The Relationship Between Job Characteristics and the Probability of Requesting Python Skills

Robert Hand

2022

Executive Summary

How do different attributes of a job in the data science and statistics space relate to the probablity that the job will want or require Python skills?

There is a dataset available (https://www.kaggle.com/datasets/rashikrahmanpritom/data-science-job-posting-on-glassdoor) which was collected from about 600 Glassdoor job postings for positions in data science, analytics, data engineering, statistics and similar roles. The data set provides information about the job, the company, and which software or programming languages are requested from a candidate. I was interested in modeling the relationship between whether a job would want a candidate to know Python and other job characteristics such as salary, industry, other skills, etc.

The model was a GLMM, a logistic regression mixed effect model, with the job title groups modeled as a random effect and other included predictors modeled as fixed effects. This was done because this model had similar performance but very slightly better to the fixed effects model alone with the same covariates in terms of predictive accuracy, a likelihood ratio test supported the random effect inclusion, and because EDA showed that it was a reasonable choice to model the job titles found in the data as a random effect.

MODEL: Logit(p(Python=1)) ~ (1 | job_title category) -5.0 + 0.92(Glassdoor_Company_Rating) + 0.0059(Salary) + 1.1(sql=requested) + 0.73(tableau=requested + 1.95(spark=requested)

Random Effect Modeled as N(0, Tauˆ2=1.6). observed jobs falling into 7 job titles

All predictors in the model are significant at least at alpha = 0.05 expect for salary, which had a p-value of 0.13, however, it was kept since it still seemed to contribute to the model and since it was selected via best subsets selection, the fixed efffect only model, and since it is practically speaking an important covariate.

The model had a 76% with fairly balanced sensitivity and specificity.

KEY FINDINGS:

Conditional on a particular job title grouping, jobs with higher salaries and better company ratings are more likely to desire Python skills.

If a job desires SQL, Tableau, or Spark skills, the odds of Python also increase considerably.

Overall, hopefully the takeaways from this model can aid in a better understanding of how requested programming skills relate to job attributes for someone in an early to mid career stage in this space.

FUTURE DIRECTIONS: - Including R programming in the model somehow. Not in original dataset.

- I was interested in explored a mixed effect model here and understanding the relationships more than classification per se. Other modeling and classification methods would be interesting to apply on this same data set, such as SVM, Random Forests, or KNN classification. An elastic not could also be applied.

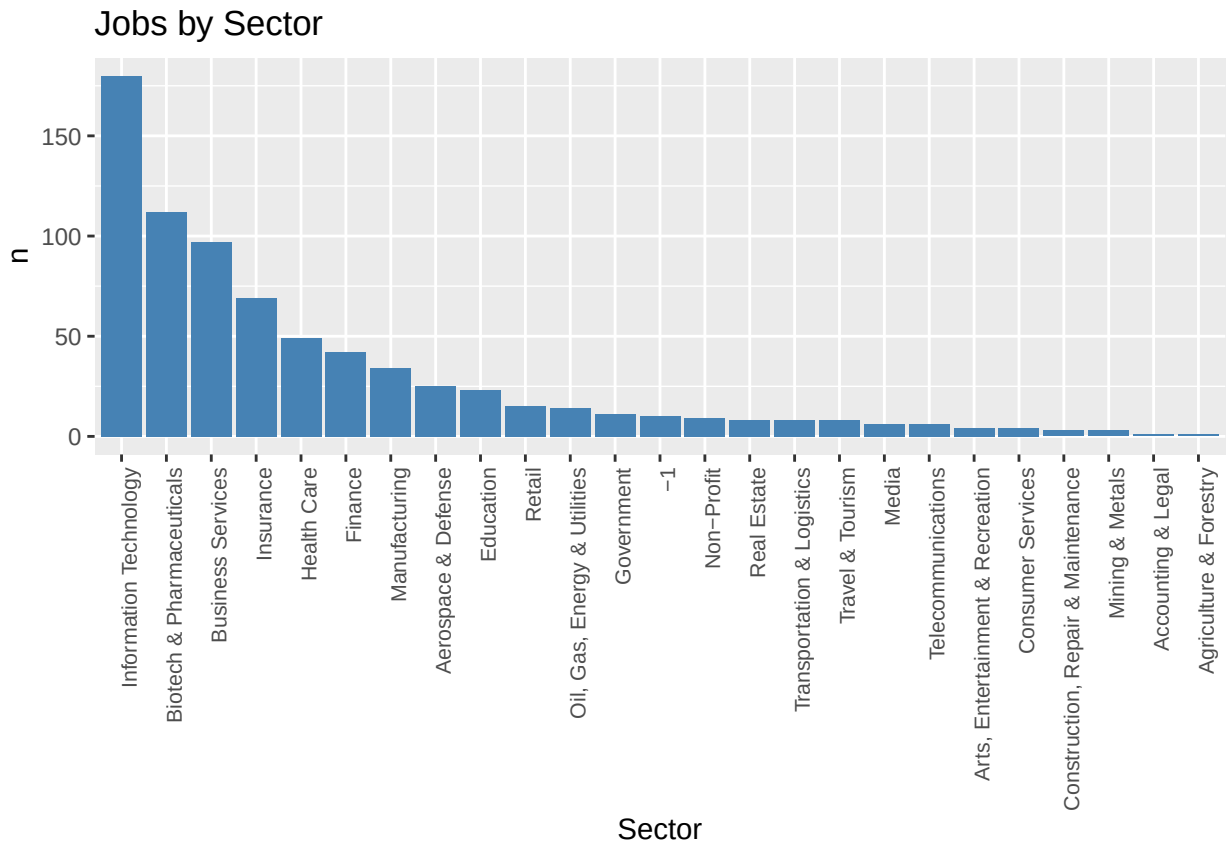
-Using k-fold cross validation for a more predictive approach to the problem.

Initial exploration, reconstruction, and which variables to include.

```
knitr::opts_chunk$set(echo = TRUE, fig.width = 15, fig.height = 15)

#read data set
df <- read.csv('data_scientist_salary_data.csv', header = TRUE, stringsAsFactors = FALSE)


#do we care about keeping all the smaller sectors in this analysis?
df %>% select(Sector) %>% group_by(Sector) %>% count() %>% arrange(desc(n)) %>% kable()
```

Sector	n
Information Technology	180
Biotech & Pharmaceuticals	112
Business Services	97
Insurance	69
Health Care	49
Finance	42
Manufacturing	34
Aerospace & Defense	25
Education	23
Retail	15
Oil, Gas, Energy & Utilities	14
Government	11
-1	10
Non-Profit	9
Real Estate	8
Transportation & Logistics	8
Travel & Tourism	8
Media	6
Telecommunications	6
Arts, Entertainment & Recreation	4
Consumer Services	4
Construction, Repair & Maintenance	3
Mining & Metals	3
Accounting & Legal	1
Agriculture & Forestry	1

```
df %>% select(Sector) %>% group_by(Sector) %>% count() %>% arrange(desc(n)) %>% ggplot(aes(x=reorder(Se
```




```
#what industries are included in some of these? Industry is nested within sector.
df %>% select(Sector, Industry) %>% filter(Sector =="Manufacturing") %>% group_by(Industry) %>% count()
```

Industry	n
Consumer Products Manufacturing	20
Food & Beverage Manufacturing	8
Health Care Products Manufacturing	1
Industrial Manufacturing	4
Transportation Equipment Manufacturing	1

```
df %>% select(Sector, Industry) %>% filter(Sector =="Aerospace & Defense") %>% group_by(Industry) %>% co
```

Industry	n
Aerospace & Defense	25

```
df %>% select(Sector, Industry) %>% filter(Sector =="Business Services") %>% group_by(Industry) %>% cou
```

Industry	n
Advertising & Marketing	25

Industry	n
Architectural & Engineering Services	4
Consulting	29
Research & Development	19
Security Services	7
Staffing & Outsourcing	10
Wholesale	3

```
df %>% select(Sector, Industry) %>% filter(Sector =="Transportation & Logistics") %>% group_by(Industry
```

Industry	n
Logistics & Supply Chain	4
Transportation Management	3
Trucking	1

```
df %>% select(Sector, Industry) %>% filter(Sector =="Oil, Gas, Energy & Utilities") %>% group_by(Industr
```

Industry	n
Energy	14

Based on these, going to create a new grouping "manufacturing and industrial production and add transportation and logistics to the business services category. Drop the remaining sectors that do not have many entries to focus on larger ones.

```
df <- df %>% mutate(Sector = case_when(
  Sector %in% c("Transportation & Logistics")~"Business Services",
  Sector %in% c("Aerospace & Defense","Construction, Repair & Maintenance","Oil, Gas, Energy & Utilities
  TRUE ~Sector
))

#remove the ones we don't need
df <- df[df$Sector %in% c("Biotech & Pharmaceuticals","Information Technology","Business Services","Insu
```

```
#check the balance of Python response variable:
df %>% select(Python) %>% count(Python) %>% kable()
```

Python	n
0	304
1	333

```
#basically an even split of yes and no.

#Job titles. Dropping the directors and project managers so it is standardized, not comparing manageria
df %>% select(job_title_sim) %>% count(job_title_sim) %>% kable()
```

job_title_sim	n
analyst	75
data analitics	8
data engineer	106
data modeler	5
data scientist	271
Data scientist project manager	16
director	5
machine learning engineer	14
na	9
other scientist	128

```
df <- df[!df$job_title_sim %in% c("director", "Data scientist project manager","na"),]
```

Restructuring salary so average of the upper and lower end of the salary estimate given, and converting the hourly wages to annual salary estimates.

```
df %>%  select(Salary.Estimate) %>% count(Salary.Estimate) %>% head() %>% kable()
```

Salary.Estimate	n
$10-$17 Per Hour(Glassdoor est.)	2
$100K-$160K (Glassdoor est.)	1
$100K-$166K (Glassdoor est.)	2
$100K-$173K (Glassdoor est.)	2
$100K-$190K (Glassdoor est.)	2
$102K-$163K (Glassdoor est.)	1

```
# Many more of these. Salary is given as a range salary estimate, in categories. I think it makes more
#changing to numeric.

#which ones are in hourly terms?
df %>% select(Salary.Estimate) %>% filter(str_detect(Salary.Estimate,"Hour")==TRUE) %>% kable()
```

Salary.Estimate
$17-$24 Per Hour(Glassdoor est.)
$21-$34 Per Hour(Glassdoor est.)
$18-$25 Per Hour(Glassdoor est.)
$21-$34 Per Hour(Glassdoor est.)
$17-$24 Per Hour(Glassdoor est.)
$21-$34 Per Hour(Glassdoor est.)
$18-$25 Per Hour(Glassdoor est.)
$24-$39 Per Hour(Glassdoor est.)
$21-$34 Per Hour(Glassdoor est.)
Employer Provided Salary:$25-$28 Per Hour
$21-$29 Per Hour(Glassdoor est.)
$10-$17 Per Hour(Glassdoor est.)
$18-$25 Per Hour(Glassdoor est.)
$24-$39 Per Hour(Glassdoor est.)

Salary.Estimate
$21-$34 Per Hour(Glassdoor est.)
Employer Provided Salary:$25-$28 Per Hour
$21-$29 Per Hour(Glassdoor est.)
$10-$17 Per Hour(Glassdoor est.)
$27-$47 Per Hour(Glassdoor est.)
$18-$25 Per Hour(Glassdoor est.)
$24-$39 Per Hour(Glassdoor est.)
$21-$34 Per Hour(Glassdoor est.)
Employer Provided Salary:$25-$28 Per Hour

```
#make a copy
df <- df %>% mutate(Salary.Estimate2 = Salary.Estimate)

#convert to a number that is average of the upper and lower ends of the salary estimate.
df$Salary.Estimate <- gsub("[^0-9-]", "", as.character(df$Salary.Estimate))
df <- separate(df, Salary.Estimate, c("lower","upper"), sep = "-", remove = FALSE)
df <- df %>% mutate(salary = (as.numeric(lower) + as.numeric(upper))/2)

#change hourly to annual salary. Base on assumption of 40 hr work week, 52 weeks in year, and divide by
df <- df %>% mutate(salary = case_when(
  str_detect(Salary.Estimate2,"Hour")==TRUE~salary*(40*52/1000),
  TRUE ~ salary
))
#check
df %>% select(Salary.Estimate2,salary) %>% filter(str_detect(Salary.Estimate2,"Hour")==TRUE) %>% kable(
```

Salary.Estimate2	salary
$17-$24 Per Hour(Glassdoor est.)	42.64
$21-$34 Per Hour(Glassdoor est.)	57.20
$18-$25 Per Hour(Glassdoor est.)	44.72
$21-$34 Per Hour(Glassdoor est.)	57.20
$17-$24 Per Hour(Glassdoor est.)	42.64
$21-$34 Per Hour(Glassdoor est.)	57.20
$18-$25 Per Hour(Glassdoor est.)	44.72
$24-$39 Per Hour(Glassdoor est.)	65.52
$21-$34 Per Hour(Glassdoor est.)	57.20
Employer Provided Salary:$25-$28 Per Hour	55.12
$21-$29 Per Hour(Glassdoor est.)	52.00
$10-$17 Per Hour(Glassdoor est.)	28.08
$18-$25 Per Hour(Glassdoor est.)	44.72
$24-$39 Per Hour(Glassdoor est.)	65.52
$21-$34 Per Hour(Glassdoor est.)	57.20
Employer Provided Salary:$25-$28 Per Hour	55.12
$21-$29 Per Hour(Glassdoor est.)	52.00
$10-$17 Per Hour(Glassdoor est.)	28.08
$27-$47 Per Hour(Glassdoor est.)	76.96
$18-$25 Per Hour(Glassdoor est.)	44.72
$24-$39 Per Hour(Glassdoor est.)	65.52
$21-$34 Per Hour(Glassdoor est.)	57.20

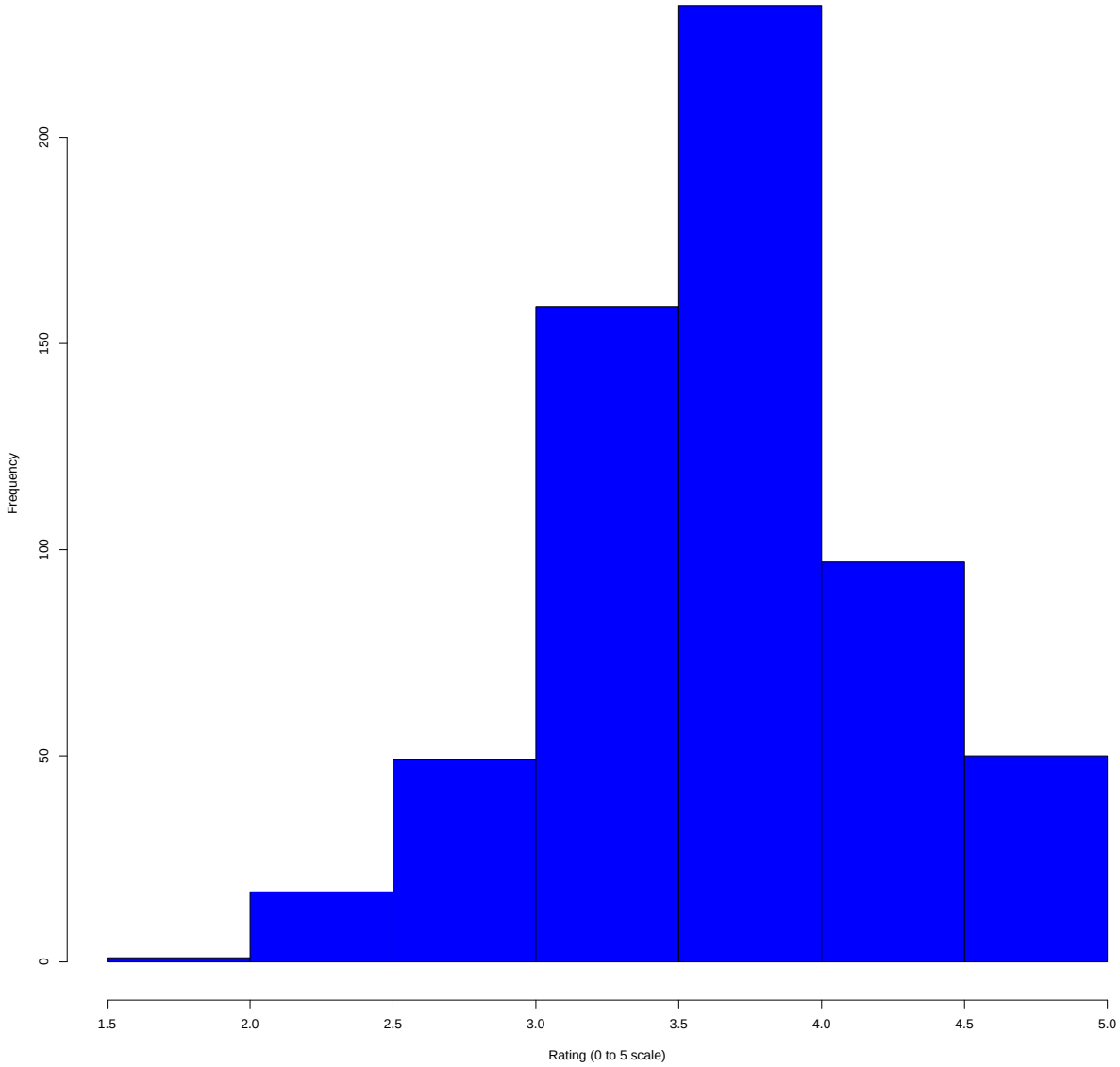
Salary.Estimate2	salary
Employer Provided Salary:$25-$28 Per Hour	55.12

Other Variables

```
knitr::opts_chunk$set(echo = TRUE, fig.width = 15, fig.height = 15)
#Rating
df %>% select(Rating) %>% count(Rating) %>% kable()
```

Rating	n
-1.0	2
1.9	1
2.1	5
2.2	1
2.3	2
2.4	7
2.5	2
2.7	14
2.8	6
2.9	16
3.0	13
3.1	21
3.2	24
3.3	39
3.4	36
3.5	39
3.6	27
3.7	58
3.8	44
3.9	63
4.0	40
4.1	17
4.2	18
4.3	25
4.4	31
4.5	6
4.6	10
4.7	29
4.8	6
5.0	5

```
df <- df[!df$Rating==-1,]
hist(df$Rating,col = "blue",main = "Histogram of Glassdoor Rating of Company",xlab = "Rating (0 to 5 sca
```




```
df %>% select(Size) %>% count(Size) %>% kable()
```

Size	n
10000+	112
1001 - 5000	120
201 - 500	103
5001 - 10000	58
501 - 1000	106
51 - 200	79
Jan-50	25
unknown	2

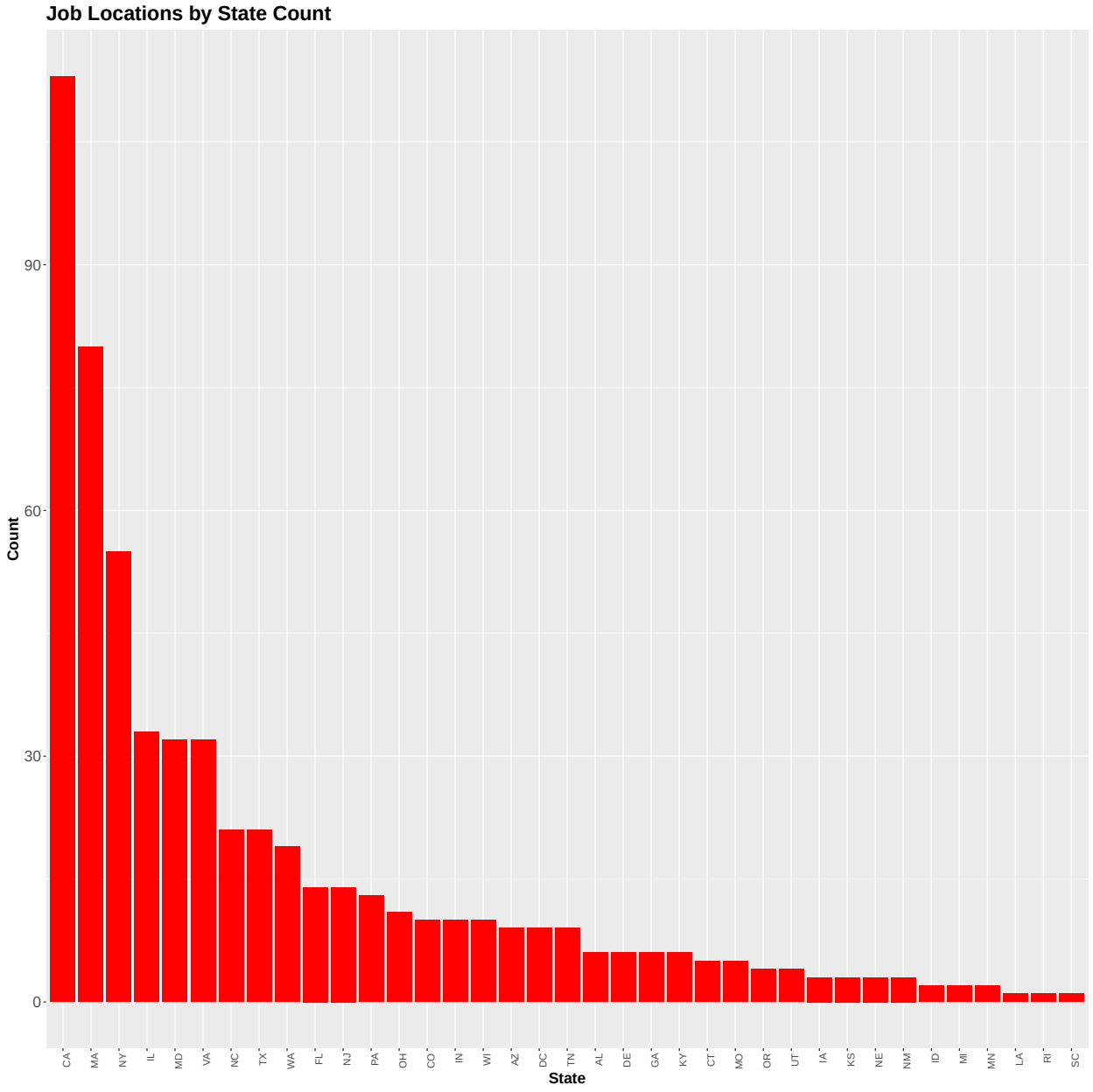
```
#drop the 2 unknowns
df <- df[!df$Size=='unknown',]

#founded
df %>% select(Founded) %>% count(Founded) %>% head() %>% kable()
```

Founded	n
-1	25
1781	14
1812	1
1830	4
1849	6
1850	1

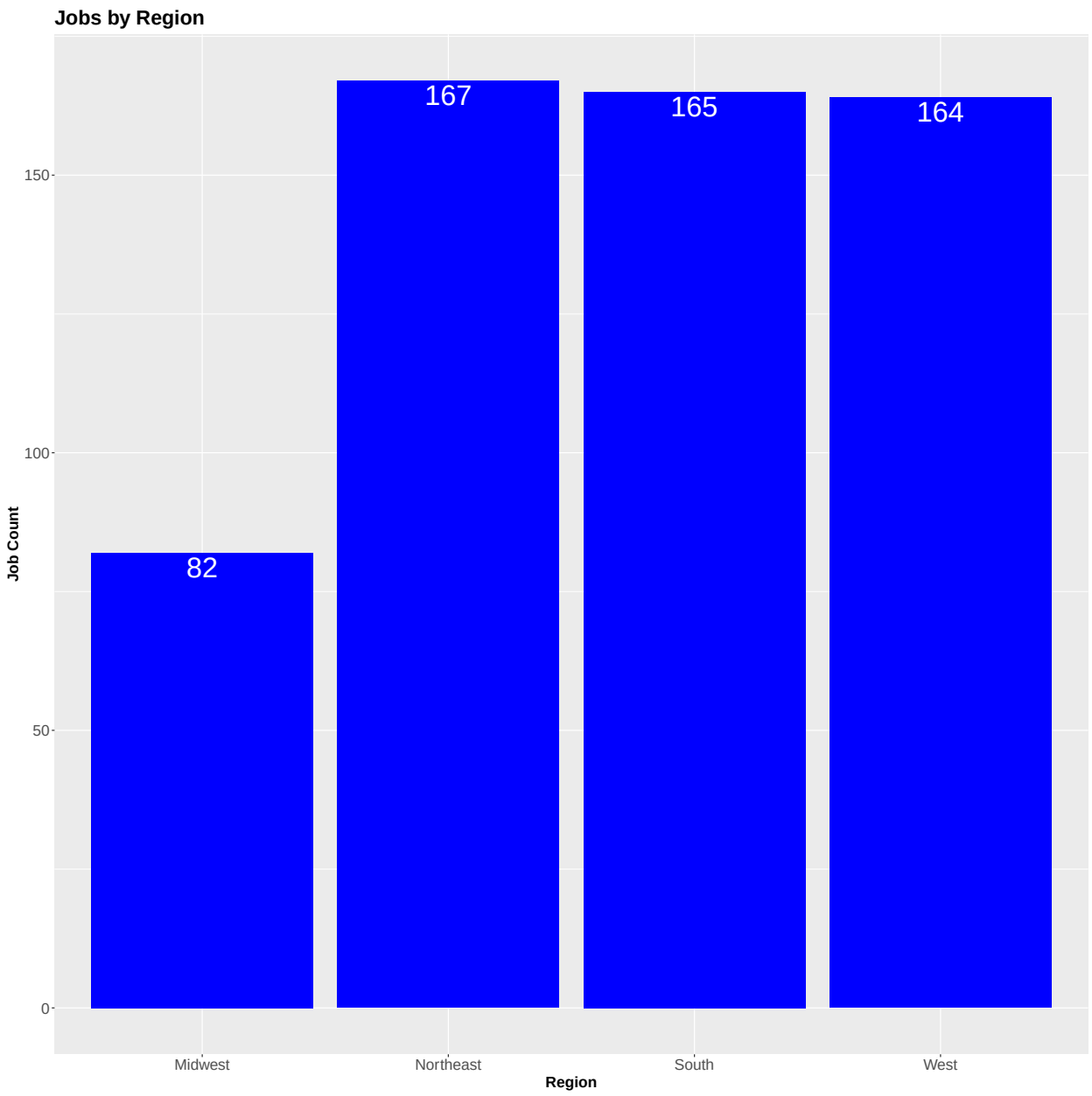
```
#remove the -1s.
df <- df[!df$Founded==-1,]
```

Transforming the sate location into four census bureau geographic regions

```
knitr::opts_chunk$set(echo = TRUE, fig.width = 15, fig.height = 15)
locations <- df %>% select(Job.Location) %>% count(Job.Location)
ggplot(locations, aes(x=reorder(Job.Location,-n), y=n)) + geom_bar(stat = "identity", fill = "red") + gg
axis.title=element_text(size=15,face="bold")) + theme(plot.title = element_text(size = 20, face = "bold
```




```r
#transform to regions based on four Census Bureau regions.
df <- df %>% mutate(
  region = case_when(
    df$Job.Location %in% c("WA","OR","CA","NV","AZ","UT","CO","NM","WY","MT","ID")~"West",
    df$Job.Location %in% c("ND","MN","SD","NE","KS","MO","IA","IL","IN","OH","WI","MI")~"Midwest",
    df$Job.Location %in% c("ME","VT","NH","MA","CT","NY","NJ","PA")~"Northeast",
    TRUE ~"South"
  )
)

df %>% select(region) %>% group_by(region) %>% count() %>% ggplot(aes(y=n, x=region)) + geom_bar(stat="
axis.title=element_text(size=15,face="bold")) + theme(plot.title = element_text(size = 20, face = "bold
```



Drop jobs requiring a phd, since aim of this is for an audience with a master's/bachelor's degree.

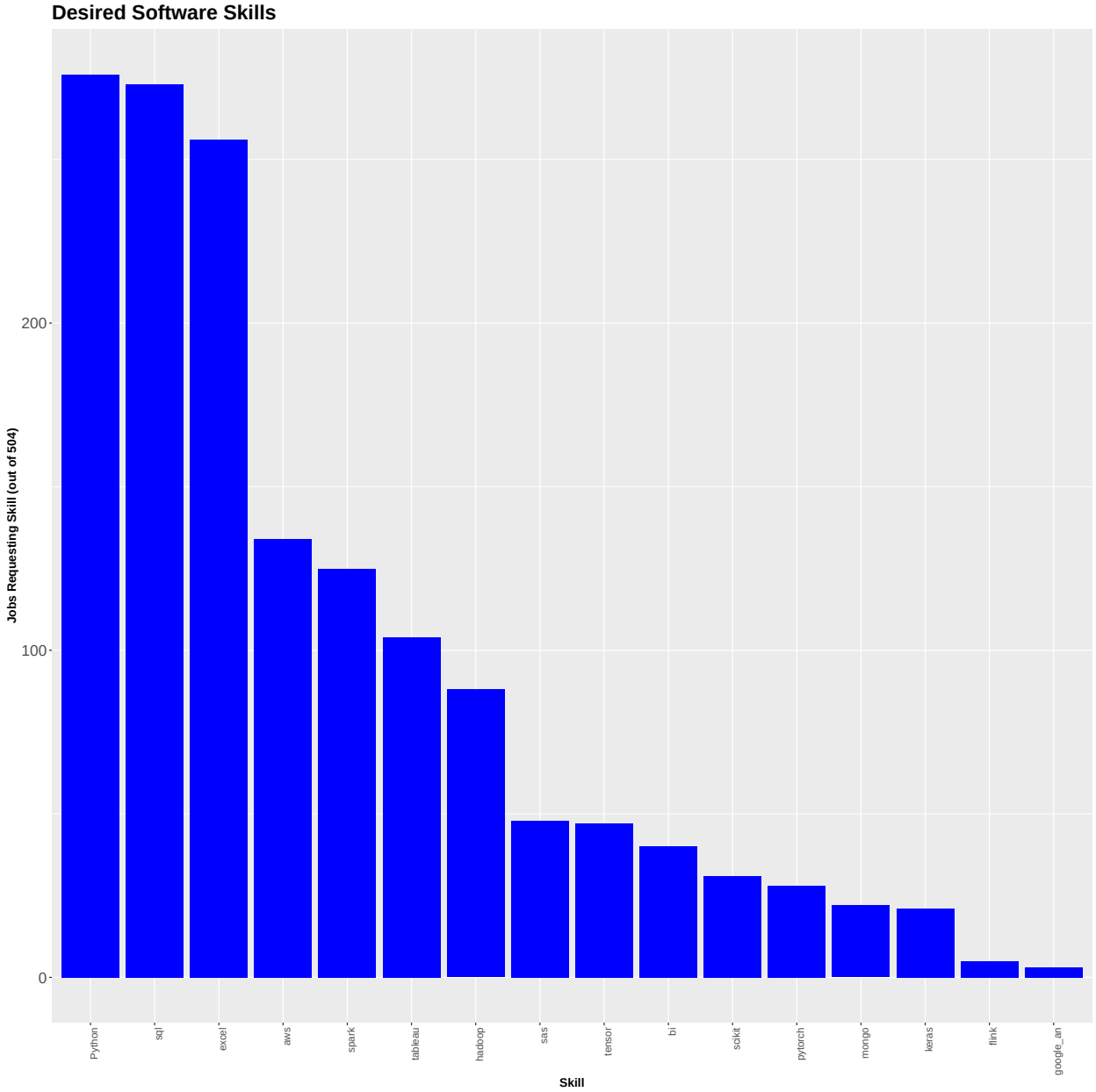
```
df %>% select(Degree) %>% group_by(Degree) %>% count() %>% kable()
```

Degree	n
M	204
na	300
P	74

```
df <- df[!df$Degree=="P",]
```

Look at how many jobs requested other skills listed as binary variables besides Python.

```
knitr::opts_chunk$set(echo = TRUE, fig.width = 15, fig.height = 15)
#look at how many jobs requested other skills listed as binary variables.
counts_want_skill <- apply(df[,26:41], 2, sum)
counts_want_skill<-as.data.frame(as.table(counts_want_skill), stringsAsFactors = FALSE)
counts_want_skill <- counts_want_skill[order(counts_want_skill$Freq, decreasing = TRUE),]
ggplot(counts_want_skill, aes(x=reorder(Var1,-Freq), y=Freq)) + geom_bar(stat = "identity", fill = "blu
axis.title=element_text(size=12,face="bold")) + theme(plot.title = element_text(size = 20, face = "bold
```



Based on this, I want to look at the larger ones, SQL, Tableau, aws, spark, hadoop, and sas. also are going to look at salary, region, sector, job rating, how new the company is (founded), and job title.

Last variables to check.

```
df %>% select(Type.of.ownership) %>% group_by(Type.of.ownership) %>% count() %>% kable()
```

Type.of.ownership	n
Company - Private	302
Company - Public	139
Government	10
Hospital	7
Nonprofit Organization	27
Other Organization	4
Subsidiary or Business Segment	15

```
#company ownership groups are sort of mixed and unclear, so not going to look at it.
df %>% select(Employer.provided) %>% group_by(Employer.provided) %>% count() %>% kable()
```

Employer.provided	n
0	499
1	5

```
#this variable seems plainly wrong. It doesn't seems like only five of these jobs would have employer p
#much insight for the model.
```

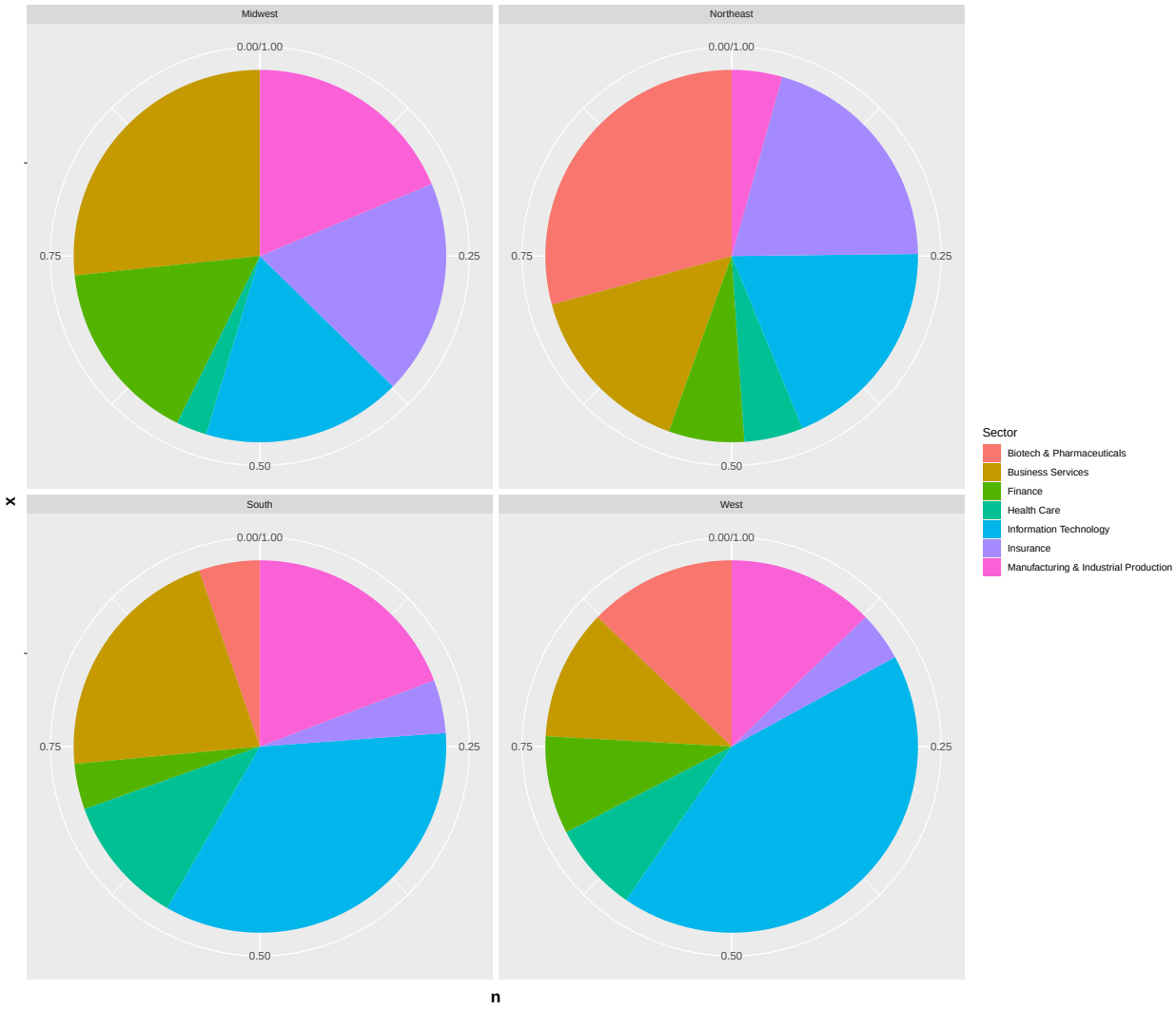
based on this, I want to look at the larger ones, SQL, Tableau, aws, spark, hadoop, and sas. also are going to look at salary, region, sector, job rating, and job title.

Not looking at the software skills with low counts, other variables such as upper, lower salary are captured by salary, company text, headquarters, and job description are uniquely specific text strings that don't fit this analysis. Since this is geared towards things that could be helpful to someone applying to jobs in this field, and for simplicity, also leaving out size and revenue of the company.

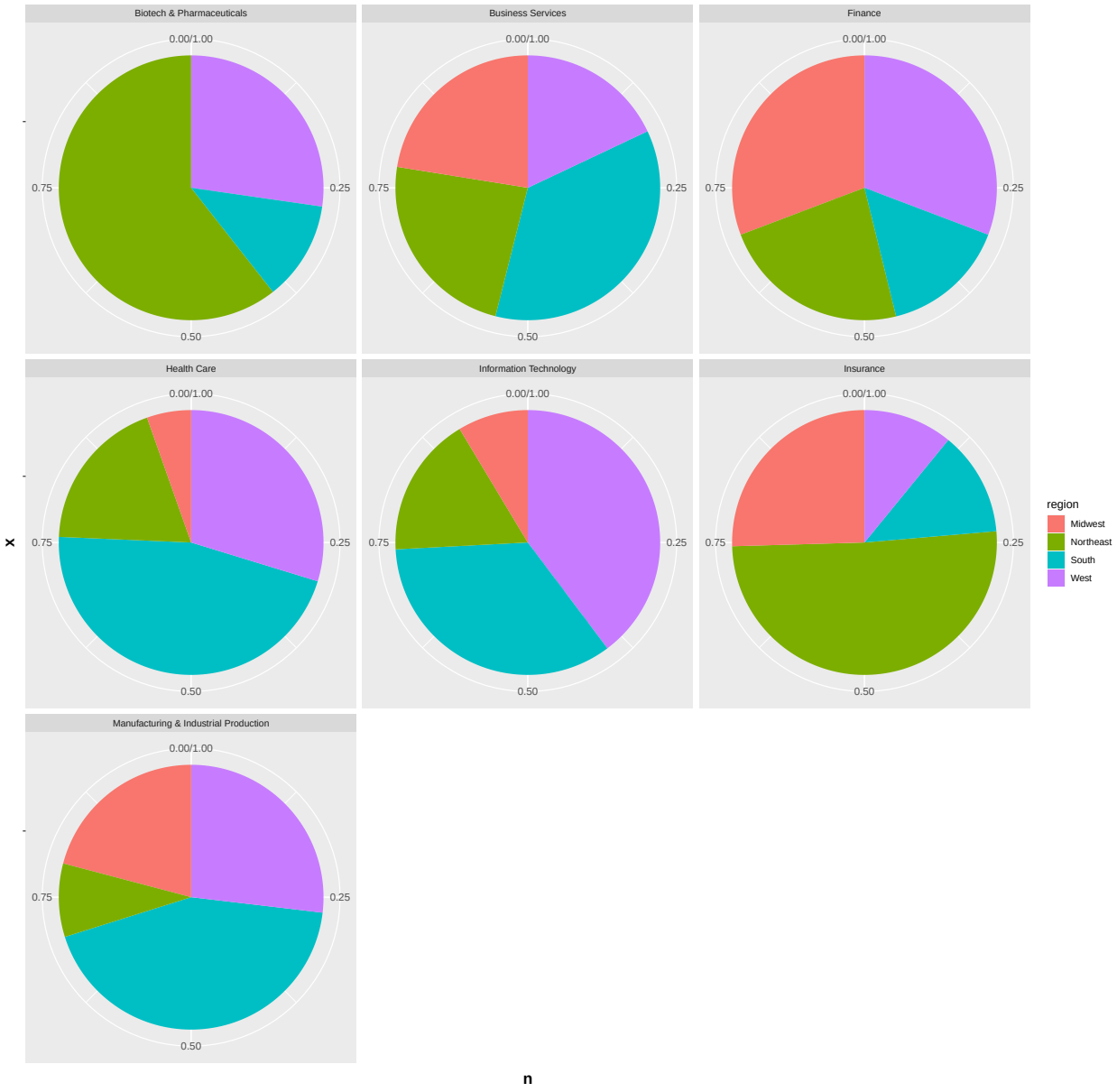
```
#so going to look at:
df2 <- df %>% select(Python,sql,sas,tableau,salary,aws,spark,Sector,region,Rating,job_title_sim,excel,ha
```

Check out sector, region, and job title for potential clustering and random effect inclusion:

```
knitr::opts_chunk$set(echo = TRUE, fig.width = 10, fig.height = 10)

regions_sector <- df %>% select(Sector,region) %>% group_by(Sector,region) %>% count() %>% arrange(-des

regions_sector %>% select(Sector,region,n) %>% ggplot(aes(fill=Sector, y=n, x="")) + geom_bar(stat = "i
axis.title=element_text(size=15,face="bold")) + theme(plot.title = element_text(size = 20, face = "bold
```



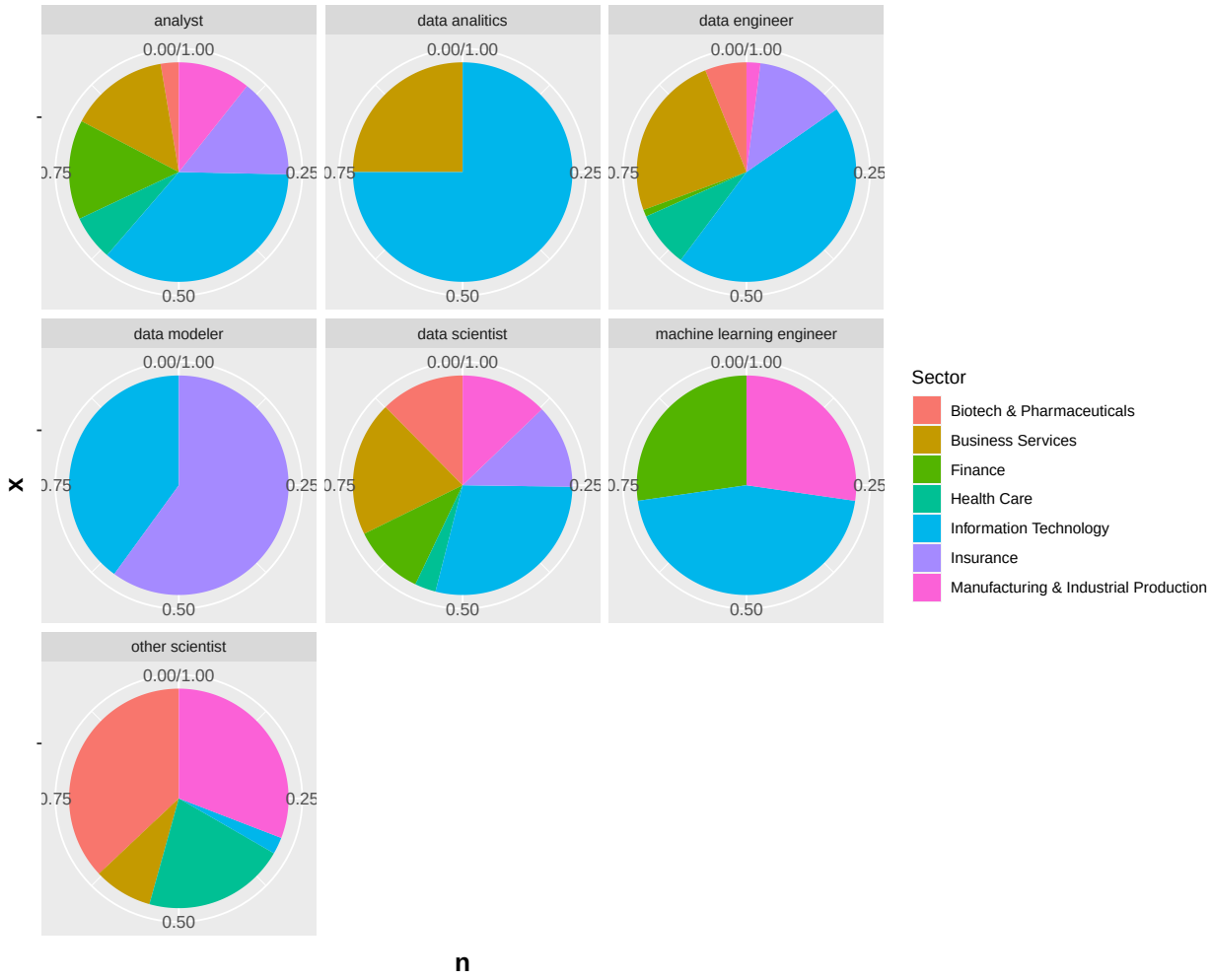

```
#or viewed the other way.
regions_sector %>% select(Sector,region,n) %>% ggplot(aes(fill=region, y=n, x="")) + geom_bar(stat = "id
axis.title=element_text(size=15,face="bold")) + theme(plot.title = element_text(size = 20, face = "bold"
```



Certainly some potential groupings there.

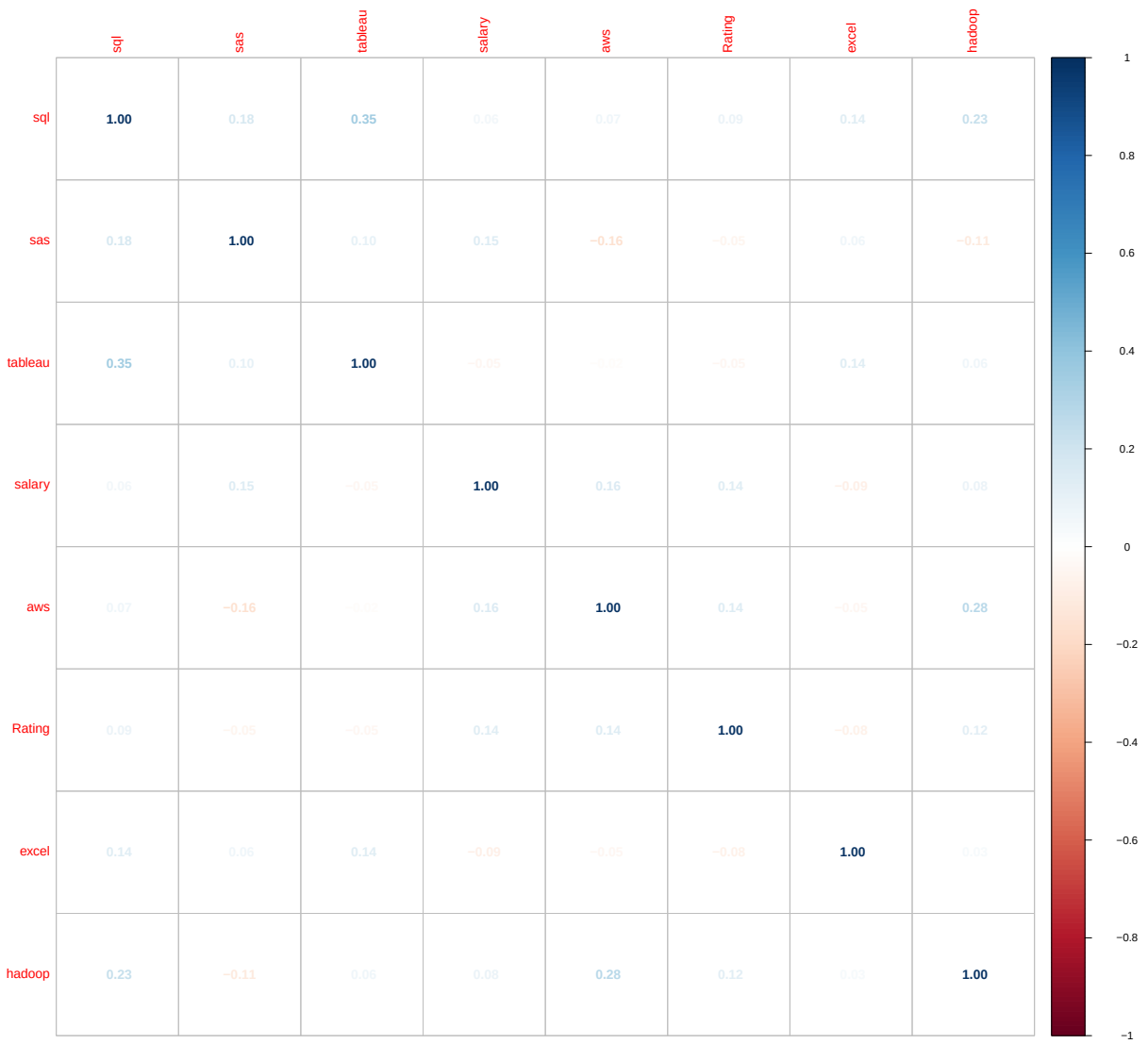
Looking at job title grouping counts and by sector.

```
knitr::opts_chunk$set(echo = TRUE, fig.width = 15, fig.height = 15)
#title
title_by_sector <- df2 %>% select(Sector,job_title_sim) %>% group_by(Sector,job_title_sim) %>% count()

title_by_sector %>% ggplot(aes(fill=Sector, y=n, x="")) + geom_bar(stat = "identity", width = 1, positi
axis.title=element_text(size=15,face="bold")) + theme(plot.title = element_text(size = 20, face = "bold
```



Also check for correlations between our potential predictors

```
knitr::opts_chunk$set(echo = TRUE, fig.width = 10, fig.height = 10)
cors <- cor(df2[,c(2:6,10,12:13)])
corrplot(cors,method = "number")
```




```
#small correlations all around.
```

So region, sector, and job title could all be seen potential as random effects. Going to test them out. Testing Models

Not a "predictive" problem as much as trying to understand the relationships between the variables, but going to create a hold out set to test the accuracy of the fitted model on.

```
set.seed(25)
train <- sample(nrow(df2),nrow(df2)*.9,replace = TRUE)
train_data <- df2[train,]
test <- df2[-train,]
test_labels <- test$Python
```

```r
knitr::opts_chunk$set(echo = TRUE, fig.width = 10, fig.height = 10)

#RANDOM EFFECTS OF SECTOR AND REGION.
fit1_hier <- glmer(Python ~ (1|Sector) + (1|region) + sql + sas + tableau + aws + spark + Rating + job_
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## unable to evaluate scaled gradient
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge: degenerate Hessian with 1 negative eigenvalues
```

```r
summary(fit1_hier)
```

```
## Warning in vcov.merMod(object, use.hessian = use.hessian): variance-covariance matrix computed from
## not positive definite or contains NA values: falling back to var-cov estimated from RX
```

```
## Warning in vcov.merMod(object, correlation = correlation, sigm = sig): variance-covariance matrix co
## not positive definite or contains NA values: falling back to var-cov estimated from RX
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | Sector) + (1 | region) + sql + sas + tableau +
##     aws + spark + Rating + job_title_sim + salary
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    418.0    483.9   -193.0    386.0      437
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -9.2914 -0.3910  0.1806  0.5416  3.3579
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  Sector (Intercept) 0.04056  0.2014
##  region (Intercept) 0.03732  0.1932
## Number of obs: 453, groups:  Sector, 7; region, 4
##
## Fixed effects:
##                                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)                         -5.427e+00  1.153e+00  -4.705 2.53e-06
## sql                                  1.098e+00  3.164e-01   3.472 0.000517
## sas                                 -1.140e-02  4.349e-01  -0.026 0.979093
## tableau                              6.810e-01  3.621e-01   1.881 0.060014
## aws                                  6.598e-01  3.473e-01   1.900 0.057438
## spark                                1.958e+00  3.866e-01   5.064 4.10e-07
## Rating                               8.343e-01  2.585e-01   3.228 0.001249
## job_title_simdata analitics          1.719e+00  1.146e+00   1.500 0.133707
## job_title_simdata engineer           1.607e-01  4.997e-01   0.322 0.747707
## job_title_simdata modeler           -9.212e-01  1.430e+00  -0.644 0.519570
```

```
## job_title_simdata scientist             2.024e+00  4.272e-01   4.737 2.17e-06
## job_title_simmachine learning engineer  1.631e+01  1.033e+03   0.016 0.987410
## job_title_simother scientist           -1.213e+00  7.496e-01  -1.618 0.105727
## salary                                  4.185e-03  4.161e-03   1.006 0.314599
##
## (Intercept)                            ***
## sql                                    ***
## sas
## tableau                                .
## aws                                    .
## spark                                  ***
## Rating                                 **
## job_title_simdata analitics
## job_title_simdata engineer
## job_title_simdata modeler
## job_title_simdata scientist            ***
## job_title_simmachine learning engineer
## job_title_simother scientist
## salary
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##
## Correlation matrix not shown by default, as p = 14 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## optimizer (Nelder_Mead) convergence code: 0 (OK)
## unable to evaluate scaled gradient
## Model failed to converge: degenerate  Hessian with 1 negative eigenvalues
```

```
#basically zero variance, seems inappropriate to use them as random effect.

#try with only one random effect term, sector
fit2_sector <- glmer(Python ~ (1|Sector) + sql + sas + tableau + aws + spark + Rating + job_title_sim +
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(fit2_sector)
```

```
## Warning in vcov.merMod(object, use.hessian = use.hessian): variance-covariance matrix computed from
## not positive definite or contains NA values: falling back to var-cov estimated from RX

## Warning in vcov.merMod(object, use.hessian = use.hessian): variance-covariance matrix computed from
## not positive definite or contains NA values: falling back to var-cov estimated from RX

## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | Sector) + sql + sas + tableau + aws + spark + Rating +
##     job_title_sim + region + salary
##    Data: train_data
```

```
##
##      AIC      BIC   logLik deviance df.resid
##    415.8    489.9   -189.9    379.8      435
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -10.8004  -0.3878   0.1852   0.5339   3.6316
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  Sector (Intercept) 0        0
## Number of obs: 453, groups:  Sector, 7
##
## Fixed effects:
##                                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)                             -5.945e+00  1.311e+00  -4.536 5.72e-06
## sql                                      1.171e+00  3.177e-01   3.685 0.000228
## sas                                      1.483e-01  4.372e-01   0.339 0.734519
## tableau                                  6.192e-01  3.621e-01   1.710 0.087298
## aws                                      6.110e-01  3.467e-01   1.762 0.078013
## spark                                    2.025e+00  3.943e-01   5.137 2.80e-07
## Rating                                   8.786e-01  2.592e-01   3.390 0.000699
## job_title_simdata analitics              1.839e+00  1.192e+00   1.543 0.122844
## job_title_simdata engineer               8.338e-02  5.075e-01   0.164 0.869513
## job_title_simdata modeler               -8.694e-01  1.432e+00  -0.607 0.543781
## job_title_simdata scientist              2.006e+00  4.261e-01   4.707 2.51e-06
## job_title_simmachine learning engineer   2.576e+01  1.076e+05   0.000 0.999809
## job_title_simother scientist            -1.349e+00  7.436e-01  -1.814 0.069752
## regionNortheast                          1.831e-01  4.380e-01   0.418 0.675853
## regionSouth                              5.141e-01  4.675e-01   1.100 0.271431
## regionWest                               9.669e-01  4.489e-01   2.154 0.031254
## salary                                   3.209e-03  4.323e-03   0.742 0.457838
##
## (Intercept)                             ***
## sql                                     ***
## sas
## tableau                                 .
## aws                                     .
## spark                                   ***
## Rating                                  ***
## job_title_simdata analitics
## job_title_simdata engineer
## job_title_simdata modeler
## job_title_simdata scientist             ***
## job_title_simmachine learning engineer
## job_title_simother scientist            .
## regionNortheast
## regionSouth
## regionWest                              *
## salary
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##
```

```
## Correlation matrix not shown by default, as p = 17 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## optimizer (Nelder_Mead) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
#basically still zero.

#with region as the one random effect
fit3_region <- glmer(Python ~ (1|region) + sql + sas + tableau + aws + spark + Rating + job_title_sim +
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(fit3_region)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | region) + sql + sas + tableau + aws + spark + Rating +
##     job_title_sim + Sector + salary
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    417.5    503.9   -187.8    375.5      432
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -9.3229 -0.3715  0.1812  0.5472  3.4923
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  region (Intercept) 0        0
## Number of obs: 453, groups:  region, 4
##
## Fixed effects:
##                                            Estimate Std. Error z value
## (Intercept)                               -5.724015   1.244883  -4.598
## sql                                        1.016372   0.344262   2.952
## sas                                       -0.172581   0.458073  -0.377
## tableau                                    0.736118   0.371650   1.981
## aws                                        0.769538   0.356443   2.159
## spark                                      1.876030   0.389631   4.815
## Rating                                     0.717346   0.267061   2.686
## job_title_simdata analitics                1.650259   1.105926   1.492
## job_title_simdata engineer                 0.242133   0.513493   0.472
## job_title_simdata modeler                 -0.878952   1.421104  -0.618
## job_title_simdata scientist                2.096786   0.450523   4.654
## job_title_simmachine learning engineer    24.118889 512.000225   0.047
## job_title_simother scientist              -0.910993   0.768032  -1.186
## SectorBusiness Services                    0.482448   0.527779   0.914
## SectorFinance                              1.439729   0.588964   2.445
```

```
## SectorHealth Care                               -0.143864   0.720268  -0.200
## SectorInformation Technology                     0.961038   0.473289   2.031
## SectorInsurance                                  0.476916   0.560474   0.851
## SectorManufacturing & Industrial Production      0.872748   0.550057   1.587
## salary                                           0.004686   0.004210   1.113
##                                                 Pr(>|z|)
## (Intercept)                                     4.26e-06 ***
## sql                                              0.00315 **
## sas                                              0.70636
## tableau                                          0.04763 *
## aws                                              0.03086 *
## spark                                           1.47e-06 ***
## Rating                                           0.00723 **
## job_title_simdata analitics                      0.13565
## job_title_simdata engineer                       0.63725
## job_title_simdata modeler                        0.53625
## job_title_simdata scientist                     3.25e-06 ***
## job_title_simmachine learning engineer           0.96243
## job_title_simother scientist                     0.23557
## SectorBusiness Services                          0.36066
## SectorFinance                                    0.01450 *
## SectorHealth Care                                0.84169
## SectorInformation Technology                     0.04230 *
## SectorInsurance                                  0.39482
## SectorManufacturing & Industrial Production      0.11259
## salary                                           0.26566
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##
## Correlation matrix not shown by default, as p = 20 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## optimizer (Nelder_Mead) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

```
#again virtually zero.

#random effect on job title?
fit4_title <- glmer(Python ~ (1|job_title_sim) + sql + sas + tableau + aws + spark + Rating + Sector + r
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.00421816 (tol = 0.002, component 1)
```

```
summary(fit4_title)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | job_title_sim) + sql + sas + tableau + aws + spark +
##     Rating + Sector + region + salary
```

```
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    428.6    502.7   -196.3    392.6      435
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -10.7947  -0.3895   0.1966   0.5191   2.6181
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  job_title_sim (Intercept) 1.407    1.186
## Number of obs: 453, groups:  job_title_sim, 7
##
## Fixed effects:
##                                             Estimate Std. Error z value
## (Intercept)                                -5.501691   1.439873  -3.821
## sql                                         1.084904   0.344743   3.147
## sas                                        -0.007587   0.459567  -0.017
## tableau                                     0.685553   0.369442   1.856
## aws                                         0.711104   0.348694   2.039
## spark                                       1.898292   0.401659   4.726
## Rating                                      0.741921   0.267929   2.769
## SectorBusiness Services                     0.496209   0.556121   0.892
## SectorFinance                               1.431725   0.598530   2.392
## SectorHealth Care                          -0.277029   0.748664  -0.370
## SectorInformation Technology                0.861719   0.515194   1.673
## SectorInsurance                             0.516257   0.558678   0.924
## SectorManufacturing & Industrial Production  0.854261   0.593388   1.440
## regionNortheast                             0.337468   0.464264   0.727
## regionSouth                                 0.446154   0.471524   0.946
## regionWest                                  0.878793   0.459366   1.913
## salary                                      0.004217   0.004368   0.965
##                                            Pr(>|z|)
## (Intercept)                                0.000133 ***
## sql                                        0.001650 **
## sas                                        0.986829
## tableau                                    0.063505 .
## aws                                        0.041416 *
## spark                                      2.29e-06 ***
## Rating                                     0.005621 **
## SectorBusiness Services                    0.372249
## SectorFinance                              0.016754 *
## SectorHealth Care                          0.711360
## SectorInformation Technology               0.094404 .
## SectorInsurance                            0.355451
## SectorManufacturing & Industrial Production 0.149971
## regionNortheast                            0.467294
## regionSouth                                0.344049
## regionWest                                 0.055741 .
## salary                                     0.334308
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Correlation matrix not shown by default, as p = 17 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## optimizer (Nelder_Mead) convergence code: 0 (OK)
## Model failed to converge with max|grad| = 0.00421816 (tol = 0.002, component 1)
```

```
#okay we see an actual effect. so let's pull some non-significant variables. dropping sector.


fit_title3 <- glmer(Python ~ (1|job_title_sim) + sql + sas + tableau + aws + spark  + Rating + region +
summary(fit_title3)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | job_title_sim) + sql + sas + tableau + aws + spark +
##     Rating + region + salary
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    426.9    476.3   -201.5    402.9      441
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -10.5750  -0.4386   0.1946   0.5471   2.4645
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  job_title_sim (Intercept) 1.863    1.365
## Number of obs: 453, groups:  job_title_sim, 7
##
## Fixed effects:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -5.265251   1.327859  -3.965 7.33e-05 ***
## sql              1.195659   0.312441   3.827 0.000130 ***
## sas              0.165262   0.434720   0.380 0.703829
## tableau          0.631867   0.359682   1.757 0.078963 .
## aws              0.600395   0.339023   1.771 0.076568 .
## spark            2.010432   0.388310   5.177 2.25e-07 ***
## Rating           0.872108   0.255708   3.411 0.000648 ***
## regionNortheast  0.155567   0.430882   0.361 0.718068
## regionSouth      0.454532   0.457712   0.993 0.320684
## regionWest       0.921052   0.442714   2.080 0.037483 *
## salary           0.003654   0.004230   0.864 0.387605
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) sql    sas    tablea aws    spark  Rating rgnNrt rgnSth
## sql         -0.260
## sas         -0.104 -0.264
```

```
## tableau     -0.168 -0.163  0.031
## aws          0.058 -0.055  0.081 -0.008
## spark       -0.049 -0.001  0.088  0.106  0.044
## Rating      -0.779  0.099  0.122  0.154 -0.083  0.014
## reginNrthst -0.401  0.109  0.118  0.006 -0.119 -0.100  0.187
## regionSouth -0.407  0.223  0.150 -0.020 -0.087  0.024  0.166  0.700
## regionWest  -0.301  0.152  0.189 -0.053 -0.075  0.018  0.114  0.683  0.687
## salary      -0.325  0.121 -0.126  0.090 -0.099 -0.040 -0.006  0.123  0.128
##             rgnWst
## sql
## sas
## tableau
## aws
## spark
## Rating
## reginNrthst
## regionSouth
## regionWest
## salary      -0.042
```

```
#drop sas p value of .9.
fit_title4 <- glmer(Python ~ (1|job_title_sim) + sql + tableau + spark + aws + Rating + region + salary
summary(fit_title4)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | job_title_sim) + sql + tableau + spark + aws +
##     Rating + region + salary
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    425.1    470.3   -201.5    403.1      442
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -10.6856  -0.4359   0.1945   0.5460   2.4066
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  job_title_sim (Intercept) 1.876    1.37
## Number of obs: 453, groups:  job_title_sim, 7
##
## Fixed effects:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -5.214029   1.320728  -3.948 7.89e-05 ***
## sql              1.228109   0.301107   4.079 4.53e-05 ***
## tableau          0.627732   0.359507   1.746 0.080795 .
## spark            1.997824   0.386925   5.163 2.43e-07 ***
## aws              0.589890   0.338849   1.741 0.081708 .
## Rating           0.860472   0.253750   3.391 0.000696 ***
## regionNortheast  0.136082   0.427436   0.318 0.750206
## regionSouth      0.428479   0.452079   0.948 0.343233
## regionWest       0.889207   0.434525   2.046 0.040718 *
```

```
## salary           0.003857   0.004190   0.921 0.357230
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) sql    tablea spark  aws    Rating rgnNrt rgnSth rgnWst
## sql         -0.299
## tableau     -0.167 -0.158
## spark       -0.039  0.026  0.102
## aws          0.067 -0.032 -0.011  0.036
## Rating      -0.776  0.137  0.152  0.001 -0.094
## reginNrthst -0.394  0.146  0.001 -0.112 -0.132  0.176
## regionSouth -0.397  0.276 -0.028  0.008 -0.103  0.150  0.695
## regionWest  -0.288  0.215 -0.063 -0.001 -0.095  0.093  0.677  0.678
## salary      -0.343  0.091  0.096 -0.028 -0.089  0.011  0.140  0.151 -0.017
```

```
#only one region is significant, and only at the 0.1 level. But, maybe trying modeling this as a random
fit5 <- glmer(Python ~ (1|job_title_sim) + (1|region) + Rating + aws + salary + sql + tableau + spark ,
summary(fit5)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | job_title_sim) + (1 | region) + Rating + aws +
##     salary + sql + tableau + spark
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    426.5    463.5   -204.2    408.5      444
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -9.2998 -0.4281  0.2106  0.5594  2.2287
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  job_title_sim (Intercept) 1.79908  1.3413
##  region        (Intercept) 0.07091  0.2663
## Number of obs: 453, groups:  job_title_sim, 7; region, 4
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.884217   1.207323  -4.045 5.22e-05 ***
## Rating       0.863636   0.250975   3.441 0.000579 ***
## aws          0.613277   0.337102   1.819 0.068871 .
## salary       0.004442   0.004105   1.082 0.279244
## sql          1.161700   0.295532   3.931 8.46e-05 ***
## tableau      0.671613   0.359474   1.868 0.061717 .
## spark        1.978484   0.378210   5.231 1.68e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##         (Intr) Rating aws    salary sql    tablea
```

```
## Rating  -0.790
## aws      0.019 -0.077
## salary  -0.320 -0.012 -0.067
## sql     -0.226  0.108 -0.014  0.049
## tableau -0.196  0.161 -0.004  0.099 -0.166
## spark   -0.068  0.016  0.029 -0.020  0.030  0.102
```

```
#very small variance. Probably not worth it. remove.

fit6 <- glmer(Python ~ (1|job_title_sim) + Rating + salary + sql + tableau + spark + aws , family = bin
summary(fit6)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | job_title_sim) + Rating + salary + sql + tableau +
##     spark + aws
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    425.5    458.5   -204.8    409.5      445
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -8.0620 -0.4284  0.1990  0.5639  2.1618
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  job_title_sim (Intercept) 1.686    1.299
## Number of obs: 453, groups:  job_title_sim, 7
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.013588   1.170703  -4.283 1.85e-05 ***
## Rating       0.880922   0.248996   3.538 0.000403 ***
## salary       0.005330   0.003955   1.347 0.177829
## sql          1.108459   0.285712   3.880 0.000105 ***
## tableau      0.715918   0.356874   2.006 0.044848 *
## spark        1.948115   0.368702   5.284 1.27e-07 ***
## aws          0.622437   0.336191   1.851 0.064107 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##         (Intr) Rating salary sql    tablea spark
## Rating  -0.797
## salary  -0.296 -0.040
## sql     -0.228  0.104  0.061
## tableau -0.199  0.166  0.078 -0.148
## spark   -0.091  0.035  0.003  0.014  0.114
## aws      0.006 -0.068 -0.060  0.002 -0.003  0.025
```

```r
#remove aws

fit6 <- glmer(Python ~ (1|job_title_sim) + Rating + salary + sql + tableau + spark, family = binomial, c
summary(fit6)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | job_title_sim) + Rating + salary + sql + tableau +
##     spark
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    427.1    455.9   -206.5    413.1      446
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -6.4424 -0.4057  0.2256  0.5515  2.0905
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  job_title_sim (Intercept) 1.596    1.263
## Number of obs: 453, groups:  job_title_sim, 7
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.080758   1.160935  -4.376 1.21e-05 ***
## Rating       0.922881   0.248184   3.719   0.0002 ***
## salary       0.005862   0.003919   1.496   0.1348
## sql          1.115337   0.285860   3.902 9.55e-05 ***
## tableau      0.729360   0.354386   2.058   0.0396 *
## spark        1.959889   0.365207   5.367 8.03e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##         (Intr) Rating salary sql    tablea
## Rating  -0.803
## salary  -0.295 -0.048
## sql     -0.234  0.110  0.063
## tableau -0.194  0.162  0.074 -0.150
## spark   -0.096  0.043  0.000  0.017  0.107
```

```r
#FITTING THE GLM MODEL VIA BACKWARDS ELIMINATION/PURPOSEFUL SELECTION. Only fixed effects. Did not show
fit_all <- glm(Python ~ ., family = binomial, data = train_data)

fit_all_2 <- glm(Python ~ .-Sector, family = binomial, data = train_data)

fit_all_3 <- glm(Python ~ .-Sector -sas, family = binomial, data = train_data)

fit_all_4 <- glm(Python ~ .-Sector -sas -excel, family = binomial, data = train_data)

fit_all_5 <- glm(Python ~ .-Sector -sas -excel - hadoop, family = binomial, data = train_data)
```

```r
fit_all_6 <- glm(Python ~ .-Sector -sas -excel - hadoop, family = binomial, data = train_data)

fit_all_7 <- glm(Python ~ .-Sector -sas -excel - hadoop - aws, family = binomial, data = train_data)

fit_all_8 <- glm(Python ~ .-Sector -sas -excel - hadoop - aws - region, family = binomial, data = train
summary(fit_all_8)
```

```
##
## Call:
## glm(formula = Python ~ . - Sector - sas - excel - hadoop - aws -
##     region, family = binomial, data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7380  -0.5534   0.3009   0.7309   2.1292
##
## Coefficients:
##                                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)                           -5.772552   1.116155  -5.172 2.32e-07
## sql                                    1.073591   0.290090   3.701 0.000215
## tableau                                0.729372   0.357291   2.041 0.041212
## salary                                 0.005287   0.003982   1.328 0.184215
## spark                                  1.958460   0.369814   5.296 1.18e-07
## Rating                                 0.926826   0.251211   3.689 0.000225
## job_title_simdata analitics            1.692092   1.139341   1.485 0.137504
## job_title_simdata engineer             0.431544   0.471093   0.916 0.359642
## job_title_simdata modeler             -0.789239   1.334134  -0.592 0.554136
## job_title_simdata scientist            2.016163   0.412187   4.891 1.00e-06
## job_title_simmachine learning engineer 16.784747 722.657381   0.023 0.981470
## job_title_simother scientist           -1.091924   0.726020  -1.504 0.132585
##
## (Intercept)                           ***
## sql                                   ***
## tableau                               *
## salary
## spark                                 ***
## Rating                                ***
## job_title_simdata analitics
## job_title_simdata engineer
## job_title_simdata modeler
## job_title_simdata scientist           ***
## job_title_simmachine learning engineer
## job_title_simother scientist
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 611.18  on 452  degrees of freedom
## Residual deviance: 390.41  on 441  degrees of freedom
## AIC: 414.41
##
## Number of Fisher Scoring iterations: 15
```

```r
#we get to the same model, but with some of the levels of job title being significant.

#if removed to keep only significant
fit_all_9 <- glm(Python ~ .-Sector -sas -excel - hadoop - aws - region - job_title_sim, family = binomia
summary(fit_all_9)
```

```
##
## Call:
## glm(formula = Python ~ . - Sector - sas - excel - hadoop - aws -
##     region - job_title_sim, family = binomial, data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6144  -0.8235   0.3950   0.8488   1.8590
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.997646   0.919211  -6.525 6.81e-11 ***
## sql          1.083274   0.242194   4.473 7.72e-06 ***
## tableau      0.716351   0.311199   2.302   0.0213 *
## salary       0.015407   0.003465   4.446 8.75e-06 ***
## spark        1.803985   0.302489   5.964 2.46e-09 ***
## Rating       1.003251   0.225820   4.443 8.88e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 611.18  on 452  degrees of freedom
## Residual deviance: 466.68  on 447  degrees of freedom
## AIC: 478.68
##
## Number of Fisher Scoring iterations: 5
```

```r
#compare this model to the model with sector job title as a random effect.
summary(fit6)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | job_title_sim) + Rating + salary + sql + tableau +
##     spark
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    427.1    455.9   -206.5    413.1      446
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -6.4424 -0.4057  0.2256  0.5515  2.0905
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
```

```
##  job_title_sim (Intercept) 1.596    1.263
## Number of obs: 453, groups:  job_title_sim, 7
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.080758   1.160935  -4.376 1.21e-05 ***
## Rating       0.922881   0.248184   3.719   0.0002 ***
## salary       0.005862   0.003919   1.496   0.1348
## sql          1.115337   0.285860   3.902 9.55e-05 ***
## tableau      0.729360   0.354386   2.058   0.0396 *
## spark        1.959889   0.365207   5.367 8.03e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##         (Intr) Rating salary sql    tablea
## Rating  -0.803
## salary  -0.295 -0.048
## sql     -0.234  0.110  0.063
## tableau -0.194  0.162  0.074 -0.150
## spark   -0.096  0.043  0.000  0.017  0.107
```

Very similar models. AIC lower for the glmm indicates a better fit. Check the confusion matrix and ROC curves for performance analysis.

```
#very similar models.
#AIC lower for the glmm indicates a better fit. Check the confusion matrix and ROC curves.
prop <- sum(test$Python/nrow(test))

predicted <- predict(fit_all_9,newdata = test)
pred_probs <- expit(predicted)
classes <- ifelse(pred_probs>prop,1,0)
xtabs(~ test_labels + classes) %>% kable()
```

	0	1
0	73	37
1	14	90

```
as.data.frame(xtabs(~ test_labels + classes)) %>% kable()
```

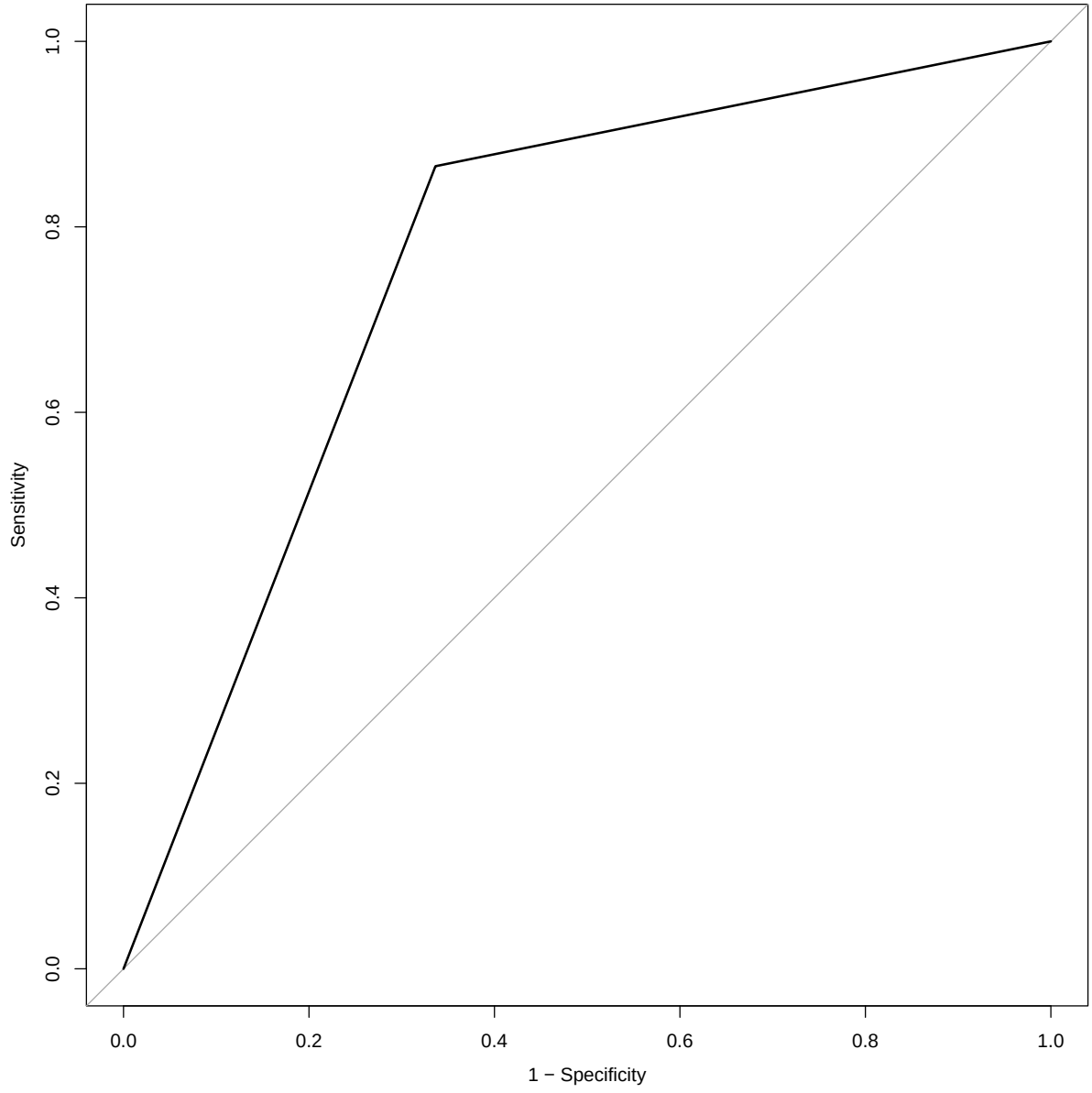
test_labels	classes	Freq
0	0	73
1	0	14
0	1	37
1	1	90

```
#Visualize with ROC CURVE:
rocplot_glm <- roc(test$Python ~ classes, data = test)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot.roc(rocplot_glm, legacy.axes=TRUE)
```




```
auc(rocplot_glm) #the accuracy, 76%.
```

```
## Area under the curve: 0.7645
```

```
#What about for the glmm?
predicted <- predict(fit6, newdata=test)
pred_probs <- expit(predicted)
classes <- ifelse(pred_probs>prop,1,0)
xtabs(~ test_labels + classes) %>% kable()
```

	0	1
0	75	35
1	16	88

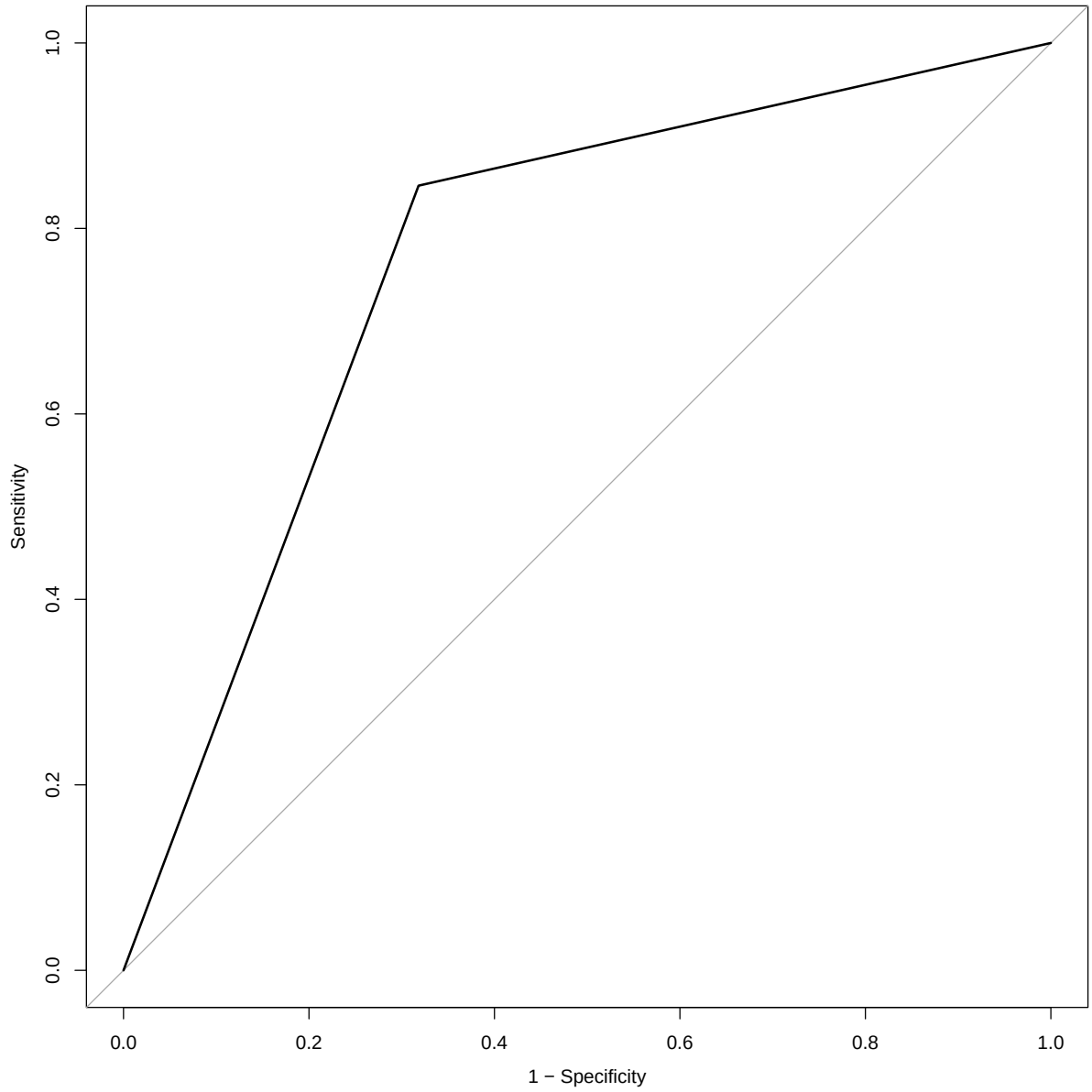
```
as.data.frame(xtabs(~ test_labels + classes)) %>% kable()
```

test_labels	classes	Freq
0	0	75
1	0	16
0	1	35
1	1	88

```
#Visualize with ROC CURVE:
rocplot_glmm <- roc(test$Python ~ classes, data = test)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
plot.roc(rocplot_glmm, legacy.axes=TRUE)
```




```
auc(rocplot_glmm) #the accuracy, 76%.
```

```
## Area under the curve: 0.764
```

How about a likelihood ratio test for the variance of the random intercept term?

deviance glmm 479.7 497 df. deviance of glm 533.64 df.

The model terms are the same besides for the random variance term. 533.64-479.7 53.94 on one degree of freedom.

p-value is half the right tail probability above 53.94 on a chi-square distribution with one df.

Our likelihood ratio test test is Ho: variance = 0 vs. Ha: variance > 0. Our p-value is half the right tail probability above the difference in residual deviances for a chi-squared distribution with 1 df. (As in section 10.1.6 and 10.2.2 in the Agresti book)

```
pchisq(53.94,1,lower.tail = FALSE)/2
```

```
## [1] 1.033532e-13
```

```
#supports that the model with the random intercept term is better.
```

Using the mixed effect model based on AIC, sensitivity and specificity scores and AUC, and lrt, and it seems reasonable for job title to be a random effect so, using the glmm model as the final model.

```
summary(fit6)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: Python ~ (1 | job_title_sim) + Rating + salary + sql + tableau +
##     spark
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##    427.1    455.9   -206.5    413.1      446
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -6.4424 -0.4057  0.2256  0.5515  2.0905
##
## Random effects:
##  Groups        Name        Variance Std.Dev.
##  job_title_sim (Intercept) 1.596    1.263
## Number of obs: 453, groups:  job_title_sim, 7
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.080758   1.160935  -4.376 1.21e-05 ***
## Rating       0.922881   0.248184   3.719   0.0002 ***
## salary       0.005862   0.003919   1.496   0.1348
## sql          1.115337   0.285860   3.902 9.55e-05 ***
## tableau      0.729360   0.354386   2.058   0.0396 *
## spark        1.959889   0.365207   5.367 8.03e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##         (Intr) Rating salary sql    tablea
## Rating  -0.803
## salary  -0.295 -0.048
## sql     -0.234  0.110  0.063
## tableau -0.194  0.162  0.074 -0.150
## spark   -0.096  0.043  0.000  0.017  0.107
```

Out of curiosity and for confirmation running bestglm.

```r
#out of curiosity and for confirmation check bestglm.
bestglm_data <- as.data.frame(cbind(as.factor(df2$sql),as.factor(df2$sas),as.factor(df2$tableau),df2$sal
a <- bestglm(bestglm_data, family = binomial)
```

```
## Morgan-Tatar search since family is non-gaussian.
```

```r
#variable order: sql, tableau, salary, spark, rating. Best glm gives the same five significant fixed ef
a
```

```
## BIC
## BICq equivalent for q in (0.298546733677389, 0.784664709374456)
## Best Model:
##                Estimate  Std. Error   z value     Pr(>|z|)
## (Intercept) -9.01775217 1.029330042 -8.760798 1.938736e-18
## V1           1.16108253 0.227515810  5.103305 3.337727e-07
## V3           0.80226615 0.290416874  2.762464 5.736689e-03
## V4           0.01681808 0.003051897  5.510698 3.574144e-08
## V6           1.26181239 0.275338224  4.582772 4.588516e-06
## V9           0.87144585 0.206956249  4.210773 2.544980e-05
```