Diabetes_and_T_Swift_Data

Jean Jimenez, Brandon Cunningham, Chafiaa Nadour

2024-10-17

Data 622 HW1

```
library(tidyverse)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
## Warning: package 'tidyr' was built under R version 4.3.3
## Warning: package 'readr' was built under R version 4.3.3
## Warning: package 'dplyr' was built under R version 4.3.2
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggcorrplot)
```

```
## Warning: package 'ggcorrplot' was built under R version 4.3.3
```

```
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 4.3.3
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(VIM)
```

```
## Warning: package 'VIM' was built under R version 4.3.3
## Loading required package: colorspace
## Warning: package 'colorspace' was built under R version 4.3.3
## Loading required package: grid
## VIM is ready to use.
##
## Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues
##
```

```
## Attaching package: 'VIM'
##
## The following object is masked from 'package:datasets':
##
##     sleep
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.3.2
```

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

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.3.3
```

```
library(boot)
```

```
##
## Attaching package: 'boot'
##
## The following object is masked from 'package:lattice':
##
##     melanoma
```

```
library(summarytools)
```

```
## Warning: package 'summarytools' was built under R version 4.3.3
```

```
##
## Attaching package: 'summarytools'
##
## The following object is masked from 'package:tibble':
##
##     view
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.3
```

```
## corrplot 0.94 loaded
```

```
library(car)
```

```
## Warning: package 'car' was built under R version 4.3.3
## Loading required package: carData
## Warning: package 'carData' was built under R version 4.3.2
##
## Attaching package: 'car'
##
## The following object is masked from 'package:boot':
##
##     logit
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
```

```
library(rpart)
```

Small Dataset (Taylor Swift Dataset)

Importing and Processing

```
# Import the data
df_taylor <- read.csv2(url('https://raw.githubusercontent.com/sleepysloth12/DATA_622_HW01/refs/heads/ma
```

```
summary(df_taylor)
```

```
##        ID         track_name        track_musical_genre  track_type
##  Min.   :  0.0   Length:530         Length:530          Length:530
##  1st Qu.:132.2   Class :character   Class :character    Class :character
##  Median :264.5   Mode  :character   Mode  :character    Mode  :character
##  Mean   :264.5
##  3rd Qu.:396.8
##  Max.   :529.0
##   duration_ms       feature          track_videoclip    videoclip_views
##  Min.   : 83253   Length:530         Length:530         Length:530
##  1st Qu.:211813   Class :character   Class :character   Class :character
##  Median :235273   Mode  :character   Mode  :character   Mode  :character
##  Mean   :239979
##  3rd Qu.:260361
##  Max.   :613026
##  spotify_streams    spotify_global_peak    album            track_number
##  Length:530         Min.   :  0.00      Length:530         Min.   : 1.00
##  Class :character   1st Qu.:  0.00      Class :character   1st Qu.: 5.00
##  Mode  :character   Median :  4.00      Mode  :character   Median :10.00
##                     Mean   : 14.81                         Mean   :11.18
##                     3rd Qu.: 19.00                         3rd Qu.:15.00
##                     Max.   :197.00                         Max.   :46.00
##  album_musical_genre  album_type        release_date       album_physical_sales
##  Length:530          Length:530         Length:530         Length:530
```

```
##  Class :character    Class :character   Class :character   Class :character
##  Mode  :character    Mode  :character   Mode  :character   Mode  :character
##
##
##
##  track_lyrics       track_theme           uri            acousticness
##  Length:530         Length:530         Length:530         Length:530
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  danceability          energy          instrumentalness     liveness
##  Length:530         Length:530         Length:530         Length:530
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##    loudness         speechiness           tempo             valence
##  Length:530         Length:530         Length:530         Length:530
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
```

```
df_taylor$videoclip_views <- gsub("\\.", "", df_taylor$videoclip_views)
df_taylor$spotify_streams <- gsub("\\.", "", df_taylor$spotify_streams)
```

```
df_taylor$feature <- ifelse(df_taylor$feature == "No", 0, 1)
df_taylor$track_videoclip <- ifelse(df_taylor$track_videoclip == "No", 0, 1)
df_taylor$total_views <- as.numeric(df_taylor$videoclip_views) + as.numeric(df_taylor$spotify_streams)
df_taylor$release_date <- as.Date(df_taylor$release_date, format = "%d/%m/%Y")
```

```
df_subset <- df_taylor[, !(colnames(df_taylor) %in% c("track_name", "ID", "spotify_global_peak", "album
```

```
numeric_cols <- c("acousticness", "danceability", "energy", "instrumentalness", "liveness", "loudness",
df_subset[numeric_cols] <- lapply(df_subset[numeric_cols], as.numeric)
```

```
## Warning in lapply(df_subset[numeric_cols], as.numeric): NAs introduced by
## coercion
```

EDA

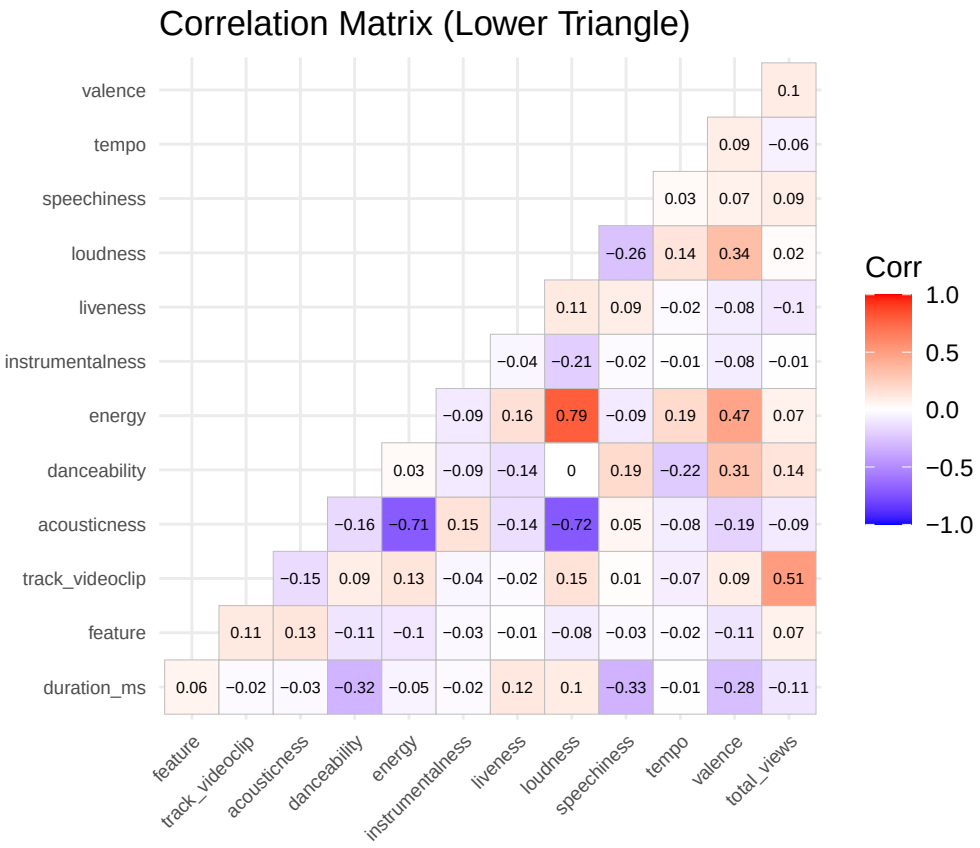
```
summary(df_subset)
```

```
##  track_musical_genre  track_type          duration_ms        feature
##  Length:530          Length:530         Min.   : 83253   Min.   :0.00000
##  Class :character    Class :character   1st Qu.:211813   1st Qu.:0.00000
##  Mode  :character    Mode  :character   Median :235273   Median :0.00000
##                                         Mean   :239979   Mean   :0.06038
##                                         3rd Qu.:260361   3rd Qu.:0.00000
##                                         Max.   :613026   Max.   :1.00000
##
```

```
##  track_videoclip    release_date        track_theme         acousticness
##  Min.   :0.00000   Min.   :2006-10-24   Length:530         Min.   :0.000184
##  1st Qu.:0.00000   1st Qu.:2012-10-22   Class :character   1st Qu.:0.036250
##  Median :0.00000   Median :2020-07-24   Mode  :character   Median :0.165000
##  Mean   :0.09057   Mean   :2017-12-04                      Mean   :0.319247
##  3rd Qu.:0.00000   3rd Qu.:2021-11-12                      3rd Qu.:0.653000
##  Max.   :1.00000   Max.   :2023-10-27                      Max.   :0.971000
##
##   danceability        energy       instrumentalness       liveness
##  Min.   :0.2430   Min.   :0.1180   Min.   :0.0000000   Min.   :0.0357
##  1st Qu.:0.5160   1st Qu.:0.4430   1st Qu.:0.0000000   1st Qu.:0.0966
##  Median :0.5955   Median :0.5895   Median :0.0000020   Median :0.1150
##  Mean   :0.5853   Mean   :0.5746   Mean   :0.0040130   Mean   :0.1635
##  3rd Qu.:0.6530   3rd Qu.:0.7298   3rd Qu.:0.0000559   3rd Qu.:0.1630
##  Max.   :0.8970   Max.   :0.9500   Max.   :0.4880000   Max.   :0.9310
##                                    NA's   :1
##     loudness        speechiness          tempo           valence
##  Min.   :-17.932   Min.   :0.02310   Min.   : 68.10   Min.   :0.0374
##  1st Qu.: -9.222   1st Qu.:0.03033   1st Qu.: 96.94   1st Qu.:0.2300
##  Median : -7.012   Median :0.03730   Median :119.03   Median :0.3855
##  Mean   : -7.505   Mean   :0.05589   Mean   :122.33   Mean   :0.3974
##  3rd Qu.: -5.362   3rd Qu.:0.05523   3rd Qu.:143.93   3rd Qu.:0.5350
##  Max.   : -1.909   Max.   :0.91200   Max.   :208.92   Max.   :0.9430
##
##   total_views
##  Min.   :5.257e+05
##  1st Qu.:6.475e+07
##  Median :1.615e+08
##  Mean   :2.966e+08
##  3rd Qu.:3.283e+08
##  Max.   :5.184e+09
##
```

```r
numeric_cols <- sapply(df_subset, is.numeric)
numeric_data <- df_subset[, numeric_cols]
```

```r
correlation_matrix <- cor(numeric_data, use = "pairwise.complete.obs")

ggcorrplot(correlation_matrix,
           type = "lower",
           lab = TRUE,
           lab_size = 2,
           colors = c("blue", "white", "red"),
           title = "Correlation Matrix (Lower Triangle)",
           ggtheme = theme_minimal(),
           tl.cex = 7
) +
# Adding theme for axis label size
theme(axis.text.x = element_text(size = 7),
      axis.text.y = element_text(size = 7))
```




```
numeric_data <- df_subset[, numeric_cols | names(df_subset) == 'release_date']
# Pairs plot

ggpairs(numeric_data,
        title = "Pairs Plot of Selected Variables",
        upper = list(continuous = wrap("cor", size = 2)),
        lower = list(continuous = wrap("points", size = 0.02)),
        diag = list(continuous = wrap("barDiag", fill = "blue")),
        labeller = label_wrap_gen(width = 10)
) +
# Adjust axis text and label size
theme(axis.text.x = element_text(size = 3),
      axis.text.y = element_text(size = 3),
      strip.text = element_text(size = 3))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
```

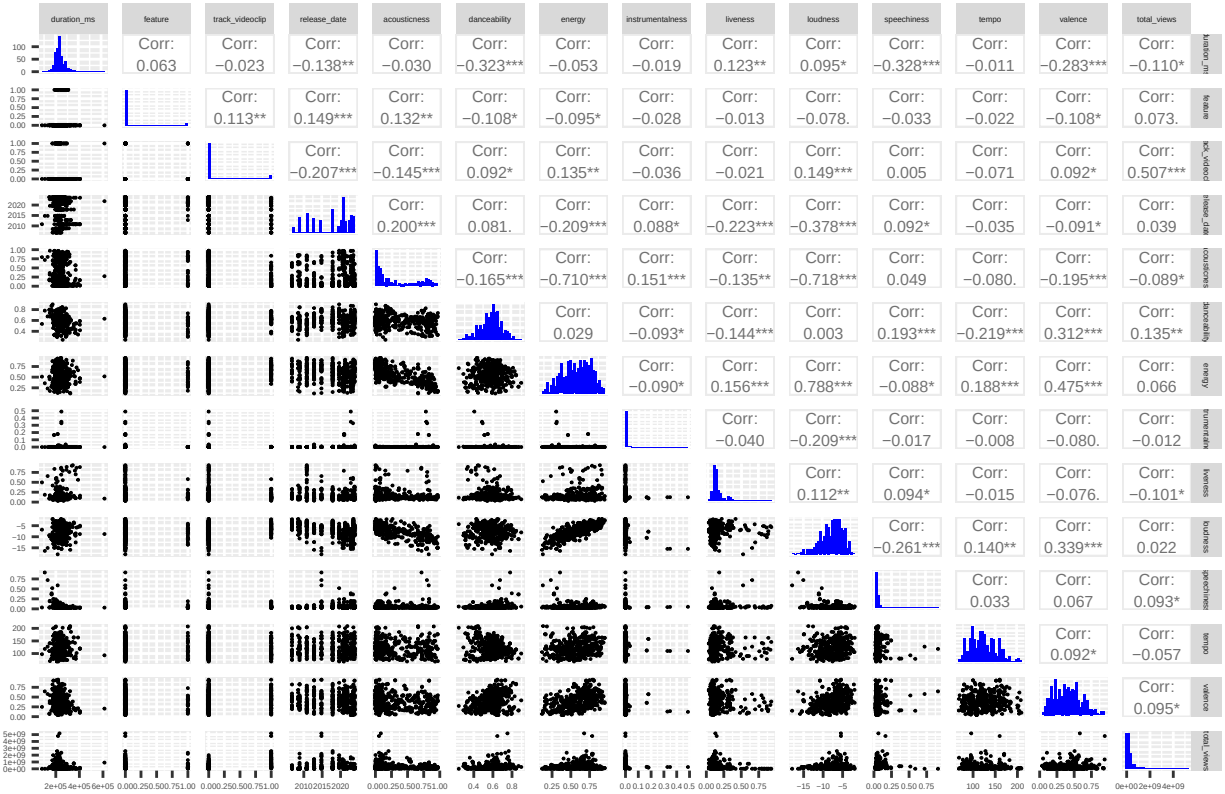
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value

## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
## Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 1 row containing non-finite outside the scale range
## (`stat_bin()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value

## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

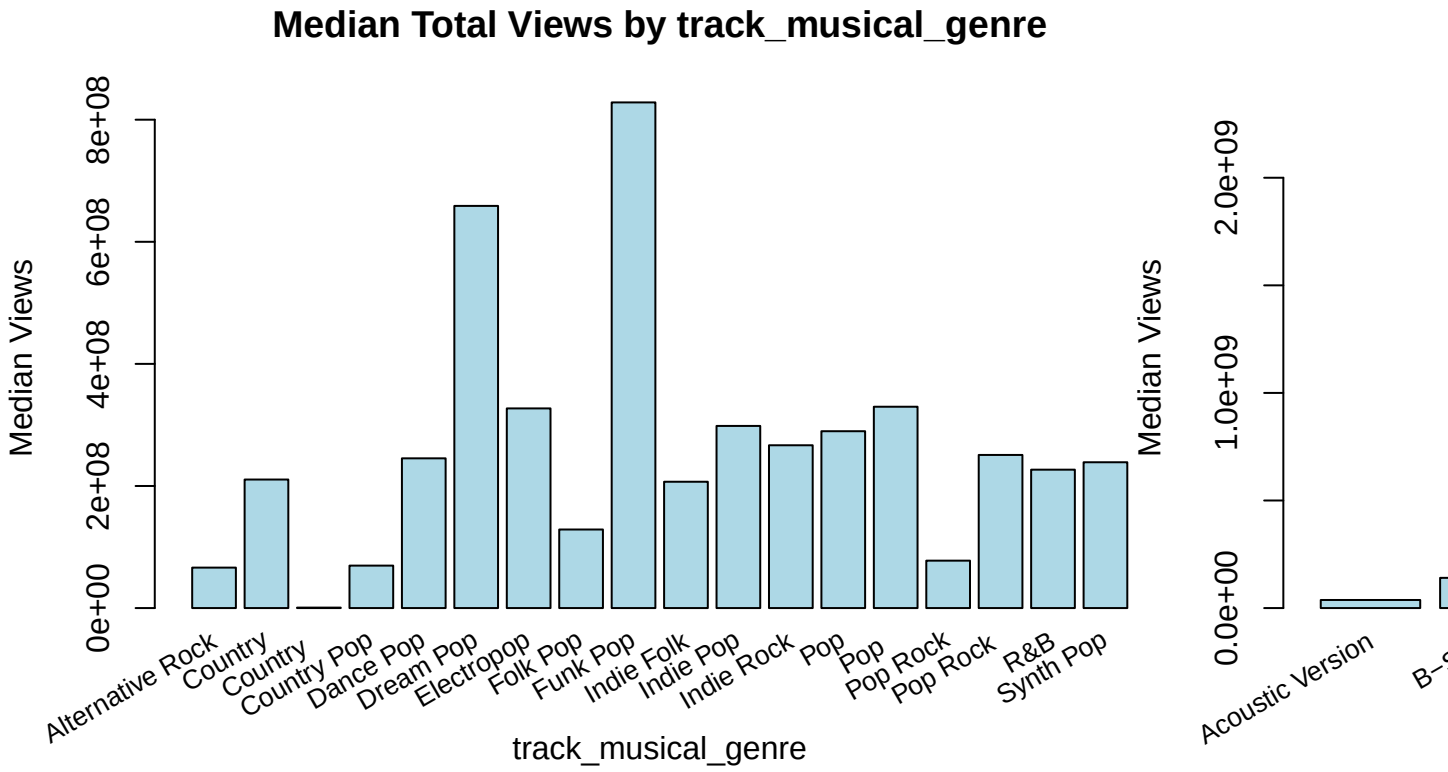
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

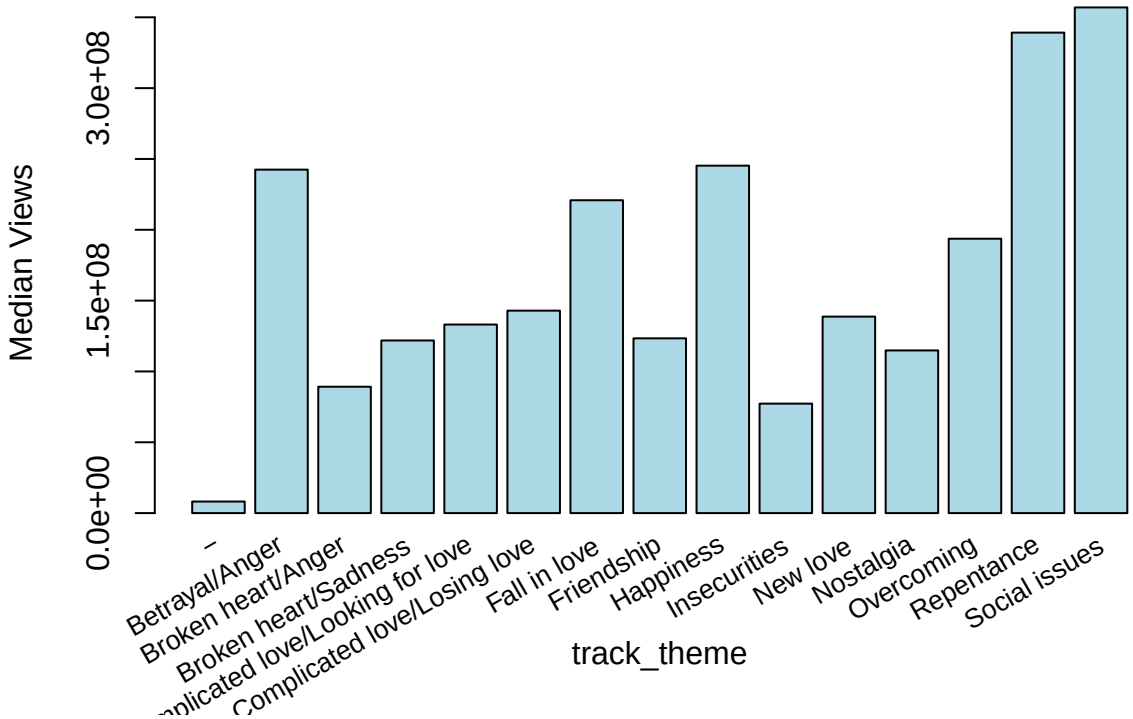
Pairs Plot of Selected Variables



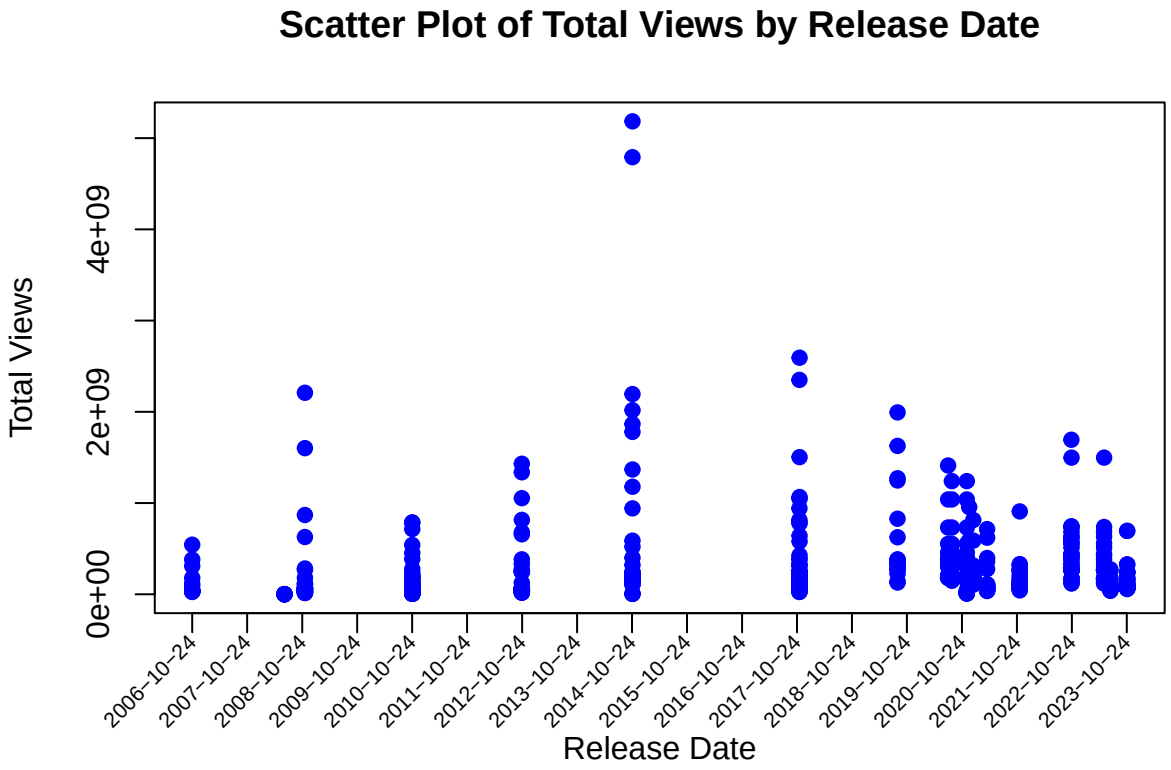

```
non_numeric_cols <- sapply(df_subset, function(col) !is.numeric(col) && !inherits(col, 'Date'))

for (col_name in names(df_subset)[non_numeric_cols]) {
  temp_table <- aggregate(total_views ~ df_subset[[col_name]], data = df_subset, FUN = median)
  colnames(temp_table) <- c("Category", "TotalViews")

  bar_positions <- barplot(temp_table$TotalViews,
                           names.arg = rep("", length(temp_table$Category)),
                           main = paste("Median Total Views by", col_name),
```

```
                          xlab = col_name,
                          ylab = "Median Views",
                          col = 'lightblue')

  text(x = bar_positions,
       y = par("usr")[3] - 0.05 * max(temp_table$TotalViews),
       labels = temp_table$Category,
       srt = 30,
       adj = 1,
       xpd = TRUE,
       cex = 0.8)
}
```






```r
plot(jitter(as.numeric(df_subset$release_date)),
     df_subset$total_views,
     xlab = "Release Date",
     ylab = "Total Views",
     main = "Scatter Plot of Total Views by Release Date",
     pch = 19,
     col = 'blue',
     xaxt = 'n')

date_breaks <- seq(min(df_subset$release_date), max(df_subset$release_date), by = "1 year")
axis(1, at = as.numeric(date_breaks), labels = FALSE)

text(x = as.numeric(date_breaks),
     y = par("usr")[3] - 0.05 * (par("usr")[4] - par("usr")[3]),
     labels = format(date_breaks, "%Y-%m-%d"),
     srt = 45,
     adj = 1,
     xpd = TRUE,
     cex = 0.7)
```



Large Datset (Diabetes Data)

For our large dataset, we will use the diabetes dataset from kaggle.

This dataset has 100k clinical records of diabetes for health analytic purposes.

Link to Dataset

Goal: For this dataset, we want to predict whether or not the patient will have diabetes.

Importing:

```
diabetes_data=read.csv(url("https://raw.githubusercontent.com/sleepysloth12/DATA_622_HW01/refs/heads/ma
```

Exploratory Data Analysis

First, we will start off by looking at each column/variable and seeing its distribution/ summary statistics.

```
names(diabetes_data)
```

```
##  [1] "year"                "gender"              "age"
##  [4] "location"            "race.AfricanAmerican" "race.Asian"
##  [7] "race.Caucasian"      "race.Hispanic"       "race.Other"
## [10] "hypertension"        "heart_disease"       "smoking_history"
## [13] "bmi"                 "hbA1c_level"         "blood_glucose_level"
## [16] "diabetes"
```

```
print(dfSummary(diabetes_data), method = "browser")
```

```
## Output file written: C:\Users\bleac\AppData\Local\Temp\RtmpGCrmf2\file699caf43e02.html
```

The `dfsummary()` function prints out the summary statistics, data type, and distribution of each column.

The data set has no missing data. There are 100k rows and 16 columns.

The column of interest, labeled `diabetes` is what we want to predict. It is an integer, 0 or 1, indicating if the patient has diabetes or not. In the current dataset, 91% of the patients have no diabetes and 8.5% of the patients have diabetes.

In order to build a predictive model, we must first go column by column and clean up the features a little bit to make this more accurate/applicable to healthcare data.

Data Cleaning

Year The first column is year. The dataset is timeseries data, collected from the years 2015-2022. However, each year has different numbers of observations. There is no way of knowing if this is longtitudnal data (one patient visited multiple year) due to the lack of unique patient identifier field. I think we can completely disregard and forget about this column.

```
diabetes_data = diabetes_data %>%
  select(-year)
```

Gender Next is gender. Gender is pretty even split, with ~60% being female and ~40% being male. There is an insignificant amount of people that answered "other" (less than 1%).

I'm going ahead and going to filter out other. Also, I am going to change the label to`is_female` so the choice is binary.

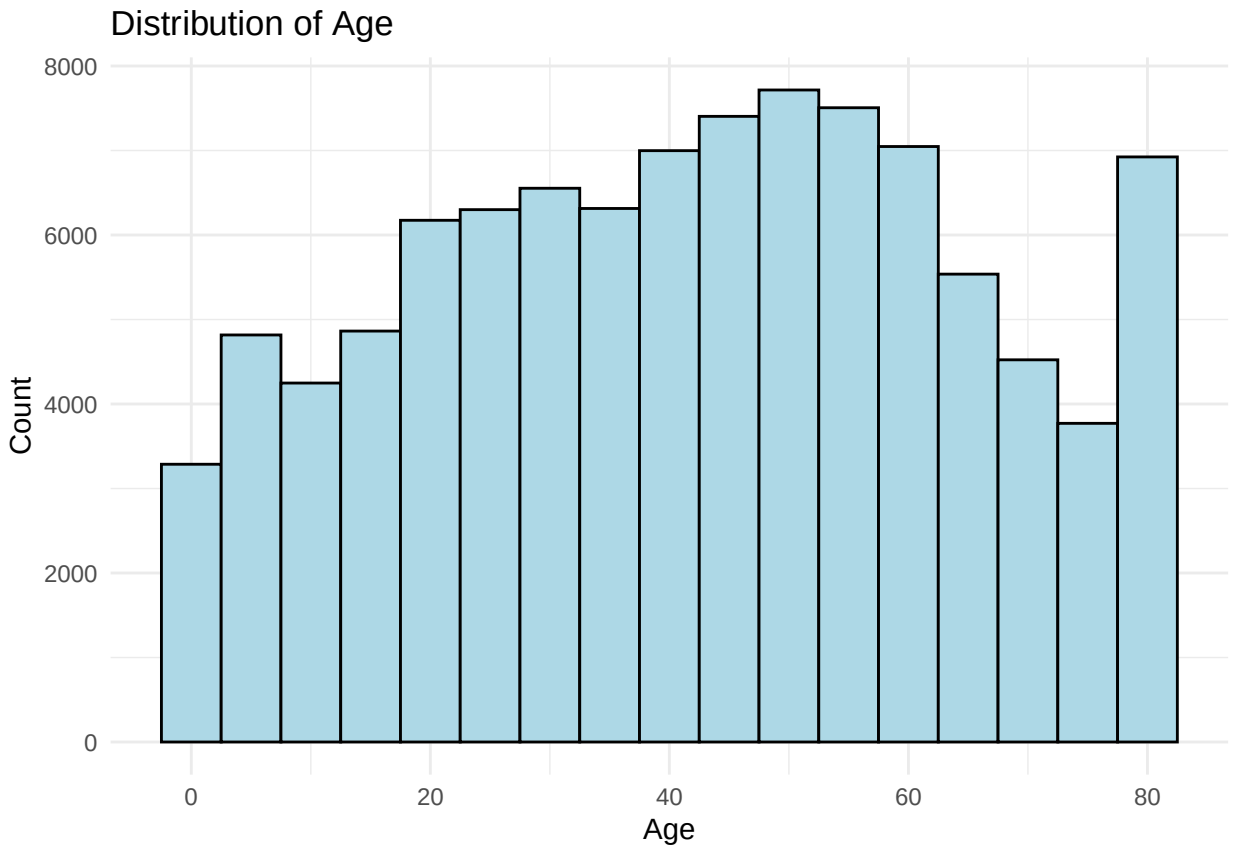
```
diabetes_data = diabetes_data %>%
  filter(gender == "Female"| gender== "Male")%>%
  mutate(is_female=ifelse(gender == "Female",1,0))%>%
  select(-gender)
```

Age Next is age. Mean age is 41.9 years old, with a standard deviation of +/-22.5 years old.

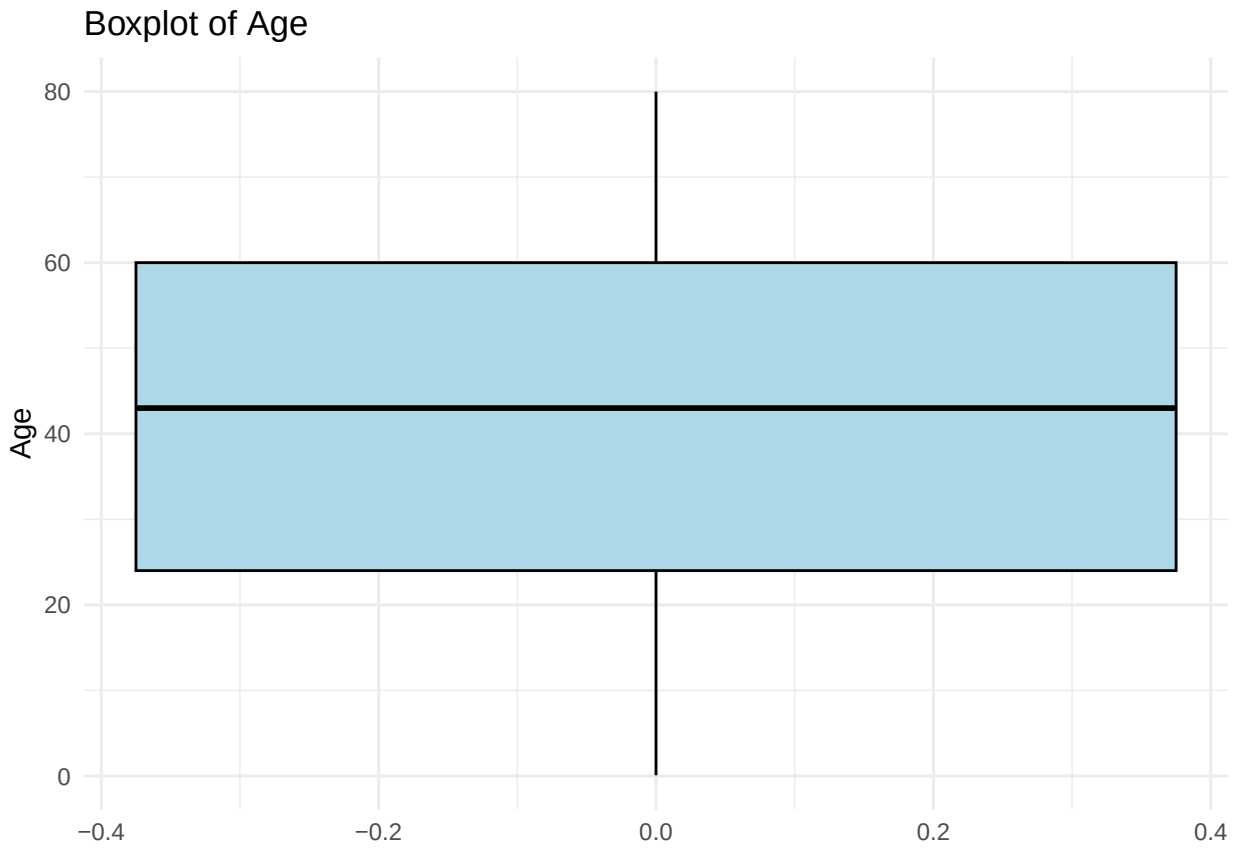
Max age is 80.

Minimum recorded age is 0.08. This might be an outlier. Therefore, lets visualize this distribution in both box plot and bar plot.

```
ggplot(diabetes_data, aes(x = age)) +
  geom_histogram(binwidth = 5, color = "black", fill = "lightblue") +
  labs(title = "Distribution of Age", x = "Age", y = "Count") +
  theme_minimal()
```




```r
ggplot(diabetes_data, aes(y = age)) +
  geom_boxplot(fill = "lightblue", color = "black") +
  labs(title = "Boxplot of Age", y = "Age") +
  theme_minimal()
```



Seems like the minimum age is an outlier. In medical research, we tend to separate adult populations from pediatric populations so lets go ahead and do that here. Lets only look at 18+.

In terms of the age distribution, it looks relatively normal. Diabetes incidence seem to increase as you get closer to middle age, then decrease. There is a spike at 80 years old.

I am going to bin age/ convert it into different categories:

`is_young` = Age 18-35

`is_middle_age` = Age 36-64

`is_old` = Age 65+

```
diabetes_data = diabetes_data %>%
  filter(age>=18)%>%
  mutate(is_young=ifelse(age>=18 & age <=35, 1,0),
         is_middle_age=ifelse(age>35 & age<65 , 1 , 0),
         is_old=ifelse(age>65,1,0))%>%
  select(-age)
```

```
length(unique(diabetes_data$location))
```

State

```
## [1] 55
```

For the location column, there are 55 different locations, corresponding to the 50 different states and territory.

Location is important for diabetes prediction. Some areas are probably more likely to develop diabetes than others. Like age, I want to create categories and bin them based on the location. Then, will create dummy

variables.

```
diabetes_data = diabetes_data %>%
  mutate(

    is_new_england = if_else(location %in% c("Connecticut", "Maine", "Massachusetts",
                                             "New Hampshire", "Rhode Island", "Vermont"), 1, 0),


    is_south = if_else(location %in% c("Alabama", "Arkansas", "Delaware", "Florida", "Georgia",
                                       "Kentucky", "Louisiana", "Maryland", "Mississippi",
                                       "North Carolina", "Oklahoma", "South Carolina",
                                       "Tennessee", "Texas", "Virginia", "West Virginia"), 1, 0),


    is_midwest = if_else(location %in% c("Illinois", "Indiana", "Iowa", "Kansas", "Michigan",
                                         "Minnesota", "Missouri", "Nebraska", "North Dakota",
                                         "Ohio", "South Dakota", "Wisconsin"), 1, 0),


    is_west = if_else(location %in% c("Alaska", "Arizona", "California", "Colorado", "Hawaii",
                                      "Idaho", "Montana", "Nevada", "New Mexico", "Oregon",
                                      "Utah", "Washington", "Wyoming"), 1, 0),


    is_northeast = if_else(location %in% c("New Jersey", "New York", "Pennsylvania"), 1, 0),


    is_territories = if_else(location %in% c("Guam", "Puerto Rico", "Virgin Islands",
                                             "District of Columbia", "United States"), 1, 0)
  )%>%
  select(-location)
```

Race, Ethnicity, Hypertension, & Heart Disease Race and ethnicity is already binned and with their individual dummy variables. Race and ethnicity are both factors that influence diabetes so will leave these columns untouched.

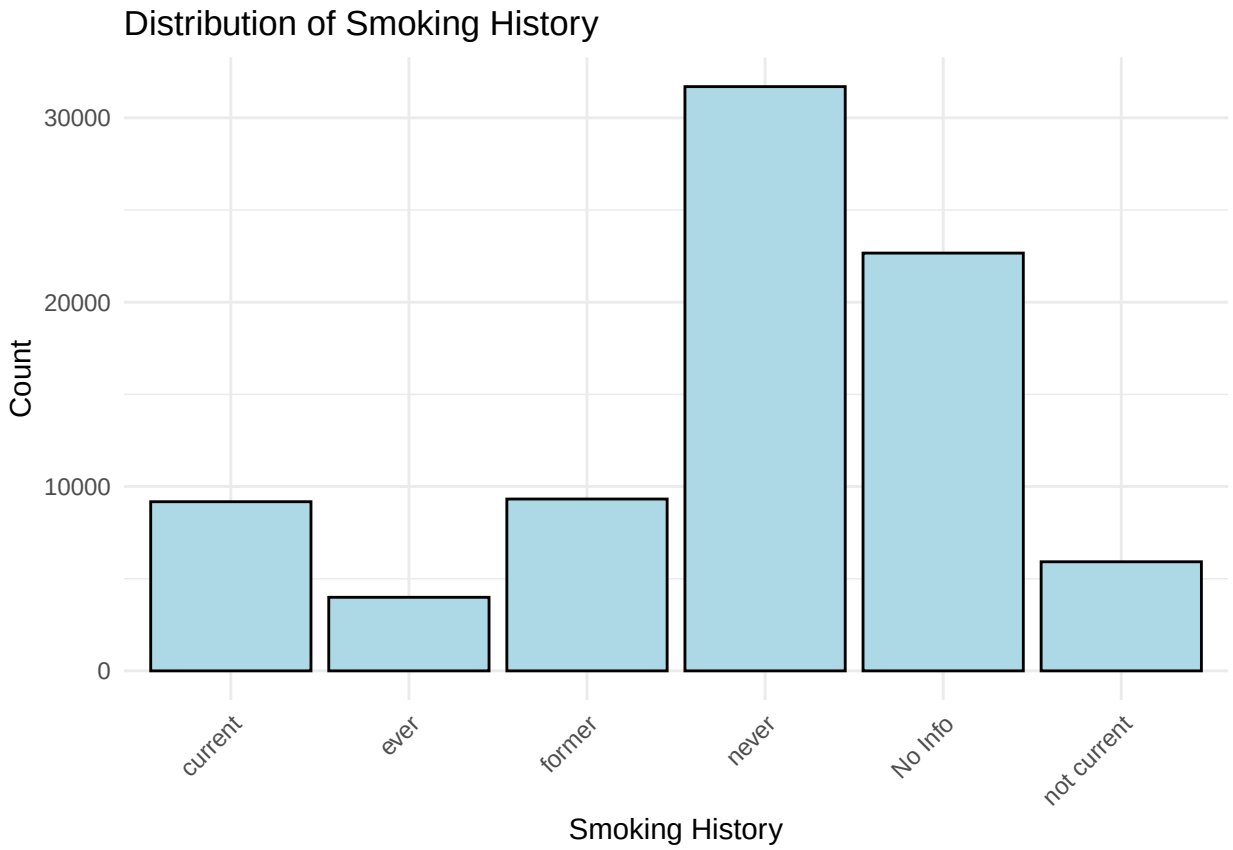
Same with the columns of hypertension and heart disease.

Smoking History There are currently 6 categories/ choices patients could respond when asked about smoking history:

```
unique(diabetes_data$smoking_history)
```

```
## [1] "never"       "not current" "current"     "No Info"     "ever"
## [6] "former"
```

```
ggplot(diabetes_data, aes(x = smoking_history)) +
  geom_bar(fill = "lightblue", color = "black") +
  labs(title = "Distribution of Smoking History", x = "Smoking History", y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The biggest group is Never smoked accounting for 35% of the data.

There is a category, 'ever' which is 'Never' mislabeled. Will fix this. Once combined, never smoked will account for40% of the data.

The second biggest is 'No info' with near 35% of the data. Since the people in 'No info' may or may not be smokers, if we leave this category in it might make our predictions inaccurate. We want to capture how smoking can influence diabetes, therefore we ill remove this group.

Also, the 'not current' and 'former' group can be combined.

```
diabetes_data = diabetes_data %>%
  filter(smoking_history!="No Info")%>%
  mutate(never_smoked=ifelse(smoking_history %in% c("ever","never"),1,0),
         former_smoker=ifelse(smoking_history %in% c("former","not current"),1,0),
         current_smoker=ifelse(smoking_history=="current",1,0)) %>%
  select(-smoking_history)
```

Biomarker Columns The distribution of BMI is normal. It is numeric and continuous. We are leaving this as is.

The hbA1c_level biomarker, although numeric, has 18 unique values. In healthcare, this biomarker is usually used to determine diabetes. We will bin this biomarker for the following categories:

A1c < 5.7% –> Normal A1C

A1c between 5.7-6.4 % –> PreDiabetes

A1C over 6.5% –> diabetes

Although, correlation analysis is needed. There might be multicollinearity between these biomarker variables.

I say this because blood glucose variable and A1c directly related to each other.

Actually going to remove blood glucose because having that and A1C is repetitive/ multicollinearity.

```
diabetes_data = diabetes_data %>%
  mutate(normal_a1c=ifelse(hbA1c_level<5.7,1,0),
         prediabetic_a1c=ifelse(hbA1c_level>=5.7 & hbA1c_level <= 6.4,1,0),
         diabetic_a1c=ifelse(hbA1c_level>6.4,1,0))%>%
  select(-c(hbA1c_level,blood_glucose_level))
```

Model Selection

```
print(dfSummary(diabetes_data), method = "browser")
```

```
## Output file written: C:\Users\bleac\AppData\Local\Temp\RtmpGCrmf2\file699c2ced78bc.html
```

Now that our dataset is clean, we can discuss what model we want to use.

The target variable to predict is diabetes (binary choice whether or not patient will have diabetes).

I think the best algorithm to use in this case is logistic regression. Logistic regression provides interpretable results. The coefficients in the model can be easily interpreted as the change in log-odds of having diabetes for a one-unit change in the predictor, holding other variables constant. This interpretability is important in healthcare.

Correlation Matrix

Before begining the logistic regression model, I want to run a correlation matrix to look for multicolinearity

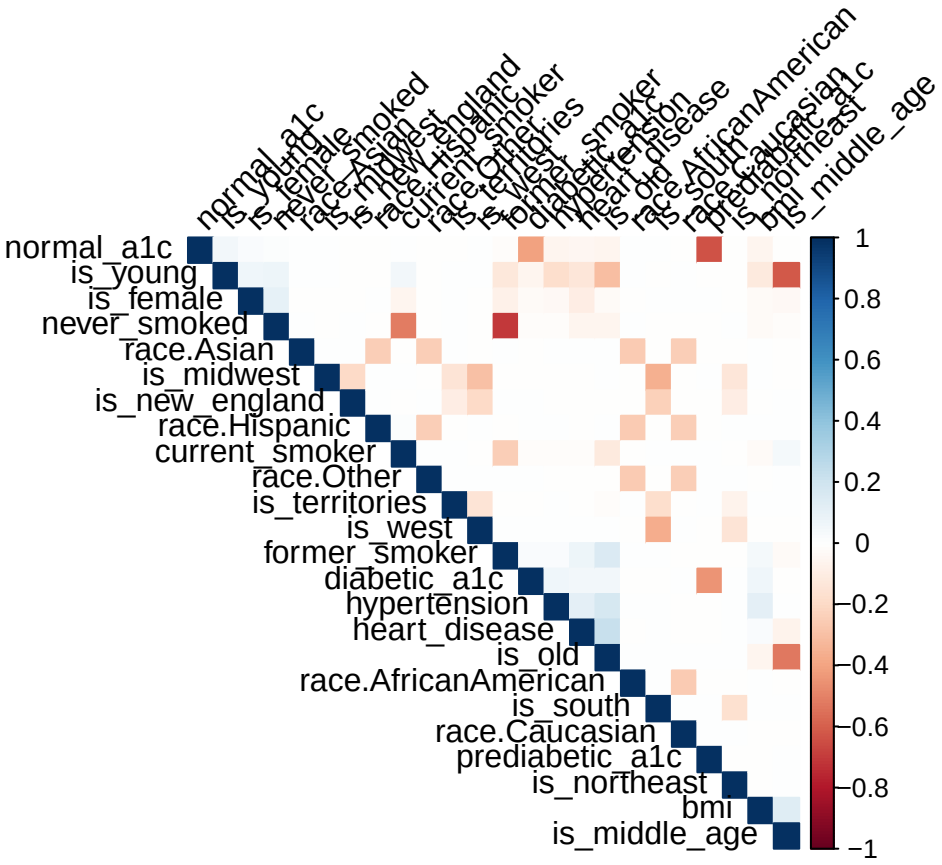
```
predictors <- diabetes_data %>%
  select(-diabetes)

cor_matrix <- cor(predictors)

high_cor <- findCorrelation(cor_matrix, cutoff = 0.7, verbose = TRUE)
```

```
## Compare row 19  and column  20 with corr  0.705
##   Means:  0.07 vs 0.049 so flagging column 19
## All correlations <= 0.7
```

```
cat("Highly correlated variables:", paste(names(predictors)[high_cor], collapse = ", "), "\n")
```

```
## Highly correlated variables: never_smoked
```

```
corrplot(cor_matrix, method = "color", type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```



There is some multicollinearity

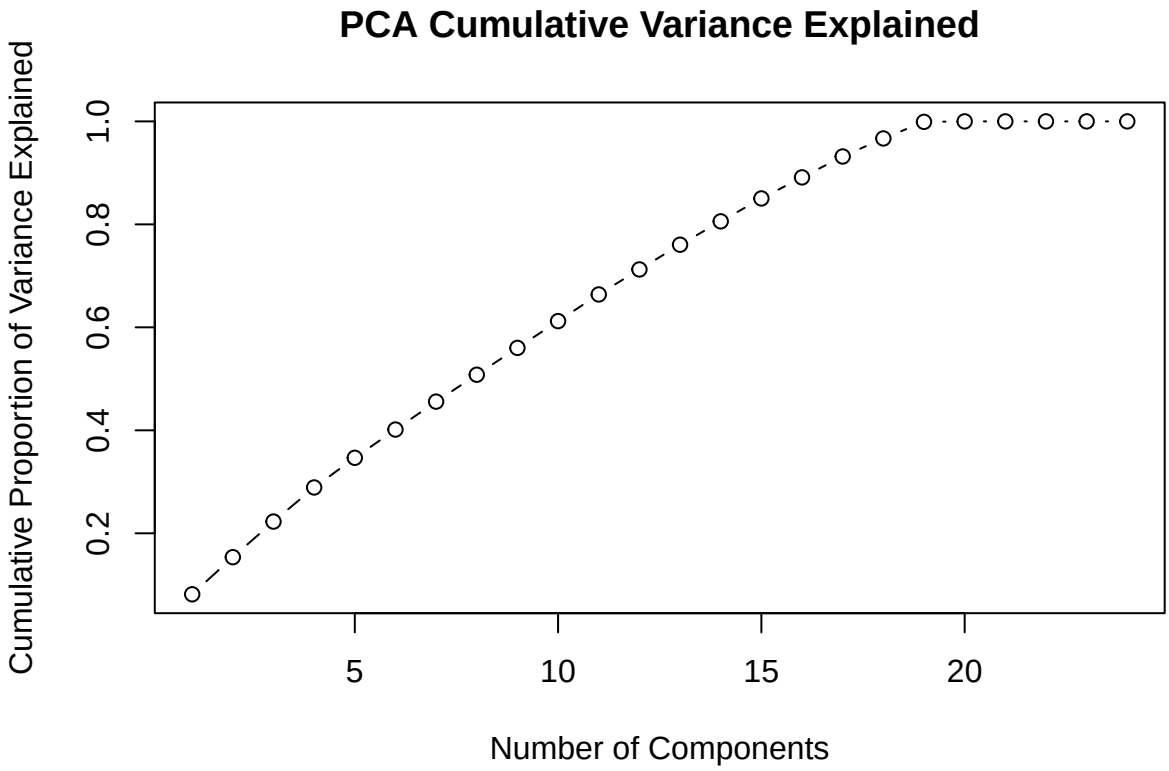
Principal Component Analysis

Conducting a PCA to determine the important components

```
pca_result <- prcomp(predictors, scale. = TRUE)
summary(pca_result)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5    PC6     PC7
## Standard deviation     1.39898 1.3155 1.28787 1.26107 1.17547 1.1478 1.14083
## Proportion of Variance 0.08155 0.0721 0.06911 0.06626 0.05757 0.0549 0.05423
## Cumulative Proportion  0.08155 0.1537 0.22276 0.28902 0.34659 0.4015 0.45572
##                           PC8     PC9    PC10    PC11    PC12    PC13   PC14
## Standard deviation     1.11981 1.11834 1.11722 1.11510 1.07939 1.07351 1.0438
## Proportion of Variance 0.05225 0.05211 0.05201 0.05181 0.04855 0.04802 0.0454
## Cumulative Proportion  0.50797 0.56008 0.61209 0.66390 0.71244 0.76046 0.8059
##                          PC15    PC16    PC17    PC18    PC19    PC20     PC21
## Standard deviation     1.03300 0.99101 0.98754 0.91645 0.87923 0.14869 3.24e-13
## Proportion of Variance 0.04446 0.04092 0.04063 0.03499 0.03221 0.00092 0.00e+00
## Cumulative Proportion  0.85032 0.89124 0.93187 0.96687 0.99908 1.00000 1.00e+00
##                            PC22      PC23     PC24
## Standard deviation     2.767e-14 2.115e-14 1.26e-14
## Proportion of Variance 0.000e+00 0.000e+00 0.00e+00
## Cumulative Proportion  1.000e+00 1.000e+00 1.00e+00
```

```r
plot(cumsum(pca_result$sdev^2 / sum(pca_result$sdev^2)),
     type = "b",
     xlab = "Number of Components",
     ylab = "Cumulative Proportion of Variance Explained",
     main = "PCA Cumulative Variance Explained")
```



Our first principal component only accounts for about 8.16% of the total variance. That's not a lot. It means no single factor dominates in predicting diabetes. This makes sense given the complex nature of the disease and the variety of factors we've included in our dataset.

We need 11 components to explain about 66% of the variance, and it takes 19 to get to nearly 100%. Looking at our cumulative variance plot, we can see this gradual climb. The fact that we need so many components to explain most of the variance suggests we shouldn't try to oversimplify our model. Most of our variables are contributing unique information about diabetes risk.

While we don't see extreme multicollinearity, there is some correlation among our variables. We can explain about 85% of the variance with 15 components, which is fewer than our original variables.

For our logistic regression model, we should probably keep most of our features, as they all seem to contribute meaningful information about diabetes risk. However, we should still be mindful of potential multicollinearity. We might want to consider using regularization techniques like Lasso or Ridge regression in our final model to handle any correlated predictors.

```r
set.seed(622)

train_split_idx=createDataPartition(diabetes_data$diabetes, p=0.7, list=FALSE)

train_diab = diabetes_data[train_split_idx,]
test_diab = diabetes_data[-train_split_idx,]

control <- trainControl(method = "cv", number = 5)
```

```
metric <- "RMSE"
```

```
set.seed(622622)

fit_logistic <- train(diabetes ~ ., data = train_diab, method = "glm", family = "binomial", trControl =
```

```
## Warning in train.default(x, y, weights = w, ...): You are trying to do
## regression and your outcome only has two possible values Are you trying to do
## classification? If so, use a 2 level factor as your outcome column.

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```
fit_logistic$resample %>%
  arrange(Resample)
```

```
##        RMSE  Rsquared       MAE Resample
## 1 0.2693181 0.2837228 0.1456491    Fold1
## 2 0.2709993 0.2898073 0.1457212    Fold2
## 3 0.2754999 0.2783674 0.1496904    Fold3
## 4 0.2748769 0.2821742 0.1501927    Fold4
## 5 0.2660817 0.2818831 0.1431876    Fold5
```

```
fit_logistic$resample %>%
  arrange(Resample) %>%
  summarise(AvgRMSE = mean(RMSE))
```

```
##     AvgRMSE
## 1 0.2713552
```

```
fit_logistic$resample %>%
  arrange(Resample) %>%
  summarise(AvgRsquared = mean(Rsquared))
```

```
##   AvgRsquared
## 1    0.283191
```

```
print(fit_logistic)
```

```
## Generalized Linear Model
##
## 42073 samples
##    24 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 33659, 33659, 33658, 33658, 33658
## Resampling results:
##
##   RMSE       Rsquared   MAE
```

```
##   0.2713552  0.283191  0.1468882
```

```
predictions <- predict(fit_logistic, newdata = test_diab)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```
head(predictions)
```

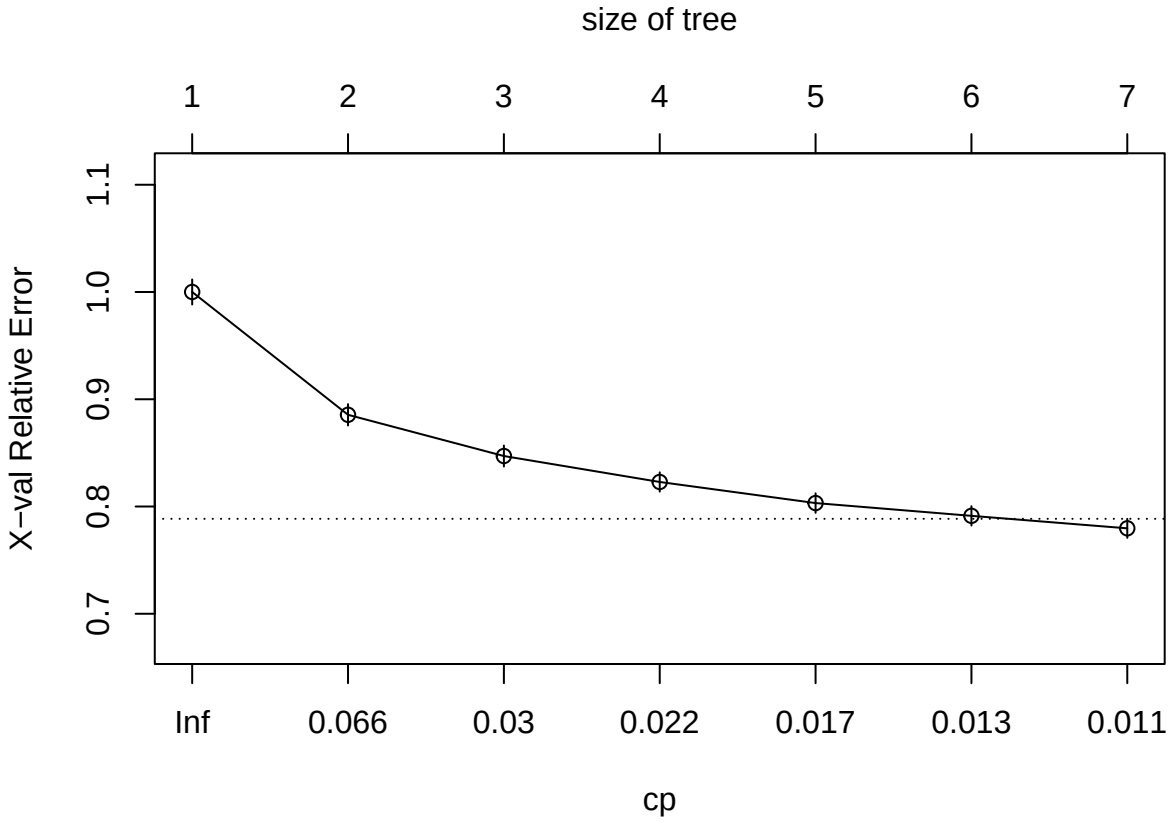
```
##            1            2            9           11           12           14
## 3.790526e-10 2.135277e-10 2.943782e-01 6.103577e-01 1.202814e-02 8.668204e-02
```

```
logistic_rmse <- RMSE(predictions, test_diab$diabetes)
print(logistic_rmse)
```

```
## [1] 0.2700682
```

Comparing with Decision Tree

```
set.seed(622)
fit_tree <- rpart(diabetes ~ ., method = 'anova', data = train_diab)

plotcp(fit_tree)
```




```
printcp(fit_tree)
```

```
##
## Regression tree:
## rpart(formula = diabetes ~ ., data = train_diab, method = "anova")
##
## Variables actually used in tree construction:
## [1] bmi             diabetic_a1c    hypertension    is_middle_age
```

```
## [5] is_young        prediabetic_a1c
##
## Root node error: 4321.7/42073 = 0.10272
##
## n= 42073
##
##         CP nsplit rel error  xerror      xstd
## 1 0.114590      0   1.00000 1.00002 0.0116758
## 2 0.038467      1   0.88541 0.88550 0.0099825
## 3 0.023022      2   0.84694 0.84707 0.0097786
## 4 0.021458      3   0.82392 0.82286 0.0090078
## 5 0.013669      4   0.80246 0.80320 0.0090972
## 6 0.013126      5   0.78879 0.79128 0.0090426
## 7 0.010000      6   0.77567 0.77966 0.0088788
```

```
summary(fit_tree)
```

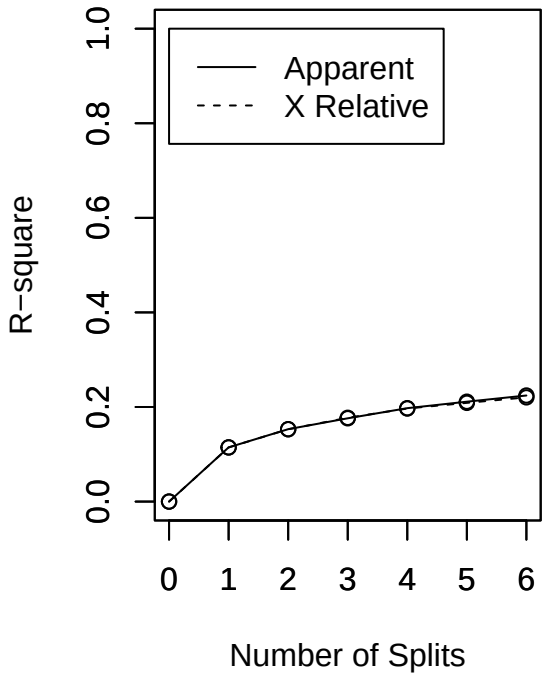
```
## Call:
## rpart(formula = diabetes ~ ., data = train_diab, method = "anova")
##   n= 42073
##
##           CP nsplit rel error    xerror        xstd
## 1 0.11458965      0 1.0000000 1.0000194 0.011675840
## 2 0.03846703      1 0.8854104 0.8854953 0.009982453
## 3 0.02302216      2 0.8469433 0.8470656 0.009778577
## 4 0.02145836      3 0.8239212 0.8228629 0.009007763
## 5 0.01366882      4 0.8024628 0.8032033 0.009097167
## 6 0.01312625      5 0.7887940 0.7912810 0.009042644
## 7 0.01000000      6 0.7756677 0.7796614 0.008878827
##
## Variable importance
##    diabetic_a1c        is_young      normal_a1c prediabetic_a1c             bmi
##              44              15               9               9               8
##    hypertension   is_middle_age          is_old   heart_disease
##               5               5               5               1
##
## Node number 1: 42073 observations,    complexity param=0.1145896
##   mean=0.1162266, MSE=0.1027179
##   left son=2 (32794 obs) right son=3 (9279 obs)
##   Primary splits:
##       diabetic_a1c < 0.5    to the left,  improve=0.11458960, (0 missing)
##       normal_a1c   < 0.5    to the right, improve=0.07597665, (0 missing)
##       hypertension < 0.5    to the left,  improve=0.03496522, (0 missing)
##       is_young     < 0.5    to the right, improve=0.03374830, (0 missing)
##       is_old       < 0.5    to the left,  improve=0.03360309, (0 missing)
##   Surrogate splits:
##       bmi < 70.255 to the left,  agree=0.78, adj=0.001, (0 split)
##
## Node number 2: 32794 observations,    complexity param=0.02302216
##   mean=0.0585168, MSE=0.05509259
##   left son=4 (15406 obs) right son=5 (17388 obs)
##   Primary splits:
##       prediabetic_a1c < 0.5    to the left,  improve=0.05506914, (0 missing)
##       normal_a1c      < 0.5    to the right, improve=0.05506914, (0 missing)
##       hypertension    < 0.5    to the left,  improve=0.02129604, (0 missing)
```

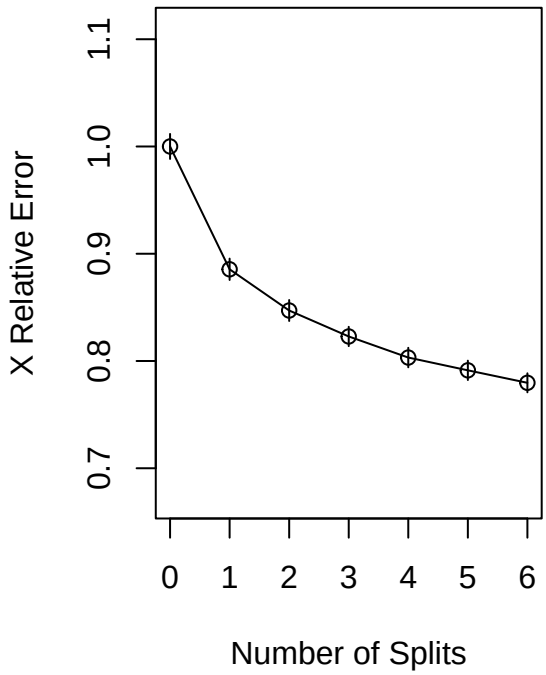
```
##       is_old          < 0.5    to the left,  improve=0.01978397, (0 missing)
##       is_young        < 0.5    to the right, improve=0.01763096, (0 missing)
##   Surrogate splits:
##       normal_a1c < 0.5    to the right, agree=1.000, adj=1.000, (0 split)
##       is_young   < 0.5    to the right, agree=0.533, adj=0.005, (0 split)
##       bmi        < 18.335 to the left,  agree=0.531, adj=0.001, (0 split)
##
## Node number 3: 9279 observations,    complexity param=0.03846703
##   mean=0.3201854, MSE=0.2176667
##   left son=6 (2019 obs) right son=7 (7260 obs)
##   Primary splits:
##       is_young      < 0.5    to the right, improve=0.08230861, (0 missing)
##       is_old        < 0.5    to the left,  improve=0.06160872, (0 missing)
##       bmi           < 30.595 to the left,  improve=0.06018096, (0 missing)
##       hypertension  < 0.5    to the left,  improve=0.05515676, (0 missing)
##       heart_disease < 0.5    to the left,  improve=0.04193962, (0 missing)
##
## Node number 4: 15406 observations
##   mean=0, MSE=0
##
## Node number 5: 17388 observations,    complexity param=0.01312625
##   mean=0.1103635, MSE=0.09818337
##   left son=10 (15562 obs) right son=11 (1826 obs)
##   Primary splits:
##       hypertension  < 0.5    to the left,  improve=0.03322790, (0 missing)
##       is_old        < 0.5    to the left,  improve=0.03265854, (0 missing)
##       is_young      < 0.5    to the right, improve=0.03129247, (0 missing)
##       bmi           < 31.015 to the left,  improve=0.02949148, (0 missing)
##       heart_disease < 0.5    to the left,  improve=0.02596564, (0 missing)
##
## Node number 6: 2019 observations
##   mean=0.06636949, MSE=0.06196458
##
## Node number 7: 7260 observations,    complexity param=0.02145836
##   mean=0.3907713, MSE=0.2380691
##   left son=14 (4635 obs) right son=15 (2625 obs)
##   Primary splits:
##       bmi           < 30.585 to the left,  improve=0.05365457, (0 missing)
##       hypertension  < 0.5    to the left,  improve=0.03846108, (0 missing)
##       is_middle_age < 0.5    to the right, improve=0.03659048, (0 missing)
##       is_old        < 0.5    to the left,  improve=0.03324120, (0 missing)
##       heart_disease < 0.5    to the left,  improve=0.03132863, (0 missing)
##
## Node number 10: 15562 observations
##   mean=0.0907981, MSE=0.0825538
##
## Node number 11: 1826 observations
##   mean=0.2771084, MSE=0.2003193
##
## Node number 14: 4635 observations,    complexity param=0.01366882
##   mean=0.3057174, MSE=0.2122543
##   left son=28 (2887 obs) right son=29 (1748 obs)
##   Primary splits:
##       is_middle_age < 0.5    to the right, improve=0.06004468, (0 missing)
```

```
##       is_old        < 0.5    to the left,  improve=0.05926324, (0 missing)
##       hypertension  < 0.5    to the left,  improve=0.03571097, (0 missing)
##       heart_disease < 0.5    to the left,  improve=0.03408260, (0 missing)
##       bmi           < 24.925 to the left,  improve=0.01063829, (0 missing)
##   Surrogate splits:
##       is_old        < 0.5    to the left,  agree=0.979, adj=0.943, (0 split)
##       heart_disease < 0.5    to the left,  agree=0.665, adj=0.110, (0 split)
##       hypertension  < 0.5    to the left,  agree=0.644, adj=0.055, (0 split)
##       bmi           < 16.565 to the right, agree=0.624, adj=0.003, (0 split)
##
## Node number 15: 2625 observations
##   mean=0.5409524, MSE=0.2483229
##
## Node number 28: 2887 observations
##   mean=0.2178732, MSE=0.1704045
##
## Node number 29: 1748 observations
##   mean=0.4508009, MSE=0.2475795
```

```
par(mfrow = c(1, 2))
rsq.rpart(fit_tree)
```

```
##
## Regression tree:
## rpart(formula = diabetes ~ ., data = train_diab, method = "anova")
##
## Variables actually used in tree construction:
## [1] bmi             diabetic_a1c    hypertension    is_middle_age
## [5] is_young        prediabetic_a1c
##
## Root node error: 4321.7/42073 = 0.10272
##
## n= 42073
##
##         CP nsplit rel error  xerror      xstd
## 1 0.114590      0   1.00000 1.00002 0.0116758
## 2 0.038467      1   0.88541 0.88550 0.0099825
## 3 0.023022      2   0.84694 0.84707 0.0097786
## 4 0.021458      3   0.82392 0.82286 0.0090078
## 5 0.013669      4   0.80246 0.80320 0.0090972
## 6 0.013126      5   0.78879 0.79128 0.0090426
## 7 0.010000      6   0.77567 0.77966 0.0088788
```






```
predictions_tree <- predict(fit_tree, newdata = test_diab)
head(predictions_tree)
```

```
##         1         2         9        11        12        14
## 0.0000000 0.0000000 0.5409524 0.5409524 0.0907981 0.0907981
```

```
tree_rmse <- RMSE(predictions_tree, test_diab$diabetes)
print(tree_rmse)
```

```
## [1] 0.2835493
```

Models for Both

```
set.seed(226)
split_index <- createDataPartition(diabetes_data$diabetes,p=0.7,list = FALSE)
```

```
training_data <- diabetes_data[split_index,]
testing_data <- diabetes_data[-split_index,]
```

```
set.seed(5)
control <- trainControl(method = "cv",number = 5)
metric = "rmse"
set.seed(1234)
fit.lm <- train(diabetes~.,data = training_data,method = "lm",trControl = control)
```

```
## Warning in train.default(x, y, weights = w, ...): You are trying to do
## regression and your outcome only has two possible values Are you trying to do
## classification? If so, use a 2 level factor as your outcome column.
```

```
fit.lm$resample %>%
  arrange(Resample)
```

```
##        RMSE  Rsquared       MAE Resample
## 1 0.2853504 0.2444014 0.1878085    Fold1
```

```
## 2 0.2807393 0.2355102 0.1853773    Fold2
## 3 0.2845958 0.2225733 0.1871477    Fold3
## 4 0.2754211 0.2215199 0.1828548    Fold4
## 5 0.2797247 0.2273393 0.1847653    Fold5
```

```
fit.lm$resample %>%
  arrange(Resample) %>%
  summarise(AVGRMSE  = mean(RMSE))
```

```
##     AVGRMSE
## 1 0.2811663
```

```
fit.lm$resample %>%
  arrange(Resample) %>%
  summarise(AvgRsquared  = mean(Rsquared))
```

```
##   AvgRsquared
## 1   0.2302688
```

```
print(fit.lm)
```

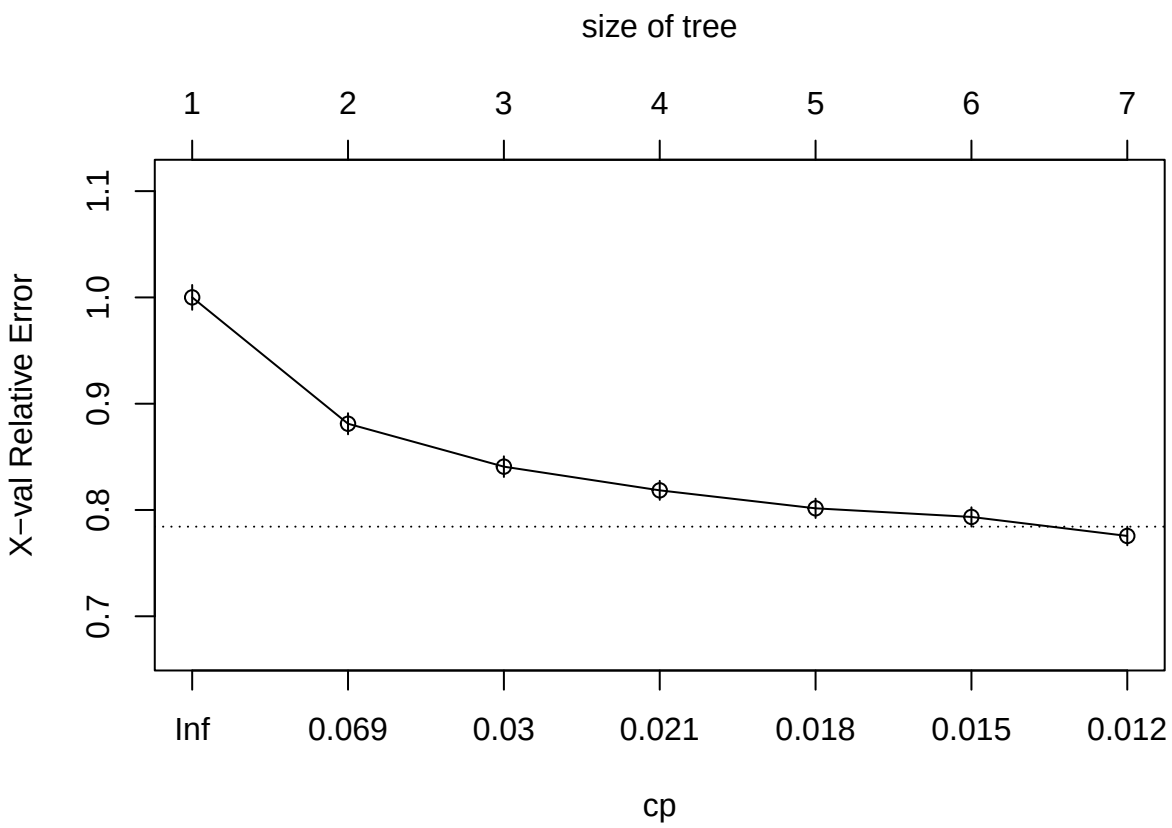
```
## Linear Regression
##
## 42073 samples
##    24 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 33658, 33659, 33658, 33658, 33659
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.2811663  0.2302688  0.1855907
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
predictions <- predict(fit.lm,newdata = testing_data)
head(predictions)
```

```
##           3           5          10          16          17          21
## -0.09185341  0.23636173  0.26215569  0.12534754  0.01689994 -0.06838148
```

```
library(rpart)
set.seed(3456)
fit <- rpart(diabetes~.,,method = 'anova',data = training_data)
```

```
plotcp(fit)
```




```
printcp(fit)
```

```
##
## Regression tree:
## rpart(formula = diabetes ~ ., data = training_data, method = "anova")
##
## Variables actually used in tree construction:
## [1] bmi             diabetic_a1c    is_middle_age   is_old
## [5] is_young        prediabetic_a1c
##
## Root node error: 4320.9/42073 = 0.1027
##
## n= 42073
##
##         CP nsplit rel error  xerror      xstd
## 1 0.118980      0   1.00000 1.00004 0.0116778
## 2 0.040373      1   0.88102 0.88114 0.0099260
## 3 0.022108      2   0.84065 0.84079 0.0097225
## 4 0.020248      3   0.81854 0.81847 0.0089506
## 5 0.015360      4   0.79829 0.80156 0.0090751
## 6 0.014636      5   0.78293 0.79347 0.0090532
## 7 0.010000      6   0.76829 0.77556 0.0087717
```

```
summary(fit)
```

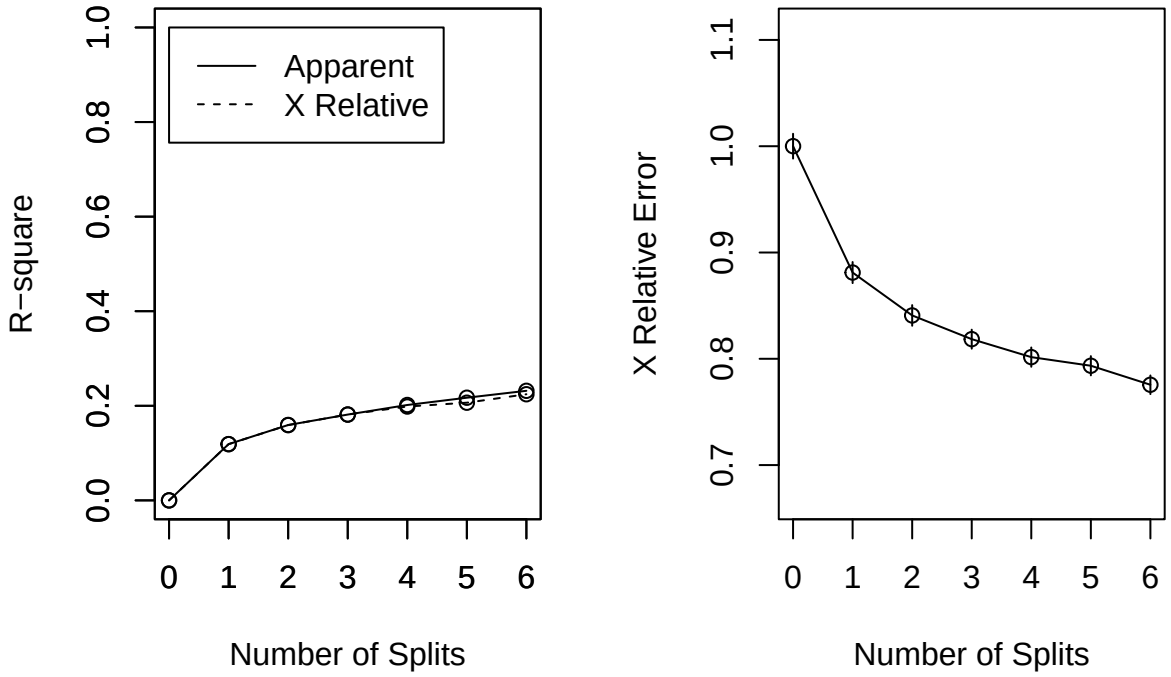
```
## Call:
## rpart(formula = diabetes ~ ., data = training_data, method = "anova")
##   n= 42073
##
```

```
##           CP nsplit rel error    xerror        xstd
## 1 0.11898044      0 1.0000000 1.0000406 0.011677840
## 2 0.04037292      1 0.8810196 0.8811440 0.009926011
## 3 0.02210788      2 0.8406466 0.8407919 0.009722525
## 4 0.02024780      3 0.8185388 0.8184707 0.008950646
## 5 0.01536002      4 0.7982910 0.8015635 0.009075059
## 6 0.01463623      5 0.7829309 0.7934664 0.009053191
## 7 0.01000000      6 0.7682947 0.7755650 0.008771679
##
## Variable importance
##    diabetic_a1c        is_young          is_old      normal_a1c prediabetic_a1c
##              44              15              11               8               8
##             bmi   is_middle_age   heart_disease
##               7               6               1
##
## Node number 1: 42073 observations,    complexity param=0.1189804
##   mean=0.1162028, MSE=0.1026997
##   left son=2 (32802 obs) right son=3 (9271 obs)
##   Primary splits:
##       diabetic_a1c < 0.5    to the left,  improve=0.11898040, (0 missing)
##       normal_a1c   < 0.5    to the right, improve=0.07575702, (0 missing)
##       is_old       < 0.5    to the left,  improve=0.03651726, (0 missing)
##       hypertension < 0.5    to the left,  improve=0.03567110, (0 missing)
##       is_young     < 0.5    to the right, improve=0.03437650, (0 missing)
##   Surrogate splits:
##       bmi < 68.54  to the left,  agree=0.78, adj=0, (0 split)
##
## Node number 2: 32802 observations,    complexity param=0.02210788
##   mean=0.05743552, MSE=0.05413668
##   left son=4 (15380 obs) right son=5 (17422 obs)
##   Primary splits:
##       prediabetic_a1c < 0.5    to the left,  improve=0.05379326, (0 missing)
##       normal_a1c      < 0.5    to the right, improve=0.05379326, (0 missing)
##       is_old          < 0.5    to the left,  improve=0.02217664, (0 missing)
##       hypertension    < 0.5    to the left,  improve=0.01938783, (0 missing)
##       bmi             < 30.705 to the left,  improve=0.01780461, (0 missing)
##   Surrogate splits:
##       normal_a1c < 0.5    to the right, agree=1.000, adj=1.000, (0 split)
##       bmi        < 16.95  to the left,  agree=0.532, adj=0.001, (0 split)
##
## Node number 3: 9271 observations,    complexity param=0.04037292
##   mean=0.324129, MSE=0.2190694
##   left son=6 (2047 obs) right son=7 (7224 obs)
##   Primary splits:
##       is_young      < 0.5    to the right, improve=0.08589237, (0 missing)
##       is_old        < 0.5    to the left,  improve=0.06661600, (0 missing)
##       hypertension  < 0.5    to the left,  improve=0.05901382, (0 missing)
##       bmi           < 30.585 to the left,  improve=0.05851540, (0 missing)
##       heart_disease < 0.5    to the left,  improve=0.04290573, (0 missing)
##
## Node number 4: 15380 observations
##   mean=0, MSE=0
##
## Node number 5: 17422 observations,    complexity param=0.01463623
```

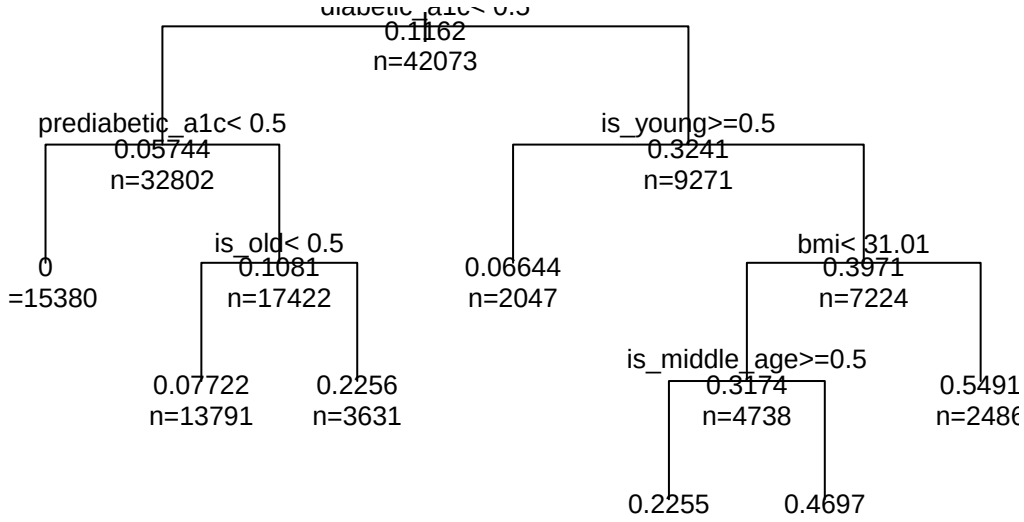
```
##   mean=0.1081391, MSE=0.09644506
##   left son=10 (13791 obs) right son=11 (3631 obs)
##   Primary splits:
##       is_old        < 0.5    to the left,  improve=0.03763778, (0 missing)
##       is_young      < 0.5    to the right, improve=0.03191637, (0 missing)
##       bmi           < 30.705 to the left,  improve=0.03134441, (0 missing)
##       hypertension  < 0.5    to the left,  improve=0.03083319, (0 missing)
##       heart_disease < 0.5    to the left,  improve=0.02683818, (0 missing)
##   Surrogate splits:
##       heart_disease < 0.5    to the left,  agree=0.803, adj=0.056, (0 split)
##
## Node number 6: 2047 observations
##   mean=0.06643869, MSE=0.06202459
##
## Node number 7: 7224 observations,    complexity param=0.0202478
##   mean=0.3971484, MSE=0.2394215
##   left son=14 (4738 obs) right son=15 (2486 obs)
##   Primary splits:
##       bmi           < 31.005 to the left,  improve=0.05058357, (0 missing)
##       is_middle_age < 0.5    to the right, improve=0.04292705, (0 missing)
##       hypertension  < 0.5    to the left,  improve=0.04047586, (0 missing)
##       is_old        < 0.5    to the left,  improve=0.03645565, (0 missing)
##       heart_disease < 0.5    to the left,  improve=0.03203931, (0 missing)
##
## Node number 10: 13791 observations
##   mean=0.07722428, MSE=0.07126069
##
## Node number 11: 3631 observations
##   mean=0.2255577, MSE=0.1746814
##
## Node number 14: 4738 observations,    complexity param=0.01536002
##   mean=0.3174335, MSE=0.2166695
##   left son=28 (2954 obs) right son=29 (1784 obs)
##   Primary splits:
##       is_middle_age < 0.5    to the right, improve=0.06465048, (0 missing)
##       is_old        < 0.5    to the left,  improve=0.06010342, (0 missing)
##       hypertension  < 0.5    to the left,  improve=0.03993358, (0 missing)
##       heart_disease < 0.5    to the left,  improve=0.03651594, (0 missing)
##       bmi           < 26.745 to the left,  improve=0.01324741, (0 missing)
##   Surrogate splits:
##       is_old        < 0.5    to the left,  agree=0.979, adj=0.943, (0 split)
##       heart_disease < 0.5    to the left,  agree=0.666, adj=0.113, (0 split)
##       hypertension  < 0.5    to the left,  agree=0.650, adj=0.070, (0 split)
##       bmi           < 16.585 to the right, agree=0.624, adj=0.002, (0 split)
##
## Node number 15: 2486 observations
##   mean=0.5490748, MSE=0.2475917
##
## Node number 28: 2954 observations
##   mean=0.225457, MSE=0.1746261
##
## Node number 29: 1784 observations
##   mean=0.4697309, MSE=0.2490838
```

```r
par(mfrow=c(1,2)) # two plots on one page
rsq.rpart(fit)
```

```
##
## Regression tree:
## rpart(formula = diabetes ~ ., data = training_data, method = "anova")
##
## Variables actually used in tree construction:
## [1] bmi             diabetic_a1c    is_middle_age   is_old
## [5] is_young        prediabetic_a1c
##
## Root node error: 4320.9/42073 = 0.1027
##
## n= 42073
##
##         CP nsplit rel error  xerror      xstd
## 1 0.118980      0   1.00000 1.00004 0.0116778
## 2 0.040373      1   0.88102 0.88114 0.0099260
## 3 0.022108      2   0.84065 0.84079 0.0097225
## 4 0.020248      3   0.81854 0.81847 0.0089506
## 5 0.015360      4   0.79829 0.80156 0.0090751
## 6 0.014636      5   0.78293 0.79347 0.0090532
## 7 0.010000      6   0.76829 0.77556 0.0087717
```




```r
# create attractive postcript plot of tree
plot(fit, uniform=TRUE,
   main="Regression Tree for Mileage ")
text(fit, use.n=TRUE, all=TRUE, cex=.8)
```




```
post(fit, title = "Regression Tree for diabetes ")
# Create predictions using the decision tree..

predictions1 <- predict(fit,newdata = testing_data)
head(predictions1)
```

```
##          3          5         10         16         17         21
## 0.00000000 0.22545701 0.22545701 0.22555770 0.07722428 0.00000000
```

```
RMSE(predictions1,testing_data$diabetes)
```

```
## [1] 0.2849052
```