Final Project

Megan Holford

11/19/2020

Summarize the problem statement you addressed.

For my final project, "can you predict the chance for an individual to have diabetes and what attributes can be used to make this prediction?" was the question I wanted to look at. I knew that data science, models, and machine learning is a growing aspect of the medical field at health care providers are looking for efficient ways to assist with diagnosing cases in their patients. This particular question can be helpful in finding if an individual has the chance to develop diabetes before onset and if measures can be taken to lessen or deter. I am not in the healthcare field, but I wanted to try and apply it myself with the knowledge I have gathered in this course. After working on my project, I definitely have a lot to learn, but this was a great way to try an understand how this is begin done and a glimpse into the future of healthcare.

Summarize how you addressed this problem statement (the data used and the methodology employed).

For this assignment, I used several different models to see which models have better results with the available data. In particular I used logistic regression, KNN, random forest, gradient boosting, and regression tree. By using different models, I am able to see how models different with their accuracy with the data, and this also allows me to try out these models and continue my learning.

Models and Comparisons.

Importing libraries and data set. Also cleaning the data set for use.

```
## convert predictors except age into factors
diabetes<- diabetes %>% mutate_if(is.character, as.factor)

names(diabetes)[3] <- "frequent.urination"
names(diabetes)[4] <- "extreme.thirst"
names(diabetes)[7] <- "extreme.hunger"
names(diabetes)[13] <- "muscle.impairment"


colnames(diabetes)
```

```
##  [1] "Age"                "Gender"             "frequent.urination"
##  [4] "extreme.thirst"     "sudden.weight.loss" "weakness"
##  [7] "extreme.hunger"     "Genital.thrush"     "visual.blurring"
## [10] "Itching"            "Irritability"       "delayed.healing"
## [13] "muscle.impairment"  "muscle.stiffness"   "Alopecia"
## [16] "Obesity"            "class"
```

```r
str(diabetes)
```

```
## 'data.frame':    520 obs. of  17 variables:
##  $ Age               : int  40 58 41 45 60 55 57 66 67 70 ...
##  $ Gender            : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
##  $ frequent.urination: Factor w/ 2 levels "No","Yes": 1 1 2 1 2 2 2 2 2 1 ...
##  $ extreme.thirst    : Factor w/ 2 levels "No","Yes": 2 1 1 1 2 2 2 2 2 2 ...
##  $ sudden.weight.loss: Factor w/ 2 levels "No","Yes": 1 1 1 2 2 1 1 2 1 2 ...
##  $ weakness          : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ extreme.hunger    : Factor w/ 2 levels "No","Yes": 1 1 2 2 2 2 2 1 2 2 ...
##  $ Genital.thrush    : Factor w/ 2 levels "No","Yes": 1 1 1 2 1 1 2 1 2 1 ...
##  $ visual.blurring   : Factor w/ 2 levels "No","Yes": 1 2 1 1 2 2 1 2 1 2 ...
##  $ Itching           : Factor w/ 2 levels "No","Yes": 2 1 2 2 2 2 1 2 2 2 ...
##  $ Irritability      : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 1 1 2 2 2 ...
##  $ delayed.healing   : Factor w/ 2 levels "No","Yes": 2 1 2 2 2 2 2 1 1 1 ...
##  $ muscle.impairment : Factor w/ 2 levels "No","Yes": 1 2 1 1 2 1 2 2 2 1 ...
##  $ muscle.stiffness  : Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 1 2 2 1 ...
##  $ Alopecia          : Factor w/ 2 levels "No","Yes": 2 2 2 1 2 2 1 1 1 2 ...
##  $ Obesity           : Factor w/ 2 levels "No","Yes": 2 1 1 1 2 2 1 1 2 1 ...
##  $ class             : Factor w/ 2 levels "Negative","Positive": 2 2 2 2 2 2 2 2 2 2 ...
```

```r
summary(diabetes)
```

```
##       Age           Gender    frequent.urination extreme.thirst
##  Min.   :16.00   Female:192   No :262            No :287
##  1st Qu.:39.00   Male  :328   Yes:258            Yes:233
##  Median :47.50
##  Mean   :48.03
##  3rd Qu.:57.00
##  Max.   :90.00
##  sudden.weight.loss weakness  extreme.hunger Genital.thrush visual.blurring
##  No :303            No :215   No :283        No :404        No :287
##  Yes:217            Yes:305   Yes:237        Yes:116        Yes:233
##
##
##
##
##  Itching   Irritability delayed.healing muscle.impairment muscle.stiffness
##  No :267   No :394      No :281         No :296           No :325
##  Yes:253   Yes:126      Yes:239         Yes:224           Yes:195
##
##
##
##
##  Alopecia  Obesity        class
##  No :341   No :432   Negative:200
##  Yes:179   Yes: 88   Positive:320
##
##
##
##
```

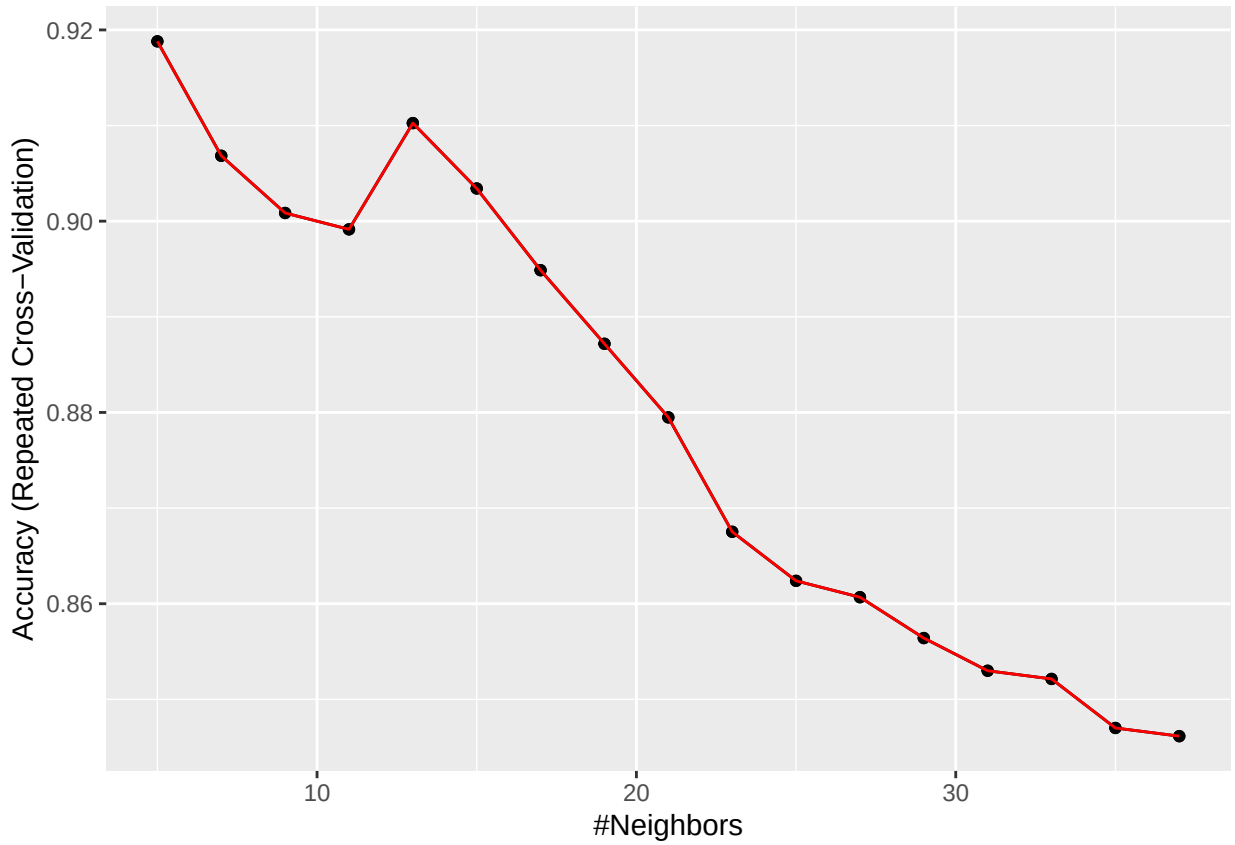
Create a training set and a test set.

```r
set.seed(100)
diabetes_data<- createDataPartition(diabetes$class, p=.75, list = F)
train_diabetes<- diabetes[diabetes_data,]
test_diabetes<- diabetes[-diabetes_data,]

## tuning the parameters
diabetes_control<- trainControl(method = "repeatedcv", number = 10, repeats = 3)
```

##KNN classification model

```
## k-Nearest Neighbors
##
## 390 samples
##  16 predictor
##   2 classes: 'Negative', 'Positive'
##
## Pre-processing: centered (16), scaled (16)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 351, 351, 351, 351, 351, 351, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.9188034  0.8325568
##    7  0.9068376  0.8099698
##    9  0.9008547  0.7983791
##   11  0.8991453  0.7966006
##   13  0.9102564  0.8187680
##   15  0.9034188  0.8054209
##   17  0.8948718  0.7892234
##   19  0.8871795  0.7748732
##   21  0.8794872  0.7601462
##   23  0.8675214  0.7379541
##   25  0.8623932  0.7284703
##   27  0.8606838  0.7250729
##   29  0.8564103  0.7170089
##   31  0.8529915  0.7105780
##   33  0.8521368  0.7089946
##   35  0.8470085  0.6997217
##   37  0.8461538  0.6981323
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```



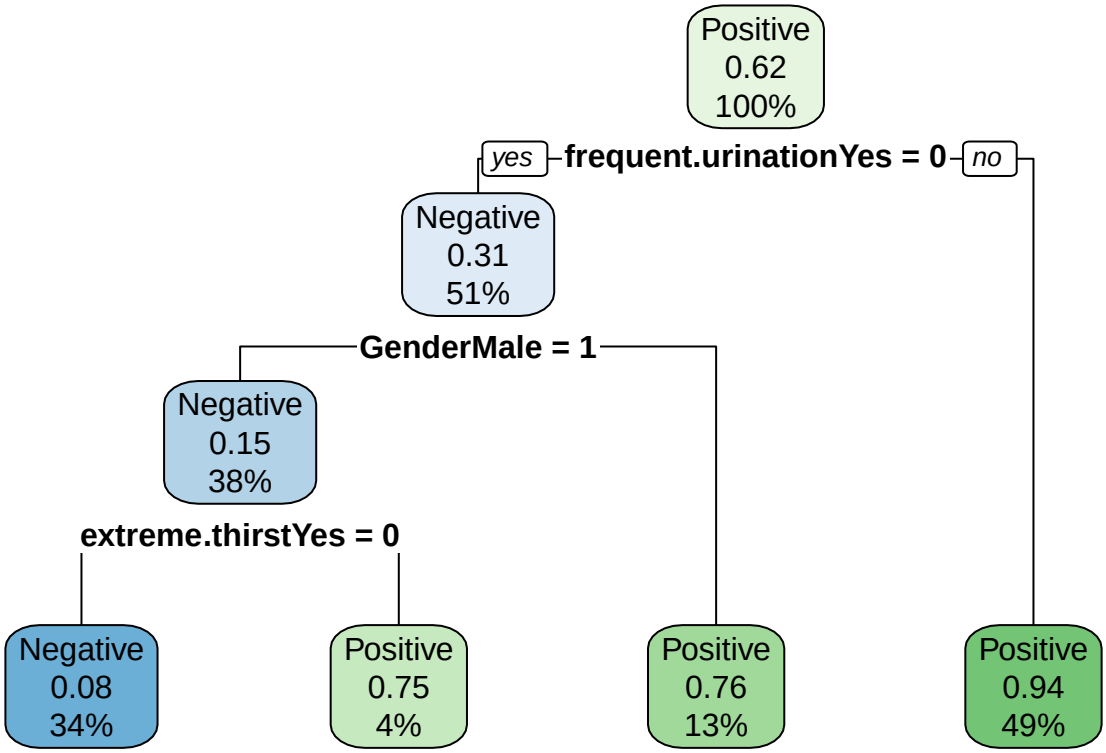

```
## <ScaleContinuousPosition>
##  Range:
##  Limits:    0 --    1
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Negative Positive
##   Negative       48        9
##   Positive        2       71
##
##                Accuracy : 0.9154
##                  95% CI : (0.8536, 0.957)
##     No Information Rate : 0.6154
##     P-Value [Acc > NIR] : 7.494e-15
##
##                   Kappa : 0.8258
##
##  Mcnemar's Test P-Value : 0.07044
##
##             Sensitivity : 0.8875
##             Specificity : 0.9600
##          Pos Pred Value : 0.9726
##          Neg Pred Value : 0.8421
##              Prevalence : 0.6154
```

```
##          Detection Rate : 0.5462
##    Detection Prevalence : 0.5615
##       Balanced Accuracy : 0.9237
##
##        'Positive' Class : Positive
##
```

Accuracy is approximately 91% for the KNN model.

##Classification Tree




```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Negative Positive
##   Negative       35        2
##   Positive       15       78
##
##                Accuracy : 0.8692
##                  95% CI : (0.7989, 0.9219)
##     No Information Rate : 0.6154
##     P-Value [Acc > NIR] : 1.399e-10
##
##                   Kappa : 0.7096
##
##  Mcnemar's Test P-Value : 0.003609
```

```
##
##             Sensitivity : 0.9750
##             Specificity : 0.7000
##          Pos Pred Value : 0.8387
##          Neg Pred Value : 0.9459
##              Prevalence : 0.6154
##          Detection Rate : 0.6000
##    Detection Prevalence : 0.7154
##       Balanced Accuracy : 0.8375
##
##        'Positive' Class : Positive
##
```

Accuracy is approximately 86% for this model.

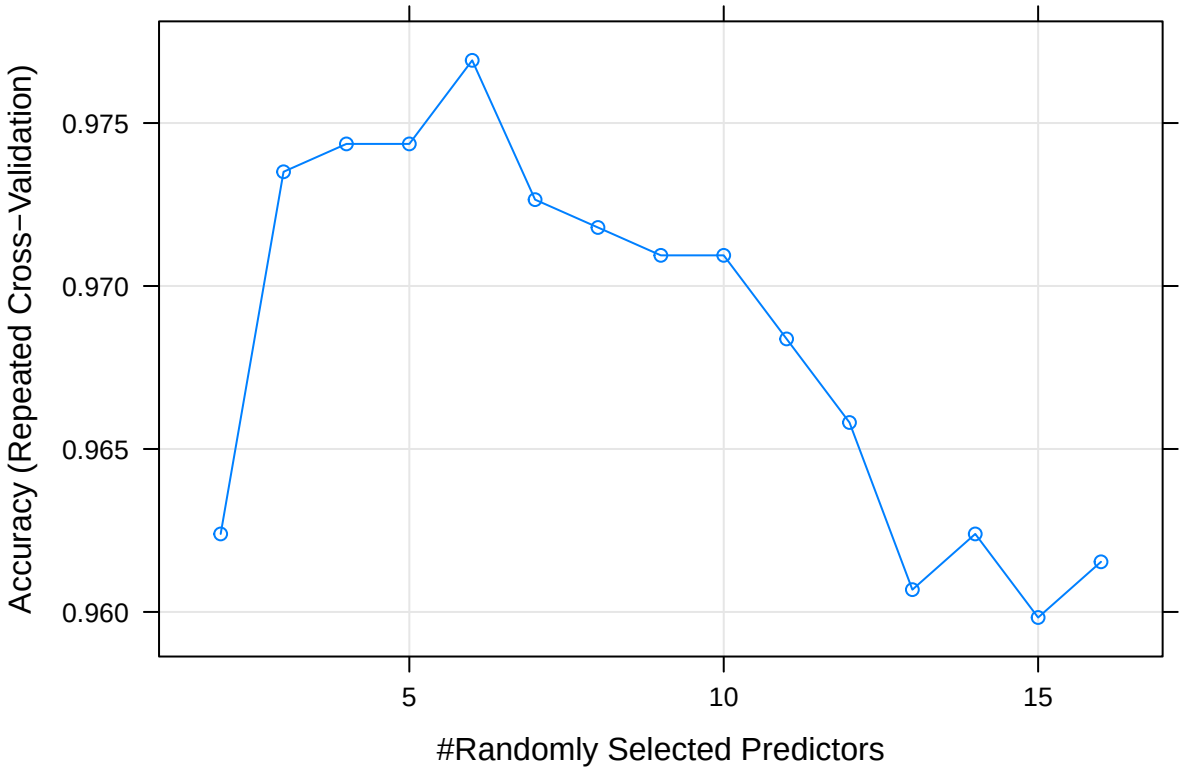
##Logistic Regression

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Negative Positive
##   Negative       44        4
##   Positive        6       76
##
##                Accuracy : 0.9231
##                  95% CI : (0.8631, 0.9625)
##     No Information Rate : 0.6154
##     P-Value [Acc > NIR] : 1.081e-15
##
##                   Kappa : 0.8363
##
##  Mcnemar's Test P-Value : 0.7518
##
##             Sensitivity : 0.9500
##             Specificity : 0.8800
##          Pos Pred Value : 0.9268
##          Neg Pred Value : 0.9167
##              Prevalence : 0.6154
##          Detection Rate : 0.5846
##    Detection Prevalence : 0.6308
##       Balanced Accuracy : 0.9150
##
##        'Positive' Class : Positive
##
```

Accuracy rate of around 92% for logistic regression.

##Random Forest

```
## note: only 15 unique complexity parameters in default grid. Truncating the grid to 15 .
```




```
## Random Forest
##
## 390 samples
##  16 predictor
##   2 classes: 'Negative', 'Positive'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 351, 351, 351, 351, 351, 351, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9623932  0.9211603
##    3    0.9735043  0.9437221
##    4    0.9743590  0.9454579
##    5    0.9743590  0.9455053
##    6    0.9769231  0.9512198
##    7    0.9726496  0.9422132
##    8    0.9717949  0.9402927
##    9    0.9709402  0.9383705
##   10    0.9709402  0.9386438
##   11    0.9683761  0.9330197
##   12    0.9658120  0.9277180
##   13    0.9606838  0.9169303
##   14    0.9623932  0.9206687
##   15    0.9598291  0.9151454
```

```
##   16    0.9615385  0.9186685
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 6.
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Negative Positive
##   Negative       48        1
##   Positive        2       79
##
##                Accuracy : 0.9769
##                  95% CI : (0.934, 0.9952)
##     No Information Rate : 0.6154
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9511
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9875
##             Specificity : 0.9600
##          Pos Pred Value : 0.9753
##          Neg Pred Value : 0.9796
##              Prevalence : 0.6154
##          Detection Rate : 0.6077
##    Detection Prevalence : 0.6231
##       Balanced Accuracy : 0.9738
##
##        'Positive' Class : Positive
##
```

Accuracy rate of around 97% for random forest.

##Gradient Boosting

```
## Loaded gbm 2.1.8
```

```
## A gradient boosted model with bernoulli loss function.
## 225 iterations were performed.
## There were 16 predictors of which 16 had non-zero influence.
```

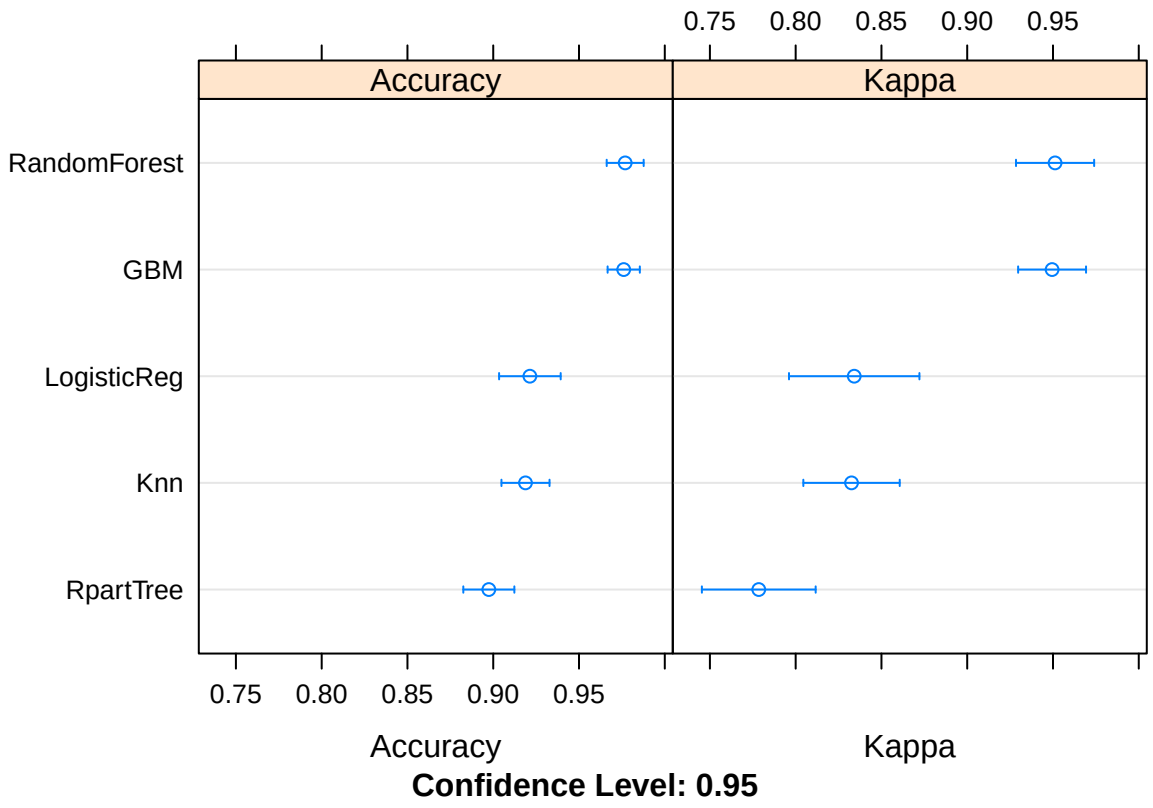
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Negative Positive
##   Negative       49        2
##   Positive        1       78
##
##                Accuracy : 0.9769
##                  95% CI : (0.934, 0.9952)
##     No Information Rate : 0.6154
```

```
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9514
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9750
##             Specificity : 0.9800
##          Pos Pred Value : 0.9873
##          Neg Pred Value : 0.9608
##              Prevalence : 0.6154
##          Detection Rate : 0.6000
##    Detection Prevalence : 0.6077
##       Balanced Accuracy : 0.9775
##
##        'Positive' Class : Positive
##
```

There is an accuracy of approximately 97% for gradient boosting.

##Comparing the models

```
##
## Call:
## summary.resamples(object = model_comp)
##
## Models: Knn, LogisticReg, RpartTree, RandomForest, GBM
## Number of resamples: 30
##
## Accuracy
##                   Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Knn          0.8205128 0.8974359 0.9230769 0.9188034 0.9487179 0.974359    0
## LogisticReg  0.7948718 0.8974359 0.9230769 0.9213675 0.9679487 1.000000    0
## RpartTree    0.8205128 0.8717949 0.8974359 0.8974359 0.9230769 0.974359    0
## RandomForest 0.8974359 0.9743590 0.9743590 0.9769231 1.0000000 1.000000    0
## GBM          0.9230769 0.9743590 0.9743590 0.9760684 1.0000000 1.000000    0
##
## Kappa
##                   Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Knn          0.6513410 0.7859127 0.8374746 0.8325568 0.8943089 0.9465021    0
## LogisticReg  0.5555556 0.7777778 0.8395062 0.8341250 0.9324380 1.0000000    0
## RpartTree    0.5955556 0.7257384 0.7777778 0.7784922 0.8384904 0.9451477    0
## RandomForest 0.7833333 0.9451477 0.9465021 0.9512198 1.0000000 1.0000000    0
## GBM          0.8354430 0.9451477 0.9465021 0.9494705 1.0000000 1.0000000    0
```



Summarize the implications to the consumer (target audience) of your analysis.

I am definitely not reinventing the wheel by using models to try and predict diabetes, but it was very interesting to work with the models and see how they compare in accuracy with my data set.

Discuss the limitations of your analysis and how you, or someone else, could improve or build on it.

One of the biggest challenges I think there is when it comes to creating models for your data is finding the right model to use with your data. One model may be a better fit depending on the data and parameters you are using. I definitely need more experience working with models and fitting them to get more understanding to which models are best to find what I am looking for.

The models that had the highest accuracy were Random Forest and Gradient Boosting models. While the other model did not have as high accuracy percentages, they are still helpful to look at and get a better look at the data.

My data set also is set around common symptoms and side effects of diabetes and not actual blood work so predictions could be a good indicator on if a healthcare professional needs to look further into a diagnosis, but data with information on blood work, glucose levels, and more may have different results.

This type of data analysis is already out of my normal wheelhouse, so I did not go into those types of attributes, but it was definitely a great experience to try and understand how data science can be used in medical fields and why it is such a fast growing subset within the field. I look forward to learning more and getting more experience with R as well as learning more about understanding the data.