7105
Bagging	0.3270	0.5000	0.9023	0.8358	0.7411
Random Forest	0.2670	0.5203	0.9050	0.8413	0.7431
Boosting	0.2515	0.5135	0.9190	0.8520	0.7479

All of the models above achieve an AUC of over 70%. All the models, in general, tend to provide great specificity and accuracy but are relatively weak in terms of sensitivity. Boosting is the best model among those in terms of AUC, which reaches 74.79%.

According to the results of our 7 models, Boosting is selected as our final model. The Boosting model includes all 11 independent variables. The relative influence chart shown in Figure 4 indicates that *city_development_index* is the most important variable in terms of predictive power. It accounts for over 60% of the reduction to the loss function given this set of features. Other important variables include *company_size*, *training_hours*, and *experience*.

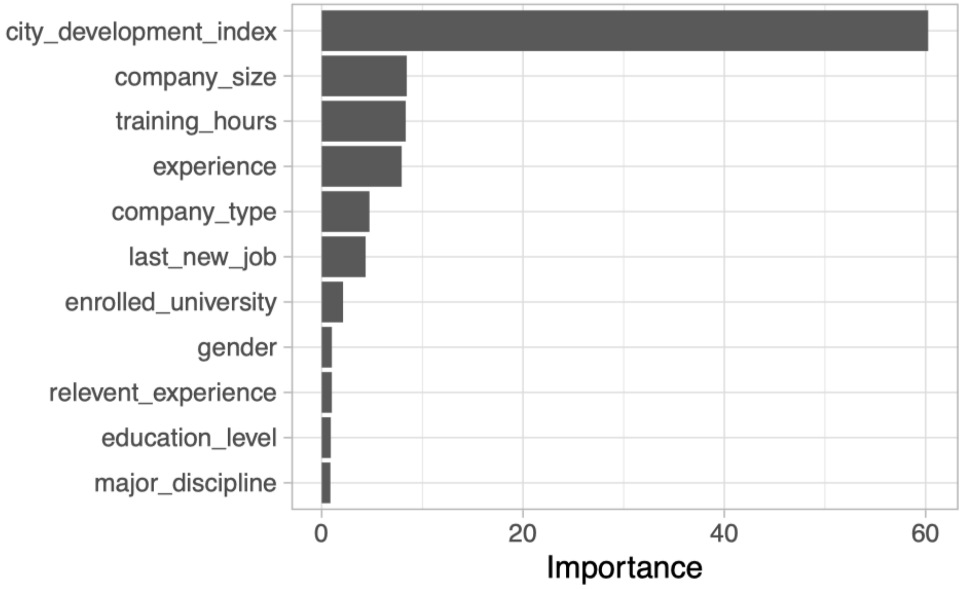


Figure 4: Relative Importance of the Boosting Model

The probability of success is plotted against the top 4 most important features, as shown in Figure 5. Current employees are more likely to have the intention to change the job when the city development index is low. It is interesting to note that it matches exactly what we found in our exploratory data analysis where the city development index has a significant impact on the probability of employees leaving.

The pattern between the probability of success and company size looks relatively random, with companies with more than 10000 employees having the highest chance of employees leaving. The pattern between the probability of success and training hours also looks random. Employee experience, on the other hand, has a negative relationship with the probability of success, indicating that the less experienced candidates are more likely to seek a job change. If we convert the graphs in Figure 5 to the same scale, as shown in Figure 6, it is clear that *city_development_index* has a more significant impact than the other three predictors.

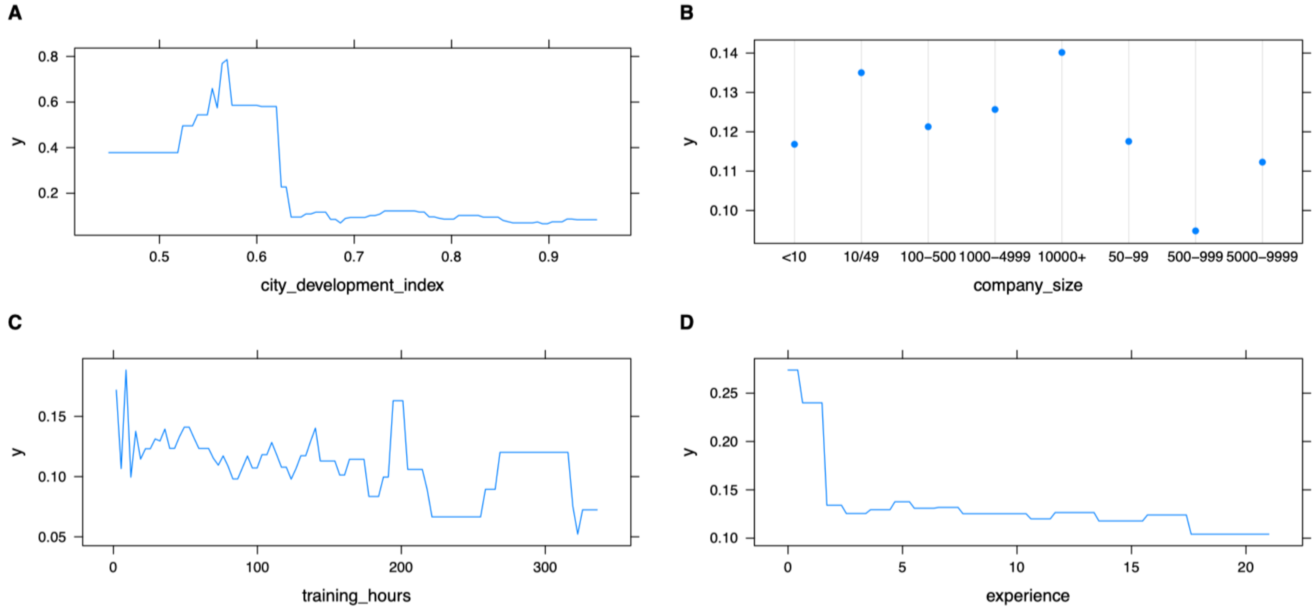


Figure 5: Probability of Success vs Top 4 Most Important Features (y-axis: different probability scale)

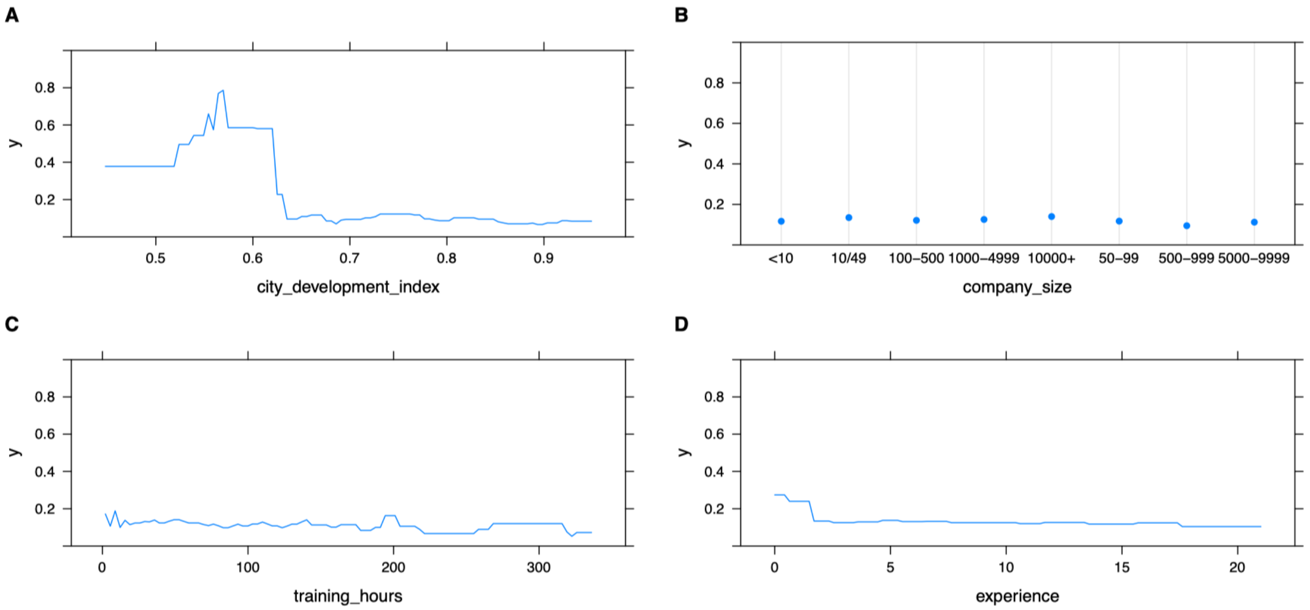


Figure 6: Probability of Success vs Top 4 Most Important Features (y-axis: same probability scale)

5. Conclusion & Discussion

We develop a statistical model to predict whether a candidate is looking for a job change by giving the candidate's demographics and experience information as well as companies' information. Building a model that has the ability to identify candidates with a higher chance of leaving can be beneficial to resource management. Using a data-driven approach is much preferred and faster than assigning candidates manually in the companies. Besides, it helps HR departments to understand the factors that lead a person to leave their current job.

There are 7 statistical models used in our analysis, including 4 traditional machine learning models (Logistic Regression, LDA, QDA, Classification Tree) and 3 black-boxed methods (Bagging, Random Forest, Boosting). Since our goal is to find a model that balances the model prediction and inference ability, it should be highly accurate and highly interpretable. According to the results of our 7 models, we decide to sacrifice some interpretation ability for better prediction results; thus, Boosting is selected as our final model. Although we cannot interpret the coefficients in terms of odds or odds ratio, we can identify significant features using feature importance ranking plot (Figure 4) and probability of success plot (Figure 5 & 6) for Boosting model.

The Boosting model also has high accuracy (85.20%) and high specificity (91.90%), while the sensitivity is not ideal (51.35%). High specificity indicates that the model is able to capture most candidates that are willing to stay in their current organizations. Low sensitivity indicates that among all the candidates who are looking for a job change, only half of them are captured by the model. The trade-off between sensitivity and specificity depends on the threshold. With the current threshold being 0.2515, we notice there is a high specificity and a relatively low sensitivity. Since our objective is to classify individuals with a high probability of leaving, we could consider improving the sensitivity by decreasing the threshold.

One limitation of this analysis could be caused by imputation. By deleting data with missing observations, a significant portion of the information that would have been relevant in fitting our model was discarded. Secondly, we may have actually biased the data depending on the data missingness mechanism. It's possible that the data missingness is systematic (ie. if the response is a 1, the probability of data missing is higher than if the response were 0), so by deleting these observations, we introduce imbalances and bias that should have been corrected for otherwise.

6. Student Contributions

All students contributed equally to this work. The data set was sourced and the scope of the analysis was developed by all team members - this includes: data preprocessing steps, techniques for EDA, models to consider for analysis, and model evaluation. Analysis was implemented in R by Runcong. Comments and code revisions were done by Amanda, Soehil, and Peter. The report was structured by Peter and contributed to by all members equally.

7. References

HR Analytics: Job Change of Data Scientists. (2020, December 7). [Dataset]. https://www.kaggle.com/arashnic/hr-analytics-job-change-of-data-scientists

Jain, S. (2020, September 14). *5 Key Reasons Why Data Scientists Are Quitting their Jobs*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2019/12/5-key-reasons-data-scientists-quit-jobs/

8. Appendix for R Code

Table of Contents

1. Preprocessing

```
# load the csv data
library(tidyverse)
HRdata <- read_csv("aug_train.csv")

# Check if there is any missing value (number of missing values = 20733)
sum(is.na(HRdata))
```

```
## [1] 20733
```

```
# Check nrows and ncols
dim(HRdata)
```

```
## [1] 19158    14
```

1.1 Handling Missing Values

```
# Function na.omit removes observation with missing values
HRdata <- na.omit(HRdata)
dim(HRdata)
```

```
## [1] 8955   14
```

1.2 Drop enrollee_id and city

```
HRdata <- subset(HRdata, select = -c(enrollee_id, city))
dim(HRdata)
```

```
## [1] 8955   12
```

1.3 Change predictor: experience to numeric

```
HRdata <- HRdata %>%
  mutate(experience = case_when(experience==">20" ~ "21",
                                experience=="<1" ~ "0",
                                TRUE ~ as.character(experience)))
HRdata$experience <- as.numeric(HRdata$experience)
```

1.4 Define Baseline for Categorical Variables -> Factor Type

```
HRdata$gender <- factor(HRdata$gender)
HRdata$relevent_experience <- factor(HRdata$relevent_experience)
HRdata$enrolled_university <- factor(HRdata$enrolled_university)
HRdata$education_level <- factor(HRdata$education_level)
HRdata$major_discipline <- factor(HRdata$major_discipline)
HRdata$company_size <- factor(HRdata$company_size)
HRdata$company_type <- factor(HRdata$company_type)
HRdata$last_new_job <- factor(HRdata$last_new_job)

#HRdata$target <- factor(HRdata$target, levels=c(0, 1), labels=c("No","Yes"))
HRdata$target <- factor(HRdata$target, levels=c(0, 1), labels=c("0","1"))

# Check the data type to see the changes
glimpse(HRdata)
```

```
## Rows: 8,955
## Columns: 12
## $ city_development_index <dbl> 0.776, 0.767, 0.762, 0.920, 0.920, 0.913, 0.926~
## $ gender                 <fct> Male, Male, Male, Male, Male, Male, Male, Male,~
## $ relevent_experience    <fct> No relevent experience, Has relevent experience~
## $ enrolled_university    <fct> no_enrollment, no_enrollment, no_enrollment, no~
## $ education_level        <fct> Graduate, Masters, Graduate, Graduate, Graduate~
## $ major_discipline       <fct> STEM, STEM, STEM, STEM, STEM, STEM, STEM, STEM,~
## $ experience             <dbl> 15, 21, 13, 7, 5, 21, 16, 11, 11, 0, 18, 21, 19~
## $ company_size           <fct> 50-99, 50-99, <10, 50-99, 5000-9999, 1000-4999,~
## $ company_type           <fct> Pvt Ltd, Funded Startup, Pvt Ltd, Pvt Ltd, Pvt ~
## $ last_new_job           <fct> >4, 4, >4, 1, 1, 3, >4, 1, 2, 1, 2, 3, >4, >4, ~
## $ training_hours         <dbl> 47, 8, 18, 46, 108, 23, 18, 68, 50, 65, 68, 40,~
## $ target                 <fct> 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,~
```

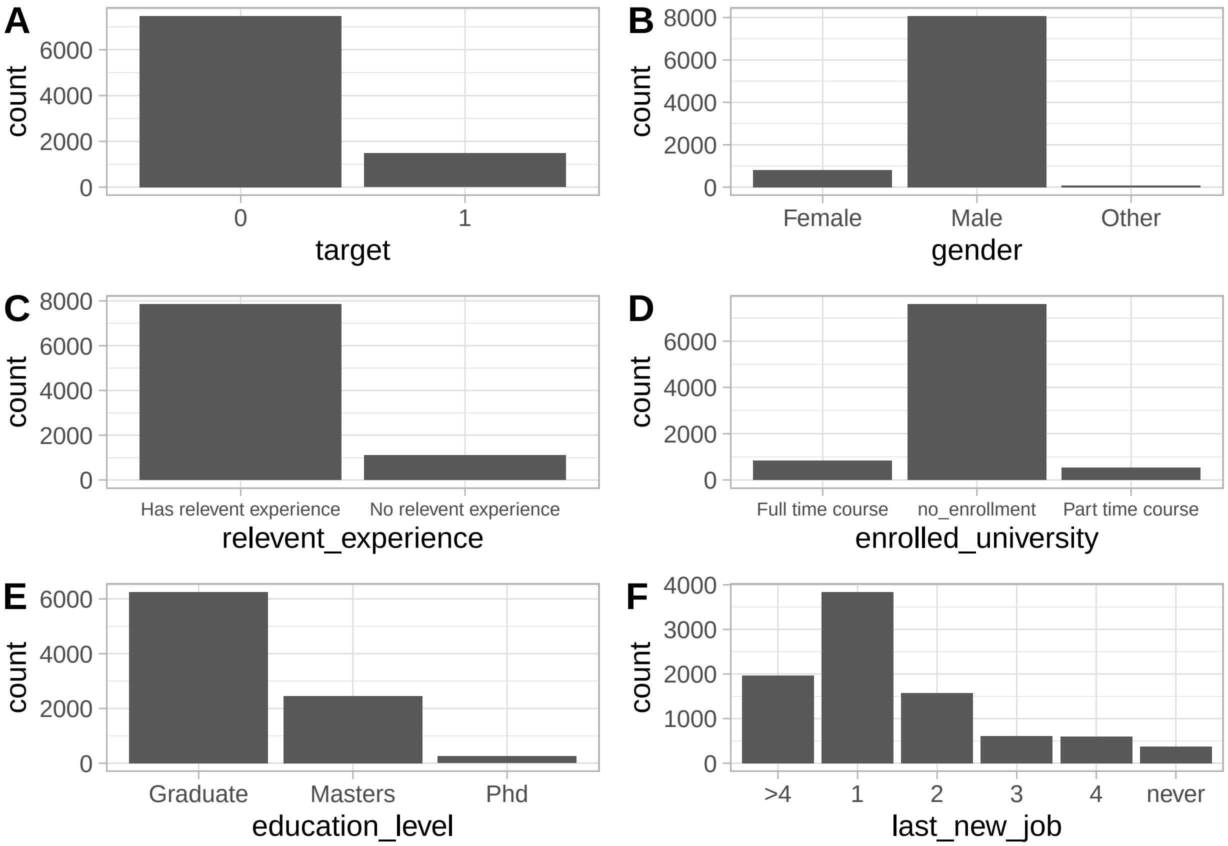
2. Exploratory Data Analysis

```r
# Generate summary statistics
summary(HRdata)
```

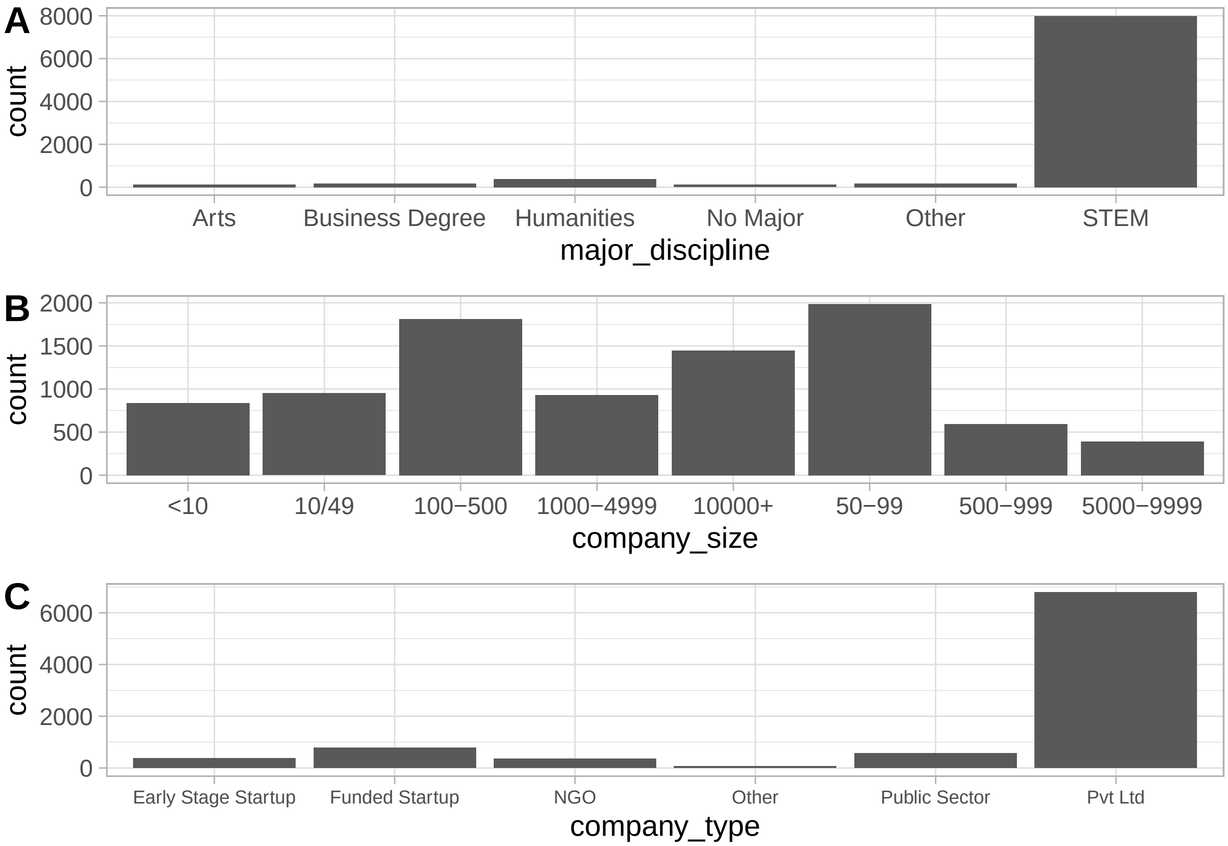
```
##  city_development_index    gender                  relevent_experience
##  Min.   :0.4480         Female: 804   Has relevent experience:7851
##  1st Qu.:0.7940         Male  :8073   No relevent experience :1104
##  Median :0.9100         Other :  78
##  Mean   :0.8446
##  3rd Qu.:0.9200
##  Max.   :0.9490
##
##        enrolled_university education_level        major_discipline
##  Full time course: 832     Graduate:6252   Arts           : 129
##  no_enrollment   :7594     Masters :2449   Business Degree: 170
##  Part time course: 529     Phd     : 254   Humanities     : 378
##                                            No Major       : 112
##                                            Other          : 177
##                                            STEM           :7989
##
##    experience       company_size              company_type  last_new_job
##  Min.   : 0.00   50-99    :1986   Early Stage Startup: 385   >4   :1965
##  1st Qu.: 6.00   100-500  :1814   Funded Startup     : 784   1    :3838
##  Median :10.00   10000+   :1449   NGO                : 356   2    :1570
##  Mean   :11.64   10/49    : 951   Other              :  72   3    : 610
##  3rd Qu.:18.00   1000-4999: 930   Public Sector      : 564   4    : 599
##  Max.   :21.00   <10      : 840   Pvt Ltd            :6794   never: 373
##                  (Other)  : 985
##  training_hours   target
##  Min.   :  1.00   0:7472
##  1st Qu.: 23.00   1:1483
##  Median : 47.00
##  Mean   : 65.07
##  3rd Qu.: 88.00
##  Max.   :336.00
##
```

2.1 Categorical Variable Distributions

```r
# target
target <- HRdata %>%
  ggplot(aes(x = target)) + geom_bar() + theme_light()

# gender
gender <- HRdata %>%
  ggplot(aes(x = gender)) + geom_bar() + theme_light()

# relevent_experience
relevent_experience <- HRdata %>%
  ggplot(aes(x = relevent_experience)) + geom_bar() + theme_light() +
  theme(axis.text.x = element_text(size = 7))

# enrolled_university
enrolled_university <- HRdata %>%
  ggplot(aes(x = enrolled_university)) + geom_bar() + theme_light() +
  theme(axis.text.x = element_text(size = 7))

# education_level
education_level <- HRdata %>%
  ggplot(aes(x = education_level)) + geom_bar() + theme_light()

# major_discipline
major_discipline <- HRdata %>%
  ggplot(aes(x = major_discipline)) + geom_bar() + theme_light()

# company_size
company_size <- HRdata %>%
  ggplot(aes(x = company_size)) + geom_bar() + theme_light()

# company_type
company_type <- HRdata %>%
  ggplot(aes(x = company_type)) + geom_bar() + theme_light() +
  theme(axis.text.x = element_text(size = 7))

# last_new_job
last_new_job <- HRdata %>%
  ggplot(aes(x = last_new_job)) + geom_bar() + theme_light()


# library allows clear labels and save plot as pdf
suppressMessages(library(cowplot))
(p_cat_1 <- plot_grid(target, gender, relevent_experience, enrolled_university,
                      education_level,last_new_job, labels = "AUTO", ncol=2,
                      rel_widths=20, rel_heights=50))
```



```
(p_cat_2 <- plot_grid(major_discipline, company_size,
                     company_type, labels = "AUTO", ncol=1,
                     rel_widths=20, rel_heights=50))
```



Categorical Variable Description:

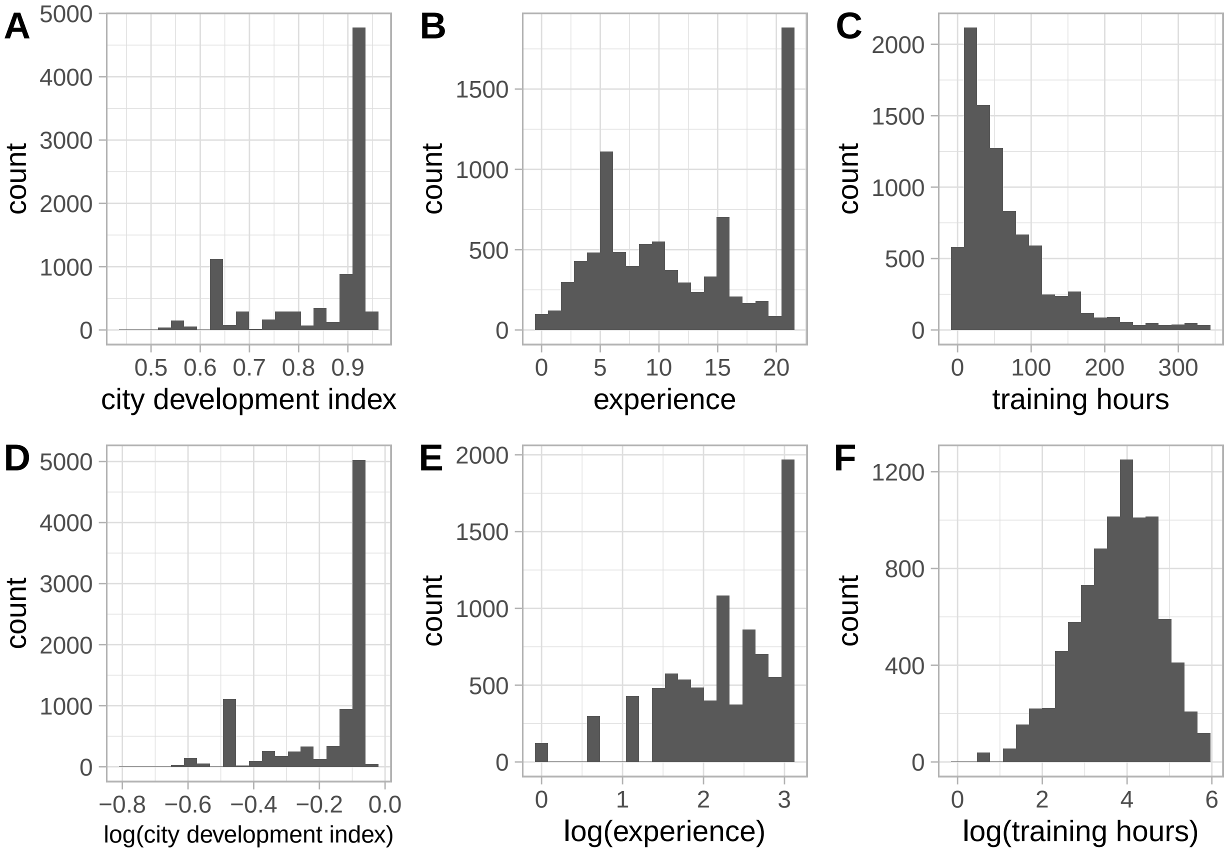
- target: 0 – Not looking for job change, 1 – Looking for a job change
- gender: Gender of candidate
- relevent_experience: Relevant experience of candidate
- enrolled_university: Type of University course enrolled if any
- education_level: Education level of candidate
- lastnewjob: Difference in years between previous job and current job
- major_discipline :Education major discipline of candidate
- company_size: No of employees in current employer's company
- company_type : Type of current employer

2.2 Continuous Variable Distributions

```
# city_development_index
city_development_index <- HRdata %>%
  ggplot(aes(x = city_development_index)) + geom_histogram(bins=20) +
  labs(x = "city development index") + theme_light()

# experience
experience <- HRdata %>%
  ggplot(aes(x = experience)) + geom_histogram(bins=20) + labs(x = "experience") +
  theme_light()

# training_hours
```

```r
training_hours <- HRdata %>%
  ggplot(aes(x = training_hours)) + geom_histogram(bins=20) + labs(x = "training hours") +
  theme_light()

# Distributions are skewed -> Taking the log
# city_development_index
log_city_development_index <- HRdata %>%
  ggplot(aes(x = log(city_development_index))) + geom_histogram(bins=20) +
  labs(x = "log(city development index)") + theme_light() +
  theme(axis.title.x = element_text(size = 9))

# experience
log_experience <- HRdata %>%
  ggplot(aes(x = log(experience))) + geom_histogram(bins=20) +
  labs(x = "log(experience)") + theme_light()

# training_hours
log_training_hours <- HRdata %>%
  ggplot(aes(x = log(training_hours))) + geom_histogram(bins=20) +
  labs(x = "log(training hours)") + theme_light()

(p_con_1 <- plot_grid(city_development_index, experience, training_hours,
                      log_city_development_index, log_experience, log_training_hours,
                      labels = "AUTO", ncol=3))
```

```
## Warning: Removed 97 rows containing non-finite values (stat_bin).
```



Continuous Variable Description:

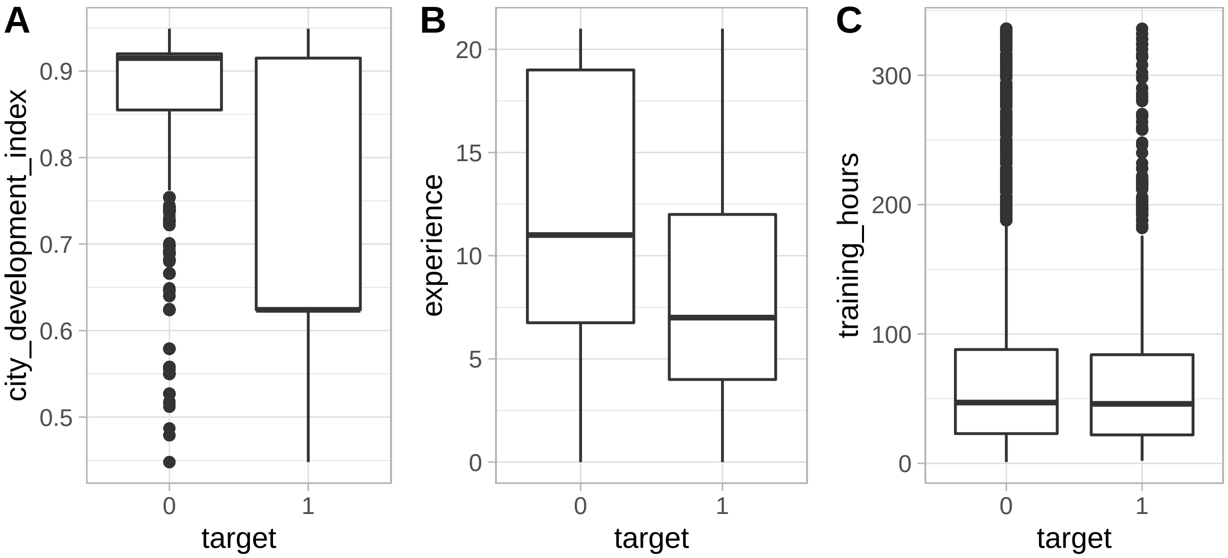
- city_ development _index : Developement index of the city (scaled)
- experience: Candidate total experience in years
- training_hours: training hours completed

Insights from above transformation:

- We can see that only the **training hours** has significant effect (approximately follow normal distribution) after taking the log.
- We can use log(training_hours) instead of training_hours in our model.

2.3 2D plot

```r
# city_development_index and target
box1 <- HRdata %>%
  ggplot(aes(x=target, y=city_development_index)) + geom_boxplot() + theme_light()

# experience and target
box2 <- HRdata %>%
  ggplot(aes(x=target, y=experience)) + geom_boxplot() + theme_light()

# training_hours and target
box3 <- HRdata %>%
  ggplot(aes(x=target, y=training_hours)) + geom_boxplot() + theme_light()

plot_grid(box1, box2, box3, labels = "AUTO", ncol=3)
```



3. Data Modelling

3.1 Data Spliting

```r
# Let's first split the data into training and test data (80/20)
library(caret)
set.seed(414)
idx_tr <- createDataPartition(HRdata$target, p=0.8, list=FALSE)

# Define training and test data
train <- HRdata[idx_tr,]
test  <- HRdata[-idx_tr,]

nrow(train)
```

```
## [1] 7165
```

```r
nrow(test)
```

```
## [1] 1790
```

```r
# proportion of target = 1 in whole dataset
summary(HRdata$target)[2]/sum(summary(HRdata$target))
```

```
##         1
## 0.1656058
```

```r
# proportion of target = 1 in train dataset
summary(train$target)[2]/sum(summary(train$target))
```

```
##         1
## 0.1656664
```

```r
# proportion of target = 1 in test dataset
summary(test$target)[2]/sum(summary(test$target))
```

```
##         1
## 0.1653631
```

- The distribution of target variable stay the same in training and testing datasets.

3.2 Model1: Logistic Regression

Using all the 11 features

```r
model1_LR <- glm(target ~ ., family=binomial, data=train)

# GOF (Hosmer-Lemeshow) -> p-value < 0.05, lack of fit
suppressMessages(library(ResourceSelection))
hoslem.test(model1_LR$y,fitted(model1_LR),g=10)
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  model1_LR$y, fitted(model1_LR)
## X-squared = 34, df = 8, p-value = 4.062e-05
```

Feature selection using AIC

```
suppressMessages(library(faraway))
# Feature selection using AIC
model1_LR_small <- step(model1_LR, trace=0)
sumary(model1_LR_small)
```

```
##                                              Estimate Std. Error  z value
## (Intercept)                                5.07220180 0.27659360  18.3381
## city_development_index                    -7.90843974 0.28303647 -27.9414
## relevent_experienceNo relevent experience  0.22758258 0.10482851   2.1710
## enrolled_universityno_enrollment          -0.34307101 0.10611573  -3.2330
## enrolled_universityPart time course       -0.53907907 0.17320185  -3.1124
## experience                                -0.03173049 0.00642043  -4.9421
## company_size10/49                          0.26131118 0.15850507   1.6486
## company_size100-500                        0.09322330 0.14964724   0.6230
## company_size1000-4999                      0.26282677 0.17159843   1.5316
## company_size10000+                         0.43859155 0.15324665   2.8620
## company_size50-99                          0.08985473 0.14449178   0.6219
## company_size500-999                        0.03347894 0.19219371   0.1742
## company_size5000-9999                      0.24302615 0.21570564   1.1267
## company_typeFunded Startup                 0.16814256 0.20949170   0.8026
## company_typeNGO                            0.18636766 0.25507380   0.7306
## company_typeOther                          1.08451663 0.38128408   2.8444
## company_typePublic Sector                  0.44302151 0.22563213   1.9635
## company_typePvt Ltd                        0.17298040 0.17707036   0.9769
## training_hours                            -0.00107045 0.00060317  -1.7747
##                                            Pr(>|z|)
## (Intercept)                               < 2.2e-16
## city_development_index                    < 2.2e-16
## relevent_experienceNo relevent experience  0.029931
## enrolled_universityno_enrollment           0.001225
## enrolled_universityPart time course        0.001856
## experience                                7.728e-07
## company_size10/49                          0.099230
## company_size100-500                        0.533315
## company_size1000-4999                      0.125612
## company_size10000+                         0.004210
## company_size50-99                          0.534029
## company_size500-999                        0.861713
## company_size5000-9999                      0.259888
## company_typeFunded Startup                 0.422193
## company_typeNGO                            0.464998
## company_typeOther                          0.004450
## company_typePublic Sector                  0.049592
## company_typePvt Ltd                        0.328618
## training_hours                             0.075948
```

```
##
## n = 7165 p = 19
## Deviance = 5219.29583 Null Deviance = 6433.42182 (Difference = 1214.12598)
```

```
# GOF: p-value < 0.05, lack of fit
hoslem.test(model1_LR_small$y,fitted(model1_LR_small),g=10)
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  model1_LR_small$y, fitted(model1_LR_small)
## X-squared = 37.5, df = 8, p-value = 9.31e-06
```

Likelihood Ratio Test: compare smaller and larger model

$$H_0 : \text{Smaller model selected by AIC is adequate} \;\; vs. \;\; H_a : \text{Larger model is adequate}$$

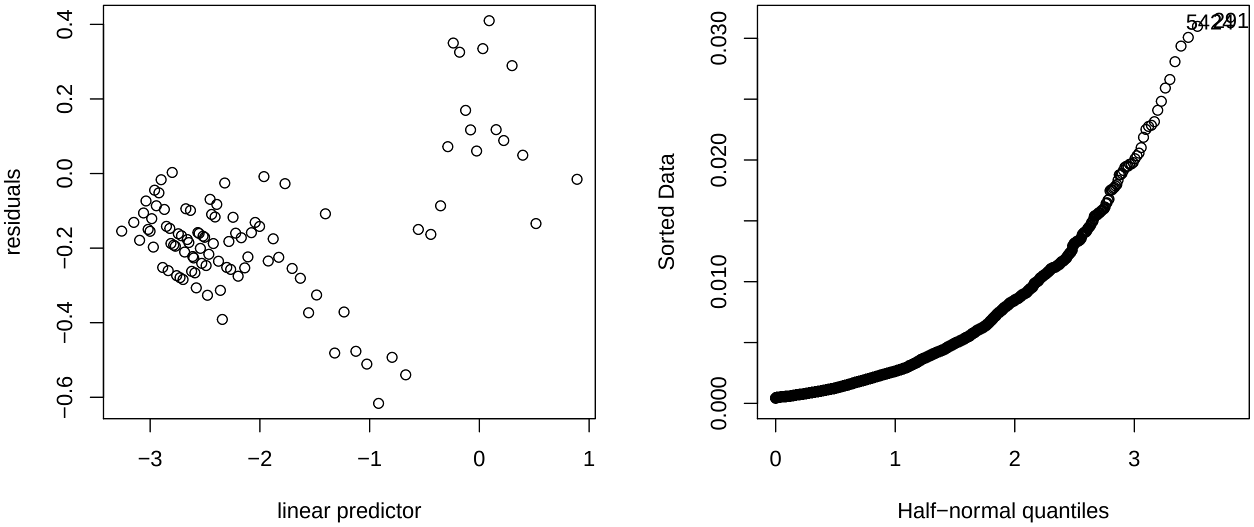
```
anova(model1_LR_small, model1_LR, test = "Chi")
```

```
## Analysis of Deviance Table
##
## Model 1: target ~ city_development_index + relevent_experience + enrolled_university +
##     experience + company_size + company_type + training_hours
## Model 2: target ~ city_development_index + gender + relevent_experience +
##     enrolled_university + education_level + major_discipline +
##     experience + company_size + company_type + last_new_job +
##     training_hours
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      7146     5219.3
## 2      7132     5201.3 14   18.004   0.2066
```

- LRT: Deviance = 18.004, follow χ^2_{14}
- The p-value is 0.2066, which is larger than 0.05, we fail to reject the null hypothesis. Thus, we prefer the smaller model selected by AIC.

Model Diagnostics

```
# Deviance Residuals plot
par(mfrow = c(1, 2))
train_assumption <- mutate(train, residuals=residuals(model1_LR_small),
                           linpred=predict(model1_LR_small))
gdf <- group_by(train_assumption, cut(linpred,
                                       breaks=c(min(linpred),
                                                unique(quantile(linpred, (1:100)/101)),
                                                max(linpred)),include.lowest = TRUE))
diagdf <- summarise(gdf, residuals=mean(residuals), linpred=mean(linpred), .groups = 'drop')

plot(residuals ~ linpred, diagdf, xlab="linear predictor")

# half-normal plot
halfnorm(hatvalues(model1_LR_small))
```



3.3 Model2: LDA

use the 7 features selected by AIC in Model 1 with log(training_hours)

```
suppressMessages(library(MASS))
model2_LDA = lda(target ~ city_development_index + relevent_experience +
                   enrolled_university + experience + company_size +
                   company_type + log(training_hours), data=train)
```

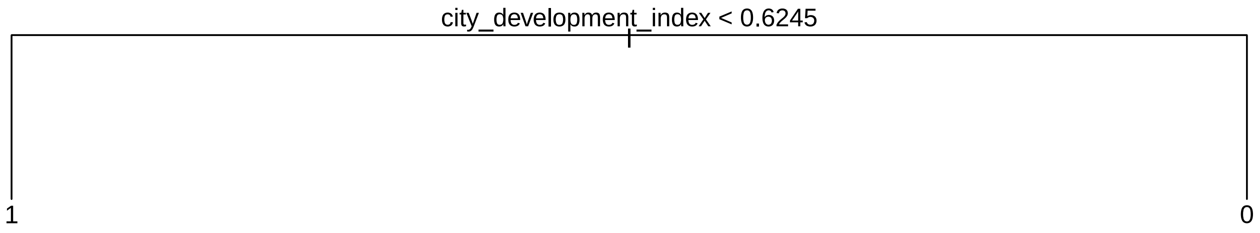
3.4 Model3: QDA

use the 7 features selected by AIC in Model 1 with log(training_hours)

```
model3_QDA = qda(target ~ city_development_index + relevent_experience +
                   enrolled_university + experience + company_size +
                   company_type + log(training_hours), data=train)
```

3.5 Model4: Classification Tree

```
suppressMessages(library(tree))
set.seed(108)
model4_tree <- tree(target ~ ., data=train)
plot(model4_tree); text(model4_tree, pretty=0)
```



```r
summary(model4_tree)
```

```
##
## Classification tree:
## tree(formula = target ~ ., data = train)
## Variables actually used in tree construction:
## [1] "city_development_index"
## Number of terminal nodes:  2
## Residual mean deviance:  0.7127 = 5105 / 7163
## Misclassification error rate: 0.1369 = 981 / 7165
```

```r
# no need to prune the tree
```

3.6 Model5: Bagging

```r
suppressMessages(library(randomForest))
set.seed(108)
model5_bagging <- randomForest(target ~ ., data=train, ntree=500,
                        mtry=11, importance=TRUE)
model5_bagging
```

```
##
## Call:
##  randomForest(formula = target ~ ., data = train, ntree = 500,      mtry = 11, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 11
##
##         OOB estimate of  error rate: 14.7%
## Confusion matrix:
##      0   1 class.error
## 0 5656 322  0.05386417
## 1  731 456  0.61583825
```

3.7 Model6: Random Forest

```r
set.seed(108)
model6_rf <- randomForest(target ~ ., data=train, ntree=500,
                        mtry=sqrt(11), importance=TRUE)
model6_rf
```

```
##
## Call:
##  randomForest(formula = target ~ ., data = train, ntree = 500,      mtry = sqrt(11), importance = TRU
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
```

```
##         OOB estimate of  error rate: 14.36%
## Confusion matrix:
##      0   1 class.error
## 0 5667 311  0.05202409
## 1  718 469  0.60488627
```

3.8 Model7: Boosting

```r
suppressMessages(library(gbm))
set.seed(108)

# gbm need character type response instead of factor
train_boost <- train
train_boost$target <- as.character(train_boost$target)
test_boost <- test
test_boost$target <- as.character(test_boost$target)


model7_boosting <- gbm(target ~ ., data=train_boost, distribution="bernoulli",
                n.trees=500, shrinkage=0.3)
model7_boosting
```

```
## gbm(formula = target ~ ., distribution = "bernoulli", data = train_boost,
##     n.trees = 500, shrinkage = 0.3)
## A gradient boosted model with bernoulli loss function.
## 500 iterations were performed.
## There were 11 predictors of which 11 had non-zero influence.
```

4. Model Selection using ROC Analysis

```r
suppressMessages(library(pROC))

# Create matrix to store the evaluation metrics for each model
eva_metrics = matrix(0, nrow=7, ncol=5)

# phat
phat1 <- predict(model1_LR_small, newdata=test, type="response")
phat2 <- predict(model2_LDA, newdata=test)$posterior[,2]
phat3 <- predict(model3_QDA, newdata=test)$posterior[,2]
phat4 <- predict(model4_tree, newdata=test)[,2]
phat5 <- predict(model5_bagging, newdata=test, type="prob")[,2]
phat6 <- predict(model6_rf, newdata=test, type="prob")[,2]
phat7 <- predict(model7_boosting, newdata=test_boost, type="response")

# create roc object
roc_obj1 <- roc(response=test$target, predictor=phat1)
roc_obj2 <- roc(response=test$target, predictor=phat2)
roc_obj3 <- roc(response=test$target, predictor=phat3)
roc_obj4 <- roc(response=test$target, predictor=phat4)
roc_obj5 <- roc(response=test$target, predictor=phat5)
roc_obj6 <- roc(response=test$target, predictor=phat6)
roc_obj7 <- roc(response=test$target, predictor=phat7)

# calculate AUC
AUC1 <- auc(roc_obj1)
AUC2 <- auc(roc_obj2)
AUC3 <- auc(roc_obj3)
AUC4 <- auc(roc_obj4)
AUC5 <- auc(roc_obj5)
AUC6 <- auc(roc_obj6)
AUC7 <- auc(roc_obj7)

# show the performance matric
roc_1 <- c(coords(roc_obj1, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC1)
eva_metrics[1,] <- t(roc_1)

roc_2 <- c(coords(roc_obj2, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC2)
eva_metrics[2,] <- t(roc_2)

roc_3 <- c(coords(roc_obj3, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC3)
eva_metrics[3,] <- t(roc_3)

roc_4 <- c(coords(roc_obj4, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC4)
eva_metrics[4,] <- t(roc_4)

roc_5 <- c(coords(roc_obj5, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC5)
```

```
eva_metrics[5,] <- t(roc_5)

roc_6 <- c(coords(roc_obj6, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC6)
eva_metrics[6,] <- t(roc_6)

roc_7 <- c(coords(roc_obj7, "b", ret=c("threshold","se","sp","accuracy"),
                         best.method="youden", transpose=TRUE), AUC7)
eva_metrics[7,] <- t(roc_7)

# Create metrics df
metrics <- as.data.frame(eva_metrics)
colnames(metrics) = c("Threshold","Sensitivity","Specificity","Accuracy","AUC")
rownames(metrics) = c("Logistic Regression","LDA","QDA","Tree","Bagging","RF","Boosting")
metrics
```

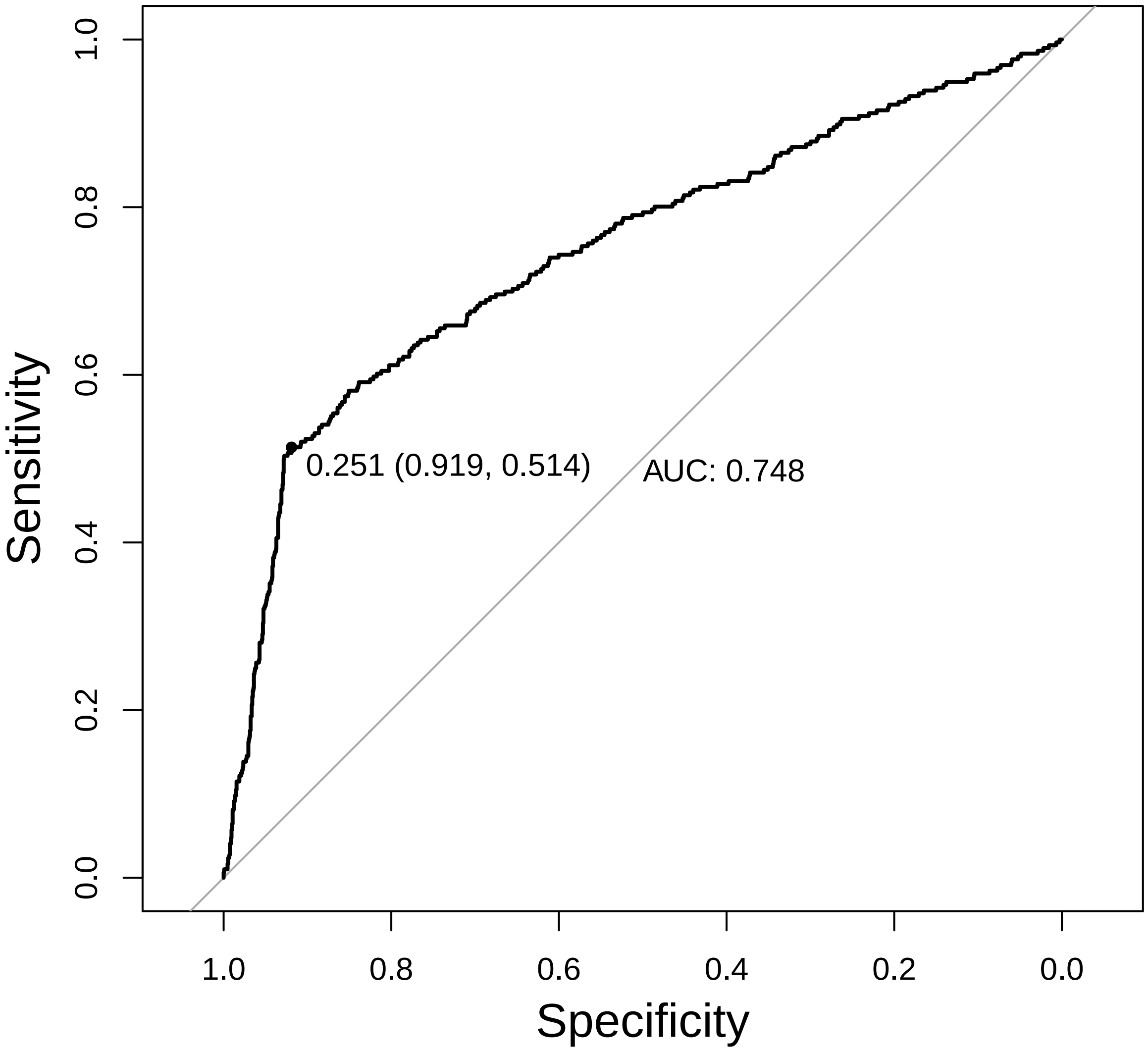
```
##                     Threshold Sensitivity Specificity  Accuracy       AUC
## Logistic Regression 0.3509160   0.5202703   0.9149933 0.8497207 0.7362988
## LDA                 0.4368809   0.5236486   0.9136546 0.8491620 0.7363350
## QDA                 0.2258706   0.5743243   0.8159304 0.7759777 0.7114675
## Tree                0.3411479   0.4966216   0.9243641 0.8536313 0.7104929
## Bagging             0.3270000   0.5000000   0.9022758 0.8357542 0.7410735
## RF                  0.2670000   0.5202703   0.9049531 0.8413408 0.7430703
## Boosting            0.2514782   0.5135135   0.9190094 0.8519553 0.7479411
```

- The best model based on the highest AUC is Boosting, the AUC = 0.7479411.

5. Analyze the Best Performing Model - Boosting

5.1 Boosting ROC Analysis

```
# produce ROC Curve
plot(roc_obj7,legacy.axes=F,print.auc=T,print.thres=T,cex.lab=1.5)
```



5.2 Boosting Confusion Matrix using Best Threshold

```
# make prediction cutoff=0.2514782
# Obtain Y_hat values for the data observation (cutoff=0.2514782)
proba_hat <- predict(model7_boosting, newdata=test_boost, type="response")

n = nrow(test); y_hat = rep(0,n)
cutoff = 0.2514782; idx = which(proba_hat > cutoff)
y_hat[idx] = 1
```

```r
# confusion matrix at cutoff=0.2514782
(conf_mat = table(predicted = y_hat, actual = test$target))
```

```
##          actual
## predicted    0    1
##         0 1373  144
##         1  121  152
```

```r
# sensitivity/recall
conf_mat[2, 2] / sum(conf_mat[, 2])
```

```
## [1] 0.5135135
```

```r
# precision/positive predictive value
conf_mat[2, 2] / sum(conf_mat[2, ])
```

```
## [1] 0.5567766
```

```r
# specificity
conf_mat[1, 1] / sum(conf_mat[, 1])
```

```
## [1] 0.9190094
```

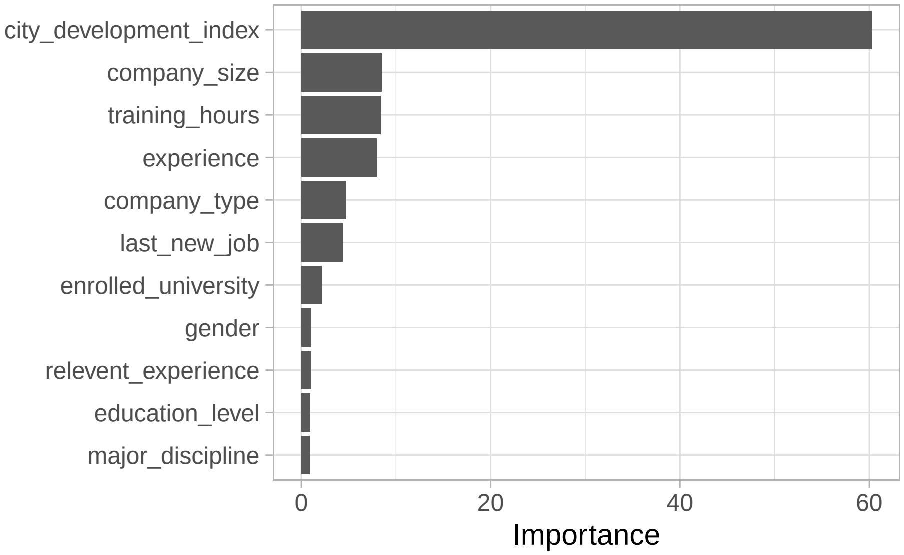
	Summary Metrics for Boosting
Sensitivity/Recall	0.5135135
Precision/Positive Predictive Value	0.5567766
Specificity	0.9190094
Accuracy	0.8519553
AUC	0.7479411

5.3 Boosting Summary and Feature Importance

```r
summary(model7_boosting, las=1, plotit=F)
```

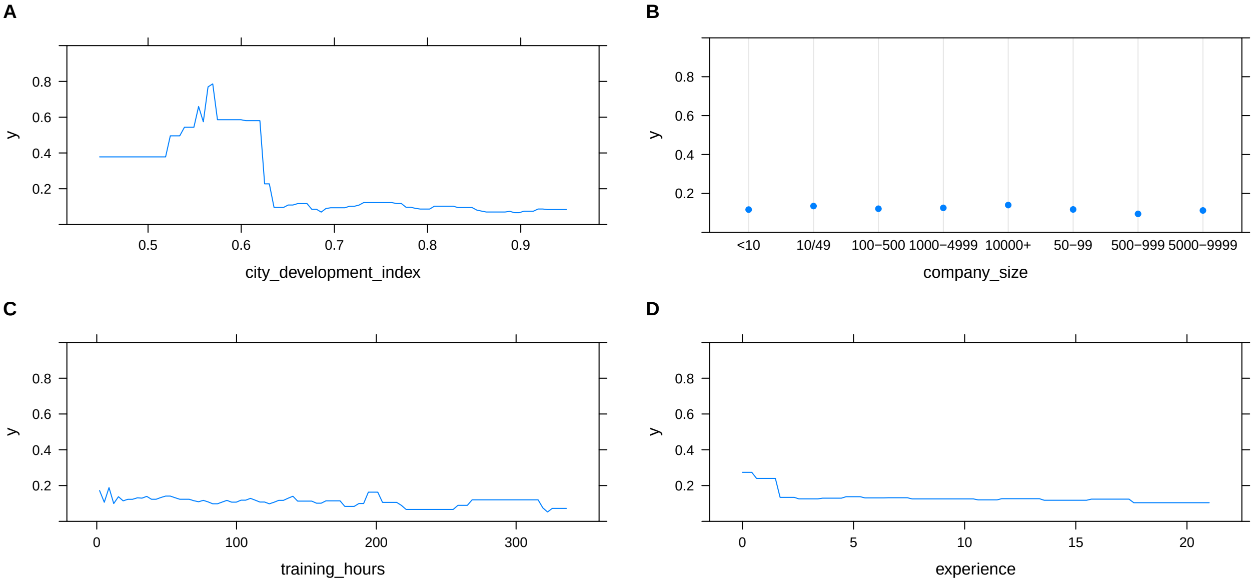
```
##                                         var    rel.inf
## city_development_index city_development_index 60.2531637
## company_size                       company_size  8.4526089
## training_hours                   training_hours  8.3464673
## experience                           experience  7.9438939
## company_type                       company_type  4.7450783
## last_new_job                       last_new_job  4.3622290
## enrolled_university         enrolled_university  2.1210473
## gender                                   gender  1.0072945
## relevent_experience         relevent_experience  1.0043377
## education_level                 education_level  0.9018400
## major_discipline               major_discipline  0.8620393
```

```r
vip::vip(model7_boosting, num_features=11) + theme_light()
```



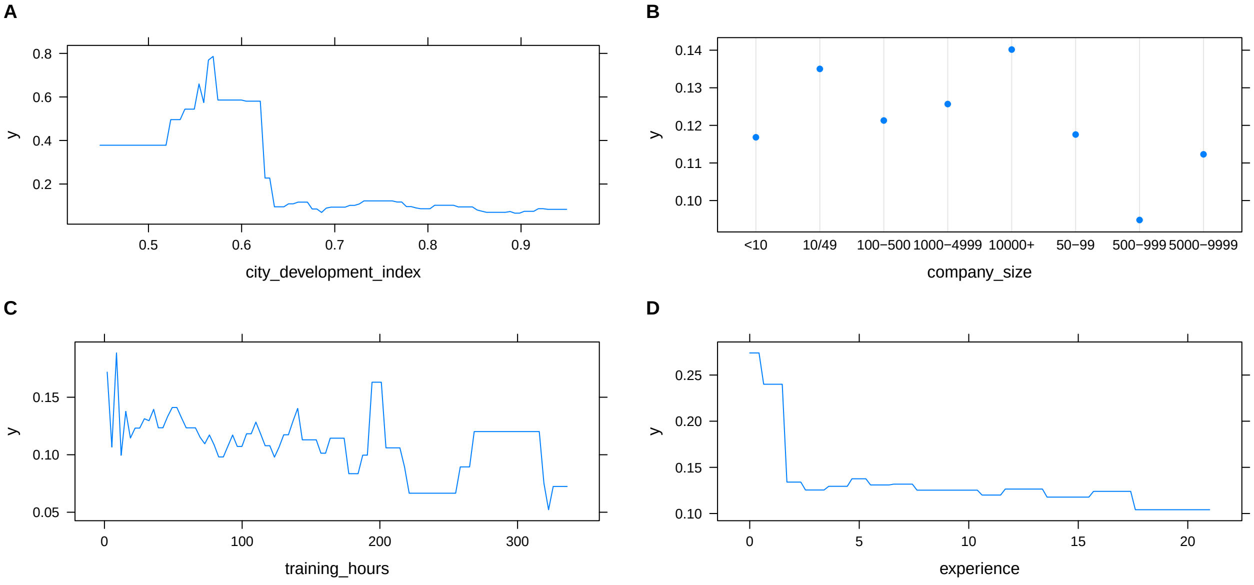
Probability of Success vs the Top 4 Most Important Feature - Same Scale

```r
city <- plot.gbm(model7_boosting, 1, type="response", ylim=range(0:1))
company <- plot.gbm(model7_boosting, 8, type="response", ylim=range(0:1))
training <- plot.gbm(model7_boosting, 11, type="response", ylim=range(0:1))
experience <- plot.gbm(model7_boosting, 7, type="response", ylim=range(0:1))

plot_grid(city, company, training, experience,
          labels = "AUTO", ncol=2)
```



Probability of Success vs the Top 4 Most Important Feature - Different Scale

```r
city <- plot.gbm(model7_boosting, 1, type="response")
company <- plot.gbm(model7_boosting, 8, type="response")
training <- plot.gbm(model7_boosting, 11, type="response")
experience <- plot.gbm(model7_boosting, 7, type="response")

plot_grid(city, company, training, experience,
          labels = "AUTO", ncol=2)
```


A
y
city_development_index
B
y
<10
10/49
company_size
C
y
training_hours
D
y
experience