Joint EDA and Feature Engineering

Table of Contents

Loading libraries

```
library(fastDummies)
library(stringr)
library(tidytext)
library(dplyr)
library(corrplot)
library(tm)
library(janitor)
library(vtree)
library(ggplot2)
library(reshape2)
library(tidyverse)
library(sns)
library(data.table)
library(caret)
library(corpus)
library(tm)
library(stringi)
library(lattice)
library(wordcloud)
library(gmodels)
library(e1071)
library(SnowballC)
library(gridExtra)
library(stringi)
library(knitr)
library(RWeka)
library(textfeatures)
require(downloader)
library(sqldf)
library(compare)
library(plotrix)
```

Alisha's Feature Engineering Pt 1

Number of NA for each column in our train set

```
na_count <-sapply(df_train, function(y) sum(length(which(is.na(y)))))
(na_count<-data.frame(na_count))

##                        na_count
## title                         0
## location                    285
## department                 9265
## salary_range              12013
## company_profile            2668
## description                   0
## requirements               2178
```

```
## benefits                 5721
## telecommuting               0
## has_company_logo            0
## has_questions               0
## employment_type          2813
## required_experience      5683
## required_education       6526
## industry                 3951
## fn                       5201
## fraudulent                  0
## index                       0
```

Creating Binary Columns if a feature exists or not

For each factor column: 1,0 exists or not. And then see the percent of each that are fraudulent

```
na_col<-function(df,vals,response){
  for(x in vals){
    new_col<-paste("has",x,sep="_")
    df[[new_col]]<-ifelse(is.na(df[[substitute(x)]]), 0, 1)


  }
  return(df)
}

df_train<-na_col(df_train, list("location","department", "salary_range",
"company_profile", "requirements", "benefits", "employment_type",
"required_experience", "required_education", "industry", "fn" ),
"fraudulent")

df_test<-na_col(df_test,list("location","department", "salary_range",
"company_profile", "requirements", "benefits", "employment_type",
"required_experience", "required_education", "industry", "fn" ),
"fraudulent")

#count number of unique
apply(df_train, 2, function(x) length(unique(x)))
```

```
##                title             location           department
##                 9305                 2770                 1162
##         salary_range      company_profile          description
##                  752                 1600                12031
##         requirements             benefits        telecommuting
##                 9707                 5205                    2
##     has_company_logo        has_questions      employment_type
##                    2                    2                    6
##  required_experience   required_education             industry
##                    8                   14                  131
```

```
##                        fn              fraudulent                     index
##                        38                       2                     14304
##              has_location          has_department          has_salary_range
##                         2                       2                         2
##       has_company_profile        has_requirements              has_benefits
##                         2                       2                         2
##       has_employment_type has_required_experience  has_required_education
##                         2                       2                         2
##              has_industry                  has_fn
##                         2                       2
```

Creating text features for each text column

```
text_cols<-function(df,vals){
  for (x in vals){
    print(x)
    #df[[substitute(x)]]<-as.character(df[[substitute(x)]])#

    features<-
textfeatures(as.character(df[[substitute(x)]])%>%replace_na(''),normalize=FAL
SE)
    print("done with 1")
    colnames(features)<-paste(x,colnames(features),sep="_")
    print("done with 2")
    df<-cbind(df, features[c(1:34)])
  }
  return(df)
}

df_train<-text_cols(df_train, list("department", "company_profile",
"description", "requirements", "benefits"))

## [1] "department"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✔[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
## [1] "company_profile"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✔[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
## [1] "description"
## [32m↪[39m [38;5;244mCounting features in text...[39m
```

```
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✓[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
## [1] "requirements"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✓[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
## [1] "benefits"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✓[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
```

```r
df_test<-text_cols(df_test, list("department", "company_profile",
"description", "requirements", "benefits"))
```

```
## [1] "department"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✓[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
## [1] "company_profile"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✓[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
## [1] "description"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✓[39m Job's done!
## [1] "done with 1"
```

```
## [1] "done with 2"
## [1] "requirements"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✓[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
## [1] "benefits"
## [32m↪[39m [38;5;244mCounting features in text...[39m
## [32m↪[39m [38;5;244mSentiment analysis...[39m
## [32m↪[39m [38;5;244mParts of speech...[39m
## [32m↪[39m [38;5;244mWord dimensions started[39m
## [32m✓[39m Job's done!
## [1] "done with 1"
## [1] "done with 2"
```

Anette's EDA and Feature Engineering

```
anette_df <- df_train[c('location', 'company_profile', 'benefits',
'has_company_logo', 'required_education', 'fraudulent')]

anette_test_df <- df_test[c('location', 'company_profile', 'benefits',
'has_company_logo', 'required_education', 'fraudulent')]
```

Mode function

```
get_mode <- function(x) {
  unique_x <- unique(x)
  tabulate_x <- tabulate(match(x, unique_x))
  unique_x[tabulate_x == max(tabulate_x)]
}
```

Location

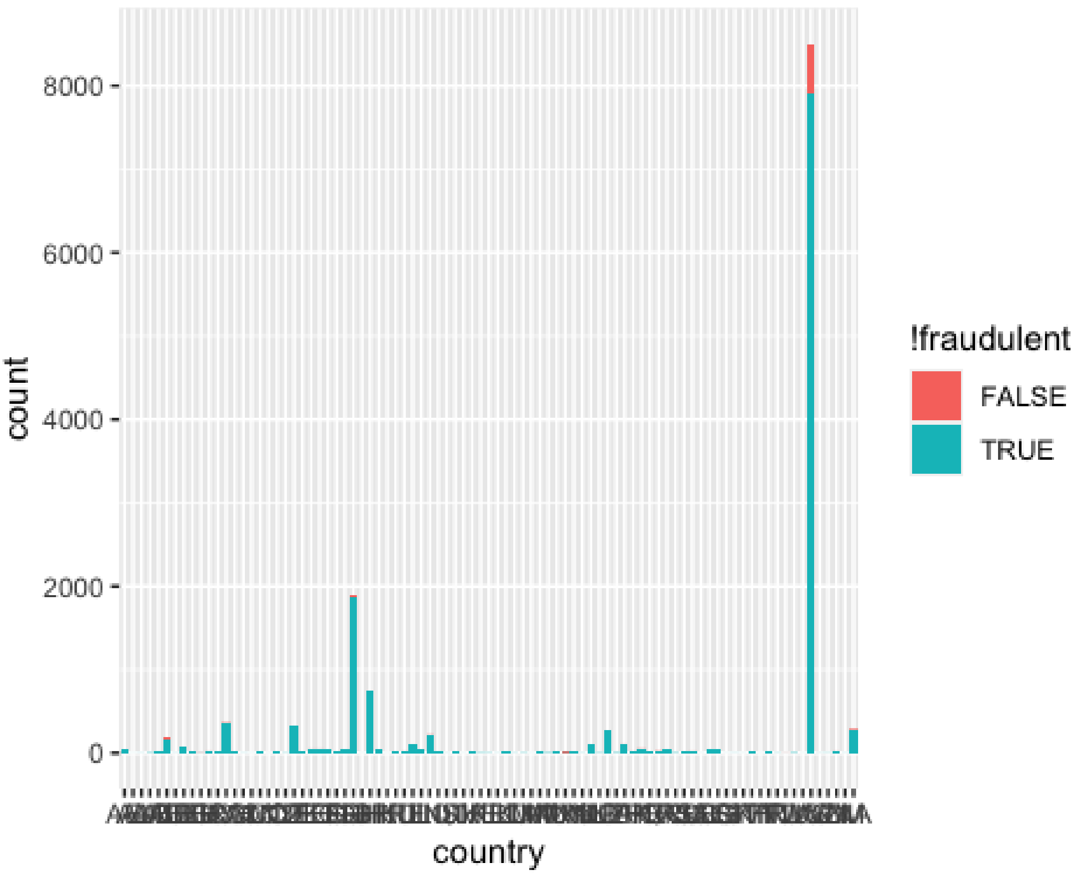
```
# splitting location
anette_df[c('country', 'state', 'city')] <-
str_split_fixed(anette_df$location, ',', 3)
anette_df = subset(anette_df, select = -c(location))

#originally just blanks filled with NA
anette_df[anette_df == "" | anette_df == " "] <- NA  # Replace blank & space
by NA

# created a seperate location dataset to play with
location <- anette_df[c('country', 'state', 'city', 'fraudulent')]
usa <- anette_df[anette_df$country == "US", ]

#take out white spaces out of state
```

```r
anette_df$state<- trimws(anette_df$state, which = c("left"))

# splitting location
anette_test_df[c('country', 'state', 'city')] <-
str_split_fixed(anette_test_df$location, ',', 3)
anette_test_df = subset(anette_test_df, select = -c(location))

#originally just blanks filled with NA
anette_test_df[anette_test_df == "" | anette_test_df == " "] <- NA  # Replace
blank & space by NA

# created a seperate location dataset to play with
location <- anette_test_df[c('country', 'state', 'city', 'fraudulent')]
usa <- anette_test_df[anette_test_df$country == "US", ]

#take out white spaces out of state
anette_test_df$state<- trimws(anette_test_df$state, which = c("left"))
```

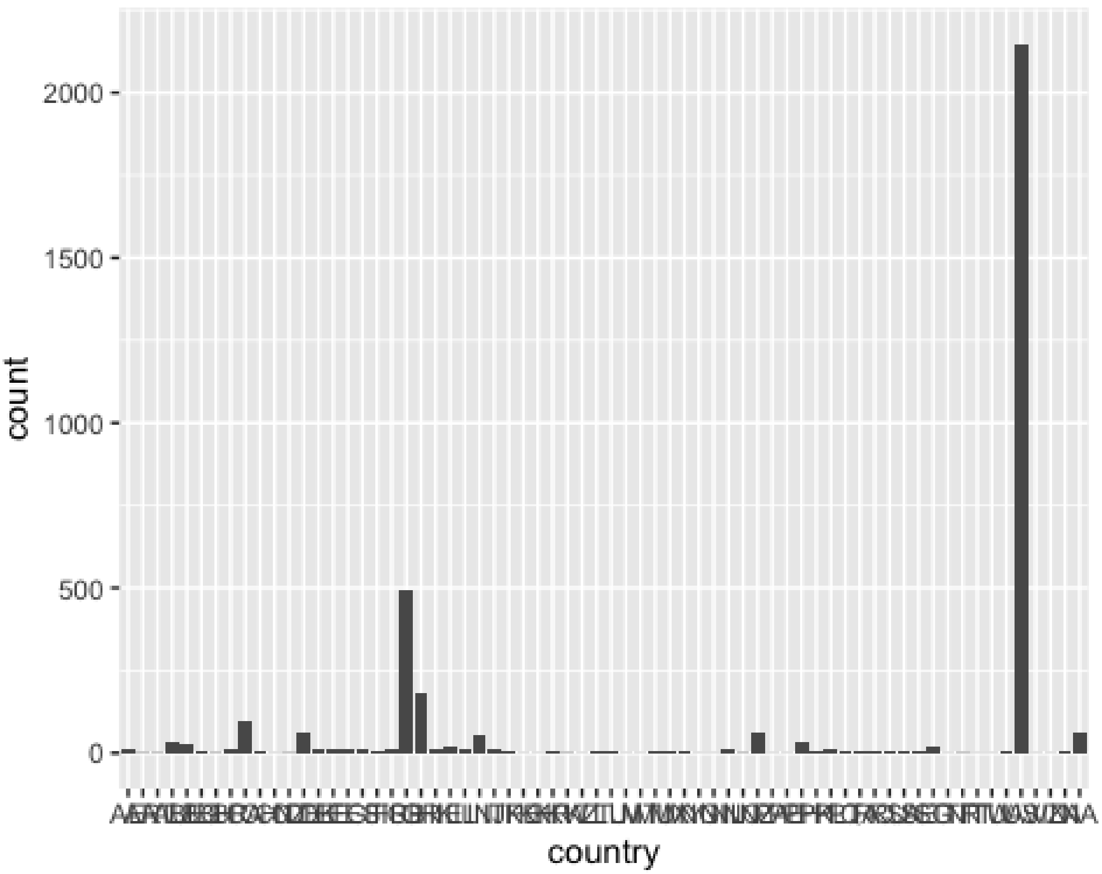
EDA for Country

```r
anette_df$YNusa<- ifelse(anette_df$country %in% "US", 1, 0)
anette_test_df$YNusa<- ifelse(anette_test_df$country %in% "US", 1, 0)

#
#agg_country_total <- location%>% group_by(country) %>%
#  summarise(sum = sum(country),
#            .groups = 'drop')

#mode of country
get_mode(anette_df$country)

## [1] "US"

# Summary of which countries are correlated with fraud
#doing this with mode
agg_country_mode <- anette_df %>% group_by(country) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

##view this model (sorted)
#view(agg_country_mode[order(agg_country_mode$fraudulent, decreasing = TRUE),
])


#doing this with mean
agg_country_mean<- anette_df %>% group_by(country) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')

#view(agg_country_mean[order(agg_country_mean$fraudulent, decreasing = TRUE),
```

```r
])


#Grouping Countries by Fraud and notFraud
# Sum for each country of notFraud
agg_country_notfraud<-location%>% group_by(country) %>%
  summarise(sum_notfraud = sum(fraudulent == '0'),
            .groups = 'drop')

#view(agg_country_notfraud[order(agg_country_notfraud$sum_notfraud,
decreasing = TRUE), ])


# group by country and sum of fraud
#sum of each country fraud
agg_country_fraud<-location%>% group_by(country) %>%
  summarise(sum_fraud = sum(fraudulent == '1'),
            .groups = 'drop')

#view(agg_country_fraud[order(agg_country_fraud$sum_fraud, decreasing =
TRUE), ])

#plotting Country
ggplot(anette_df, aes(x = country, fill =  !fraudulent))+
  geom_bar(stat = "count")
```




```r
#plotting Country
ggplot(data = location) +
  geom_bar(mapping = aes(x = country))
```




```
# creating a numeric set of countries in case we want it
anette_df$country_num=anette_df$country
anette_df$country_num<-factor(anette_df$country_num)
anette_df$country_num<-unclass(anette_df$country_num)
```

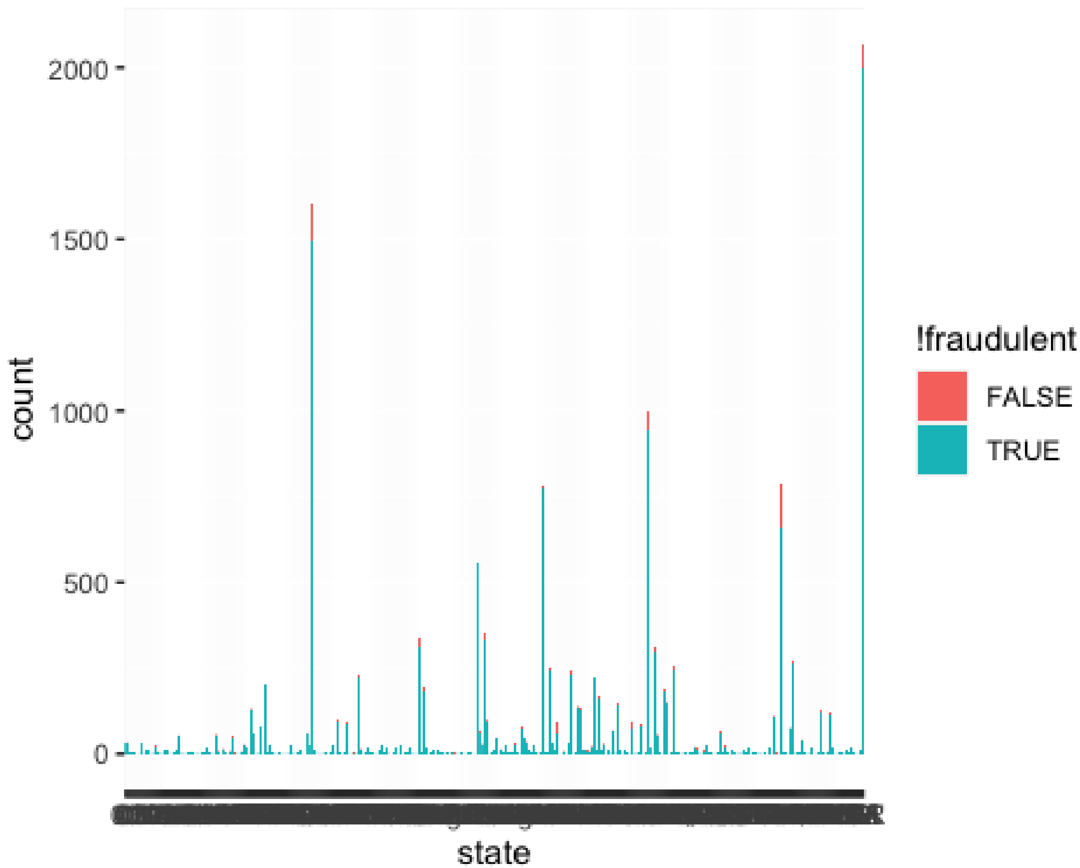
Dealing with NA's for country

```
# replacing NA's with US (mode)
data2 <- anette_df
data2$country<-anette_df$country %>% replace_na('US')

#same summaries above but replacing NA's with mode
#data but grouping by mode
agg_country_mode_replace_NAmode<-data2 %>% group_by(country) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

##viewing the model above, when NA's are replace by the mode of the original
dataset. This is displayed by mode
#view(agg_country_mode_replace_NAmode[order(agg_country_mode_replace_NAmode$f
raudulent, decreasing = TRUE), ])
```

```
#data but grouping by mean
agg_country_mean_replace_NAmode<-data2 %>% group_by(country) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')

##viewing the model above, when NA's are replace by the mode of the original
dataset. This is displayed by mean
#view(agg_country_mean_replace_NAmode[order(agg_country_mean_replace_NAmode$f
raudulent, decreasing = TRUE), ])



#replacing NA's with new category called NA (this is the same as original
analysis)
data3<-anette_df
data3$country<-data3$country %>% replace_na('MIS')

agg_country_mode_replace_NAmis<- data3%>% group_by(country) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

#doing this with mean
agg_country_mean_replace_NAmis<-data3 %>% group_by(country) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')
```

EDA for State

```
#this first analysis is just from the main dataset
#model for just state based off of mean
agg_state_mean <- anette_df %>% group_by(state) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')

#view(agg_state_mean[order(agg_state_mean$fraudulent, decreasing = TRUE), ])

#model for just state based off of mode
agg_state_mode <- anette_df %>% group_by(state) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

#view(agg_state_mode[order(agg_state_mode$fraudulent, decreasing = TRUE), ])


#plotting state
ggplot(anette_df, aes(x = state, fill =  !fraudulent))+
  geom_bar(stat = "count")
```




```
#this is from just the unites states dataset
#model for just state based off of mean
agg_state_mean_us <- usa %>% group_by(state) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')

#view(agg_state_mean_us[order(agg_state_mean_us$fraudulent, decreasing =
TRUE), ])

#model for just state based off of mode
agg_state_mode_us <- usa %>% group_by(state) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

#view(agg_state_mode_us[order(agg_state_mode_us$fraudulent, decreasing =
TRUE), ])


#Grouping states(from usa) by Fraud and notFraud
# Sum for each state of notFraud
agg_country_notfraud_usa <- usa %>% group_by(state) %>%
  summarise(sum_notfraud = sum(fraudulent == '0'),
            .groups = 'drop')
```

```r
#view(agg_country_notfraud_usa[order(agg_country_notfraud_usa$sum_notfraud,
decreasing = TRUE), ])


# group by country and sum of fraud
#sum of each state fraud
agg_country_fraud_usa <- usa %>% group_by(state) %>%
  summarise(sum_fraud = sum(fraudulent == '1'),
            .groups = 'drop')

#view(agg_country_fraud_usa[order(agg_country_fraud_usa$sum_fraud, decreasing
= TRUE), ])

# I do not believe this is as useful at the region one since there are more
states than regions


# I believe this model is very interesting
agg_state_mode <- anette_df %>% group_by(state, country) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

#view(agg_state_mode[order(agg_state_mode$fraudulent, decreasing = TRUE), ])
```

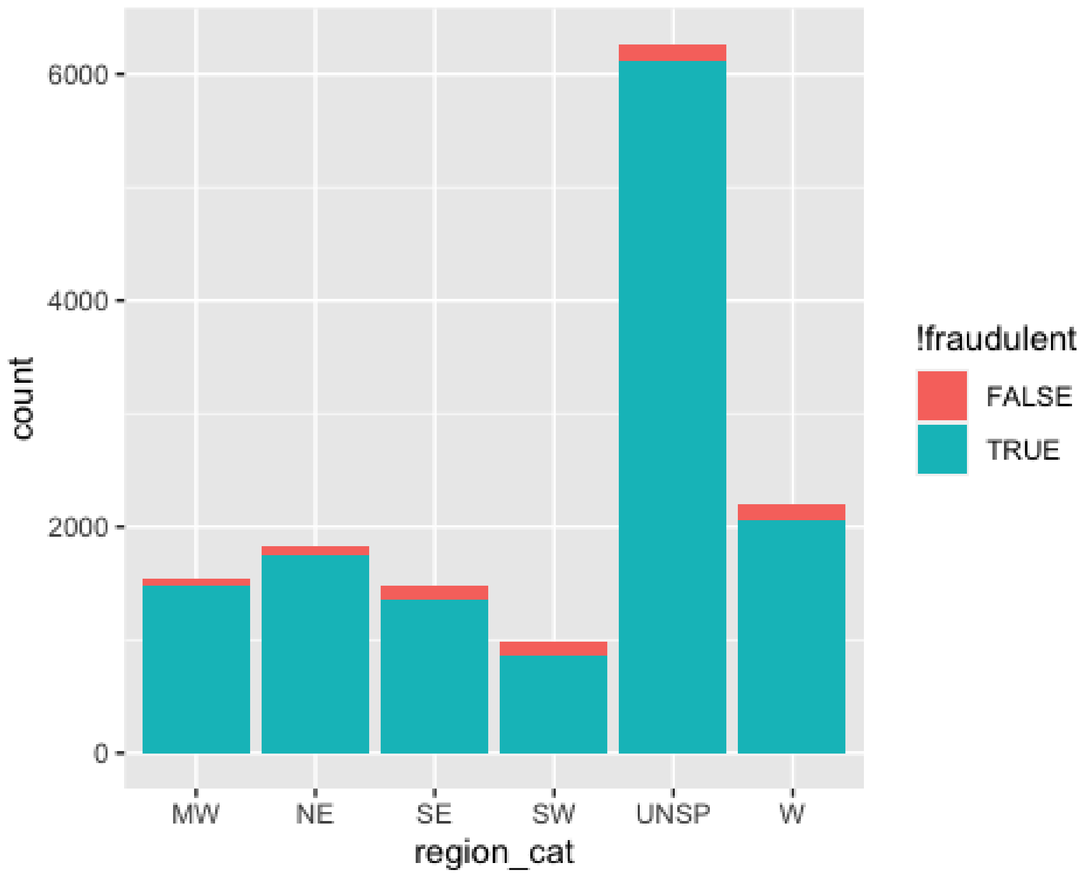
Dealing with NA's for state? –> make them into MISS (missing) or UNSP (unspecified)

Making a column named region based off of State

```r
#creating a new column called region which does regions based off of state
(in the US)
anette_df<-anette_df %>% mutate(region_cat = case_when(state == 'IL' ~ "MW",
                          state == 'IN' ~ "MW", state == 'WI' ~ "MW",
                          state == 'MI' ~ "MW", state == 'OH' ~ "MW",
                          state == 'MO' ~ "MW", state == 'KS' ~ "MW",
                          state == 'NE' ~ "MW", state == 'SD' ~ "MW",
                          state == 'ND' ~ "MW", state == 'MN' ~ "MW",
                          state == 'IA' ~ "MW",

                          state == 'ME' ~ "NE", state == 'VT' ~ "NE",
                          state == 'MA' ~ "NE", state == 'RI' ~ "NE",
                          state == 'CT' ~ "NE", state == 'NY' ~ "NE",
                          state == 'NJ' ~ "NE", state == 'PA' ~ "NE",
                          state == 'NH' ~ "NE",

                          state == 'DE' ~ "SE", state == 'MD' ~ "SE",
                          state == 'WV' ~ "SE", state == 'VA' ~ "SE",
                          state == 'NC' ~ "SE", state == 'SC' ~ "SE",
                          state == 'GA' ~ "SE", state == 'AL' ~ "SE",
                          state == 'FL' ~ "SE", state == 'MS' ~ "SE",
```

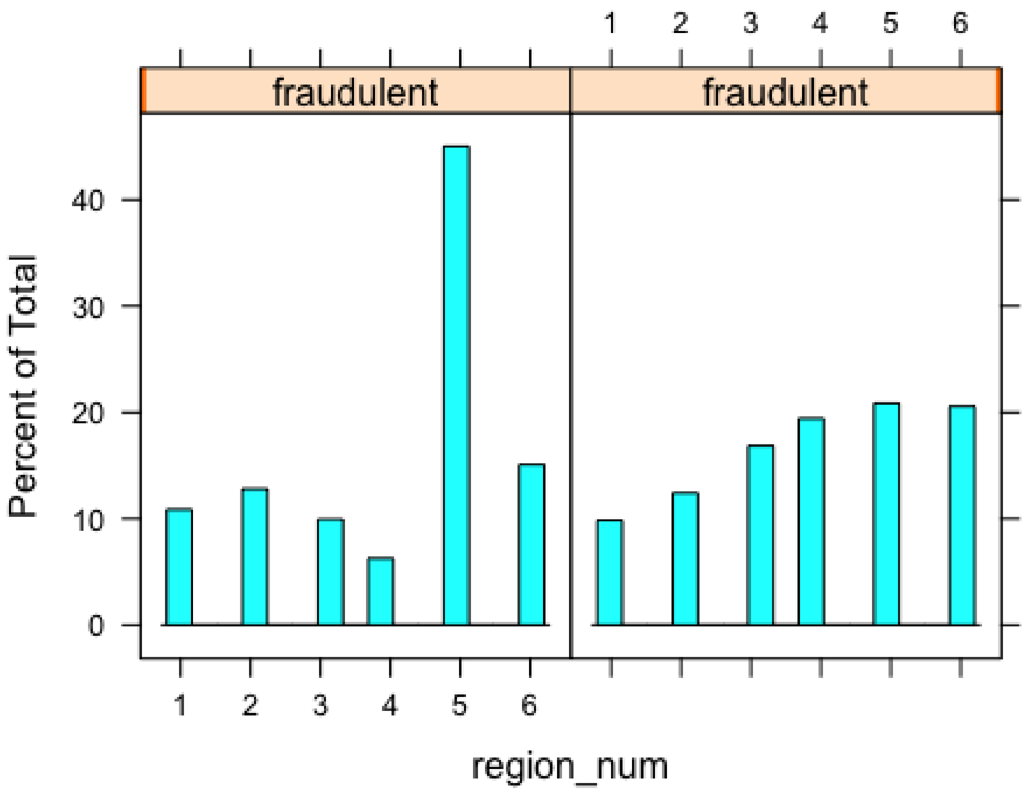
```r
                          state == 'TN' ~ "SE", state == 'KY' ~ "SE",
                          state == 'LA' ~ "SE", state == 'AR' ~ "SE",

                          state == 'AZ' ~ "SW", state == 'NM' ~ "SW",
                          state == 'OK' ~ "SW", state == 'TX' ~ "SW",

                          state == 'WA' ~ "W", state == 'MT' ~ "W",
                          state == 'OR' ~ "W", state == 'ID' ~ "W",
                          state == 'WY' ~ "W", state == 'CA' ~ "W",
                          state == 'NV' ~ "W", state == 'UT' ~ "W",
                          state == 'CO' ~ "W", state == 'HI' ~ "W",
                          state == 'AK' ~ "W"))

# to deal with NA's I created a seperate variable called UNSP (unspecified)
anette_df$region_cat<- anette_df$region_cat %>% replace_na('UNSP')




anette_test_df<-anette_test_df %>% mutate(region_cat = case_when(state ==
'IL' ~ "MW",
                          state == 'IN' ~ "MW", state == 'WI' ~ "MW",
                          state == 'MI' ~ "MW", state == 'OH' ~ "MW",
                          state == 'MO' ~ "MW", state == 'KS' ~ "MW",
                          state == 'NE' ~ "MW", state == 'SD' ~ "MW",
                          state == 'ND' ~ "MW", state == 'MN' ~ "MW",
                          state == 'IA' ~ "MW",

                          state == 'ME' ~ "NE", state == 'VT' ~ "NE",
                          state == 'MA' ~ "NE", state == 'RI' ~ "NE",
                          state == 'CT' ~ "NE", state == 'NY' ~ "NE",
                          state == 'NJ' ~ "NE", state == 'PA' ~ "NE",
                          state == 'NH' ~ "NE",

                          state == 'DE' ~ "SE", state == 'MD' ~ "SE",
                          state == 'WV' ~ "SE", state == 'VA' ~ "SE",
                          state == 'NC' ~ "SE", state == 'SC' ~ "SE",
                          state == 'GA' ~ "SE", state == 'AL' ~ "SE",
                          state == 'FL' ~ "SE", state == 'MS' ~ "SE",
                          state == 'TN' ~ "SE", state == 'KY' ~ "SE",
                          state == 'LA' ~ "SE", state == 'AR' ~ "SE",

                          state == 'AZ' ~ "SW", state == 'NM' ~ "SW",
                          state == 'OK' ~ "SW", state == 'TX' ~ "SW",

                          state == 'WA' ~ "W", state == 'MT' ~ "W",
                          state == 'OR' ~ "W", state == 'ID' ~ "W",
                          state == 'WY' ~ "W", state == 'CA' ~ "W",
```

```
                          state == 'NV' ~ "W", state == 'UT' ~ "W",
                          state == 'CO' ~ "W", state == 'HI' ~ "W",
                          state == 'AK' ~ "W"))

# to deal with NA's I created a seperate variable called UNSP (unspecified)
anette_test_df$region_cat<- anette_test_df$region_cat %>% replace_na('UNSP')





#Grouping states(from usa) by Fraud and notFraud
# Sum for each state of notFraud
agg_country_notfraud_region <- anette_df %>% group_by(region_cat) %>%
  summarise(sum_notfraud = sum(fraudulent == '0'),
            .groups = 'drop')

#view(agg_country_notfraud_region[order(agg_country_notfraud_region$sum_notfr
aud, decreasing = TRUE), ])

# group by country and sum of fraud
#sum of each state fraud
agg_country_fraud_region<- anette_df %>% group_by(region_cat) %>%
  summarise(sum_fraud = sum(fraudulent == '1'),
            .groups = 'drop')

#view(agg_country_fraud_region[order(agg_country_fraud_region$sum_fraud,
decreasing = TRUE), ])

#plotting Region
ggplot(anette_df, aes(x = region_cat, fill =  !fraudulent))+
  geom_bar(stat = "count")
```




```r
#create another region column which is numeric
anette_df$region_num= anette_df$region_cat
anette_df$region_num<- factor(anette_df$region_num)
anette_df$region_num<- unclass(anette_df$region_num)

#doing the same thing for the numeric
#Grouping states(from usa) by Fraud and notFraud
# Sum for each state of notFraud
agg_country_notfraud_region<- anette_df %>% group_by(region_num) %>%
  summarise(sum_notfraud = sum(fraudulent == '0'),
            .groups = 'drop')

#view(agg_country_notfraud_region[order(agg_country_notfraud_region$sum_notfr
aud, decreasing = TRUE), ])


# group by country and sum of fraud
#sum of each state fraud
agg_country_fraud_region<-anette_df %>% group_by(region_num) %>%
  summarise(sum_fraud = sum(fraudulent == '1'),
            .groups = 'drop')

#view(agg_country_fraud_region[order(agg_country_fraud_region$sum_fraud,
```

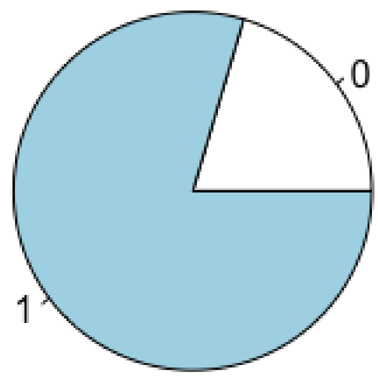
```
decreasing = TRUE), ])

#another type of visual histogram of this
histogram(~region_num | fraudulent, data = anette_df)
```



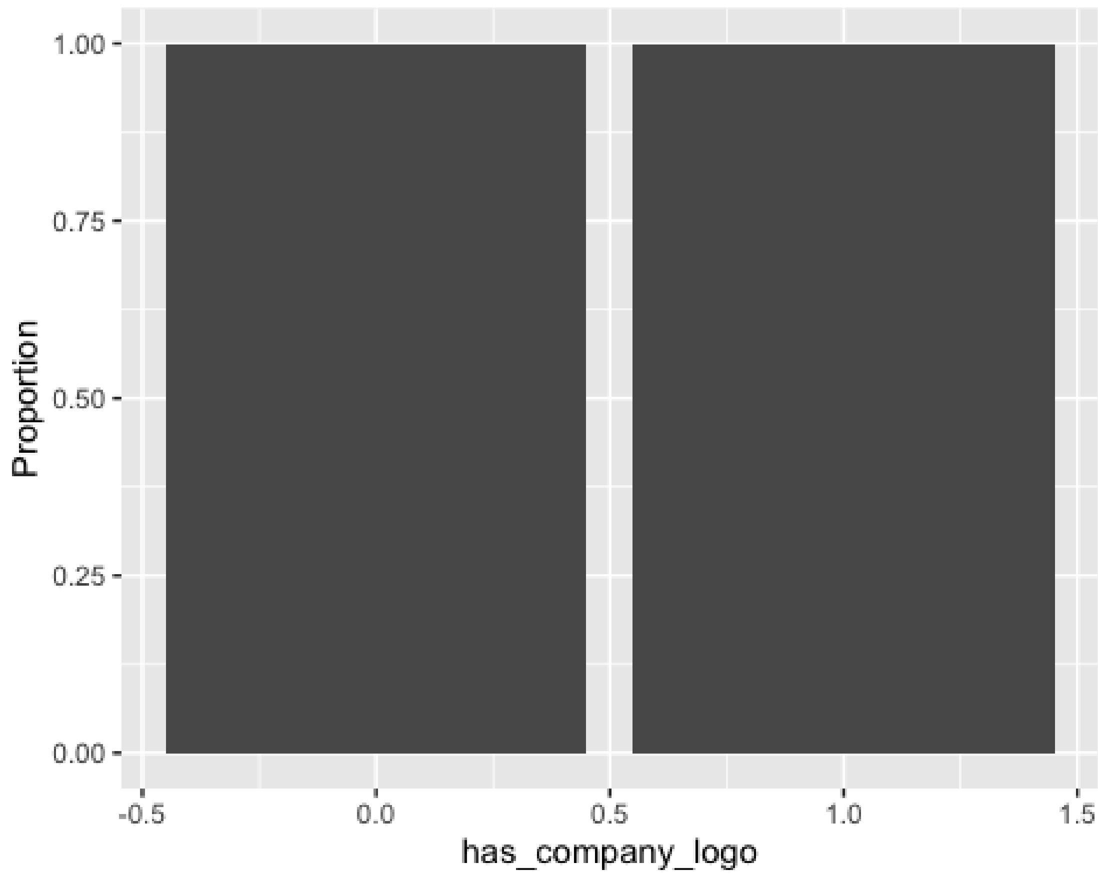
has_company_logo

```
# does not have any NA
table(df_train['has_company_logo'])

##
##     0     1
##  2920 11384

sum(is.na(anette_df$has_company_logo))

## [1] 0

#doing this with mode
agg_hasLogo_mode<- anette_df %>% group_by(has_company_logo) %>%
  summarise(fraudulent = get_mode(fraudulent),
            .groups = 'drop')

#view(agg_hasLogo_mode[order(agg_hasLogo_mode$fraudulent, decreasing = TRUE),
])
```

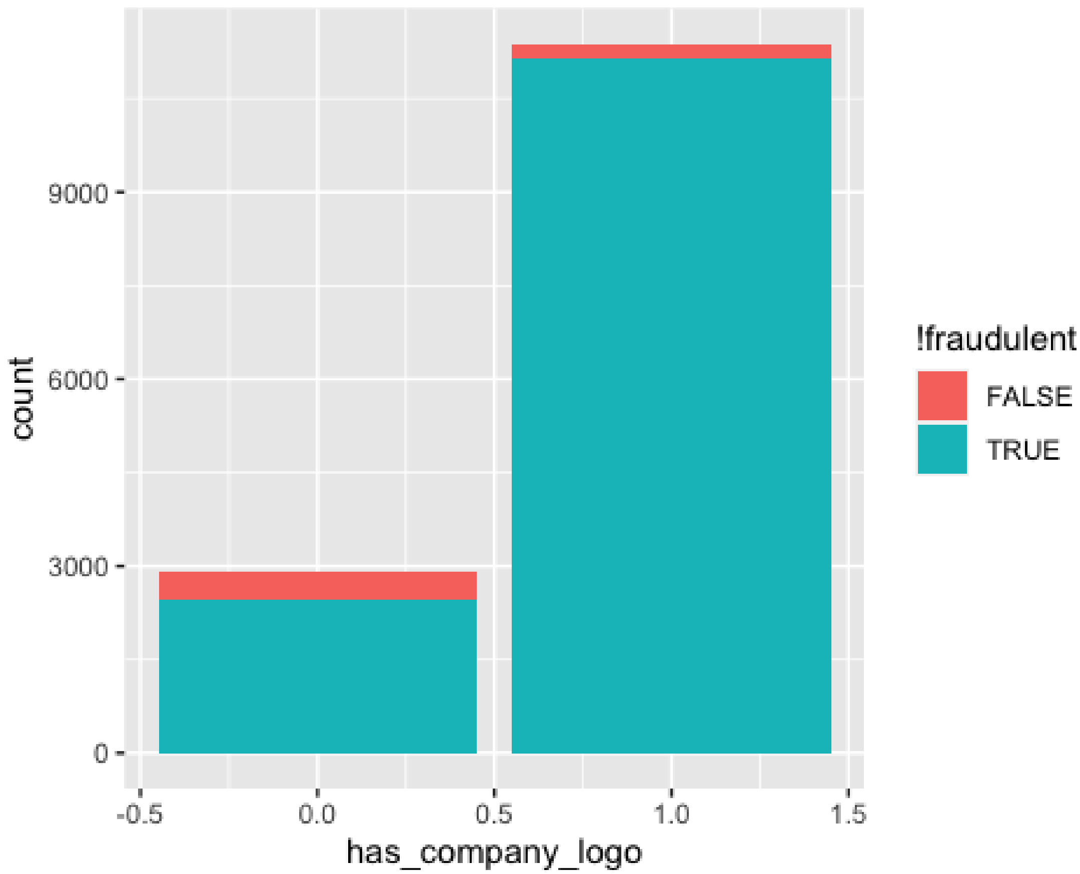
```
#doing this with mean
agg_hasLogo_mean<- anette_df %>% group_by(has_company_logo) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')

#view(agg_hasLogo_mean[order(agg_hasLogo_mean$fraudulent, decreasing = TRUE),
])


# Has company logo and is fraudulent
nrow(anette_df[anette_df$has_company_logo == '1' & anette_df$fraudulent ==
'1', ])

## [1] 229

# Has company logo and isn't fraudulent
nrow(anette_df[anette_df$has_company_logo == '1' & anette_df$fraudulent ==
'0', ])

## [1] 11155

# Doesn't have company logo and is fraudulent
nrow(anette_df[anette_df$has_company_logo == '0' & anette_df$fraudulent ==
'1', ])

## [1] 471

# Doesn't have company logo and isn't fraudulent
nrow(anette_df[anette_df$has_company_logo == '0' & anette_df$fraudulent ==
'0', ])

## [1] 2449

#counts
(typeCounts<- table(anette_df$has_company_logo))

##
##     0     1
##  2920 11384

#percents
prop.table(typeCounts)

##
##         0         1
## 0.2041387 0.7958613

#display
pie(typeCounts)
```



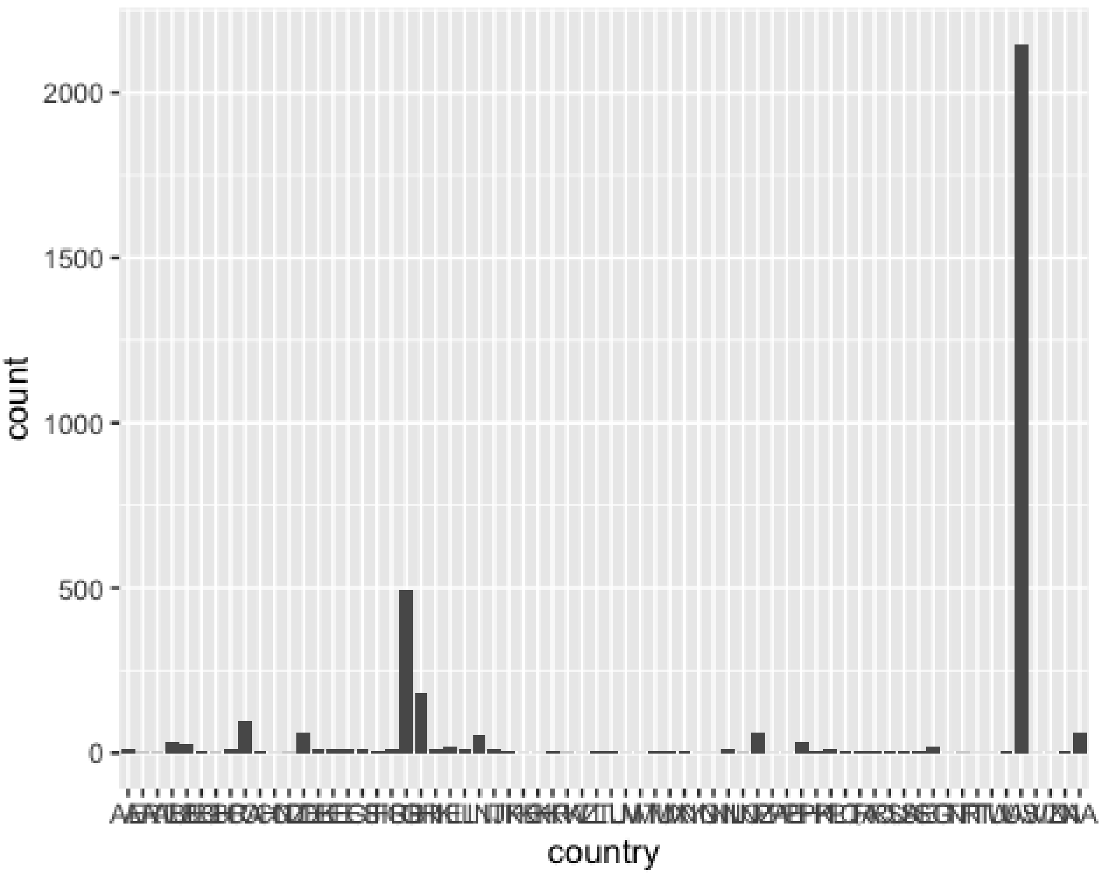

```
ggplot(anette_df,aes(x=has_company_logo,fill=fraudulent))+geom_bar(position="
fill")+labs(y="Proportion")

## Warning: The following aesthetics were dropped during statistical
transformation: fill
## ℹ This can happen when ggplot fails to infer the correct grouping
structure in
##   the data.
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```




```
#plotting Country
ggplot(anette_df, aes(x = has_company_logo, fill =  !fraudulent))+
  geom_bar(stat = "count")
```




```
#plotting Country
ggplot(data = location) +
  geom_bar(mapping = aes(x = country))
```

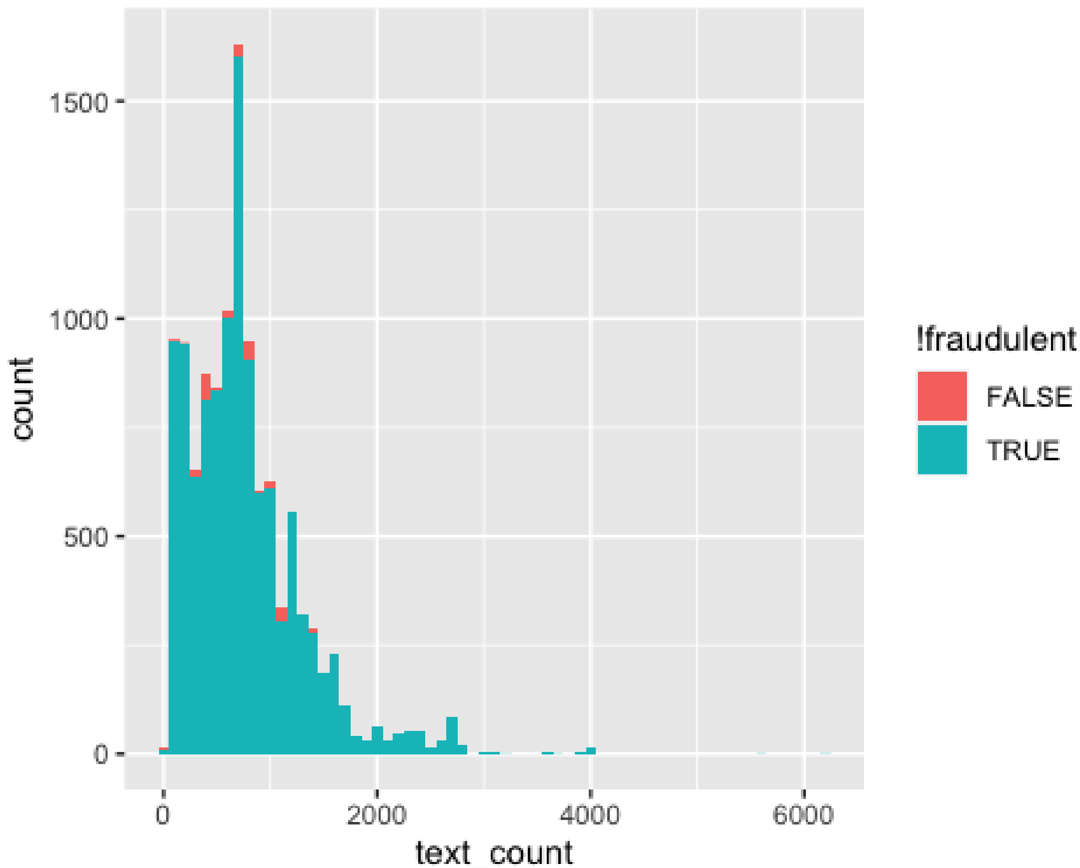



```
# creating a numeric set of countries in case we want it
anette_df$country_num= anette_df$country
anette_df$country_num<- factor(anette_df$country_num)
anette_df$country_num<- unclass(anette_df$country_num)

###########################################################################
#
# need to drop NA's before running the models
##########################################################################
data1 <- anette_df
sample <- sample(c(TRUE, FALSE), nrow(data1), replace=TRUE, prob=c(0.8,0.2))
train <- data1[sample, ]
test <- data1[!sample, ]
model <- glm(fraudulent~has_company_logo, family="binomial", data=train)
#use model to predict probability of default
predicted <- predict(model, test, type="response")
```

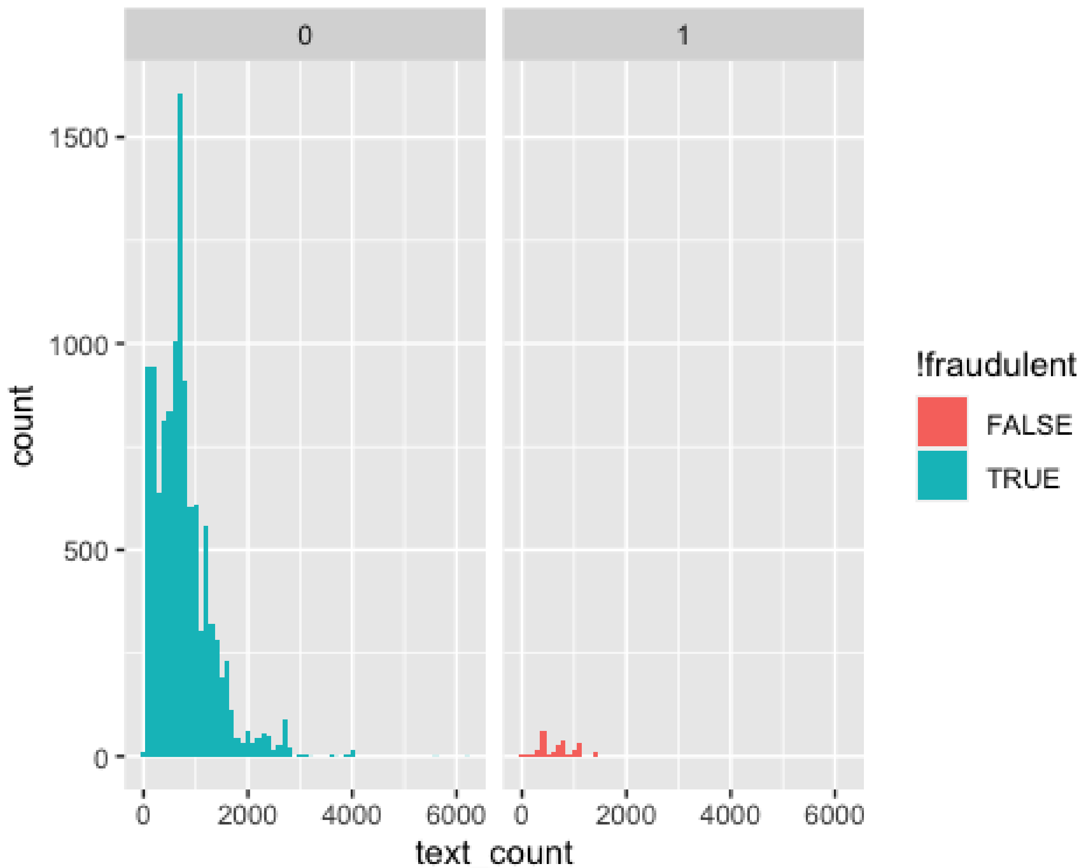
Company profile

```
#Analysis
#head(anette_df$company_profile)
text_count<- str_count(anette_df$company_profile)
anette_df<- cbind(anette_df, text_count)
myvars<- c("company_profile", "text_count", "fraudulent")
```

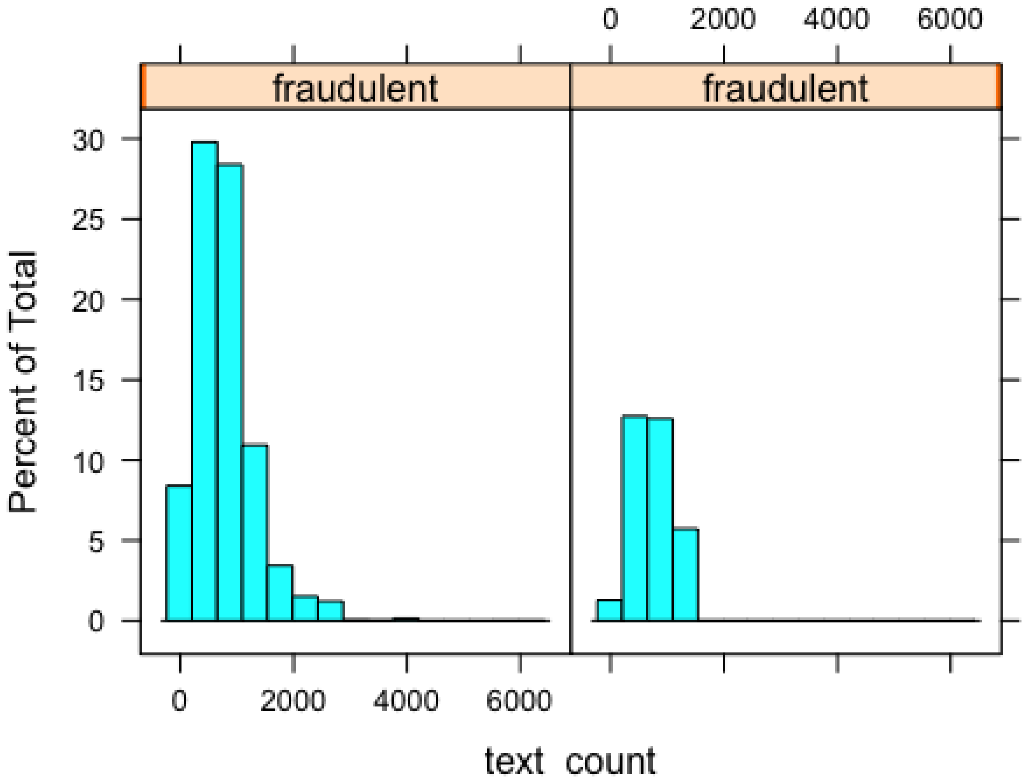
```r
copy<- anette_df[myvars]
#copy%>% arrange(desc(text_count))
#copy%>% arrange(text_count)

sum(is.na(anette_df$company_profile))

## [1] 2668

#anette_df %>% dplyr::mutate(company_profile = replace_na(company_profile,
"NAVALUES"))

anette_df$company_profile<-as.character(anette_df$company_profile)
anette_df$company_profile<- anette_df$company_profile %>% replace_na("UNSP")
sum(is.na(anette_df$company_profile))

## [1] 0

anette_df$company_profile<-as.factor(anette_df$company_profile)
anette_df = subset(anette_df, select = -c(text_count))

# a histogram of text length and visual comparison to fraud and not fraud
#layered
ggplot(anette_df, aes(text_count, fill = !fraudulent)) +
geom_histogram(binwidth = 100)

## Warning: Removed 2668 rows containing non-finite values (`stat_bin()`).
```




```
#unlayered
ggplot(anette_df, aes(text_count, fill = !fraudulent)) +
geom_histogram(binwidth = 100) + facet_wrap(~fraudulent)

## Warning: Removed 2668 rows containing non-finite values (`stat_bin()`).
```




```
#another type of visual histogram of this
histogram(~text_count | fraudulent, data = anette_df)
```



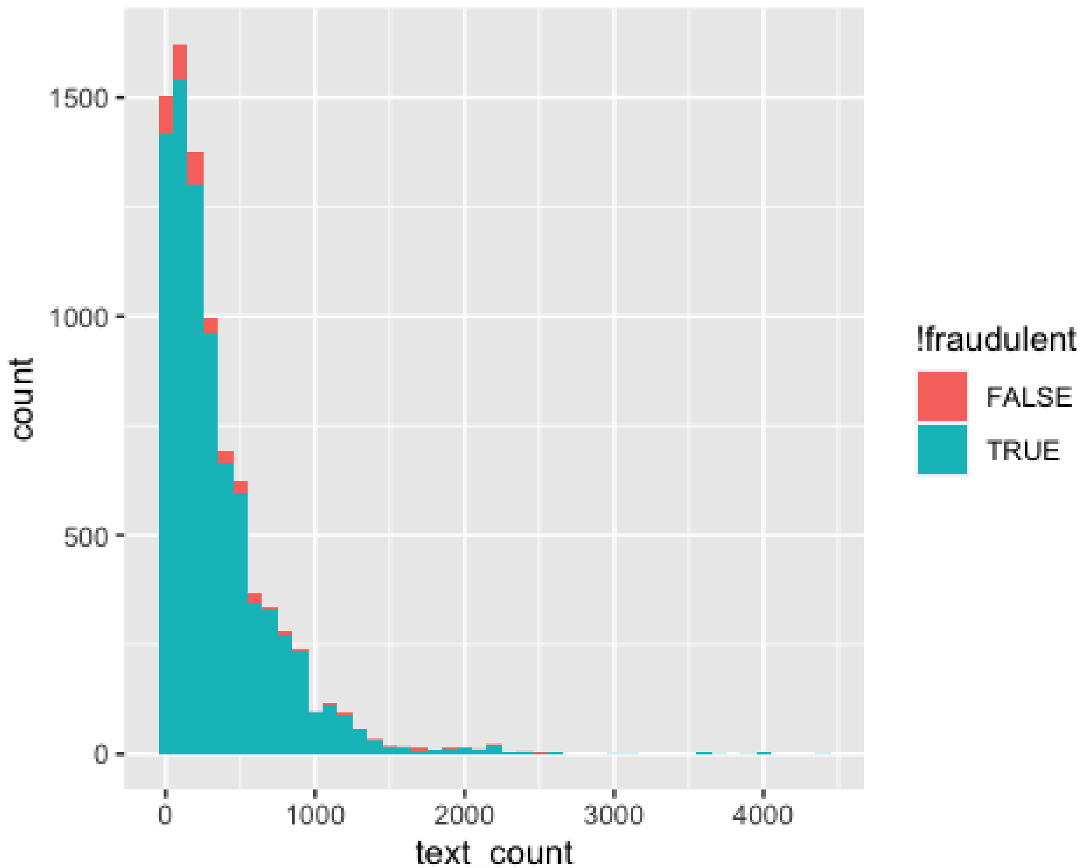

```
sms_corpus<- VCorpus(VectorSource(anette_df$company_profile))
#sms_corpus_clean <- tm_map(sms_corpus, tolower) #all letters to lowercase
sms_corpus_clean<- tm_map(sms_corpus, content_transformer(tolower))
sms_corpus_clean<- tm_map(sms_corpus_clean, removeNumbers) #removes numbers
sms_corpus_clean<- tm_map(sms_corpus_clean, removePunctuation) #removes
punctuation
sms_corpus_clean<- tm_map(sms_corpus_clean, removeWords, stopwords())
#sms_corpus_clean <- tm_map(sms_corpus_clean, removeWords, stopwords('â'))
sms_corpus_clean<- tm_map(sms_corpus_clean, stripWhitespace)

#look at the cleaned corpus
#inspect(sms_corpus_clean[1:3])

#document term matrix used for analyzing tokenization
sms_dtm<- DocumentTermMatrix(sms_corpus_clean)

# displaying frequent terms in a wordcloud
wordcloud(sms_corpus_clean, min.freq = 1000, scale=c(4, .5), colors =
brewer.pal(8,"Set2"), random.order = F)
```


```
#frequent terms
#findFreqTerms(sms_dtm, lowfreq =  100)
#findFreqTerms(sms_dtm, lowfreq =  1000)

#find associations
findAssocs(sms_dtm, 'applications', .5)

## $applications
##  americas microsoft          apisenable applicationsdevelop 
applicationstools
##                 0.58                0.58                0.58
0.58
##        consolidation          containing          data saras
etc saras
##                 0.58                0.58                0.58
0.58
##       excellence can              hyperv          microsofts
netmanage
##                 0.58                0.58                0.58
0.58
##         remotemobile               saras         silverlight
soap
##                 0.58                0.58                0.58
```

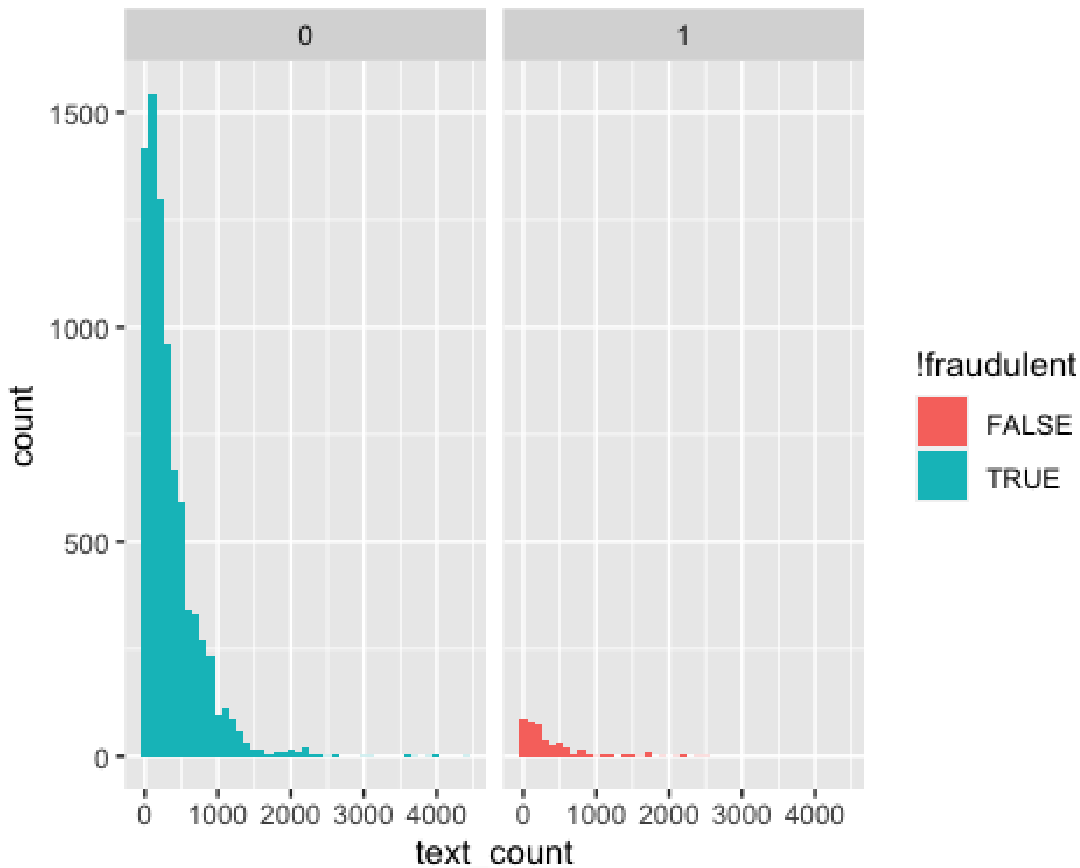
```
0.58
##      systemsmigrate  technologies saras      timeline saras
toolkitreengineer
##                0.58                0.58                0.58
0.58
##        unstructured     upgradesdevelop            visually
wcf
##                0.58                0.58                0.58
0.58
##                 wpf        departmental           youdesign
server
##                0.58                0.55                0.55
0.53
##                 sql            analyses
##                0.52                0.51
```

benefits

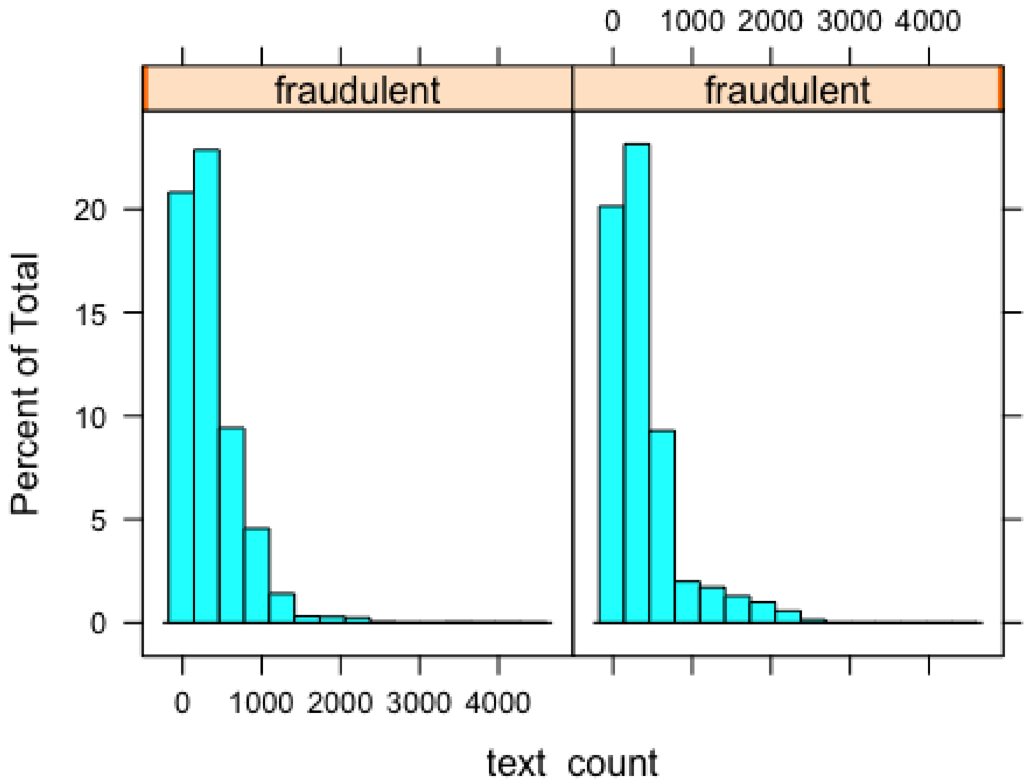
```
# General data info
copy4<- df_train

# Analysis
stricount_benef <- str_count(anette_df$benefits)
copy4 <- cbind(anette_df, stricount_benef)

myvars <- c("benefits", "stricount_benef", "fraudulent")
copy4 <- copy4[myvars]

#copy4 %>% arrange(desc(stricount_benef))
#copy4 %>% arrange(stricount_benef)


#Text count and Histograms
text_count <- str_count(copy4$benefits)
# a histogram of text length and visual comparison to fraud and not fraud
#layered
ggplot(copy4, aes(text_count, fill = !fraudulent)) + geom_histogram(binwidth
= 100)

## Warning: Removed 5724 rows containing non-finite values (`stat_bin()`).
```




```
#unlayered
ggplot(copy4, aes(text_count, fill = !fraudulent)) + geom_histogram(binwidth
= 100) + facet_wrap(~fraudulent)

## Warning: Removed 5724 rows containing non-finite values (`stat_bin()`).
```




```r
#another type of visual histogram of this
histogram(~text_count | fraudulent, data = copy4)
```




```
###########################################################################
###
sms_corpus <- VCorpus(VectorSource(copy4$benefits))
#sms_corpus_clean <- tm_map(sms_corpus, tolower) #all letters to lowercase
sms_corpus_clean <- tm_map(sms_corpus, content_transformer(tolower))
sms_corpus_clean <- tm_map(sms_corpus_clean, removeNumbers) #removes numbers
sms_corpus_clean <- tm_map(sms_corpus_clean, removePunctuation) #removes
punctuation
sms_corpus_clean <- tm_map(sms_corpus_clean, removeWords, stopwords())
#sms_corpus_clean <- tm_map(sms_corpus_clean, removeWords, stopwords('â'))
sms_corpus_clean <- tm_map(sms_corpus_clean, stripWhitespace)

course_corpus5 <- tm_map(sms_corpus_clean, stemDocument)
#analyze_corpus("Stemmed Corpus", course_corpus5)

#look at the cleaned corpus
#inspect(sms_corpus_clean[1:3])

#document term matrix used for analyzing tokenization
sms_dtm <- DocumentTermMatrix(sms_corpus_clean)

# displaying frequent terms in a wordcloud
```

```
wordcloud(sms_corpus_clean, min.freq = 1000, scale=c(4, .5), colors =
brewer.pal(8,"Set2"), random.order = F)

## Warning in wordcloud(sms_corpus_clean, min.freq = 1000, scale = c(4, 0.5),
:
## environment could not be fit on page. It will not be plotted.
```


```
#frequent terms
#findFreqTerms(sms_dtm, lowfreq =  100)
#findFreqTerms(sms_dtm, lowfreq =  1000)

#find associations
findAssocs(sms_dtm, 'applications', .5)

## $applications
##                  treated                       consider
##                     0.57                           0.54
##                  chances                       detailed
##                     0.52                           0.52
##             multilingual urlcffaceabdaccedbfdcdbfcacca
##                     0.52                           0.52
##                vacancies                  eurodyncareers
##                     0.52                           0.51
##                  quoting                        section
```

```
##                              0.51                             0.51
##                           seeking                         visiting
##                              0.51                             0.51
```

required_education

```
# General set up
copy5 <- df_train # copy dataset
copy5num <- copy5
# gives us amount of categories
categories <- unique(anette_df$required_education)
numberOfCategories <- length(categories)
categories
```

```
##  [1] "Unspecified"                        NA
##  [3] "Master's Degree"                    "High School or equivalent"
##  [5] "Bachelor's Degree"                  "Associate Degree"
##  [7] "Professional"                       "Certification"
##  [9] "Vocational"                         "Some College Coursework
Completed"
## [11] "Some High School Coursework"        "Vocational - Degree"
## [13] "Doctorate"                          "Vocational - HS Diploma"
```

```
#the way i dealt with NA's is I grouped them under unspecified
anette_df[anette_df=="Vocational - Degree"] <- "Vocational"
anette_df[anette_df=="Vocational - HS Diploma"] <- "Vocational"
anette_df$required_education <- anette_df$required_education %>%
replace_na("Unspecified")
unique(anette_df$required_education)
```

```
##  [1] "Unspecified"                        "Master's Degree"
##  [3] "High School or equivalent"          "Bachelor's Degree"
##  [5] "Associate Degree"                   "Professional"
##  [7] "Certification"                      "Vocational"
##  [9] "Some College Coursework Completed" "Some High School Coursework"
## [11] "Doctorate"
```

```
anette_df <- droplevels(anette_df)

#gives us the info above in a table
anette_df %>%
  count(required_education)
```

```
##                   required_education    n
## 1                   Associate Degree  218
## 2                  Bachelor's Degree 4117
## 3                      Certification  127
## 4                          Doctorate   19
## 5          High School or equivalent 1656
## 6                    Master's Degree  336
## 7                       Professional   60
## 8  Some College Coursework Completed   79
```

```
## 9        Some High School Coursework   18
## 10                       Unspecified 7624
## 11                        Vocational   50
```

```
# Making required education numeric
anette_df$educ_num = anette_df$required_education
anette_df$educ_num <- factor(anette_df$required_education)
anette_df$educ_num <- unclass(anette_df$required_education)


table1<- anette_df %>%
  count(educ_num)

mean(anette_df$educ_num)
```

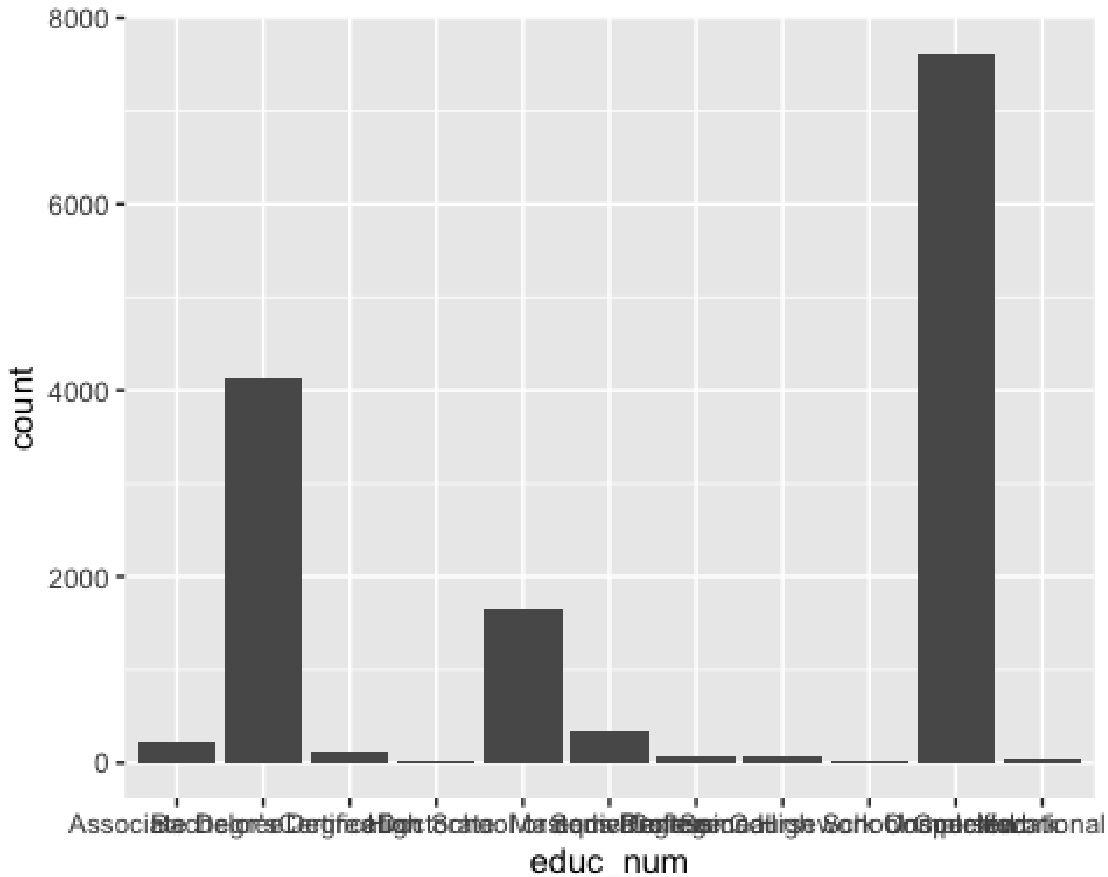
```
## Warning in mean.default(anette_df$educ_num): argument is not numeric or
logical:
## returning NA
```

```
## [1] NA
```

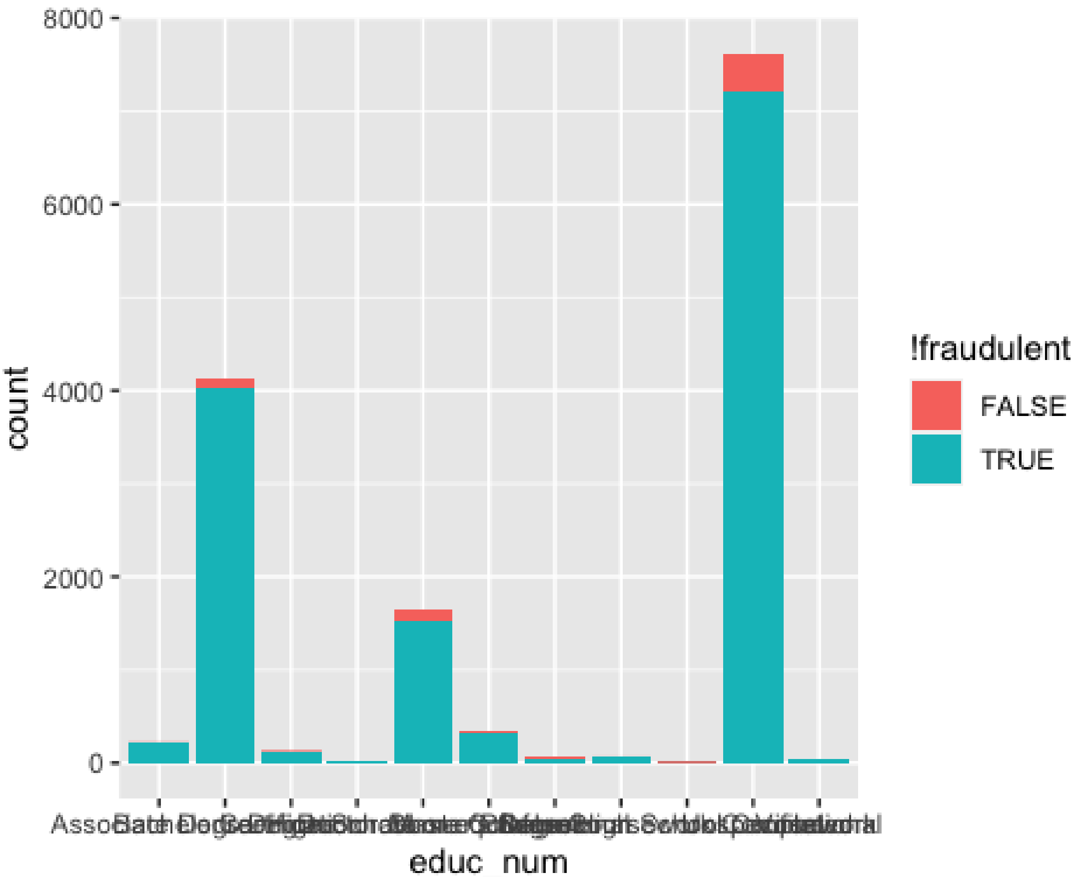
```
mode(anette_df$educ_num)
```

```
## [1] "character"
```

```
# Plotting
ggplot(data = anette_df) +
  geom_bar(mapping = aes(x = educ_num))
```




```r
#plotting state
ggplot(anette_df, aes(x = educ_num, fill =  !fraudulent))+
  geom_bar(stat = "count")
```




```
# the most common one is NA which I do not want to replace it with. If I
accumulate together NA and unspesified this also is the case where
unspecified is the most common.I ended up handeling the NA's in this case by
just making them unspecified.

# Mode
agg_tbl1 <- anette_df %>% group_by(required_education) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

# Mean
agg_tbl2 <- anette_df %>% group_by(required_education) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')
##############################################################################
###########
# This was the original way that I dealt with NA's and displayed them
(ignore)
##############################################################################
###########
# General Analysis

# dealing with NA's
```

```
# dropping NA
noNAcopy5 <- anette_df[!is.na(anette_df$required_education),]

ednoNAmode <- noNAcopy5 %>% group_by(required_education) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

# Mean
ednoNAmean <- noNAcopy5 %>% group_by(required_education) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')

# replacing with MIS
copy5$required_education<-as.character(anette_df$required_education)
anette_df$required_education<-as.character(anette_df$required_education)
copy5$required_education<- anette_df$required_education %>% replace_na('MIS')

agg_tbl1 <- anette_df %>% group_by(required_education) %>%
  summarise(fraudulent= get_mode(fraudulent),
            .groups = 'drop')

# Mean
agg_tbl2 <- anette_df %>% group_by(required_education) %>%
  summarise(fraudulent= mean(fraudulent),
            .groups = 'drop')
```

Adding Anette's features to the dataframe

Yes or no if country is USA, dummy variables for region and required_education

```
df_train$YNusa<-anette_df$YNusa
df_train$region_cat<-anette_df$region_cat

df_test$YNusa<-anette_test_df$YNusa
df_test$region_cat<-anette_test_df$region_cat

df_train<-dummy_cols(df_train, select_columns="region_cat")
df_test<-dummy_cols(df_test,select_columns="region_cat")
df_test<-dummy_cols(df_test,select_columns="required_education")

df_train<-dummy_cols(df_train,select_columns="required_education")
```

Tiffany's EDA and Feature Engineering

```
tiff_attrs <- c("index", "department", "description", "has_questions",
"employment_type", "industry", "fraudulent")
tiff_df <- df_train[tiff_attrs]

# convert fraudulent as factor type
tiff_df$fraudulent <- as.factor(tiff_df$fraudulent)
```

```r
# clean department column
departments <- tiff_df$department
# set to lowercase
departments <- tolower(departments)
tiff_df$department <- departments

# clean industry column
industries <- tiff_df$industry
# set to lowercase
industries <- tolower(industries)
tiff_df$industry <- industries

#head(tiff_df)
```

```r
tiff_test_df <- df_test[tiff_attrs]

# convert fraudulent as factor type
tiff_test_df$fraudulent <- as.factor(tiff_test_df$fraudulent)

# clean department column
departments <- tiff_test_df$department
# set to lowercase
departments <- tolower(departments)
tiff_test_df$department <- departments

# clean industry column
industries <- tiff_test_df$industry
# set to lowercase
industries <- tolower(industries)
tiff_test_df$industry <- industries

#head(tiff_test_df)
```

create dataframe on column info

```r
# create a SEPARATE df for info abt my subset of variables
# look at num of missing values
info <- data.frame(sapply(tiff_df, function(x) sum(is.na(x))))
names(info) <- c("missing values")

# ratio of missing values
info$missingratio <- info$"missing values"/nrow(tiff_df)

# look at unique values of each column
uniquevals <- c()
for (column in tiff_attrs) {
  uniquevals <- append(uniquevals, nrow(unique(tiff_df[column])))
```

```
}
info$unique <- uniquevals

# number of unique values of each attribute
nuniquevals <- c()
for (column in tiff_attrs) {
  nuniquevals <- append(nuniquevals, n_distinct(tiff_df[column]))
}
info$nunique <- nuniquevals

# get data type of each
info$'data type' <- sapply(tiff_df, typeof)

# find frequency of most common value
frequency_common <- c()
common_values <- c()
freq_ratio <- c()
for (column in tiff_attrs) {
  common_info <- as.data.frame(head(sort(table(tiff_df[column]),
decreasing=TRUE), 1) )
  common_values <- append(common_values, common_info$Var1)
  frequency_common <- append(frequency_common, common_info$Freq)
  freq_ratio <- append(freq_ratio, (common_info$Freq/nrow(tiff_df))*100)
}
info$'most common value' <- common_values
info$'frequency of MCV' <- frequency_common
info$'MCV ratio' <- freq_ratio

info

##                  missing values missingratio unique nunique data type
## index                         0    0.0000000  14304   14304   integer
## department                 9265    0.6477209   1120    1120 character
## description                   0    0.0000000  12031   12031 character
## has_questions                 0    0.0000000      2       2   integer
## employment_type            2813    0.1966583      6       6 character
## industry                   3951    0.2762164    131     131 character
## fraudulent                    0    0.0000000      2       2   integer
##
most common value
## index
2
## department
sales
## description     Play with kids, get paid for it Love travel? Jobs in
Asia$1,500+ USD monthly ($200 Cost of living)Housing provided
(Private/Furnished)Airfare ReimbursedExcellent for student loans/credit
cardsGabriel Adkins :
#URL_ed9094c60184b8a4975333957f05be37e69d3cdb68decc9dd9a4242733cfd7f7##URL_75
db76d58f7994c7db24e8998c2fc953ab9a20ea9ac948b217693963f78d2e6b#12 month
```

```
contract : Apply today
## has_questions
0
## employment_type
Full-time
## industry
information technology and services
## fraudulent
0
##                    frequency of MCV    MCV ratio
## index                             1  0.006991051
## department                      452  3.159955257
## description                     310  2.167225951
## has_questions                  7228 50.531319911
## employment_type                9281 64.883948546
## industry                       1378  9.633668904
## fraudulent                    13604 95.106263982
```

```
info$"missing values"/nrow(tiff_df)
```

```
## [1] 0.0000000 0.6477209 0.0000000 0.0000000 0.1966583 0.2762164 0.0000000
```

department

group similar departments together

```
# replace certain department names
n_distinct(tiff_test_df$department)
```

```
## [1] 442
```

```
# customer support
tiff_test_df$department[grep("support", tiff_test_df$department)]<-"customer
support"
# human resources
tiff_test_df$department[grep("hr", tiff_test_df$department)]<-"human
resources"
tiff_test_df$department[grep("human", tiff_test_df$department)]<-"human
resources"
tiff_test_df$department[grep("resources", tiff_test_df$department)]<-"human
resources"
# information technology
tiff_test_df$department[grep("information", tiff_test_df$department)]<-
"information technology"
tiff_test_df$department[grep("technology", tiff_test_df$department)]<-
"information technology"
tiff_test_df$department[grep("tech", tiff_test_df$department)]<-"information
technology"
tiff_test_df$department[grep("i. t.", tiff_test_df$department)]<-"information
technology"
tiff_test_df$department[grep("it", tiff_test_df$department)]<-"information
```

```
technology"
# engineering
tiff_test_df$department[grep("engineer", tiff_test_df$department)]<-
"engineering"
# sales
tiff_test_df$department[grep("sales", tiff_test_df$department)]<-"sales"
# finance
tiff_test_df$department[grep("finance", tiff_test_df$department)]<-"finance"
# marketing
tiff_test_df$department[grep("marketing", tiff_test_df$department)]<-
"marketing"
tiff_test_df$department[grep("market", tiff_test_df$department)]<-"marketing"
tiff_test_df$department[grep("mkt", tiff_test_df$department)]<-"marketing"
# accounting
tiff_test_df$department[grep("accounting", tiff_test_df$department)]<-
"accounting"
# healthcare
tiff_test_df$department[grep("health", tiff_test_df$department)]<-
"healthcare"

tiff_test_df$department[grep("admin", tiff_test_df$department)]<-
"administrative"
# customer service
tiff_test_df$department[grep("customer", tiff_test_df$department)]<-"customer
service"
tiff_test_df$department[grep("client", tiff_test_df$department)]<-"customer
service"
tiff_test_df$department[grep("csd", tiff_test_df$department)]<-"customer
service"

tiff_test_df$department[grep("oil", tiff_test_df$department)]<-"oil"
tiff_test_df$department[grep("operation", tiff_test_df$department)]<-
"operations"
tiff_test_df$department[grep("retail", tiff_test_df$department)]<-"retail"
tiff_test_df$department[grep("recruit", tiff_test_df$department)]<-
"recruiting"
tiff_test_df$department[grep("construction", tiff_test_df$department)]<-
"construction"
tiff_test_df$department[grep("content", tiff_test_df$department)]<-"product
content"
tiff_test_df$department[grep("dev", tiff_test_df$department)]<-"developer"
tiff_test_df$department[grep("software", tiff_test_df$department)]<-
"software"
tiff_test_df$department[grep("hardware", tiff_test_df$department)]<-
"hardware"

tiff_test_df$department[grep("design", tiff_test_df$department)]<-"design"

n_distinct(tiff_test_df$department)
```

```
## [1] 292
```

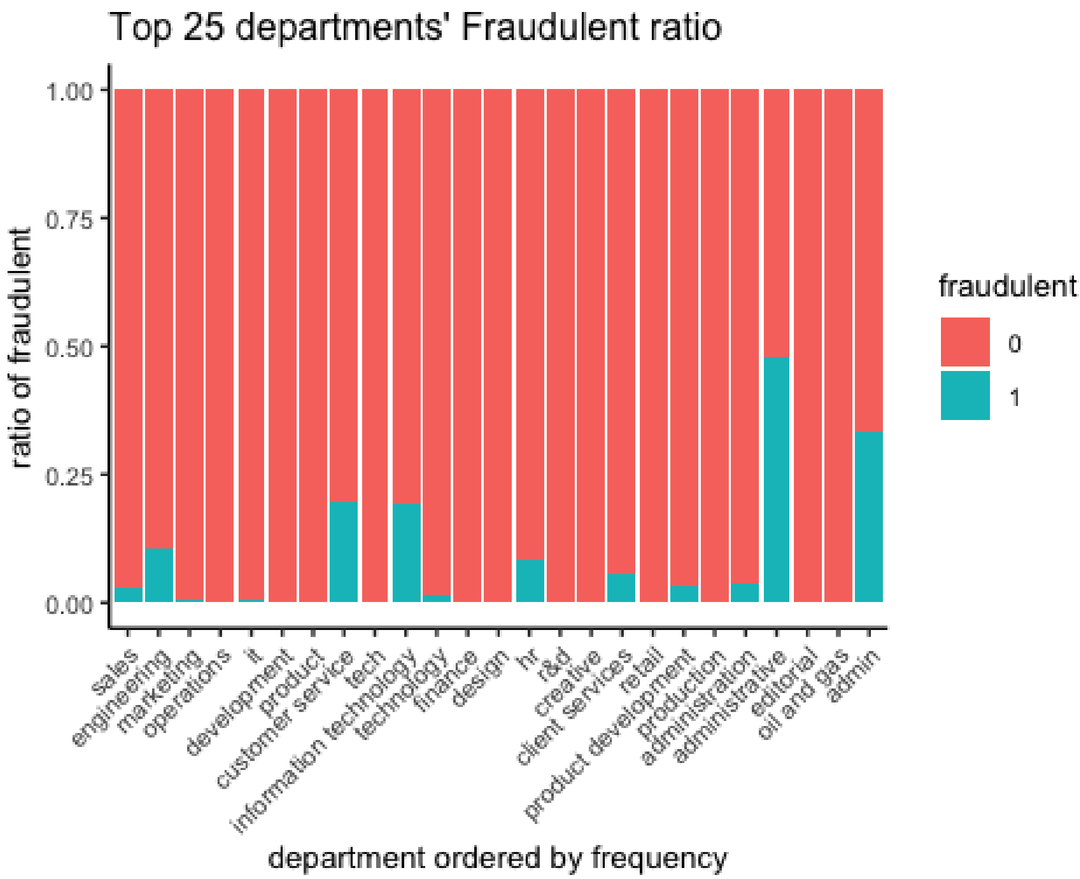
top 10 departments

```
topdepartments <- as.data.frame(head(sort(table(tiff_df$department),
decreasing=TRUE), 25))
topdepartments$ratio <- topdepartments$Freq*100/nrow(tiff_df)
# top10department$fraud_ratio <- sum(subset(tiff_df, department %in%
top10department$Var1)$fraudulent) / nrow(tiff_df)
topdepartments
```

```
##                      Var1 Freq     ratio
## 1                   sales  452 3.1599553
## 2             engineering  411 2.8733221
## 3               marketing  312 2.1812081
## 4              operations  210 1.4681208
## 5                      it  183 1.2793624
## 6             development  111 0.7760067
## 7                 product   92 0.6431767
## 8        customer service   71 0.4963647
## 9                    tech   69 0.4823826
## 10 information technology   63 0.4404362
## 11             technology   63 0.4404362
## 12                finance   58 0.4054810
## 13                 design   54 0.3775168
## 14                     hr   49 0.3425615
## 15                    r&d   42 0.2936242
## 16               creative   38 0.2656600
## 17        client services   35 0.2446868
## 18                 retail   34 0.2376957
## 19    product development   32 0.2237136
## 20             production   27 0.1887584
## 21         administration   26 0.1817673
## 22         administrative   25 0.1747763
## 23              editorial   25 0.1747763
## 24            oil and gas   25 0.1747763
## 25                  admin   24 0.1677852
```

look at top 10 departments and their fraud percentages

```
df_topdep <- subset(tiff_df, department %in% topdepartments$Var1)
ggplot(df_topdep, aes(x = reorder(department, department, function(x)-
length(x)), fill = fraudulent)) +
    geom_bar(position = "fill") +
    theme_classic() +
  ggtitle("Top 25 departments' Fraudulent ratio") +
  theme(text = element_text(size=10),
        axis.text.x = element_text(angle=45, hjust=1)) +
  xlab("department ordered by frequency") +
  ylab("ratio of fraudulent")
```



Listings with department = engineering, administrative, oil, accounting, maintenance, and clerical have the highest fraudulent postings.

dep_top

create binary var to see if listing has department that is in top 25 departments

```
tiff_df$dep_top <- ifelse(tiff_df$department %in% topdepartments$Var1, 1, 0)
sum(tiff_df$dep_top)/nrow(tiff_df)

## [1] 0.1769435

tiff_test_df$dep_top<-ifelse(tiff_test_df$department %in%
topdepartments$Var1,1,0)
```

dep_admin

```
tiff_df$dep_admin <- ifelse(tiff_df$department %in% "administrative", 1, 0)
sum(tiff_df$dep_admin)/nrow(tiff_df)

## [1] 0.001747763
```

dep_engineering

```
tiff_df$dep_engineering <- ifelse(tiff_df$department %in% "engineering", 1,
0)
sum(tiff_df$dep_engineering)/nrow(tiff_df)

## [1] 0.02873322
```

dep_oil

```
tiff_df$dep_oil <- ifelse(tiff_df$department %in% "oil", 1, 0)
sum(tiff_df$dep_oil)/nrow(tiff_df)

## [1] 0
```

dep_admin

```
tiff_test_df$dep_admin <- ifelse(tiff_test_df$department %in%
"administrative", 1, 0)
sum(tiff_test_df$dep_admin)/nrow(tiff_test_df)

## [1] 0.003914989
```

dep_engineering

```
tiff_test_df$dep_engineering <- ifelse(tiff_test_df$department %in%
"engineering", 1, 0)
sum(tiff_test_df$dep_engineering)/nrow(tiff_test_df)

## [1] 0.026566
```

dep_oil

```
tiff_test_df$dep_oil <- ifelse(tiff_test_df$department %in% "oil", 1, 0)
sum(tiff_test_df$dep_oil)/nrow(tiff_test_df)

## [1] 0.003914989
```

has_department

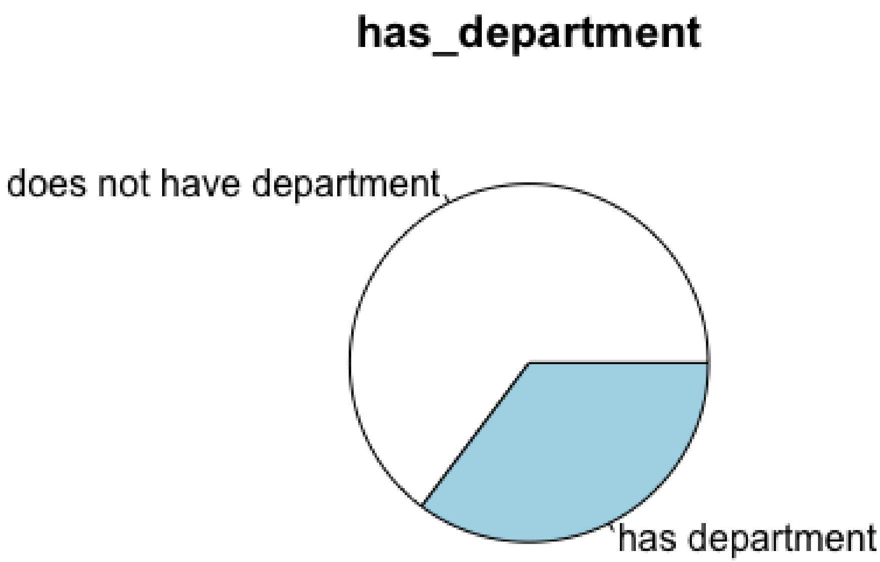
```
# create binary variable on department exist or not
tiff_df$has_department <- sapply(tiff_df$department, function(f)
{as.numeric(!(is.na(f)))})
tiff_test_df$has_department <- sapply(tiff_test_df$department, function(f)
{as.numeric(!(is.na(f)))})
#head(tiff_df)
```

visuals

Majority of listed does NOT have department included.

```
tiff_df2 <- tiff_df %>%
      mutate(has_department = ifelse(has_department == "1","has
department","does not have department"))
#counts
(typeCounts <- table(tiff_df2$has_department))
```

```
##
## does not have department            has department
##                     9265                      5039

#percents
prop.table(typeCounts)

##
## does not have department            has department
##                0.6477209                 0.3522791

#display
pie(typeCounts, main = "has_department")
```



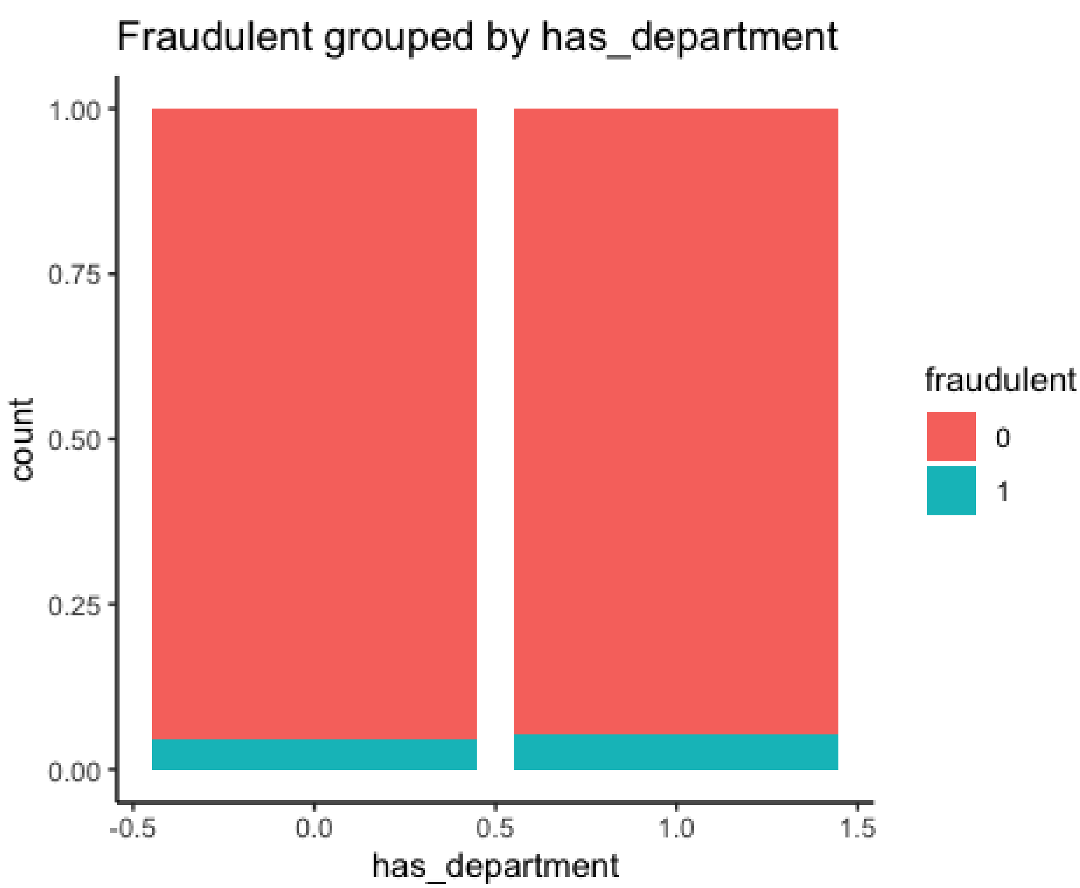

```
(plotdata <- tiff_df %>%
  group_by(has_department, fraudulent) %>%
  summarize(n = n()) %>%
  mutate(pct = n/sum(n),
         lbl = scales::percent(pct)))

## `summarise()` has grouped output by 'has_department'. You can override
using
## the `.groups` argument.
```

```
## # A tibble: 4 × 5
## # Groups:   has_department [2]
##   has_department fraudulent     n    pct lbl  
##            <dbl> <fct>      <int>  <dbl> <chr>
## 1              0 0           8834 0.953  95%  
## 2              0 1            431 0.0465 5%   
## 3              1 0           4770 0.947  95%  
## 4              1 1            269 0.0534 5%
```

Proportion of fraudulent to non-fraudulent is the same whether or not the listing has the department listed or not.

It's 95 to 5 for non-fraudulent to fraudulent listings.

```
ggplot(tiff_df, aes(x = has_department, fill = fraudulent)) +
    geom_bar(position = "fill") +
    theme_classic() +
  ggtitle("Fraudulent grouped by has_department")
```



industry

group similar industries together

```r
# replace certain industry names
n_distinct(tiff_df$industry)

## [1] 131

# customer support
tiff_df$industry[grep("support", tiff_df$industry)]<-"customer support"
# human resources
tiff_df$industry[grep("hr", tiff_df$industry)]<-"human resources"
tiff_df$industry[grep("human", tiff_df$industry)]<-"human resources"
tiff_df$industry[grep("resources", tiff_df$industry)]<-"human resources"
# information technology
tiff_df$industry[grep("information", tiff_df$industry)]<-"information
technology"
tiff_df$industry[grep("technology", tiff_df$industry)]<-"information
technology"
tiff_df$industry[grep("tech", tiff_df$industry)]<-"information technology"
tiff_df$industry[grep("i. t.", tiff_df$industry)]<-"information technology"
tiff_df$industry[grep("it", tiff_df$industry)]<-"information technology"
# engineering
tiff_df$industry[grep("engineer", tiff_df$industry)]<-"engineering"
# sales
tiff_df$industry[grep("sales", tiff_df$industry)]<-"sales"
# finance
tiff_df$industry[grep("finance", tiff_df$industry)]<-"finance"
# marketing
tiff_df$industry[grep("marketing", tiff_df$industry)]<-"marketing"
tiff_df$industry[grep("market", tiff_df$industry)]<-"marketing"
tiff_df$industry[grep("mkt", tiff_df$industry)]<-"marketing"
# accounting
tiff_df$industry[grep("accounting", tiff_df$industry)]<-"accounting"
# healthcare
tiff_df$industry[grep("health", tiff_df$industry)]<-"healthcare"

tiff_df$industry[grep("admin", tiff_df$industry)]<-"administrative"
# customer service
tiff_df$industry[grep("customer", tiff_df$industry)]<-"customer service"
tiff_df$industry[grep("client", tiff_df$industry)]<-"customer service"
tiff_df$industry[grep("csd", tiff_df$industry)]<-"customer service"

tiff_df$industry[grep("oil", tiff_df$industry)]<-"oil"
tiff_df$industry[grep("operation", tiff_df$industry)]<-"operations"
tiff_df$industry[grep("retail", tiff_df$industry)]<-"retail"
tiff_df$industry[grep("recruit", tiff_df$industry)]<-"recruiting"
tiff_df$industry[grep("construction", tiff_df$industry)]<-"construction"
tiff_df$industry[grep("content", tiff_df$industry)]<-"product content"
tiff_df$industry[grep("dev", tiff_df$industry)]<-"developer"
tiff_df$industry[grep("software", tiff_df$industry)]<-"software"
tiff_df$industry[grep("hardware", tiff_df$industry)]<-"hardware"
```

```
tiff_df$industry[grep("design", tiff_df$industry)]<-"design"

n_distinct(tiff_df$industry)

## [1] 103

# replace certain industry names
n_distinct(tiff_test_df$industry)

## [1] 117

# customer support
tiff_test_df$industry[grep("support", tiff_test_df$industry)]<-"customer
support"
# human resources
tiff_test_df$industry[grep("hr", tiff_test_df$industry)]<-"human resources"
tiff_test_df$industry[grep("human", tiff_test_df$industry)]<-"human
resources"
tiff_test_df$industry[grep("resources", tiff_test_df$industry)]<-"human
resources"
# information technology
tiff_test_df$industry[grep("information", tiff_test_df$industry)]<-
"information technology"
tiff_test_df$industry[grep("technology", tiff_test_df$industry)]<-
"information technology"
tiff_test_df$industry[grep("tech", tiff_test_df$industry)]<-"information
technology"
tiff_test_df$industry[grep("i. t.", tiff_test_df$industry)]<-"information
technology"
tiff_test_df$industry[grep("it", tiff_test_df$industry)]<-"information
technology"
# engineering
tiff_test_df$industry[grep("engineer", tiff_test_df$industry)]<-"engineering"
# sales
tiff_test_df$industry[grep("sales", tiff_test_df$industry)]<-"sales"
# finance
tiff_test_df$industry[grep("finance", tiff_test_df$industry)]<-"finance"
# marketing
tiff_test_df$industry[grep("marketing", tiff_test_df$industry)]<-"marketing"
tiff_test_df$industry[grep("market", tiff_test_df$industry)]<-"marketing"
tiff_test_df$industry[grep("mkt", tiff_test_df$industry)]<-"marketing"
# accounting
tiff_test_df$industry[grep("accounting", tiff_test_df$industry)]<-
"accounting"
# healthcare
tiff_test_df$industry[grep("health", tiff_test_df$industry)]<-"healthcare"

tiff_test_df$industry[grep("admin", tiff_test_df$industry)]<-"administrative"
# customer service
tiff_test_df$industry[grep("customer", tiff_test_df$industry)]<-"customer
service"
```

```
tiff_test_df$industry[grep("client", tiff_test_df$industry)]<-"customer
service"
tiff_test_df$industry[grep("csd", tiff_test_df$industry)]<-"customer service"

tiff_test_df$industry[grep("oil", tiff_test_df$industry)]<-"oil"
tiff_test_df$industry[grep("operation", tiff_test_df$industry)]<-"operations"
tiff_test_df$industry[grep("retail", tiff_test_df$industry)]<-"retail"
tiff_test_df$industry[grep("recruit", tiff_test_df$industry)]<-"recruiting"
tiff_test_df$industry[grep("construction", tiff_test_df$industry)]<-
"construction"
tiff_test_df$industry[grep("content", tiff_test_df$industry)]<-"product
content"
tiff_test_df$industry[grep("dev", tiff_test_df$industry)]<-"developer"
tiff_test_df$industry[grep("software", tiff_test_df$industry)]<-"software"
tiff_test_df$industry[grep("hardware", tiff_test_df$industry)]<-"hardware"

tiff_test_df$industry[grep("design", tiff_test_df$industry)]<-"design"

n_distinct(tiff_test_df$industry)

## [1] 95
```

look at top 10 industries

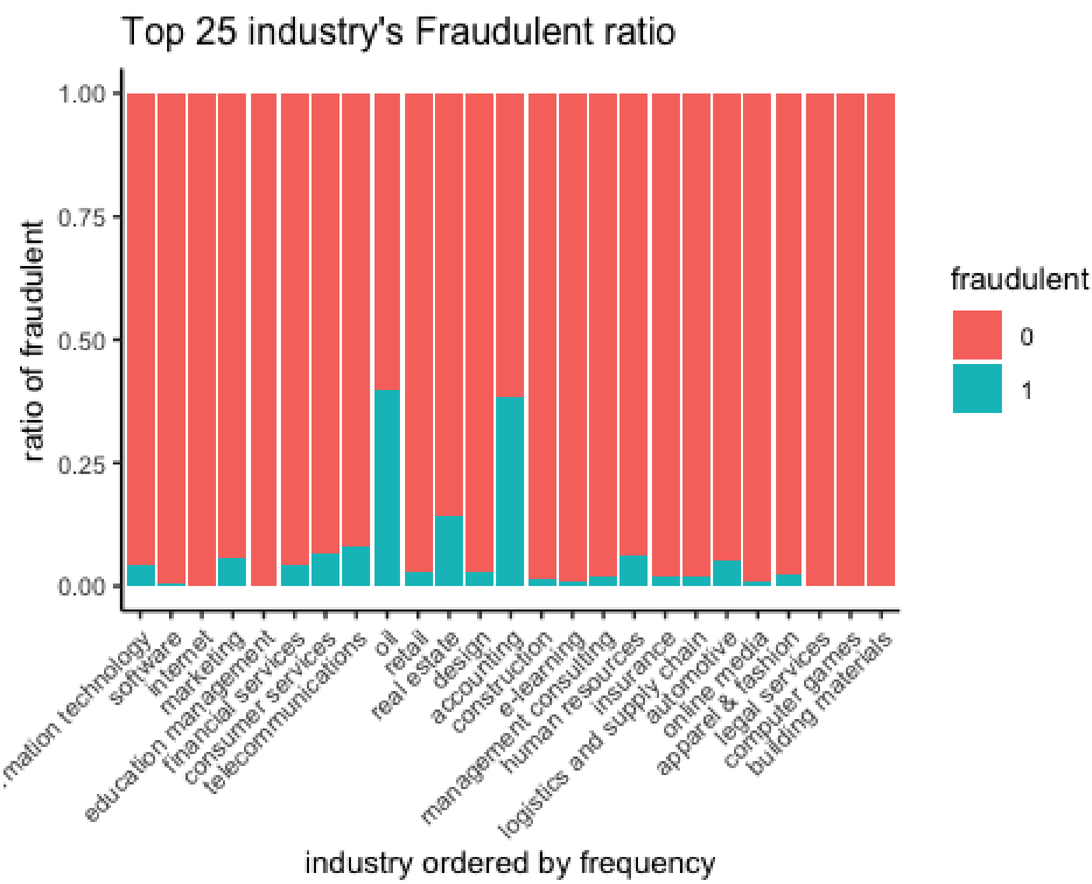
```
topindustry <- as.data.frame(head(sort(table(tiff_df$industry),
decreasing=TRUE), 25))
topindustry

##                          Var1 Freq
## 1      information technology 2377
## 2                    software 1071
## 3                    internet  844
## 4                   marketing  700
## 5       education management  661
## 6          financial services  627
## 7           consumer services  290
## 8          telecommunications  265
## 9                         oil  234
## 10                     retail  168
## 11                real estate  139
## 12                     design  134
## 13                 accounting  130
## 14               construction  123
## 15                 e-learning  117
## 16      management consulting  110
## 17            human resources   99
## 18                  insurance   98
## 19 logistics and supply chain   98
## 20                 automotive   95
## 21               online media   84
```

```
## 22          apparel & fashion   78
## 23             legal services   72
## 24             computer games   69
## 25         building materials   66
```

Majority of listings are in technology related roles. Top 3 are information technology, software, and internet.

look at top 10 industries and their fraud percentages

```
df_topind <- subset(tiff_df, industry %in% topindustry$Var1)
ggplot(df_topind, aes(x = reorder(industry, industry, function(x)-length(x)),
fill = fraudulent)) +
    geom_bar(position = "fill") +
    theme_classic() +
  ggtitle("Top 25 industry's Fraudulent ratio") +
  theme(text = element_text(size=10),
        axis.text.x = element_text(angle=45, hjust=1)) +
   xlab("industry ordered by frequency") +
  ylab("ratio of fraudulent")
```



Listings with industry=accounting or industry=oil&energy have the highest fraudulent postings.

industry_top create binary var to see if listing has department that is in top 25 departments

```
tiff_df$industry_top <- ifelse(tiff_df$industry %in% topindustry$Var1, 1, 0)
tiff_test_df$industry_top <- ifelse(tiff_test_df$industry %in%
topindustry$Var1, 1, 0)
sum(tiff_df$industry_top)/nrow(tiff_df)

## [1] 0.6116471
```

industry_acc

```
tiff_df$industry_acc <- ifelse(tiff_df$industry %in% "accounting", 1, 0)
sum(tiff_df$industry_acc)/nrow(tiff_df)

## [1] 0.009088367
```

industry_oilenergy

```
tiff_df$industry_oilenergy <- ifelse(tiff_df$industry %in% "oil", 1, 0)
sum(tiff_df$industry_oilenergy)/nrow(tiff_df)

## [1] 0.01635906
```

has_industry

```
# create binary variable on industry exist or not
tiff_df$has_industry <- sapply(tiff_df$industry, function(f)
{as.numeric(!(is.na(f)))})
```

industry_acc

```
tiff_test_df$industry_acc <- ifelse(tiff_test_df$industry %in% "accounting",
1, 0)
sum(tiff_test_df$industry_acc)/nrow(tiff_test_df)

## [1] 0.00810962
```

industry_oilenergy

```
tiff_test_df$industry_oilenergy <- ifelse(tiff_test_df$industry %in% "oil",
1, 0)
sum(tiff_test_df$industry_oilenergy)/nrow(tiff_test_df)

## [1] 0.01482103
```

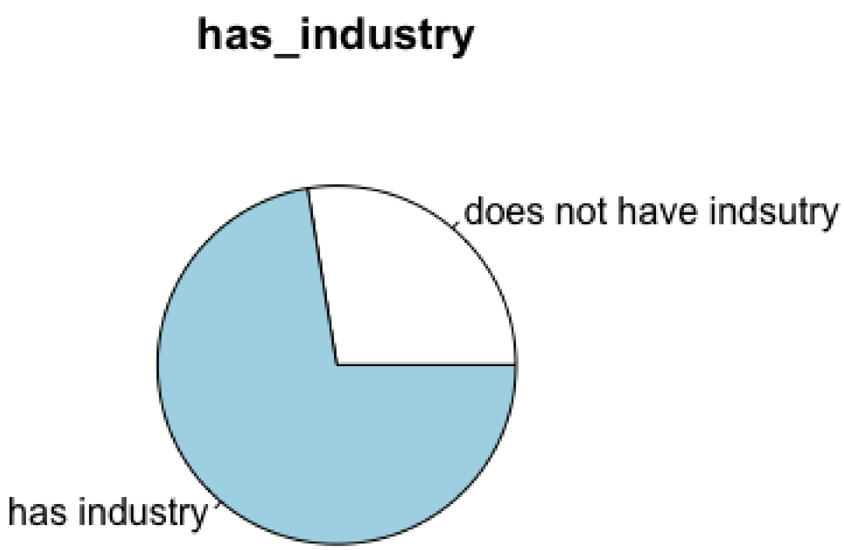
has_industry

```
# create binary variable on industry exist or not
tiff_test_df$has_industry <- sapply(tiff_test_df$industry, function(f)
{as.numeric(!(is.na(f)))})
```

visuals

About 75% of listings does DO have industry included.

```
tiff_df2 <- tiff_df %>%
      mutate(has_industry = ifelse(has_industry == "1","has industry","does
not have indsutry"))
#counts
(typeCounts <- table(tiff_df2$has_industry))

##
## does not have indsutry           has industry
##                   3951                  10353

#percents
prop.table(typeCounts)

##
## does not have indsutry           has industry
##              0.2762164              0.7237836

#display
pie(typeCounts, main = "has_industry")
```




```
(plotdata <- tiff_df %>%
  group_by(has_industry, fraudulent) %>%
  summarize(n = n()) %>%
```

```
  mutate(pct = n/sum(n),
         lbl = scales::percent(pct)))
```

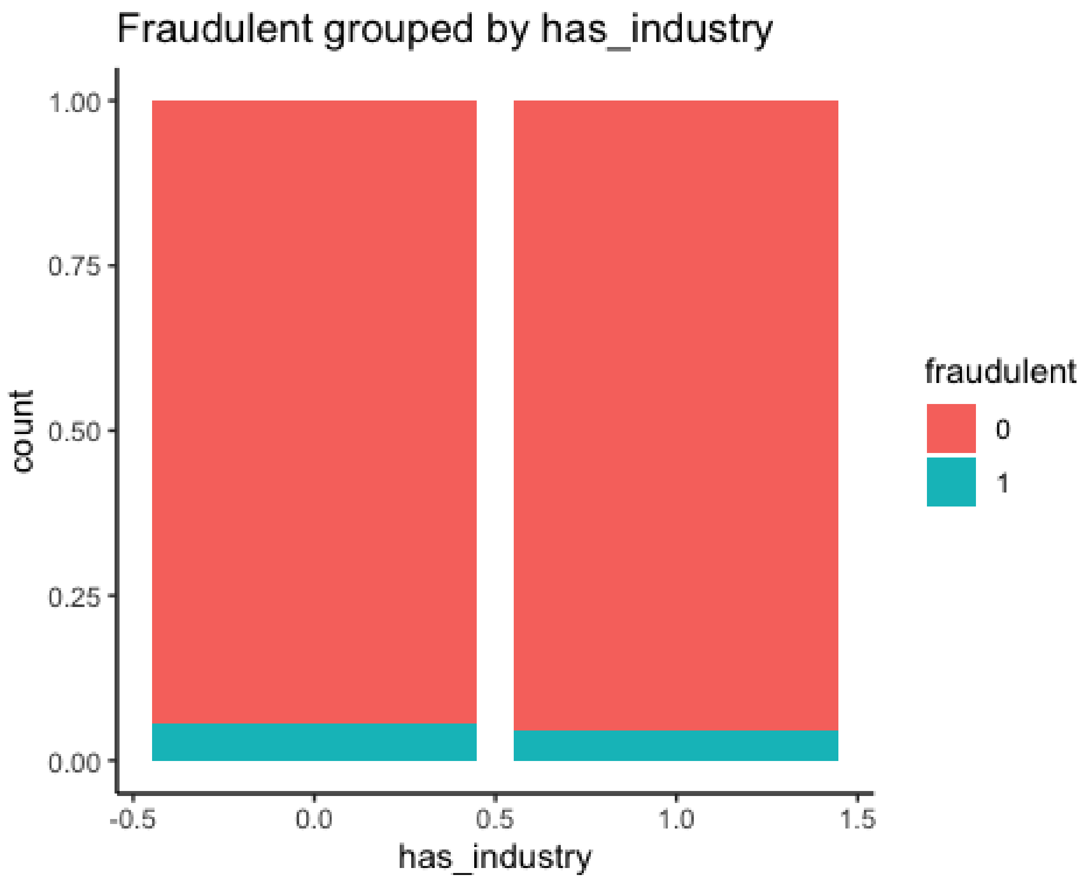
```
## `summarise()` has grouped output by 'has_industry'. You can override using
the
## `.groups` argument.
```

```
## # A tibble: 4 × 5
## # Groups:   has_industry [2]
##   has_industry fraudulent     n    pct lbl
##          <dbl> <fct>      <int>  <dbl> <chr>
## 1            0 0           3734 0.945  95%
## 2            0 1            217 0.0549 5%
## 3            1 0           9870 0.953  95%
## 4            1 1            483 0.0467 5%
```

Proportion of fraudulent to non-fraudulent is the similar whether or not the listing has the industry listed or not.

It's 95 to 5 for non-fraudulent to fraudulent listings.

```
ggplot(tiff_df, aes(x = has_industry, fill = fraudulent)) +
    geom_bar(position = "fill") +
    theme_classic() +
  ggtitle("Fraudulent grouped by has_industry")
```



description

cleaned_description column

```
# clean description column
descripts <- tiff_df$description

wordcount<-str_count(descripts)
max_wordcount<-max(wordcount)
text_corpus<-VCorpus(VectorSource(descripts))
toSpace <- content_transformer(function(x, pattern) gsub(pattern, "", x))
text_corpus <- tm_map(text_corpus, toSpace, "[^[:print:]]")
text_corpus <- tm_map(text_corpus, removePunctuation)
text_corpus <- tm_map(text_corpus, removeNumbers)
text_corpus <- tm_map(text_corpus, content_transformer(tolower))
text_corpus <- tm_map(text_corpus, stripWhitespace)
text_corpus_no_stopwords <- tm_map(text_corpus, removeWords,
stopwords("english"))
text_corpus_no_stopwords <- tm_map(text_corpus_no_stopwords, stripWhitespace)

# Remove mentions, urls, emojis, numbers, punctuations, etc.
descripts <- gsub("@\\w+", "", descripts)
descripts <- gsub("https?://.+", "", descripts)
```

```r
descripts <- gsub("\\d+\\w*\\d*", "", descripts)
descripts <- gsub("#\\w+", "", descripts)
descripts <- gsub("[^\x01-\x7F]", "", descripts)
descripts <- gsub("[[:punct:]]", " ", descripts)

# Remove spaces and newlines
descripts <- gsub("\n", " ", descripts)
descripts <- gsub("^\\s+", "", descripts)
descripts <- gsub("\\s+$", "", descripts)
descripts <- gsub("[ |\t]+", " ", descripts)

# Put the data to a new column
tiff_df["cleaned_description"] <- descripts
```

description word count

```r
# get word count
tiff_df$cleandescription_length <-
sapply(strsplit(tiff_df$cleaned_description, " "), length)
```

has_description

```r
# create binary variable on department exist or not
tiff_df$has_description <- sapply(tiff_df$description, function(f)
{as.numeric(!(is.na(f)))})
```

cleaned_description column

```r
# clean description column
descripts <- tiff_test_df$description

wordcount<-str_count(descripts)
max_wordcount<-max(wordcount)
text_corpus<-VCorpus(VectorSource(descripts))
toSpace <- content_transformer(function(x, pattern) gsub(pattern, "", x))
text_corpus <- tm_map(text_corpus, toSpace, "[^[:print:]]")
text_corpus <- tm_map(text_corpus, removePunctuation)
text_corpus <- tm_map(text_corpus, removeNumbers)
text_corpus <- tm_map(text_corpus, content_transformer(tolower))
text_corpus <- tm_map(text_corpus, stripWhitespace)
text_corpus_no_stopwords <- tm_map(text_corpus, removeWords,
stopwords("english"))
text_corpus_no_stopwords <- tm_map(text_corpus_no_stopwords, stripWhitespace)

# Remove mentions, urls, emojis, numbers, punctuations, etc.
descripts <- gsub("@\\w+", "", descripts)
descripts <- gsub("https?://.+", "", descripts)
descripts <- gsub("\\d+\\w*\\d*", "", descripts)
descripts <- gsub("#\\w+", "", descripts)
descripts <- gsub("[^\x01-\x7F]", "", descripts)
descripts <- gsub("[[:punct:]]", " ", descripts)
```

```
# Remove spaces and newlines
descripts <- gsub("\n", " ", descripts)
descripts <- gsub("^\\s+", "", descripts)
descripts <- gsub("\\s+$", "", descripts)
descripts <- gsub("[ |\t]+", " ", descripts)

# Put the data to a new column
tiff_test_df["cleaned_description"] <- descripts
```

description word count

```
# get word count
tiff_test_df$cleandescription_length <-
sapply(strsplit(tiff_test_df$cleaned_description, " "), length)
```

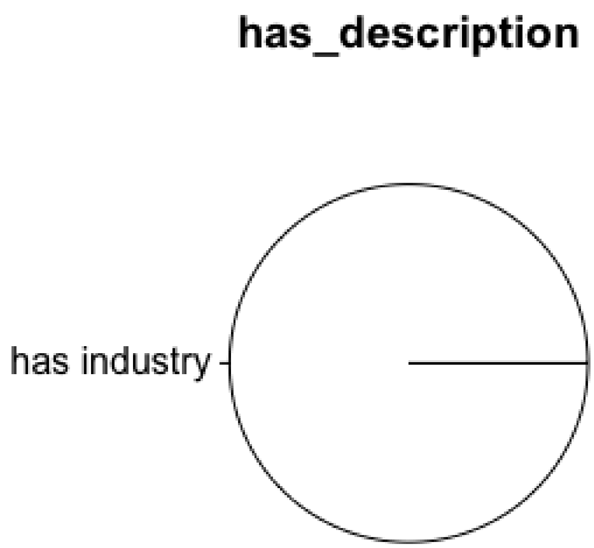
has_description

```
# create binary variable on department exist or not
tiff_test_df$has_description <- sapply(tiff_test_df$description, function(f)
{as.numeric(!(is.na(f)))})
```

visuals

About 75% of listings does DO have description included.

```
tiff_df2 <- tiff_df %>%
      mutate(has_description = ifelse(has_description == "1","has
industry","does not have indsutry"))
#counts
(typeCounts <- table(tiff_df2$has_description))

##
## has industry
##        14304

#percents
prop.table(typeCounts)

##
## has industry
##            1

#display
pie(typeCounts, main = "has_description")
```




```
(plotdata <- tiff_df %>%
  group_by(has_description, fraudulent) %>%
  summarize(n = n()) %>%
  mutate(pct = n/sum(n),
         lbl = scales::percent(pct)))
```

```
## `summarise()` has grouped output by 'has_description'. You can override
using
## the `.groups` argument.
```

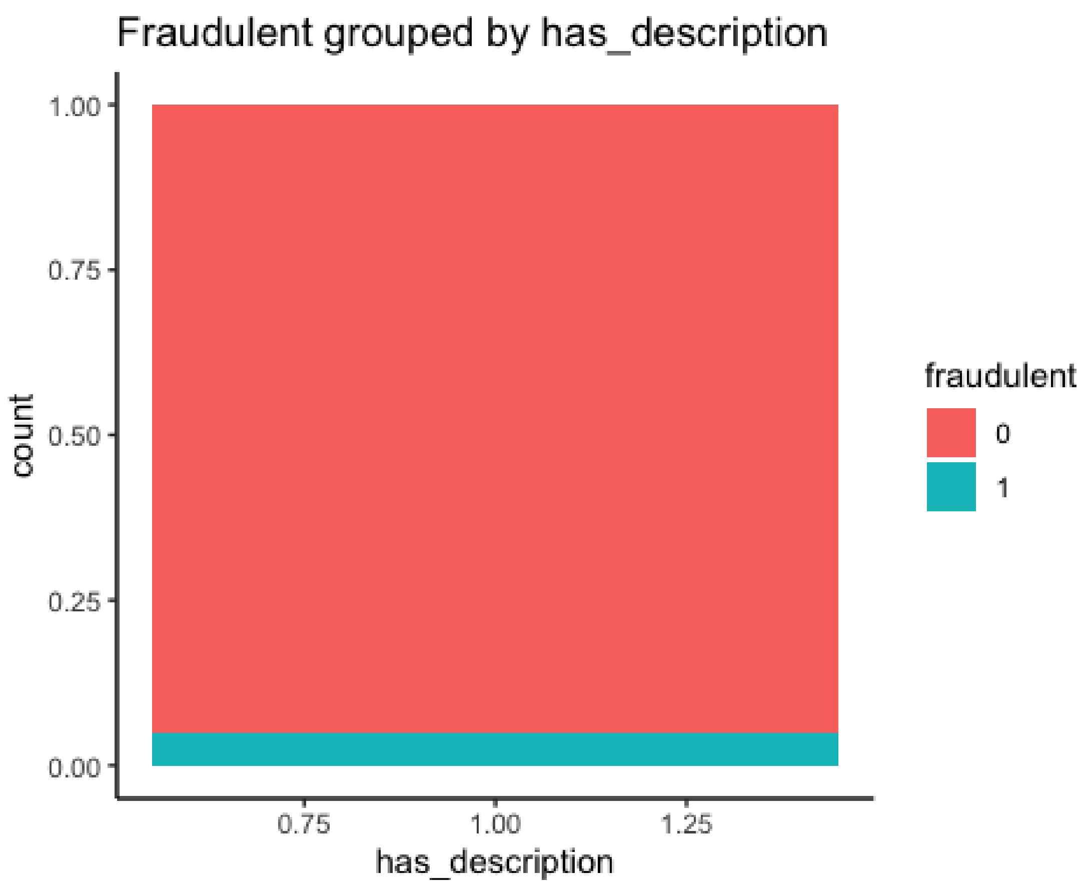
```
## # A tibble: 2 × 5
## # Groups:   has_description [1]
##   has_description fraudulent     n    pct lbl
##             <dbl> <fct>      <int>  <dbl> <chr>
## 1               1 0          13604 0.951  95%
## 2               1 1            700 0.0489 5%
```

There are no listings without description.

It's 95 to 5 for non-fraudulent to fraudulent listings.

```
ggplot(tiff_df, aes(x = has_description, fill = fraudulent)) +
    geom_bar(position = "fill") +
```

```
  theme_classic() +
ggtitle("Fraudulent grouped by has_description")
```



employment_type

has_employment

```
# create binary variable on employment_type exist or not
tiff_df$has_employmenttype <- sapply(tiff_df$employment_type, function(f)
{as.numeric(!(is.na(f)))})
tiff_test_df$has_employmenttype <- sapply(tiff_test_df$employment_type,
function(f) {as.numeric(!(is.na(f)))})
```

look at ranking of employment_types

```
# tiff_df$employment_type[is.na(tiff_df$employment_type)] <- "NA"
rankedemployment <- as.data.frame(head(sort(table(tiff_df$employment_type),
decreasing=TRUE), 10))
# create ratio of listings
rankedemployment$ratio <- rankedemployment$Freq*100/nrow(tiff_df)
# create row for NA types
NAtypes <- data.frame("NAs", nrow(tiff_df)-(sum(rankedemployment$Freq)), 100-
(sum(rankedemployment$ratio)))
names(NAtypes) <- names(rankedemployment)
```

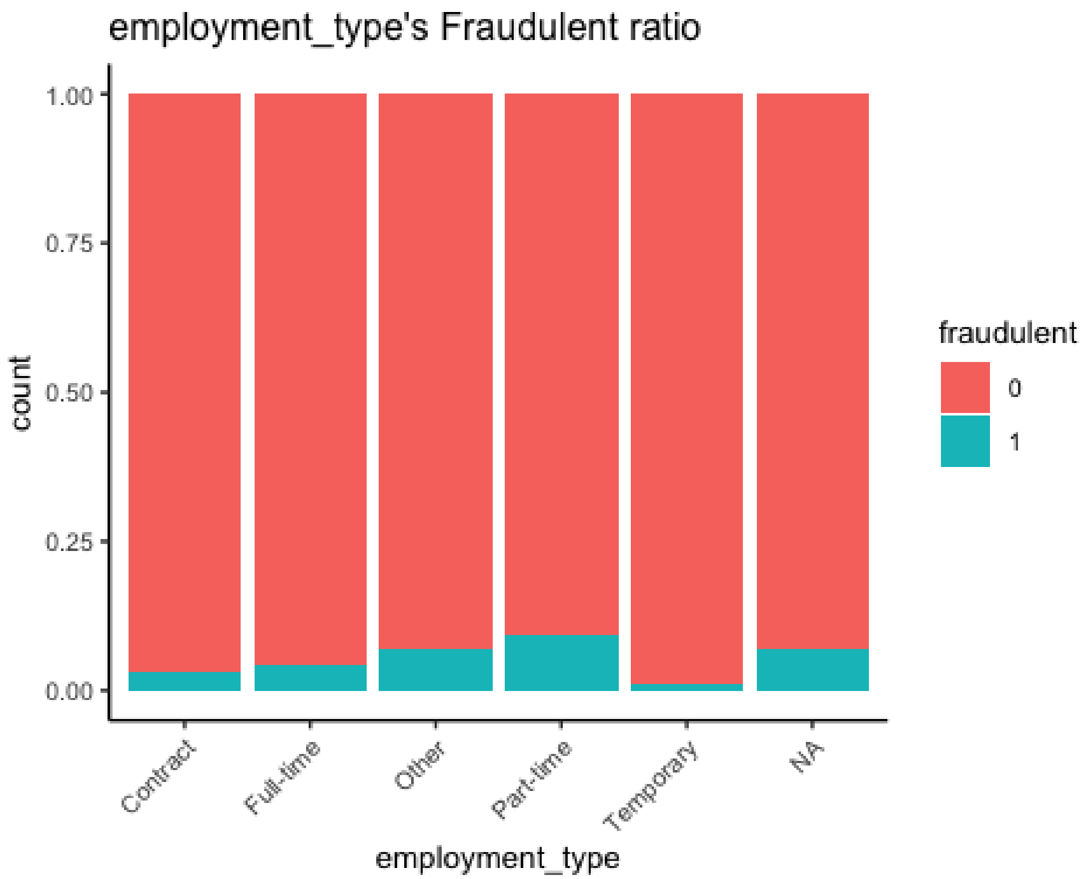
```r
# add NA types row to rankedemployment df
rankedemployment <- rbind(rankedemployment, NAtypes)
# sanity check
# sum(rankedemployment$Freq)
# nrow(tiff_df)
# sum(rankedemployment$ratio)

rankedemployment
```

```
##        Var1 Freq     ratio
## 1 Full-time 9281 64.883949
## 2  Contract 1205  8.424217
## 3 Part-time  642  4.488255
## 4 Temporary  188  1.314318
## 5     Other  175  1.223434
## 6       NAs 2813 19.665828
```

Majority of listings are Full-time employment types (64.99%). A lot of the listings also have employment_type not listed (~20%).

look at the employment_tyoes and their fraud percentages

```r
# df_topemployment <- subset(tiff_df, employment_type %in%
rankedemployment$Var1)
ggplot(tiff_df, aes(x = employment_type, fill = fraudulent)) +
    geom_bar(position = "fill") +
    theme_classic() +
  ggtitle("employment_type's Fraudulent ratio") +
  theme(text = element_text(size=10),
        axis.text.x = element_text(angle=45, hjust=1))
```



Listings with employment_type=part-time are more fraudluent than other specific employment types.

create dummy variables for employment_type

```
# dummy variable for employment type (6 types)
# "employment_type_Contract", "employment_type_Full-time",
"employment_type_Other", "employment_type_Part-time",
"employment_type_Temporary", "employment_type_NA"
tiff_df <- dummy_cols(tiff_df, select_columns = 'employment_type')
tiff_test_df<-dummy_cols(tiff_test_df,select_columns='employment_type')
#head(tiff_df)
```

sanity check of dummy variables for employment_types

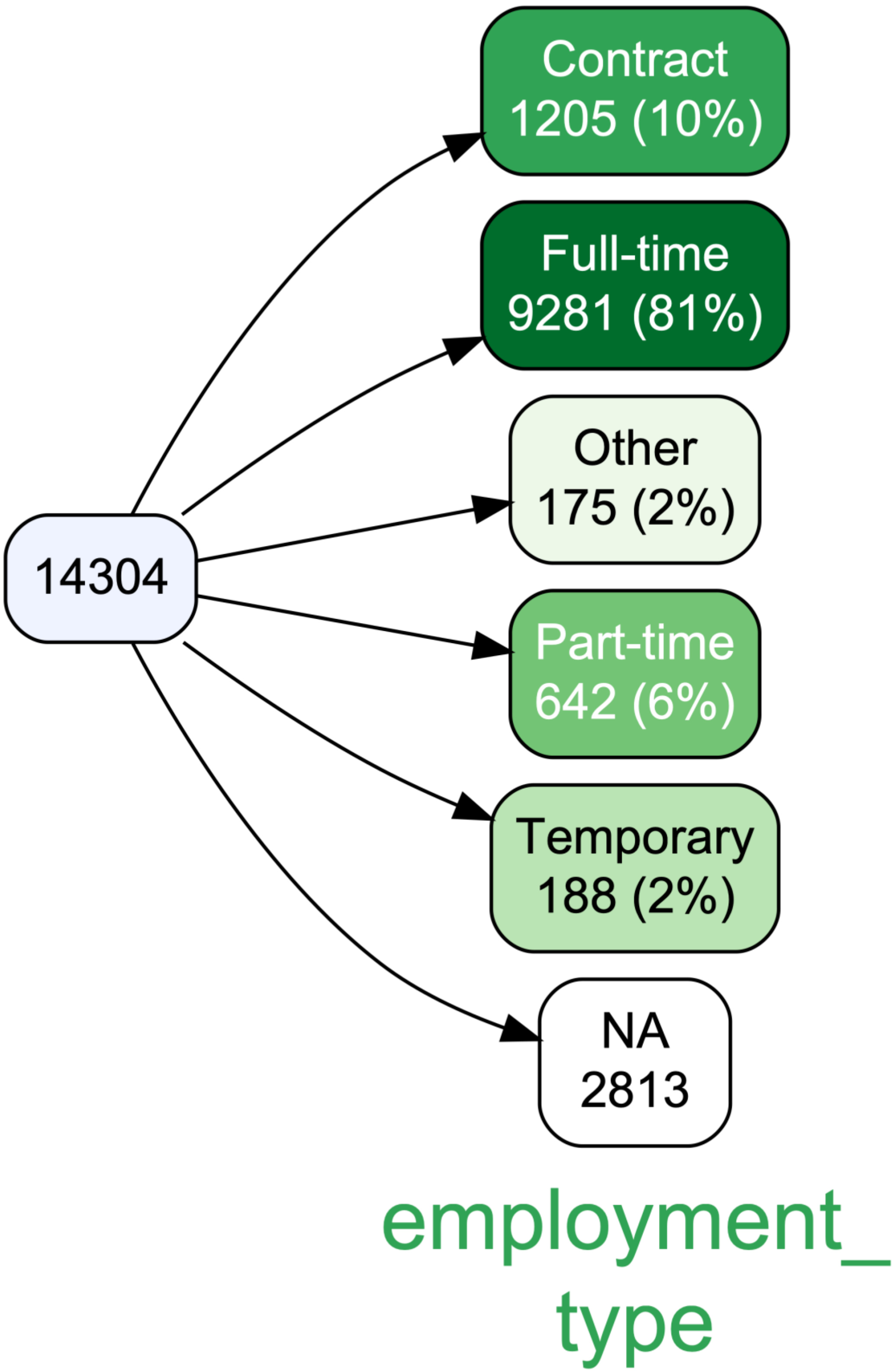
```
sum(is.na(tiff_df$employment_type))

## [1] 2813

nrow(tiff_df) - sum((tiff_df$has_employmenttype) )

## [1] 2813
```

plot count of fraudulent by employment_type

```r
tabyl(tiff_df, employment_type, fraudulent) %>%
  adorn_percentages("col") %>%
  adorn_pct_formatting(digits = 1)
```

```
##  employment_type     0     1
##         Contract  8.6%  5.1%
##        Full-time 65.3% 56.0%
##            Other  1.2%  1.7%
##        Part-time  4.3%  8.6%
##        Temporary  1.4%  0.3%
##             <NA> 19.2% 28.3%
```

```r
vtree(tiff_df, "employment_type", palette = 3, sortfill = TRUE)
```

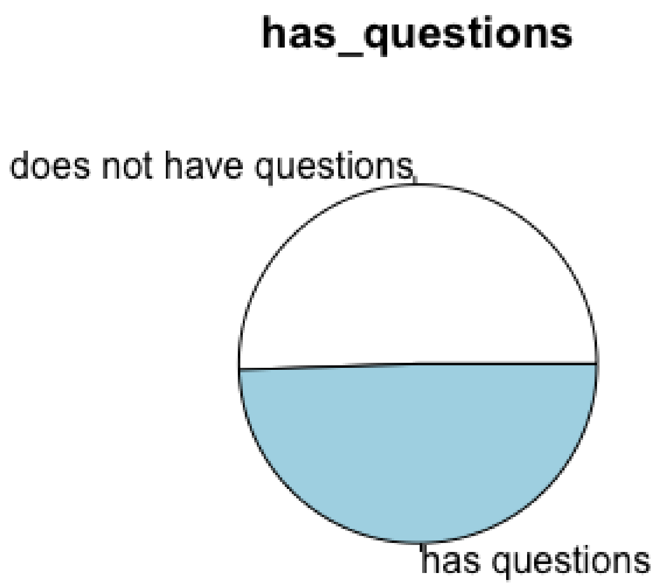

14304
Contract
1205 (10%)
Full-time
9281 (81%)
Other
175 (2%)
Part-time
642 (6%)
Temporary
188 (2%)
NA
2813
employment_
type

has_questions

visuals

About 50% of listings does DO have questions and the other 50 does not.

```
tiff_df2 <- tiff_df %>%
      mutate(has_questions = ifelse(has_questions == "1","has
questions","does not have questions"))
#counts
(typeCounts <- table(tiff_df2$has_questions))

##
## does not have questions           has questions
##                    7228                    7076

#percents
prop.table(typeCounts)

##
## does not have questions           has questions
##               0.5053132               0.4946868

#display
pie(typeCounts, main = "has_questions")
```

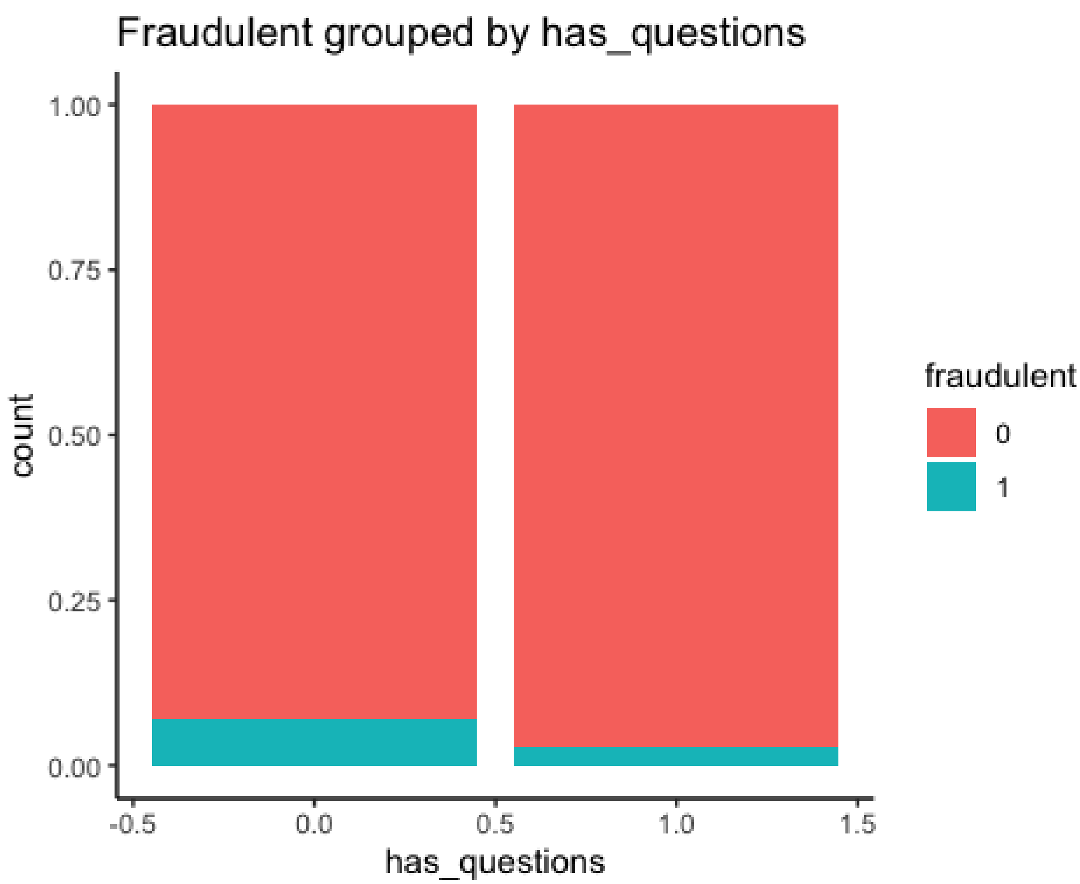



```
(plotdata <- tiff_df %>%
  group_by(has_questions, fraudulent) %>%
  summarize(n = n()) %>%
  mutate(pct = n/sum(n),
         lbl = scales::percent(pct)))

## `summarise()` has grouped output by 'has_questions'. You can override
using the
## `.groups` argument.

## # A tibble: 4 × 5
## # Groups:   has_questions [2]
##   has_questions fraudulent     n    pct lbl
##           <int> <fct>      <int>  <dbl> <chr>
## 1             0 0           6731 0.931  93%
## 2             0 1            497 0.0688 7%
## 3             1 0           6873 0.971  97%
## 4             1 1            203 0.0287 3%
```

For listings with questions, 3% of the listings are fraudulent. For listings without questions, 7% of the listings are fraudulent.

```
ggplot(tiff_df, aes(x = has_questions, fill = fraudulent)) +
    geom_bar(position = "fill") +
```

```
    theme_classic() +
  ggtitle("Fraudulent grouped by has_questions")
```




```
names(tiff_df)

##  [1] "index"                      "department"
##  [3] "description"                "has_questions"
##  [5] "employment_type"            "industry"
##  [7] "fraudulent"                 "dep_top"
##  [9] "dep_admin"                  "dep_engineering"
## [11] "dep_oil"                    "has_department"
## [13] "industry_top"               "industry_acc"
## [15] "industry_oilenergy"         "has_industry"
## [17] "cleaned_description"        "cleandescription_length"
## [19] "has_description"            "has_employmenttype"
## [21] "employment_type_Contract"   "employment_type_Full-time"
## [23] "employment_type_Other"      "employment_type_Part-time"
## [25] "employment_type_Temporary"  "employment_type_NA"
```

correlation plots

replace NAs in dummy variables

Adding Tiffany's features to the dataframe

drop the following attributes:

- index
- department
- description
- employment_type
- industry

```r
#head(tiff_df)
#view(tiff_df)
#view(tiff_test_df)
tiff_dfuseful <- subset(tiff_df, select=-c(index, department, description,
employment_type, industry))
#head(tiff_dfuseful)

tiff_dfuseful_test<-subset(tiff_test_df, select=-c(index, department,
description, employment_type, industry))
#write.csv(tiff_dfuseful,
"/Users/tiffwong/Desktop/csp571/project/tiff_attrs.csv", row.names = FALSE)

colnames(tiff_dfuseful)

##  [1] "has_questions"              "fraudulent"
##  [3] "dep_top"                    "dep_admin"
##  [5] "dep_engineering"            "dep_oil"
##  [7] "has_department"             "industry_top"
##  [9] "industry_acc"               "industry_oilenergy"
## [11] "has_industry"               "cleaned_description"
## [13] "cleandescription_length"    "has_description"
## [15] "has_employmenttype"         "employment_type_Contract"
## [17] "employment_type_Full-time" "employment_type_Other"
## [19] "employment_type_Part-time" "employment_type_Temporary"
## [21] "employment_type_NA"

new_cols<-colnames(tiff_dfuseful)[c(3:6,8:10,12:13,16:21)]
df_train<-cbind(df_train,tiff_dfuseful[,new_cols])

new_cols<-colnames(tiff_dfuseful_test)[c(3:6,8:10,12:13,16:21)]
df_test<-cbind(df_test,tiff_dfuseful_test[,new_cols])
```

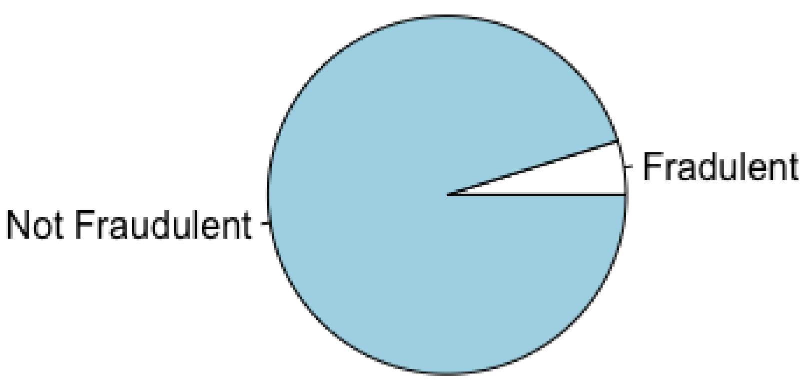
Alisha's EDA and Feature Engineering

```r
alisha_train<-df_train[c("telecommuting", "title", "salary_range",
"requirements", "required_experience", "fn", "fraudulent")]
#head(alisha_train)

alisha_test<-df_test[c("telecommuting", "title", "salary_range",
```

```
"requirements", "required_experience", "fn", "fraudulent")]
#head(alisha_test)
```

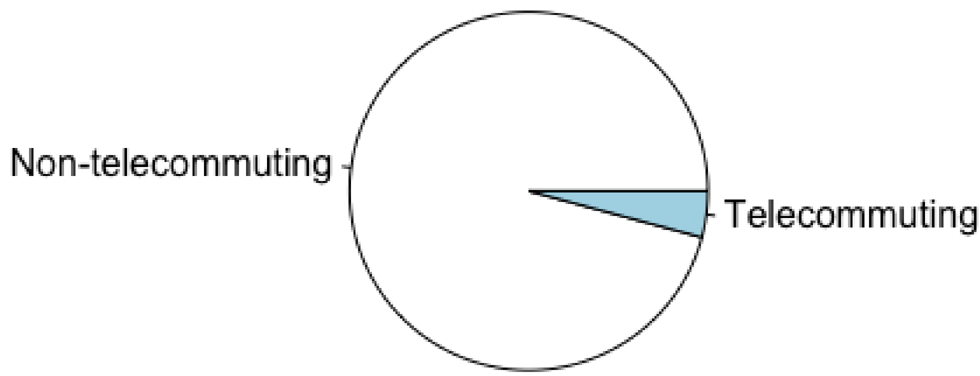
Fradulent

```
alisha_train$fraudulent<-as.factor(alisha_train$fraudulent)
alisha_train <- alisha_train %>%
      mutate(fraudulent = ifelse(fraudulent == "1","Fradulent","Not
Fraudulent"))
#counts
(typeCounts <- table(alisha_train$fraudulent))

##
##      Fradulent Not Fraudulent
##            700          13604

#percents
prop.table(typeCounts)

##
##      Fradulent Not Fraudulent
##     0.04893736     0.95106264

#display
pie(typeCounts)
```

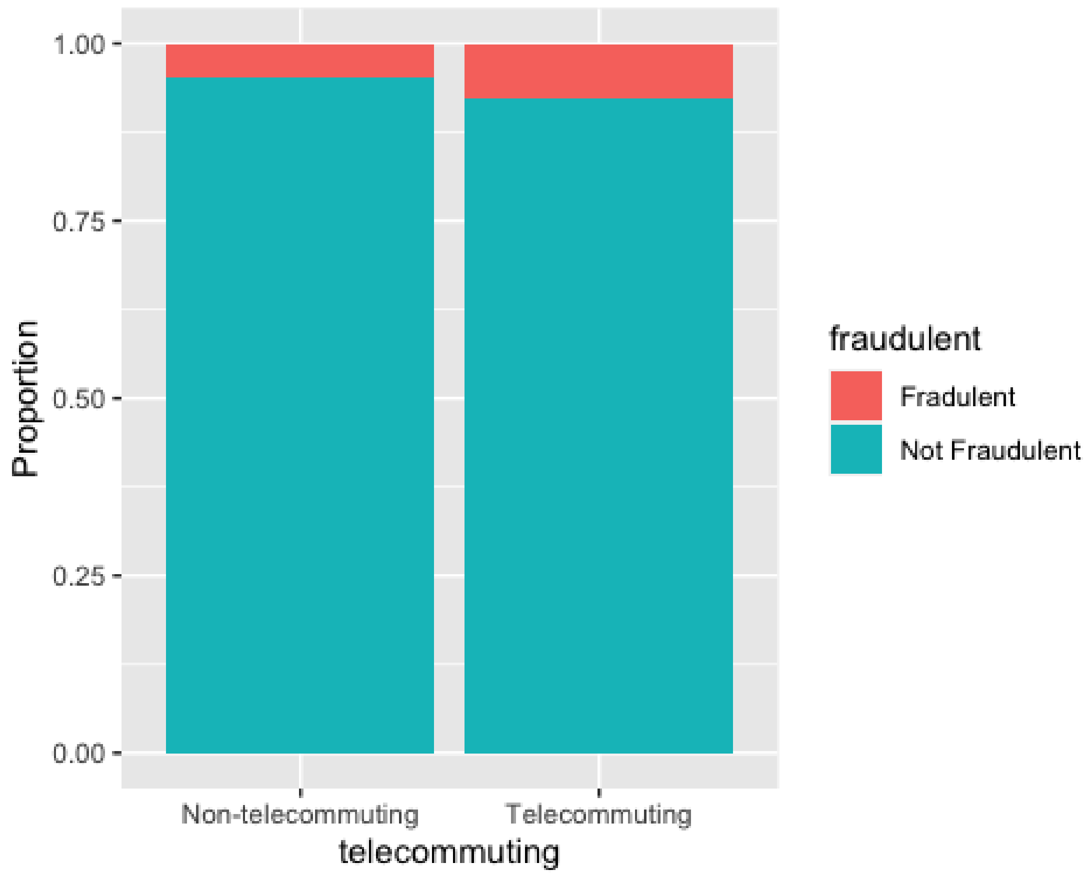


Telecommuting

```
alisha_train$telecommuting<-as.factor(alisha_train$telecommuting)
alisha_train <- alisha_train %>%
      mutate(telecommuting = ifelse(telecommuting ==
"1","Telecommuting","Non-telecommuting"))

alisha_test$telecommuting<-as.factor(alisha_test$telecommuting)
alisha_test <- alisha_test %>%
      mutate(telecommuting = ifelse(telecommuting ==
"1","Telecommuting","Non-telecommuting"))

#counts
(typeCounts <- table(alisha_train$telecommuting))

##
## Non-telecommuting     Telecommuting
##             13709               595

#percents
prop.table(typeCounts)

##
## Non-telecommuting     Telecommuting
##        0.95840324        0.04159676

#display
pie(typeCounts)
```




```
## # A tibble: 4 × 5
## # Groups:   telecommuting [2]
##   telecommuting     fraudulent         n    pct lbl
##   <chr>             <chr>          <int>  <dbl> <chr>
## 1 Non-telecommuting Fradulent        654 0.0477 5%
## 2 Non-telecommuting Not Fraudulent 13055 0.952  95%
## 3 Telecommuting     Fradulent         46 0.0773 8%
## 4 Telecommuting     Not Fraudulent   549 0.923  92%
```



Title

On brief observation, there are many forms of English Teacher Abroad, we should group all of these together

```
#normalizing to all lowercase
alisha_train$title=tolower(alisha_train$title)
alisha_train <- alisha_train %>%
  mutate(title = str_trim(title))
#head(summary(alisha_train$title))

alisha_test$title=tolower(alisha_test$title)
alisha_test <- alisha_test %>%
  mutate(title = str_trim(title))

alisha_train$title[grep(".*(english.*teacher|teacher.*english).*",alisha_trai
n$title)]<-"english teacher"
```

group customer service together

```
alisha_train$title[grep(".*(customer.*service|service.*customer).*",alisha_tr
ain$title)]<-"customer service"

#head(summary(alisha_train$title))
```

group managers together

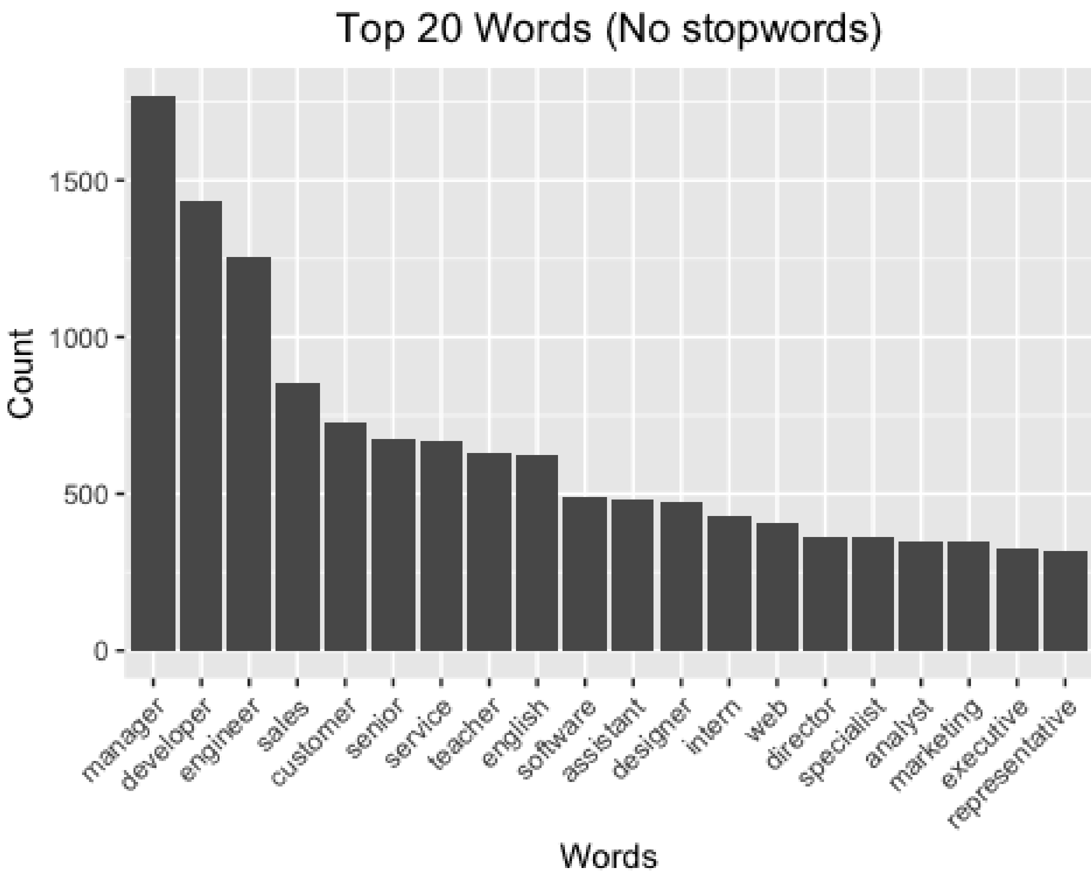
```
alisha_train$title[grep("manager",alisha_train$title)]<-"manager"
alisha_train$title[grep("assistant",alisha_train$title)]<-"assistant"
alisha_train$title[grep("intern",alisha_train$title)]<-"intern"
alisha_train$title<-as.factor(alisha_train$title)
#head(summary(alisha_train$title))

alisha_test$title[grep(".*(english.*teacher|teacher.*english).*",alisha_test$
title)]<-"english teacher"
```

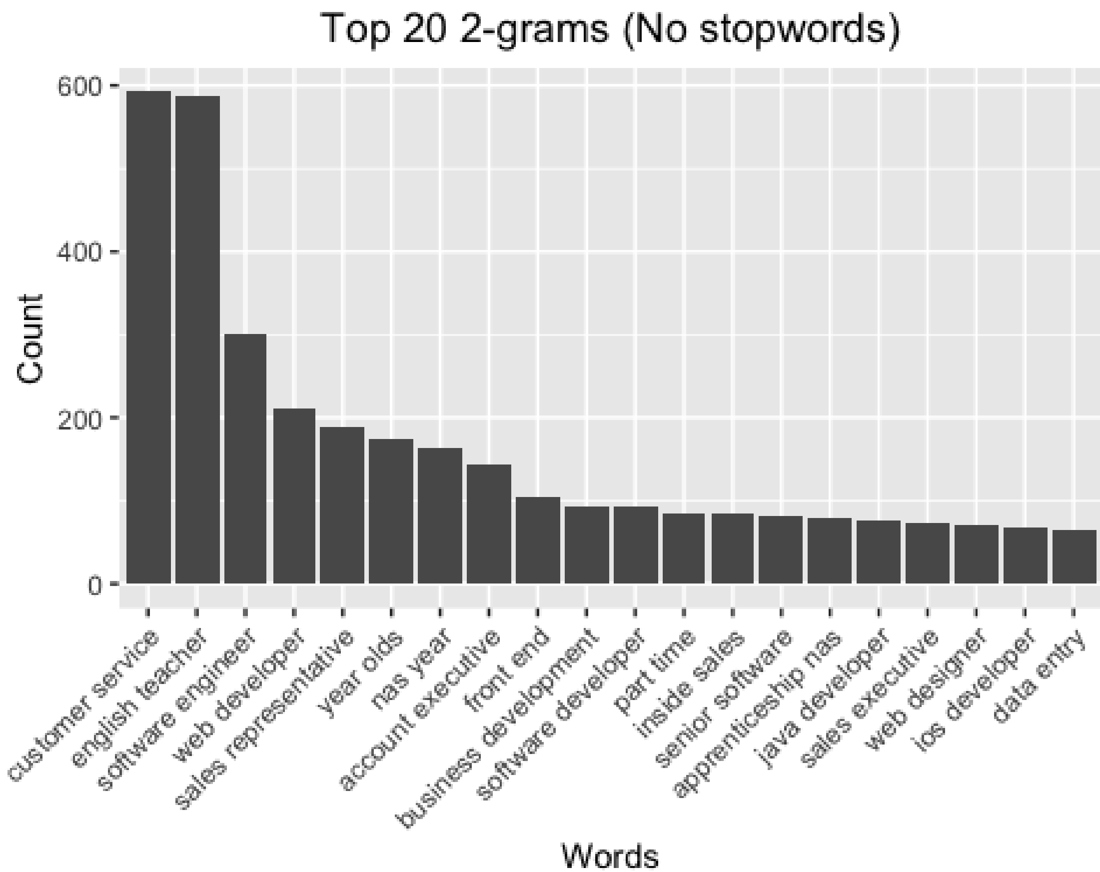
group customer service together

```
alisha_test$title[grep(".*(customer.*service|service.*customer).*",alisha_tes
t$title)]<-"customer service"

#head(summary(alisha_test$title))
```

group managers together

```
alisha_test$title[grep("manager",alisha_test$title)]<-"manager"
alisha_test$title[grep("assistant",alisha_test$title)]<-"assistant"
alisha_test$title[grep("intern",alisha_test$title)]<-"intern"
alisha_test$title<-as.factor(alisha_test$title)
#head(summary(alisha_test$title))
```


Top 20 Words (No stopwords)
Count
1500
1000
500
0
manager
developer
engineer
sales
customer
senior
service
teacher
english
software
assistant
designer
intern
web
director
specialist
analyst
marketing
executive
representative
Words



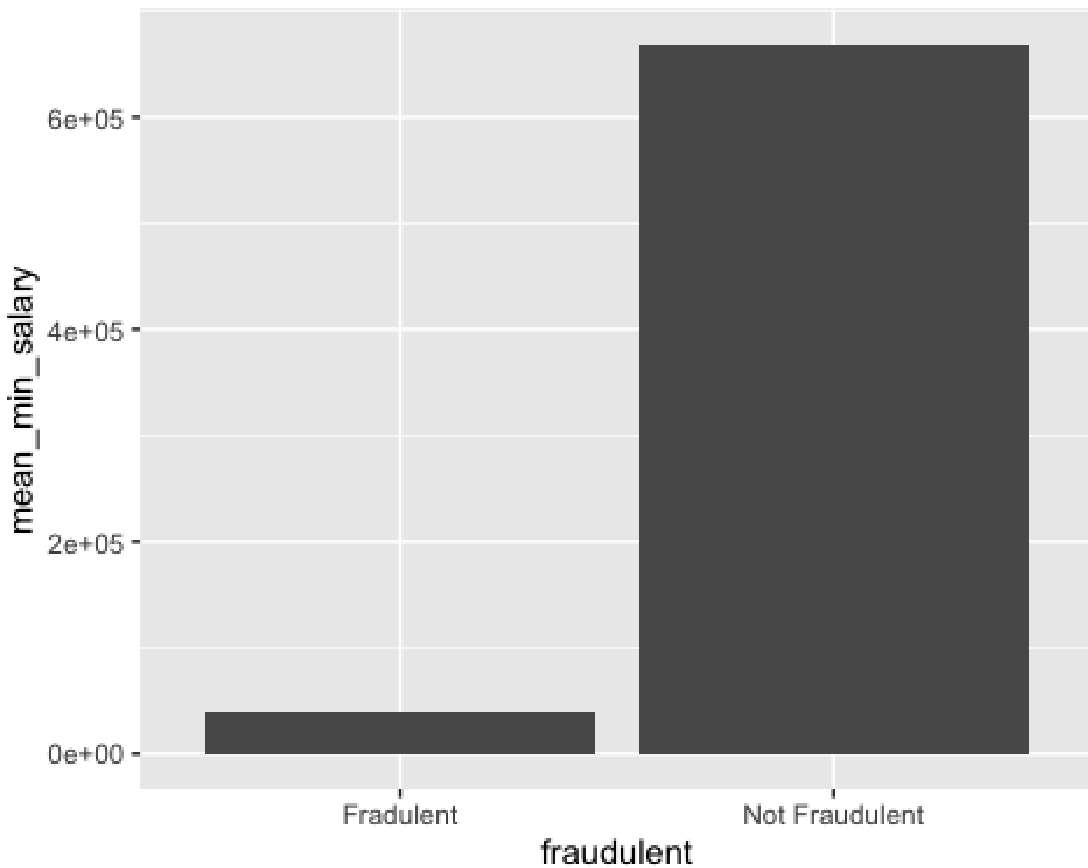
Creating Binary Variables for common values in the title column

```
binary<-function(df, col, vals){
  for (x in vals){
    new_col<-paste(col,x,sep="_")
    #print(new_col)
    df[[substitute(new_col)]]<-ifelse(grepl(x,df[[substitute(col)]]),1,0)
    #print(df[[substitute(new_col)]])
  }
  return(df)
}

alisha_train<-binary(alisha_train, "title", list("manager", "developer",
"engineer", "sales", "customer", "senior", "teacher", "assistant",
"software", "director", "intern"))
alisha_test<-binary(alisha_test, "title", list("manager", "developer",
"engineer", "sales", "customer", "senior", "teacher", "assistant",
"software", "director", "intern"))
```

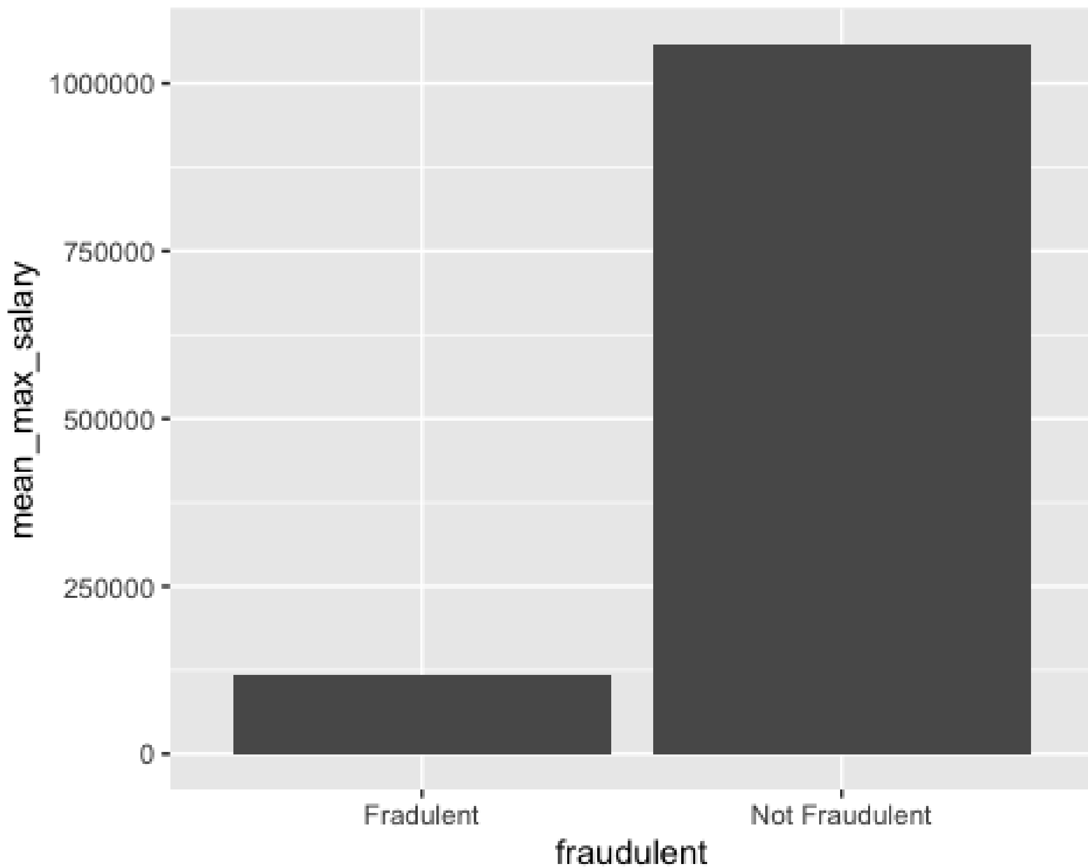
salary_range

There may be some outliers for salary. But this looks like a significant column

```
alisha_train$min_salary<-
as.numeric(sapply(str_split(alisha_train$salary_range,"-",2),`[`,1))

## Warning: NAs introduced by coercion

alisha_train$max_salary<-
as.numeric(sapply(str_split(alisha_train$salary_range,"-",2),`[`,2))

## Warning: NAs introduced by coercion

alisha_test$min_salary<-
as.numeric(sapply(str_split(alisha_test$salary_range,"-",2),`[`,1))

## Warning: NAs introduced by coercion

alisha_test$max_salary<-
as.numeric(sapply(str_split(alisha_test$salary_range,"-",2),`[`,2))

## Warning: NAs introduced by coercion

summary(alisha_train$min_salary)

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
##         0     18000     35000    621178     60000 800000000     12014

summary(alisha_train$max_salary)

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
## 0.000e+00 2.500e+04 5.000e+04 9.874e+05 9.000e+04 1.200e+09     12028

(plotdata<-alisha_train %>%
  group_by(fraudulent) %>%
  summarize(mean_min_salary=mean(min_salary,na.rm=TRUE)))

## # A tibble: 2 × 2
##   fraudulent     mean_min_salary
##   <chr>                    <dbl>
## 1 Fradulent               39019.
## 2 Not Fraudulent         669049.

ggplot(plotdata,
       aes(x = fraudulent,
           y = mean_min_salary)) +
  geom_bar(stat = "identity")
```




```
(plotdata<-alisha_train %>%
  group_by(fraudulent) %>%
  summarize(mean_max_salary=mean(max_salary,na.rm=TRUE)))

## # A tibble: 2 × 2
##   fraudulent     mean_max_salary
##   <chr>                    <dbl>
## 1 Fradulent              118419.
## 2 Not Fraudulent        1059340.

ggplot(plotdata,
       aes(x = fraudulent,
           y = mean_max_salary)) +
  geom_bar(stat = "identity")
```



required experience

```
alisha_train$required_experience<-
as.character(alisha_train$required_experience)
alisha_train$required_experience=alisha_train$required_experience%>%replace_n
a('missing')
alisha_train$required_experience<-as.factor(alisha_train$required_experience)
alisha_train<-dummy_cols(alisha_train, select_columns="required_experience")

alisha_test$required_experience<-
as.character(alisha_test$required_experience)
alisha_test$required_experience=alisha_test$required_experience%>%replace_na(
'missing')
alisha_test$required_experience<-as.factor(alisha_test$required_experience)
alisha_test<-dummy_cols(alisha_test, select_columns="required_experience")
```

fn

```
alisha_train$fn<-as.character(alisha_train$fn)
alisha_train$fn=alisha_train$fn%>%replace_na('missing')
alisha_train$fn<-as.factor(alisha_train$fn)
alisha_train<-dummy_cols(alisha_train, select_columns="fn")

alisha_test$fn<-as.character(alisha_test$fn)
alisha_test$fn=alisha_test$fn%>%replace_na('missing')
```

```
alisha_test$fn<-as.factor(alisha_test$fn)
alisha_test<-dummy_cols(alisha_test, select_columns="fn")
```

Adding Alisha's columns to the dataframe

```
alisha_train<-dummy_cols(alisha_train, select_columns="telecommuting")


alisha_test<-dummy_cols(alisha_test, select_columns="telecommuting")


alisha_train=select(alisha_train, -telecommuting, -title, -salary_range, -
requirements, -required_experience, -fn, -fraudulent)
alisha_test=select(alisha_test, -telecommuting, -title, -salary_range, -
requirements, -required_experience, -fn, -fraudulent)

df_train<-cbind(df_train,alisha_train)
df_train_numeric<-subset(df_train, select=-c(X, title, location, department,
salary_range, company_profile, description, requirements, benefits,
employment_type, required_experience, required_education, industry, fn,
region_cat, cleaned_description, index))

df_test<-cbind(df_test,alisha_test)
df_test_numeric<-subset(df_test, select=-c(X, title, location, department,
salary_range, company_profile, description, requirements, benefits,
employment_type, required_experience, required_education, industry, fn,
region_cat, cleaned_description, index))

write.csv(df_train,
"/Users/alishakhan/Desktop/School/FALL22/CSP571/project/joint.csv")

write.csv(df_train_numeric,
"/Users/alishakhan/Desktop/School/FALL22/CSP571/project/joint_numeric.csv")

write.csv(df_test,
"/Users/alishakhan/Desktop/School/FALL22/CSP571/project/joint_test.csv")

write.csv(df_test_numeric,
"/Users/alishakhan/Desktop/School/FALL22/CSP571/project/joint_test_numeric.cs
v")

#head(df_test)
#head(df_train)
```