Writing\_Assignment

2023-05-02

#Installing Packages:  
  
#install.packages("GGally")  
#install.packages("caret")  
#install.packages("mgcv")  
#install.packages("glmnet")  
#install.packages("randomForest")  
#install.packages("mice")  
#install.packages("rmarkdown")  
#install.packages("knitr")

#Import the libraries:  
  
library(ggplot2)  
library(GGally)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(readxl) # Load readxl package  
library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

#Data Preparation:

#Load the data:  
  
df2 <- read\_excel("/Users/hetal/Downloads/RedactedClientConstituent\_File.xlsx")  
head(df2)

## # A tibble: 6 × 31  
## CnBio\_ID `First Gift Date` `Last Gift Date` `Largest Gift Date`  
## <dbl> <dttm> <dttm> <dttm>   
## 1 200001488 NA NA NA   
## 2 200001489 NA NA NA   
## 3 200001490 NA NA NA   
## 4 200001491 NA NA NA   
## 5 200001492 NA NA NA   
## 6 200001493 NA NA NA   
## # ℹ 27 more variables: CnBio\_DateAdded <dttm>, CnBio\_DateChanged <dttm>,  
## # CnBio\_Key\_Indicator <chr>, CnBio\_Deceased <chr>, CnBio\_Title\_1 <chr>,  
## # CnBio\_Marital\_status <chr>, City <chr>, State <chr>,  
## # `Reunions attended` <dbl>, Zip <chr>, CnAdrPrf\_Type <chr>,  
## # CnCnstncy\_1\_01\_CodeLong <chr>, CnCnstncy\_1\_02\_CodeLong <chr>,  
## # CnCnstncy\_1\_03\_CodeLong <chr>, CnCnstncy\_1\_04\_CodeLong <chr>,  
## # CnSpPrBs\_RecordImportID <chr>, CnRelEdu\_1\_01\_Class\_of <dbl>, …

# Check for missing values  
print(colSums(is.na(df2)))

## CnBio\_ID First Gift Date   
## 0 27073   
## Last Gift Date Largest Gift Date   
## 27073 27073   
## CnBio\_DateAdded CnBio\_DateChanged   
## 0 0   
## CnBio\_Key\_Indicator CnBio\_Deceased   
## 0 0   
## CnBio\_Title\_1 CnBio\_Marital\_status   
## 29027 24985   
## City State   
## 1267 1374   
## Reunions attended Zip   
## 0 1372   
## CnAdrPrf\_Type CnCnstncy\_1\_01\_CodeLong   
## 0 4882   
## CnCnstncy\_1\_02\_CodeLong CnCnstncy\_1\_03\_CodeLong   
## 37625 42088   
## CnCnstncy\_1\_04\_CodeLong CnSpPrBs\_RecordImportID   
## 42280 42162   
## CnRelEdu\_1\_01\_Class\_of CnRelEdu\_1\_01\_Degree   
## 25089 20635   
## CnRelEdu\_1\_02\_Degree Alumni Board Member   
## 39900 42259   
## Married to an Alum Personal Email End   
## 41713 18106   
## Total Lifetime Giving Last 10 Years Giving   
## 0 0   
## Last 5 Years Giving (FY17-21) CnSpSpBio\_ID   
## 0 39885   
## Alumni   
## 0

#Converting the columns for easy accessibility:  
  
colnames(df2)[colnames(df2) == "Total Lifetime Giving"] <- "Total\_Lifetime\_Giving"  
colnames(df2)[colnames(df2) == "Last 5 Years Giving (FY17-21)"] <- "Last\_5\_Years\_Giving"  
colnames(df2)[colnames(df2) == "Last 10 Years Giving"] <- "Last\_10\_Years\_Giving"  
colnames(df2)[colnames(df2) == "Alumni Board Member"] <- "Alumni\_Board\_Member"  
colnames(df2)[colnames(df2) == "Last Gift Date"] <- "Last\_Gift\_Date"  
colnames(df2)[colnames(df2) == "First Gift Date"] <- "First\_Gift\_Date"  
colnames(df2)[colnames(df2) == "Largest Gift Date"] <- "Largest\_Gift\_Date"  
colnames(df2)[colnames(df2) == "Personal Email End"] <- "Personal\_Email\_End"  
colnames(df2)[colnames(df2) == "Reunions attended"] <- "Reunions\_attended"  
colnames(df2)[colnames(df2) == "Married to an Alum"] <- "Married\_to\_an\_Alum"  
colnames(df2)[colnames(df2) == "Last Gift Date"] <- "Last\_Gift\_Date"

##Shape of DataFrame:  
dim(df2)

## [1] 42287 31

#Finding the datatype of variables:  
sapply(df2, class)

## $CnBio\_ID  
## [1] "numeric"  
##   
## $First\_Gift\_Date  
## [1] "POSIXct" "POSIXt"   
##   
## $Last\_Gift\_Date  
## [1] "POSIXct" "POSIXt"   
##   
## $Largest\_Gift\_Date  
## [1] "POSIXct" "POSIXt"   
##   
## $CnBio\_DateAdded  
## [1] "POSIXct" "POSIXt"   
##   
## $CnBio\_DateChanged  
## [1] "POSIXct" "POSIXt"   
##   
## $CnBio\_Key\_Indicator  
## [1] "character"  
##   
## $CnBio\_Deceased  
## [1] "character"  
##   
## $CnBio\_Title\_1  
## [1] "character"  
##   
## $CnBio\_Marital\_status  
## [1] "character"  
##   
## $City  
## [1] "character"  
##   
## $State  
## [1] "character"  
##   
## $Reunions\_attended  
## [1] "numeric"  
##   
## $Zip  
## [1] "character"  
##   
## $CnAdrPrf\_Type  
## [1] "character"  
##   
## $CnCnstncy\_1\_01\_CodeLong  
## [1] "character"  
##   
## $CnCnstncy\_1\_02\_CodeLong  
## [1] "character"  
##   
## $CnCnstncy\_1\_03\_CodeLong  
## [1] "character"  
##   
## $CnCnstncy\_1\_04\_CodeLong  
## [1] "character"  
##   
## $CnSpPrBs\_RecordImportID  
## [1] "character"  
##   
## $CnRelEdu\_1\_01\_Class\_of  
## [1] "numeric"  
##   
## $CnRelEdu\_1\_01\_Degree  
## [1] "character"  
##   
## $CnRelEdu\_1\_02\_Degree  
## [1] "character"  
##   
## $Alumni\_Board\_Member  
## [1] "character"  
##   
## $Married\_to\_an\_Alum  
## [1] "character"  
##   
## $Personal\_Email\_End  
## [1] "character"  
##   
## $Total\_Lifetime\_Giving  
## [1] "numeric"  
##   
## $Last\_10\_Years\_Giving  
## [1] "numeric"  
##   
## $Last\_5\_Years\_Giving  
## [1] "numeric"  
##   
## $CnSpSpBio\_ID  
## [1] "numeric"  
##   
## $Alumni  
## [1] "numeric"

#Finding the summary of dataframe:  
summary(df2)

## CnBio\_ID First\_Gift\_Date   
## Min. : 4 Min. :1901-01-01 00:00:00.000   
## 1st Qu.: 173994 1st Qu.:1989-10-24 00:00:00.000   
## Median : 292194 Median :2003-02-17 00:00:00.000   
## Mean : 24034109 Mean :1998-11-12 07:39:31.447   
## 3rd Qu.: 435660 3rd Qu.:2010-10-18 00:00:00.000   
## Max. :200003047 Max. :2022-03-14 00:00:00.000   
## NA's :27073   
## Last\_Gift\_Date Largest\_Gift\_Date   
## Min. :1901-01-01 00:00:00.00 Min. :1901-01-01 00:00:00.00   
## 1st Qu.:1998-03-19 00:00:00.00 1st Qu.:1995-07-31 00:00:00.00   
## Median :2008-12-18 00:00:00.00 Median :2006-12-07 00:00:00.00   
## Mean :2006-04-01 19:16:48.49 Mean :2003-03-17 06:09:53.45   
## 3rd Qu.:2016-02-29 00:00:00.00 3rd Qu.:2013-04-30 00:00:00.00   
## Max. :2022-03-15 00:00:00.00 Max. :2022-03-14 00:00:00.00   
## NA's :27073 NA's :27073   
## CnBio\_DateAdded CnBio\_DateChanged   
## Min. :2017-07-17 00:00:00.00 Min. :2017-07-18 00:00:00.00   
## 1st Qu.:2017-07-17 00:00:00.00 1st Qu.:2021-11-27 00:00:00.00   
## Median :2017-07-17 00:00:00.00 Median :2022-02-07 00:00:00.00   
## Mean :2017-10-04 02:19:28.86 Mean :2021-08-05 14:07:51.11   
## 3rd Qu.:2017-07-17 00:00:00.00 3rd Qu.:2022-02-11 00:00:00.00   
## Max. :2022-03-17 00:00:00.00 Max. :2022-03-18 00:00:00.00   
##   
## CnBio\_Key\_Indicator CnBio\_Deceased CnBio\_Title\_1 CnBio\_Marital\_status  
## Length:42287 Length:42287 Length:42287 Length:42287   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## City State Reunions\_attended Zip   
## Length:42287 Length:42287 Min. :0.000000 Length:42287   
## Class :character Class :character 1st Qu.:0.000000 Class :character   
## Mode :character Mode :character Median :0.000000 Mode :character   
## Mean :0.007378   
## 3rd Qu.:0.000000   
## Max. :6.000000   
##   
## CnAdrPrf\_Type CnCnstncy\_1\_01\_CodeLong CnCnstncy\_1\_02\_CodeLong  
## Length:42287 Length:42287 Length:42287   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## CnCnstncy\_1\_03\_CodeLong CnCnstncy\_1\_04\_CodeLong CnSpPrBs\_RecordImportID  
## Length:42287 Length:42287 Length:42287   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## CnRelEdu\_1\_01\_Class\_of CnRelEdu\_1\_01\_Degree CnRelEdu\_1\_02\_Degree  
## Min. :1929 Length:42287 Length:42287   
## 1st Qu.:1985 Class :character Class :character   
## Median :2004 Mode :character Mode :character   
## Mean :1997   
## 3rd Qu.:2014   
## Max. :2021   
## NA's :25089   
## Alumni\_Board\_Member Married\_to\_an\_Alum Personal\_Email\_End  
## Length:42287 Length:42287 Length:42287   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## Total\_Lifetime\_Giving Last\_10\_Years\_Giving Last\_5\_Years\_Giving  
## Min. : 0 Min. : 0 Min. : 0   
## 1st Qu.: 0 1st Qu.: 0 1st Qu.: 0   
## Median : 0 Median : 0 Median : 0   
## Mean : 2318 Mean : 989 Mean : 456   
## 3rd Qu.: 35 3rd Qu.: 0 3rd Qu.: 0   
## Max. :11321975 Max. :8805800 Max. :4121000   
##   
## CnSpSpBio\_ID Alumni   
## Min. : 1512 Min. :0.0000   
## 1st Qu.: 199564 1st Qu.:0.0000   
## Median : 357472 Median :0.0000   
## Mean : 68234755 Mean :0.4386   
## 3rd Qu.:200000817 3rd Qu.:1.0000   
## Max. :200003047 Max. :1.0000   
## NA's :39885

#Getting the column names:  
names(df2)

## [1] "CnBio\_ID" "First\_Gift\_Date"   
## [3] "Last\_Gift\_Date" "Largest\_Gift\_Date"   
## [5] "CnBio\_DateAdded" "CnBio\_DateChanged"   
## [7] "CnBio\_Key\_Indicator" "CnBio\_Deceased"   
## [9] "CnBio\_Title\_1" "CnBio\_Marital\_status"   
## [11] "City" "State"   
## [13] "Reunions\_attended" "Zip"   
## [15] "CnAdrPrf\_Type" "CnCnstncy\_1\_01\_CodeLong"  
## [17] "CnCnstncy\_1\_02\_CodeLong" "CnCnstncy\_1\_03\_CodeLong"  
## [19] "CnCnstncy\_1\_04\_CodeLong" "CnSpPrBs\_RecordImportID"  
## [21] "CnRelEdu\_1\_01\_Class\_of" "CnRelEdu\_1\_01\_Degree"   
## [23] "CnRelEdu\_1\_02\_Degree" "Alumni\_Board\_Member"   
## [25] "Married\_to\_an\_Alum" "Personal\_Email\_End"   
## [27] "Total\_Lifetime\_Giving" "Last\_10\_Years\_Giving"   
## [29] "Last\_5\_Years\_Giving" "CnSpSpBio\_ID"   
## [31] "Alumni"

#Feature Engineering:

#After analyzing the data, group all the similar features in same category using encoding method:  
  
# create a new column with grouped values for CnCnstncy\_1\_01\_CodeLong:   
df2$grouped\_CnCnstncy\_1\_01\_CodeLong <- ifelse(df2$CnCnstncy\_1\_01\_CodeLong %in% c("Board", "Previous Board"), "Board",   
 ifelse(df2$CnCnstncy\_1\_01\_CodeLong %in% c("Current Fac/Staff", "Former Fac/Staff"), "Fac/Staff",   
 ifelse(df2$CnCnstncy\_1\_01\_CodeLong %in% c("Student", "Parent", "Education Certificate"), "Education/Family",   
 ifelse(df2$CnCnstncy\_1\_01\_CodeLong %in% c("Friend", "Friends / Memorial"), "Friendship/Memorial",   
 ifelse(df2$CnCnstncy\_1\_01\_CodeLong %in% c("Prospective Benefactor", "Organization", "Alumni", "Business", "Foundation", "Trust / Business", "Dominican Colleges and Universities", "WAICU", "Government", "Religious Org", "Unknown - Historical"), "OtherDonors", "NA")))))  
  
# encode the labels using factor()  
# encode the labels using factor() and replace NA and 0 values with 0  
  
df2$CnCnstncy\_1\_01\_CodeLong\_Encoded <- ifelse(is.na(df2$grouped\_CnCnstncy\_1\_01\_CodeLong) | df2$grouped\_CnCnstncy\_1\_01\_CodeLong == "NA" | df2$grouped\_CnCnstncy\_1\_01\_CodeLong == 0, 0,  
 as.integer(factor(df2$grouped\_CnCnstncy\_1\_01\_CodeLong,  
 levels = c("Education/Family", "Friendship/Memorial", "Fac/Staff", "Board", "OtherDonors"),  
 labels = c(1, 2, 3, 4, 5))))

#After analyzing the data, group all the similar features in same category using encoding method:  
  
# create a new column with grouped values for CnCnstncy\_1\_02\_CodeLong:  
df2$grouped\_CnCnstncy\_1\_02\_CodeLong <- ifelse(df2$CnCnstncy\_1\_02\_CodeLong %in% c("Board", "Previous Board", "Board of Visitors Advisory Group"), "Board",   
 ifelse(df2$CnCnstncy\_1\_02\_CodeLong %in% c("Current Fac/Staff", "Former Fac/Staff"), "Fac/Staff",   
 ifelse(df2$CnCnstncy\_1\_02\_CodeLong %in% c("Student", "Parent", "Education Certificate"), "Education/Family",   
 ifelse(df2$CnCnstncy\_1\_02\_CodeLong %in% c("Friend", "Friends / Memorial", "Friends / Athletics", "Friends / Agency"), "Friendship/Memorial",   
 ifelse(df2$CnCnstncy\_1\_02\_CodeLong %in% c("Prospective Benefactor", "Organization", "Alumni", "Business", "Foundation", "Trust", "Cutting Edge Alumni", "Religious Org", "Unknown - Historical"), "OtherDonors", "NA")))))  
  
# encode the labels using factor()  
# encode the labels using factor() and replace NA and 0 values with 0  
df2$CnCnstncy\_1\_02\_CodeLong\_Encoded <- ifelse(is.na(df2$grouped\_CnCnstncy\_1\_02\_CodeLong) | df2$grouped\_CnCnstncy\_1\_02\_CodeLong == "NA" | df2$grouped\_CnCnstncy\_1\_02\_CodeLong == 0, 0,  
 as.integer(factor(df2$grouped\_CnCnstncy\_1\_02\_CodeLong,  
 levels = c("Education/Family", "Friendship/Memorial", "Fac/Staff", "Board", "OtherDonors"),  
 labels = c(1, 2, 3, 4, 5))))

#After analyzing the data, group all the similar features in same category using encoding method:  
  
# create a new column with grouped values for CnCnstncy\_1\_03\_CodeLong:  
df2$grouped\_CnCnstncy\_1\_03\_CodeLong <- ifelse(df2$CnCnstncy\_1\_03\_CodeLong %in% c("Board", "Previous Board"), "Board",   
 ifelse(df2$CnCnstncy\_1\_03\_CodeLong %in% c("Current Fac/Staff", "Former Fac/Staff"), "Fac/Staff",   
 ifelse(df2$CnCnstncy\_1\_03\_CodeLong %in% c("Student", "Parent", "Education Certificate"), "Education/Family",   
 ifelse(df2$CnCnstncy\_1\_03\_CodeLong %in% c("Friend", "Friends / Memorial"), "Friendship/Memorial",   
 ifelse(df2$CnCnstncy\_1\_03\_CodeLong %in% c("Prospective Benefactor", "Organization", "Alumni"), "OtherDonors", "NA")))))  
  
# encode the labels using factor() and replace NA and 0 values with 0  
df2$CnCnstncy\_1\_03\_CodeLong\_Encoded <- ifelse(is.na(df2$grouped\_CnCnstncy\_1\_03\_CodeLong) | df2$grouped\_CnCnstncy\_1\_03\_CodeLong == "NA" | df2$grouped\_CnCnstncy\_1\_03\_CodeLong == 0, 0,  
 as.integer(factor(df2$grouped\_CnCnstncy\_1\_03\_CodeLong,  
 levels = c("Education/Family", "Friendship/Memorial", "Fac/Staff", "Board", "OtherDonors"),  
 labels = c(1, 2, 3, 4, 5))))

#After analyzing the data, group all the similar features in same category using encoding method:  
  
# Create a mapping between the original labels and new encoded labels for CnCnstncy\_1\_04\_CodeLong:  
label\_mapping <- c("NA" = 0, "Former Fac/Staff" = 1, "Previous Board" = 2, "Education Certificate" = 3)  
  
# Replace missing values with "NA" label  
df2$CnCnstncy\_1\_04\_CodeLong[is.na(df2$CnCnstncy\_1\_04\_CodeLong)] <- "NA"  
  
# Convert the column to a factor with the new encoding  
df2$CnCnstncy\_1\_04\_CodeLong\_Encoded <- factor(df2$CnCnstncy\_1\_04\_CodeLong, labels = label\_mapping)

#After analyzing the data, group all the similar features in same category using encoding method:  
  
# create a new column with grouped values for CnBio\_Marital\_status:  
df2$grouped\_CnBio\_Marital\_status <- ifelse(df2$CnBio\_Marital\_status %in% c("Married", "Partner", "Cohabitation", "Engaged"), "Committed",   
 ifelse(df2$CnBio\_Marital\_status %in% c("Divorced", "Single", "Widowed", "Separated"), "Single",   
 ifelse(df2$CnBio\_Marital\_status %in% c("Religious"), "Worshipper",   
 ifelse(is.na(df2$CnBio\_Marital\_status) | df2$CnBio\_Marital\_status == "Unknown", "Unknowns", "NA"))))  
  
# encode the labels using factor() and replace NA and 0 values with 0  
df2$CnBio\_Marital\_status\_Encoded <- ifelse(is.na(df2$grouped\_CnBio\_Marital\_status) | df2$grouped\_CnBio\_Marital\_status == "NA" | df2$grouped\_CnBio\_Marital\_status == "Unknowns", 0,  
 as.integer(factor(df2$grouped\_CnBio\_Marital\_status,  
 levels = c("Committed", "Single", "Worshipper", "Unknowns"),  
 labels = c(1, 2, 3, 4))))

##Label Encoding to States Column:  
  
## First find top 5 and least 5 from total\_lifetime, 5years and 10 years and seee consistent or not  
# Convert the category column to a factor  
df2$State <- factor(df2$State)  
  
# Convert the factor levels to integer values using as.integer()  
df2$State\_encoded <- as.integer(df2$State)  
  
# Replace null and NA values with 0 using ifelse() and is.na()  
df2$State\_encoded <- ifelse(is.na(df2$State\_encoded), 0, df2$State\_encoded)  
  
# View the resulting data frame  
df2

## # A tibble: 42,287 × 41  
## CnBio\_ID First\_Gift\_Date Last\_Gift\_Date Largest\_Gift\_Date  
## <dbl> <dttm> <dttm> <dttm>   
## 1 200001488 NA NA NA   
## 2 200001489 NA NA NA   
## 3 200001490 NA NA NA   
## 4 200001491 NA NA NA   
## 5 200001492 NA NA NA   
## 6 200001493 NA NA NA   
## 7 200001494 NA NA NA   
## 8 200001495 NA NA NA   
## 9 200001496 NA NA NA   
## 10 200001497 NA NA NA   
## # ℹ 42,277 more rows  
## # ℹ 37 more variables: CnBio\_DateAdded <dttm>, CnBio\_DateChanged <dttm>,  
## # CnBio\_Key\_Indicator <chr>, CnBio\_Deceased <chr>, CnBio\_Title\_1 <chr>,  
## # CnBio\_Marital\_status <chr>, City <chr>, State <fct>,  
## # Reunions\_attended <dbl>, Zip <chr>, CnAdrPrf\_Type <chr>,  
## # CnCnstncy\_1\_01\_CodeLong <chr>, CnCnstncy\_1\_02\_CodeLong <chr>,  
## # CnCnstncy\_1\_03\_CodeLong <chr>, CnCnstncy\_1\_04\_CodeLong <chr>, …

#Replacing for all the values of alumni board member to 1 and 0 for not there:  
df2$Alumni\_Board\_Member <- ifelse(!is.na(df2$Alumni\_Board\_Member) & df2$Alumni\_Board\_Member != "", 1, 0)

# Replace "O" with 0 and "l" with 1 in CnBio\_Key\_Deceased column:  
df2$CnBio\_Deceased <- ifelse(df2$CnBio\_Deceased == "No", 0, 1)  
  
# print updated data frame  
print(df2)

## # A tibble: 42,287 × 41  
## CnBio\_ID First\_Gift\_Date Last\_Gift\_Date Largest\_Gift\_Date  
## <dbl> <dttm> <dttm> <dttm>   
## 1 200001488 NA NA NA   
## 2 200001489 NA NA NA   
## 3 200001490 NA NA NA   
## 4 200001491 NA NA NA   
## 5 200001492 NA NA NA   
## 6 200001493 NA NA NA   
## 7 200001494 NA NA NA   
## 8 200001495 NA NA NA   
## 9 200001496 NA NA NA   
## 10 200001497 NA NA NA   
## # ℹ 42,277 more rows  
## # ℹ 37 more variables: CnBio\_DateAdded <dttm>, CnBio\_DateChanged <dttm>,  
## # CnBio\_Key\_Indicator <chr>, CnBio\_Deceased <dbl>, CnBio\_Title\_1 <chr>,  
## # CnBio\_Marital\_status <chr>, City <chr>, State <fct>,  
## # Reunions\_attended <dbl>, Zip <chr>, CnAdrPrf\_Type <chr>,  
## # CnCnstncy\_1\_01\_CodeLong <chr>, CnCnstncy\_1\_02\_CodeLong <chr>,  
## # CnCnstncy\_1\_03\_CodeLong <chr>, CnCnstncy\_1\_04\_CodeLong <chr>, …

# replace "O" with 0 and "l" with 1 in CnBio\_Key\_Indicator column  
df2$CnBio\_Key\_Indicator <- ifelse(df2$CnBio\_Key\_Indicator == "O", 0, 1)

## Any Degrees Present in Education:  
df2$Any\_Degree\_Present <- ifelse((is.na(df2$CnRelEdu\_1\_01\_Degree) & is.na(df2$CnRelEdu\_1\_02\_Degree)) |   
 (df2$CnRelEdu\_1\_01\_Degree %in% c("None", "Unknown")) |   
 (df2$CnRelEdu\_1\_02\_Degree %in% c("None", "Unknown")), 0,  
 ifelse(!is.na(df2$CnRelEdu\_1\_01\_Degree) & !is.na(df2$CnRelEdu\_1\_02\_Degree), 2,1))

# Extract year and month from datetime  
df2$First\_Gift\_Year <- year(df2$First\_Gift\_Date)  
df2$First\_Gift\_Month <- month(df2$First\_Gift\_Date)

#Data Cleaning:

# Remove the rows with 0 in column 'Total\_Lifetime\_Giving' which is the target variable to remove distortion in the analysis:  
  
df2 <- df2[df2$Total\_Lifetime\_Giving != 0, ]  
  
# print new dataframe  
print(df2)

## # A tibble: 15,116 × 44  
## CnBio\_ID First\_Gift\_Date Last\_Gift\_Date Largest\_Gift\_Date   
## <dbl> <dttm> <dttm> <dttm>   
## 1 100001361 1973-02-01 00:00:00 2021-12-27 00:00:00 2021-12-27 00:00:00  
## 2 100001363 1986-08-11 00:00:00 2014-06-03 00:00:00 1986-08-11 00:00:00  
## 3 100001364 1971-12-26 00:00:00 2008-11-10 00:00:00 1987-11-19 00:00:00  
## 4 100001366 1977-09-17 00:00:00 2018-04-22 00:00:00 2016-04-24 00:00:00  
## 5 100001368 1985-06-10 00:00:00 1996-11-06 00:00:00 1985-06-10 00:00:00  
## 6 100001373 1977-05-31 00:00:00 2020-03-16 00:00:00 2006-11-14 00:00:00  
## 7 100001376 1977-05-24 00:00:00 2021-11-24 00:00:00 2021-11-24 00:00:00  
## 8 100001378 1968-10-05 00:00:00 2020-06-08 00:00:00 1991-03-13 00:00:00  
## 9 341286 2009-05-11 00:00:00 2011-11-11 00:00:00 2011-11-11 00:00:00  
## 10 337744 2020-05-16 00:00:00 2020-05-16 00:00:00 2020-05-16 00:00:00  
## # ℹ 15,106 more rows  
## # ℹ 40 more variables: CnBio\_DateAdded <dttm>, CnBio\_DateChanged <dttm>,  
## # CnBio\_Key\_Indicator <dbl>, CnBio\_Deceased <dbl>, CnBio\_Title\_1 <chr>,  
## # CnBio\_Marital\_status <chr>, City <chr>, State <fct>,  
## # Reunions\_attended <dbl>, Zip <chr>, CnAdrPrf\_Type <chr>,  
## # CnCnstncy\_1\_01\_CodeLong <chr>, CnCnstncy\_1\_02\_CodeLong <chr>,  
## # CnCnstncy\_1\_03\_CodeLong <chr>, CnCnstncy\_1\_04\_CodeLong <chr>, …

#Dropping the irrelevant columns for the donation purpose:  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

df2 <- df2 %>% select(-CnSpSpBio\_ID,-Personal\_Email\_End,-Largest\_Gift\_Date,-Zip,-CnSpPrBs\_RecordImportID,-CnBio\_Title\_1,-City,-Last\_Gift\_Date,-Married\_to\_an\_Alum,-CnRelEdu\_1\_01\_Class\_of)

#library(dplyr)  
#df2 <- df2 %>% select(-CnRelEdu\_1\_01\_Class\_of)

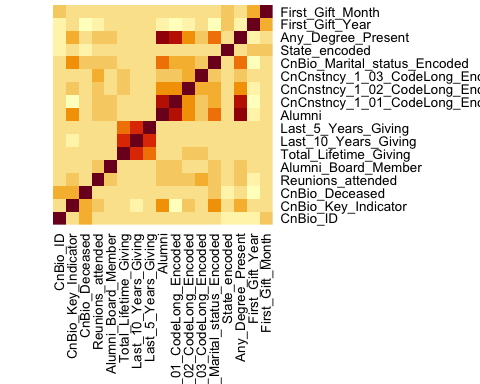
#Remove the maximum value row from Total\_Lifetime\_Giving  
max\_row <- which.max(df2$Total\_Lifetime\_Giving)  
df2 <- df2[-max\_row, ]

#Taking only numeric variables:  
numeric\_df <- subset(df2, select = which(sapply(df2, is.numeric)))

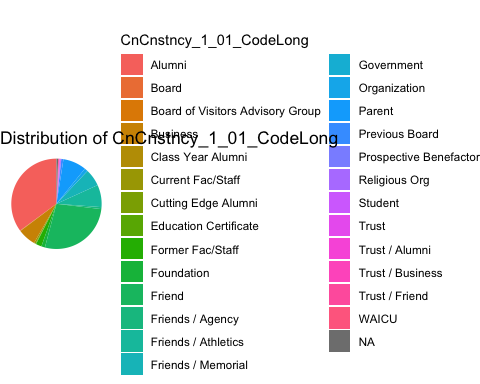
# compute the correlation matrix between all pairs of numeric columns  
print(cor(numeric\_df))

## CnBio\_ID CnBio\_Key\_Indicator CnBio\_Deceased  
## CnBio\_ID 1.000000000 -0.02707257 0.173067828  
## CnBio\_Key\_Indicator -0.027072570 1.00000000 0.157417708  
## CnBio\_Deceased 0.173067828 0.15741771 1.000000000  
## Reunions\_attended 0.011722215 0.03607486 -0.037181403  
## Alumni\_Board\_Member -0.018515463 0.01404689 -0.015135426  
## Total\_Lifetime\_Giving 0.010025236 -0.06476337 -0.005745462  
## Last\_10\_Years\_Giving -0.000172846 -0.08877176 -0.025284441  
## Last\_5\_Years\_Giving 0.021639108 -0.05990592 -0.023797930  
## Alumni -0.075115148 0.24895521 -0.017145338  
## CnCnstncy\_1\_01\_CodeLong\_Encoded -0.039611696 -0.27438305 -0.043194507  
## CnCnstncy\_1\_02\_CodeLong\_Encoded -0.016985372 0.03112182 -0.018741522  
## CnCnstncy\_1\_03\_CodeLong\_Encoded -0.027017278 0.02799232 -0.008792243  
## CnBio\_Marital\_status\_Encoded -0.060076252 0.26824803 0.072669025  
## State\_encoded -0.134263327 -0.02439766 -0.094301389  
## Any\_Degree\_Present -0.105045861 0.23137718 -0.058129480  
## First\_Gift\_Year -0.136690635 -0.01927302 -0.296002621  
## First\_Gift\_Month 0.036036393 -0.03994877 -0.052067518  
## Reunions\_attended Alumni\_Board\_Member  
## CnBio\_ID 0.011722215 -0.0185154632  
## CnBio\_Key\_Indicator 0.036074863 0.0140468939  
## CnBio\_Deceased -0.037181403 -0.0151354262  
## Reunions\_attended 1.000000000 0.0748671877  
## Alumni\_Board\_Member 0.074867188 1.0000000000  
## Total\_Lifetime\_Giving 0.003519344 0.0008028766  
## Last\_10\_Years\_Giving 0.014176013 0.0020996682  
## Last\_5\_Years\_Giving 0.026596257 0.0059125106  
## Alumni 0.079893332 0.0564233770  
## CnCnstncy\_1\_01\_CodeLong\_Encoded 0.076210996 0.0432673310  
## CnCnstncy\_1\_02\_CodeLong\_Encoded 0.100902301 0.0179823034  
## CnCnstncy\_1\_03\_CodeLong\_Encoded 0.159452322 0.0200510561  
## CnBio\_Marital\_status\_Encoded 0.100662864 0.0297046868  
## State\_encoded 0.011511602 0.0157708281  
## Any\_Degree\_Present 0.074956572 0.0740118917  
## First\_Gift\_Year -0.142611822 -0.0108378874  
## First\_Gift\_Month -0.028161465 0.0054782915  
## Total\_Lifetime\_Giving Last\_10\_Years\_Giving  
## CnBio\_ID 0.0100252364 -1.728460e-04  
## CnBio\_Key\_Indicator -0.0647633747 -8.877176e-02  
## CnBio\_Deceased -0.0057454619 -2.528444e-02  
## Reunions\_attended 0.0035193443 1.417601e-02  
## Alumni\_Board\_Member 0.0008028766 2.099668e-03  
## Total\_Lifetime\_Giving 1.0000000000 5.920496e-01  
## Last\_10\_Years\_Giving 0.5920495964 1.000000e+00  
## Last\_5\_Years\_Giving 0.4190867332 6.422503e-01  
## Alumni -0.0230186643 -3.200903e-02  
## CnCnstncy\_1\_01\_CodeLong\_Encoded 0.0154420560 1.550557e-02  
## CnCnstncy\_1\_02\_CodeLong\_Encoded 0.0297843131 2.197695e-02  
## CnCnstncy\_1\_03\_CodeLong\_Encoded 0.0713673700 1.404653e-03  
## CnBio\_Marital\_status\_Encoded 0.0106121228 3.156821e-05  
## State\_encoded 0.0040689666 -3.191102e-03  
## Any\_Degree\_Present -0.0155931585 -2.995966e-02  
## First\_Gift\_Year -0.0276762147 9.896727e-03  
## First\_Gift\_Month 0.0113455100 2.590825e-02  
## Last\_5\_Years\_Giving Alumni  
## CnBio\_ID 0.021639108 -0.07511515  
## CnBio\_Key\_Indicator -0.059905923 0.24895521  
## CnBio\_Deceased -0.023797930 -0.01714534  
## Reunions\_attended 0.026596257 0.07989333  
## Alumni\_Board\_Member 0.005912511 0.05642338  
## Total\_Lifetime\_Giving 0.419086733 -0.02301866  
## Last\_10\_Years\_Giving 0.642250279 -0.03200903  
## Last\_5\_Years\_Giving 1.000000000 -0.02264843  
## Alumni -0.022648431 1.00000000  
## CnCnstncy\_1\_01\_CodeLong\_Encoded 0.008444149 0.76569753  
## CnCnstncy\_1\_02\_CodeLong\_Encoded 0.026330669 0.32290206  
## CnCnstncy\_1\_03\_CodeLong\_Encoded -0.002389547 0.08729421  
## CnBio\_Marital\_status\_Encoded 0.003896422 0.39861807  
## State\_encoded -0.004823998 -0.04192007  
## Any\_Degree\_Present -0.021808978 0.89129643  
## First\_Gift\_Year 0.006392624 -0.14787781  
## First\_Gift\_Month 0.021706413 -0.02750813  
## CnCnstncy\_1\_01\_CodeLong\_Encoded  
## CnBio\_ID -0.039611696  
## CnBio\_Key\_Indicator -0.274383052  
## CnBio\_Deceased -0.043194507  
## Reunions\_attended 0.076210996  
## Alumni\_Board\_Member 0.043267331  
## Total\_Lifetime\_Giving 0.015442056  
## Last\_10\_Years\_Giving 0.015505574  
## Last\_5\_Years\_Giving 0.008444149  
## Alumni 0.765697533  
## CnCnstncy\_1\_01\_CodeLong\_Encoded 1.000000000  
## CnCnstncy\_1\_02\_CodeLong\_Encoded 0.282285411  
## CnCnstncy\_1\_03\_CodeLong\_Encoded 0.069207481  
## CnBio\_Marital\_status\_Encoded 0.234442371  
## State\_encoded -0.017327307  
## Any\_Degree\_Present 0.678909063  
## First\_Gift\_Year -0.224824614  
## First\_Gift\_Month -0.015940561  
## CnCnstncy\_1\_02\_CodeLong\_Encoded  
## CnBio\_ID -0.01698537  
## CnBio\_Key\_Indicator 0.03112182  
## CnBio\_Deceased -0.01874152  
## Reunions\_attended 0.10090230  
## Alumni\_Board\_Member 0.01798230  
## Total\_Lifetime\_Giving 0.02978431  
## Last\_10\_Years\_Giving 0.02197695  
## Last\_5\_Years\_Giving 0.02633067  
## Alumni 0.32290206  
## CnCnstncy\_1\_01\_CodeLong\_Encoded 0.28228541  
## CnCnstncy\_1\_02\_CodeLong\_Encoded 1.00000000  
## CnCnstncy\_1\_03\_CodeLong\_Encoded 0.22385491  
## CnBio\_Marital\_status\_Encoded 0.18261191  
## State\_encoded 0.02382638  
## Any\_Degree\_Present 0.30161287  
## First\_Gift\_Year -0.10473342  
## First\_Gift\_Month -0.01918667  
## CnCnstncy\_1\_03\_CodeLong\_Encoded  
## CnBio\_ID -0.027017278  
## CnBio\_Key\_Indicator 0.027992322  
## CnBio\_Deceased -0.008792243  
## Reunions\_attended 0.159452322  
## Alumni\_Board\_Member 0.020051056  
## Total\_Lifetime\_Giving 0.071367370  
## Last\_10\_Years\_Giving 0.001404653  
## Last\_5\_Years\_Giving -0.002389547  
## Alumni 0.087294213  
## CnCnstncy\_1\_01\_CodeLong\_Encoded 0.069207481  
## CnCnstncy\_1\_02\_CodeLong\_Encoded 0.223854913  
## CnCnstncy\_1\_03\_CodeLong\_Encoded 1.000000000  
## CnBio\_Marital\_status\_Encoded 0.072242463  
## State\_encoded 0.024987907  
## Any\_Degree\_Present 0.099180314  
## First\_Gift\_Year -0.058744451  
## First\_Gift\_Month -0.011761106  
## CnBio\_Marital\_status\_Encoded State\_encoded  
## CnBio\_ID -6.007625e-02 -0.134263327  
## CnBio\_Key\_Indicator 2.682480e-01 -0.024397660  
## CnBio\_Deceased 7.266903e-02 -0.094301389  
## Reunions\_attended 1.006629e-01 0.011511602  
## Alumni\_Board\_Member 2.970469e-02 0.015770828  
## Total\_Lifetime\_Giving 1.061212e-02 0.004068967  
## Last\_10\_Years\_Giving 3.156821e-05 -0.003191102  
## Last\_5\_Years\_Giving 3.896422e-03 -0.004823998  
## Alumni 3.986181e-01 -0.041920074  
## CnCnstncy\_1\_01\_CodeLong\_Encoded 2.344424e-01 -0.017327307  
## CnCnstncy\_1\_02\_CodeLong\_Encoded 1.826119e-01 0.023826376  
## CnCnstncy\_1\_03\_CodeLong\_Encoded 7.224246e-02 0.024987907  
## CnBio\_Marital\_status\_Encoded 1.000000e+00 0.017226068  
## State\_encoded 1.722607e-02 1.000000000  
## Any\_Degree\_Present 3.614488e-01 -0.012859653  
## First\_Gift\_Year -1.903332e-01 0.079489176  
## First\_Gift\_Month -3.643123e-02 0.028969371  
## Any\_Degree\_Present First\_Gift\_Year  
## CnBio\_ID -0.10504586 -0.136690635  
## CnBio\_Key\_Indicator 0.23137718 -0.019273023  
## CnBio\_Deceased -0.05812948 -0.296002621  
## Reunions\_attended 0.07495657 -0.142611822  
## Alumni\_Board\_Member 0.07401189 -0.010837887  
## Total\_Lifetime\_Giving -0.01559316 -0.027676215  
## Last\_10\_Years\_Giving -0.02995966 0.009896727  
## Last\_5\_Years\_Giving -0.02180898 0.006392624  
## Alumni 0.89129643 -0.147877814  
## CnCnstncy\_1\_01\_CodeLong\_Encoded 0.67890906 -0.224824614  
## CnCnstncy\_1\_02\_CodeLong\_Encoded 0.30161287 -0.104733420  
## CnCnstncy\_1\_03\_CodeLong\_Encoded 0.09918031 -0.058744451  
## CnBio\_Marital\_status\_Encoded 0.36144880 -0.190333220  
## State\_encoded -0.01285965 0.079489176  
## Any\_Degree\_Present 1.00000000 -0.118145454  
## First\_Gift\_Year -0.11814545 1.000000000  
## First\_Gift\_Month -0.02140397 0.180048917  
## First\_Gift\_Month  
## CnBio\_ID 0.036036393  
## CnBio\_Key\_Indicator -0.039948769  
## CnBio\_Deceased -0.052067518  
## Reunions\_attended -0.028161465  
## Alumni\_Board\_Member 0.005478291  
## Total\_Lifetime\_Giving 0.011345510  
## Last\_10\_Years\_Giving 0.025908253  
## Last\_5\_Years\_Giving 0.021706413  
## Alumni -0.027508133  
## CnCnstncy\_1\_01\_CodeLong\_Encoded -0.015940561  
## CnCnstncy\_1\_02\_CodeLong\_Encoded -0.019186667  
## CnCnstncy\_1\_03\_CodeLong\_Encoded -0.011761106  
## CnBio\_Marital\_status\_Encoded -0.036431225  
## State\_encoded 0.028969371  
## Any\_Degree\_Present -0.021403968  
## First\_Gift\_Year 0.180048917  
## First\_Gift\_Month 1.000000000

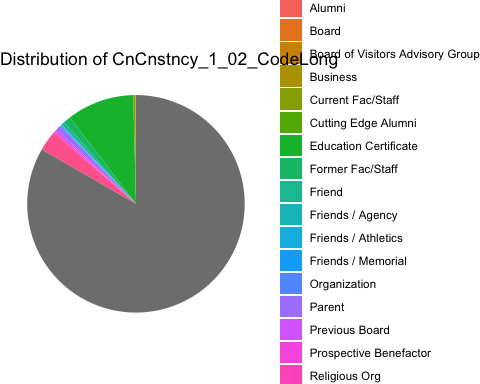
cor\_matrix <- cor(numeric\_df)  
  
# create a heatmap of the correlation matrix  
heatmap(cor\_matrix,   
 Rowv = NA, Colv = NA, # turn off row and column dendrograms  
 symm = TRUE, # use symmetric color scale  
 margins = c(10, 10))

 #Exploratory Data Analysis:

##Graph for :CnCnstncy\_1\_01\_CodeLong\_Encoded:  
  
# Group the data by CnCnstncy\_1\_01\_CodeLong and calculate the total count  
grouped\_df <- df2 %>%   
 group\_by(CnCnstncy\_1\_01\_CodeLong) %>%   
 summarize(total\_count = n())  
  
# Create the pie chart  
ggplot(grouped\_df, aes(x = "", y = total\_count, fill = CnCnstncy\_1\_01\_CodeLong)) +  
 geom\_bar(stat = "identity", width = 1) +  
 coord\_polar(theta = "y") +  
 labs(x = NULL, y = NULL, fill = "CnCnstncy\_1\_01\_CodeLong",   
 title = "Distribution of CnCnstncy\_1\_01\_CodeLong") +  
 theme\_void()



##Graph for :CnCnstncy\_1\_02\_CodeLong\_Encoded:  
  
# Group the data by CnCnstncy\_1\_02\_CodeLong and calculate the total count  
grouped\_df <- df2 %>%   
 group\_by(CnCnstncy\_1\_02\_CodeLong) %>%   
 summarize(total\_count = n())  
  
# Create the pie chart  
ggplot(grouped\_df, aes(x = "", y = total\_count, fill = CnCnstncy\_1\_02\_CodeLong)) +  
 geom\_bar(stat = "identity", width = 1) +  
 coord\_polar(theta = "y") +  
 labs(x = NULL, y = NULL, fill = "CnCnstncy\_1\_02\_CodeLong",   
 title = "Distribution of CnCnstncy\_1\_02\_CodeLong") +  
 theme\_void()

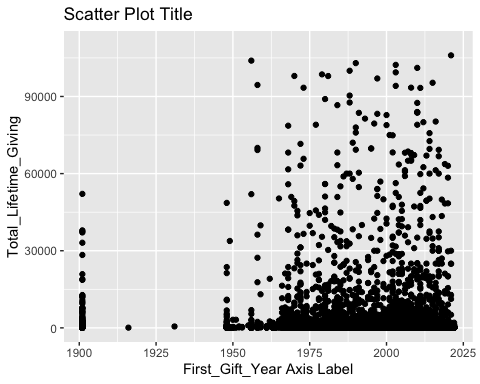


# create scatter plot  
ggplot(df2, aes(x = df2$First\_Gift\_Year, y = df2$Total\_Lifetime\_Giving)) +  
 geom\_point() +  
 scale\_y\_continuous(limits = c(0, 110000)) +  
 labs(x = "First\_Gift\_Year Axis Label", y = "Total\_Lifetime\_Giving", title = "Scatter Plot Title")

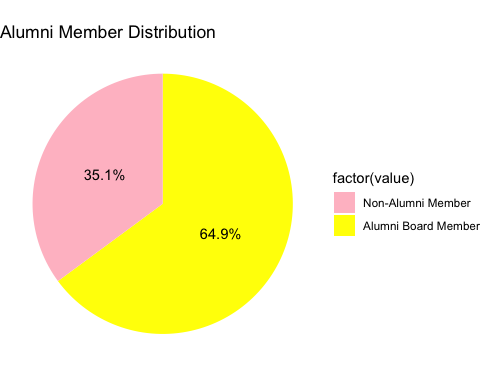
## Warning: Use of `df2$First\_Gift\_Year` is discouraged.  
## ℹ Use `First\_Gift\_Year` instead.

## Warning: Use of `df2$Total\_Lifetime\_Giving` is discouraged.  
## ℹ Use `Total\_Lifetime\_Giving` instead.

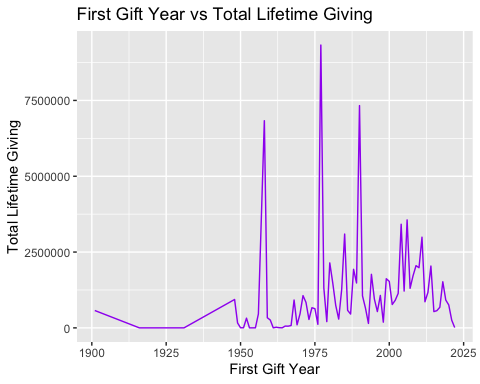
## Warning: Removed 131 rows containing missing values (`geom\_point()`).



# Count the frequency of each value  
freq <- table(df2$Alumni)  
  
# Create a dataframe with the counts and percentages  
df\_freq <- data.frame(value = as.numeric(freq),   
 percentage = round(as.numeric(freq) / sum(as.numeric(freq)) \* 100, 1))  
  
# Create the pie chart with percentage labels  
ggplot(data = df\_freq, aes(x = "", y = value, fill = factor(value))) +  
 geom\_bar(width = 1, stat = "identity") +  
 coord\_polar(theta = "y") +  
 geom\_text(aes(label = paste0(percentage, "%")),   
 position = position\_stack(vjust = 0.5)) +  
 labs(title = "Alumni Member Distribution") +  
 scale\_fill\_manual(values = c("Pink", "Yellow"),   
 labels = c("Non-Alumni Member", "Alumni Board Member")) +  
 theme\_void()

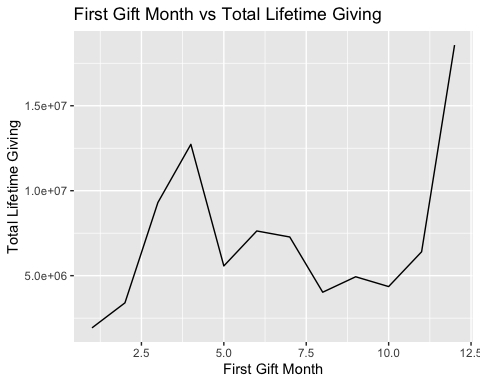


# Group the data by year and calculate the total lifetime giving for each year  
total\_giving\_by\_year <- df2 %>%  
 group\_by(First\_Gift\_Year) %>%  
 summarize(Total\_Lifetime\_Giving = sum(Total\_Lifetime\_Giving))  
  
ggplot(total\_giving\_by\_year, aes(x = First\_Gift\_Year, y = Total\_Lifetime\_Giving)) +  
 geom\_line(color = "purple") +  
 labs(x = "First Gift Year", y = "Total Lifetime Giving", title = "First Gift Year vs Total Lifetime Giving")



##Dropping the rows as the values in The First Gift Year from 1900 to 1940:  
library(dplyr)  
  
numeric\_df <- numeric\_df %>%  
 filter(First\_Gift\_Year > 1940)

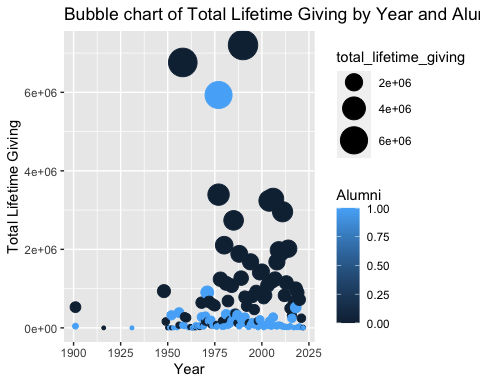
# Group the data by year and calculate the total lifetime giving for each year  
total\_giving\_by\_month <- numeric\_df %>%  
 group\_by(First\_Gift\_Month) %>%  
 summarize(Total\_Lifetime\_Giving = sum(Total\_Lifetime\_Giving))  
  
ggplot(total\_giving\_by\_month, aes(x = First\_Gift\_Month, y = Total\_Lifetime\_Giving)) +  
 geom\_line() +  
 labs(x = "First Gift Month", y = "Total Lifetime Giving", title = "First Gift Month vs Total Lifetime Giving")



library(ggplot2)  
##1  
# Group the data by Year and Alumni and calculate the total lifetime giving  
grouped\_df <- df2 %>%   
 group\_by(First\_Gift\_Year, Alumni) %>%   
 summarize(total\_lifetime\_giving = sum(Total\_Lifetime\_Giving))

## `summarise()` has grouped output by 'First\_Gift\_Year'. You can override using  
## the `.groups` argument.

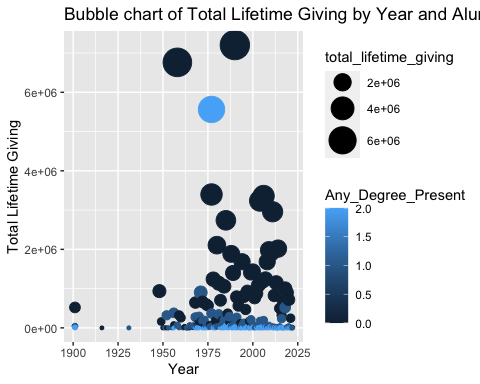
# Create the bubble chart  
ggplot(grouped\_df, aes(x = First\_Gift\_Year ,y = total\_lifetime\_giving, size = total\_lifetime\_giving, color = Alumni)) +  
 geom\_point() +  
 scale\_size(range = c(1, 10)) +  
 labs(x = "Year", y = "Total Lifetime Giving",   
 title = "Bubble chart of Total Lifetime Giving by Year and Alumni")



##2  
  
# Group the data by Year and Any\_Degree\_Present and calculate the total lifetime giving  
grouped\_df <- df2 %>%   
 group\_by(First\_Gift\_Year,Any\_Degree\_Present ) %>%   
 summarize(total\_lifetime\_giving = sum(Total\_Lifetime\_Giving))

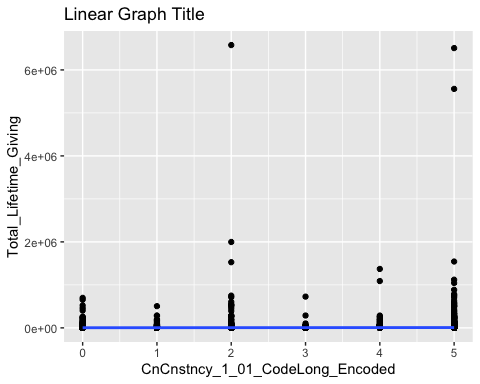
## `summarise()` has grouped output by 'First\_Gift\_Year'. You can override using  
## the `.groups` argument.

# Create the bubble chart  
ggplot(grouped\_df, aes(x = First\_Gift\_Year ,y = total\_lifetime\_giving, size = total\_lifetime\_giving, color = Any\_Degree\_Present)) +  
 geom\_point() +  
 scale\_size(range = c(1, 10)) +  
 labs(x = "Year", y = "Total Lifetime Giving",   
 title = "Bubble chart of Total Lifetime Giving by Year and Alumni Board Member")



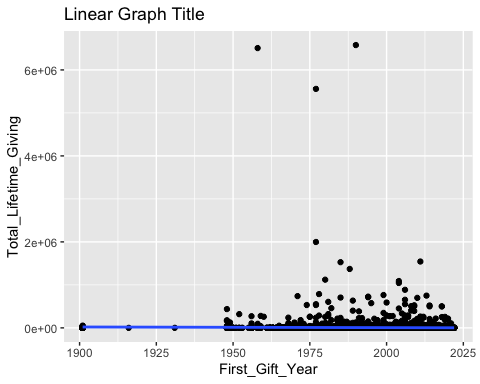
##1  
  
# create the linear graph  
ggplot(df2, aes(CnCnstncy\_1\_01\_CodeLong\_Encoded, Total\_Lifetime\_Giving)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se = FALSE) +  
 labs(x = "CnCnstncy\_1\_01\_CodeLong\_Encoded", y = "Total\_Lifetime\_Giving", title = "Linear Graph Title")

## `geom\_smooth()` using formula = 'y ~ x'



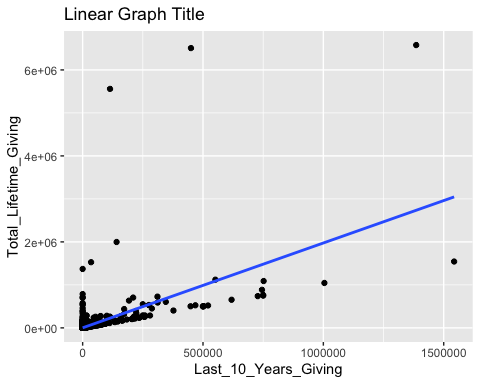
##2  
# create the linear graph  
ggplot(df2, aes(First\_Gift\_Year, Total\_Lifetime\_Giving)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se = FALSE) +  
 labs(x = "First\_Gift\_Year", y = "Total\_Lifetime\_Giving", title = "Linear Graph Title")

## `geom\_smooth()` using formula = 'y ~ x'



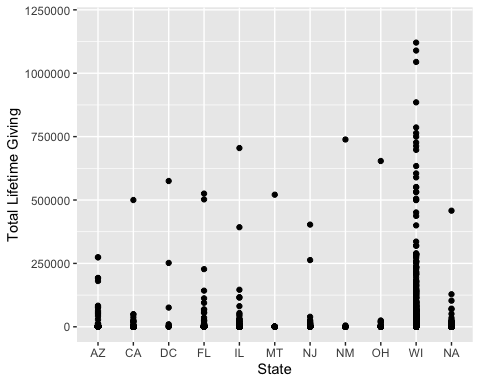
##3  
# create the linear graph  
ggplot(df2, aes(Last\_10\_Years\_Giving, Total\_Lifetime\_Giving)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se = FALSE) +  
 labs(x = "Last\_10\_Years\_Giving", y = "Total\_Lifetime\_Giving", title = "Linear Graph Title")

## `geom\_smooth()` using formula = 'y ~ x'

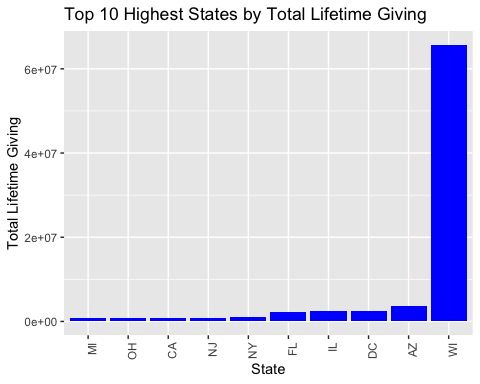


library(dplyr) # load the dplyr package for data manipulation  
  
# sort the data frame by Total\_Lifetime\_Giving in descending order, and select the top 10 states  
topstates <- df2 %>%  
 arrange(desc(Total\_Lifetime\_Giving)) %>%  
 slice(1:40) %>%  
 pull(State)  
  
# create a scatter plot for the top states  
ggplot(data = filter(df2, State %in% topstates),  
 aes(x = State, y = Total\_Lifetime\_Giving)) +  
 geom\_point() +  
 labs(x = "State", y = "Total Lifetime Giving") +  
 ylim(0, 1200000) # adjust the y-axis limits to accommodate the highest value

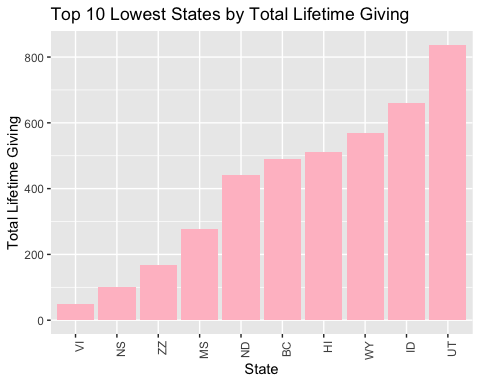
## Warning: Removed 7 rows containing missing values (`geom\_point()`).



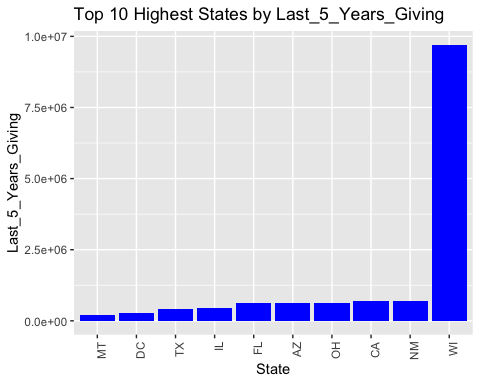
library(ggplot2)  
library(dplyr)  
##Topwith Total   
df\_top10 <- df2 %>%  
 group\_by(State) %>%  
 summarise(Total\_Lifetime\_Giving = sum(Total\_Lifetime\_Giving)) %>%  
 filter(State != "N/A") %>%  
 arrange(desc(Total\_Lifetime\_Giving)) %>%  
 head(10)  
  
ggplot(df\_top10, aes(x = reorder(State, Total\_Lifetime\_Giving), y = Total\_Lifetime\_Giving)) +  
 geom\_bar(stat = "identity", fill = "blue") +  
 ggtitle("Top 10 Highest States by Total Lifetime Giving") +  
 xlab("State") +  
 ylab("Total Lifetime Giving") +  
 theme(axis.text.x = element\_text(angle = 90, hjust = 1))



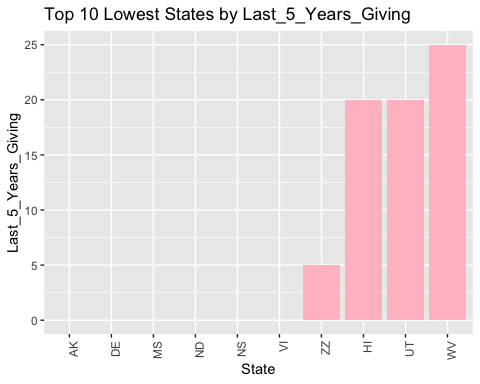
##bottom with total  
df\_bottom10 <- df2 %>%  
 group\_by(State) %>%  
 summarise(Total\_Lifetime\_Giving = sum(Total\_Lifetime\_Giving)) %>%  
 filter(State != "N/A") %>%  
 arrange(Total\_Lifetime\_Giving) %>%  
 head(10)  
  
ggplot(df\_bottom10, aes(x = reorder(State, Total\_Lifetime\_Giving), y = Total\_Lifetime\_Giving)) +  
 geom\_bar(stat = "identity", fill = "pink") +  
 ggtitle("Top 10 Lowest States by Total Lifetime Giving") +  
 xlab("State") +  
 ylab("Total Lifetime Giving") +  
 theme(axis.text.x = element\_text(angle = 90, hjust = 1))



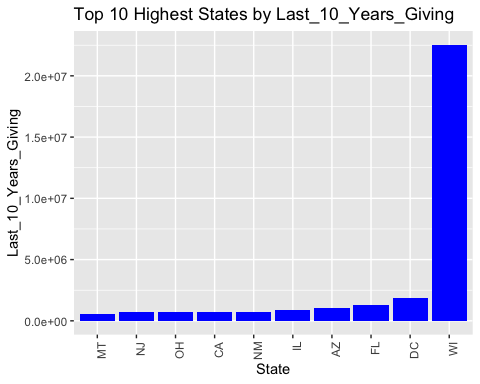
#Top with 5years  
  
  
df\_top10 <- df2 %>%  
 group\_by(State) %>%  
 summarise(Last\_5\_Years\_Giving = sum(Last\_5\_Years\_Giving)) %>%  
 filter(State != "N/A") %>%  
 arrange(desc(Last\_5\_Years\_Giving)) %>%  
 head(10)  
  
ggplot(df\_top10, aes(x = reorder(State, Last\_5\_Years\_Giving), y = Last\_5\_Years\_Giving)) +  
 geom\_bar(stat = "identity", fill = "blue") +  
 ggtitle("Top 10 Highest States by Last\_5\_Years\_Giving") +  
 xlab("State") +  
 ylab("Last\_5\_Years\_Giving") +  
 theme(axis.text.x = element\_text(angle = 90, hjust = 1))



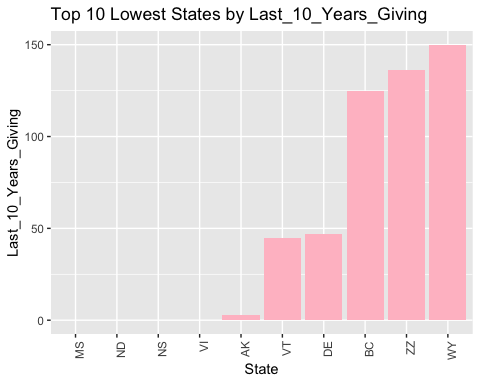
#bottom with the 5 years  
  
df\_bottom10 <- df2 %>%  
 group\_by(State) %>%  
 summarise(Last\_5\_Years\_Giving = sum(Last\_5\_Years\_Giving)) %>%  
 filter(State != "N/A") %>%  
 arrange(Last\_5\_Years\_Giving) %>%  
 head(10)  
  
ggplot(df\_bottom10, aes(x = reorder(State, Last\_5\_Years\_Giving), y = Last\_5\_Years\_Giving)) +  
 geom\_bar(stat = "identity", fill = "pink") +  
 ggtitle("Top 10 Lowest States by Last\_5\_Years\_Giving") +  
 xlab("State") +  
 ylab("Last\_5\_Years\_Giving") +  
 theme(axis.text.x = element\_text(angle = 90, hjust = 1))



#top with 10 years  
df\_top10 <- df2 %>%  
 group\_by(State) %>%  
 summarise(Last\_10\_Years\_Giving = sum(Last\_10\_Years\_Giving)) %>%  
 filter(State != "N/A") %>%  
 arrange(desc(Last\_10\_Years\_Giving)) %>%  
 head(10)  
  
ggplot(df\_top10, aes(x = reorder(State, Last\_10\_Years\_Giving), y = Last\_10\_Years\_Giving)) +  
 geom\_bar(stat = "identity", fill = "blue") +  
 ggtitle("Top 10 Highest States by Last\_10\_Years\_Giving") +  
 xlab("State") +  
 ylab("Last\_10\_Years\_Giving") +  
 theme(axis.text.x = element\_text(angle = 90, hjust = 1))



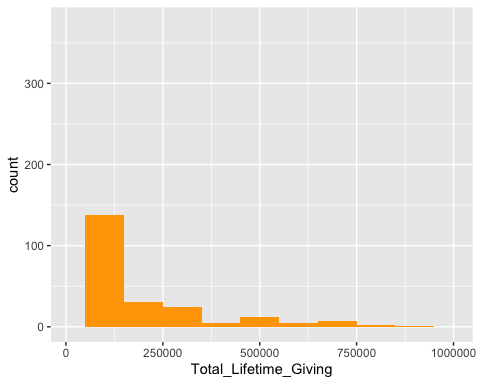
#Bottom 10 with Last 10 years  
df\_bottom10 <- df2 %>%  
 group\_by(State) %>%  
 summarise(Last\_10\_Years\_Giving = sum(Last\_10\_Years\_Giving)) %>%  
 filter(State != "N/A") %>%  
 arrange(Last\_10\_Years\_Giving) %>%  
 head(10)  
  
ggplot(df\_bottom10, aes(x = reorder(State, Last\_10\_Years\_Giving), y = Last\_10\_Years\_Giving)) +  
 geom\_bar(stat = "identity", fill = "pink") +  
 ggtitle("Top 10 Lowest States by Last\_10\_Years\_Giving") +  
 xlab("State") +  
 ylab("Last\_10\_Years\_Giving") +  
 theme(axis.text.x = element\_text(angle = 90, hjust = 1))



#1  
  
ggplot(data=numeric\_df, aes(x=Total\_Lifetime\_Giving)) +  
 geom\_histogram(binwidth=100000, fill= "orange") +  
 xlim(10000, 1000000)

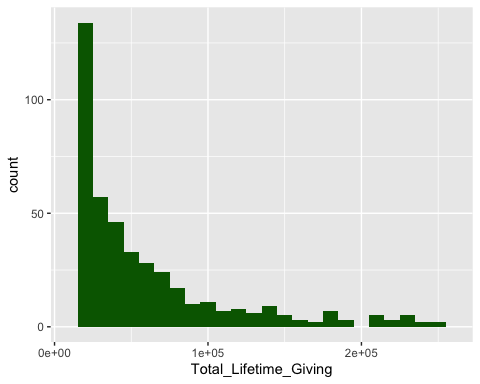
## Warning: Removed 14326 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 2 rows containing missing values (`geom\_bar()`).



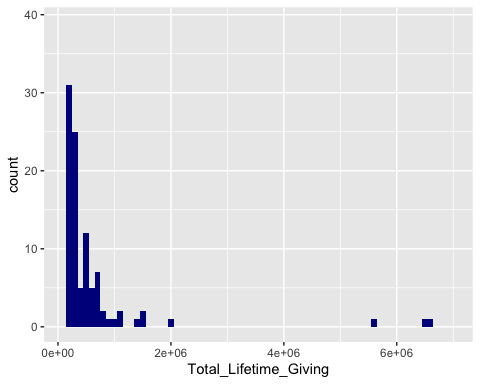
#2  
ggplot(data=numeric\_df, aes(x=Total\_Lifetime\_Giving)) +  
 geom\_histogram(binwidth=10000, fill="darkgreen") +  
 xlim(10000, 260000)

## Warning: Removed 14375 rows containing non-finite values (`stat\_bin()`).  
## Removed 2 rows containing missing values (`geom\_bar()`).



#3  
ggplot(data=numeric\_df, aes(x=Total\_Lifetime\_Giving)) +  
 geom\_histogram(binwidth=100000, fill="darkblue") +  
 xlim(100000, 7000000)

## Warning: Removed 14790 rows containing non-finite values (`stat\_bin()`).  
## Removed 2 rows containing missing values (`geom\_bar()`).



df=numeric\_df  
  
dim(df)

## [1] 14927 17

df$log\_Total\_Lifetime\_Giving <- log(df$Total\_Lifetime\_Giving)

# count missing values  
colSums(is.na(numeric\_df))

## CnBio\_ID CnBio\_Key\_Indicator   
## 0 0   
## CnBio\_Deceased Reunions\_attended   
## 0 0   
## Alumni\_Board\_Member Total\_Lifetime\_Giving   
## 0 0   
## Last\_10\_Years\_Giving Last\_5\_Years\_Giving   
## 0 0   
## Alumni CnCnstncy\_1\_01\_CodeLong\_Encoded   
## 0 0   
## CnCnstncy\_1\_02\_CodeLong\_Encoded CnCnstncy\_1\_03\_CodeLong\_Encoded   
## 0 0   
## CnBio\_Marital\_status\_Encoded State\_encoded   
## 0 0   
## Any\_Degree\_Present First\_Gift\_Year   
## 0 0   
## First\_Gift\_Month   
## 0

library(dplyr)  
#numeric\_df <- numeric\_df %>% select(-CnRelEdu\_1\_01\_Class\_of)  
  
#numeric\_df <- numeric\_df %>% select(-CnSpSpBio\_ID)

# Build a linear regression model  
model <- lm(Total\_Lifetime\_Giving ~ ., data = numeric\_df)  
  
# Get the fitted values and residuals  
fitted <- predict(model)  
residuals <- residuals(model)  
  
# Create a data frame with the fitted values and residuals  
plot\_df <- data.frame(Fitted\_Values = fitted, Residuals = residuals)  
  
# Plot the residuals against the fitted values  
ggplot(plot\_df, aes(Fitted\_Values, Residuals)) +  
 geom\_point() +  
 geom\_smooth(se = FALSE) +  
 labs(x = "Fitted Values", y = "Residuals", title = "Residual Plot")

## `geom\_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'



##When conducting a residual analysis, a "residuals versus fits plot" is the most frequently created plot. It is a scatter plot of residuals on the y axis and fitted values (estimated responses) on the x axis. The plot is used to detect non-linearity.

#Model Evaluation:

#Finding the optimal features for analysis:

## Lasso find the optimal variables:  
# Load the glmnet package  
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-7

# Convert the data to matrix format  
x <- as.matrix(numeric\_df[, -which(names(numeric\_df) == "Total\_Lifetime\_Giving")])  
y <- numeric\_df$Total\_Lifetime\_Giving  
  
# Perform LASSO regularization  
lasso <- glmnet(x, y, alpha = 1)  
  
# Plot the LASSO regularization path  
plot(lasso, xvar = "lambda", label = TRUE,width = 25, height = 20)

## Warning in plot.window(...): "width" is not a graphical parameter

## Warning in plot.window(...): "height" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "width" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "height" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "width" is not a  
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "height" is not a  
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "width" is not a  
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "height" is not a  
## graphical parameter

## Warning in box(...): "width" is not a graphical parameter

## Warning in box(...): "height" is not a graphical parameter

## Warning in title(...): "width" is not a graphical parameter

## Warning in title(...): "height" is not a graphical parameter

# Choose the optimal lambda value using cross-validation  
cv.lasso <- cv.glmnet(x, y, alpha = 1)  
lambda <- cv.lasso$lambda.min  
  
# Extract the coefficients for the optimal lambda value  
lasso.coef <- coef(lasso, s = lambda)  
lasso.coef <- lasso.coef[-1, ] # Exclude the intercept term  
  
# Identify the most important features  
lasso.features <- names(numeric\_df)[-which(names(numeric\_df) == "Total\_Lifetime\_Giving")]  
lasso.features <- lasso.features[which(lasso.coef != 0)]  
  
  
plot(lasso, xvar = "lambda", label = TRUE, width = 25, height = 20)

## Warning in plot.window(...): "width" is not a graphical parameter

## Warning in plot.window(...): "height" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "width" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "height" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "width" is not a  
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "height" is not a  
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "width" is not a  
## graphical parameter

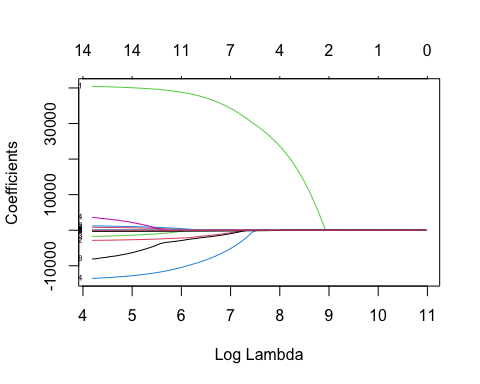
## Warning in axis(side = side, at = at, labels = labels, ...): "height" is not a  
## graphical parameter

## Warning in box(...): "width" is not a graphical parameter

## Warning in box(...): "height" is not a graphical parameter

## Warning in title(...): "width" is not a graphical parameter

## Warning in title(...): "height" is not a graphical parameter



#Analyzing with Linear Regression with Train-Test and Validating the data:

##On Training test and validation for linear regression:  
library(caret)

## Loading required package: lattice

# Split the data into training/validation and testing sets  
set.seed(123)  
trainIndex <- createDataPartition(numeric\_df$Total\_Lifetime\_Giving, p = 0.75, list = FALSE)  
train\_val <- numeric\_df[trainIndex, ]  
test <- numeric\_df[-trainIndex, ]  
  
# Split the training/validation set into a training set and a validation set  
  
trainIndex2 <- createDataPartition(train\_val$Total\_Lifetime\_Giving, p = 0.7, list = FALSE)  
train <- train\_val[trainIndex2, ]  
validation <- train\_val[-trainIndex2, ]  
  
# Fit the model using the training set  
model <- lm(Total\_Lifetime\_Giving ~ CnCnstncy\_1\_02\_CodeLong\_Encoded + CnBio\_Key\_Indicator+ CnCnstncy\_1\_03\_CodeLong\_Encoded+Last\_10\_Years\_Giving+Reunions\_attended+Last\_5\_Years\_Giving+CnCnstncy\_1\_01\_CodeLong\_Encoded,  
 data = train)  
  
  
# Use the model to predict the testing set  
predictions\_test <- predict(model, newdata = test)  
  
# Calculate the R-squared value of the model on the validation set  
rsq\_val <- summary(model)$r.squared  
print(rsq\_val)

## [1] 0.6009925

# Calculate the R-squared value of the model on the testing set  
rsq\_test <- 1 - sum((test$Total\_Lifetime\_Giving - predictions\_test)^2) / sum((test$Total\_Lifetime\_Giving - mean(test$Total\_Lifetime\_Giving))^2)  
print(rsq\_test)

## [1] 0.1075281

#Analyzing with Ridge Regression with Train-Test and Validating the data:

library(glmnet)  
  
# Split Ridge the data into training, validation, and testing sets  
set.seed(123)  
train\_index <- sample(1:nrow(numeric\_df), 0.7 \* nrow(numeric\_df))  
val\_index <- sample(setdiff(1:nrow(numeric\_df), train\_index), 0.2 \* nrow(numeric\_df))  
test\_index <- setdiff(setdiff(1:nrow(numeric\_df), train\_index), val\_index)  
train <- numeric\_df[train\_index, ]  
val <- numeric\_df[val\_index, ]  
test <- numeric\_df[test\_index, ]  
  
# Prepare the data for modeling  
y\_train <- train$Total\_Lifetime\_Giving  
y\_val <- val$Total\_Lifetime\_Giving  
y\_test <- test$Total\_Lifetime\_Giving  
x\_train <- data.matrix(train[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
x\_val <- data.matrix(val[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
x\_test <- data.matrix(test[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
x\_val <- data.matrix(val[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
  
# Fit the Ridge regression model using cross-validation on the training set  
cv\_model <- cv.glmnet(x\_train, y\_train, alpha = 0, lambda = seq(0.001, 1, length = 100))  
  
# Extract the best lambda value using the validation set  
best\_lambda <- cv\_model$lambda.min  
  
# Fit the Ridge regression model with the best lambda value using the training set  
best\_model <- glmnet(x\_train, y\_train, alpha = 0, lambda = best\_lambda)  
  
# Use the fitted model to make predictions on the test set  
y\_predicted <- predict(best\_model, s = best\_lambda, newx = x\_test)  
  
# Find SST and SSE  
sst <- sum((y\_test - mean(y\_train))^2)  
sse <- sum((y\_predicted - y\_test)^2)  
  
# Find R-squared  
rsq <- 1 - sse/sst  
rsq

## [1] 0.05516379

#Analyzing with Lasso Regression with Train-Test and Validating the data:

##lasso with validation:  
  
# Load necessary libraries  
library(caret)  
library(glmnet)  
  
# Split data into training, validation, and test sets  
set.seed(123)  
train\_index <- createDataPartition(numeric\_df$Total\_Lifetime\_Giving, p = 0.7, list = FALSE)  
train\_data <- numeric\_df[train\_index, ]  
valid\_test\_data <- numeric\_df[-train\_index, ]  
valid\_index <- createDataPartition(valid\_test\_data$Total\_Lifetime\_Giving, p = 0.5, list = FALSE)  
valid\_data <- valid\_test\_data[valid\_index, ]  
test\_data <- valid\_test\_data[-valid\_index, ]  
  
# Define response variable  
y\_train <- train\_data$Total\_Lifetime\_Giving  
y\_valid <- valid\_data$Total\_Lifetime\_Giving  
y\_test <- test\_data$Total\_Lifetime\_Giving  
  
# Define matrix of predictor variables  
x\_train <- data.matrix(train\_data[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Last\_10\_Years\_Giving','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
x\_valid <- data.matrix(valid\_data[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Last\_10\_Years\_Giving','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
x\_test <- data.matrix(test\_data[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Last\_10\_Years\_Giving','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
  
# Fit Lasso regression model using cross-validation on the training set  
cv\_model <- cv.glmnet(x\_train, y\_train, alpha = 1, lambda = seq(0.001, 1, length = 100))  
  
# Extract the best lambda value  
best\_lambda <- cv\_model$lambda.min  
  
# Fit Lasso regression model with best lambda value using the combined training and validation set  
best\_model <- glmnet(x\_train, y\_train, alpha = 1, lambda = best\_lambda)  
y\_valid\_predicted <- predict(best\_model, s = best\_lambda, newx = x\_valid)  
  
  
# Make predictions on test set using the fitted model  
y\_test\_predicted <- predict(best\_model, s = best\_lambda, newx = x\_test)  
  
# Find SST and SSE for test set  
sst\_test <- sum((y\_test - mean(y\_test))^2)  
sse\_test <- sum((y\_test\_predicted - y\_test)^2)  
  
# Find R-Squared for test set  
rsq\_test <- 1 - sse\_test/sst\_test  
rsq\_test

## [1] 0.05918001

#Analyzing with Decision tree with Train-Test and Validating the data:

##validation for decision\_tree  
library(pROC)

## Type 'citation("pROC")' for a citation.

##   
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

library(rpart)  
  
# Define response variable  
y <- numeric\_df$Total\_Lifetime\_Giving  
  
# Define matrix of predictor variables  
x <- data.matrix(numeric\_df[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
  
# Split the data into training, validation, and testing sets  
set.seed(123)  
n <- nrow(numeric\_df)  
train\_index <- sample(n, 0.75 \* n)  
valid\_index <- sample(setdiff(1:n, train\_index), 0.2 \* n)  
test\_index <- setdiff(setdiff(1:n, train\_index), valid\_index)  
x\_train <- x[train\_index, ]  
y\_train <- y[train\_index]  
x\_valid <- x[valid\_index, ]  
y\_valid <- y[valid\_index]  
x\_test <- x[test\_index, ]  
y\_test <- y[test\_index]  
  
# Fit decision tree regression model on training set  
model <- rpart(y\_train ~ ., data = data.frame(x\_train, y\_train), method = "anova")  
  
# Use fitted model to make predictions on validation set  
y\_predicted\_valid <- predict(model, newdata = data.frame(x\_valid))  
  
# Calculate RMSE and R-squared on validation set  
rmse\_valid <- sqrt(mean((y\_predicted\_valid - y\_valid)^2))  
sst\_valid <- sum((y\_valid - mean(y\_valid))^2)  
sse\_valid <- sum((y\_predicted\_valid - y\_valid)^2)  
rsq\_valid <- 1 - sse\_valid/sst\_valid  
  
# Use fitted model to make predictions on test set  
y\_predicted\_test <- predict(model, newdata = data.frame(x\_test), type = "vector")  
  
# Calculate R-squared on test set  
sst\_test <- sum((y\_test - mean(y\_test))^2)  
sse\_test <- sum((y\_predicted\_test - y\_test)^2)  
rsq\_test <- 1 - sse\_test/sst\_test  
  
# Print the R-squared values for test sets  
  
cat("Test set:\n")

## Test set:

cat(paste0("R-squared: ", rsq\_test, "\n"))

## R-squared: 0.358177370788151

#Analyzing with Random forest Regression with Train-Test and Validating the data:

library(randomForest)

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:ggplot2':  
##   
## margin

# Split the data into training, validation, and testing sets  
set.seed(123)  
train\_index <- sample(nrow(numeric\_df), 0.75 \* nrow(numeric\_df))  
val\_index <- sample(setdiff(1:nrow(numeric\_df), train\_index), 0.2 \* nrow(numeric\_df))  
test\_index <- setdiff(setdiff(1:nrow(numeric\_df), train\_index), val\_index)  
x\_train <- x[train\_index, ]  
y\_train <- y[train\_index]  
x\_val <- x[val\_index, ]  
y\_val <- y[val\_index]  
x\_test <- x[test\_index, ]  
y\_test <- y[test\_index]  
  
  
# Fit random forest regression model on training set  
model <- randomForest(y\_train ~ ., data = data.frame(x\_train, y\_train))  
  
# Use fitted model to make predictions on validation set  
y\_predicted\_val <- predict(model, newdata = data.frame(x\_val))  
  
# Calculate R-squared on validation set  
sst\_val <- sum((y\_val - mean(y\_val))^2)  
sse\_val <- sum((y\_predicted\_val - y\_val)^2)  
rsq\_val <- 1 - sse\_val/sst\_val  
  
  
# Use fitted model to make predictions on test set  
y\_predicted\_test <- predict(model, newdata = data.frame(x\_test))  
  
# Calculate R-squared on test set  
sst\_test <- sum((y\_test - mean(y\_test))^2)  
sse\_test <- sum((y\_predicted\_test - y\_test)^2)  
rsq\_test <- 1 - sse\_test/sst\_test  
  
# Print the R-squared values for test set  
print(paste0("Test Set R-squared: ", rsq\_test))

## [1] "Test Set R-squared: 0.369206316866739"

#Analyzing with SVM with Train-Test and Validating the data:

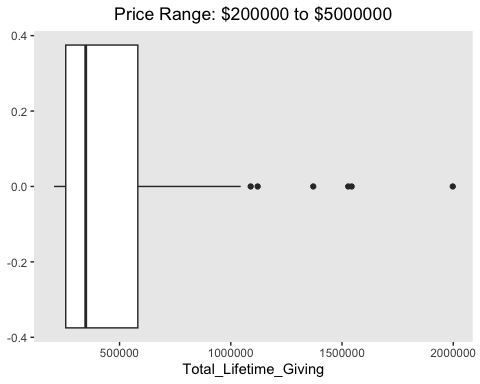
library(e1071)  
  
# Define response variable  
y <- numeric\_df$Total\_Lifetime\_Giving  
  
# Define matrix of predictor variables  
x <- data.matrix(df2[, c('CnCnstncy\_1\_02\_CodeLong\_Encoded', 'CnCnstncy\_1\_03\_CodeLong\_Encoded','Last\_10\_Years\_Giving','Reunions\_attended','Last\_5\_Years\_Giving','CnCnstncy\_1\_01\_CodeLong\_Encoded','CnBio\_Key\_Indicator')])  
  
# Split the data into training, validation, and testing sets  
set.seed(123)  
train\_index <- sample(nrow(numeric\_df), 0.75 \* nrow(numeric\_df))  
val\_index <- sample(setdiff(1:nrow(numeric\_df), train\_index), 0.2 \* nrow(numeric\_df))  
test\_index <- setdiff(setdiff(1:nrow(numeric\_df), train\_index), val\_index)  
x\_train <- x[train\_index, ]  
y\_train <- y[train\_index]  
x\_val <- x[val\_index, ]  
y\_val <- y[val\_index]  
x\_test <- x[test\_index, ]  
y\_test <- y[test\_index]  
  
# Fit support vector regression model on training set with default hyperparameters  
model <- svm(y\_train ~ ., data = data.frame(x\_train, y\_train), kernel = "radial")  
  
# Use validation set to tune hyperparameters  
tuned\_model <- tune.svm(y\_train ~ ., data = data.frame(x\_train, y\_train), kernel = "radial", gamma = 10^(-6:1), cost = 10^(-1:2), tunecontrol = tune.control(cross = 2))  
summary(tuned\_model)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 2-fold cross validation   
##   
## - best parameters:  
## gamma cost  
## 0.1 10  
##   
## - best performance: 12748660349   
##   
## - Detailed performance results:  
## gamma cost error dispersion  
## 1 1e-06 0.1 12750101515 3894746525  
## 2 1e-05 0.1 12750100153 3894744768  
## 3 1e-04 0.1 12749670276 3894864814  
## 4 1e-03 0.1 12749645704 3895053113  
## 5 1e-02 0.1 12749857221 3894855873  
## 6 1e-01 0.1 12749735411 3895231495  
## 7 1e+00 0.1 12749897209 3894993204  
## 8 1e+01 0.1 12749809724 3894968427  
## 9 1e-06 1.0 12750100018 3894744947  
## 10 1e-05 1.0 12749611506 3894944453  
## 11 1e-04 1.0 12749689822 3894926097  
## 12 1e-03 1.0 12749757067 3894841483  
## 13 1e-02 1.0 12749516370 3895008346  
## 14 1e-01 1.0 12749920515 3896409468  
## 15 1e+00 1.0 12749911132 3896780961  
## 16 1e+01 1.0 12751529132 3896038710  
## 17 1e-06 10.0 12749434664 3895208452  
## 18 1e-05 10.0 12749727253 3895026400  
## 19 1e-04 10.0 12749714278 3894886585  
## 20 1e-03 10.0 12749666210 3894949280  
## 21 1e-02 10.0 12750042864 3895955836  
## 22 1e-01 10.0 12748660349 3897233244  
## 23 1e+00 10.0 12751137266 3896695392  
## 24 1e+01 10.0 12756238532 3893894592  
## 25 1e-06 100.0 12749823678 3894865986  
## 26 1e-05 100.0 12749800633 3894987331  
## 27 1e-04 100.0 12749650795 3894965142  
## 28 1e-03 100.0 12749524068 3894966449  
## 29 1e-02 100.0 12751372092 3898442548  
## 30 1e-01 100.0 12752780998 3892234372  
## 31 1e+00 100.0 12776942958 3866406718  
## 32 1e+01 100.0 12757790091 3895998057

best\_gamma <- tuned\_model$best.parameters$gamma  
best\_cost <- tuned\_model$best.parameters$cost  
  
# Fit support vector regression model on training set with tuned hyperparameters  
final\_model <- svm(y\_train ~ ., data = data.frame(x\_train, y\_train), kernel = "radial", gamma = best\_gamma, cost = best\_cost)  
  
# Use fitted model to make predictions on test set  
y\_predicted <- predict(final\_model, newdata = data.frame(x\_test))  
  
# Calculate R-squared on test set  
sst <- sum((y\_test - mean(y\_test))^2)  
sse <- sum((y\_predicted - y\_test)^2)  
rsq <- 1 - sse/sst  
  
# Print the R-squared values  
print(paste0("R-squared: ", rsq))

## [1] "R-squared: -0.0474218863771314"

#Filter data for price between 1,000 USD and 200,000 USD  
df\_filtered <- subset(numeric\_df, Total\_Lifetime\_Giving >= 200000 & Total\_Lifetime\_Giving <= 5000000)  
  
#Load the ggplot2 library  
library(ggplot2)  
  
#Create a box plot of the 'price' column of the filtered DataFrame  
ggplot(df\_filtered, aes(x = Total\_Lifetime\_Giving)) +  
geom\_boxplot() +  
ggtitle("Price Range: $200000 to $5000000") +  
theme(plot.title = element\_text(hjust = 0.5),  
panel.grid.major = element\_blank(),  
panel.grid.minor = element\_blank())

 #Create a R2 frame:

library(dplyr)  
library(ggplot2)  
  
# create a data frame for R-squared values  
rsq\_df <- data.frame(Model = c("Linear Regression","Ridge Regression","Lasso Regression","Decision Tree","Random Forest","SVM"),   
 R\_Squared = c(0.1075281,0.055,0.128, 0.3581, 0.8424,0.70267))  
  
# reorder the data frame by R-squared values in descending order  
rsq\_df <- rsq\_df %>%   
 arrange(desc(R\_Squared)) %>%   
 mutate(Model = factor(Model, levels = Model))  
  
# create a bar plot for R-squared values with adjusted width  
ggplot(rsq\_df, aes(x = Model, y = R\_Squared, width = 0.3)) +  
 geom\_bar(stat = "identity", fill = "darkblue") +  
 labs(x = "Model", y = "R-Squared", title = "Comparison of R-Squared Values") +  
 theme(axis.text.x = element\_text(angle = 0, vjust = 0.2, hjust = 0.5))

