Project 3 - Data Scientist Skills

DATA607 - Acquisition of Data and Management - Instructor: Andrew Catlin

Date: 3/23/2021

Load needed libraries

Function call to MySQL db to connect and use the return command in a function.

```
# MySQL Connect Function
conn.MySQL <- function(db_parms)
{
  db_conn <- dbConnect(MySQL(), user=db_user, password=db_password, dbname=db_name, host=db_host)
  return(db_conn)
}
```

Source the credentials parameter file from your local directory. Do not store in github repository since R has no encryption capability.

```
# Read the user MySQl Parameter credentials file
filename <- "/Users/Audiorunner13/CUNY MSDS Course Work/DATA607 Spring 2021/Week7/MySQL_parms.csv"
db_parms <- read.csv(filename)
```

Set the login application credential to pass to the db log in function

```
# Set  up DB parms
db_user = db_parms$db_user
db_password = db_parms$db_password
db_name = db_parms$db_name
db_host = db_parms$db_host
db_result_set = ""

db_parms <- c(db_user, db_password, db_name, db_host, db_result_set)
```

Call the conn.MySql connect function to access the movies database

```
db_conn <- conn.MySQL(db_parms)
```

Use the dbListTables() function to list the tables in the database

```
(dbListTables(db_conn))
```

```
## [1] "job_opening_tbl" "location_dim"
```

Use the dbListFields() function to list the fields in a database table

```
(dbListFields(db_conn, "location_dim"))
```

```
## [1] "uniq_id"  "location" "city"     "country"  "state"    "zip_code"
```

Load CSV Source the job locations dimension file from the github repository to load to the locations dimension

```
filename <- getURL("https://raw.githubusercontent.com/gcampos100/DATA607_CUNY_2021_Project3/main/Peter%
location_dim_df <- read.csv(text=filename)
```

Write to database tables from their respective data frames.

```
dbSendQuery(db_conn, "SET GLOBAL local_infile = true;")
```

```
## <MySQLResult:0,0,2>
```

```
dbWriteTable(db_conn, name = "location_dim", value = location_dim_df, append = TRUE,row.names = FALSE)
```

```
## [1] TRUE
```

Source the job openings csv from the github repository to load to the job openings table.

```
filename <- "/Users/Audiorunner13/CUNY MSDS Course Work/DATA607 Spring 2021/Week7/Data/job_openings.csv
job_opening_df <- read.csv(filename)
```

```
# append initial records to the job opening table
dbSendQuery(db_conn, "SET GLOBAL local_infile = true;")
```

```
## <MySQLResult:12,0,5>
```

```
dbWriteTable(db_conn, name = "job_opening_tbl", value = job_opening_df, append = TRUE, row.names = FALS
```

```
## [1] TRUE
```

Manipulating Imported CSV Dataset

```
# additional un tidy data set
filename <- "/Users/Audiorunner13/CUNY MSDS Course Work/DATA607 Spring 2021/Week7/Data/alldata.csv"
DS_job_df <- read.csv(filename)
```

```
# add unique ID column based on row name of each entry
DS_job_df<-cbind( data.frame("uniq_id" = as.integer(rownames(DS_job_df))), DS_job_df)
# add 40K to each uniq_id to ensure none match the first 30K entries of the other sets
DS_job_df<-DS_job_df %>%
  mutate_at( vars("uniq_id") ,
             funs(.+40000))
```

```
# splitting up the location column to create state, city, zip_code
temp <- as.data.frame(str_split_fixed(str_trim(DS_job_df$location),", ",2))
```

```
temp$V2 <- str_replace_all(temp$V2, "\\s", "=")
```

```
DS_job_df <- (temp<-cbind(DS_job_df,"city" = temp$V1,
             as.data.frame(str_split_fixed(temp$V2,"=",2)))%>%
                                              dplyr::rename("state"=V1, "zip_code"=V2))
```

Loading country column with a constant “United States” value

```
# set all country rows to United States
DS_job_df$country<-"United States"
```

Quick rearrange

```
# rearrange data.frame fields
DS_job_df<-DS_job_df %>%
  select(uniq_id,position,description,location,city,state,country,zip_code,company,reviews)
```

```
# create a job openings df
job_opening_df <- DS_job_df %>%
  select(1,2,3) %>%
    dplyr::rename( "uniq_id" = ‘uniq_id‘,
                   "job_title" = ‘position‘,
                   "job_descr" = ‘description‘)
```

```
# create a locations df
location_dim_df <-DS_job_df %>%
                         select(1,4,5,6,7,8)
```

Write to Database Using dbWriteTable function, we are able to write these imported dataframes into the appropriate Database tables on our local Database.

```
# Append new records to location dimension
dbSendQuery(db_conn, "SET GLOBAL local_infile = true;")
```

```
## <MySQLResult:111,0,8>
```

```
dbWriteTable(db_conn, name = "location_dim", value = location_dim_df, append = TRUE,row.names = FALSE)
```

```
## [1] TRUE
```

```
# append new records to the job opening table
dbSendQuery(db_conn, "SET GLOBAL local_infile = true;")
```

```
## <MySQLResult:100,0,11>
```

```r
dbWriteTable(db_conn, name = "job_opening_tbl", value = job_opening_df, append = TRUE, row.names = FALS
```

```
## [1] TRUE
```

Source the sql file in the github repositpry. The sql will extract all job openings data that containt Data Scientist in the job title.

```r
# source sql file and substitute each new line "\n" with a space
# filename <- getURL("https://raw.githubusercontent.com/gcampos100/DATA607_CUNY_2021_Project3/main/Pete
filename <- "/Users/Audiorunner13/CUNY MSDS Course Work/DATA607 Spring 2021/Week7/Project3/Sql/jobs_loca
db_sql <- read_file(filename)
db_sql <- gsub("\n", " ",db_sql)
db_sql
```

```
## [1] "SELECT j.uniq_id, j.job_title, j.job_descr, l.location FROM jobs.job_opening_tbl j, jobs.locati
```

Execute the sql query joining the fact table to the dimension tables and return all records in the result set. Specify the number of records to return by adjusting the "n =" argument.

```r
# execute sql query and return result set
db_data = dbSendQuery(db_conn, db_sql)
result_set = fetch(db_data, n = -1)
```

The result_set containing the extracted data is a data.frame.

```r
# display the class of the result set
class(result_set)
```

```
## [1] "data.frame"
```

Analysis

```r
# extract all job descriptions
job_descr_phrases <- result_set$job_descr
```

```r
# use the ngram to extract single word, double word phrases and calc the frequency
createNgram <- function(stringVector,ngramSize){

  ngram <- data.table()
  ng <- textcnt(stringVector,method = "string", n = ngramSize, tolower=FALSE)

  if(ngramSize==1){
#  ngram <- data.table(w1 = names(ng), freq = unclass(ng), length=nchar(names(ng)))
    ngram <- data.table(w1 = names(ng), freq = unclass(ng))
  }
  else {
#    ngram <- data.table(w1w2 = names(ng), freq = unclass(ng), length=nchar(names(ng)))
    ngram <- data.table(w1w2 = names(ng), freq = unclass(ng))
  }
  return(ngram)
}
```

```
# call the ngram function to extract all single word phrases from the job descriptions
job_descr_text <- job_descr_phrases
job_phrases_1 <- createNgram(job_descr_text,1)
job_phrases_1$w1 <- tolower(job_phrases_1$w1)  # convert to lowercase for accurate counting
names(job_phrases_1)
```

```
## [1] "w1"   "freq"
```

```
job_phrases_1 <- job_phrases_1 %>% arrange(desc(freq))
```

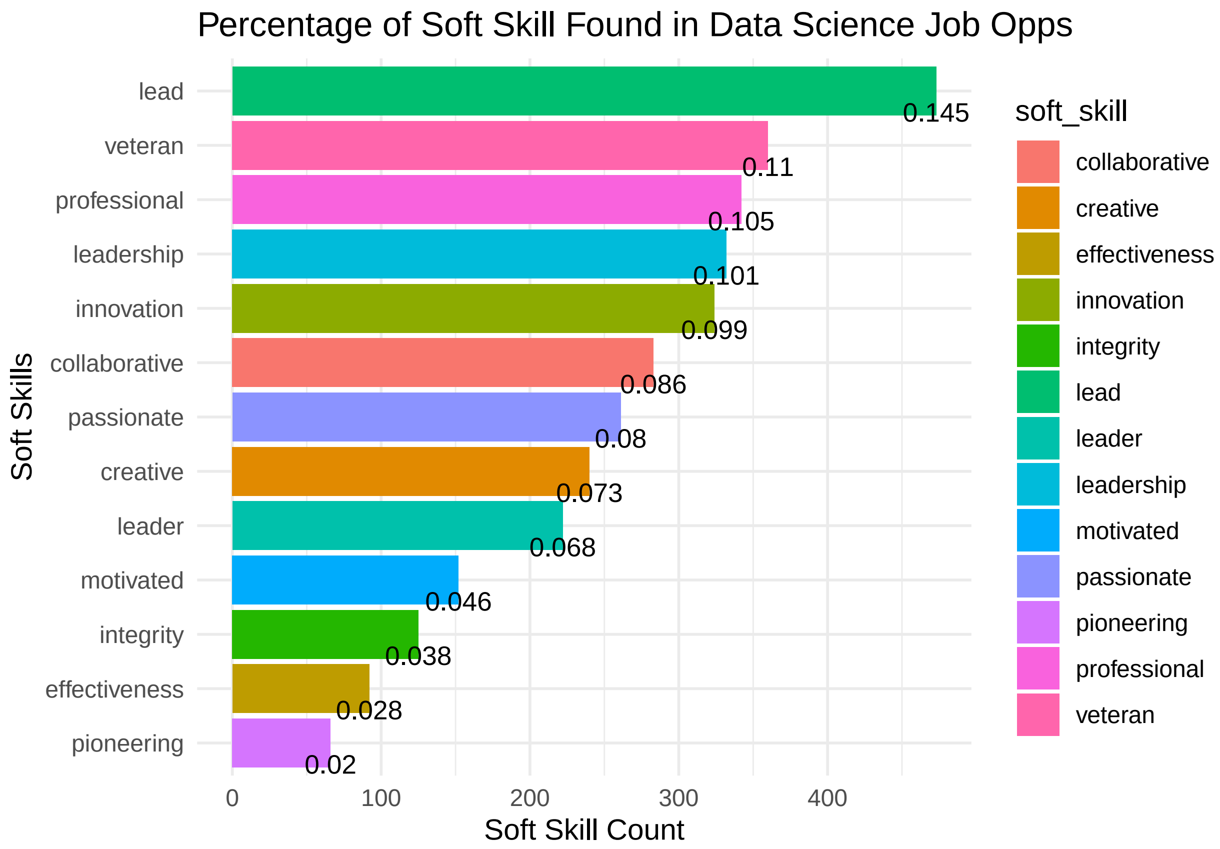
```
# create a soft skills data.frame and agg for accurate counts
lookup_soft_skills <- c("professional", "veteran", "lead", "leadership", "leader", "innovation", "collab
soft_skills <- filter(job_phrases_1, w1 %in% lookup_soft_skills)
soft_skills_tot <- aggregate.data.frame(x = soft_skills$freq,             # Sum by group
          by = list(soft_skills$w1),
          FUN = sum)
```

```
# rename soft skill df columns
soft_skills_tot <- soft_skills_tot %>%
   dplyr::rename("soft_skill" = Group.1, "count" = x)
```

```
# calc a prercentage field and append to repsective rows
tot_count <- as.integer(sum(soft_skills_tot$count))
skill_count_pct <- round(soft_skills_tot$count/tot_count, 3)
(soft_skills_tot <- cbind(soft_skills_tot, skill_count_pct))
```

```
##       soft_skill count skill_count_pct
## 1  collaborative   283           0.086
## 2       creative   240           0.073
## 3  effectiveness    92           0.028
## 4     innovation   324           0.099
## 5      integrity   125           0.038
## 6           lead   473           0.145
## 7         leader   222           0.068
## 8     leadership   332           0.101
## 9      motivated   152           0.046
## 10    passionate   261           0.080
## 11    pioneering    66           0.020
## 12  professional   342           0.105
## 13       veteran   360           0.110
```

```
# plot the soft skills
soft_skills_tot %>%
  ggplot(aes(y=reorder(soft_skill,count),x=count,fill=soft_skill)) +
  geom_bar(stat = ’identity’,position=position_dodge()) +
  geom_text(aes(label=skill_count_pct), vjust=1.6, color="black",
            position = position_dodge(0.9), size=3.5) +
  labs(y = ("Soft Skills"),x = ("Soft Skill Count"),
       title = ("Percentage of Soft Skill Found in Data Science Job Opps")) +
  theme_minimal()
```



```r
# create a one technical skills data.frame and agg for accurate counts
lookup_skills_1 <- c("python", "r", "sql", "hadoop", "spark", "tableau", "statistics", "analytics", "sa
skills_mentioned <- filter(job_phrases_1, w1 %in% lookup_skills_1)
tech_skills_1 <- aggregate.data.frame(x = skills_mentioned$freq,            # Sum by group
          by = list(skills_mentioned$w1),
          FUN = sum)
```

```r
# call the ngram function to extract all double word phrases from the job descriptions
job_descr_text <- job_descr_phrases
job_phrases_2 <- createNgram(job_descr_text,2)
job_phrases_2$w1w2 <- tolower(job_phrases_2$w1w2)
names(job_phrases_2)
```

```
## [1] "w1w2" "freq"
```

```r
job_phrases_2 <- job_phrases_2 %>% arrange(desc(freq))
```

```r
# create a second of double word technical skills data.frame and agg for accurate counts
lookup_skills_2 <- c("machine learning","articial intelligence","computer science", "data mining","data
skills_mentioned <- filter(job_phrases_2, w1w2 %in% lookup_skills_2)
tech_skills_2 <- aggregate.data.frame(x = skills_mentioned$freq,            # Sum by group
          by = list(skills_mentioned$w1),
          FUN = sum)
```

```r
# create a final tech skills df to hold all tech skills
tech_skills_tot <- tech_skills_1
```

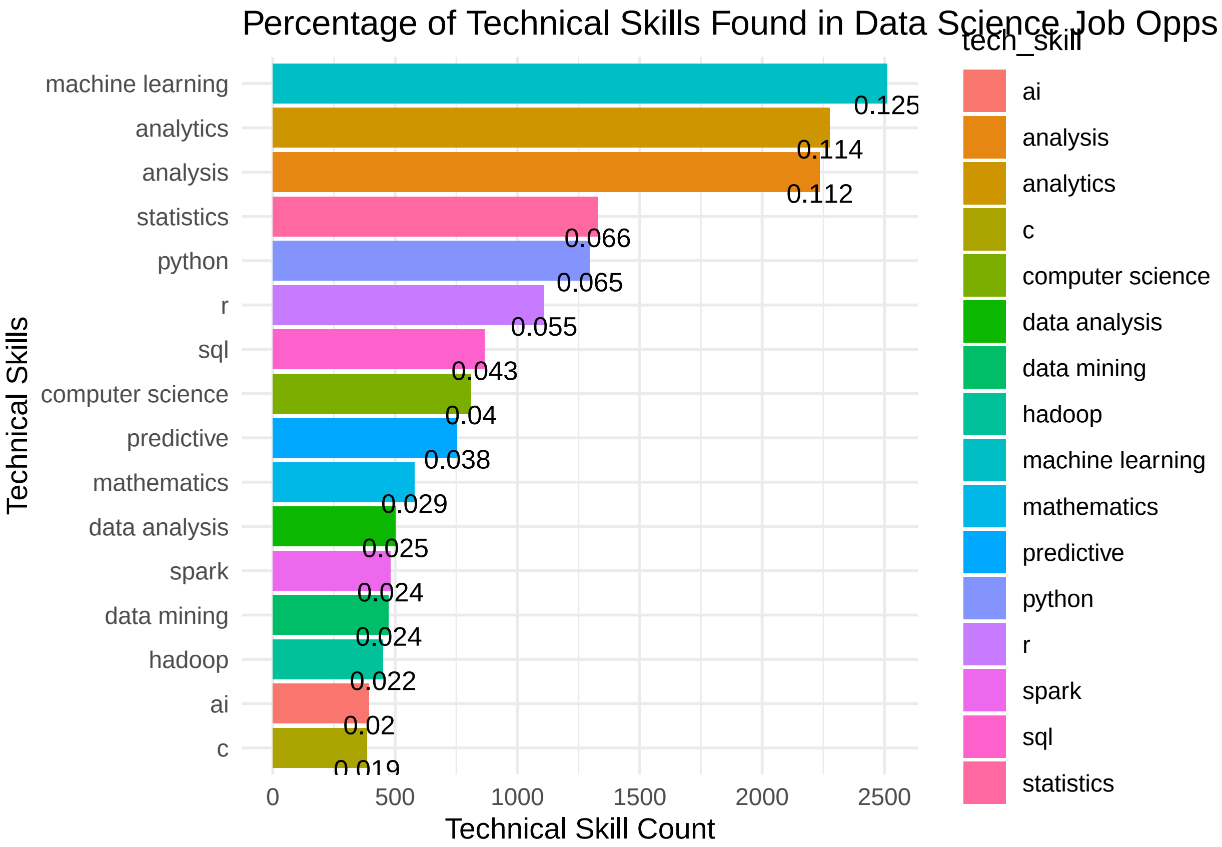
```r
tech_skills_tot <- rbind(tech_skills_tot,tech_skills_2) %>%
   dplyr::rename("tech_skill"=Group.1, "count"=x)
```

```r
# calc a percentage field and append to respective rows to tech skills df
tech_tot_count <- as.integer(sum(tech_skills_tot$count))
tot_skill_count_pct <- round(tech_skills_tot$count/tech_tot_count, 3)
tech_skills_tot <- cbind(tech_skills_tot, tot_skill_count_pct)
```

```r
# filter those tech skills where the percentage is less than 0.020
(tech_skills_tot_10 <- tech_skills_tot %>% filter(tot_skill_count_pct >= 0.019))
```

```
##          tech_skill count tot_skill_count_pct
## 1                ai   394               0.020
## 2          analysis  2236               0.112
## 3         analytics  2277               0.114
## 4                 c   385               0.019
## 5            hadoop   451               0.022
## 6       mathematics   579               0.029
## 7        predictive   754               0.038
## 8            python  1296               0.065
## 9                 r  1109               0.055
## 10            spark   481               0.024
## 11              sql   866               0.043
## 12       statistics  1328               0.066
## 13 computer science   810               0.040
## 14    data analysis   501               0.025
## 15      data mining   474               0.024
## 16 machine learning  2511               0.125
```

```r
# plot the top technical skills
tech_skills_tot_10 %>%
  ggplot(aes(y=reorder(tech_skill,count),x=count,fill=tech_skill)) +
  geom_bar(stat = 'identity',position=position_dodge()) +
  geom_text(aes(label=tot_skill_count_pct), vjust=1.6, color="black",
            position = position_dodge(0.9), size=3.5) +
  labs(y = ("Technical Skills"),x = ("Technical Skill Count"),
       title = ("Percentage of Technical Skills Found in Data Science Job Opps")) +
  theme_minimal()
```


Percentage of Technical Skills Found in Data Science Job Opps
Technical Skills
machine learning
analytics
analysis
statistics
python
r
sql
computer science
predictive
mathematics
data analysis
spark
data mining
hadoop
ai
c
0.125
0.114
0.112
0.066
0.065
0.055
0.043
0.04
0.038
0.029
0.025
0.024
0.024
0.022
0.02
0.019
0
500
1000
1500
2000
2500
Technical Skill Count
tech_skill
ai
analysis
analytics
c
computer science
data analysis
data mining
hadoop
machine learning
mathematics
predictive
python
r
spark
sql
statistics

Conclusion