Business Problem

Process Flow

Data Cleaning

Univariate Analysis

Bivariate Analysis

Hypothesis/Model

Results

Findings

Recommendations

Limitations – data from particular period need more data, cant be generalized for all, more models , skewed variables were used. Few variables were not present in the data.

R –

https://machinelearningmastery.com/machine-learning-in-r-step-by-step/

<https://towardsdatascience.com/collecting-data-science-cheat-sheets-d2cdff092855> - list of all

<https://www.cheatography.com/gabriellerab/cheat-sheets/matplotlib-pyplot/>

<https://python-graph-gallery.com/wp-content/uploads/Seaborn_Cheatsheet_Datacamp.png>

<https://www.datacamp.com/community/blog/pandas-cheat-sheet-python> - important

https://s3.amazonaws.com/assets.datacamp.com/blog\_assets/Python\_Bokeh\_Cheat\_Sheet.pdf

/// dataquest pandas cheat sheet

/// <https://github.com/mk9440/ZS-Young-Data-scientist-2018-Winner-Solution-rank-3> - Mayank

<https://github.com/sat14Siv/ZS-Data-Science-Challenge/tree/master/Offline%20Data-a-thon> – Satz with ppt

Coolchen

# importing required packages

import pandas as pd

import pandas\_profiling

import numpy as np

import seaborn as sns

from scipy import stats

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

from scipy.spatial.distance import cdist

import math

import more\_itertools as mit

#model libraries

from sklearn import preprocessing, svm, metrics

from sklearn.feature\_selection import SelectKBest, chi2

from sklearn.feature\_selection import f\_classif

from sklearn.preprocessing import MinMaxScaler

from sklearn.model\_selection import train\_test\_split, StratifiedKFold

from imblearn.over\_sampling import SMOTE, ADASYN

from sklearn.ensemble import (RandomForestClassifier, AdaBoostClassifier,

RandomForestRegressor, ExtraTreesClassifier, GradientBoostingClassifier, BaggingClassifier)

from sklearn.linear\_model import LogisticRegression

from sklearn import tree

from sklearn.feature\_selection import RFE

from sklearn.neighbors import KNeighborsClassifier

from scipy.stats import mstats, stats

from sklearn.ensemble import VotingClassifier

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

from sklearn.impute import SimpleImputer

from sklearn.model\_selection import cross\_val\_score

from scipy.spatial.distance import cdist

from sklearn.metrics import roc\_curve, roc\_auc\_score

from sklearn.svm import SVC, LinearSVC

from sklearn.naive\_bayes import GaussianNB, BernoulliNB

from sklearn import linear\_model

from sklearn.gaussian\_process import GaussianProcessClassifier

from sklearn.gaussian\_process.kernels import RBF

from sklearn.neural\_network import MLPClassifier

# importing the data

df = pd.read\_csv('telecom\_churn.txt',sep='\t')

df.columns = [c.replace(' ', '\_') for c in df.columns]

#Profile report

profile = df.copy().profile\_report(title='Pandas Profiling Report')

#Check dtypes

print(df.dtypes)

#remove strings from a numeric column

def ConvNum(col):

return(pd.to\_numeric(col, errors='coerce'))

#Find columns that are binary

def CheckBinary(df):

flaglist=[]

checkFlag = df.mode(numeric\_only=True).head(1)

for i in checkFlag.columns:

if checkFlag[i][0] == 0 or checkFlag[i][0]==1:

flaglist.append(i)

return flaglist

#Find highly unique columns

def CheckUnique(df):

unq\_list = []

unq = df.nunique()/len(df)

for i in unq:

if i >= 0.98:

unq\_list.append(unq[unq==i].idxmax())

return unq\_list

#Outlier removal

def RemOutlier(df):

print("Z-score method")

numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']

cols = list(df.select\_dtypes(include=numerics).columns)

getFlags = CheckBinary(df[cols])

cols = set(cols) - set(getFlags)

unq\_cols = CheckUnique(df[cols])

cols = cols - set(unq\_cols)

newdf = df[cols]

newdf = newdf[(np.abs(stats.zscore(newdf)) < 3).all(axis=1)]

df = df[set(df.columns)-set(cols)].join(newdf,how='inner')

return df, getFlags, unq\_cols

def categorical\_summarized(dataframe, x=None, y=None, hue=None, palette='Set1', verbose=True):

if x == None:

column\_interested = y

else:

column\_interested = x

series = dataframe[column\_interested]

print(series.describe())

print('mode: ', series.mode())

if verbose:

print('='\*80)

print(series.value\_counts())

sns.countplot(x=x, y=y, hue=hue, data=dataframe, palette=palette)

plt.show()

df['Night\_Calls'] = df['Night\_Calls'].apply(ConvNum)

#numeric columns

#This'll include "Flags" - that can be considered as categorical

numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']

print(df.select\_dtypes(include=numerics))

#categorical columns

#sample

print(df.select\_dtypes(exclude=numerics))

#Check for nan

cols = df.columns[df.isna().any()].tolist()

print(cols)

df[cols] = df[cols].fillna(np.mean) #any other approach would also work

#remove outliers

df, binaryCols, unq\_cols = RemOutlier(df)

print(binaryCols, unq\_cols)

#numeric, catrgorical and Flags

num\_cols = df.select\_dtypes(include=numerics).columns

flags = CheckBinary(df[num\_cols])

num\_cols = set(num\_cols) - set(flags)

categ\_cols = set(list(df.columns)) - num\_cols

categ\_cols = categ\_cols - set(unq\_cols) - set(CheckUnique(df))

#CountPlots

fig, ax = plt.subplots(2,2)

fig.suptitle("Count Plots", fontsize="x-large")

x = y = 0

for i in categ\_cols:

sns.countplot(data=newdf, x=i, ax=ax[x][y])

y+=1

if y==2:

x, y = x+1, 0

if x==2:

x = y = 0

fig, ax = plt.subplots(2,2)

fig.suptitle("Count Plots", fontsize="x-large")

#basic functions (python)

a = [1,2,3]

print(a[0:2])

print(a[0::2])#2 elements from 1st position

a.extend([5])

print(a)

df.describe() #summary stats

df.info() #null, non-null info

df.sort\_values(by=['State','Night\_Mins'],ascending=True,inplace=False) #sort in pandas

df.drop\_duplicates() #drop dups

df.drop\_duplicates(['State','Churn'])

#lambda functions

df['Churn\_Flag'] = df['Churn'].apply(lambda x: "True" if x==1 else "False")

#replace

data = pd.Series([1., -999., 2., -999., -1000., 3.])

data.replace([-999,-1000],np.nan,inplace=True)

#rename

df.rename(columns={'Churn\_Flag':'ChurnF'})

#value counts

pd.value\_counts(df['Churn\_Flag'])

#groupby

df.groupby(['Churn','VMail\_Plan'])['Night\_Mins'].mean()

#slicing

df.iloc[0:5] #rows

df.iloc[3] #4th row - index = 3

df.loc[:,'Churn':'VMail\_Plan'] #set of columns

df.ix[:,15:] #slice columns

df[df['State'].isin(['OH','KS'])] #isin

df.query('Night\_Mins > Eve\_Mins | Eve\_Mins < Day\_Mins') #where A > B or B < C

#another way of grouping by

df.pivot\_table(index=["Int'l\_Plan","VMail\_Plan"], columns='Churn', values=["Account\_Length"], aggfunc="count",margins=True).reset\_index()

df.shape #shape of dataframe

df.isnull().sum() #no. nulls total

df.dropna() #drop NA

#Models

#Pre-model functions

#Correlated columns

#X - Dataframe

def correlation(X,heatmap=False):

corr = X.corr(method="pearson").abs()

upper = corr.where(np.triu(np.ones(corr.shape), k=1).astype(np.bool))

# Find features with correlation greater than 0.95

to\_drop = [column for column in upper.columns if any(upper[column] > 0.80)]

X = X.drop(to\_drop,axis=1)

if heatmap==True:

sns.heatmap(corr[(corr >= 0.5) | (corr <= -0.4)],

cmap='viridis', vmax=1.0, vmin=-1.0, linewidths=0.1,

annot=True, annot\_kws={"size": 8}, square=True)

return X

#Significance Testing

#KruskalWallis - for categorical data - This changes depending on the problem and the function will change as well

def sigTest(X):

sig\_list = []

for col in X:

H, pval = mstats.kruskalwallis(list(X[col]),list(y['col'])) #y['col'] is the predictor variable

if pval < 0.05:

sig\_list.append(col)

X = X[sig\_list]

return X

#Feature Selectin Example

#Top 20 Features by default - Works with RandomForestClassifier

#THis varies depending on the given case study

#Useful link - https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/

def featSelect(X, y, model, topN=20):

model.fit(X,y)

feat\_importances = pd.Series(model.feature\_importances\_, index=X.columns)

feat\_importances = feat\_importances.nlargest(topN).to\_frame(name='values')

feat\_importances.reset\_index(level=0, inplace=True)

cols = feat\_importances['index'].values

X = X[cols]

return X, feat\_importances

#Model Evaluation

#Split Train and Test

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.20, random\_state=42, stratify=y)

#Pre-model

X\_train = correlation(X\_train) #Remove highly correlated columns

X\_train = sigTest(X\_train) #Don't forget to change y['col'] to point to y\_train predictor flag

#models

#Choose any

model = RandomForestClassifier(n\_estimators=25,class\_weight='balanced')

model = LogisticRegression(C=800, solver='lbfgs')

model = AdaBoostClassifier(learning\_rate=0.9,base\_estimator=LogisticRegression(penalty='l2',C=700))

model = GradientBoostingClassifier(learning\_rate=0.0099,n\_estimators=200)

model = RandomForestClassifier(n\_estimators=100,min\_samples\_leaf=60,class\_weight='balanced')

model = BaggingClassifier(n\_estimators=50,base\_estimator=LogisticRegression(penalty='l2',C=700),verbose=20)

model = BernoulliNB()

model = linear\_model.SGDClassifier(max\_iter=1000, tol=1e-3)

model = LinearSVC(penalty='l2',tol=1e-5)

kernel = 1.0 \* RBF(1.0)

X\_sel, feat\_importances = featSelect(X\_train, y\_train['col'], model, 25)

#USE SMOTE for imbalance data

#sm = SMOTE(random\_state=20)

#x\_res, y\_res = sm.fit\_sample(X\_train, y\_train['col'])

clf = model #chosen from above list or our own

clf.fit(X\_train, y\_train['col']) #Fit the model

output = clf.predict(X\_test[X\_train.columns])

#classification report

#Precision, recall, f-score

classif\_report = metrics.classification\_report(y\_true=y\_test,y\_pred=output, output\_dict=True)

classif\_report = pd.DataFrame(classif\_report).transpose()

conf\_matrix = pd.crosstab(y\_test['col'], output, rownames=['True'], colnames=['Predicted'], margins=True)

balanced\_accuracy = (((conf\_matrix[0][0] \* 1.0)/conf\_matrix['All'][0]) + ((conf\_matrix[1][1] \* 1.0)/conf\_matrix['All'][1]))/2

#Auc-roc-score

auc = roc\_auc\_score(y\_test['CKD\_Flag'], probs)

# calculate roc curve

fpr, tpr, thresholds = roc\_curve(y\_test['CKD\_Flag'], probs)

plt.title('Receiver Operating Characteristic')

plt.plot(fpr, tpr, 'k', label = 'AUC = %0.2f' % auc)

plt.legend(loc = 'lower right')

plt.ylabel('True Positive Rate')

plt.xlabel('False Positive Rate')

plt.grid(False)

plt.show()

#KMeans

#Elbow method Example

distortions = []

K = range(1,10)

for k in K:

kmeanModel = KMeans(n\_clusters=k).fit(df\_clust)

kmeanModel.fit(df\_clust)

distortions.append(sum(np.min(cdist(df\_clust, kmeanModel.cluster\_centers\_, 'euclidean'), axis=1)) / df\_clust.shape[0])

# Plot the elbow

plt.plot(K, distortions, 'bx-')

plt.xlabel('k')

plt.ylabel('Distortion')

plt.title('The Elbow Method showing the optimal k')

plt.show()

#Silhoutte score example

from sklearn.metrics import silhouette\_score

range\_n\_clusters = list (range(2,10))

for n\_clusters in range\_n\_clusters:

clusterer = KMeans (n\_clusters=n\_clusters)

preds = clusterer.fit\_predict(df\_clust)

centers = clusterer.cluster\_centers\_

score = silhouette\_score (df\_clust, preds, metric='euclidean')

print (n\_clusters, score)

#Get cluster

km = KMeans(n\_clusters=5 ,random\_state=21).fit(df\_clust)

cluster\_map = pd.DataFrame()

cluster\_map['data\_index'] = df\_clust.index.values

cluster\_map['cluster'] = km.labels\_

merged = pd.merge(df, cluster\_map, left\_index=True, right\_on='data\_index')

#PCA example

from sklearn.decomposition import PCA

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

merged3 = sc.fit\_transform(merged2)

merged3 = sc.transform(merged2)

pca = PCA(n\_components=10)

principalComponents = pca.fit\_transform(merged3)

principalDf = pd.DataFrame(data = principalComponents

, columns = ['pc1','pc2','pc3', 'pc4', 'pc5', 'pc6', 'pc7', 'pc8', 'pc9', 'pc10'])

print(pca.explained\_variance\_ratio\_)

print(pca.components\_)

/// pvalue, rsquared, rsquared\_adj and so on using OLS

<https://www.kaviglobal.com/blog/linear-regression-analysis-python-quick-start-guide/> - R2 vif in python

import pandas as pd

import numpy as np

from sklearn import datasets, linear\_model

from sklearn.linear\_model import LinearRegression

import statsmodels.api as sm

from scipy import stats

import matplotlib.pyplot as plt

diabetes = datasets.load\_diabetes()

X = diabetes.data

y = diabetes.target

X2 = sm.add\_constant(X)

est = sm.OLS(y, X2)

est2 = est.fit()

print("summary()\n",est2.summary())

print("pvalues\n",est2.pvalues)

print("tvalues\n",est2.tvalues)

print("rsquared\n",est2.rsquared)

print("rsquared\_adj\n",est2.rsquared\_adj)

#All the attributes

for attr in dir(est2):

if not attr.startswith('\_'):

print(attr)

predictions = est2.predict(X2)

print(est2.predict(X2[:3,:]))

from sklearn.metrics import r2\_score

print("r2\_score",r2\_score(y,predictions))

/// <https://www.geeksforgeeks.org/ml-adjusted-r-square-in-regression-analysis/>

/// R Chapter 1

#=======================================

# IDS 462, Session 1

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#=======================================

# First steps in R

#====

# Orientation

#=

getwd() # shows us where the working directory is

dir.create('IDS462') # creates a directory called IDS462. Note that R is, for the most part, accpets both single and double quatation marks

setwd("IDS462")

getwd()

# Install a package

#=

# use RStudio's gui, or

install.packages("dplyr")

library(dplyr) # makes all functions in dplyr available, or

dplyr:: # brings up a list of functions to choose from

#=====================

# Create a folder called IDS462 under a temp directory (create it if you need to)

# In RStudio, change your working directory to the newly created directory

# Install the tidyverse package

#=====================

## Solution

# you can create a folder by using your computer's operating system, or through R:

dir.create("C:/temp/IDS462")

setwd("C:/temp/IDS462")

install.packages('tidyverse')

# Basic operations

#=

5+6

8\*7

10\*5 / 0.5

# Vectors

somet

x <- "IDS462"

y <- "is where you will learn that"

z <- "equals"

something <- c(x,y, a, "plus", b, z, c) # c is a function, stands for combine, or concatenate

something

length(something) # length is also a function

something; length(something)

#=====================

# Assign the numbers 4, 3.7, 4, 3.5 to object score

# Assign the number 4 to object tests

# Assign the result of score divided by tests to object avg.score

# Print the content of avg.score

# Assign the number 4 to object score

# Did avg.score change? Why?

#=====================

## Solution

score <- c(4,3.7,4,3.5)

tests <- 4

avg.score <- score/tests

avg.score

score <- 4

avg.score

# It's content did not change because the object avg.score was unchanged

# if u say avg.score <- score/tests again then the result ll be 1. need to define avg.score again

# Data types

#=

is.vector(something)

is.vector(something)==FALSE

class(something) # class is also a function

class(c)

class(a>b)

a>b

?class # help on a function

help(package="ggplot2") # help on a package (along with its functions)

# Classes

#=

a <-5

b<-"hello"

c<-FALSE

class(a)

class(b)

class(c)

is.logical(a)

is.numeric(a)==FALSE

class(a)!=class(c)

#=====================================

# What is the data class of avg.score?

# Add the value "employee" to avg.score like this: avg.score <- c("employee", avg.score)

# Print the content of avg.score

# What is the class of avg.score now? Why is it different?

#=====================================

## Solution

class(avg.score)

# [1] "numeric"

avg.score <- c("employee", avg.score)

avg.score

class(avg.score)

# [1] "character"

# The data type has changed because we added a character value to avg.score

#. Functions (note that R is a functional prog. language)

a1<-1:10

a2<-seq(1:10)

identical(a1,a2)

?identical

?seq()

a3 <- seq(from=1, to=10, by=0.01)

a3 # oops. don't forget to assign!

a3 <- round(a3,1)

a3

head(a3)

tail(a3)

#=====================

# Generate a sequence of numbers from 7 to 100, by 4.05, and assign it to my.seq

# View the last 4 numbers like this: tail(my.seq,n=4)

# Round the last four numbers of my.seq to a single decimal, and assign them to new.seq

#=====================

## Solution

my.seq <- seq(from=7, to=100, by=4.05)

new.seq <- round(tail(my.seq,n=4),1)

# Environment

#=

ls() # shows a list of objects that were saved into the R environment (also see them in the top-right pane)

save(a,b,c, file="firststeps.Rdata")

list.files()

rm(list=ls())

ls()

load("firststeps.Rdata")

ls()

/// Chapter – 2

#=======================================

# IDS 462, Session 2

# Copyright Zack Kertcher, PhD, 2017. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#=======================================

#### WORKING WITH DATA

## Load libraries

library(tidyverse)

library(lubridate)

## Business Challenge:

# A supply firm wants to target with coupons and other

# incentives clients who are 1 standard deviation or more below the

# mean.

# Below is a small data subset

#ID Plumbing HVAC Electric Member StartDate

#2045 10040 238 63 0 5/4/2012

#2046 12000 248 55 0 6/1/2008

#2047 8240 201 45 1 12/9/2015

#2048 7160 206 38 0 4/25/2009

#2049 9900 188 50 1 5/19/2016

#2050 10240 213 70 1 9/1/2010

#2051 8200 201 38 0 6/12/2014

#2052 12500 238 75 0 9/15/2015

#2053 11460 223 68 0 1/30/2009

#2054 10440 216 45 0 6/23/2012

# Inputing the data into R

#==

# Method 1: Using vectors

ID <- c(2045,2046,2047, 2048, 2049,2050,2051,2052,2053,2054) # Note that R is \*not\* sensitive to spaces

Plumbing <- c(10040,12000,8240,7160,9900,10240,8200,12500,11460,10440)

HVAC <- c(238,248,201,206,188,213,201,238,223,216)

Electric <- c(63,55,45,38,50,70,38,75,68,45)

Member <- c(0,0,1,0,1,1,0,0,0,0)

StartDate<-c("5/4/2012","6/1/2008","12/9/2015"

,"4/25/2009","5/19/2016","9/1/2010","6/12/2014"

,"9/15/2015","1/30/2009","6/23/2012")

# If vectors are the same length, you may combine them into a data.frame

# check if vectors are the same length:

length(ID); length(Plumbing) # ... Yes! all have 10 elements

# Combine into a data frame

supply1 <- data.frame(ID, Plumbing, HVAC, Electric, Member, StartDate)

# Alternatively, I can combine the columns using cbind

supply2 <- cbind(ID, Plumbing, HVAC, Electric, Member, StartDate)

class(supply1); class(supply2) # Interesting

supply2 <- as.data.frame(supply2)

class(supply1); class(supply2) # Good!

# Method 2: Input all data directly

supplydatainput <- data.frame(

ID=c(2045,2046,2047, 2048, 2049,2050,2051,2052,2053,2054),

Plumbing=c(10040,12000,8240,7160,9900,10240,8200,12500,11460,10440),

HVAC=c(238,248,201,206,188,213,201,238,223,216),

Electric=c(63,55,45,38,50,70,38,75,68,45),

Member=c(0,0,1,0,1,1,0,0,0,0),

StartDate=c("5/4/2012","6/1/2008","12/9/2015","4/25/2009","5/19/2016",

"9/1/2010","6/12/2014","9/15/2015","1/30/2009","6/23/2012")

)

# Method 3: Read from clipboard

# First, copy content

# Then, run the following:

supplyclip <- read.delim("clipboard")

# Method 4: Read from file

supplyfile <- read.csv("supplysamp.csv", header=T) # Or specify the full path, e.g., "/home/data/supplysamp.csv"

# Also, try using file.choose()

# Method 5: Read using tidyverse (readr)

supplytibble <- read\_csv("C:/Users/DilipKumar/Desktop/462 R/supplysamp.csv")

class(supplytibble)

is.data.frame(supplytibble)

is.tibble(supplytibble)

# We can also convert a dataframe to a tibble, using the function: tibble()

# using data.frame() will convert to a data frame.

#=============================

# (1) Download and open the employees.csv file from Blackboard (e.g., Excel, Google Sheets)

# (2) Use Method 1 above (using vectors) to recreate the data

# Save as a data frame called EmpManual

# (3) Load the csv file directly into R as a tibble called employees

#=============================

# Solution

#=============================

employee <- c('AA','BJ','FS', 'MS', 'NR', 'RW')

gender <- c(0, 1, 0, 1, 0, 1)

salarykd <- c(6.1,8.3,7,122.7,88,100.8)

months <- c(87,2,14,43,18,5)

EmpManual <- data.frame(employee, gender, months, salarykd)

employees <- read\_csv("C:/Users/DilipKumar/Downloads/employees.csv")

## Data Types

#=

# Start by examining each column and note expected data type

# We have 4 common data types in R:

# Numeric (commonly, integer and double -- for precision)

# Character (strings)

# Factor (also numeric, but with distinct levels)

# Date (character, but is recognized as having date properties)

# What are the data types in our data?

supplyfile

# Look at each column more closely

str(supplyfile)

str(supplytibble)

# Alternatively, using tidyverse

glimpse(supplyfile)

# Do not worry about the difference between integer (whole numbers) and double (decimals, double-precision floating point number)

class(supplyfile$ID)

is.numeric(supplyfile$ID)

# Change columns to their correct type

supplyfile$ID <- as.character(supplyfile$ID)

supplyfile$Member <- as.factor(supplyfile$Member)

supplyfile$StartDate <- as.character(supplyfile$StartDate) # more on dates shortly

glimpse(supplyfile)

# Examine data in a spreadsheet-like format

View(supplyfile)

# All looks fine, except StartDate. It is a date variable but it is currently a character.

# We will learn more about handling date variables, but let's take a shortcut

# Inspect date format

head(supplyfile$StartDate) # Look for the format! This one is in m/d/y format

# Use lubridate (loaded with tidyverse) to convert the character into a date

supplyfile$StartDate <- mdy(supplyfile$StartDate)

glimpse(supplyfile)

summary(supplyfile$StartDate) # Nice!

#=============================

# (1) Examine the variables (columns) in the employees data frame/tibble

# (2) Think what data type each one should be

# (3) If needed, convert to the correct data type

#=============================

# Solution

#=============================

# Gender needs to change to a factor. All other columns are fine.

employees$gender\_factor <- as.factor(employees$gender)

glimpse(employees)

# Also notice:

class(employees$gender) # but

typeof(employees$gender)

## Sorting Data

# Method 1: Base R (order)

supplyfile[order(supplyfile$Plumbing),] # default is ascending order

supplyfile[order(-supplyfile$Plumbing),] # use negative sign for descending order

supplyfile <- supplyfile[order(-supplyfile$Plumbing, # first, descending order of Plumbing

Member),] # then by member

# Method 2: Tidyverse (dplyr)

supplyfile %>% arrange(desc(HVAC), Plumbing)

# If you are getting warning messages, ignore them. Or ask R to suppress them: options(warn=-1); options(warn=0) returns to default.

#=============================

# (1) Sort the employees data frame by salary (descending)

# and gender (ascending)

# (2) Using methods 1 (based R) and 2 (tidyverse)

##=============================

# Solution

#=============================

employees[order(-employees$salarykd, employees$gender),]

employees %>% arrange(desc(salarykd), gender)

## Selecting (subsetting) Data

# We now want to "extract" or subset those customers who are 1 sd below others

# First, consider how to subset data in R, with a couple simple examples

# Let's start with subsetting all customers who are members

# then selecting all customers who are purchasing 1sd below the mean

# Method 1: Using R Base

supplyfile$Member==1 # This is a logical statement that evaluates if membership equals 1

# What we now need it subset all the TRUEs

supplyfile[supplyfile$Member==1,] # From supplyfile df, select rows that are Member==1 (TRUE), and subset all columns.

# If we also want to select columns

supplyfile[supplyfile$Member==1, c("ID","Member", "Plumbing")] # Note change in structure!

# If we want to select rows and columns by position:

supplyfile[c(1:3), c(2:4)]

# Remember to assign the result to a new object:

supplymembers <- supplyfile[supplyfile$Member==1,]

# Now for the 1sd below the mean

# First, we will generate a variable called total\_purchase

supplyfile$total\_purchase <- supplyfile$Plumbing+supplyfile$HVAC+supplyfile$Electric

glimpse(supplyfile)

# Next, compute threshold

summary(supplyfile$total\_purchase)

mean(supplyfile$total\_purchase); sd(supplyfile$total\_purchase)

threshold <- mean(supplyfile$total\_purchase)-sd(supplyfile$total\_purchase)

# Finally, select target customers

target <- supplyfile[supplyfile$total\_purchase<=threshold,]

# Method 2: Using the subset function in base R

target1 <- subset(supplyfile, total\_purchase<=threshold, # selects rows

select=c("Member", "total\_purchase")) # selects columns

# Method 3: Using tidyverse

target2 <- supplyfile %>% filter(total\_purchase<=threshold) %>% select(StartDate, Member, total\_purchase) # selects columns

#=============================

# (1) Subset the rows with male employees

# then compute the highest salary for this gender

# (2) Repeat the above with female employees

# (3) Subset the gender and salary of employees who are in the

# company less than the average number of months

#

# Brainstorm as many ways you can think of!

#=============================

# Solution

#=============================

empmale <- employees[employees$gender\_factor==1,]

MinSalary\_Male <- min(empmale$salarykd)

MinSalary\_Female <- employees %>% filter(gender\_factor==0) %>% select(salarykd) %>% min()

avgmon <- mean(employees$months)

LessAvgMon <- employees %>% filter(months < avgmon) %>% select(gender\_factor, salarykd)

## Data values conversion

# We'll start with a simple conversion known as recoding.

# Suppose that instead of 0,1 for membership we want to change to meaningful labels

# 0=NotMember, 1=Member

# Method 1: Using R base (simple assignment)

supplyfile$MemberFactor[supplyfile$Member==1]<-"Member"

supplyfile$MemberFactor[supplyfile$Member==0]<-"NotMember"

# Note the structure: NewVariableName[DataFrame$ColumnName == (or any other condition) value ] <- "NewLabel" (can also be a number, TRUE/FALSE, etc.

# Method 2: Using tidyverse (dplyr)

supplyfile$MemberFactor1 <- recode\_factor(supplyfile$Member, "0"="NotMember", "1"="Member")

# There is also the fct\_recode from the forcats package, however, it is still a bit buggy.

# Now let's recode our customers as "Target" and "NonTarget"

# (1 sd below the mean are the Target).

# Method 1: Assign values

supplyfile$target[supplyfile$total\_purchase<=threshold]<-"Target"

supplyfile$target[supplyfile$total\_purchase>threshold]<-"NonTarget"

table(supplyfile$target)

# Method 2: ifelse

ifelse(supplyfile$total\_purchase<=threshold, "Target", "NonTarget") %>% table()

## Extra: Consider variable distribution

#=

# Before manipulating data, you should check the distribution of the variables you are working with

# For example, you would need to scale or perform other mathematical transformations

# before subsetting and recoding values

# We will work more on this in descriptive analytics, but meanwhile:

# Let's run a histogram and a boxplot

hist(supplyfile$Plumbing)

boxplot(supplyfile$Plumbing)

# Actually, let's plot them side-by-side

par(mfrow=c(2,1))

hist(supplyfile$Plumbing)

boxplot(supplyfile$Plumbing)

# Now the other three variables

par(mfrow=c(2,2))

hist(supplyfile$Plumbing)

hist(supplyfile$Electric)

hist(supplyfile$HVAC)

hist(supplyfile$total\_purchase)

boxplot(supplyfile$Plumbing)

boxplot(supplyfile$Electric)

boxplot(supplyfile$HVAC)

boxplot(supplyfile$total\_purchase)

###############

# Additional resources:

# swirl: modules: 6, 7, and 12

# Learning R: ch. 4 (only vectors), ch. 5 (data frames), ch. 7

# Practice the above using other data sets.

# Find a list of data sets here: https://vincentarelbundock.github.io/Rdatasets/datasets.html

/// chapter 3

#=======================================

# IDS 462, Session 3

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#=======================================

## Load libraries and data

#=

library(tidyverse)

library(lubridate)

library(stringr)

# Getting data

#=

# Building permits from the City of Chicago: https://data.cityofchicago.org/Buildings/Building-Permits/ydr8-5enu/data

building\_permits <- read\_csv("Building\_Permits.csv") # read\_csv is much faster compared to read.csv.

# Random sample

#=

# The building permit data file is too large.

# For efficiency, we are going to work on a smaller random sample.

# We then can apply the code to the initial (full) data

# Using R base

numrows <- nrow(building\_permits)

sampsize <- round(numrows\*0.05,0)

set.seed(462)

samp <- sample(numrows, sampsize)

bp <- building\_permits[samp,]

dim(bp)

# Using tidyverse

set.seed(462)

bp <- sample\_frac(building\_permits, 0.05) # this function takes a random sample equals to a percentage

# sample\_n samples using a perset number of cases

# Missing data

#=

# Start by examining the data. There are often columns with many missing cases.

# NA, without quotations, is the indicator for missing data

# The is.na function returns logical vector to identify missing values

is.na(bp$PERMIT\_TYPE)

table(is.na(bp$PERMIT\_TYPE))

# Let's try another one

table(is.na(bp$ISSUE\_DATE))

# You get the idea.

# The problem is that we have 131 columns!

# This number of columns is typical for real data.

# Running the is.na function per column is not particularly efficient.

# Here is a better approach:

colSums(is.na(bp)) # colSums sums up the values from a numeric vector/column

View(colSums(is.na(bp)))

# It is preferable to view NAs in percentage

missing <- round(colSums(is.na(bp))/nrow(bp),2)

View(missing)

# What if you want the opposite, how many rows HAVE data?

notmissing <- round(colSums(!is.na(bp))/nrow(bp),2)

View(notmissing)

# Drop selected columns with NAs

# We know that we can't do much with columns with too many missing values

keep <- missing[missing < 0.4]

keep <- names(keep)

bp1 <- bp[,keep]

bp1

# Examine NAs for patterns

# the easiest way is to consider NAs by specific columns of interest, especially the DV

summary(bp[is.na(bp$CONTRACTOR\_1\_PHONE), c("ESTIMATED\_COST","AMOUNT\_PAID")]) # oops!

str(bp1)

# we need to convert these rows to numeric

# you already know how to convert one column at a time.

# here is one way to convert multiple columns to the same type.

convert <- c("ESTIMATED\_COST","AMOUNT\_PAID")

pb[convert] <- sapply(bp[convert],as.numeric) # not good enough #sapply loop function

head(bp$ESTIMATED\_COST); head(bp$AMOUNT\_PAID) # that's why!

# String manipulation will help addressing this problem

# NOTES

# For the most part, the tidyverse approach to getting percentage missing values per column is cumbursome.

# See options at: https://stackoverflow.com/questions/41819422/how-to-drop-columns-by-passing-variable-name-with-dplyr#41819540

# To drop columns with tidyverse:

bp %>% select(-c("PIN1", "PIN2"))

####################

# Go to Data.gov and find a CSV data set with at least 200 rows and 10 columns

# If your data is very large, take a smaller random sample.

# Your data likely has missing values in different columns

# Examine all the data frame for missing values

# What is the average (mean) missing values in the data frame?

# How many columns have more than 20% missing values?

# Save all the columns that have 80% or more values as a separate data frame

####################

# Manipulating strings (characters)

#=

# We use a common procedure in computing languages, called regular expression

# It offers a simple yet powerful way to handle messy data.

# It is used to find, extract, replace, modify, and add characters.

# Other uses include splitting and gathering data.

# Match patterns

# R base

grep("$", cost) # row output

grep("$", cost, value=T) # value output

grepl("$", cost) # logical output

grep("$50", cost)

grep("\\$50", cost) # escape special symbols, such as $,.() etc.

grep("\\$50|\\$20", cost) # the | stands for "or"

grep("\\b500\\b", cost, value=T) # \\b indicates boundaries

# Tidyverse

# No good alternative to grep, however:

cost %>% str\_detect("\\$50") # same as grepl

# Finding location of pattern

cost %>% str\_locate("\\$50") # R base equivalent is gregexpr

# Replace patterns

# R base

digitpattern <- "\\$|\\.00|\\,"

gsub(digitpattern, "", cost) # remember to reassign the processed output and convert to the correct data type if needed

bp$ESTIMATED\_COST <- as.numeric(gsub(digitpattern, "", bp$ESTIMATED\_COST))

# Tidyverse

cost %>% str\_replace(digitpattern, "") %>% as.numeric() %>% head()

# Extracting a pattern

# R base

nums <- "\\d{1,}" # Extract only digits that follow one another

regmatches(cost, regexpr(nums, cost))

# Tidyverse

# Much easier!

cost %>% str\_extract(nums)

####################

# Copy several columns that contain missing values in your data to another name

# Change missing values in two columns in your data to "Missing"

# Change all other values in these columns to "NotMissing"

# Remove odd characters in the data, such as commas

# and other non-numeric symbols in numeric variables

####################

# Reshaping data with string patterns

#=

# Splitting a column

# First test the pattern

issue <- bp1$ISSUE\_DATE[1:20]

# R base

strsplit(issue, "/")

splitissue <- strsplit(issue, "/")

class(splitissue)

sapply(splitissue, "[", 1) # month "column"

sapply(splitissue, "[", 3) # year "column"

# Tidyverse

bp1 %>% separate(ISSUE\_DATE, c("month", "day", "year"), sep="/") %>% select(ISSUE\_DATE, month, day, year) %>% head() # oy!

bp1 %>% separate(ISSUE\_DATE, c("month", "day", "year"), sep="/", remove=F) %>% select(ISSUE\_DATE, month, day, year) %>% head()

# Combining multiple columns

# R base

paste(bp$STREET\_NUMBER, bp$STREET\_DIRECTION, bp$STREET\_NAME, bp$SUFFIX, sep=" ") %>% head()

# doesn't work because there are spaces in the column name "STREET DIRECTION"! let's fix this

paste(bp$STREET\_NUMBER, bp$"STREET DIRECTION", bp$STREET\_NAME, bp$SUFFIX, sep=" ") %>% head()

# create a new column

bp1$address <- paste(bp$STREET\_NUMBER, bp$"STREET DIRECTION", bp$STREET\_NAME, bp$SUFFIX, sep=" ")

# Tidyverse

bp1 <- bp1 %>% unite(address1, c("STREET\_NUMBER","STREET DIRECTION","STREET\_NAME"), sep=" ")

head(bp1$address); head(bp1$address1)

#################

# Separate your date column(s) into year and months.

# And/or select other columns that you think should be separated for analysis

# Combine back the above and/or other relevant columns into one

#################

# Date (and time) data

#=

# R's time objects are based on UNIX chron. This means dates and times (seconds), since 1970.

# Time zone needs to be provided.

issue %>% class()

# R base

as.Date(issue) # nope

as.Date(issue, format="%m/%d/%Y") %>% class #Y is for four digit year. y is for two.

# ?strptime for a comprehensive list.

date1 <- as.Date(issue, format="%m/%d/%Y")

# Lubridate

# First, understand date format, in this case, month, day, year.

mdy(issue) %>% print() %>% class()

# Higher frequency time data

# For higher frequency times, it is best to use POSIX

sometimes <- c("1/1/2017 7:00:01", "1/2/2017 7:00:20", "1/3/2017 7:02:15")

as.POSIXct(sometimes, format="%m/%d/%Y %H:%M:%S") # POSIXct (calendar time) based on seconds since UNIX epoch; POSIXlt (local time).

# Time zone

# Default is local machine.

Sys.timezone()

# To input a specific time zone:

as.POSIXct(sometimes, format="%m/%d/%Y %H:%M:%S", tz="Africa/Abidjan")

# Lubridate

times <- mdy\_hms(sometimes, tz="Africa/Abidjan") %>% print()

# NOTES

# For a list of timezones: https://en.wikipedia.org/wiki/List\_of\_tz\_database\_time\_zones

# Extracting time components (year, month, etc.)

#=

# R base

# We simply use regex

format(times, "%Y")

format(times, "%y")

format(times, "%B")

weekdays(times)

months(times)

quarters(times)

format(times, "%S")

# Lubridate

year(times)

yday(times)

minute(times)

second(times)

# Applied to the building permit data

bp1$issue\_date <- mdy(bp1$ISSUE\_DATE)

summary(bp1$issue\_date) # looks like we have NAs

####################

# Using R base, convert the date columns in your data to a date class

# Using lubridate, convert the date columns in your data to a date class

# What is the median date?

# Extract quarter into a new column

# Extract year into a new column

# Combine quarter and year into a new column

####################

###############

# Additional resources:

# Learning R, ch 13

# Boehmke, Bradley. 2016. Data Wrangling with R , 1st ed. Springer.

# http://vita.had.co.nz/papers/tidy-data.pdf

# https://www.rstudio.com/wp-content/uploads/2016/09/RegExCheatsheet.pdf

# <https://www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf>

/// chapter – 4

#=======================================

# IDS 462, Session 4

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#=======================================

# Load libraries

#=

library(car)

library(corrplot)

library(gmodels)

library(psych)

library(tidyverse)

library(vcd)

# Inspect the data

#=

set.seed(462)

bp %>% sample\_n(200) %>% View()

# Wrangle

#=

bp1 <- bp %>% select(permit="PERMIT\_TYPE", date="ISSUE\_DATE", est\_cost="ESTIMATED\_COST",

amnt\_waived="AMOUNT\_WAIVED", amnt\_paid="AMOUNT\_PAID", total\_fee="TOTAL\_FEE",

contractor\_city="CONTRACTOR\_1\_CITY"

) %>% drop\_na()

nrow(bp1)/nrow(bp)

gsub("PERMIT - ", "", bp1$permit) %>% head()

bp1$permit <- gsub("PERMIT - ", "", bp1$permit) %>% as.factor()

bp1$date <- lubridate::mdy(bp1$date)

numcols <- c("est\_cost", "amnt\_waived", "amnt\_paid", "total\_fee")

bp1[numcols] <- bp1[numcols] %>% mutate\_all(funs(gsub("\\$","",.))) %>% mutate\_all(funs(as.numeric)) # convert to numeric

set.seed(462)

samp <- sample(1:nrow(bp), 20)

bp1[samp,] %>% select(est\_cost, amnt\_paid, amnt\_waived)

bp1$contractor\_city <- as.factor(bp1$contractor\_city)

bp1 <- bp1 %>% select(PERMIT\_TYPE, ISSUE\_DATE, ESTIMATED\_COST, AMOUNT\_WAIVED, AMOUNT\_PAID, CONTRACTOR\_1\_CITY) %>% glimpse()

# Descriptive univariate statistics

#=

summary(bp1) # Summarizes all the variables. This is a good starting point.

# instead of scientific notation

options(scipen=99)

summary(bp1)

# I don't like the 9999999999. How many do we have?

bp1 %>% filter(est\_cost==9999999999) %>% count() # got it

bp1 <- bp1 %>% filter(est\_cost!=9999999999) # got it

# Now for factor variables

table(bp1$contractor\_city)

# Tidyverse

bp1 %>% group\_by(contractor\_city) %>% count()

bp1 %>% group\_by(contractor\_city) %>% count() %>% arrange(desc(n)) # too many levels (cities)

bp1$contractor\_city <- gsub("CHGO", "CHICAGO", bp1$contractor\_city)

bp1$city <- ifelse(grepl("CHICAGO", bp1$contractor\_city), "Chicago", "Other")

## Analysis

#=

# Univariate

# Factor variables

tab\_city <- table(bp1$city)

tab\_permit <- table(bp1$permit)

prop.table(tab\_city)

prop.table(tab\_permit)

plot(tab\_city)

barplot(tab\_city)

barplot(prop.table(tab\_city))

barplot(prop.table(tab\_city), col=c("red", "orange"),

main="Distribution of Contractors by Location",

ylim=c(0,0.6))

box(lwd=2)

# Numeric variables

# There are many functions and packages that provide excellent descriptive statistics for numeric variables, and we can use individual functions from base R, like mean(), and sd().

# Explore

mean(bp1$est\_cost)

median(bp1$est\_cost)

sd(bp1$est\_cost)

fivenum(bp1$est\_cost)

quantile(bp1$est\_cost, c(0.1, 0.3, 0.9))

IQR(bp1$est\_cost)

# Tidyverse

bp1 %>% summarize(avg = mean(est\_cost), median = median(est\_cost), std = sd(est\_cost))

# Psych is another great option

desc<-describe(bp1[numcols])

class(desc)

View(round(desc,2))

# In addition to parameters, we should inspect distributions

# Histogram and density plots

hist(bp1$est\_cost)

hist(bp1$est\_cost[bp1$est\_cost<200000])

hist(bp1$est\_cost[bp1$est\_cost<50000])

hist(bp1$est\_cost[bp1$est\_cost<50000], breaks=50) # perhaps focus on <20k

hist(bp1$est\_cost[bp1$est\_cost<20000], main="Histogram of Estimated Cost (without extremes)")

plot(density(bp1$est\_cost))

plot(density(log(bp1$est\_cost))) # Consider this transformation

# overlaid hisogram and density plots

est\_cost\_cut <- bp1$est\_cost[bp1$est\_cost<20000]

hist(est\_cost\_cut, prob=T, col="steelblue", main="Distribution of Estimated Cost (without extremes)")

rug(est\_cost\_cut, col="gray", lwd=0.5)

lines(density(est\_cost\_cut), col="orange", lwd=3)

boxplot(bp1$est\_cost) # not great

boxplot(est\_cost\_cut, col="orange", main="Distribution of Estimated Cost (without extremes)") # better!

# qqplot plots the distribution of the variable against a normal distribution

qqnorm(bp1$est\_cost)

qqline(bp1$est\_cost, col="red", lwd=2) # again...

qqnorm(log(est\_cost\_cut+1))

qqline(log(est\_cost\_cut+1), col="red", lwd=2) # not surprising

#################

# Examine the other numeric variables in the bp data. What did you find?

# Examine a couple of numeric variables in "your" data. What did you find?

#################

# Bivariate relationships

#=

# Two factors

# Explore

tab\_permit\_city <- table(bp1$permit, bp1$city)

ftab\_permit\_city <- ftable(bp1$permit, bp1$city)

xtab\_permit\_city <- xtabs(~bp1$permit + bp1$city)

tab\_permit\_city

ftab\_permit\_city # supports more than 2 factors

xtab\_permit\_city # supports more than 2 factors + statistical tests

p1\_ftab\_permit\_city <- prop.table(ftab\_permit\_city, margin=1) # row %

p2\_ftab\_permit\_city <- prop.table(ftab\_permit\_city, margin=2) # column %

addmargins(p1\_ftab\_permit\_city) # can also be used without prop.table

addmargins(p2\_ftab\_permit\_city)

# Plot

barplot(prop.table(xtab\_permit\_city), col=1:10, ylim=c(0,1), main="Permits by City", beside=T)

barplot(prop.table(xtab\_permit\_city), col=1:10, ylim=c(0,1), main="Permits by City")

legend("toprigh", levels(bp1$permit), lty=1,lwd=4, col=1:10)

box(lwd=1.5)

mosaic(xtab\_permit\_city, shade=T) # better when there are less levels

# Test

summary(xtab\_permit\_city) # or more directly

chisq.test(bp1$permit, bp1$city) # but there's a better option

CrossTable(bp1$permit, bp1$city, prop.c = F, prop.r = F,

dnn = c('Permit', 'City'), format=c("SPSS"), chisq = T)

#################

# Examine the relationships among factor variables in "your" data. What did you find?

#################

# Two numeric variables

# Plot

plot(bp1$total\_fee~bp1$est\_cost, col="steelblue", pch=20, cex=0.75) # right, we need to address the outliers

bp2 <- bp1[bp1$est\_cost<20000,]

plot(bp2$total\_fee~bp2$est\_cost, col="steelblue", pch=20, cex=0.75)

plot(log(bp2$total\_fee+1)~log(bp2$est\_cost+1), col="steelblue", pch=20, cex=0.75)

abline(lm(log(total\_fee+1)~log(est\_cost+1), data=bp2), col="red", lwd=2) # Not great

# Correlation matrix

cormat <- cor(bp2[,numcols])

round(cormat,2)

corrplot(cormat, method="circle", addCoef.col="grey", type="upper")

# pairs(bp1[,numcols]) # this one is slow. Try it later!

# Test

# Correlation

cor(bp1$est\_cost, bp1$total\_fee)

cor.test(bp1$est\_cost, bp1$total\_fee) # As expected

#################

# Examine the relationship between other numeric and variables in the bp data. What did you find?

# Examine the relationships among numeric variables in "your" data. What did you find?

#################

# Factor and a numeric variable

# Explore

aggregate(est\_cost ~ city, data = bp1, FUN="mean", na.rm=T)

aggregate(est\_cost ~ city, data = bp1, FUN="sd", na.rm=T)

# Tidyverse

bp1 %>% group\_by(city) %>% summarize(avg=mean(est\_cost), median=median(est\_cost), sd=sd(est\_cost))

# Plot

boxplot(est\_cost ~ city, data=bp2, main="Comparing Distributions by City",

xlab="City", ylab="Estimated Cost", col=c("orange", "steelblue"))

# Test

est\_cost\_model <- aov(log(est\_cost+1)~city, data=bp2)

summary(est\_cost\_model)

# Tukey pairwise comparisons

TukeyHSD(est\_cost\_model)

#################

# Examine the relationships among numeric and factor variables in the data.

# Which variables seem to be highly correlated?

#################

###############

# Additional resources:

# Learning R, ch 14

/// chapter 5

#=========================================================

# IDS 462, Session 5

# Data Visualization

#=========================================================

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#====================================

## (INSTALL) LOAD LIBRARIES

#=

library(ggplot2) # already intalled

library(gridExtra)

library(plotly)

library(RColorBrewer)

library(vcd) # already intalled

## LOAD DATA

load("Session 5.RData")

## BACKGROUND

#=

# graphics is what base R uses

# developed later, grid is the foundation for both lattice and ggplot2

# lattice is now a part of R

# ggplot2 is a separate package

# R base

#= Plotting parameters

# Univariate plots

#==

#plot\_par <- par()

# Factor

barplot(table(cars$transmission))

barplot(table(cars$transmission), ylim=c(0,140),

axes=F, #to remove axes

col=terrain.colors(2),

cex.main=1.5,

col.main="darkgray",

main="Cars by Transmission Type")

axis(side = 2, at = seq(from=0, to=140, by=20))# zoom in!

box(lwd=5)

#pdf("barplot.pdf")

#########

# Generate a bar plot for driveway

# Use different colors for having/not-having a driveway

# Title the plot

# Encapsulate in a box

#########

#ans:

barplot(table(realestate$driveway))

barplot(table(realestate$driveway), ylim=c(0,500),

axes=F, #to remove axes

col=terrain.colors(3),

cex.main=1.5,

col.main="darkgray",

main="Real Estate by Driveway")

axis(side = 2, at = seq(from=0, to=500, by=20))# zoom in!

box(lwd=3)

pdf("barplot.pdf")

dev.off()

# Numeric variable

# Histogram

hist(baseball$pay,

col=rainbow(10),

breaks=10,

main="Histogram of budget")

# detour: advanced color option

display.brewer.all()

brewer.pal.info

# apply a color theme

hist(baseball$pay,

col=brewer.pal(10, "Set3"),

breaks=10,

main="Histogram of budget")

# change lable names

hist(baseball$pay,

col=brewer.pal(10, "Set3"), #set3, paird choosing one pallette #if 10 choosing 10 colors. if choose 2 here ll keep circlng i.e. repeat

breaks=10,

xlab="Budget in million $",

ylab="Number of teams",

main="Histogram of budget")

box(lwd=2) # lwd is line width

# Histogram with overlaid density and rug plots

hist(baseball$pay,

freq=F,

col=brewer.pal(10, "Dark2"),

breaks=10,

xlab="Budget in million $",

main="Histogram of budget")

rug(jitter(baseball$pay), col="darkgray")

lines(density(baseball$pay), col="yellow", lwd=3) # Note how the lines function is used to overlay the density plot

box(lwd=2)

# Boxplot

boxplot(baseball$pay,

col="coral",

main="Boxplot of budget")

# Side-by-side

par(mfrow=c(1,2)) # par is for plotting parameters; mfrow is for number of rows/columns

hist(baseball$pay, col=c("steelblue", "red"), freq=F, xlab="budget in Millions", main="Distribution of budget")

rug(jitter(baseball$pay), col="darkgray")

lines(density(baseball$pay), col="yellow", lwd=3) # Note how the lines function is used to overlay the density plot

box(lwd=1.5)

boxplot(baseball$pay, col="orange", main="Boxplot of budget") #imp for exam

dev.off() # closes and resets the plotting canvas

# Saving output

pdf("dist\_budget.pdf", height=7, width=8)

par(mfrow=c(1,2)) #think 1 row 2 column chnag 2,1 # exam need to giv verything lik this

hist(baseball$pay, col=c("steelblue", "red"), freq=F, xlab="budget in Millions", main="Distribution of budget")

rug(jitter(baseball$pay), col="darkgray")

lines(density(baseball$pay), col="yellow", lwd=3) # Note how the lines function is used to overlay the density plot

boxplot(baseball$pay, col="orange", main="Boxplot of budget")

box(lwd=1.5)

dev.off() # closes the pdf file

dev.off() # resets the plotting canvas

#########

# Generate plots for the distribution of realestate lotsize

# and the distribution of bedroorms (as a factor)

# x and y axes should be labeled properly,

# and the plots should have a title

# Save the plots side-by-side as a PDF file

#########

#solution:par(mfrow=c(1,2)) # par is for plotting parameters; mfrow is for number of rows/columns

hist(realestate$lotsize, col=c("steelblue", "red"), freq=F, xlab="budget in Millions", main="Distribution of lotsize")

rug(jitter(realestate$lotsize), col="darkgray")

lines(density(realestate$lotsize), col="yellow", lwd=3) # Note how the lines function is used to overlay the density plot

box(lwd=1.5)

boxplot(realestate$bedrooms, col="orange", main="Boxplot of budget")

# Bivariate plots

#==

# Two numeric variables

#-

plot(baseball$pct~baseball$pay)

# Improved version

plot(baseball$pct~baseball$pay, data=baseball,

main="Relationship between budget and PCT",

ylab="% wins",

xlab="budget in $million",

pch=20,

col="orange",

xlim=c(35, 200))

abline(lm(baseball$pct~baseball$pay), col="blue", lwd=2, lty=2) #linetype

lines(loess.smooth(baseball$pay,baseball$pct), col="red", lwd=2, lty=1)#shows we have smwth of a linear reln#regrn assumption is normally distributed but this is not so trying to smooth it out

# add identifiers (label observations)

text(baseball$pct~baseball$pay, cex=.6, col="steelblue", labels=baseball$team)

ticks <- c(35, 70, 100, 150, 200)

axis(side=1,at=ticks) # sides are clockwise from bottom=1 to right=4

abline(h=0.5, v=ticks, lty=2, col="gray") # also may use grid to add gridlines

box(lwd=2.5, lty=3) #wont ask in exam few things few might

#########

# Plots the relationship of price and lotsize

# Add colors, and change tick marks

# Add a regression line with a different color

# x and y axes should be labeled properly, and the plot should have a title

# Save the plots as a PDF file

#########

# Two factors

table(realestate$bedrooms, realestate$airco) # too few observations

re <- realestate %>% filter(bedrooms<5 & bedrooms >1)

re$bedrooms<-as.factor(re$bedrooms)

re$bathrms<-as.factor(re$bathrms)

tab<-table(re$bedrooms,re$airco)

proptab<-prop.table(tab)

barplot(proptab, main="Air Conditioning by Bedrooms",

names.arg=c("No", "Yes"),col=brewer.pal(3, "Set2"), legend=c("2 bdr","3 bdr","4 bdr"), ylim=c(0,1.1))

box(lwd=2, col="steelblue")

#########

# Plots the relationship of airco and recroom

# Add colors

# Title the plot

# Add a legend

# Save the plots as a PDF file

#########

# Numeric variable and a factor

boxplot(pay~division, data=baseball,

col=brewer.pal(3, "RdGy"),

main="Distribution of budget\nby divisions") # \n is a new line

#########

# Plot the relationship of realestate price and lotsize,

# by prefarea

# Add a regression line

# Add a legend

#########

## PLOTTING WITH GGPLOT2

#=

# Today, this is the most popular library for plotting in R (and increasingly in Python)

# It can get a bit complicated, because there are many options

# It is based on the grammar of graphics, similar to

# the notion of verbs (like dplyr, tidyr, and other packages in tidyverse)

# Uses the following grammar:

# data - must be mapped to aesthetic attributes

# aesthetics - ways to represent data

# geometry - describe plot type (bars, lines, points...)

# statistics - smoothers, bins...

# scale - associate spaces and actual data values

# coordinates - coordinate system on which data are displayed

# faceting - grouping subsets of data

# qplot = simple version, but less options

#=

# Example

#-

# Plotting three variables (two numeric and a factor)

ggplot(data=baseball, # data

aes(x=pay, y=pct, color=division)) + # aesthetic = variables

geom\_point(pch=20, size=4) + # geometry = plot type

#stat\_smooth(method="lm", se=T, linetype=1, lwd=1.5) + # added statistical plot (default is with CI, se=T)

labs(title="Relationship between PCT and Pay by League"

, x="Pay", y="%wins") # annotation

# Univariate

#-

ggplot(data=baseball, aes(x=pay)) +

geom\_histogram(fill="steelblue", color="gray", alpha=0.8, bins=10) +

geom\_rug(aes(x = pay, y = 0))

ggplot(data=baseball, aes(x=pay)) +

geom\_density()

# While overlaying histogram and density plot

# is possible with ggplot, it is overly complicated.

# Bivariate

#=

# Two numeric

ggplot(data=baseball) + aes(x=pay, y=pct) +

geom\_point(pch=16, color="coral") +

labs(title='Relationship between PCT and Pay',

x="Pay", y="PCT") + # x for xlab, y for ylab

geom\_smooth(method="lm", color="black", lwd=2)

# Two factors

ggplot(data=re) + aes(x=airco, fill=bedrooms) +

geom\_bar(position="stack")+

scale\_fill\_brewer(palette="Pastel2") +

labs(title='Relationship between Bedrooms and AC',

x="AC", y="Frequency")

ggplot(data=re) + aes(x=airco, fill=bedrooms) +

geom\_bar(position="dodge")+

scale\_fill\_brewer(palette="Pastel2") +

labs(title='Relationship between Bedrooms and AC',

x="AC", y="Frequency")

ggplot(data=re) + aes(x=airco, fill=bedrooms) +

geom\_bar(position="fill")+

scale\_fill\_brewer(palette="Pastel2") +

labs(title='Relationship between Bedrooms and AC',

x="AC", y="Frequency")

# Numeric and factor

baseball %>% ggplot(aes(x=pay, fill=division)) +

geom\_density(alpha=.3) # comparative density plot

# Three or more variables

# Facets are useful because they allow breaking up plots by groups.

ggplot(data=cars, aes(x=price)) +

geom\_histogram(bins=10) +

facet\_wrap(~transmission)

ggplot(data=cars, aes(x=mileage, y=price,

color=transmission)) +

geom\_point() +

facet\_grid(.~model) # grid used for two variables, wrap for one

# Two numeric and two factors!

ggplot(baseball, aes(x=wins, y=pay,

color=division,

shape=league)) +

geom\_point() + geom\_jitter()

# Statistical plots (e.g., linear and smoothers)

#-

p1 <- ggplot(data=realestate, aes(x=log(lotsize),y=log(price))) +

geom\_point() # it is possible, and advisable, to assign ggplot output into an object

p1 + geom\_smooth(method="loess") +

geom\_smooth(method="lm") # se = F will remove the CI band

# Arranging ggplots on canvas

#-

par(mfrow=c(1,2))

ggplot(data=baseball, aes(x=pay)) + geom\_histogram()

ggplot(baseball, aes(x=pay, y=pct, color=league, shape=division)) +

geom\_point() + geom\_jitter() # nope

p1 <- ggplot(data=baseball, aes(x=pay)) + geom\_histogram()

p2 <- ggplot(baseball, aes(x=pay, y=pct, color=league, shape=division)) +

geom\_point() + geom\_jitter()

grid.arrange(p1, p2, ncol=2)

# Saving ggplots

#-

ggsave(file="budgethist.png", plot=p2, width=5, height=5.5) # width and height are in inches

# As in R base you can also save as .pdf and many other formats

# Interactive Plots

#=

p1 <- plot\_ly(data = cars, x = ~mileage, y = ~price)

p1

# or better

p1.1 <- plot\_ly(data = cars, x = ~mileage, y = ~price,

type="scatter", mode="markers" ,

marker=list(color="steelblue" , size=10 , opacity=0.5))

p1.1

# heatmap (two factors with many levels)

p2 <- plot\_ly(data = cars, x = ~color,

y = ~as.factor(year))

p2

## ADDITIONAL RESOURCES

#=

# Grahpics in base R and packages:

# run demo(graphics)

# graphics parameters: http://www.statmethods.net/advgraphs/parameters.html

# https://cran.r-project.org/doc/contrib/Short-refcard.pdf

# Gohil, A. 2015. R Data Visualization Cookbook.

# ggplot2:

# Wickham, H. 2016. ggplot2: Elegant Graphics for Data Analysis, 2nd edition.

# Comprehensive resource for ggplot2: http://ggplot2.org/book/

# More up to date references: http://docs.ggplot2.org/current/index.html

# https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf

# Additional cool plots (with code):

# <http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html#Scatterplot>

/// chapter 6

#=========================================================

# IDS 462, Session 6

# Data Visualization, OLS

#=========================================================

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#====================================

library(corrplot)

library(gridExtra)

library(tidyverse)

library(plotly) #tableau related

library(RColorBrewer)

load("session6.Rdata") # baseball, cars, realestate

## Viz. continued ....

# Two factors

table(realestate$bedrooms, realestate$airco) # too few observations

re <- realestate %>% filter(bedrooms<5 & bedrooms >1)

re$bedrooms<-as.factor(re$bedrooms)

re$bathrms<-as.factor(re$bathrms)

tab<-table(re$bedrooms,re$airco)

proptab<-prop.table(tab)

barplot(proptab, main="Air Conditioning by Bedrooms",

names.arg=c("No", "Yes"),col=brewer.pal(3, "Set2"), legend=c("2 bdr","3 bdr","4 bdr"), ylim=c(0,1.1))

box(lwd=2, col="steelblue")

#########

# Plots the relationship of airco and recroom

# Add colors

# Title the plot

# Add a legend

# Save the plots as a PDF file

#########

#my ans:

table(realestate$recroom, realestate$airco) # too few observations

realestate$recroom<-as.factor(realestate$recroom)

realestate$airco<-as.factor(realestate$airco)

tab<-table(realestate$recroom,realestate$airco)

proptab<-prop.table(tab)

proptab

barplot(proptab, main="Air Conditioning by recrooms",

names.arg=c("No", "Yes"),col=brewer.pal(3, "Set2"), legend=c("yesr","nor"), ylim=c(0,1.1))

box(lwd=2, col="steelblue")

chisq.test(realestate$recroom, realestate$airco)

# Numeric variable and a factor

boxplot(pay~division, data=baseball,

col=brewer.pal(3, "RdGy"),

main="Distribution of budget\nby divisions") # \n is a new line

# A comparative kernel density plot

plot(density(baseball$pay[baseball$division=="Central"]), col="red", lwd=2, ylim=c(0,0.02))

lines(density(baseball$pay[baseball$division=="East"]), col="blue", lwd=2)

lines(density(baseball$pay[baseball$division=="West"]), col="green", lwd=2)

#########

# Plot the relationship of realestate lotsize

# and prefarea

#########

boxplot(lotsize~prefarea, data=realestate,

col=brewer.pal(3, "RdGy"),

main="Distribution of lotsize\nby prefarea") # \n is a new line

# A comparative kernel density plot

plot(density(realestate$lotsize[realestate$prefarea=="Yes"]), col="red", lwd=2, ylim=c(0,0.02))

lines(density(realestate$lotsize[realestate$prefarea=="No"]), col="blue", lwd=2)

## PLOTTING WITH GGPLOT2

#=

# Today, this is the most popular library for plotting in R (and increasingly in Python)

# It can get a bit complicated, because there are many options

# It is based on the grammar of graphics, similar to

# the notion of verbs (like dplyr, tidyr, and other packages in tidyverse)

#gg means grammer of graphics

# Uses the following grammar:

# data - must be mapped to aesthetic attributes

# aesthetics - ways to represent data

# geometry - describe plot type (bars, lines, points...)

# statistics - smoothers, bins...

# scale - associate spaces and actual data values

# coordinates - coordinate system on which data are displayed

# faceting - grouping subsets of data

# qplot = simple version, but less options

#=

# Example

#-

# Plotting three variables (two numeric and a factor)

ggplot(data=baseball, # data

aes(x=pay, y=pct, color=division)) + # aesthetic = variables

geom\_point(pch=20, size=4) + # geometry = plot type

#stat\_smooth(method="lm", se=T, linetype=1, lwd=1.5) + # added statistical plot (default is with CI, se=T)

labs(title="Relationship between PCT and Pay by League"

, x="Pay", y="%wins") # annotation

# But let's start with the basics

# Univariate

#-

# numeric

ggplot(data=baseball, aes(x=pay)) +

geom\_density()

ggplot(data=baseball, aes(x=pay)) +

geom\_histogram(fill="steelblue", color="gray", alpha=0.8, bins=10) +

geom\_rug(aes(x = pay, y = 0))

# factor

ggplot(data=baseball, aes(x=division))

# While overlaying histogram and density plot

# is possible with ggplot, it is overly complicated.

# Likewise, generating a boxplot for a single variable.

# Bivariate

#=

# Two numeric

ggplot(data=baseball) + aes(x=pay, y=pct) +

geom\_point(pch=16, color="coral") +

labs(title='Relationship between PCT and Pay',

x="Pay", y="PCT") + # x for xlab, y for ylab

geom\_smooth(method="lm", color="black", lwd=2)

# Two factors

ggplot(data=re) + aes(x=airco, fill=bedrooms) +

geom\_bar(position="stack")+

scale\_fill\_brewer(palette="Pastel2") +

labs(title='Relationship between Bedrooms and AC',

x="AC", y="Frequency")

ggplot(data=re) + aes(x=airco, fill=bedrooms) +

geom\_bar(position="dodge")+

scale\_fill\_brewer(palette="Pastel2") +

labs(title='Relationship between Bedrooms and AC',

x="AC", y="Frequency")

#this stretches the plot to 100% - there r 2 options above.. i orefer this use this in exam.. practise a lot of this

ggplot(data=re) + aes(x=airco, fill=bedrooms) +

geom\_bar(position="fill")+

scale\_fill\_brewer(palette="Pastel2") +

labs(title='Relationship between Bedrooms and AC',

x="AC", y="Frequency")

# Numeric and factor

baseball %>% ggplot(aes(x=division, y=pay)) +

geom\_boxplot(aes(fill = division)) +

scale\_fill\_brewer(palette="Set2")

# foll is density kernel plot

baseball %>% ggplot(aes(x=pay, fill=division)) +

scale\_fill\_brewer(palette="Spectral")+

geom\_density(alpha=.5) #alpha capcity of plot

# Three (or more) variables

# Facets are useful because they allow breaking up plots by groups.

ggplot(data=cars, aes(x=price)) +

geom\_histogram(bins=10) +

facet\_wrap(~transmission) #if u want his side by side split canvas mfrow kinda

ggplot(data=cars, aes(x=mileage, y=price,

color=transmission)) +

geom\_point() +

facet\_grid(.~model) #4factors #price vs mileage by tansmission by model #facets always #no much detail

# grid used for two variables, wrap for one

# Two numeric and two factors! #!another way yto do it

ggplot(baseball, aes(x=wins, y=pay,

color=division,

shape=league)) +

geom\_point() + geom\_jitter() #both difficukt to interpret

# Statistical plots (e.g., linear and smoothers)

#-

p1 <- ggplot(data=realestate, aes(x=log(lotsize),y=log(price))) +

geom\_point() # it is possible, and advisable, to assign ggplot output into an object

p1 + geom\_smooth(method="loess") +

geom\_smooth(method="lm") # se = F will remove the CI band #!not a perfect linear reln

# Arranging ggplots on canvas

#-

par(mfrow=c(1,2))#multple ggplot on same canvas #in r base use mfrow#lil tricky with ggplot

ggplot(data=baseball, aes(x=pay)) + geom\_histogram()

ggplot(baseball, aes(x=pay, y=pct, color=league, shape=division)) +

geom\_point() + geom\_jitter() # nope

p1 <- ggplot(data=baseball, aes(x=pay)) + geom\_histogram()

p2 <- ggplot(baseball, aes(x=pay, y=pct, color=league, shape=division)) +

geom\_point() + geom\_jitter()

grid.arrange(p1, p2, ncol=2)

# Saving ggplots

#-

ggsave(file="budgethist.png", plot=p2, width=5, height=5.5) # width and height are in inches

# As in R base you can also save as .pdf and many other formats

#########

# Using ggplot

# Plot the relationship of lotsize and price,

# by prefarea

# explain your findings

#########

ans:

ggplot(data=realestate) + aes(x=lotsize, y=price) +

geom\_point(pch=16, color="coral") +

labs(title='Relationship between LotSize and Price',

x="LotSize", y="Price") + # x for xlab, y for ylab

geom\_smooth(method="lm", color="black", lwd=2)

#shud use 2numeric one factor 2 scatterplot thing or use facet

# Interactive Plots

#=

p1 <- plot\_ly(data = cars, x = ~mileage, y = ~price)

p1 #too many bins

# or better

p1.1 <- plot\_ly(data = cars, x = ~mileage, y = ~price,

type="scatter", mode="markers" ,

marker=list(color="steelblue" , size=10 , opacity=0.5))

p1.1 #moving pointer says things

# heatmap (two factors with many levels)

p2 <- plot\_ly(data = cars, x = ~color,

y = ~as.factor(year))

p2

### OLS #least sq method#to predict given a dependent variable

# Regression assumptions

# The DV is normally distributed

# IV values are independent

# There is a linear relationship between the DV and IVs

# There is constant variance, such that the variance of the DV

# does not change with the levels of IVs (homoscedasticity)

# There is no multicollinearity (two or more IVs are highly correlated)

## Simple linear regression

#============================

# Our DV is price

# Let's correlate numeric vars (in this case only the DV and two IVs: mileage, year)

carsnum <- cars[,c(1,3,4)]

class(carsnum)

cormat <- cor(carsnum)

corrplot(cormat, addCoef.col = "gray")

# Both IVs exhibit a relationship

# Now we can model (regress) price given mileage #we seee correlan betn proce yr mileage#including in model

mod1<-lm(price ~ mileage, data=cars)

mod1

options(scipen=99)

summary(mod1) #adj rsq. by knowing mileage we can predict price with 65% accuracy

confint(mod1) #interpret it as fr a unit incease of mile per unit wats the price range

# What about price given year?

mod2<-lm(price ~ year, data=cars)

mod2

summary(mod2)

# we can also set confidence interval at 99%

confint(mod2, level=0.99)

# Predict specific values based on a model

# For example, the predicted price of a car with 5,10,100k miles:

predict(mod1 , data.frame(mileage =(c(5000 ,10000 ,100000) )),

interval ="confidence", level=0.95)

# For example, the predicted price of a car from years 2002,2005,2008:

predict(mod2 , data.frame(year =(c(2002 ,2005 ,2008) )),

interval ="confidence", level=0.95) # 1 2 3 is years

# How well did our model perform?

head(cars$price)-head(predict(mod1)) # or using this function

summary(residuals(mod1))

boxplot(residuals(mod1))

plot(cars$price~cars$mileage, pch=16, col="lightblue")

abline(mod1, col="red", lwd=3) # and this is what we've been plotting all along!

# R-sqrd

summary(mod1)$adj.r.squared # The simple (single IV) model explains about 65% of the variance

########################

# Build a regression model using lotsize as the regressor

# What is the predicted price for lot 500, 1000, 2000, at 99% confidence level?

# Find the adjusted R-sqrd

########################

# What about price given year?

mod2<-lm(price ~ lotsize, data=realestate)

mod2

summary(mod2)

# we can also set confidence interval at 99%

confint(mod2, level=0.99)

# Predict specific values based on a model

# For example, the predicted price of a car with 5,10,100k miles:

# For example, the predicted price of a car from years 2002,2005,2008:

predict(mod2 , data.frame(year =(c(500 ,1000 ,2000) )),

interval ="confidence", level=0.95)

# Diagnostics (residuals and influence/leverage) #nt fr exam

#=

par(mfrow=c(2,2)) # sets up the canvas to 4 plots, so that we don't need to go through them one by one

plot(mod1)

# or

dev.off()

plot(predict (mod1), residuals (mod1)) # we are looking for a "no pattern"/non-linearity

# finding outliers can be done this way as well

plot(hatvalues(mod1))

identify(hatvalues(mod1), col="red")

# or because it looks like we have 2 outliers, we can do this

tail(sort(hatvalues(mod1)), n=2)

outliers <- c(90, 149, 1)

cars1<-cars[-outliers,]

mod1.1<-lm(price ~ mileage, data=cars1)

summary(mod1.1); summary(mod1) # some improvement

########################

# Are there any outliers in the model you built?

# Regress without 5 most "extreme" outliers

# Did model performance change?

# Now try to model price using a log transformation of price and lotsize

# Which model do you think is better (logged or non-logged)? Why?

# Perform regression diagnostics to the logged model.

# Finally, run the above (logged) regression without outliers.

# Explain your findings.

########################

## ADDITIONAL RESOURCES

# DATA VISUALIZATION

#=

# Grahpics in base R and packages:

# run demo(graphics)

# graphics parameters: http://www.statmethods.net/advgraphs/parameters.html

# https://cran.r-project.org/doc/contrib/Short-refcard.pdf

# Gohil, A. 2015. R Data Visualization Cookbook.

# ggplot2:

# Wickham, H. 2016. ggplot2: Elegant Graphics for Data Analysis, 2nd edition.

# Comprehensive resource for ggplot2: http://ggplot2.org/book/

# More up to date references: http://docs.ggplot2.org/current/index.html

# https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf

# Additional cool plots (with code):

# http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html#Scatterplot

# OLS

#=

# OpenIntro Book, ch. 7

# Learning R Book, ch. 15 (Linear Regressions)

///c hapter 7

#=========================================================

# IDS 462, Session 7

# Midterm Review Session

#=========================================================

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#====================================

# Instructions

#=============

# An RData file has been prepared for you. The file consists of data you already know, the realestate data, and a new data frame, credit.

# Answer each one of the questions assigned to your group \*on your own\* -- a total of three questions. You have 1.25 hour.

# Discuss your solutions as a group. You have 0.5 hour, including a 10 min. break.

# Present solutions to class. Discuss issues/concerns. Each group gets 15 min.

# Tips:

# Make sure to properly allocate time! Start by exploring the new data frame and consider how to best answer these questions.

# Group I

#========

# realestate data

#=

# Is there a difference in the distribution of air conditioning by bedrooms as a factor? Use relevant statistics, plots and statistical tests. For your plots, use at least one ggplot (properly titled, annotated, and visually reasonable). Detail your findings.

# Common question: Which variables in the data in your view exhibit the strongest relationship with price? Provide evidence and explain your answer.

# credit data

#=

# \*Throughly\* examine the relationships of Income, Balance, Age, Gender, Ethnicity (all of these are potential IVS), with Rating (our DV). Which of these IVs are good predictors? Use statistics, plots, and tests, and provide a detailed answer.

# Group II

#=========

# realestate data

#=

# What is the relationship between price and lotsize. Use relevant statistics, plots and statistical tests. For your plots, use at least one ggplot (properly titled, annotated, and visually reasonable). Detail your findings.

# Common question: Which variables in the data in your view exhibit the strongest relationship with price? Provide evidence and explain your answer.

# credit data

#=

# \*Throughly\* examine the relationships of Limit, Cards, Education, Student, Married (all of these are potential IVS), with Rating (our DV). Which of these IVs are good predictors? Use statistics, plots, and tests, and provide a detailed answer.

load("Session 7 (review).RData")

View(realestate)

glimpse(realestate)

colSums(is.na(realestate))

summary(realestate$lotsize)

summary(realestate$price)

options(scipen=99)

#Question 1

#univariate analysis

summary(realestate$lotsize)

summary(realestate$price)

outlier\_values <- boxplot.stats(realestate$lotsize)$out

outlier\_values

outlier\_values1<-boxplot.stats(realestate$price)$out1

outlier\_values1

realestate <- realestate[-c(outlier\_values,outlier\_values1),]

ggplot(data=realestate) + aes(x=lotsize, y=price) +

geom\_point(pch=16, color="coral") +

labs(title='Relationship between price and lotsize',

x="lotsize", y="price") + # x for xlab, y for ylab

geom\_smooth(method="lm", color="black", lwd=2)

cor.test(realestate$lotsize, realestate$price)

#there is relationship between lotsize and price. Positive relationship. As the lotsize increases price increases

#Question 2

mod1<-lm(price ~ lotsize, data=realestate)

mod1

summary(mod1)

# we can also set confidence interval at 99%

confint(mod1, level=0.99)

predict(mod1 , data.frame(lotsize =(c(4000 ,10000 ,12000) )),

interval ="confidence", level=0.99)

boxplot(residuals(mod1))

plot(realestate$price~realestate$lotsize, pch=16, col="lightblue")

abline(mod1, col="red", lwd=3)

#there is a positive relationship between lotsize and price.

mod3<-lm(realestate$price~realestate$lotsize+realestate$bedrooms+realestate$bathrms+realestate$stories+realestate$driveway+realestate$recroom+realestate$fullbase+realestate$airco+realestate$gashw+realestate$garagepl+realestate$prefarea)

summary(mod3)

plot(realestate$price~realestate$lotsize+realestate$bedrooms+realestate$bathrms+realestate$stories+realestate$driveway+realestate$recroom+realestate$fullbase+realestate$airco+realestate$gashw+realestate$garagepl+realestate$prefarea)

#the variable that exhibit strongest relationship are lotsize,bathrooms,stroies,fullbaseyes,aircoyes,gashwyes,garagepl2

#question 3

colSums(is.na(credit))

glimpse(credit)

credit$Income <- as.numeric(credit$Income)

credit$Limit<-as.numeric(credit$Limit)

gsub("\\$","",credit)

summary(credit$Limit)

credit1<-credit

credit1<-na.omit(credit1)

View(credit1)

summary(credit1)

#check the distribution,multicollinearity

mod2<-lm(Rating~Limit+Cards+Education+Student+Married,data=credit)

mod2

Predictions<-predit.lm(mod2,credit1)

predict(mod2,credit1)

summary(mod2)

boxplot(residuals(mod2))

plot(credit1$Rating~credit1$Limit+credit1$Cards+credit1$Education+credit1$Student+credit1$Married, pch=16, col="lightblue")

anova(mod2)

#correlation test

#The independent variable that is significant is Limit to predit the rating.

/// chapter 9

#=========================================================

# IDS 462, Session 9 - OLS (cont.)

#=========================================================

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#====================================

## LOAD LIBRARIES AND DATA

#=

library(car)

library(effects)

library(psych)

load("session9.Rdata") # retail, realestate

# Regression assumptions (recap)

# The DV is normally distributed

# IV values are independent

# There is a linear relationship between the DV and IVs

# There is constant variance, such that the variance of the DV

# does not change with the levels of IVs (homoscedasticity)

# There is no multicollinearity (two or more IVs are highly correlated)

# Outlier detection utility function

outliers <- function(column) {

lowerq <- as.vector(quantile(column)[2]) # returns 1st quartile

upperq <- as.vector(quantile(column)[4]) # returns 1st quartile

iqr <- upperq-lowerq

# Moderate outliers

mod.outliers.upper <- (iqr \* 1.5) + upperq

mod.outliers.lower <- lowerq - (iqr \* 1.5)

mod.outliers <- which(column > mod.outliers.upper |

column < mod.outliers.lower)

print(paste("Moderate outlier:", mod.outliers))

#Extreme outliers

extreme.outliers.upper <- (iqr \* 3) + upperq

extreme.outliers.lower <- lowerq - (iqr \* 3)

extreme.outliers<-which(column > extreme.outliers.upper

| column < extreme.outliers.lower)

print(paste("Extreme outlier:", extreme.outliers))

}

# Step 1: Make sure data are clean

#=

summary(retailer)

# remove columns/rows, address NAs, convert types, if needed #catalog has only 4 so it shud be a factor

#suspect in categorical values is few levels having too few observation or no values # shud remove

#here nothing needed

# Step 2: Examine at a univariate level (transform if needed) #we learnt log #few might not be useful after changing to log so gonna c other transformations

#=

# factors have reasonable levels and distribution

# let's examine the numeric variables

View(describe(retailer[,c("income", "expense")])) #talk abt skew n spread #expense median 944 sd also similar so this has wider spread than first

plot(density(retailer$income)) # not perfect

plot(density(retailer$expense)) # skewed #since it had more sd has more skew as expectd

# outliers

boxplot(retailer$income) # outlier #no outlier

# detect and remove outlier

outliers(retailer$income)

#retailer <- retailer[-101,] #initially had so removed but now no

boxplot(retailer$expense) # quite a few outliers

outliers(retailer$expense) # only a few are extreme outliers #no extreme outlier now

retailer <- retailer[-c(48,193,448,497,603,675),]#extreme outliers r the first to go# only 50 rows dont want to remove even 5 moderate ouliers

#! instead of outliers can do transformations

# transformations

plot(density(log(retailer$expense))) # works

plot(density(sqrt(retailer$expense))) # better #wen we interpret results we ve to rem its in sqrt

#one unit of increase in sqrt of expense makes income increase by

plot(density(sqrt(retailer$income))) # nope

plot(density(log(retailer$income))) # not perfect #at this level ok to wrk if its smwt normal

#log sqrt can explain back but complex to explain with power transfor but betr to examine

# When dealing with complex transformations try:

summary(powerTransform(retailer$expense)) # interesting... let's compare #est power .25 power of the..taking sqrt twice p-value is 7.5e-9 so it means this transformation ll wrk

summary(powerTransform(retailer$income)) # confirms our choice! #power is .47 rounded power is .5 confirms sqrt n p-value also

par(mfrow=c(2,2))

plot(density(retailer$expense))

plot(density(log(retailer$expense)))

plot(density(sqrt(retailer$expense)))

plot(density(retailer$expense^0.23)) #this is power transformation power of .25 income power is 0.5 cuz sqrt

#do it fr income

par(mfrow=c(2,2))

plot(density(retailer$income))

plot(density(log(retailer$income)))

plot(density(sqrt(retailer$income)))

plot(density(retailer$income^0.5)) #if power transform too says .5 n gives gud p=value accept it #under time

# What if we have a "bad" numeric variable? #expense power plot betr than sqrt cuz both end tails. can use sqrt too

retailer$catalog <- as.numeric(retailer$catalog)

cor.test(retailer$expense, retailer$catalog) # pretty good, but #!low p-value so there is correlation

plot(density(retailer$catalog)) # months would have been better #multimodal distribution #read it #one yr #2yr

# Option 1: Try a power transformation (likely to fail)

summary(powerTransform(retailer$catalog))

plot(density(retailer$catalog^0.5)) # as expected #not very usful

#! survey data too ll be so lik 5 data we ll be der ll get same plots

#if transformations fail we ll try to build index #else try to turn it into fctr

# Option 2: Make an index/scale

# This option requires multiple comparable variables (high correlation, capture the same dimension)

# Very common in survey data. But we don't have such variables in the retailer data

# Option 3: Convert it into a factor

# First, check the distribution. If uneven, "bin" it to categories of about the same size

table(retailer$catalog) # looks fine

# Second, create a factor variable

retailer$catalog <- as.factor(retailer$catalog)

# Third, use the factor variable instead of the original variable in the model

#u got correlation n less p-value but dont rush it see the distribution step by step

# Final thoughts on transformations:

# Try the simplest forms, i.e., log, sqrt

# Try creating an index that consists of multiple highly related variables

# Try creating a factor out of "bad" variables

# Ultimately, we balance meeting modeling assumptions with interpretation. Avoid overly complex transformations!

#################

# Examine the realestate data for univariate numeric transformations

# If needed, create new transformed variable(s)

#################

#my ans:

par(mfrow=c(2,2))

plot(density(realestate$lotsize))

plot(density(log(realestate$lotsize)))

plot(density(sqrt(realestate$lotsize)))

plot(density(realestate$lotsize^-0.21))

dev.off()

#log looks betrfr lotsize

summary(powerTransform(realestate$lotsize))

summary(powerTransform(realestate$price))

par(mfrow=c(2,2))

plot(density(realestate$price))

plot(density(log(realestate$price)))

plot(density(sqrt(realestate$price)))

plot(density(realestate$price^-0.23))

#log or power trans both looks gud

#proper way i guess

boxplot(realestate$lotsize) # outlier

# detect and remove outlier

outliers(realestate$lotsize)

realestate <- realestate[-c(365,369,359,434,446,365),] #removing those rows which has outliers

boxplot(realestate$price) # outlier #no outlier

# detect and remove outlier

outliers(realestate$price)

realestate <- realestate[-c(332,362,375,416,93,94,416,433),]

#after this do the plots

summary(powerTransform(realestate$lotsize)) #pvalue is .15 so dont use it. devansh says shudn use rounded

summary(powerTransform(realestate$price))#pvalue not good

par(mfrow=c(2,2))

plot(density(realestate$lotsize))

plot(density(log(realestate$lotsize)))

plot(density(sqrt(realestate$lotsize)))

#log looks betr

dev.off()

#log looks betrfr lotsize

par(mfrow=c(2,2))

plot(density(realestate$price))

plot(density(log(realestate$price)))

plot(density(sqrt(realestate$price)))

#log looks betr

dev.off()

# Step 3: Examine bivariate relations (transform if needed)

#=

# Which IVs do we need to included

# Explore correlation between DV and IVs, and among IVs

# The idea is to have a simple model that explains the DV, but not too simple

# Before transformation

dev.off()

plot(retailer$expense~retailer$income, pch=16, col="darkgrey")#scatterplot#can c positive linear reln with few exceptions

cor.test(retailer$expense,retailer$income)#getting high correlation

plot(sqrt(retailer$expense)~retailer$income, pch=16, col="darkgrey")#even betr positive

# A correlation plot would be useful, but we only have two numeric variables in the data

# Let's explore other relationships in the data

# Now DV and IVs

summary(aov(expense ~ agegroup, data=retailer)) #\*\*\*

summary(aov(expense ~ gender, data=retailer)) #\*\*\*

summary(aov(expense ~ homeownership, data=retailer)) #\*\*\*

summary(aov(expense ~ maritalstatus, data=retailer)) #\*\*\*

summary(aov(expense ~ location, data=retailer)) #\*\*\*

#all r stat significant #so they gonna have a effect onDV so shud add them to analysis

#ve to determine reln among them to c collinearity

# Also consider relationships among IVs

chisq.test(retailer$catalog, retailer$agegroup) #\*\*\*

chisq.test(retailer$catalog, retailer$gender) # (NS)

chisq.test(retailer$catalog, retailer$homeownership) #\*\*\*

chisq.test(retailer$catalog, retailer$maritalstatus) #.

chisq.test(retailer$catalog, retailer$location) #.

chisq.test(retailer$agegroup, retailer$gender) #\*\*\*

chisq.test(retailer$agegroup, retailer$homeownership) #\*\*\*

chisq.test(retailer$agegroup, retailer$maritalstatus) #\*\*\*

chisq.test(retailer$agegroup, retailer$location) #\*

chisq.test(retailer$agegroup, retailer$gender) #\*\*\*

# you've got the gist

# we need to pay attention to these relationships!

###################

# Perform steps 1-3 for predicting house prices

# Note that we have already done much of this work in the past

###################

# Step 4: modeling

#=

mod1 <- lm(expense~income+agegroup+gender+homeownership+maritalstatus+location, data=retailer)

#usualyy we wont start by adding all variable #since all add stat sig added them

mod1 <- lm(sqrt(expense)~income+agegroup+gender+homeownership+maritalstatus+location, data=retailer)

#sqrt#first say signn or not n postive or negative then #for every one unit of increase of income it increases sqrt of one unit of expense

#the diff betn old n young ppl stat sig #the diff betn middle n young r not sig

#wen comparing factor see with other category not with DV # ppl out of state vs in state

#compared to female

#started with correlation so all had so included #plots too confirmed

#but after doing test will include only stat signifcantS

# Interpreting the results

summary(mod1)

# coefficients

summary(mod1)$coefficients[,1]

# dummies

# if we want another reference category

retailer$agegroup<-relevel(retailer$agegroup, ref=3)#usually r takes refernce which has low alphanumeric so female n not male #here fr young changing

#change ref=2 n c if it changes d defult refernce

mod2 <- lm(expense~income+agegroup+gender+homeownership+maritalstatus+location, data=retailer)

summary(mod2)

# Notice the R and Adjusted R-squared

summary(mod1)$r.squared

summary(mod1)$adj.r.squared # "penalty" for adding more variables #increase of 5% but shud lookat adj rsq

# Not bad, but we need to do some more work to fine-tune our model.

###################

# Build a model to predict house prices

# Find a model that has the best adj-r-squared

# Know how to interpret the results!

###################

#my ans

summary(aov(price ~ lotsize, data=realestate)) #shud be regrn

summary(aov(price ~ bedrooms, data=realestate))

summary(aov(price ~ bathrooms, data=realestate))

summary(aov(price ~ stories, data=realestate))

summary(aov(price ~ driveway, data=realestate))

summary(aov(price ~ recroom, data=realestate))

summary(aov(price ~ fullbase, data=realestate))

summary(aov(price ~ gashw, data=realestate))

summary(aov(price ~ airco, data=realestate))

summary(aov(price ~ garagepl, data=realestate))

summary(aov(price ~ prefarea, data=realestate))

moda <- lm(price~lotsize+bedrooms+bathrms+stories+driveway+recroom+fullbase+gashw+airco+garagepl+prefarea, data=realestate)

moda <- lm(log(price)~lotsize+bedrooms+bathrms+stories+driveway+recroom+fullbase+gashw+airco+garagepl+prefarea, data=realestate)

moda <- lm(sqrt(price)~lotsize+bedrooms+bathrms+stories+driveway+recroom+fullbase+gashw+airco+garagepl+prefarea, data=realestate)

summary(moda)

summary(moda)$r.squared

summary(moda)$adj.r.squared

# coefficients

summary(moda)$coefficients[,1]

# sir ans

#plot density price skwd

#so takes plot log #fr bathrms he wants to c if it is factor#convert #mod log cuz log helped #

#table (as.factor(real$bath)) #u see 4 bthm - 1 3-10 #dont skip any step

#1bthrm missing here is d ref uz 1 loe alpha #all stat sig #

# Interactions

#=

# Used when we suspect that the effect of a predictor might vary at different values (levels)

# This means, that the relationships between an IV and the DV is dependent on the value of another IV

mod2\_terms<-lm(log(price)~lotsize+bedrooms+airco+lotsize:prefarea, data=realestate) # adding an interaction term of lotsize and location

#we taking log cuz we saw tats d crct approach

#prefarea want to know interactionso including it with all other terms

#intercation is stat sig

summary(mod2\_terms) # Something is certainly going on here

library(effects)

#compares slopes of diff area #if there i effect there is interaction

plot(effect(term="lotsize:prefarea", mod=mod2\_terms)) #yes no diff so putting in one plot

plot(effect(term="lotsize:prefarea", mod=mod2\_terms, default.levels=20), multiline=T) # same plot #sig diff we c now in plot

#if we c both same on top of other no interaction

# You can try ^n interactions, but we try to avoid them unless they add considerable contribution to the model. We do it this way

mod2\_n\_terms<-lm(log(price)~lotsize+bedrooms+airco+bathrms\*bedrooms\*lotsize\*prefarea, data=realestate)

summary(mod2\_n\_terms) # Indeed, not very useful. #result - none of them sign #decision tree splits based on interaction

#################

# What other interactions exist in the realestate data?

#################

#my ans

str(realestate)

realestate$bedrooms <- as.factor(realestate$bedrooms)

realestate$bathrms <- as.factor(realestate$bathrms)

mod3\_terms<-lm(log(price)~lotsize+bedrooms+airco+bedrooms:bathrms, data=realestate)

summary(mod3\_terms)

#bedrooms:bathrms 5.743e-02 7.623e-03 7.534 2.44e-13 \*\*\* # so there is interaction effect

plot(effect(term="bedrooms:bathrms", mod=mod3\_terms)) #yes no diff so putting in one plot

plot(effect(term="bedrooms:bathrms", mod=mod3\_terms, default.levels=20), multiline=T) #jumps quickly from 5-6 pink

#green from 4-5 drops #bedrooms has effect on rpice but its mitigated by interaction it has with bathrms #guess this is wat he said

#gotta do the table

table(realestate$bedrooms) #remove 5 6

table(realestate$bathrms) #remove

#there is interaction betn 2 in terms of preidtcing the price

# Diagnostics

#=

plot(mod2) # This is a great way to identify multiple possible concerns with the model

# For example, outliers. #this gives outlier in bivariate #gotta remove #in all 4 plots the outliers reapats we gotta remove them

# Better yet:

par(mfrow=c(2,2)) # This function sets up the plotting canvas to 2x2, so that we don't need to go through them one by one. Now again:

plot(mod2)

# We could also plot diagnostics "manually", such as:

dev.off()

plot(predict (mod2), residuals (mod2))

plot(hatvalues(mod2))

identify(hatvalues(mod2), col="steelblue") # Neat, but this interactive feature doesn't always work, so:

tail(sort(hatvalues(mod2)), n=5)

# Cook's distance indicates other "suspects" as well. We need to examine them and decide what to do with these cases

realestate[c(103,185,414,491,365,369,383,366), c("price", "lotsize", "bedrooms", "airco", "prefarea")]

# We could remove these observations, and re-run our model

outliers <- c(103,185,414,491,365,369,383,366)

realestate\_new <- realestate[-outliers,]

nrow(realestate\_new)-nrow(realestate) # We removed 8

mod3<-lm(log(price)~lotsize+bedrooms+airco+lotsize:prefarea, data=realestate\_new)

# Did our model improve?

# We want to know to find the extent to which our models have improved, after "cleaning"

summary(mod2)$adj.r.squared-summary(mod1)$adj.r.squared

summary(mod3)$adj.r.squared-summary(mod2)$adj.r.squared

#################

# Add other variables to the model. Which outliers (observations/rows) did you find? What is the best adjusted R-squared that you got?

#################

## ADDITIONAL RESOURCES

#=

# Learning R, Ch. 15

# OpenIntro Statistics. Ch. 7

# R in Action (2nd. Edition), Ch. 8

/// chapter 10

#=========================================================

# IDS 462, Session 10 - (Wrapping up OLS), Logit

#=========================================================

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#====================================

library(effects)

load("Session10data.RData")

### Finish OLS

mod2\_terms<-lm(log(price)~lotsize+bedrooms+airco+lotsize:prefarea, data=realestate) # adding an interaction term of lotsize and prefarea

#prefarea doesnt ve main effect here but only has a stat sig intercation effect tats y it was inclded. we use log fr price to make it normal dist cuz assumption of ols is normal dist

summary(mod2\_terms) # Something is certainly going on here #adj rq is given here itself

#interpret as log(price)

#here we wanna c if lotsize really has effect or cuz of interaction with pref area the lotsize is having effect

plot(effect(term="lotsize:prefarea", mod=mod2\_terms, default.levels=20), multiline=T) # same plot

#wat we look here is - is there a diff slope? if so yes. Here bigger lotisize houses is pref area increases their price soon. intercation effect. so wen both lotsize n pref area meets has highr slope

# Another example of interaction of two factors

mod3\_terms<-lm(log(price)~lotsize+prefarea\*recroom, data=realestate) # adding an interaction term and main effect using the \* symbol

summary(mod3\_terms)

plot(effect(term="prefarea\*recroom", mod=mod3\_terms, default.levels=20), multiline=T) # same plot

#this is a 2 factor interaction n they r simple factors yes or no.

#here we put multiply sign

#interpretation the co-eff r small cuz of log(price). we need toexponentiate bt exam no time so it has .0007 units of log price; each sqft of lotsize increases the?? log price by ??

#prefrareayes has a effect on 25% higher as compared to not being in pref area

#the log price is higher bby 25%

#being in a recroom has a effect

#when having a recroom, the log price is higher bby 25% when compared to not havng a recroom

#prefarea n recroom has a slight sign interaction #when a house is in prefarea and has a recroom, the log price decreases by 15% as compared to not having both recroom and not being in prefarea

#wehn lots of levels r dr more complex right now ignore n also wen number n factor #focus on sign n stars

#adj rsq the indep var explains variance in log price by 38%

mod3<-lm(log(price)~lotsize+prefarea+recroom, data=realestate) # adding an interaction term and main effect using the \* symbol

summary(mod3)

#now adj r2 is 38 adding interaction terms increased the model performance by 1% which is not much. #in exam simply pay attention to it

#its because the interaction p-value was one star

# Diagnostics

#=

plot(mod3\_terms)#gives 4 plots # This is a great way to identify multiple possible concerns with the model

# For example, outliers.

#always ll ve residual i want to make sure residual is normally distributed

#initially saw outlier fr univariate now seeing at model level

# Better yet:

par(mfrow=c(2,2)) # This function sets up the plotting canvas to 2x2, so that we don't need to go through them one by one. Now again:

plot(mod3\_terms)

# We could also plot diagnostics "manually", such as: #similare plots using diff codes

dev.off()

plot(predict (mod3\_terms), residuals (mod3\_terms))

plot(hatvalues(mod3\_terms))

identify(hatvalues(mod3\_terms), col="steelblue") # Neat, but this interactive feature doesn't always work, so:

tail(sort(hatvalues(mod3\_terms)), n=5)

# Cook's distance indicates other "suspects" as well. We need to examine them and decide what to do with these cases

realestate[c(103,104,414), c("price", "lotsize", "recroom", "prefarea")]

#414 lotsize more n price is so less

#103, 104 highre price tats y

# We could remove these observations, and re-run our model

outliers <- c(103,104,414)

realestate\_new <- realestate[-outliers,]

nrow(realestate\_new)-nrow(realestate)

#removed the 3rows

mod3\_terms\_1<-lm(log(price)~lotsize+prefarea\*recroom, data=realestate\_new)

summary(mod3\_terms\_1)

# Did our model improve?

# We want to know to find the extent to which our models have improved, after "cleaning"

summary(mod3\_terms\_1)$adj.r.squared-summary(mod3\_terms)$adj.r.squared

#removing 3 outliers from 500 cases improved our model by 2%

#here cook's distance model didnt help so here we focusing on residuals mode3\_terms\_n1 was done by removing 93, 365 based oncook's distance

# Check for multicollinearity

sqrt(vif(mod3\_terms\_1))>2 # car package is needed fr vif. Looks fine. If TRUE (vif > 4), consider interaction effect. If marginal, drop.

# But some multicollinearity is expected when interactions are present. If no interactions in model, address variables and re-run model.

#dont worry abt inetraction term other iv shud be false

#################

# From the bank data set

# Build a \*good\* model for the balance variable

# Focus on positive balance that is less than $10k

#################

## Logistic Regression

## Load libraries

#=

library(car)

library(Deducer)

library(tidyverse)

library(rms)

# We are modeling the odds of (y)

# a client's acceptance of a product promotion

## Some preliminaries

#=

# With logistic regression (logit) we model the \*log odds\* of an outcome=1

# As in: got hired; quit job; purchased product; visited website; admitted to hospital; credit card application approved... you've got it!

# We model the log odds of an event from a binomial distribution, where the probability of 1 (TRUE/HIGH/SUCCESS...) is found by a transformation of a linear model of the IVs

# As opposed to OLS, there is no assumption of nomral distribution. These models are called Generalized Linear models (glm).

# GLM: (1) probability distribution of an event; (2) a linear model; (3) a function that "links" the model to the parameter of the predicted distribution

# Logit: (1) probability (0-1); (2) model with numeric and factor IVs; (3) links to a value between -inf to inf

# Let's plot the logit function (for the log odds of an event)

curve((1/(1+exp(-x))),-10,10, main="Logit Function",

col ="steelblue", lwd=3)

## Step 1. Make sure the data are clean

#=

# let's assume that I want to predict y

# based on: age, balance, education, marital status, loan, default)

# inspect the data

head(bank)

tail(bank)

bank %>% sample\_n(20) %>% View()

# or using R base: View(bank[sample(nrow(bank),10),])

## Step 2: Examine at a univariate level (transform if needed)

str(bank)

# job will be hard to model, and month is a bit on an odd variable

# Outcome variable

tab <- table(bank$y)

barplot(tab, col=c("orange", "steelblue"), main="Accepted Offer")

## Step 3: Examine bivariate relations

# y and education

t\_educ\_y<-table(bank$education, bank$y)

p\_educ\_y <- prop.table(t\_educ\_y)

addmargins(p\_educ\_y,c(1,2))

# how about a financial variables, such as default

t\_default\_y<-table(bank$default, bank$y)

p\_default\_y <- prop.table(t\_default\_y)

addmargins(p\_default\_y, c(1,2))

# better yet

xt\_def\_y <- xtabs(~y+default, data=bank)

xtp\_def\_y <- round(prop.table(xt\_def\_y), 3)

addmargins(xtp\_def\_y, c(1,2))

summary(xt\_def\_y)

# We also want to explore a factor DV with a numeric IV

plot(density(bank$balance))

boxplot(bank$balance~bank$y,

main="Relationship between balance level and age",

col=rainbow(2)) # both are skewed

# outlier = 75% + 1.5IQR

outlier <- quantile(bank$balance, probs = c(0.75))+1.5\*IQR(bank$balance) #over 73

# how many do we have

balance\_outliers <- which(bank$balance>outlier)

balance\_outliers

length(balance\_outliers)/nrow(bank)

bank1 <- bank[-balance\_outliers,]

plot(density(bank1$balance)) #

boxplot(bank1$balance~bank1$y,

main="Relationship between balance level and age",

col=rainbow(2)) # better

# or we could have simply used a transformation

car::powerTransform(bank$balance+abs(bank$balance)+1) # pretty close to log

bank$scaled\_balance <- (bank$balance+abs(min(bank$balance))+1)^0.17

# Still not ideal

#################

# Examine the quit data

# And consider univariate and bivariate relationships

# With ynquit

#################

## Step 4. Modeling

#=

# Background on logistic regression

bank\_mod1 <- glm(y~marital+loan+default, data=bank, family=binomial)

summary(bank\_mod1)

# Interpretation

# Look at the p-values (and asterisks) to decide

# which variables are significantly related to the DV

# For each significant variable, interpret the "meaning."

# This stage is more difficult compared to OLS because the coefficients are log odds

# For example, having taken a loan bank

# reduces the log odds

# but log odds are hard to understand

# a. It is easier to convert log odds to odds ratios

# using the exponent function

exp(coef(bank\_mod1))

# For example, taking a loan multiplies the odds of y by about .64,

# compared to those who did not take a loan,

# holding the other IVs at their constant

# b. It is often more useful to also get confidence intervals

# for predictions

exp(confint(bank\_mod1, level=.99)) # default is .95

# So, for those who took a loan, the odds multiply between .3 and .7

# compare to those who did not take a loan.

#################

# Select two IVs in the quit data

# Interpret the odds for quitting for a model with both IVs

# Interpret the odds with confidence intervals

/// chapter 11

#=========================================================

# IDS 462, Session 11 - Logit

#=========================================================

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission from the instructor.

#====================================

rm(list=ls())

## Load libraries

#=

library(car)

library(Deducer)

library(tidyverse)

library(rms)

## Load data

# Same data as last time, Session 10

## Logit Regression

#==================

# We are modeling the odds of (y), which is a binary outcome variable

# In this case, a client's acceptance of a product promotion

## Some preliminaries

#=

# With logistic regression (logit) we model the \*log odds\* of an outcome=1

# As in: got hired; quit job; purchased product; visited website; admitted to hospital; credit card application approved... you've got it!

# We model the log odds of an event from a binomial distribution, where the probability of 1 (TRUE/HIGH/SUCCESS...) is found by a transformation of a linear model of the IVs

# How it works: GLM

# (1) probability distribution of an event

# (2) a linear model

# (3) a function that "links" the model to the parameter of the predicted distribution

# Logit: (1) probability (0-1); (2) model with numeric and factor IVs; (3) links to a value between -inf to inf

# Let's plot the logit function (for the log odds of an event)

curve((1/(1+exp(-x))),-10,10, main="Logit Function",

col ="steelblue", lwd=3)

# Assumptions:

# Like OLS

# - Independence of errors (no repeated measures, for example)

# - Predictors should not be highly correlated with each other (multicollinearity).

# Unlike OLS

# There is no assumption of nomral distribution.

# Assumes linear relationship between continuous IV and the \*logit of DV\*.

## Step 1. Examine data, clean if needed

#=

# In the bank data our DV is y

# And the IVs are: age, balance, education, marital status, loan, default)

# inspect the data

head(bank)

tail(bank)

bank %>% sample\_n(20) %>% View()

# or using R base: View(bank[sample(nrow(bank),10),])

#y column is whether they ccepted or not

## Step 2: Examine at a univariate level (transform if needed)

#=

# Outcome variable

tab <- table(bank$y)

#table(bank$y) %>%prop.table %>% round(2)

barplot(tab, col=c("orange", "steelblue"), main="Accepted Offer")

# Predictor variables

# this step is to select relevant IVs and prepare them as needed

str(bank)

# job will be hard to used in a model, and month is a bit on an odd variable

#job 12 level so drop cuz diff in regrn model to work with them #if have time change it into blue or white collared

# Also pay attention to highly skewed variables.

# While there is no assumption of linearity, there is an assumption about linearity with logit of the DV.

# It is best to transform, or remove extrme outliers (high leverage observations).

## Step 3: Examine bivariate relations with relevant IVs and DV

#=

# Default might be relevant

table(bank$default) # very few default cases

# still, is there a relationship?

xt\_def\_y <- xtabs(~y+default, data=bank) #xtabs fr header

summary(xt\_def\_y) # chisq.test; nope

# Education?

xt\_educ\_y <- xtabs(~y+education, data=bank)

summary(xt\_educ\_y) # chisq.test; better!

# We also want to explore a balance (numeric)

plot(density(bank$balance))

boxplot(bank$balance~bank$y,

main="Relationship between balance and offer acceptance",

col=rainbow(2)) # both are skewed #using comparitive boxplot diff in balance of thse who accepted n not

# outlier = 75% + 1.5IQR

outlier <- quantile(bank$balance, probs = c(0.75))+1.5\*IQR(bank$balance) #over 73

# how many do we have

balance\_outliers <- which(bank$balance>outlier)

balance\_outliers

length(balance\_outliers)/nrow(bank)

bank1 <- bank[-balance\_outliers,]

plot(density(bank1$balance)) #

boxplot(bank1$balance~bank1$y,

main="Relationship between balance and offer acceptance",

col=rainbow(2)) # better #wo outliers #from the boxplot we see tat ppl who accept offer have higher balance

#################

# Examine the quit data

# And consider univariate and bivariate relationships

# With ynquit

#################

## Step 4. Modeling

#=

bank\_mod1 <- glm(y~education+balance, data=bank1, family=binomial) #binomial approximates logit

#glm(y~education+balance, data=bank1, family=binomial) %>% summary #if u dont want to save model n just view

summary(bank\_mod1) #fisher iterations #it tries to fit #how many tmimes # smtimes cant fit

# Interpretation

# Look at the p-values (and asterisks) to decide

# which variables are significantly related to the DV

# For each significant variable, interpret the "meaning."

# This stage is more difficult compared to OLS because the coefficients are log odds

# For example, having taken a loan bank

# reduces the log odds

# but log odds are hard to understand #co-effs r log odds

# a. It is easier to convert log odds to odds ratios

# using the exponent function

exp(coef(bank\_mod1)) #log odds cant interpret so taking exp #.578 increase #very dollar in balance increases the odds of accepting the offer by .000 on avg by holding other var constnt. multiplies the odds by 1.00

# For example, tertiary education multiplies the odds of y by about .58,

# compared to people with primary education,

# holding the other IVs at their constant

# b. It is often more useful to also get confidence intervals

# for predictions

exp(confint(bank\_mod1, level=.99)) # default is .95 #here doing 99% conf level#edu multip by .4199 to twice

# So, for those who took a loan, the odds multiply between .3 and .7

# compare to those who did not take a loan.

#################

# Select two IVs in the quit data

# Interpret the odds for quitting for a model with both IVs

# Interpret the odds with confidence intervals

#################

#################

# Select two IVs in the quit data

# Interpret the odds

# Interpret the odds with confidence intervals

#################

# Add more variables

bank\_mod2 <- glm(y~balance+education+age+marital+loan+poutcome,

data=bank1, family=binomial)

summary(bank\_mod2)

#if poutcome success increases the odds of accepting loan

#being married reduces the odds of accepting the loan

# Let's check if the second model is significantly better:

# First, look at AIC

# The lower AIC the better fit. More on this later.

#second model aic dropped by 200 so its betr go with it

#read 382 ch-8

# Second, test the difference

anova(bank\_mod1, bank\_mod2, test="Chisq")

# only if we the first mode uses variables in the second

# answer: yes

# Now re-run the previous procedure

exp(coef(bank\_mod2))

# Tertiary education is still significant (p<0.01),

# has changed a little, dropped to .47 from .58 in the first model.

#poutcome success has big co-eff so sig #odds multipled by 11

# balance is also significant (p<0.01)

# an increase in one unit of balance (assume 1 $USD),

# increases the odds of y by 0.02% on average.

# so, $1000 years increases the odds

# by: (1.00022185^1000) = 1.25, on average.

# Depdning on number of cases in the data, I prefer IVs with p-value<0.01

bank\_mod3 <- glm(y~balance+marital+loan+poutcome, data=bank1, family=binomial)

summary(bank\_mod3)

# Test the difference

anova(bank\_mod2, bank\_mod3, test="Chisq")

#this model less variables so betr to go with it

# Predict outcome #we r going to get odds so ll be betn 0 n 1

set.seed(123)

samp <- sample(1:nrow(bank), 100)

pred <- predict(bank\_mod3,

bank[samp, c("balance", "marital", "loan","poutcome")],

type="response") %>% round(1) #predict fn dont forget type response #instead of eyeballing select greater than .5

ylikely <- which(pred>0.5) %>% names %>% as.numeric # more likely to accept the offer

bank[ylikely,c("balance", "marital", "loan", "poutcome", "y")] %>% View

#acceptance is low

#our model has issues #only captured 2

#################

# Select others IVs, including the two selected earlier

# from the quit data

# Is there a difference between the reduced, to the fuller model?

# Have any significance levels changed (e.g., from significant to not significant)?

# Interpret the odds with confidence intervals

# Compare the with the reduced model

# Predict the probability of quitting in the first 5 cases

#################

# Interactions?

# very much like OLS

bank\_mod4 <- glm(y~marital+loan+balance\*poutcome, data=bank, family=binomial)

summary(bank\_mod4) # This interaction term is not quite statistically significant. We will omit it.

#################

# Find a "good" model to predict quitting.

# Check for possible interactions.

# Interpret the results.

#################

/// chapter 12

#=========================================================================

# IDS 462, Session 12

#========================================================================

# Copyright Zack Kertcher, PhD, 2018. All rights reserved.

# Do not distribute or use outside this class without explicit permission

# from the instructor.

#========================================================================

## LOAD DATA AND LIBRARIES

#=

library(car)

library(corrplot)

library(rms)# for pseudo-R-Squared

library(psych)

library(tidyverse)

load("session12data.Rdata")

## Common issues and tips

## Logit (cont.)

## Exploratory factor analysis

## Common issues and tips

## Common issues and tips

#=

# The problem: identify and remove outliers

# Outlier detection utility function

outliers <- function(column) {

lowerq <- as.vector(quantile(column)[2]) # returns 1st quartile

upperq <- as.vector(quantile(column)[4]) # returns 1st quartile

iqr <- upperq-lowerq

# Moderate outliers

mod.outliers.upper <- (iqr \* 1.5) + upperq

mod.outliers.lower <- lowerq - (iqr \* 1.5)

mod.outliers <- which(column > mod.outliers.upper |

column < mod.outliers.lower)

print(paste(mod.outliers))

#Extreme outliers

extreme.outliers.upper <- (iqr \* 3) + upperq

extreme.outliers.lower <- lowerq - (iqr \* 3)

extreme.outliers<-which(column > extreme.outliers.upper

| column < extreme.outliers.lower)

print(paste(extreme.outliers))

}

# Either compute +/-1.5\*IQR, or use the utility function I provided (outliers)

# If you use the outlier function, you can find the outlier rows, but you can't directly use

# to remove these rows. So,

# How many outliers are in the data?

num\_outliers <- outliers(bank$balance) %>% length()

num\_outliers/nrow(bank) # around 5%, so let's drop them

outliersvec <- outliers(bank$balance) %>% as.numeric # generate a numeric vector of outliers#outliers shows with quotes cant subset with characters so ocnvert as number

bank1 <- bank[-outliersvec,] # bank1 is without outliers in balance

plot(density(bank$balance)) # bad

plot(density(bank1$balance)) # not too bad

# The problem: can you start with a regression model then perform EDA?

# Not recommnded. Remember the debt variable from the bank model? What about poutcome?

bank\_mod3 <- glm(y~balance+marital+loan+poutcome, data=bank1, family=binomial)

summary(bank\_mod3)

# Wow! Look at the z-value for poutcome sucess. Unkown is significant as well.#z value gves magnitude more value

#But wait, how many cases are there?

table(bank1$poutcome) #gud fr prediction but not fr analysis

# So, how meaningful and useful is this variable?

# Useful for prediction, but practial application is likely limited.

# The problem: how to recode a variable with many levels? #ifelse, gsub, with subsetting

# Remember that you can recode a variable with an ifelse function. For example,

bank1$marital\_binary <- ifelse(bank1$marital=="married", "married", "not married")

table(bank1$marital\_binary)

# But if we have more levels it becomes complicated (a lot of nested ifelse statements), or clunky assignments.

# The car package has what I think is the best option, with the recode function. Here's how it works:

bank1$season <- car::recode(bank1$month, "c('sep', 'oct', 'nov')='fall';

c('dec', 'jan', 'feb')='winter';

c('mar', 'apr', 'may')='spring';

c('jun', 'jul', 'aug')='summer' ")

table(bank1$season) # a lot more contact in spring and summer

# The problem: Can't transform a variable. Getting an error message. For example,

log(bank$balance)

sqrt(bank$balance)

# The issue are the negatives and zeros. How many do we have?

bank$balance[bank$balance<=0] %>% length/nrow(bank) # Quite a few, and it makes sense, so #16% ve begative zero balnce its possible# i can remove

bank$balance\_std <- bank$balance+abs(min(bank$balance))+1 #min bank bal was -3313 tajing 3313 addig it still ll giv 0 so add +1 n add that to allvalues# simple stadardization of balance to beomce positive

head(bank$balance); head(bank$balance\_std)

# Now try the transformation

log(bank$balance\_std) # yes!

## Logistic regression (cont.)

# Stopped at exploring interactions with logit

#################

# Find a "good" model to predict quitting.

# Your model might include interactions

# Interpret the results.

#################

quitmod <- glm(ynquit~jobsatisfaction\*age, data=quit, family="binomial") #job satisfcation likert scake n not cont var#: adds only interaction \* adds main effect n inetraction

summary(quitmod)

#use exp to interpret results

plot(effect(term="jobsatisfaction:age", mod=quitmod, default.levels=20), multiline=T) # intersting #plot only interaction #job satisfaction has no effect fr age 60#effect of JS decreases the older u get on d odds of quitting

## 4. Diagnostics

#=

# Logit does not have exactly the same diagnostics tools as OLS.

# But here is what we can do:

# a. Examine for possible multicollinearity

vif(quitmod)

sqrt(vif(quitmod))>2 # There is multicollinearity in this model!

# However, this is not data-related multicollinearity. This type of structured-multicollinearity, is less concerning. #we intrducd the interaction acknowledge n move on# if vif was der fr age n JS then need to consider#dont panic if cuz of inetraction term

# Still, know that coefficients are not going to be precise.

# b. Check for overdispersion

# The logit model assumes a binomial distribution.

# Sometimes the variance of the DV is larger than

# specified for a binomial distribution.

# The is called \*overdispersion\* (we don't typically find underdispersion)

# We identify overdispersion by dividing the residual deviance

# with the residual degrees of freedom in the model.

# If the result is \*much higher than 1\*, there is overdispersion. #run chi sq n compare nt sure

# Returning to the bank model

bank\_mod3 <- glm(y~balance+marital+loan+poutcome, data=bank1, family=binomial)

deviance(bank\_mod3)/df.residual(bank\_mod3) # Not bad. #deviance /residul the value is less than 1 so no overdispersion #we can test fr this using chi sq too

# But how do we know for sure?

# We compare the model to a model that assumes overdispersion,

# and see if there is a difference

bank\_mod3\_od <- glm(y~balance+marital+loan+poutcome,

data=bank1, family=quasibinomial) # note the quasibinomial distribution

pchisq(summary(bank\_mod3\_od)$dispersion \* bank\_mod3$df.residual,

bank\_mod3$df.residual, lower=F)

# p-value is much higher than 0.05 #if value is less than .05 then tehre is overdispersion #anything above std p=value no overdispersion

# There is no overdispersion.

# If there is overdispersion, fit a quasibinomial distribution intead of binomial.

#################

# Using the quit data, build a model

# Check for multicollinearity

# Check for overdispersion

# What did you find?

#################

rm(list=ls())

# Model fit

#=

# Instead of Adjusted-R-Squared we get AIC.

# AIC: Index of fit that penalizes the number of parameters.

# It is computed as:

# -2\*(max log-likelihood)+2\*(number of parameters)

# So, the smaller the AIC, the better.

# We can compare models using AIC

# but we don't get a good sense of our model's performance

# Instead, we use pseudo-R-Squared measures.

# There are a few of them. Here's a good one from the rms package:

mod\_fit\_3<- lrm(y ~ balance+marital+loan+poutcome,

data = bank1)

mod\_fit\_3$stats["R2"]

# What if we take out one IV, like poutcome?

mod\_fit\_3.1 <- lrm(y~balance+marital+loan,

data = bank1)

mod\_fit\_3.1$stats["R2"] #pseudo r sq dropped from 13 to 3% #but cant tel a story

# poutcome, despite having few values in sucess, is an important IV!

# We could have also figure this out but simply looking at the z-value

summary(bank\_mod3)

#################

# Build three models and find the one with the best fit.

# What are the best AIC?

# What is the best pseudo-R-Squared?

#################

## BACKGROUND ON (EXPLORATORY) FACTOR ANALYSIS #common #other type is conformity factor analsis rare sorta lik tree

#=

# Aims to find underlying (latent) factors #obj is to find hidden structure in data

# that account for observed relationships

# among numeric variables

# Used to reduce variables, and build a scale/index (e.g., social status scale, work personality index)

# The process is:

#1. Examine the data

#2. Scale the data (if needed)

#3. Consider number of factors

#4. Extract factors

#5. (optional) use the factors discovered in analysis and modeling

#1. Examine the data

#=

View(brands)

str(brands)

summary(brands)

#2. Scale the data (if needed) #change value but not distribution #not ransformed

#=

# Data may have different scales/values. We want to center them (xi-mean(x)).

# Better yet, standardize them. This is how we standardize a variable:

#(xi-mean(x))/sd(x) #most common

# Or use R's scale function #all values seems to be likert and looks lik ll be simiar n can be grouped # diff scale lik 1 to mil n others in diff scale

# This not needed when the variables have the same scale, as in the brands data.

# But, this is the general procedure:

brands\_s <- data.frame(scale(brands[,-1])) # omit the brand variable (factor)

describe(brands\_s) # as expected #with just describe mght get diff results clashes with other package so change code as pstch::describe

corrplot(cor(brands\_s), method="circle", addCoef.col="grey", type="upper") #find var which correlate with one another but not one which is super correlated(lik both same) n dont want with no correlation we want .5 .7 etc

# We are looking for variables with relatively high correlation with one or a few others

# brands\_s$brand <- brands$brand # add the factor variable

#3. Consider number of factors #tell how many factors to extract

#=

# We'll use the fa.parallel function from the psych package

# To use the current data against simulated data

# The procedure in FA is to "rotate" the data to maximize variation for each factor.

# Default rotation is varimax. Oblique rotations allow correlation among factors, whereas orthogonal rotations do not.

fa.parallel(brands\_s, fa="both", n.iter=100, show.legend=F) #its gonna rotate columns together to c if there is correlation w usualy do varimax. we here trying to max the dist betn them #diff rotation alogorithms exst

#our obj here is least no.of factors which gives most variance

#eigen value to look at distance

# Results suggest the optimal number of factors based on the data

# Look for an "elbow", especially around eigenvalue = 1. #it ll giv a reco itslf

## Perform FA

#=

# Many methods to extract factors (unlike PCA)

brandsfa1 <- factanal(brands\_s, 3) #factor analysis #rem it has to be scaled data # here we ve same scale but still we use this

brandsfa1$loadings

# loading values are essentially correlation coefficient of variable with factor

# The higher the loading, the higher the "correlation" with the factor

# Look for loading of .5 or higher (conservatively, .7 or higher)

# When developing a scale, you might include the negative with the positive loadings

# But typically use either all the positive, or negative that have a high loading

# So,

# Factor1 (bargain, value) - value proposition etc

# Factor2 (perform, leader, serious) - quality proposition #notice the fun variable

# Factor (latest, trendy) -hip factor

#if i get -.8 #include either all + or all -

#create separate with value proposition, , etc n also other left out var n then the story i can tel ll be much easier

#loadings which is above .7 can be conidered factor those tat didnt come are indepfrom the rest #prop var --.206 factor 1 explains 20% of variance

##############################################

# Perform factor analysis on the decathlon data.

# How many factors were discovered?

# Which events loaded the best on these factors?

# Can you come up with names for these underlying factors?

##############################################

## ADDITIONAL RESOURCES

#=

# Books:

#-

# Multiple Factor Analysis by Example Using R. 2014. Chapman and Hall/CRC

# Exploratory Multivariate Analysis by Example, 2nd. Edition. 2017. CRC Press.

# The above procedure is for exploratory factor analysis. If you want confirmatory analysis, use structural equation modeling (from the sem package)

# A good tutorial of structural equation modeling using R is found at: http://socserv.socsci.mcmaster.ca/jfox/Misc/sem/SEM-paper.pdf

/// chapter 13

emp$department <- as.factor (emp$department)

emp$travel <- as.factor (emp$travel)

emp$quit <- as.factor (emp$quit)

str(emp)

table(emp$department)

emp$department <- as.factor (emp$department)

emp\_factors <- c("department", "gender", "marital", "overtime", "quit", "travel")

emp[emp\_factors] <- lapply(emp[emp\_factors], factor)

emp1 <- emp[,!emp\_factors]

corrplot()

corrplot(cormat)

describe(emp1)

fa.parallel(emp, fa="both", n.iter=100, show.legend=F)

#factor analysis - 1st step scale it. else it wont rotate data properly which has high variance

#we can scale only nueric data

#do describe just to make sure everything is fine

#fa.parallel to see no.of factors . it suggests 3 factors. now we need to extract the factors n c d loadings

#beyond .4 shud be correated with factor exam time pressue need not name #factor1 experience #factor2 role index #seniority, monthly - they r correlated so dono if we want to extract

#extracting factors n building index

emp\_nums <- emp[sapply(emp,is.numeric)]

cormat <- cor(emp\_nums)

#corrplot::

#first add all other than the factors too many der cuz for factor analysis #addd index n after add leftover. name as factor1 too in time pressure

emp\_mod1 <- glm(quit~age+travel+marital+job\_satisfaction+miles\_from\_home+years\_since\_promotion+pct\_salary\_increase+seniority+overtime+monthly, data=emp, family=binomial)

summary(emp\_mod1)

#overtimeYes \*\*\*, job satisfcation, overtimeyes, travel none, marital single

factor3 <- emp[,""]

emp\_mod2 <- glm(quit~age+travel+marital+job\_satisfaction+miles\_from\_home+years\_since\_promotion+overtime+monthly, data=emp, family=binomial)

summary(emp\_mod2)

emp\_mod5 <- glm(quit~travel+marital+job\_satisfaction+job\_involvement+past\_firms+miles\_from\_home+years\_since\_promotion+overtime, data=emp, family=binomial)

summary(emp\_mod5)

emp\_mod3 <- glm(quit~., data=emp, family=binomial)

summary(emp\_mod3)

vif(emp\_mod3)

sqrt(vif(emp\_mod3))>2

#too few levels of factors

#exlore interactions it may matter. if tme was der will explore interactions

#do exp and do the interpretations

#log odds after exp it is odds every unit of exp decreases d odd of quitting the job by less than 1% 76% less

#reduces by 40% travel

#single multiplied by 2.7

#check fr dispersion

#2 models check aic check pseudo rsq nwhch model is bter

#interpret effects most imp factor is \_\_ talk abt co-eff

#mention problems in diagnostics