A Statistical Approach to Predicting Diabetes

Matt Parker, Zooun Park

University of Minnesota Twin Cities

STAT 4052: Applied Statistics II

Sara Algeri

December 15, 2023

Introduction

Our STAT 4052 final project tasked us with analyzing a dataset containing medical and demographic information of 5000 medical patients. Each patient was given a diagnosis of diabetic status, either negative or positive. Our goal was to conduct statistical analysis on our diabetes dataset considering two objectives: one, to determine the most important variables in predicting diabetic status, and two, creating a statistical model to help predict the diabetic status of future patients. In doing so, we employed several statistical methods that were carefully chosen given the distribution and assumptions of our dataset. Our analysis determined variables HbA1c Levels and Blood Glucose Levels were the most statistically significant variables in predicting diabetes, and thus should be strongly considered when analyzing the diabetic status of future medical patients.

Methods

We will begin our analysis by conducting the necessary preliminary imputations to our dataset. This includes imputing the mean for any missing values contained in our continuous variables, and excluding any observations with missingness for our categorical variables. This will set the stage for the three models we will produce in the first stage of our analysis. Stage two will include the same models tested in stage one, utilizing a dataset imputed using iterative regression.

The first statistical model we will apply to our dataset is a logistic regression (LR) model. Since our response variable is binary with values 0 and 1, we can not fit a simple linear regression model. Given that the proportion of people who have diabetes could be more than 100% when using linear regression, logistic regression is the more suitable model for this dataset.

Prior to constructing our LR model, we will implement a forward stepwise selection model. One of the main objectives of this project is to find an optimal predictive model for future medical patients. Given that we have 15 possible predictors, it’s possible that only a select few amongst the

15 are statistically significant. To complete our objective effectively, selecting a portion of the available variables may lead to a more optimal model, using the AIC as a means of model comparison. In addition, the stepwise method is ideal for model selection given the amount of predictors present, avoiding computationally expensive calculations involved in best subset selection.

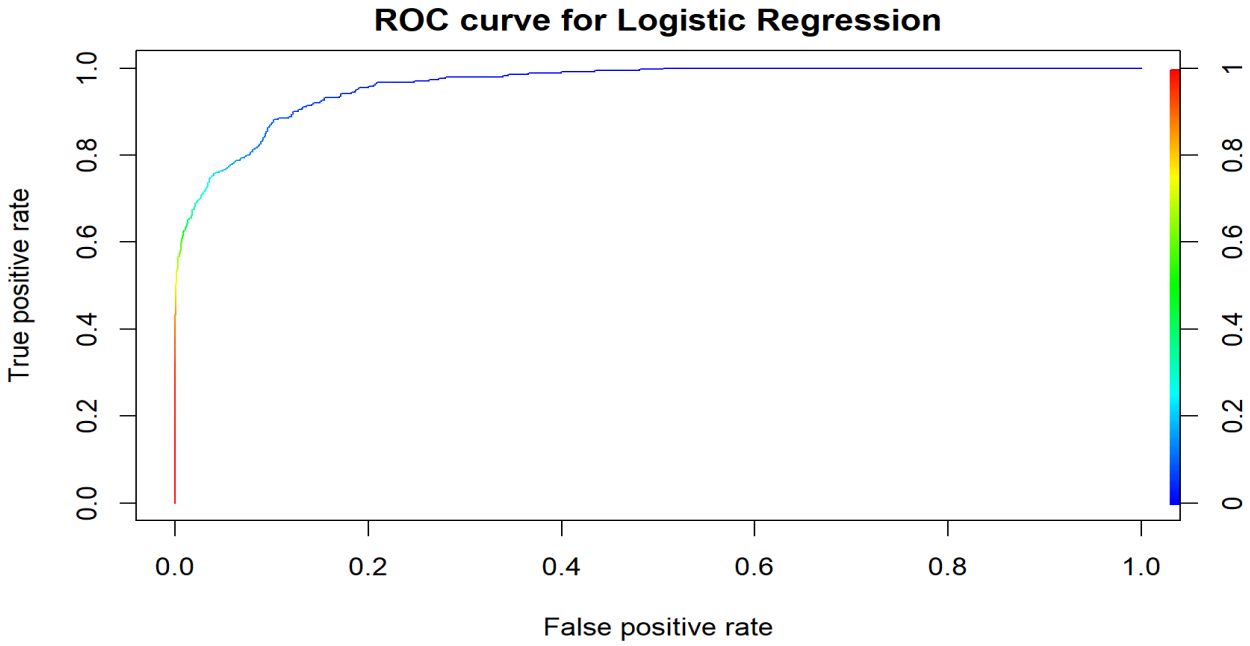
The next method we will apply is K nearest neighbor (KNN). KNN usually performs best when gaussianity assumptions amongst predictors and/or equal covariance are not met, no assumptions are made about the shape of the decision boundary, and when we have a lot of observations relative to the number of predictors. In our case, it is difficult to claim that all predictors meet the assumptions of gaussianity since we have both categorical variables and continuous variables (ex. Blood glucose level) that are more skewed than others when observing plots. In addition, we have more than 4000 observations, so KNN may fit best amongst our classification methods.

Lastly, we will implement a random forest (RF) model. RF will be beneficial to explore given the statistical power of bootstrap sampling, as well as its strength in modeling a few powerful predictors. Although boosting and bagging could be considered, boosting is computationally expensive in terms of tuning parameters, while bagging faces problems when a select few predictors are substantially important.

Results

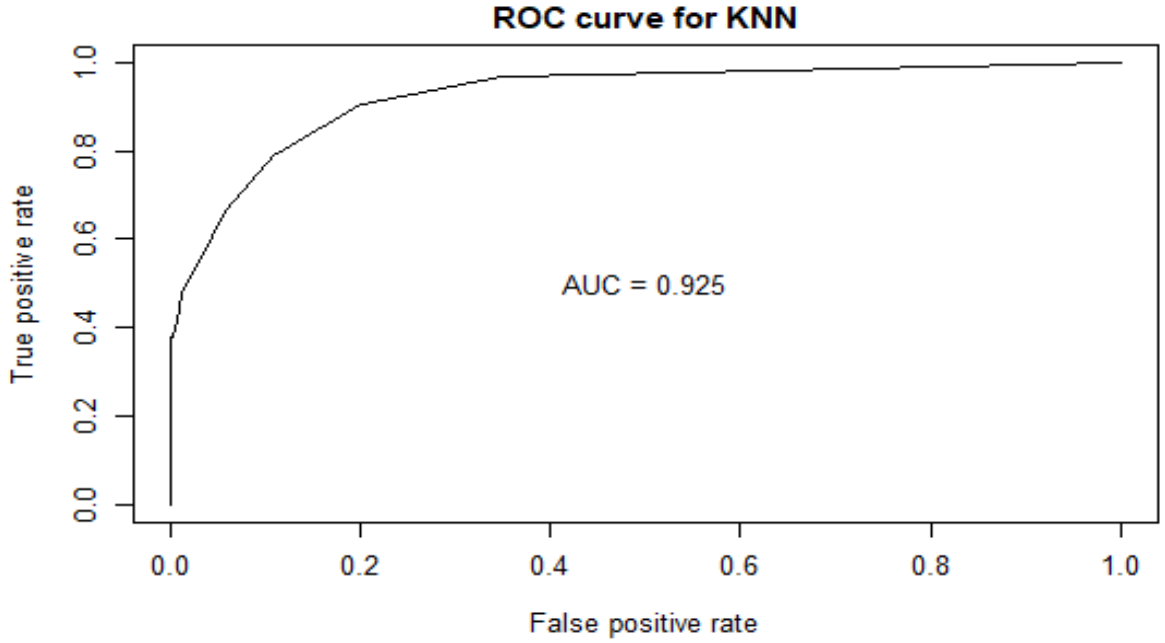
We'll begin our results by analyzing the relevant plots and outputs of our logistic regression model. In selecting our optimal model using forward selection, 9 of the 15 potential predictor variables were included, specifically HbA1c levels, Blood Glucose Levels, Age, BMI, Hypertension, Heart Disease, No Information on Smoking History, Has a Patient Ever Smoked, and

Male. Utilizing the bayes classifier to assign observations to class Diabetes or No Diabetes, our error rate is equal to 0.0387. Below are the results of our ROC curve, with an AUC score equal to 0.9615.

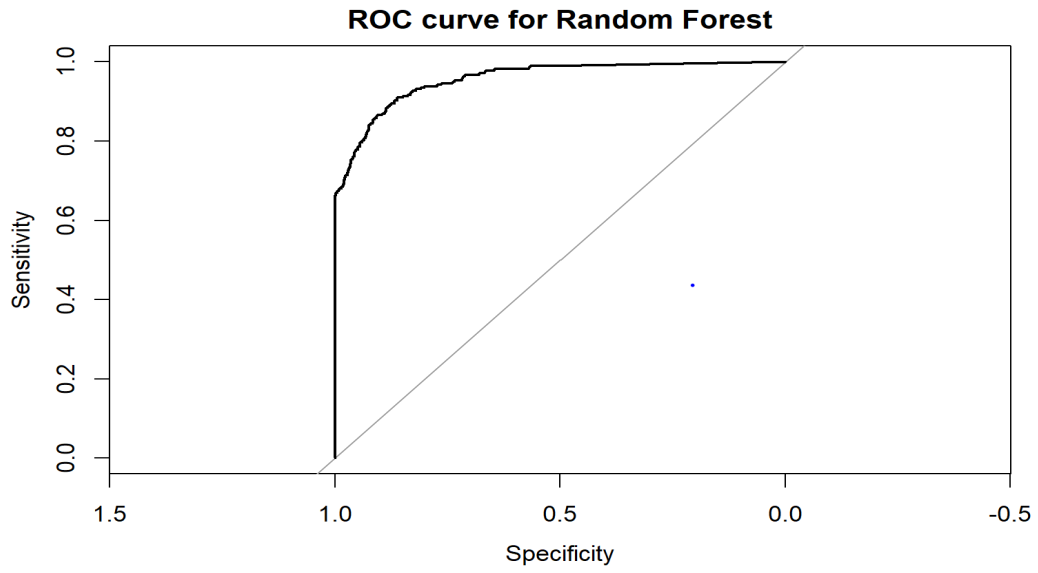


Next, we'll visualize the results of our KNN models. We start by constructing an 80/20 split of our dataset into training and validation sets. Next, we produce six KNN models tested at k equal to 3, 5, 10, 20, 60, and 80. Below are the error rates for each respective KNN model, as well as the ROC curve for k equals 20. We can see that the error rate is lowest at k equals 20 and increases as we get larger, so k equals 20 will be our optimal KNN model.

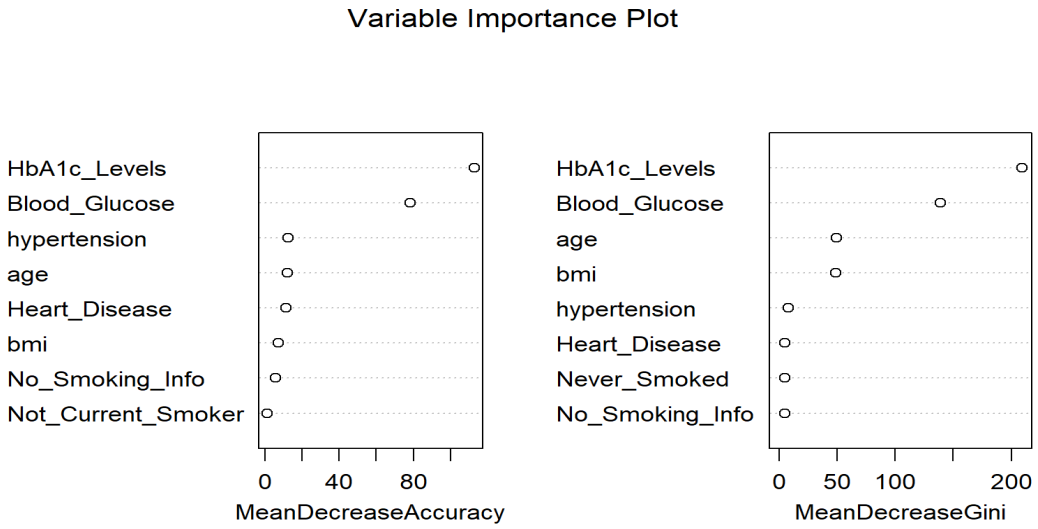
K	3	5	10	20	60	80
Test Error Rate	0.054	0.053	0.051	0.048	0.050	0.051



Lastly, in utilizing our simple imputation dataset, we'll convey the results of our random forest model. In producing this model, we took 5 variables at each split (p/3, p is the number of predictors) and created 500 trees. Our error rate was equal to 0.0278, with an OOB error of 0.029, indicating a strong fit considering the similarity of these values. Below is a visualization of the ROC curve with an AUC of 0.9567.



In addition to our ROC curve, we'll construct a variable importance plot of the 8 most important variables in our RF model. We can see that HbA1c Levels and Blood Glucose Levels are the most important variables.

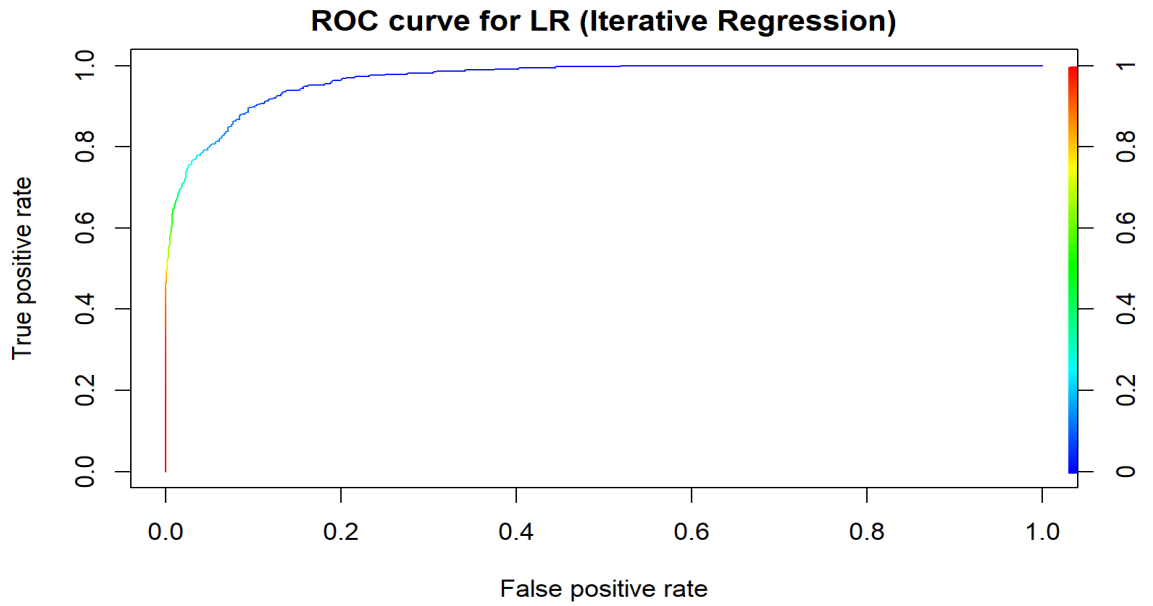


After producing the output and plots of each model over the simple imputation dataset, how can we adequately interpret our results? Let's begin by noting our prior probability and its relation to our error rates. For our simple imputation dataset, approximately 8.2% of individuals were diagnosed with diabetes. This means that if we were to assign 0 (negative diabetic status) to all patients, our estimated test error rate would be 0.082. If we take a look at our estimated error rates produced by each model, they are equal to 0.0387 for LR, 0.0508 for KNN, and 0.0278 for RF. All of these values are significantly smaller than our prior probability, and thus they all provide a statistically powerful means of classification.

Next, we will move to stage two: implementing data imputed using iterative regression. Hb1Ac Levels, Diabetes and Hypertension were imputed using the iterative regression process.

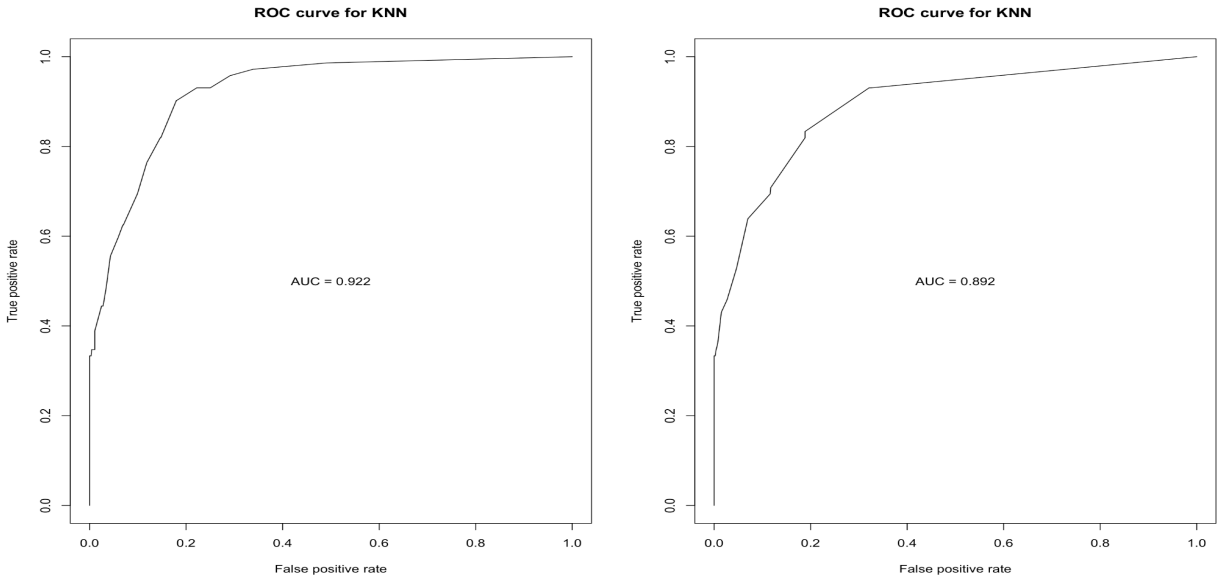
The first model post iterative regression imputation we will analyze is our forward stepwise selection model. In utilizing the lowest AIC criterion, a model with variables HbA1c Levels, Blood Glucose Levels, Age, BMI, Heart Disease, Hypertension and No Information on Smoking History

were chosen. Applying the bayes classifier to diabetic status, an error rate of 0.0346 was produced. Below is a visualization of the ROC curve with AUC equal to 0.968.



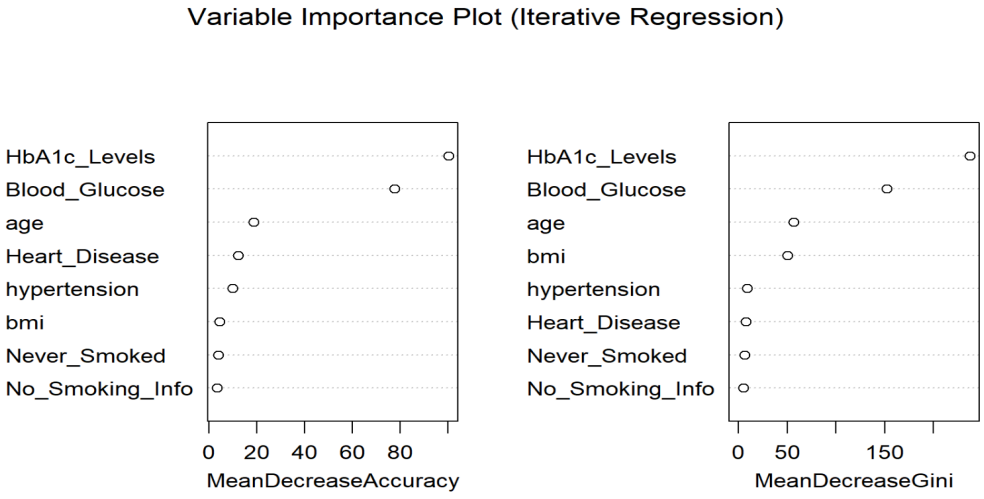
The next method in our analysis is KNN. The table below shows the test error rates for each KNN model. Similar to the KNN models in stage one, the test error rates fluctuate, which can be expected for KNN models given that test errors may not follow the typical "U" shape found in test MSEs. Though k equals 20 and 60 have nearly identical error rates, k equals 60 is preferred, as larger k decreases flexibility, leading to a smoother decision boundary. Below are the respective error rates and ROC curves for KNN models with k equal to 60 (AUC equals 0.922) and k equal to 20 (AUC equals 0.892).

K	3	5	10	20	60	80
Test Error Rate	0.055	0.056	0.051	0.049	0.049	0.050

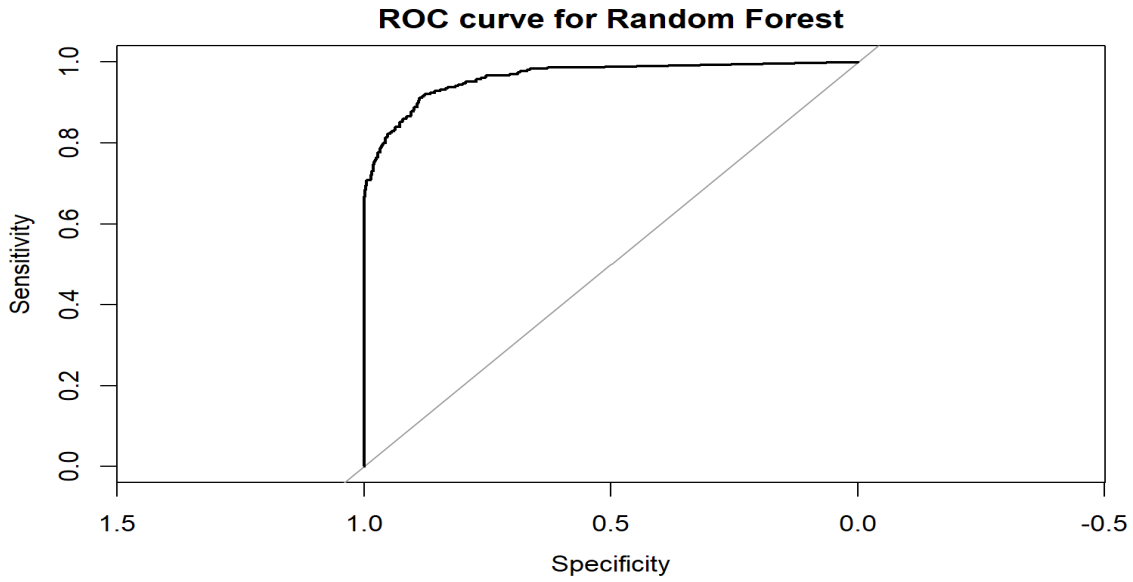


Given the higher AUC value of the model with k equal to 60, this supports the theoretical explanation of why this model outperforms k equal to 20 although both have the same test error rate.

The final method we implement for our data with iterative regression imputation is random forest. We have 15 possible predictors and by the threshold of p/3, we choose 5 predictors tried at each split. Our error rate is 0.023 and our out of bag error rate is 0.025, indicating that RF performs well since these values are similar. Below is a visualization of our variable importance plot.



Here we can observe HbA1c Levels and Blood Glucose Levels are the two most important variables, same as the RF model in stage one. Below is the ROC curve of our RF model with an AUC value of 0.9628.



Finally, we'll compare simple and interactive regression error rates and AUC values for all six methods with the table below. Ultimately, we can decipher a similar trend amongst these methods and across the two datasets using the error rate: KNN has the worst statistical performance, random forest has the best, while logistic regression is somewhere in the middle. In terms of AUC, RF and LR are similarly strong across datasets, while KNN produces a smaller, yet strong AUC in its own right.

Methods	**Simple imputation** **Logistic regression**	**Simple imputation** **KNN**	**Simple imputation** **Random Forest**	**Iterative regression Imputation** **Logistic regression**	**Iterative regression Imputation** **KNN**	**Iterative regression imputation** **Random forest**
Test Error Rate	0.0387	0.0484	0.0278	0.0346	0.049	0.023
AUC	0.9615	0.925	0.956	0.968	0.922	0.9628

But what differentiates these models and their respective statistical performance? Let's start with KNN. Considering we choose relatively large k values at k equal to 20 and 60, our analysis may suffer from high bias. Although our decision boundary is smoother as a result, its lack of flexibility resulted in a less statistically significant model in comparison to our other modes of classification. For LR, since we worked with a dataset that includes several highly important and normally distributed variables (i.e. Age, HbAlc levels, Blood Glucose Levels, BMI), it's performance was expectedly strong and a slight improvement over KNN. Lastly, we know RF performs particularly well when a select portion of the predictors strongly impact our model, which we can observe in the variable importance plot, where HbA1c Levels and Blood Glucose Levels are substantially affecting our response. In addition, since random forest utilizes a bootstrap sampling algorithm, variance was reduced compared to our other methods that are taken individually, and overall model performance was improved as a result.

Lastly, although both imputation methods show similar results in terms of error rate and AUC value between methods, differences are still evident. Based on our results, iterative regression imputation has lower error rates compared to simple imputation. Though simple imputation can account for missingness in an efficient manner, it doesn't capture the statistical power of iterative regression. Iterative regression imputation is a method that extends beyond simple imputation by imputing missingness utilizing the other predictors in the dataset and updating these variables over several iterations. By including more information as a means of imputing missingness and implementing several iterations to update the variables, we are producing a stronger means of imputation that was evident when comparing the error rates across our datasets.

Discussion

To conclude, we analyzed a dataset of medical patients and their diabetic status using KNN, random forest, and logistic regression models. In doing so, we created six models in total, with three models utilizing a dataset of simply imputed missing values, and three models implementing an iterative regression imputation dataset. In conducting our analysis, questions arose regarding model optimization and overall performance, more particularly why classification methods outperformed others, and where we can identify the differences between our models utilizing simple imputation and iterative imputation. Ultimately, our results concluded that HbA1c Levels and Blood Glucose Levels were the most important variables in both our simple imputation and iterative regression datasets. In addition, the random forest model proved to be the most statistically powerful model amongst the three classification methods for both datasets.

In going beyond the classification methods explored in this project, it may be worth exploring the statistical power of a support vector classifier (SVC) model in future analysis. Though it's difficult to say whether or not an SVC model would outperform any of our three methods given the nature of our dataset, it's worth extending to further classification models to assist in predicting diabetic status for future medical patients.

Appendix

```
#library settings
library(class)
library(randomForest)
library(nnet)
library(mltools)
library(data.table)
library(MASS)
library(glmnet)
library(ROCR)
library(smallstuff)
library(pROC)

# Preliminaries
df<- read.table("/Users/pzun0217/Desktop/Diabetes12.txt")
data <- df
data$HbA1c_level[is.na(data$HbA1c_level)] <- mean(data$HbA1c_level, na.rm = TRUE)
data <- data[!is.na(data$hypertension), ]
data <- data[!is.na(data$diabetes), ]
data$gender <- as.factor(data$gender)
data$smoking_history <- as.factor(data$smoking_history)
df1 <- one_hot(as.data.table(data),dropUnusedLevels = TRUE)
df1$diabetes <- as.factor(df1$diabetes)
df1$hypertension <- as.factor(df1$hypertension)
df1$heart_disease <- as.factor(df1$heart_disease)
colnames(df1)[11]<-"smoking_history_no_info"
colnames(df1)[12]<-"smoking_history_not_current"


#logistic for optimal model using forward backward selection for stage1
small1<- glm(diabetes~1,data = df1,family = binomial)
large1<- glm(diabetes~., data = df1,family = binomial)
forward<- step(small1, scope = list(upper = large1, lower = small1), method = "forward")


#optimal model for logistic regression and  error rate
y<- as.numeric(df1$diabetes) - 1
m1_final<- glm(diabetes~ HbA1c_level+blood_glucose_level+ age+
bmi+hypertension+heart_disease +smoking_history_no_info+
gender_Male+smoking_history_ever,data = df1,family = binomial )
pred_lr <- predict(m1_final, type = "response")
```

```
y_lr <- ifelse(pred_lr > 0.5, 1, 0)
ER_lr <- mean((y - as.numeric(y_lr))^2)
ER_lr

# KNN stage 1
set.seed(1234)
n <- nrow(df1)
train_ind <- sample(1:n, 0.8 * n)
train <- df1[train_ind, ]
test <- df1[-train_ind, ]

knn3 <- knn(as.matrix(train[, c(1:15)]), as.matrix(test[, c(1:15)]), cl = train$diabetes, k = 3,
prob = TRUE)
knn5 <- knn(as.matrix(train[, c(1:15)]), as.matrix(test[, c(1:15)]), cl = train$diabetes, k = 5,
prob = TRUE)
knn10 <- knn(as.matrix(train[, c(1:15)]), as.matrix(test[, c(1:15)]), cl = train$diabetes, k =
10, prob = TRUE)
knn20 <- knn(as.matrix(train[, c(1:15)]), as.matrix(test[, c(1:15)]), cl = train$diabetes, k =
20, prob = TRUE)
knn60 <- knn(as.matrix(train[, c(1:15)]), as.matrix(test[, c(1:15)]), cl = train$diabetes, k =
60, prob = TRUE)
knn80 <- knn(as.matrix(train[, c(1:15)]), as.matrix(test[, c(1:15)]), cl = train$diabetes, k =
80, prob = TRUE)

err3 <- mean(test$diabetes != knn3)
err5 <- mean(test$diabetes != knn5)
err10 <- mean(test$diabetes != knn10)
err20 <- mean(test$diabetes != knn20)
err60 <- mean(test$diabetes != knn60)
err80 <- mean(test$diabetes != knn80)

knn_result1<-data.frame(k = c(3,5,10,20,60,80),Val_error = c(err3, err5, err10, err20,
err60, err80))
knn_result1

# Random Forest stage1
set.seed(1234)
colnames(train)[1] <- "Female"
colnames(train)[2] <- "Male"
colnames(train)[3] <- "Non_Binary"
colnames(train)[6] <- "Heart_Disease"
colnames(train)[7] <- "Current_Smoker"
```

```
colnames(train)[8] <- "Has_Smoked"
colnames(train)[9] <- "Former_Smoker"
colnames(train)[10] <- "Never_Smoked"
colnames(train)[11] <- "No_Smoking_Info"
colnames(train)[12] <- "Not_Current_Smoker"
colnames(train)[14] <- "HbA1c_Levels"
colnames(train)[15] <- "Blood_Glucose"

colnames(test)[1] <- "Female"
colnames(test)[2] <- "Male"
colnames(test)[3] <- "Non_Binary"
colnames(test)[6] <- "Heart_Disease"
colnames(test)[7] <- "Current_Smoker"
colnames(test)[8] <- "Has_Smoked"
colnames(test)[9] <- "Former_Smoker"
colnames(test)[10] <- "Never_Smoked"
colnames(test)[11] <- "No_Smoking_Info"
colnames(test)[12] <- "Not_Current_Smoker"
colnames(test)[14] <- "HbA1c_Levels"
colnames(test)[15] <- "Blood_Glucose"

rf1<- randomForest(diabetes~.,data = train,mtry = 5,importance = TRUE)
varImpPlot(rf1, n.var = 8, main = "Variable Importance Plot")

pred_rf_1 <- predict(rf1, type = "response")
y_rf <- ifelse(as.numeric(pred_rf_1) > 0.5, 1, 0)
ER_rf <- mean((y - as.numeric(y_rf))^2)
ER_rf
pred_rf<- predict(rf1,test)
table(pred_rf,test$diabetes)
err_rf1<-mean(pred_rf!=test$diabetes)
err_rf1

#ROC curve and AUC value  for KNN, logistic,random forest
library(smallstuff)

ROCknn(knn20, test$diabetes)

pred_logit <- prediction(pred_lr, df1$diabetes)
perf_logit <- performance(pred_logit, "tpr", "fpr")
plot(perf_logit, colorize = TRUE, main = "ROC curve for Logistic Regression")
AUC_logit <- performance(pred_logit, "auc")@y.values[[1]]
```

```
AUC_logit

rf1.roc <- roc(train$diabetes, rf1$votes[,2])
plot.roc(rf1.roc, main = "ROC curve for Random Forest")
auc(rf1.roc)
varImpPlot(rf1, n.var = 8, main = "Variable Importance Plot")

#iterative regression stage2

#original dataset with one hot coding
df<- read.table("/Users/pzun0217/Desktop/Diabetes12.txt")
data1<-df
data1$smoking_history<- as.factor(as.factor(data1$smoking_history))
data1$gender<- as.factor(as.factor(data1$gender))
data1<- one_hot(as.data.table(data1))
data1$diabetes <- as.factor(data1$diabetes)
data1$hypertension <- as.factor(data1$hypertension)
data1$heart_disease <- as.factor(data1$heart_disease)

# one hot coding with missingness
new_data <- df
new_data$diabetes[is.na(new_data$diabetes)] <- 0
new_data$hypertension[is.na(new_data$hypertension)] <- 0
new_data$HbA1c_level[is.na(new_data$HbA1c_level)] <-
mean(new_data$HbA1c_level,na.rm= TRUE)

new_data$smoking_history<- as.factor(as.factor(new_data$smoking_history))
new_data$gender<- as.factor(as.factor(new_data$gender))
df2 <- one_hot(as.data.table(new_data))

df2$diabetes <- as.factor(df2$diabetes)
df2$hypertension <- as.factor(df2$hypertension)
df2$heart_disease <- as.factor(df2$heart_disease)

colnames(df2)[11]<-"smoking_history_no_info"
colnames(df2)[12]<-"smoking_history_not_current"

#iterative regression loop

n<- 10
for(i in 1:n){
  m_level<- lm(HbA1c_level~.,df2, subset=!is.na(data1$HbA1c_level))
```

```
  pred_level<- predict(m_level,df2[is.na(data1$HbA1c_level),])
  df2$HbA1c_level[is.na(data1$HbA1c_level)]<- pred_level

  library(nnet)
  m_hypertension<- multinom(hypertension~.,df2, subset=!is.na(data1$hypertension),trace
= FALSE)
  pred_hypertension<- predict(m_hypertension,df2[is.na(data1$hypertension),])
  df2$hypertension[is.na(data1$hypertension)]<- pred_hypertension

  m_diabetes<-multinom(diabetes~.,df2 ,subset=!is.na(data1$diabetes),trace=FALSE)
  pred_diabetes<- predict(m_diabetes,df2[is.na(data1$diabetes),])
  df2$diabetes[is.na(data1$diabetes)]<- pred_diabetes
}
#logistic for optimal model using forward backward selection

small2<- glm(diabetes~1,data = df2,family = binomial)
large2<- glm(diabetes~., data = df2,family = binomial)
step(small2, scope = list(upper = large2, lower = small2),
    method = "forward")

#optimal model for logistic regression and  error rate

m2_final<- glm(formula = diabetes ~ HbA1c_level + blood_glucose_level +
               age + bmi + heart_disease + hypertension + smoking_history_no_info,
             family = binomial, data = df2)
y2<- as.numeric(df2$diabetes) - 1
pred_lr2 <- predict(m2_final, type = "response")
y_lr2 <- ifelse(pred_lr2 > 0.5, 1, 0)
ER_lr2 <- mean((y2 - as.numeric(y_lr2))^2)
ER_lr2

# KNN stage 2

set.seed(4052)
n <- nrow(df2)
train_ind <- sample(1:n, 0.8 * n)
train2 <- df2[train_ind, ]
test2 <- df2[-train_ind, ]

new_knn3 <- knn(as.matrix(train2[, c(1:15)]), as.matrix(test2[, c(1:15)]), cl =
train2$diabetes, k = 3, prob = TRUE)
```

```
new_knn5 <- knn(as.matrix(train2[, c(1:15)]), as.matrix(test2[, c(1:15)]), cl =
train2$diabetes, k = 5,prob = TRUE)
new_knn10 <- knn(as.matrix(train2[, c(1:15)]), as.matrix(test2[, c(1:15)]), cl =
train2$diabetes, k = 10,prob = TRUE)
new_knn20 <- knn(as.matrix(train2[, c(1:15)]), as.matrix(test2[, c(1:15)]), cl =
train2$diabetes, k = 20,prob = TRUE)
new_knn60 <- knn(as.matrix(train2[, c(1:15)]), as.matrix(test2[, c(1:15)]), cl =
train2$diabetes, k = 60,prob = TRUE)
new_knn80 <- knn(as.matrix(train2[, c(1:15)]), as.matrix(test2[, c(1:15)]), cl =
train2$diabetes, k = 80,prob = TRUE)

new_err3 <- mean(test2$diabetes != new_knn3)
new_err5 <- mean(test2$diabetes != new_knn5)
new_err10 <- mean(test2$diabetes != new_knn10)
new_err20 <- mean(test2$diabetes != new_knn20)
new_err60 <- mean(test2$diabetes != new_knn60)
new_err80 <- mean(test2$diabetes != new_knn80)

knn_result1<-data.frame(k = c(3,5,10,20,60,80),Val_error = c(new_err3, new_err5,
new_err10,new_err20,new_err60,new_err80))
knn_result1

#random forest stage2
rf2<- randomForest(diabetes~.,data = train2,mtry = 5,importance = TRUE)
pred_rf2<- predict(rf2,test2)
table(pred_rf2,test2$diabetes)
err_rf2<-mean(pred_rf2!=test2$diabetes)
err_rf2
varImpPlot(rf2,main = "random forest variable importance plot using iterative regression")

#ROC curve and AUC value  for KNN, logistic,random forest
par(mfrow  = c(1,2))
ROCknn(new_knn60, test2$diabetes)
ROCknn(new_knn20,test2$diabetes)

pred_lr2 <- predict(m2_final, type = "response")
pred_logit2 <- prediction(pred_lr2, df2$diabetes)
perf_logit2 <- performance(pred_logit2, "tpr", "fpr")
plot(perf_logit2, colorize = TRUE, main = "ROC curve for Logistic
Regression")
AUC_logit <- performance(pred_logit2, "auc")@y.values[[1]]
AUC_logit
```

```
library(pROC)
rf2.roc <- roc(train2$diabetes, rf2$votes[,2])
plot.roc(rf2.roc, main = "ROC curve for Random Forest")
par(mfrow = c(1,2))
auc(rf2.roc)

#plot comparsion for simple and iterative regression imputation
par(mfrow=c(1,2))
hist(df1$HbA1c_level,breaks=20,main="Imputed data for simple imputation for hbA1c
level",xlab="HbA1c level",freq=FALSE)
hist(df2$HbA1c_level,breaks=20,main="Imputed data for iterative regression hbA1c
level",xlab="HbA1c level",freq=FALSE)
```