Pima Indians | Predicting Presence of Diabetes

Antonio Pano Flores

9/22/2022

Variables Used:

Pregnancy: Number of times pregnant.
Glucose: Oral Glucose Tolerance Test.
Blood Pressure: Diastolic blood pressure (mm Hg)
Skin Thickness: Triceps skin fold thickness (mm)
Insulin: 2-Hour serum insulin (mu U/ml)
BMI: Body mass index (weight in kg/(height in m)^2)
DiabetesPedigreeFunction: Diabetes pedigree function (function which scores likelihood of diabetes based on family history).
Age: Years of Age.
Outcome: 0 = Does Not Have Diabetes, 1 = Does Have Diabetes

Preparing the Data for Use:

1. Adjusting Outcome Integer into a Factor.
2. Replacing 0's with NA values within the `Insulin` , `BMI` , and `SkinThickness` fields.
3. Creating a 'GlucoseGroup' variable based on labels set by the American Diabetes Association (https://diabetes.org/diabetes/a1c/diagnosis)
4. Creating a new variable 'WeightGroup' based on BMI value. Using bins set my the American Cancer Society (https://www.cancer.org/healthy/cancer-causes/diet-physical-activity/body-weight-and-cancer-risk/adult-bmi.html).

```
glimpse(PI)
```

```
## Rows: 768
## Columns: 9
## $ Pregnancies              <int> 6, 1, 8, 1, 0, 5, 3, 10, 2, 8, 4, 10, 10, 1, ~
## $ Glucose                  <int> 148, 85, 183, 89, 137, 116, 78, 115, 197, 125~
## $ BloodPressure            <int> 72, 66, 64, 66, 40, 74, 50, 0, 70, 96, 92, 74~
## $ SkinThickness            <int> 35, 29, 0, 23, 35, 0, 32, 0, 45, 0, 0, 0, 0, ~
## $ Insulin                  <int> 0, 0, 0, 94, 168, 0, 88, 0, 543, 0, 0, 0, 0, ~
## $ BMI                      <dbl> 33.6, 26.6, 23.3, 28.1, 43.1, 25.6, 31.0, 35.~
## $ DiabetesPedigreeFunction <dbl> 0.627, 0.351, 0.672, 0.167, 2.288, 0.201, 0.2~
## $ Age                      <int> 50, 31, 32, 21, 33, 30, 26, 29, 53, 54, 30, 3~
## $ Outcome                  <int> 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, ~
```

```
PI$Outcome = as.factor(PI$Outcome)
PI$Glucose = na_if(PI$Glucose, 0)
PI$Insulin = na_if(PI$Insulin, 0)
PI$BMI = na_if(PI$BMI, 0)
PI$SkinThickness = na_if(PI$SkinThickness, 0)
PI$BloodPressure = na_if(PI$BloodPressure, 0)
#PI$GlucosePerInsulin = PI$Glucose / PI$Insulin


PI <- PI %>%
  mutate(WeightGroup = cut(PI$BMI, breaks = c(0, 18.4, 24.9, 29.9, Inf),
          labels = c("Underweight", "NormalWeight", "Overweight", "Obese")),
         GlucoseGroup = cut(PI$Glucose, breaks = c(0, 99.9, 124.9,Inf),
          labels = c("Normal", "Prediabetes", "Diabetes"))) %>%
  select(Pregnancies, Glucose, GlucoseGroup, BMI, WeightGroup,everything())

glimpse(PI)
```

```
## Rows: 768
## Columns: 11
## $ Pregnancies              <int> 6, 1, 8, 1, 0, 5, 3, 10, 2, 8, 4, 10, 10, 1, ~
## $ Glucose                  <int> 148, 85, 183, 89, 137, 116, 78, 115, 197, 125~
## $ GlucoseGroup             <fct> Diabetes, Normal, Diabetes, Normal, Diabetes,~
## $ BMI                      <dbl> 33.6, 26.6, 23.3, 28.1, 43.1, 25.6, 31.0, 35.~
## $ WeightGroup              <fct> Obese, Overweight, NormalWeight, Overweight, ~
## $ BloodPressure            <int> 72, 66, 64, 66, 40, 74, 50, NA, 70, 96, 92, 7~
## $ SkinThickness            <int> 35, 29, NA, 23, 35, NA, 32, NA, 45, NA, NA, N~
## $ Insulin                  <int> NA, NA, NA, 94, 168, NA, 88, NA, 543, NA, NA,~
## $ DiabetesPedigreeFunction <dbl> 0.627, 0.351, 0.672, 0.167, 2.288, 0.201, 0.2~
## $ Age                      <int> 50, 31, 32, 21, 33, 30, 26, 29, 53, 54, 30, 3~
## $ Outcome                  <fct> 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, ~
```

```
Critique 1: This dataset only contains the diastolic blood pressure levels. In order to get a better picture o
f artery health, systolic blood pressure levels should have been included.

Critique 2: No data was recorded on the births themselves and whether there were miscarriages or failed birth
s.

Critique 3: No data on whether the group was measured after a fast so I don't feel confident enough to use a c
alculation for Insulin Resistance.

All 0 values were changed to NA values as they aren't close to typical measurements in BMI, BloodPressure, Ski
nThickness, or Insulin levels in any population. I assume they are missing values, instead
```

Exploratory Data Analysis

- First, looking at statistic summaries with skim()– keeping the data *with* NA values. The reason for skim() over boxplots is because after glancing at the data, I can only assume that standard deviations may be small for some variables. This prompts me to look the spread of the data numerically, rather than visually, in order to be precise.

Noticing that the oldest recorded woman is 81 years old– an outlier and 11 years older than than second oldest woman in these samples.

If we look at the skim()'s `BMI` histogram, it seems right-skewed which can be expected if we have more samples in which the person was negative for diabetes (assuming younger people naturally have diabetes less often). Using skim()'s second output, I learn that there are 500 rows for those that did not have diabetes which definitely is more than 268 for those that did. Still, since BMI is a strong predictor of diabetes, I plot the distribution individually– assigning color red for the subset that have diabetes/ blue for those without diabetes for presentation purposes.

After doing so, I find that BMI statistics are actually normally distributed despite the right-skewed histogram found in `skimr`. I find that the mean is ~33 for both those with and without diabetes– indicating that the general population *is* very prone to the health condition. This can be confirmed if you perform research on the Arizona Pima Indians & their extraordinarily high rate of kidney disease and failure as a leading cause of death.

`Insulin`, on the other hand, *is* right-skewed. This means the samples tend to have lower Insulin values more often than higher Insulin values.

```
skimr::skim(PI)
```

Data summary

Name	PI
Number of rows	768
Number of columns	11
_______________________	
Column type frequency:	
factor	3
numeric	8
________________________	
Group variables	None

Variable type: factor

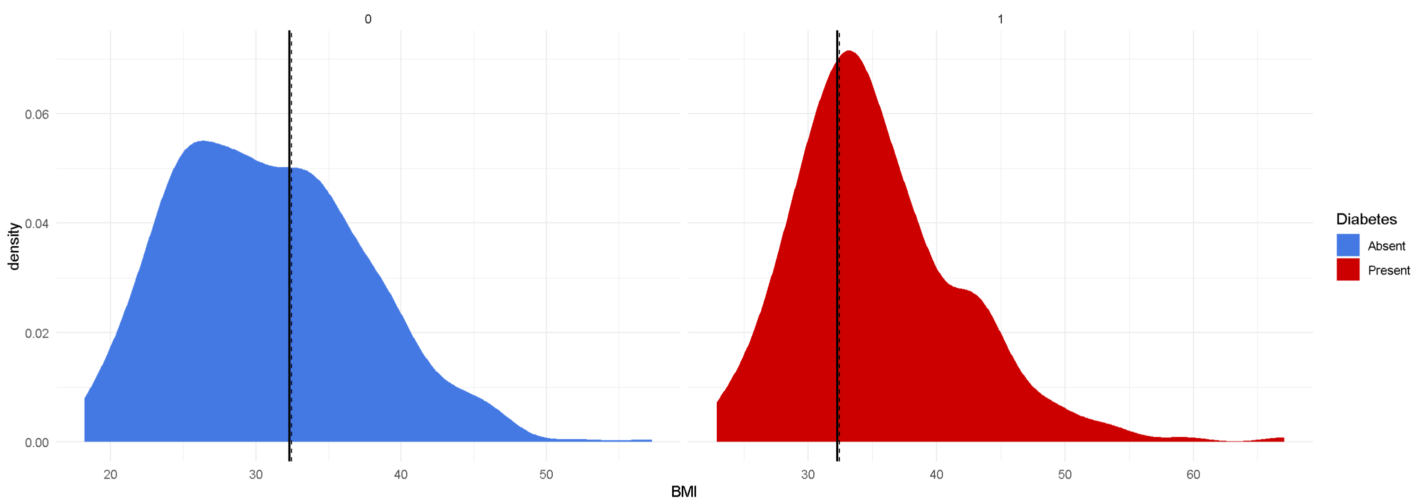
skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
GlucoseGroup	5	0.99	FALSE	3	Dia: 311, Pre: 260, Nor: 192
WeightGroup	11	0.99	FALSE	4	Obe: 472, Ove: 179, Nor: 102, Und: 4
Outcome	0	1.00	FALSE	2	0: 500, 1: 268

Variable type: numeric

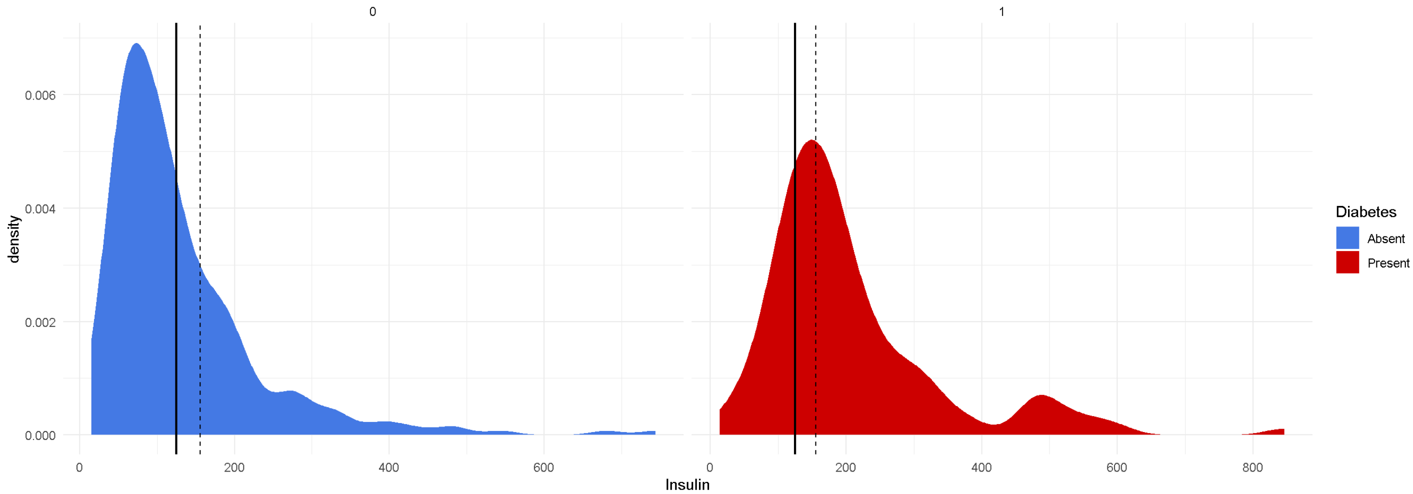
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Pregnancies	0	1.00	3.85	3.37	0.00	1.00	3.00	6.00	17.00	▇▃▂▁▁

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Glucose	5	0.99	121.69	30.54	44.00	99.00	117.00	141.00	199.00	
BMI	11	0.99	32.46	6.92	18.20	27.50	32.30	36.60	67.10	
BloodPressure	35	0.95	72.41	12.38	24.00	64.00	72.00	80.00	122.00	
SkinThickness	227	0.70	29.15	10.48	7.00	22.00	29.00	36.00	99.00	
Insulin	374	0.51	155.55	118.78	14.00	76.25	125.00	190.00	846.00	
DiabetesPedigreeFunction	0	1.00	0.47	0.33	0.08	0.24	0.37	0.63	2.42	
Age	0	1.00	33.24	11.76	21.00	24.00	29.00	41.00	81.00	

```
PI_BMI <- PI[!is.na(PI$BMI),] %>% select(BMI, Outcome)

ggplot(PI_BMI, aes(x=BMI, fill=Outcome)) + theme_minimal() +
  geom_density(color=NA) +
  facet_wrap(vars(Outcome), scale="free_x") +
  geom_vline(aes(xintercept = median(BMI)), size = .8, color = "black") +
  geom_vline(aes(xintercept = mean(BMI)), size=.5, linetype="dashed") +
  scale_fill_manual(values=c("#4479E4", "red3"),
                    name = "Diabetes", labels = c("Absent", "Present"))
```



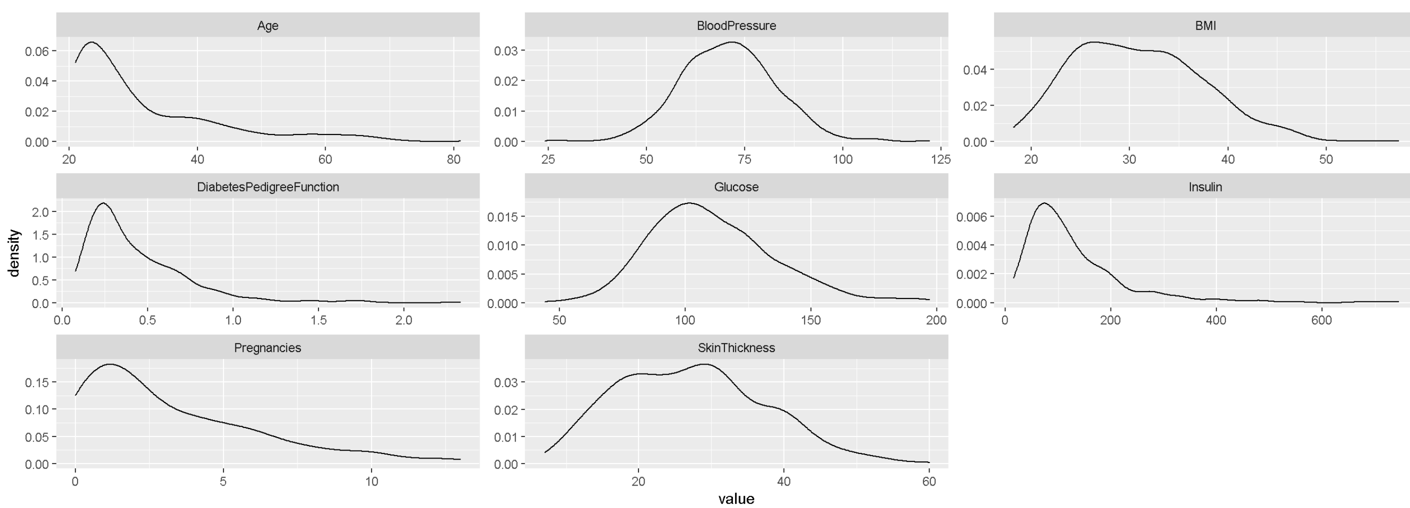

```
PI_Insulin <- PI[!is.na(PI$Insulin),] %>% select(Insulin, Outcome)

ggplot(PI_Insulin, aes(x=Insulin,fill=Outcome)) + theme_minimal() +
  geom_density(color=NA) + facet_wrap(vars(Outcome), scale="free_x") +
  geom_vline(aes(xintercept = median(Insulin)), size = .8, color = "black") +
  geom_vline(aes(xintercept = mean(Insulin)), size=.5, linetype="dashed") +
  scale_fill_manual(values=c("#4479E4", "red3"),
                    name = "Diabetes", labels = c("Absent", "Present"))
```



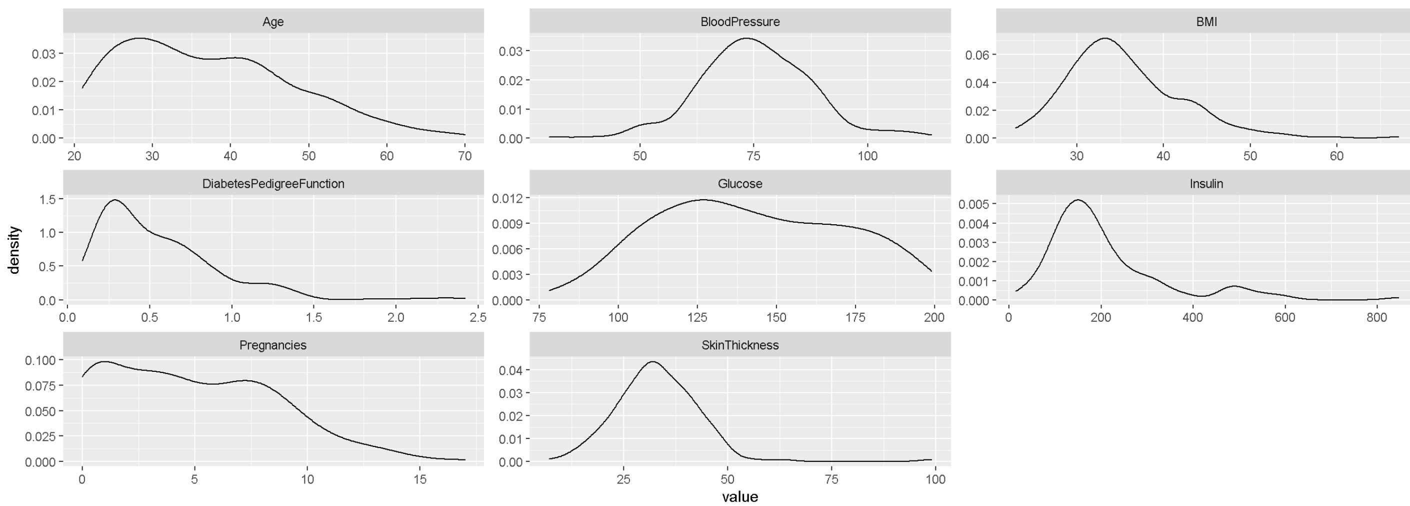

```
Creating two vectors in which I eliminate NA values for each field, independently. This way, I avoid eliminati
ng data from the neighboring columns and portray an accurate distribution.

- Keeping only necessary fields from the vectors in order to save space in the Global Environment.
```

- Second, looking at distributions for the samples that had Diabetes.
- Then, I look at distributions for the samples that did *not* have Diabetes.
- After creating distributions using facet_wrap() and adjusting for a free x-scale, I find that there are either normal or right-skewed distributions only:
    - Right-Skewed: This is true for variables Age, Insulin, Pregnancies, and Pedigree Function.

```
PI %>% filter(Outcome == 0) %>%
  purrr::keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_density()
```



```
PI %>% filter(Outcome == 1) %>%
  purrr::keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_density()
```



Exploring same statistics for all samples that had every attribute present.

The number of women with all attributes recorded is392 rows rather than 768. This is around half of the dataframe: ~ 51%.

- Noticing that every field’s average dropped slightly.
  - BMI still large.
  - Average Glucose levels are normal.
  - Average Blood Pressure is normal but I still see a high max at p100. This is expected, however, as the number of those diagnosed with diabetes now make up ~33% of the total samples.
  - Average Insulin levels for those with all attributes recorded are higher than what is considered normal by 16 mg/dL.
- For those with all attributes recorded, using the quartile ranges, I can determine that at least 75% of the women in the samples will:
  - Have had at least one pregnancy.
  - Be 23 years of age or older.
  - Have glucose levels of 99 or higher. (The average is higher than what’s considered normal by 16 miligrams / decilitre).
  - Have a BMI of 28.4 or higher.
  - Have blood pressure of 62 or more (Nothing out of the ordinary).

```
skimr::skim(PI %>% drop_na())
```

Data summary

Name	PI %>% drop_na()
Number of rows	392
Number of columns	11
________________________	
Column type frequency:	

factor	3
numeric	8
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
GlucoseGroup	0	1	FALSE	3	Dia: 168, Pre: 122, Nor: 102
WeightGroup	0	1	FALSE	4	Obe: 262, Ove: 85, Nor: 44, Und: 1
Outcome	0	1	FALSE	2	0: 262, 1: 130

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Pregnancies	0	1	3.30	3.21	0.00	1.00	2.00	5.00	17.00	
Glucose	0	1	122.63	30.86	56.00	99.00	119.00	143.00	198.00	
BMI	0	1	33.09	7.03	18.20	28.40	33.20	37.10	67.10	
BloodPressure	0	1	70.66	12.50	24.00	62.00	70.00	78.00	110.00	
SkinThickness	0	1	29.15	10.52	7.00	21.00	29.00	37.00	63.00	
Insulin	0	1	156.06	118.84	14.00	76.75	125.50	190.00	846.00	
DiabetesPedigreeFunction	0	1	0.52	0.35	0.09	0.27	0.45	0.69	2.42	
Age	0	1	30.86	10.20	21.00	23.00	27.00	36.00	81.00	

Comparing the mean values for women who have diabetes against those who did not.

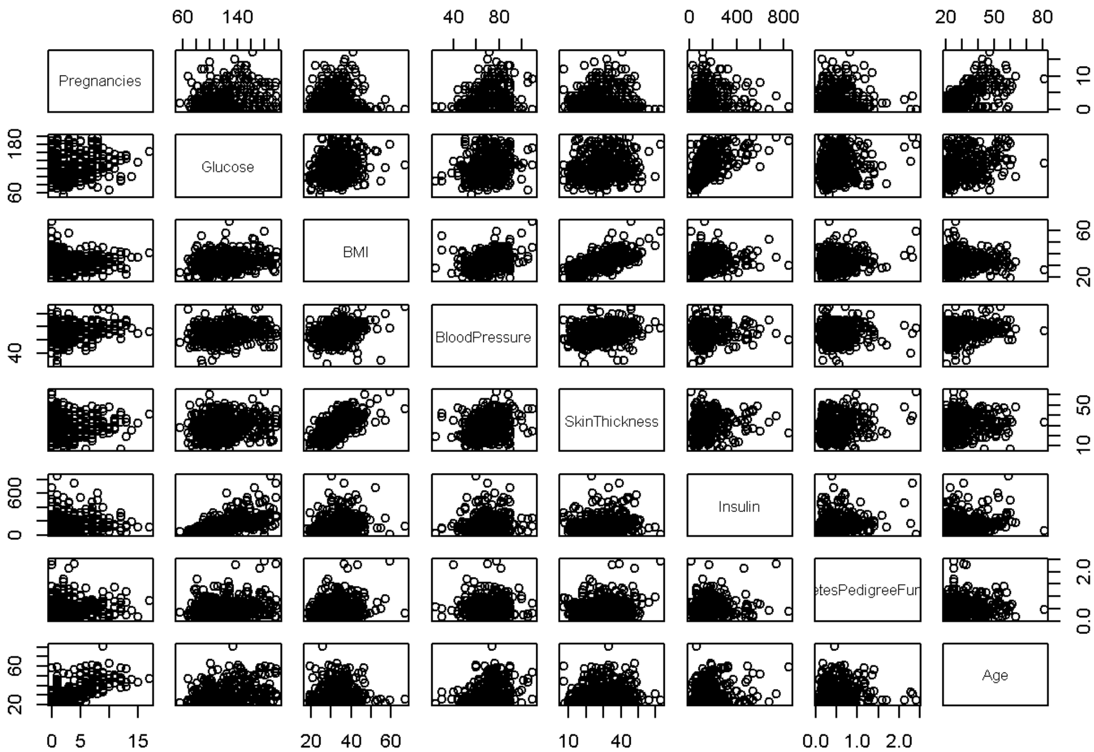
- Both subsets of the data contain an obesity-level average BMI. Those who weren’t diagnosed with diabetes fell, on average, into Class 1 Obesity (BMI of 30 to < 35) while those who were, fell into Class 2 Obesity (BMI of 35 to < 40).
- Women with diabetes tend to have had more pregnancies. As previously stated, no data is recorded on the births themselves and whether there were miscarriages or failed births.

```
num_0 <- PI %>% filter(Outcome==0) %>% drop_na() %>% select(Pregnancies, Glucose, BMI, BloodPressure, SkinThic
kness, Insulin, DiabetesPedigreeFunction, Age) %>%
  colMeans() %>% as.data.frame() %>% tibble::rownames_to_column() %>%
  rename("0" = ".")

num_1 <- PI %>% filter(Outcome==1) %>% drop_na() %>% select(Pregnancies, Glucose, BMI, BloodPressure, SkinThic
kness, Insulin, DiabetesPedigreeFunction, Age) %>%
  colMeans() %>% as.data.frame() %>% tibble::rownames_to_column() %>%
  rename("1" = ".")

stat_means <- inner_join(num_0,num_1, by="rowname") %>% rename("Variable" = "rowname")
stat_means
```

```
##                  Variable           0           1
## 1             Pregnancies   2.7213740   4.4692308
## 2                 Glucose 111.4312977 145.1923077
## 3                     BMI  31.7507634  35.7776923
## 4           BloodPressure  68.9694656  74.0769231
## 5           SkinThickness  27.2519084  32.9615385
## 6                 Insulin 130.8549618 206.8461538
## 7 DiabetesPedigreeFunction   0.4721679   0.6255846
## 8                     Age  28.3473282  35.9384615
```

Checking correlations between numeric variables (before imputation) in order to avoid multi-collinearity for future predictions.

Noticing that the variables that have the highest correlation are: 1) Pregnancy and Age 2) Skin Thickness and BMI 3) Insulin and Glucose




```
# Setting seed to compare predictive models.

set.seed(0)
```

LOGISTIC REGRESSION

- Using k-NN to impute data since we have many samples which are partially filled (MCAR).

```
library(caret)
library(caTools)
library(VIM)



split_log <- sample.split(PI, SplitRatio = 0.8)
train_log <- subset(PI, split_log == TRUE)
test_log <- subset(PI, split_log == FALSE) %>% drop_na()

train_log_data <- kNN(train_log, k=round(sqrt(nrow(train_log)))-1, imp_var = FALSE)

logmod1 <- glm(Outcome~., family="binomial", data=train_log_data)
logmod2 <- glm(Outcome ~.-WeightGroup, family = "binomial", data = train_log_data)
logmod3 <- glm(Outcome~Pregnancies+Glucose+BMI, family="binomial", data=train_log_data)

# you could do cv logistic, as well, if you wanted in the future.

predicted_log <- test_log %>%
              mutate(p1=predict(logmod1, newdata=test_log, type="response"),
                     p2=predict(logmod2, newdata=test_log, type="response"),
                     p3=predict(logmod3, newdata=test_log, type="response")) %>%
              mutate(S1=ifelse(p1<0.5,0,1),
                     S2=ifelse(p2<0.5,0,1),
                     S3=ifelse(p3<0.5,0,1)) %>%
              select(Outcome, S1, S2, S3)
```

Creating confusion matrices and looking at the accuracy metrics. Realizing that my model with all variables, whether statistically significant or not, is the best at predicting accurately.

```
predicted_log <- na.omit(predicted_log)

one <- table(predicted_log$Outcome, predicted_log$S1) %>% prop.table()
two <- table(predicted_log$Outcome, predicted_log$S2) %>% prop.table()
three <- table(predicted_log$Outcome, predicted_log$S3) %>% prop.table()


ERROR.RESULTS <- tibble(
  Model=c("One", "Two", "Three"),
  Sensitivity=c(one[1,1]/sum(one[1,]),
                two[1,1]/sum(two[1,]),
                three[1,1]/sum(three[1,])),

  Specificity=c(one[2,2]/sum(one[2,]),
                two[2,2]/sum(two[2,]),
                three[2,2]/sum(three[2,])),

  FalsePositives=c(one[2,1]/sum(one[2,]),
                   two[2,1]/sum(two[2,]),
                   three[2,1]/sum(three[2,])),

  FalseNegatives=c(one[1,2]/sum(one[1,]),
                   two[1,2]/sum(two[1,]),
                   three[1,2]/sum(three[1,]))
)

ERROR.RESULTS
```

```
## # A tibble: 3 x 5
##   Model Sensitivity Specificity FalsePositives FalseNegatives
##   <chr>       <dbl>       <dbl>          <dbl>          <dbl>
## 1 One         0.831       0.579          0.421          0.169
## 2 Two         0.831       0.526          0.474          0.169
## 3 Three       0.831       0.526          0.474          0.169
```

K NEAREST NEIGHBORS

Since the creation of a logistic model showed me that categorical data was never significant, I proceed with making a k-NN predictive model, which doesn't make use of factor/dummy variables (0's & 1's), utilizing cross validation!

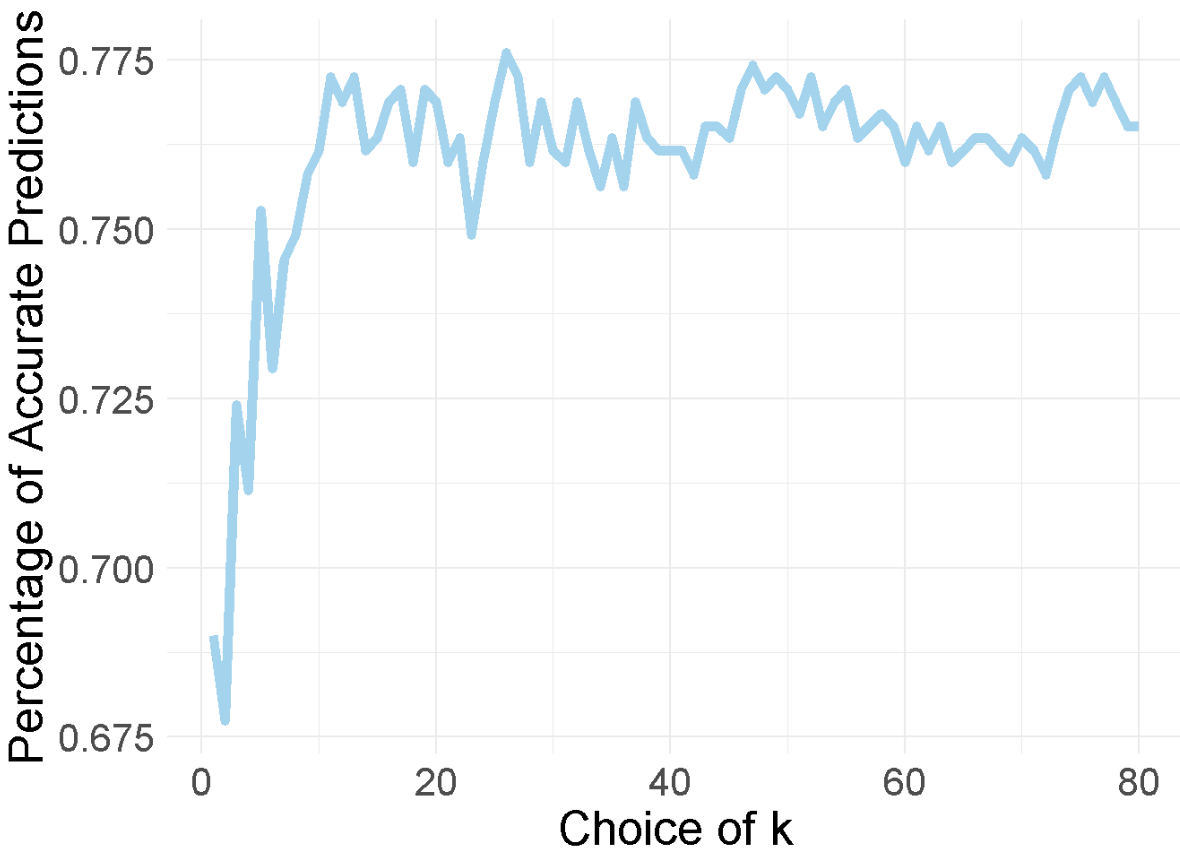
- Creating a dataset with imputated kNN values for each field (which is recommended for MCAR).
- Using sqrt(# rows) for my 'k' in my imputation method as this is a known rule of thumb
- Then, normalizing all the numerical columns within my training data as well as testing data.

Finding Best 'k'.

```
split_knn <- sample.split(PI, SplitRatio = 0.8)
train_knn <- subset(PI, split_log == TRUE)
test_knn <- subset(PI, split_log == FALSE) %>% drop_na()

train_knn_data <- kNN(train_knn, k=round(sqrt(nrow(train_knn))) - 1, imp_var = FALSE)

# normalizer data function
normalizer <- function(x){
  return((x-min(x))/(max(x)-min(x)))
}

# selecting numerical columns that will be used to perform predictions
train_knn_scaled <-
  as.data.frame(lapply(train_knn_data %>% select(where(is.numeric)), normalizer)) %>%
  mutate(Outcome = train_knn_data$Outcome)


test_knn_scaled <-
  as.data.frame(lapply(test_knn %>% select(where(is.numeric)), normalizer)) %>%
  mutate(Outcome = test_knn$Outcome)
```

Performing "leave-one-out" CV (each sample of test voted by on every sample in testing) in order to find the best k.

```r
library(class)

possible_k=1:80
accuracy_k=rep(NA,80)

# knn.cv() is a function that automatically does the knn cross validation for you as it predicts.

for(k in 1:80){
  cv.out <- knn.cv(train=train_knn_scaled %>% select(-Outcome),
              cl=factor(train_knn_scaled$Outcome,levels=c(0,1),
                        labels=c("Absent","Present")), k=k)

  correct=mean(cv.out==factor(train_knn_scaled$Outcome,levels=c(0,1),
                              labels=c( "Absent","Present")))

  accuracy_k[k]=correct
}

ggplot(data=tibble(possible_k,accuracy_k)) +
  geom_line(aes(x=possible_k,y=accuracy_k),color="lightskyblue2",size=2) +
  theme_minimal() +
  xlab("Choice of k") +
  ylab("Percentage of Accurate Predictions") +
  theme(text=element_text(size=20))
```



Extracting the optimal predictions using k-NN. using the k index.

```
knn_predictions = test_knn_scaled %>%
          mutate(Predict=knn(train=select(train_knn_scaled, -Outcome),
                             test=select(test_knn_scaled, -Outcome),
                             cl=factor(train_knn_scaled$Outcome,levels=c(0,1)),
                             k=which.max(accuracy_k)))
```

Assessing KNN model predictions' accuracy.

```
cm <- table(knn_predictions$Outcome, knn_predictions$Predict) %>% prop.table()



ERROR.RESULTS <- tibble(
  Sensitivity=c(cm[1,1]/sum(cm[1,])),
  Specificity=c(cm[2,2]/sum(cm[2,])),
  FalsePositives=c(cm[2,1]/sum(cm[2,])),
  FalseNegatives=c(cm[1,2]/sum(cm[1,]))
)

ERROR.RESULTS
```

```
## # A tibble: 1 x 4
##   Sensitivity Specificity FalsePositives FalseNegatives
##         <dbl>       <dbl>          <dbl>          <dbl>
## 1       0.831       0.526          0.474          0.169
```

Conclusion:

Logistic model has a better predictive accuracy for those who have diabetes.