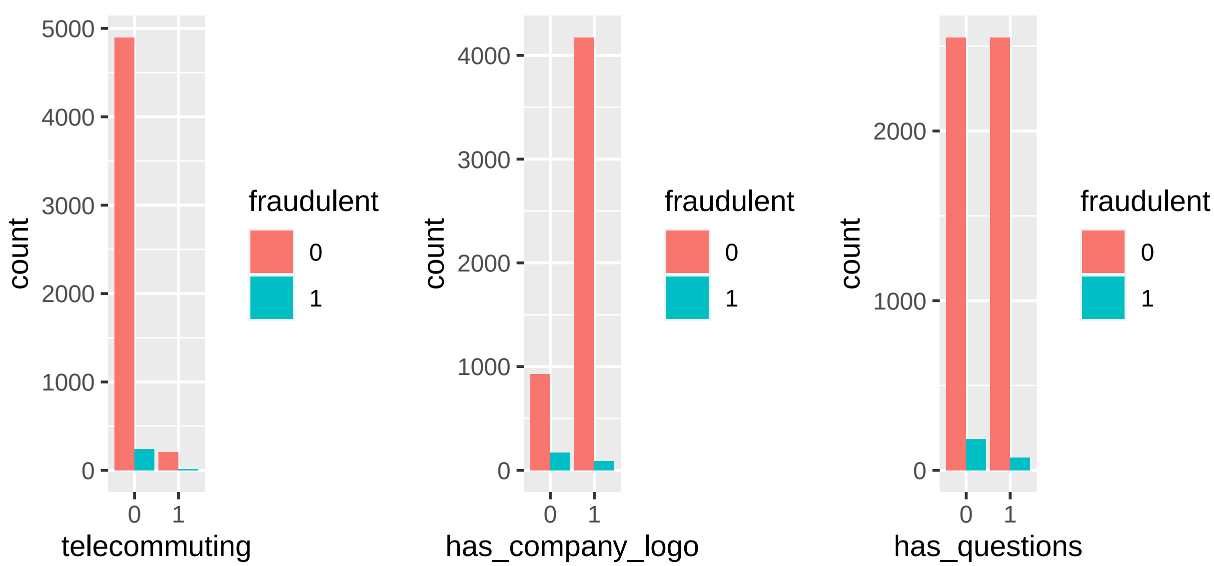
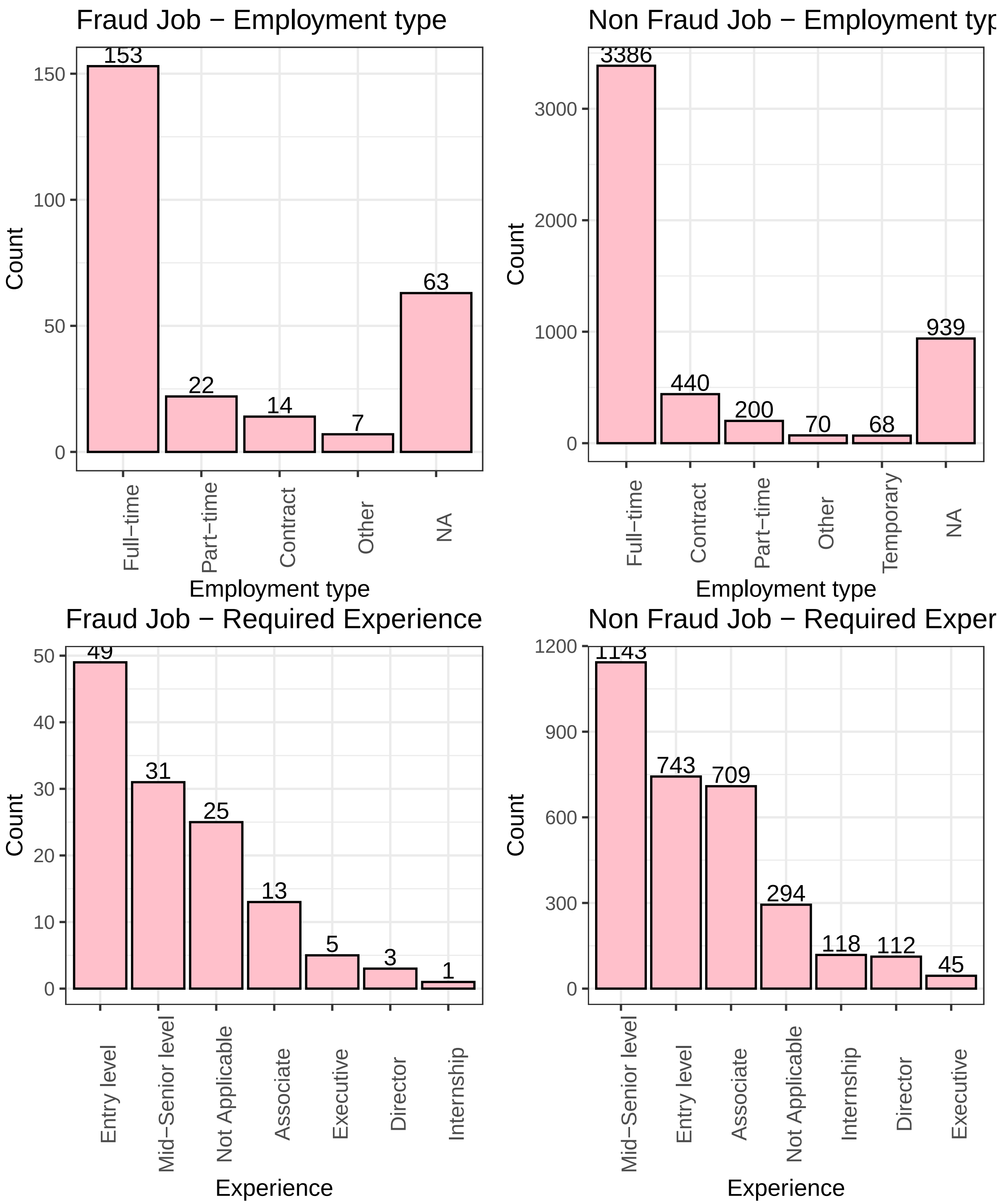
1 | Data Description

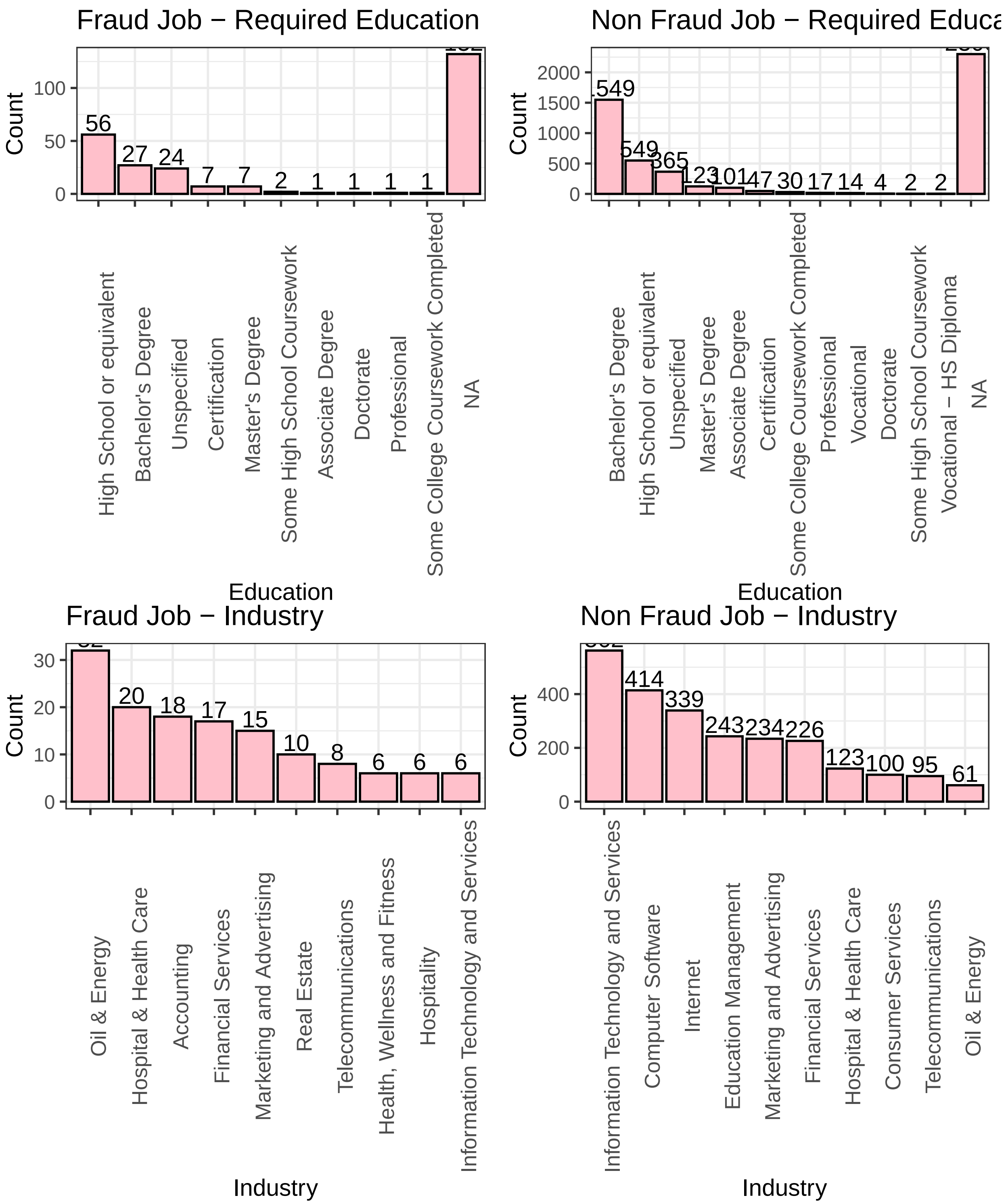
We will analyze the data on fraudulent behavior of job posts. The dataset contains information for 5,362 observations that define job postings. These job postings are categorized as either fake or real. The dataset is highly unbalanced, with 259(4.8% of the jobs) being fraudulent jobs and 5,103(around 95.2% of the jobs) the real jobs. For each job posting recorded, the dataset contains the following 18 variables where *fraudulent* is treated as our target variable:

Variable Name	Type	Description
`job_id`	int	Unique Job ID
`title`	text	The title of the job ad entry
`location`	text	Geographical location of the job ad
`department`	text	Corporate department
`salary_range`	text	Expected salary range
`company_profile`	text	Company description
`description`	text	The details description of the job ad
`requirements`	text	Enlisted requirements for the job opening
`benefits`	text	Enlisted offered benefits by the employer
`telecommuting`	boolean	output class [1: telecommuting, 0: Not]
`has_company_logo`	boolean	output class [1: has company logo, 0: Not]
`has_questions`	boolean	output class [1: has questions, 0: Not]
`employment_type`	text	Employment type e.g(Full-time, Part-time)
`required_experience`	text	Required experience e.g(Intership)
`required_education`	text	Required education level
`industry`	text	Job industry
`function`	text	job position
`fraudulent`	boolean	output class [1: fraudulent, 0: Not]



The graph above shows most jobs do not require telecommuting, and has company logo. *has_question* graphs shows almost same value for has question or not if real.


Fraud Job − Employment type
Count
150
100
50
0
153
22
14
7
63
Full−time
Part−time
Contract
Other
NA
Employment type
Non Fraud Job − Employment typ
Count
3000
2000
1000
0
3386
440
200
70
68
939
Full−time
Contract
Part−time
Other
Temporary
NA
Employment type
Fraud Job − Required Experience
Count
50
40
30
20
10
0
49
31
25
13
5
3
1
Entry level
Mid−Senior level
Not Applicable
Associate
Executive
Director
Internship
Experience
Non Fraud Job − Required Exper
Count
1200
900
600
300
0
1143
743
709
294
118
112
45
Mid−Senior level
Entry level
Associate
Not Applicable
Internship
Director
Executive
Experience


Fraud Job − Required Education
Count
0
50
100
56
27
24
7
7
2
1
1
1
1
High School or equivalent
Bachelor's Degree
Unspecified
Certification
Master's Degree
Some High School Coursework
Associate Degree
Doctorate
Professional
Some College Coursework Completed
NA
Education
Non Fraud Job − Required Educa
Count
0
500
1000
1500
2000
549
365
123
101
47
30
17
14
4
2
2
Bachelor's Degree
High School or equivalent
Unspecified
Master's Degree
Associate Degree
Certification
Some College Coursework Completed
Professional
Vocational
Doctorate
Some High School Coursework
Vocational − HS Diploma
NA
Education
Fraud Job − Industry
Count
0
10
20
30
20
18
17
15
10
8
6
6
6
Oil & Energy
Hospital & Health Care
Accounting
Financial Services
Marketing and Advertising
Real Estate
Telecommunications
Health, Wellness and Fitness
Hospitality
Information Technology and Services
Industry
Non Fraud Job − Industry
Count
0
200
400
414
339
243
234
226
123
100
95
61
Information Technology and Services
Computer Software
Internet
Education Management
Marketing and Advertising
Financial Services
Hospital & Health Care
Consumer Services
Telecommunications
Oil & Energy
Industry

The graphs above show that most fraud jobs belong to the full-time category, requiring entry-level, and high school or equivalent. Most non-fraud jobs belong to the full-time category, requiring a mid-senior level, and bachelor's degree. The graphs for education, The Oil&Energy category is first on fraud jobs but lasts on the non-fraud job graphs. Engineering and Information Technology are most on fraud and non-fraud function graphs, and Engineering and Sales are most on fraud and non-fraud department graphs. From the distributions between categories, fraud and non-fraud jobs have similar features. It is not easy for humans to classify these posts as fraudulent or not.

2 | Feature Creation

We created the corpus objects for each complex text feature(title, company_profile, description, requirements, benefits). To clean the text features, we converted all words to lowercase and then removed numbers, punctuations, stop words and stripped extra whitespaces. After that process, we conducted stemming to extract stems for the words and made the term frequency matrix. We removed sparse terms by setting the maximal allowed sparsity to 0.8. This is because when we set higher values as the maximal allowed sparsity, there were so many features that made the computation so expensive. We also indicated which column of the original dataset the word feature came from in the column name of each word feature. We combined the word features with the binary features and the dummy variables from categorical features. As a result, the dimension of my feature matrix is (5362, 76). This means we end up with 82 features as of now.

3 | Unsupervised Feature Filtering

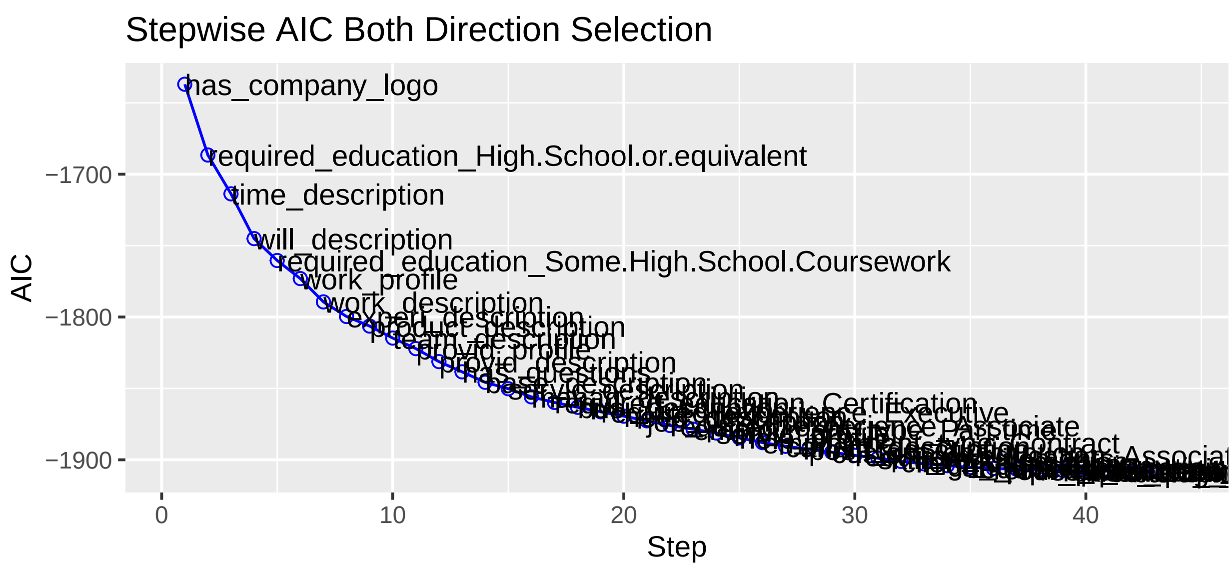


Figure 1: Stepwise Selection

In order to filter the features provided from the above process, we used the stepwise selection process and selected the best model using cross-validated prediction error of AIC. There was no problem using the "ols_step_both_aic" function from the olsrr package with our feature dimension, and it yielded a better result than the forward and backward selection. By applying this method, we reduced are features to 43 making our matrix dimension (5362, 43).

4 | Power Feature Creation

These are power features we included in our model.

Variable Name	Description
`state_TX`	Whether the job post is from Texas
`length_des`	The length of description
`length_ben`	The length of benefits
`state_NY`	Whether the job post is from New York
`state_CA`	Whether the job post is from California
`contai_email`	Whether the job post contains email address

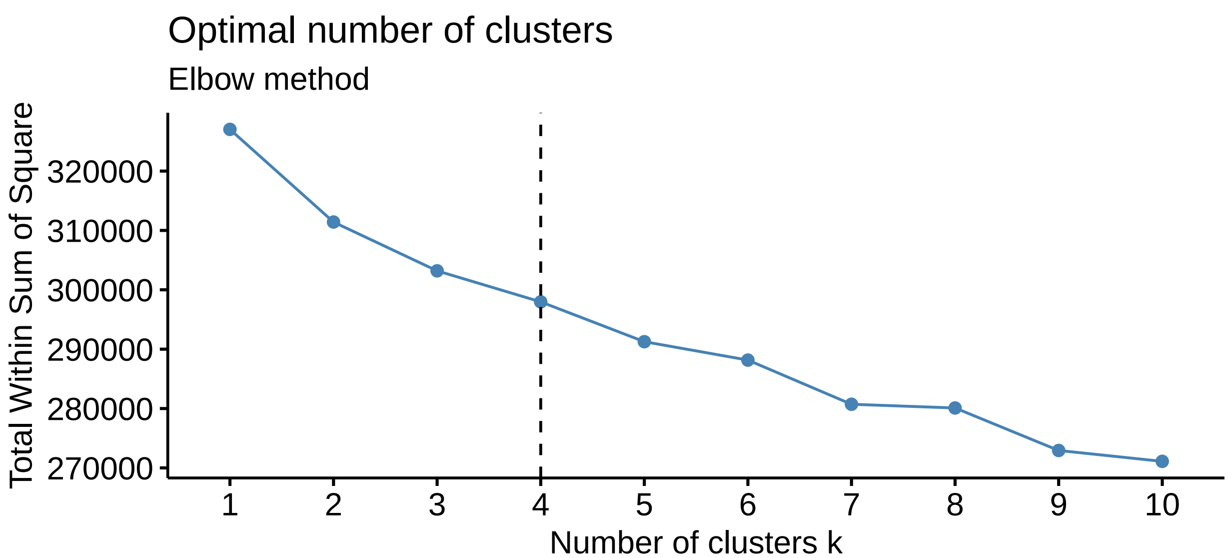
Variable Name	Description
length_req	The length of requirements
contain_phone	Whether the job post contains phone number
has_salary	Whether the job post shows its salary
oil_ind	Whether the job post is from 'Oil & Energy' industry
hos_ind	Whether the job post is from 'Hospital & Health Care' industry
acc_ind	Whether the job post is from 'Accounting' industry
oil_dept	Whether the job post is from 'Oil & Energy' department
length_profile	The length of profile
eng_dept	Whether the job post is from 'Engineering' department
upper_des	The number of uppercase letter in description
upper_req	The number of uppercase letter in requirements
upper_ben	The number of uppercase letter in benefits

5 | Feature and Power Feature Combination

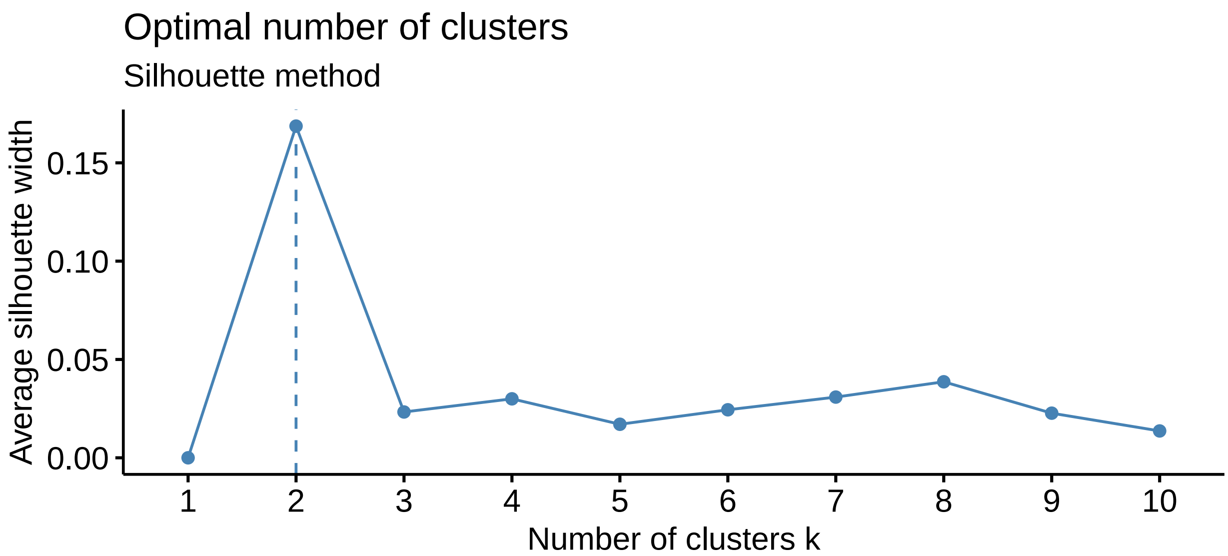
6 | Classification on the Feature Matrix

ROC Curve

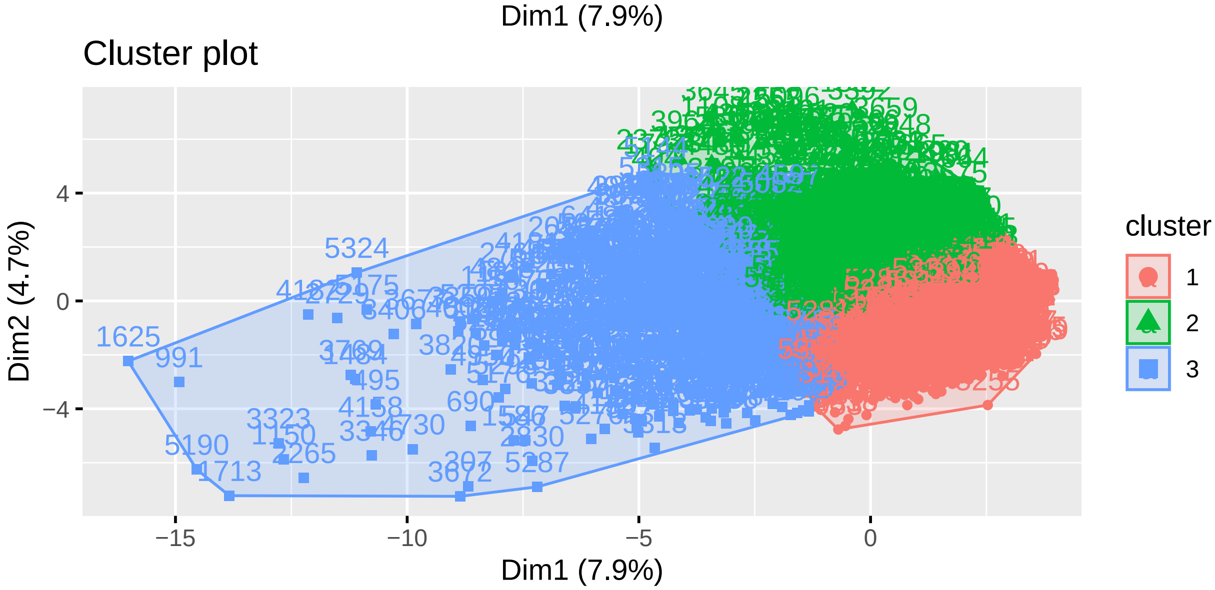
7 | Clustering

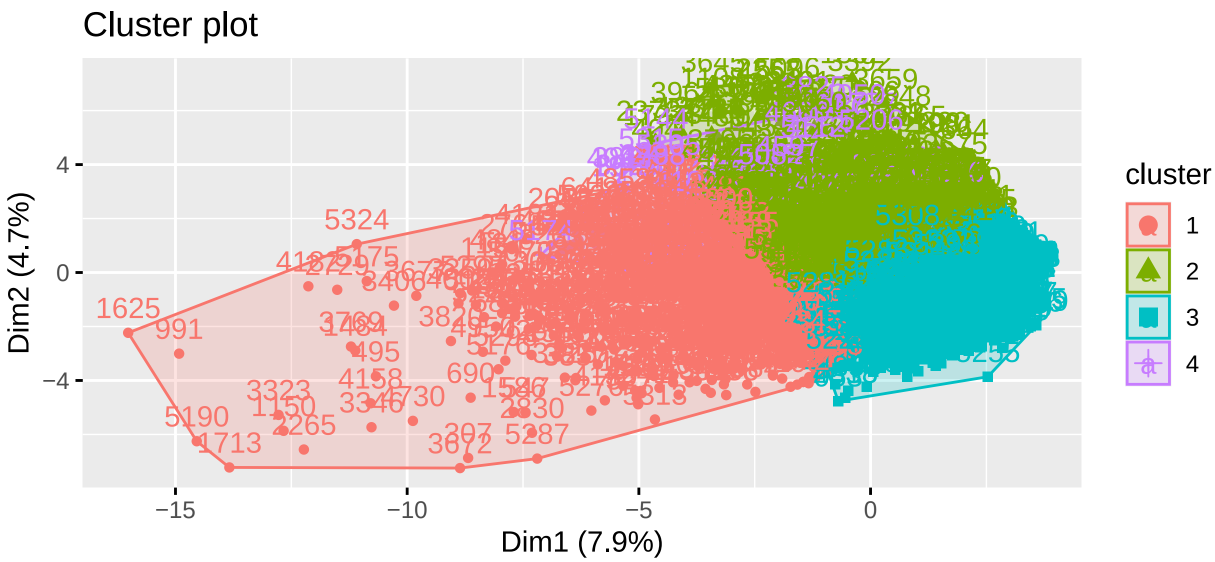



```
## Warning: did not converge in 10 iterations
```


Optimal number of clusters
Silhouette method
Average silhouette width
0.00
0.05
0.10
0.15
1
2
3
4
5
6
7
8
9
10
Number of clusters k


Cluster plot
Dim2 (4.7%)
Dim1 (7.9%)
−15
−10
−5
0
−4
0
4
cluster
1
2

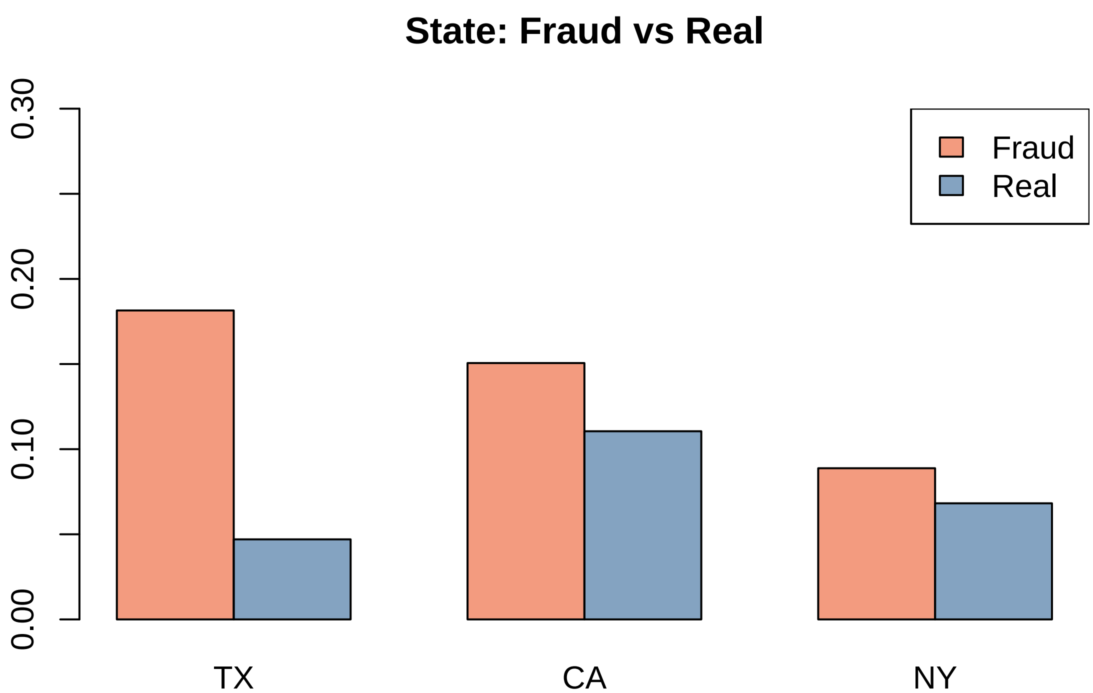

Cluster plot
Dim2 (4.7%)
Dim1 (7.9%)
−15
−10
−5
0
−4
0
4
cluster
1
2
3



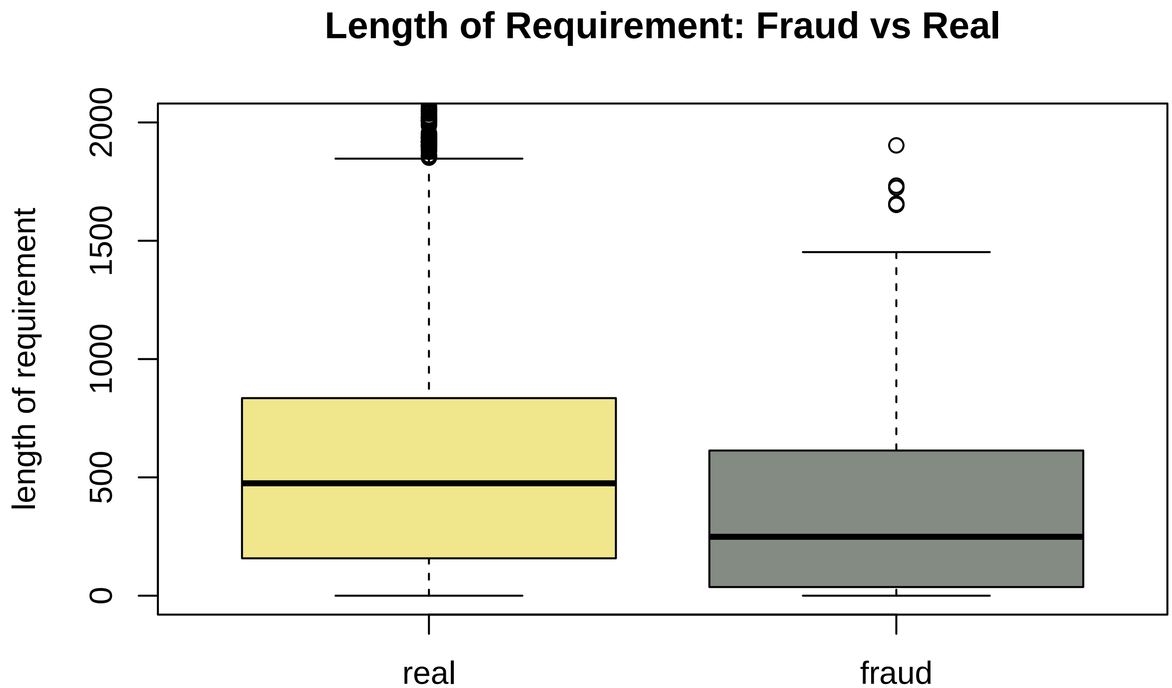
The Elbow method suggests 4 as the number of optimal clusters, and the Silhouette method shows that 2 is optimal. So we plotted 2, 3, and 4 clusters to see which one is better for humans to classify, and we tried all clusters. 4 cluster is the better one, but it is highly unbalanced that cluster 1 contains 750 observations, cluster 2 contains 2536, cluster 3 contains 1960, and cluster 4 contains 116. Cluster 1 contains 15.83% of fraudulent observations, cluster 2 contains 17.76%, cluster 3 contains 64.09%, and cluster 4 contains 2.31%. Clustering does not helpful in our feature.

8 | Improve Features

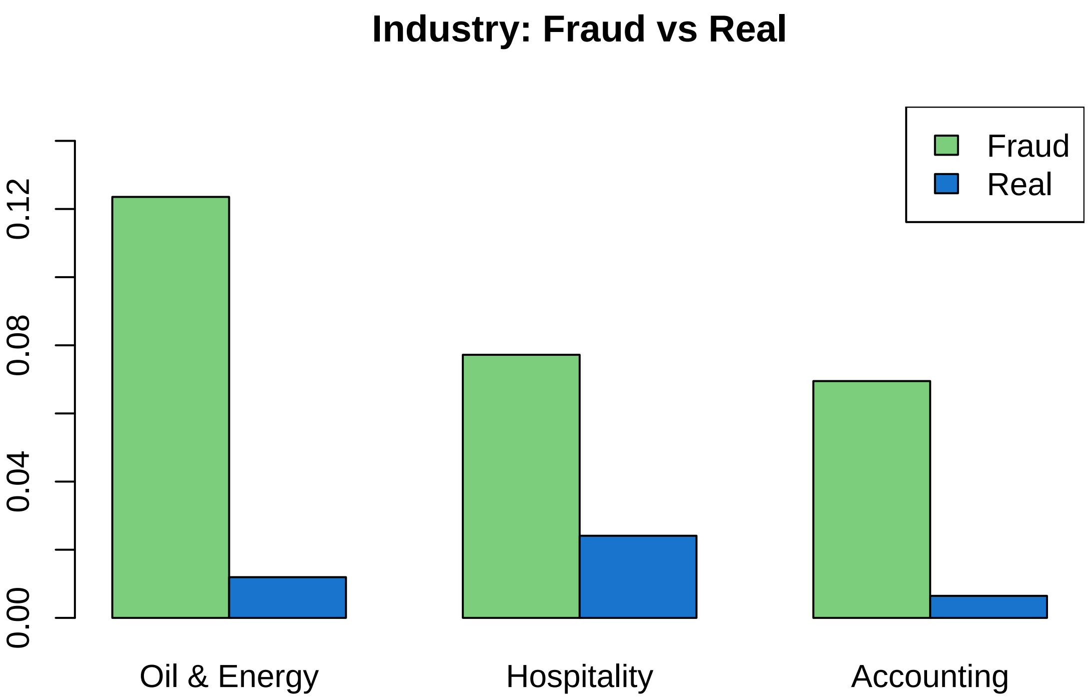
These are power features for improving the model. We selected some of these features that showed significant difference to classify fraudulent job post.



There is a difference in proportion of state between fraudulent job post and real job post. Many job posts were from Texas and the proportion was greater than the real job post.



The length of requirement was longer in the real job post.



There is a significant difference in proportion of industry between fraudulent job post and real job post.

9 | Validation Set

10 | Appendix

```r
library(tidyverse)
library(ggplot2)
library(gridExtra)
library(dplyr)
library(plyr)
library(ggthemes)
library(skimr)
library(randomForest)
library(e1071)
library(DataExplorer)
library(cluster)
library(haven)
library(ggdendro)
library(NbClust)
library(factoextra)
library(klaR)
library(tm)
library(SnowballC)
library(tidytext)
library(data.table)
library(rlang)
library(ggpubr)
library(corrplot)
job <- read_csv("~/Desktop/job_training_data.csv")
#bivariate distributions
#factor
job$telecommuting = as.factor(job$telecommuting)
job$has_company_logo = as.factor(job$has_company_logo)
job$has_questions = as.factor(job$has_questions)
job$fraudulent = as.factor(job$fraudulent)

#filter location
#telecommuniting
bar_tele = ggplot(job, aes(x = telecommuting,
           fill = fraudulent)) + geom_bar(position = "dodge")

#has_company_logo
bar_logo = ggplot(job, aes(x = has_company_logo,
           fill = fraudulent)) + geom_bar(position = "dodge")

#has_questions
bar_questions = ggplot(job, aes(x = has_questions,
           fill = fraudulent)) + geom_bar(position = "dodge")

grid.arrange(bar_tele, bar_logo, bar_questions,
             ncol=3, nrow=1)

#employment type
```

```r
fjobemp = job %>% filter(fraudulent == 1) %>% group_by(employment_type) %>% dplyr::summarize(Freq = n()
  ggplot(aes(x = reorder(employment_type, -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Fraud Job - Employment type", x="Employment type", y="Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))

nfjobemp = job %>% filter(fraudulent == 0) %>% group_by(employment_type) %>%
  dplyr::summarize(Freq = n()) %>% arrange(desc(Freq)) %>%
  ggplot(aes(x = reorder(employment_type, -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Non Fraud Job - Employment type", x="Employment type", y="Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))

ggarrange(fjobemp, nfjobemp)

#required experience

fjobexp = job %>% filter(fraudulent == 1) %>% group_by(required_experience) %>% dplyr::summarize(Freq =
  ggplot(aes(x = reorder(required_experience, -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Fraud Job - Required Experience", x="Experience", y="Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))

nfjobexp = job %>% filter(fraudulent == 0) %>% group_by(required_experience) %>%
  dplyr::summarize(Freq = n()) %>% arrange(desc(Freq)) %>% slice(2:11) %>%
  ggplot(aes(x = reorder(required_experience, -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Non Fraud Job - Required Experience", x="Experience", y="Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))

ggarrange(fjobexp, nfjobexp)

#Fraud job required education

fjobedu <- job %>% filter(fraudulent == 1) %>% group_by(required_education) %>% dplyr::summarize(Freq =
  ggplot(aes(x = reorder(required_education, -Freq), y = Freq)) + geom_bar(stat = "identity", color = "
  theme_bw() + labs(title = "Fraud Job - Required Education",
                    x = "Education",

                    y = "Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) + theme(axis.text.x=element_text(size=10, angle=90,h

nfjobedu <- job %>% filter(fraudulent == 0) %>% group_by(required_education)%>% dplyr::summarize(Freq =
  ggplot(aes(x = reorder(required_education, -Freq), y = Freq)) + geom_bar(stat = "identity", color = "
  theme_bw() + labs(title = "Non Fraud Job - Required Education",
                    x = "Education",
                    y = "Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) + theme(axis.text.x=element_text(size=10, angle=90,h
```

```r
ggarrange(fjobedu, nfjobedu)

#industry
fjobind = job %>% filter(fraudulent == 1) %>% group_by(industry) %>% dplyr::summarize(Freq = n()) %>% a
  ggplot(aes(x = reorder(industry, -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Fraud Job - Industry",
                    x = "Industry", y = "Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))

nfjobind = job %>% filter(fraudulent == 0) %>% group_by(industry) %>%
  dplyr::summarize(Freq = n()) %>% arrange(desc(Freq)) %>% slice(2:11) %>%
  ggplot(aes(x = reorder(industry, -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Non Fraud Job - Industry",
                    x = "Industry", y = "Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))
ggarrange(fjobind, nfjobind)

#fuction

fjobfunc = job %>% filter(fraudulent == 1) %>% group_by(function.) %>% dplyr::summarize(Freq = n()) %>%
  ggplot(aes(x = reorder(function., -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Fraud Job - Function.", x="Function.", y="Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))

nfjobfunc = job %>% filter(fraudulent == 0) %>% group_by(function.) %>%
  dplyr::summarize(Freq = n()) %>% arrange(desc(Freq)) %>% slice(2:11) %>%
  ggplot(aes(x = reorder(function., -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Non Fraud Job - function.", x="Function.", y="Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))

ggarrange(fjobfunc, nfjobfunc)

#department
fjobdep = job %>% filter(fraudulent == 1) %>% group_by(department) %>% dplyr::summarize(Freq = n()) %>%
  ggplot(aes(x = reorder(department, -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Fraud Job - Department", x="Department", y="Count") +
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))

nfjobdep = job %>% filter(fraudulent == 0) %>% group_by(department) %>%
  dplyr::summarize(Freq = n()) %>% arrange(desc(Freq)) %>% slice(2:11) %>%
  ggplot(aes(x = reorder(department, -Freq), y = Freq)) +
  geom_bar(stat = "identity", color = "black", fill = "pink") +
  theme_bw() + labs(title = "Non Fraud Job - Department", x="Department", y="Count") +
```

```
  geom_text(aes(label=round(Freq,0)), vjust= -0.2) +
  theme(axis.text.x=element_text(size=10, angle=90,hjust=0.5,vjust=1))
ggarrange(fjobdep, nfjobdep)
job <- read_csv("~/Desktop/job_training_data.csv")
fraud <- job %>% filter(fraudulent == 1)
not_fraud <- job %>% filter(fraudulent == 0)
head(fraud)
head(not_fraud)
dim(fraud)
dim(not_fraud)
colSums(is.na(job))
# Create the dummy variables for categorical data(employment_type, required_experience, required_educat
library(fastDummies)
categorical <- subset(job, select = c(employment_type, required_experience, required_education))
ctgr_dummy <- dummy_cols(categorical, select_columns=c('employment_type', 'required_experience', 'requi
ctgr_dummy[is.na(ctgr_dummy)] <- 0
colnames(ctgr_dummy) <- gsub(" ", ".", colnames(ctgr_dummy))
colnames(ctgr_dummy) <- gsub("-", ".", colnames(ctgr_dummy))
colnames(ctgr_dummy) <- gsub("'", ".", colnames(ctgr_dummy))

# Parse out the word features from the complex text features(title, company_profile, description, requi

title_corpus <- VCorpus(VectorSource(job$title))
profile_corpus <- VCorpus(VectorSource(job$company_profile))
description_corpus <- VCorpus(VectorSource(job$description))
requirements_corpus <- VCorpus(VectorSource(job$requirements))
benefits_corpus <- VCorpus(VectorSource(job$benefits))

clean_text <- function(corpus){
  corpus <- tm_map(corpus, content_transformer(tolower), lazy = T)
  corpus <- tm_map(corpus, removeNumbers, lazy = T)
  corpus <- tm_map(corpus, removePunctuation, lazy = T)
  corpus <- tm_map(corpus, removeWords, stopwords(kind = "en"), lazy = T )
  corpus <- tm_map(corpus, stripWhitespace, lazy = T )
  corpus <- tm_map(corpus, stemDocument, lazy = T)
  corpus <- tm_map(corpus, stripWhitespace, lazy = T)
  word_freq <- DocumentTermMatrix(corpus)
  remove_sparse <- removeSparseTerms(word_freq, 0.8)
  remove_sparse_df <- as.data.frame(as.matrix(remove_sparse))
  return(remove_sparse_df)
}


cleaned_title <- clean_text(title_corpus)
if(ncol(cleaned_title) > 0){
  cleaned_title <- cleaned_title %>% rename_all(paste0, "_title")
}


cleaned_profile <- clean_text(profile_corpus)
if(ncol(cleaned_profile) > 0){
  cleaned_profile <- cleaned_profile %>% rename_all(paste0, "_profile")
}
```

```r
cleaned_description <- clean_text(description_corpus)
if(ncol(cleaned_description) > 0){
  cleaned_description <- cleaned_description %>% rename_all(paste0, "_description")
}

cleaned_requirements <- clean_text(requirements_corpus)
if(ncol(cleaned_requirements) > 0){
  cleaned_requirements <- cleaned_requirements %>% rename_all(paste0, "_requirements")
}

cleaned_benefits <- clean_text(benefits_corpus)
if(ncol(cleaned_benefits) > 0){
  cleaned_benefits <- cleaned_benefits %>% rename_all(paste0, "_benefits")
}

# Combine features
# the word features (from title, company_profile, description, requirements, benefits)
# + binary features (telecommuting, has_company_logo, has_questions)
# + categorical features (employment_type, required_experience, required_education)

library(plyr)
cleaned_description$fraudulent <- job$fraudulent
features <- cbind(cleaned_title, cleaned_profile, cleaned_description, cleaned_requirements, cleaned_ben

# Feature selection using stepwise selection

features_model = lm(fraudulent ~., data = features)
library(MASS)
library(olsrr)
ols_model = ols_step_both_aic(features_model, details= TRUE)
ols_model$predictors
plot(ols_model)
library(tidyverse)

make_power_features <- function(df) {

  # country, state
  split_location <- strsplit(df$location, ", ")

  country <- c()
  for (i in 1:length(split_location)) {
    country <- c(country, split_location[[i]][1])
  }
  country[is.na(country)] <- 0

  state <- c()
  for (i in 1:length(split_location)) {
    state <- c(state, split_location[[i]][2])
  }
  state[is.na(state)] <- 0

  country_state <- data.frame(country = country, state = state)
  country_state[country_state$country != "US", "state"] <- ""
```

```r
# state_TX
state_TX <- country_state$state
state_TX <- ifelse(state_TX == "TX", 1, 0)
#state_TX[state_TX != "TX"] <- 0
#state_TX[state_TX == "TX"] <- 1
state_TX = as.numeric(state_TX)

# state_NY
state_NY <- country_state$state
state_NY <- ifelse(state_NY == "NY", 1, 0)
#state_NY[state_NY != "NY"] <- 0
#state_NY[state_NY == "NY"] <- 1
state_NY = as.numeric(state_NY)

# state_CA
state_CA <- country_state$state
state_CA <- ifelse(state_CA == "CA", 1, 0)
#state_CA[state_CA != "CA"] <- 0
#state_CA[state_CA == "CA"] <- 1
state_CA = as.numeric(state_CA)

# length_des
length_des <- nchar(df$description)
length_des[is.na(length_des)] <- 0

# length_ben
length_ben <- nchar(df$benefits)
length_ben[is.na(length_ben)] <- 0

# contain_email
contain_email <- str_extract(df$company_profile, "#PHONE_(.*)#")
contain_email[is.na(contain_email)] <- 0
contain_email[contain_email != "0"] <- 1
contain_email = as.numeric(contain_email)

# length_req
length_req <- nchar(df$requirements)
length_req[is.na(length_req)] <- 0

# contain_phone
contain_phone <- str_extract(df$company_profile, "#EMAIL_(.*)#")
contain_phone[is.na(contain_phone)] <- 0
contain_phone[contain_phone != "0"] <- 1
contain_phone = as.numeric(contain_phone)

# has_salary
has_salary <- ifelse(is.na(df$salary_range), 0, 1)

oil_ind <- ifelse(df$industry == "Oil & Energy", 1, 0)
oil_ind[is.na(oil_ind)] <- 0

hos_ind <- ifelse(df$industry == "Hospital & Health Care", 1, 0)
hos_ind[is.na(hos_ind)] <- 0
```

```r
  acc_ind <- ifelse(df$industry == "Accounting", 1, 0)
  acc_ind[is.na(acc_ind)] <- 0

  oil_dept <- ifelse(df$department == "Oil & Energy", 1, 0)
  oil_dept[is.na(oil_dept)] <- 0

  length_profile <- nchar(df$company_profile)
  length_profile[is.na(length_profile)] <- 0

  eng_dept <- ifelse(df$department == "Engineering", 1, 0)
  eng_dept[is.na(eng_dept)] <- 0

  #customer_dept <- ifelse(df$department == "Customer Service", 1, 0)
  #customer_dept[is.na(customer_dept)] <- 0

  #clerical_dept <- ifelse(df$department =="Clerical", 1, 0)
  #clerical_dept[is.na(clerical_dept)] <- 0

  #acc_dept <- ifelse(df$department == "Account", 1, 0)
  #acc_dept[is.na(acc_dept)] <- 0

  #admin_dept <- ifelse(df$department == "admin", 1, 0)
  #admin_dept[is.na(admin_dept)] <- 0

  # uppercase_des
  upper_des <- lengths(str_extract_all(df$description, "[A-Z]{3,}+"))
  upper_des[is.na(upper_des)] <- 0


# uppercase_req
  upper_req <- lengths(str_extract_all(df$requirements, "[A-Z]{3,}+"))
  upper_req[is.na(upper_req)] <- 0


# uppercase_ben
  upper_ben <- lengths(str_extract_all(df$benefits, "[A-Z]{3,}+"))
  upper_ben[is.na(upper_ben)] <- 0

# star_des
  star_des <- as.numeric(grepl("*", df$description, fixed = TRUE))
  star_des[is.na(star_des)] <- 0


# star_req
  star_req <- as.numeric(grepl("*", df$requirements, fixed = TRUE))
  star_req[is.na(star_req)] <- 0

# star_ben
  star_ben <- as.numeric(grepl("*", df$benefits, fixed = TRUE))
  star_ben[is.na(star_ben)] <- 0

# na_company
  na_company <- as.numeric(is.na(df$company_profile))
```

```r
  features <- data.frame(state_TX = state_TX,
                         length_des = length_des,
                         length_ben = length_ben,
                         state_NY = state_NY,
                         state_CA = state_CA,
                         contain_email = contain_email,
                         length_req = length_req,
                         contain_phone = contain_phone,
                         has_salary = has_salary,
                         oil_ind = oil_ind,
                         hos_ind = hos_ind,
                         acc_ind = acc_ind,
                         oil_dept = oil_dept,
                         length_profile = length_profile,
                         eng_dept = eng_dept,
                         upper_des = upper_des,
                         upper_req = upper_req,
                         upper_ben = upper_ben
                         )

  return(features)

}

# power features
power_features <- make_power_features(job)

final_features = cbind(subset(features, select = ols_model$predictors), power_features)

# Split the dataset into train and test
final_features$fraudulent <- job$fraudulent

set.seed(12345)

split <- sample(c(TRUE, FALSE), nrow(final_features), replace=TRUE, prob=c(0.8, 0.2))
job_train <-  final_features[split,]
job_test <- final_features[!split, ]

job_train
job_test


job_train$fraudulent = as.factor(job_train$fraudulent)
job_test$fraudulent = as.factor(job_test$fraudulent)

final_features$fraudulent <- as.factor(job$fraudulent)
dim(final_features)
write.csv(final_features, "final_features.csv")

ff = subset(final_features, select = -fraudulent) %>% scale()

ncol <- ncol(ff)
```

```r
# Store variable names
var <- list()
for(i in 1:ncol){
   var[[i]] <- names(ff)[i]
}
names(ff)[1:ncol] <- paste("var", 1:ncol, sep="")


######clustering
### Hierarchical Clustering
## Determine the number of clusters(after scale)

# Elbow method
fviz_nbclust(ff, kmeans, method = "wss") +
    geom_vline(xintercept = 4, linetype = 2)+
  labs(subtitle = "Elbow method")

# Silhouette method
fviz_nbclust(ff, kmeans, method = "silhouette")+
  labs(subtitle = "Silhouette method")


 set.seed(123)
final = kmeans(ff, 2, nstart =25)
print(final)
fviz_cluster(final, data = ff)

final = kmeans(ff, 3, nstart =25)
print(final)
fviz_cluster(final, data = ff)

final = kmeans(ff, 4, nstart =25)
print(final)
fviz_cluster(final, data = ff)

kmeans = as.data.frame(final$cluster)
colnames(kmeans)[1] = "Cluster"
kmeans = cbind(kmeans, fraudulent = job$fraudulent)

nrow(filter(kmeans, Cluster == 1))
nrow(filter(kmeans, Cluster == 2))
nrow(filter(kmeans, Cluster == 3))
nrow(filter(kmeans, Cluster == 4))

sum(filter(kmeans, Cluster == 1)$fraudulent)
sum(filter(kmeans, Cluster == 1)$fraudulent)/sum(filter(job, fraudulent == 1)$fraudulent)
sum(filter(kmeans, Cluster == 1)$fraudulent)/nrow(filter(databind, Cluster == 1))

sum(filter(kmeans, Cluster == 2)$fraudulent)
sum(filter(kmeans, Cluster == 2)$fraudulent)/sum(filter(job, fraudulent == 1)$fraudulent)
sum(filter(kmeans, Cluster == 2)$fraudulent)/nrow(filter(databind, Cluster == 2))

sum(filter(kmeans, Cluster == 3)$fraudulent)
```

```r
sum(filter(kmeans, Cluster == 3)$fraudulent)/sum(filter(job, fraudulent == 1)$fraudulent)
sum(filter(kmeans, Cluster == 3)$fraudulent)/nrow(filter(databind, Cluster == 3))

sum(filter(kmeans, Cluster == 4)$fraudulent)
sum(filter(kmeans, Cluster == 4)$fraudulent)/sum(filter(job, fraudulent == 1)$fraudulent)
sum(filter(kmeans, Cluster == 4)$fraudulent)/nrow(filter(databind, Cluster == 4))

fraud <- final_features[final_features$fraudulent == 1, ]
real <- final_features[final_features$fraudulent == 0, ]

TX_prop_fraud <- sum(fraud$state_TX) / dim(fraud)[1]
TX_prop_real <- sum(real$state_TX) / dim(real)[1]

CA_prop_fraud <- sum(fraud$state_CA) / dim(fraud)[1]
CA_prop_real <- sum(real$state_CA) / dim(real)[1]

NY_prop_fraud <- sum(fraud$state_NY) / dim(fraud)[1]
NY_prop_real <- sum(real$state_NY) / dim(real)[1]

state_mat <- matrix(c(TX_prop_fraud, TX_prop_real, CA_prop_fraud, CA_prop_real, NY_prop_fraud, NY_prop_r
state_df <- as.data.frame(state_mat)
colnames(state_df) <- c("TX", "CA", "NY")
rownames(state_df) <- c("fraud", "real")
state_mat <- as.matrix(state_df)

barplot(state_mat,
        beside = TRUE,
        main = "State: Fraud vs Real",
        col = c("#F39B7FFF", rgb(0.2, 0.4, 0.6, 0.6)),
        ylim = c(0, 0.3))
legend("topright",
       c("Fraud", "Real"),
       fill = c("#F39B7FFF", rgb(0.2, 0.4, 0.6, 0.6)))
boxplot(final_features$length_req ~ final_features$fraudulent,
        ylim = c(0, 2000),
        main = "Length of Requirement: Fraud vs Real",
        names = c("real", "fraud"),
        ylab = "length of requirement",
        xlab = "",
        col = c("khaki", "honeydew4"))

oil_ind_fraud <- sum(fraud$oil_ind) / dim(fraud)[1]
oil_ind_real <- sum(real$oil_ind) / dim(real)[1]

hos_ind_fraud <- sum(fraud$hos_ind) / dim(fraud)[1]
hos_ind_real <- sum(real$hos_ind) / dim(real)[1]

acc_ind_fraud <- sum(fraud$acc_ind) / dim(fraud)[1]
acc_ind_real <- sum(real$acc_ind) / dim(real)[1]

ind_mat <- matrix(c(oil_ind_fraud, oil_ind_real, hos_ind_fraud, hos_ind_real, acc_ind_fraud, acc_ind_rea
ind_df <- as.data.frame(ind_mat)
colnames(ind_df) <- c("Oil & Energy", "Hospitality", "Accounting")
```

```r
rownames(ind_df) <- c("fraud", "real")
ind_mat <- as.matrix(ind_df)

barplot(ind_mat,
        beside = TRUE,
        main = "Industry: Fraud vs Real",
        col = c("palegreen3", "dodgerblue3"),
        ylim = c(0, 0.15))
legend("topright",
       c("Fraud", "Real"),
       fill = c("palegreen3", "dodgerblue3"))

# SVM
svm_model <- svm(fraudulent~., data=job_train , kernel ="radial", scale=TRUE)
summary(svm_model)

train_pred_svm <- predict(svm_model, subset(job_train, select = -fraudulent))
table(train_pred_svm, job_train$fraudulent)

test_pred_svm <- predict(svm_model, subset(job_test, select = -fraudulent))
table(test_pred_svm, job_test$fraudulent)

# Random Forest Model
rf_model <- randomForest(fraudulent~., data=job_train, ntree=120, mtry = 25, importance =TRUE)

train_pred_rf <- predict(rf_model, subset(job_train, select = -fraudulent))
table(train_pred_rf, job_train$fraudulent)

test_pred_rf <- predict(rf_model ,subset(job_test, select = -fraudulent))
table(test_pred_rf, job_test$fraudulent)

# Random Forest Model
rf_model <- randomForest(fraudulent~., data=final_features, cutoff = c(0.8, 0.2), mtry = 23, importance

saveRDS(rf_model, file = "rf_model1.RDS")
```