Final Report

Will Gullion, Brian Nalley, Nate Nellis, McKade Thomas

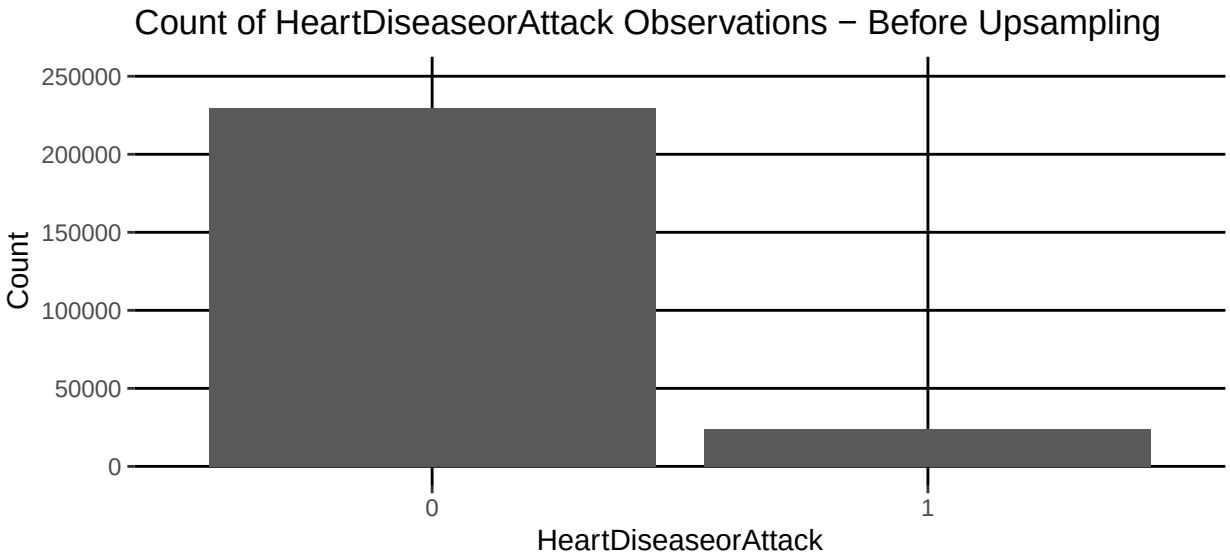
STAT 5650

Introduction

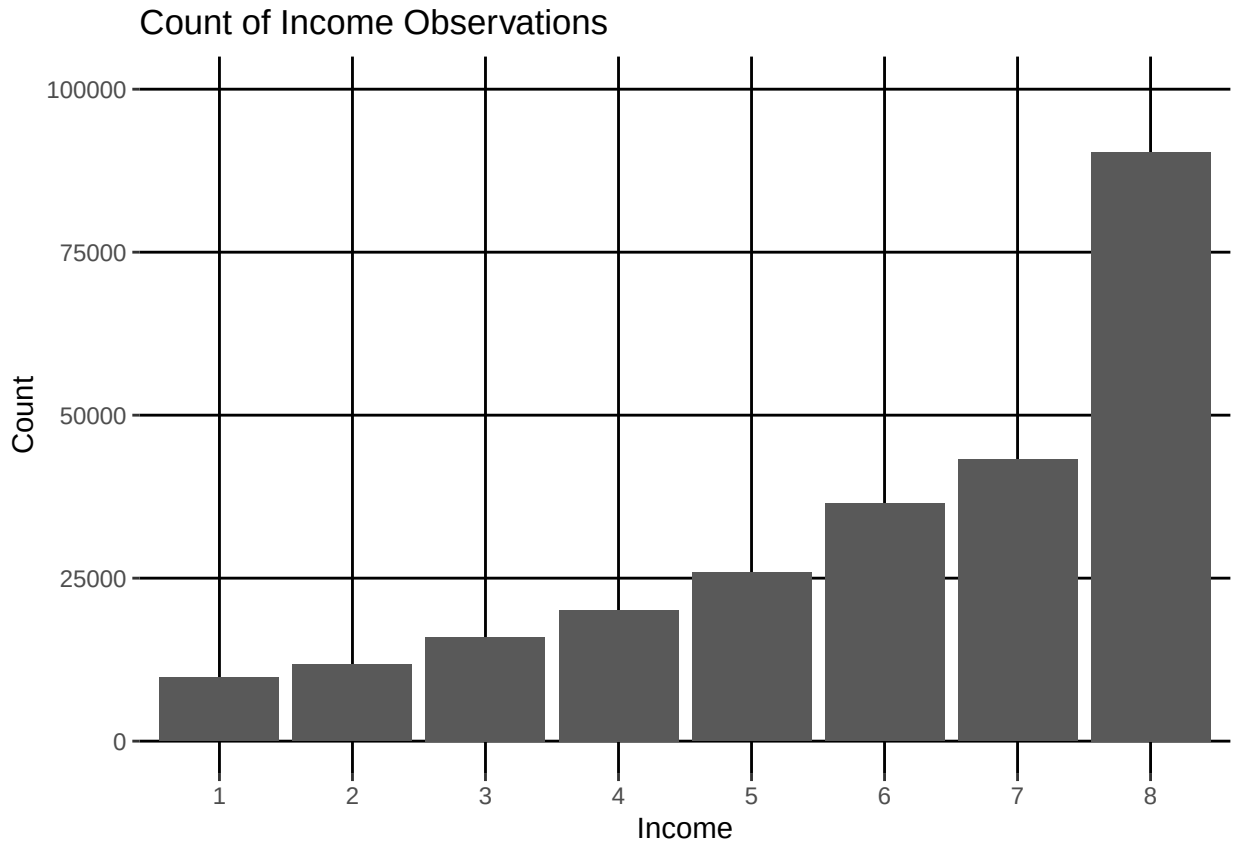
We chose to focus on data having to do with heart disease/heart attack indicators to understand which factors impact a person's probability to have a heart attack or suffer from strong levels of heart disease. The data was obtained from Kaggle and was modified from a CDC survey. See the *Data Dictionary* section for more details at the end of the document. The dataset contains 252,680 observations with the response variable being binary, an indicator of whether the individual suffered from a heart attack or not. There are a total of 20 available predictor variables.

Exploratory Data Analysis

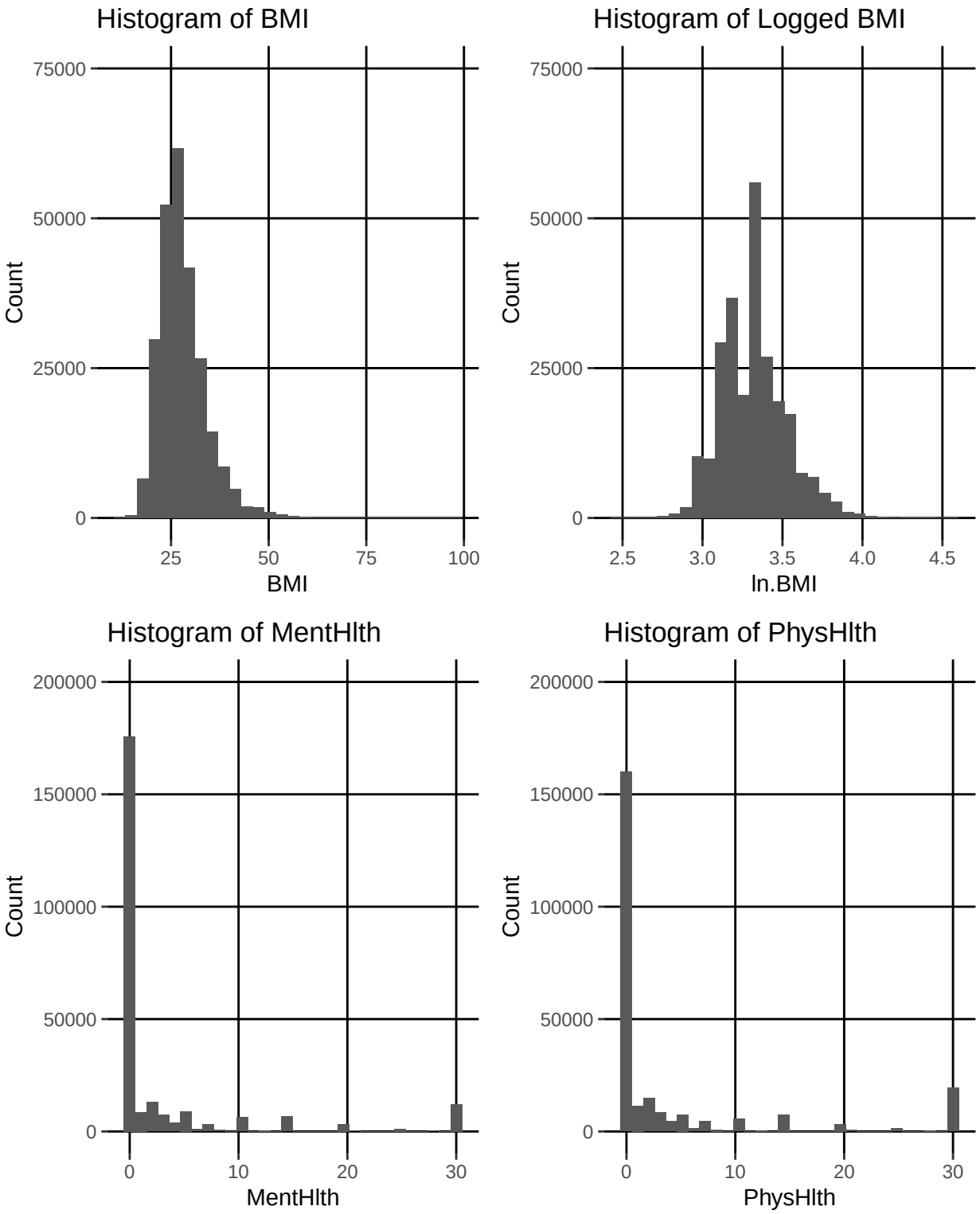
Taking a closer look at our response variable, we noticed that only 9.4% of participants in the survey had heart disease or a heart attack, 23,893 responded yes versus 229,787 who responded no. This means that we could just predict a 0 for all observations and we could get an accuracy of over 90%. Since we aren't concerned in this instance with a high accuracy, but instead sensitivity (getting accurate results when someone is at risk of a heart attack or heart disease), we utilized Upsampling in some of our methods to create synthetic instances of the positive class. The details of this method are explained in a later section. This approach decreased overall accuracy but resulted in a much better prediction sensitivity, finding those people with heart disease or who had a heart attack. Even with a loss in overall accuracy, we felt that this was a wise move as predicting someone who has heart disease is much more important than identifying someone who doesn't have heart disease. See the below bar chart to compare the two categories visually.



When it comes to the predictor variables there are 12 binary, 3 numerical, and 5 factor variables. Having fewer numerical variables is less than desirable, especially when it comes to how some of the factor variables are coded/binned. An example of this terrible survey coding is the **Income** variable where the CDC survey sets the highest bucket capped at $75,000+. Due to how low that is for the maximum response allowed, a full third of the observations fall into that category.



For the three numerical variables, **BMI**, **MentHlth**, and **PhysHlth**, there is a definite skewness that we can see in their distributions. This is especially true for **PhysHlth** and **MentHlth**, which are how many times in the last 30 days a survey respondent reported having poor physical or mental health respectively. A huge percentage of respondents reported zero days, 70% for **MentHlth** and 63% for **PhysHlth**. To correct for these skewed variables, we logged **BMI** to enforce normality constraints. For **MentHlth** and **PhysHlth** we created binary variables where any value greater than 0 was coded as a 1 and was coded as a 0 otherwise. We originally attempted to log these variables but the skew wasn't reduced very much and we ended up finding that the binary variables worked out much better. These transformations were valuable in the various classification methods that we tried, even though they did reduce our numerical variables to really just the **ln.BMI** variable.



See more information on the variables in the **Data Dictionary** Section at the end of the document.

Feature subsets from RF and LASSO

To determine which variables were most important to our analysis, we performed variable selection both through random forest and LASSO logistic regression. The random forest determined that **Age**, **GenHlth**, **ln.BMI**, **Income**, **HighBP**, **Education** and **HighChol** were the most important variables. On the other hand, LASSO preferred **HighBP**, **HighChol**, **CholCheck**, **Smoker**, **Stroke**, **Diabetes**, **HvyAlcoholConsump**, **NoDocbcCost**, **GenHlth**, **DiffWalk**, **Sex**, **Age**, **Income** and **bin.PhysHlth**. While there is some overlap between the two, LASSO's approach selected quite a few more variables as being important to the analysis.

Upsampling to Handle Class Imbalance

As previously mentioned, one of the difficulties with this dataset was the class imbalance in the response, since those that had heart attacks were outnumbered 10 to 1. This resulted in poor sensitivity in many of the original methods we attempted even with feature selection from Lasso and Random Forest. to improve the model's ability to accurately classify the positive class, we performed upsampling using caret to create new synthetic instances of the positive class. This resulted in a dataset that had more balanced values of the response variable. We then used this upsampled dataset along with the most importanct features from random forest and LASSO to perform an analysis.

Modeling

After identifying what features would be most helpful for predicting heart disease or attack and handling the class imbalance in our response, we used a variety of modeling strategies for prediction. In general, we paritioned the data using a 70% split for training and test sets and evaluated the model performance based on the test data (after cross-validation).

Survey of Methods

Below is an explanation of the various methods that we tried out on our dataset. We were able to perform most of the methods discussed so far in class, although KNN wasn't feasible due to the large number of observations in our dataset, creating computational intensity beyond the time constraint of the project. The rest of our methods will be discussed in the subsections below with a summarized table afterwards.

LDA

First, we conducted linear discriminant analysis (LDA) on our health data using the random forest derived variables and obtained a cross-validated predictive accuracy of 73.57% and a sensitivity of 48.36%. Cross-validated predictive accuracy increased slightly while sensitivity decreased quite a bit, to 76.81% and 20.90% respectively using the LASSO selected subset of predictive variables.

QDA

We also conducted quadratic discriminant analysis (QDA), though we were unable to compute any results for the random forest selected variables. However, we were able to obtain results for the LASSO preferred dependent variables. Here, the cross-validated percent correctly classified is relatively low at 70.94% and the cross-validated sensitivity is also quite low at 10.10%.

LASSO Logistic Regression

In addition to variable selection, we also used LASSO logistic regression to make predictions on our test set. We first compared the minimum and 1-SE values for lambda, for which the test set predictions were nearly identical between the two models, each with a percent correctly classified of just north of 77%. However, the 1-SE version selected fewer variables as important compared with the minimum model which selected all variables except for **AnyHealthcare** and **ln.BMI**. Because the 1-SE version gave us a more parsimonious model for approximately equivalent prediction accuracy, we chose to proceed with this version both for variable selection as well as to make predictions on our test set.

We then set out to make predictions with both the these variables as well as those which our random forest model determined were most important. The cross-validated percent correctly classified was respectable at 76.68% when using the LASSO variables; however, the sensitivity was rather low at only 21.36%. The results with the random forest variables are roughly equivalent with a cross-validated percent correctly classified of 73.91% with a still low 22.10% sensitivity.

Random Forests

We built the random forest using the predictors that the variable importance plot indicated had the largest effects on the random forest's classification ability. Those variables ended up being **Age**, **GenHlth**, **ln.BMI**, **Income**, **HighBP**, **Education**, and **HighChol**. After creating the random forest, we found that it classified approximately 75% of the training set correctly. The next step was to validate the random forest using the testing set. We saw similar performance, with just under 75% correctly classified. Perhaps more important than the PCC was the sensitivity and specificity which came in at 79% and 70%, respectively. We see a huge increase in the sensitivity, thanks to our upsampling technique.

Gradient Boosting

Using the same predictors determined important by Random Forest, the gradient boosting models with and without tuning performed fairly well on the test data and similar to random forest, resulting in about 75% correctly classified percentage on the test data and 74% sensitivity. In general for this model, the features selected by random forest resulted in higher performance on the test data.

SVM

The next model, support vector machines, after tuning performed similar to gradient boosting with tuning with an overall classification accuracy of 75% and specificity of 70% for the RF selected features. However, the model outperformed nearly all other methods once trained on the LASSO select features, with a classification accuracy of 77% and sensitivity of 75%. The only model to outperform SVM was random forest in terms of classification accuracy, though SVM also outperformed all other models in terms of sensitivity.

Summary Table

Table 1: Prediction Results for Individual Methods

	PCC	Specificity	Sensitivity
RF w/ LASSO-Selected Variables	79.11	74.45	87.36
SVM Tuned w/ LASSO-Selected Variables	77.60	75.43	79.90
LDA w/LASSO-Selected Variables	76.81	74.63	20.90
LASSO Logistic Regression w/RF-Selected Variables	76.68	74.78	21.36
GBM Tuned w/ LASSO-Selected Variables	76.10	73.72	78.52
SVM Tuned w/ RF-Selected Variables	75.40	70.03	78.92
GBM Tuned w/ RF-Selected Variables	75.11	72.77	77.76
RF w/ RF-Selected Variables	74.73	69.69	82.00
LASSO Logistic Regression w/LASSO-Selected Variables	73.91	69.81	22.10
LDA w/RF-Selected Variables	73.57	48.60	48.36
QDA w/LASSO-Selected Variables	70.94	52.63	10.10
QDA w/RF-Selected Variables	NA	NA	NA

Conclusion

There were many modeling strategies that proved effective in predicting heart disease or attack. Using random forest and LASSO, we identified features that were considered most important for prediction. Next, we handled the class imbalance in the response using upsampling. After taking these measures and performing necessary transformations on the predictor variables, there were several models that performed with about 80% overall classification accuracy as well as 79% sensitivity and 75% specificity.

The models that achieved these results were random forest, support vector machines (with tuning) and LASSO logisitic regression. Additionally, the features deemed most important by LASSO resulted in higher performance than the random forest selected features on average. The other modeling strategies we employed were not too far behind in terms of performance, but also tended to over or under fit the data.

Data Dictionary

Data found by us at https://www.kaggle.com/datasets/alexteboul/heart-disease-health-indicators-dataset

Description of variables found in https://www.cdc.gov/brfss/annual_data/2015/pdf/codebook15_llcp.pdf

Response Variable

HeartDiseaseorAttack, binary

```
##      0      1
## 229787  23893
```

Predictor Variables

HighBP, binary
- Adults who have been told they have high blood pressure by a doctor, nurse, or other health professional

```
##      0      1
## 144851 108829
```

HighChol, binary
- Have you EVER been told by a doctor, nurse or other health professional that your blood cholesterol is high?

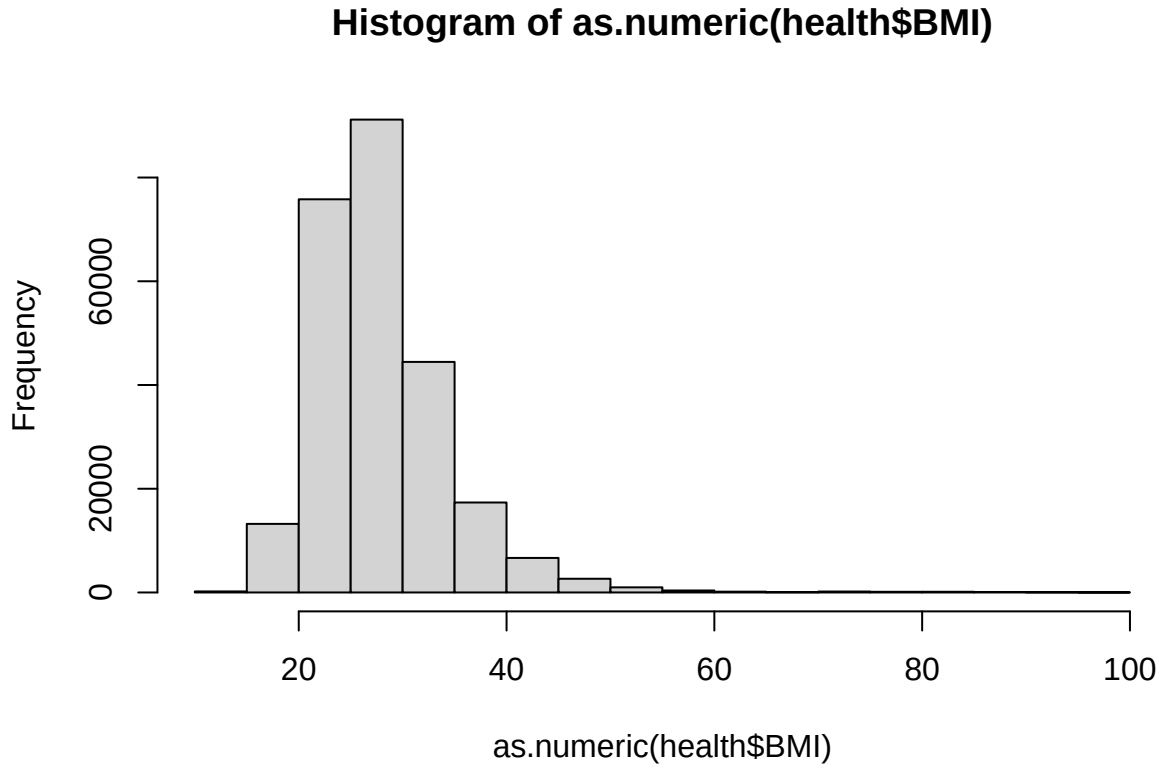
```
##      0      1
## 146089 107591
```

CholCheck, binary
- Cholesterol check within past five years

```
##      0      1
##   9470 244210
```

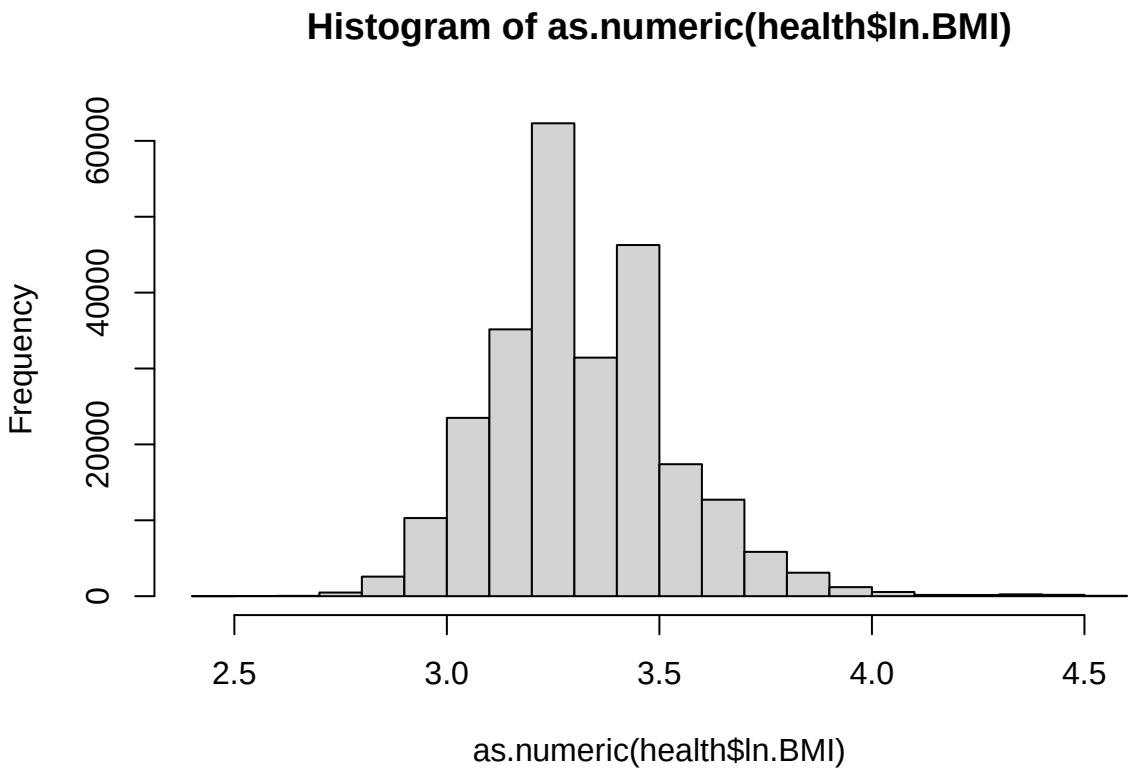
BMI, numerical

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.00   24.00   27.00   28.38   31.00   98.00
```



ln.BMI, numerical
- Logged version of the BMI variable to reduce the impact of outliers and enforce better normality contstraints.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.485   3.178   3.296   3.323   3.434   4.585
```



Smoker, binary
- Have you smoked at least 100 cigarettes in your entire life?

```
##      0      1
## 141257 112423
```

Stroke, binary

```
##      0      1
## 243388  10292
```

Diabetes, factor
- 0 is no diabetes,
- 1 is pre-diabetes
- 2 is diabetes

```
##      0      1      2
## 213703   4631  35346
```

PhysActivity, binary
- During the past month, other than your regular job, did you participate in any physical activities or exercises such as running, calisthenics, golf, gardening, or walking for exercise?

```
##      0      1
##  61760 191920
```

Fruits, binary
- Consume Fruit 1 or more times per day

```
##      0      1
##  92782 160898
```

Veggies, binary
- Consume Vegetables 1 or more times per day

```
##      0      1
##  47839 205841
```

HvyAlcoholConsum, binary
- Heavy drinkers (adult men having more than 14 drinks per week and adult women having more than 7 drinks per week)

```
##      0      1
## 239424  14256
```

AnyHealthcare, binary
- Do you have any kind of health care coverage, including health insurance, prepaid plans such as HMOs, or government plans such as Medicare, or Indian Health Service?

```
##      0      1
##  12417 241263
```

NoDocbcCost, binary
- Was there a time in the past 12 months when you needed to see a doctor but could not because of cost?

```
##      0      1
## 232326  21354
```

GenHlth, factor
- Would you say that in general your health is:
- 1 = Excellent
- 2 = Very Good
- 3 = Good
- 4 = Fair
- 5 = Poor

```
##     1     2     3     4     5
## 45299 89084 75646 31570 12081
```

MentHlth, numerical
- Now thinking about your mental health, which includes stress, depression, and problems with emotions, for how many days during the past 30 days was your mental health not good?

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   0.000   3.185   2.000  30.000
```

bin.MentHlth, binary
- Binary version of **MentHlth** where 0 days is coded as 0 and any number of days is coded as 1.

```
##      0      1
## 175680  78000
```

PhysHlth, numerical
- Now thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good?

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   0.000   4.242   3.000  30.000
```

bin.PhysHlth, binary
- Binary version of **PhysHlth** where 0 days is coded as 0 and any number of days is coded as 1.

```
##      0      1
## 160052  93628
```

DiffWalk, binary
- Do you have serious difficulty walking or climbing stairs?

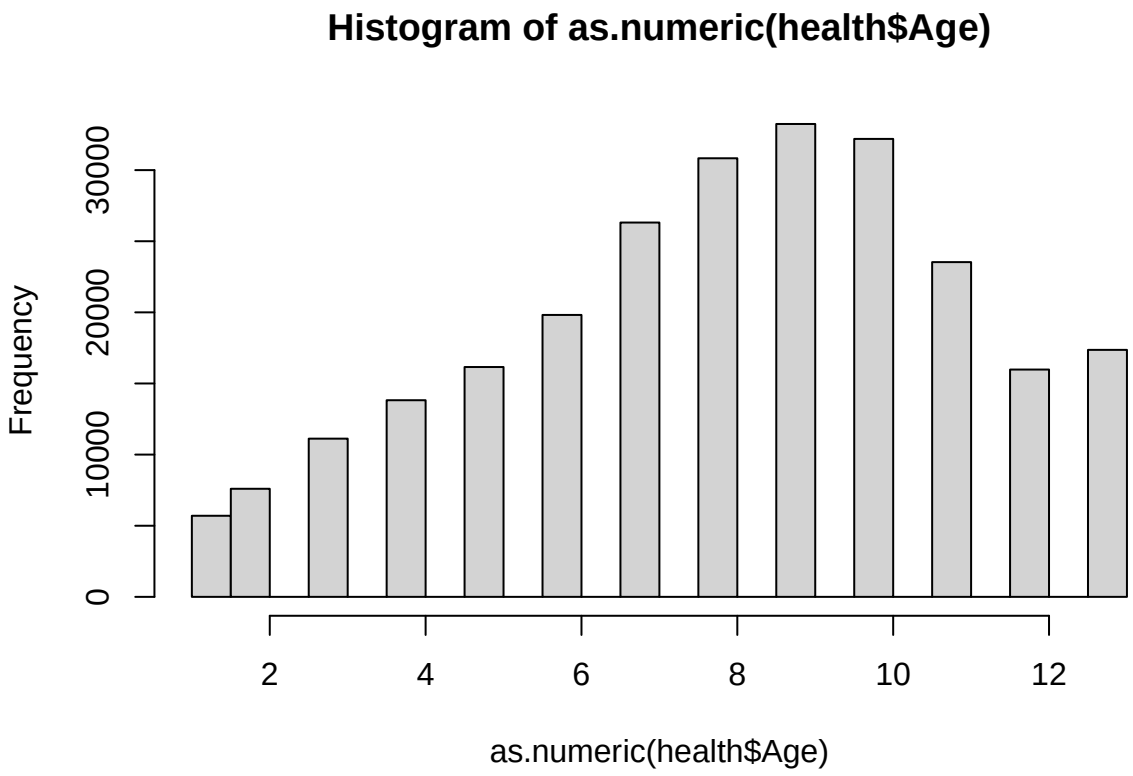
```
##      0      1
## 211005  42675
```

Sex, binary
- 0 = Female
- 1 = Male

```
##      F      M
## 141974 111706
```

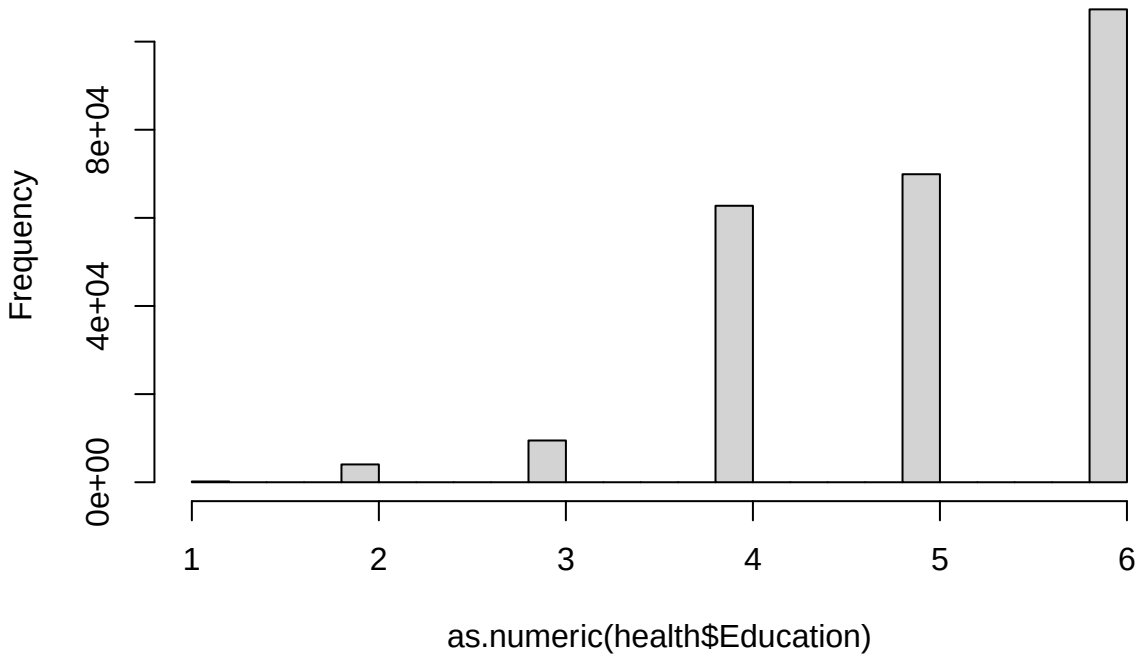
Age, factor
- 1 Age 18 to 24
- 2 Age 25 to 29
- 3 Age 30 to 34
- 4 Age 35 to 39
- 5 Age 40 to 44
- 6 Age 45 to 49
- 7 Age 50 to 54
- 8 Age 55 to 59
- 9 Age 60 to 64
- 10 Age 65 to 69
- 11 Age 70 to 74
- 12 Age 75 to 79
- 13 Age 80 or older
- 14 Don't Know / Refused to answer (I removed these as well)

```
##     1     2     3     4     5     6     7     8     9    10    11    12    13
##  5700  7598 11123 13823 16157 19819 26314 30832 33244 32194 23533 15980 17363
```



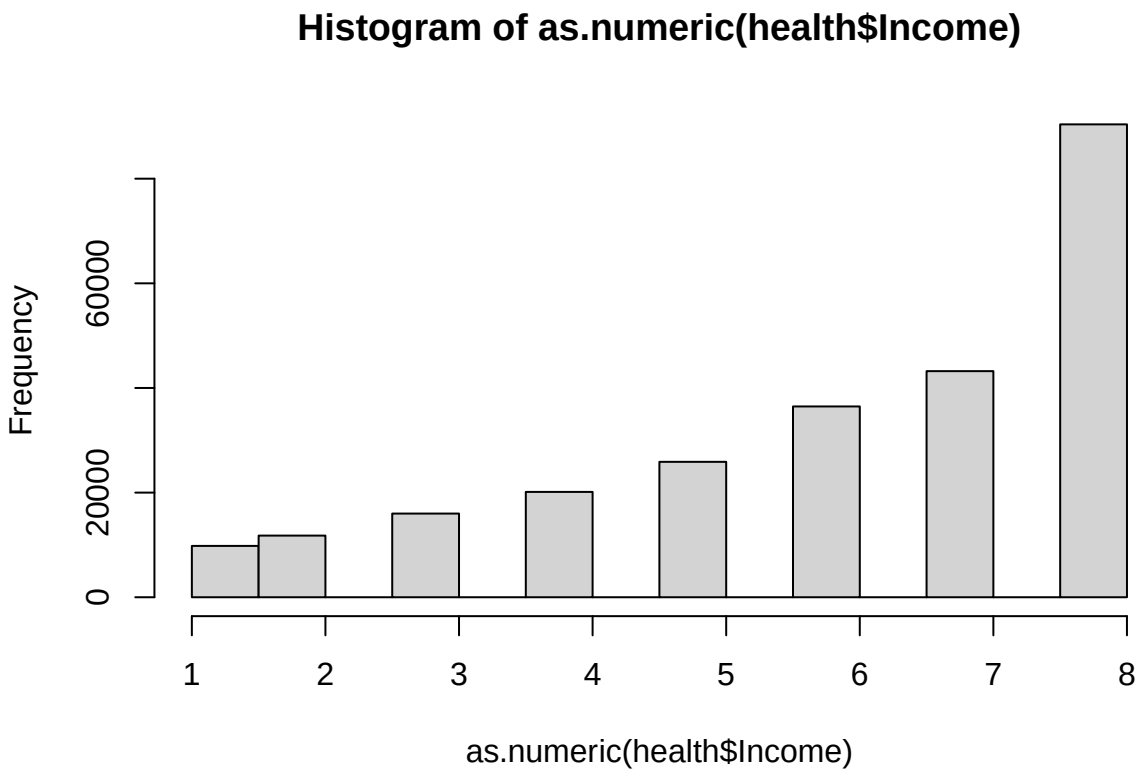
Education, factor
- What is the highest grade or year of school you completed?
- 1 Never attended school or only kindergarten
- 2 Grades 1 through 8 (Elementary)
- 3 Grades 9 through 11 (Some high school)
- 4 Grade 12 or GED (High school graduate)
- 5 College 1 year to 3 years (Some college or technical school)
- 6 College 4 years or more (College graduate)

```
##      1      2      3      4      5      6
##    174   4043   9478  62750  69910 107325
```



Income, factor
- Is your annual household income from all sources
- 1 Less than $10,000
- 2 Less than $15,000 ($10,000 to less than $15,000)
- 3 Less than $20,000 ($15,000 to less than $20,000)
- 4 Less than $25,000 ($20,000 to less than $25,000)
- 5 Less than $35,000 ($25,000 to less than $35,000)
- 6 Less than $50,000 ($35,000 to less than $50,000)
- 7 Less than $75,000 ($50,000 to less than $75,000)
- 8 $75,000 or more

```
##     1     2     3     4     5     6     7     8
##  9811 11783 15994 20135 25883 36470 43219 90385
```



Appendix: All code for this report

```
knitr::opts_chunk$set(echo = TRUE)
library(dplyr)
library(MASS)
library(verification)
library(caTools)
library(klaR)
library(randomForest)
library(glmnet)
library(ggplot2)
library(knitr)
library(gridExtra)
library(kableExtra)
library(caret)
source('kappa_and_class_sum.R')
health <- read.csv("heart_disease_health_indicators.csv")

health <- health %>% mutate(
  HeartDiseaseorAttack = as.factor(HeartDiseaseorAttack),
  HighBP = as.factor(HighBP),  # LASSO approved
  HighChol = as.factor(HighChol),  # LASSO approved
  CholCheck = as.factor(CholCheck),
```

```
  ln.BMI = log(BMI),  # RF approved
  Smoker = as.factor(Smoker),  # LASSO approved
  Stroke = as.factor(Stroke),  # LASSO 1-SE only, RF approved
  Diabetes = as.factor(Diabetes),  # LASSO approved, RF approved
  PhysActivity = as.factor(PhysActivity),
  Fruits = as.factor(Fruits),
  Veggies = as.factor(Veggies),
  HvyAlcoholConsump = as.factor(HvyAlcoholConsump),
  AnyHealthcare = as.factor(AnyHealthcare),
  NoDocbcCost = as.factor(NoDocbcCost),  # LASSO approved
  GenHlth = as.factor(GenHlth),  # LASSO approved, RF approved
  ln.MentHlth = log(MentHlth + 1), # Probably shouldn't use
  bin.MentHlth = as.factor(ifelse(MentHlth == 0, 0, 1)),  #RF approved
  fac.MentHlth = as.factor(MentHlth), # Probably shouldn't use
  ln.PhysHlth = log(PhysHlth + 1), # Probably shouldn't use
  bin.PhysHlth = as.factor(ifelse(PhysHlth == 0, 0, 1)),  # LASSO approved, RF approved
  fac.PhysHlth = as.factor(PhysHlth), # Probably shouldn't use
  DiffWalk = as.factor(DiffWalk),  # LASSO , RF approved
  Sex = as.factor(ifelse(Sex == 0, "F", "M")),  # LASSO approved, RF approved
  Age = as.factor(Age),  # LASSO approved, RF approved
  Education = as.factor(Education),  # RF approved
  Income = as.factor(Income)  # LASSO approved, RF approved
)


best_features <- c(1:4, 6:15, 18:23, 25, 28)
health_subset <- health[,best_features]


# Train and Test Sets
health2_split <- sample.split(health_subset, SplitRatio = 0.3)
health_test <- subset(health_subset, health2_split == TRUE)
health_train <- subset(health_subset, health2_split == FALSE)

ggplot(health, aes(HeartDiseaseorAttack)) +
  geom_bar() +
  scale_y_continuous(limits = c(0, 250000), breaks = seq(0, 250000, by = 50000)) +
  ylab("Count") +
  ggtitle("Count of HeartDiseaseorAttack Observations - Before Upsampling") +
  theme(panel.grid.major = element_line(colour = "black"),
        panel.background = element_rect(fill = NA))
ggplot(health, aes(Income)) +
  geom_bar() +
  scale_y_continuous(limits = c(0, 100000), breaks = seq(0, 100000, by = 25000)) +
  ylab("Count") +
  ggtitle("Count of Income Observations") +
  theme(panel.grid.major = element_line(colour = "black"),
        panel.background = element_rect(fill = NA))
grid.arrange(
ggplot(health, aes(BMI)) +
  geom_histogram(bins=30) +
  scale_y_continuous(limits = c(0, 75000), breaks = seq(0, 75000, by = 25000)) +
  ylab("Count") +
```

```
  ggtitle("Histogram of BMI") +
  theme(panel.grid.major = element_line(colour = "black"),
        panel.background = element_rect(fill = NA))
,
ggplot(health, aes(ln.BMI)) +
  geom_histogram(bins=30) +
  scale_y_continuous(limits = c(0, 75000), breaks = seq(0, 75000, by = 25000)) +
  ylab("Count") +
  ggtitle("Histogram of Logged BMI") +
  theme(panel.grid.major = element_line(colour = "black"),
        panel.background = element_rect(fill = NA))
,
nrow = 1)
grid.arrange(
ggplot(health, aes(MentHlth)) +
  geom_histogram(bins=30) +
  scale_y_continuous(limits = c(0, 200000), breaks = seq(0, 200000, by = 50000)) +
  ylab("Count") +
  ggtitle("Histogram of MentHlth") +
  theme(panel.grid.major = element_line(colour = "black"),
        panel.background = element_rect(fill = NA))
,
ggplot(health, aes(PhysHlth)) +
  geom_histogram(bins=30) +
  scale_y_continuous(limits = c(0, 200000), breaks = seq(0, 200000, by = 50000)) +
  ylab("Count") +
  ggtitle("Histogram of PhysHlth") +
  theme(panel.grid.major = element_line(colour = "black"),
        panel.background = element_rect(fill = NA))
,
nrow = 1)
## UpSample to deal with class imbalance
new_train <- upSample(health_train[,2:22], as.factor(health_train[,1]), list=FALSE) %>%
  mutate(HeartDiseaseorAttack = as.numeric(Class)-1)

# new_train$HeartDiseaseorAttack <- new_train$Class
# new_train <- new_train[,-22]

## Test Set Upsample ##
new_test <- upSample(health_test[,2:22], as.factor(health_test[,1]), list = FALSE) %>%
  mutate(HeartDiseaseorAttack = as.numeric(Class)-1)
# new_test$HeartDiseaseorAttack <- new_test$Class
# new_test <- new_test[,-22]

rf_best_features <- new_train %>% dplyr::select(HeartDiseaseorAttack,
                                                Age, GenHlth, ln.BMI, Income,
                                                HighBP, Education, HighChol)

rf_test_features <- new_test %>% dplyr::select(HeartDiseaseorAttack, Age,
                                               GenHlth, ln.BMI, Income,
                                               HighBP, Education, HighChol)

lasso_best_features <- new_train %>% dplyr::select(HeartDiseaseorAttack, HighBP,
```

```
                                                    HighChol, CholCheck, Smoker,
                                                    Stroke, Diabetes, HvyAlcoholConsump,
                                                    NoDocbcCost, GenHlth, DiffWalk,
                                                    Sex, Age, Income, bin.PhysHlth)

lasso_test_features <- new_test %>% dplyr::select(HeartDiseaseorAttack, HighBP,
                                                    HighChol, CholCheck, Smoker,
                                                    Stroke, Diabetes, HvyAlcoholConsump,
                                                    NoDocbcCost, GenHlth, DiffWalk,
                                                    Sex, Age, Income, bin.PhysHlth)
set.seed(89729)
train_sample <- sample_n(rf_best_features, 10000)
test_sample <- sample_n(rf_test_features, 10000)

rf <- randomForest(as.factor(HeartDiseaseorAttack) ~ .,
                   data = train_sample, n.trees = 100, importance = TRUE)

class.sum(train_sample$HeartDiseaseorAttack,
          predict(rf, type="prob")[,2], cap = "Training Data RF Performance")
class.sum(test_sample$HeartDiseaseorAttack,
          predict(rf, test_sample, type="prob")[,2], cap = "Test Data RF Performance")

# Random Forest with LASSO selected variables
train_sample.lasso <- sample_n(lasso_test_features, 10000)
test_sample.lasso <- sample_n(lasso_test_features, 10000)

rf.lasso <- randomForest(as.factor(HeartDiseaseorAttack) ~ ., data = train_sample.lasso, n.trees = 100)

class.sum(train_sample.lasso$HeartDiseaseorAttack, predict(rf.lasso, type="prob")[,2], cap = "Training
class.sum(test_sample.lasso$HeartDiseaseorAttack, predict(rf.lasso, test_sample.lasso, type="prob")[,2]

## Performance on K Folds CV (10 Folds)
gbm.xvalpr = rep(0, nrow(train_sample))
xvs=rep(1:10, length = nrow(train_sample))
xvs=sample(xvs)
for(i in 1:10){
      train = train_sample[xvs!=i,]
      test = train_sample[xvs==i,]
      glub=gbm(HeartDiseaseorAttack ~ . ,distribution="bernoulli",
               n.trees = 50, data=train)
      gbm.xvalpr[xvs==i] = predict(glub, newdata = test,
                                   type = "response", n.trees=50)
}

class.sum(train_sample$HeartDiseaseorAttack, gbm.xvalpr,
          cap = "CV ADA Performance")

## Performance on Test Set
health.gbm = gbm(HeartDiseaseorAttack ~ . ,distribution="bernoulli",
                 n.trees=50, data=train_sample)
class.sum(test_sample$HeartDiseaseorAttack,
          predict(health.gbm, newdata = test_sample,
                  type = "response", n.trees = 50),
```

```
          cap = "Test Data ADA Performance")

## Performance on Test Set
health.gbm = gbm(HeartDiseaseorAttack ~ . ,distribution="bernoulli",
                 n.trees=50, data=train_sample.lasso)
class.sum(test_sample.lasso$HeartDiseaseorAttack,
          predict(health.gbm, newdata = test_sample.lasso,
                  type = "response", n.trees = 50),
          cap = "Test Data ADA Performance")
x = as.matrix(train_sample[,-1])
y = as.vector(train_sample[,1])

health.svm.tune <- eztune(x, y, method="svm", fast = TRUE, cross = 5)
health.svm.tune
## Best hyperparamters:
#   gamma:  0.03125
#   cost: 32
#   loss: 0
#   n: 200

## Performance on K Folds CV (10 Folds)
svm.xvalpred = rep(0, nrow(train_sample))
xvs = rep(1:10, length = nrow(train_sample))
xvs = sample(xvs)
for(i in 1:10){
      train = train_sample[xvs!=i,]
      test = train_sample[xvs==i,]
      glub = svm(as.factor(HeartDiseaseorAttack) ~ . ,
                 probability=TRUE,
                 gamma = 0.03125,
                 cost = 32,
                 loss = 0,
                 n = 200,
                 data = train)
      svm.xvalpred[xvs==i] = attr(predict(glub, test, probability = TRUE),
                                  "probabilities")[,2]
}

class.sum(train_sample$HeartDiseaseorAttack, svm.xvalpred,
          cap = "CV SVM Performance")

## Performance on Test Set
health.svm = svm(as.factor(HeartDiseaseorAttack)~ . ,
                 probability = TRUE,
                 data = train_sample,
                 gamma = 0.03125,
                 cost = 32,
                 loss = 0,
                 n = 200,)
health.svm.resubpred=predict(health.svm, test_sample, probability=TRUE)
class.sum(test_sample$HeartDiseaseorAttack,
          attr(health.svm.resubpred,"probabilities")[,2],
          cap = "Test Data SVM Performance")
```

```
results <- data.frame(PCC = c(73.57, 76.81, NA, 70.94, 73.91, 76.68,
                              74.73, 79.11, 75.11, 76.10, 75.40, 77.60),
                      Specificity = c(48.60, 74.63, NA, 52.63, 69.81,
                                      74.78, 69.69,74.45,72.77,73.72,70.03, 75.43),
                      Sensitivity = c(48.36, 20.90, NA, 10.10, 22.10, 21.36,
                                      82.00, 87.36, 77.76, 78.52, 78.92, 79.90))

rownames(results) <- c("LDA w/RF-Selected Variables",
                       "LDA w/LASSO-Selected Variables",
                       "QDA w/RF-Selected Variables",
                       "QDA w/LASSO-Selected Variables",
                       "LASSO Logistic Regression w/LASSO-Selected Variables",
                       "LASSO Logistic Regression w/RF-Selected Variables",
                       "RF w/ RF-Selected Variables",
                       "RF w/ LASSO-Selected Variables",
                       "GBM Tuned w/ RF-Selected Variables",
                       "GBM Tuned w/ LASSO-Selected Variables",
                       "SVM Tuned w/ RF-Selected Variables",
                       "SVM Tuned w/ LASSO-Selected Variables")

results <- results[order(results$PCC,decreasing=TRUE),]

results.table <- results %>%
  kbl(caption = "Prediction Results for Individual Methods") %>%
  kable_classic(full_width = FALSE, html_font = "Cambria",
                latex_options = "HOLD_position")

results.table
summary(health$HeartDiseaseorAttack)
summary(health$HighBP)
summary(health$HighChol)
summary(health$CholCheck)
summary(health$BMI)
hist(as.numeric(health$BMI))
summary(health$ln.BMI)
hist(as.numeric(health$ln.BMI))
summary(health$Smoker)
summary(health$Stroke)
summary(health$Diabetes)
summary(health$PhysActivity)
summary(health$Fruits)
summary(health$Veggies)
summary(health$HvyAlcoholConsum)
summary(health$AnyHealthcare)
summary(health$NoDocbcCost)
summary(health$GenHlth)
summary(health$MentHlth)
summary(health$bin.MentHlth)
summary(health$PhysHlth)
summary(health$bin.PhysHlth)
summary(health$DiffWalk)
summary(health$Sex)
summary(health$Age)
```

```r
hist(as.numeric(health$Age))
summary(health$Education)
hist(as.numeric(health$Education))
summary(health$Income)
hist(as.numeric(health$Income))

## LASSO ##

health_train_x <- data.matrix(train_sample[ , 2:8])
health_train_y <- as.matrix(train_sample[ , 1])

health_lasso <- glmnet(health_train_x, health_train_y, family = "binomial",
                       alpha = 1)
health_cv_lasso <- cv.glmnet(health_train_x, health_train_y, family = "binomial",
                             alpha = 1)

## 1-SE ##
lambda_1se <- health_cv_lasso$lambda.1se
table(new_train$HeartDiseaseorAttack, predict(health_lasso, health_train_x,
                                              s = lambda_1se, type = "class"))
class.sum(new_train$HeartDiseaseorAttack, predict(health_lasso, health_train_x,
                                                  s = lambda_1se, type = "response"))
# PCC = 77.08
# Specificity = 79.61

coef(health_lasso, s = lambda_1se)
# Kept  = HighBP, HighChol, CholCheck, Smoker, Stroke, Diabetes, HvyAlcoholConsump,
# NoDocbcCost, GenHlth, DiffWalk, Sex, Age, Income, bin.PhysHlth

lasso_best_features <- new_train %>% dplyr::select(HeartDiseaseorAttack, HighBP,
                                                   HighChol, CholCheck, Smoker,
                                                   Stroke, Diabetes, HvyAlcoholConsump,
                                                   NoDocbcCost, GenHlth, DiffWalk,
                                                   Sex, Age, Income, bin.PhysHlth)

lasso_test_features <- new_test %>% dplyr::select(HeartDiseaseorAttack, HighBP,
                                                  HighChol, CholCheck, Smoker,
                                                  Stroke, Diabetes, HvyAlcoholConsump,
                                                  NoDocbcCost, GenHlth, DiffWalk,
                                                  Sex, Age, Income, bin.PhysHlth)

## Min ##
lambda_min <- health_cv_lasso$lambda.min
table(new_train$HeartDiseaseorAttack, predict(health_lasso, health_train_x,
                                              s = lambda_min, type = "class"))
# PCC = 77.06
# Specificity = 79.38

coef(health_lasso, s = lambda_min)
# Kept everything but AnyHealthcare and ln.BMI


## Test Set Upsample ##
```

```
new_test <- upSample(test[,2:22], as.factor(test[,1]), list = FALSE)
new_test$HeartDiseaseorAttack <- new_test$Class
new_test <- new_test[,-22]

health_test_x <- data.matrix(new_test[ , 2:8])
health_test_y <- as.matrix(new_test[ , 1])

table(new_test$HeartDiseaseorAttack, predict(health_lasso, health_test_x,
                                             s = lambda_min, type = "class"))


# PCC = 77.09
# Specificity = 79.34
#

table(new_test$HeartDiseaseorAttack, predict(health_lasso, health_test_x,
                                             s = lambda_1se, type = "class"))
class.sum(new_test$HeartDiseaseorAttack, predict(health_lasso, health_test_x,
                                              s = lambda_1se, type = "response"))

# PCC = 77.10
# Specificity = 79.53
# 0.5

# 1-SE model seems best, every so slightly better predictions, but more interpretable

# RF variables #

rf_train_x <- data.matrix(train_sample[ , 2:8])
rf_train_y <- as.matrix(train_sample[ , 1])

rf_lasso <- glmnet(rf_train_x, rf_train_y, family = "binomial",
                       alpha = 1)
rf_cv_lasso <- cv.glmnet(rf_train_x, rf_train_y, family = "binomial",
                             alpha = 1)

rf_lambda_1se <- rf_cv_lasso$lambda.1se

table(train_sample$HeartDiseaseorAttack, predict(rf_lasso, rf_train_x,
                                                 s = rf_lambda_1se, type = "class"))

# PCC = 73.91
# Specificity = 69.81
# Sensitivity = 22.10

# LASSO variables #

las_train_x <- data.matrix(train_sample.lasso[ , 2:15])
las_train_y <- as.matrix(train_sample.lasso[ , 1])

las_lasso <- glmnet(las_train_x, las_train_y, family = "binomial",
                       alpha = 1)
las_cv_lasso <- cv.glmnet(las_train_x, las_train_y, family = "binomial",
```

```
                              alpha = 1)

las_lambda_1se <- las_cv_lasso$lambda.1se

table(train_sample.lasso$HeartDiseaseorAttack, predict(las_lasso, las_train_x,
                                                       s = las_lambda_1se, type = "class"))

# PCC = 76.68
# Specificity = 74.78
# Sensitivity = 21.36


## LDA ##

health_lda <- lda(HeartDiseaseorAttack ~ . , CV = TRUE, data = new_train)

# LOO CV Confusion Matrix
table(new_train$HeartDiseaseorAttack, health_lda$class)
# PCC = 77
# Specificity = 79.94

# 10 fold CV
health_lda_xval <- rep(0, length = nrow(new_train))
x <- rep(1:10, length = nrow(new_train))
x <- sample(x)
for(i in 1:10){
  xtrain <- new_train[x != i, ]
  xtest <- new_train[x == i, ]
  glub <- lda(HeartDiseaseorAttack ~ . , data = xtrain)
  health_lda_xval[x == i] <- predict(glub, xtest)$class
}

table(new_train$HeartDiseaseorAttack, health_lda_xval)
# PCC = 76.99
# Specificity = 79.91
# 50


# Using RF variables
health_lda_rf <- lda(HeartDiseaseorAttack ~ . , CV = TRUE, data = train_sample)

# LOO CV Confusion Matrix
table(rf_best_features$HeartDiseaseorAttack, health_lda_rf$class)
# PCC = 75.54
# Specificity = 79.10
# Slightly lower than full feature set

# 10 fold CV
health_lda_xval <- rep(0, length = nrow(train_sample))
x <- rep(1:10, length = nrow(train_sample))
x <- sample(x)
for(i in 1:10){
  xtrain <- train_sample[x != i, ]
```

```
  xtest <- train_sample[x == i, ]
  glub <- lda(HeartDiseaseorAttack ~ . , data = xtrain)
  health_lda_xval[x == i] <- predict(glub, xtest)$class
}

table(train_sample$HeartDiseaseorAttack, health_lda_xval)
# PCC = 73.57
# Specificity = 48.60
# Sensitiviey = 48.36
# Slightly lower than full feature set


# Using LASSO variables
health_lda_lasso <- lda(HeartDiseaseorAttack ~ . , CV = TRUE, data = test_sample.lasso)

# LOO CV Confusion Matrix
table(lasso_best_features$HeartDiseaseorAttack, health_lda_lasso$class)
# PCC = 77.07
# Specificity = 79.99
# Slightly better than full feature set

# 10 fold CV
health_lda_xval <- rep(0, length = nrow(test_sample.lasso))
x <- rep(1:10, length = nrow(test_sample.lasso))
x <- sample(x)
for(i in 1:10){
  xtrain <- test_sample.lasso[x != i, ]
  xtest <- test_sample.lasso[x == i, ]
  glub <- lda(HeartDiseaseorAttack ~ . , data = xtrain)
  health_lda_xval[x == i] <- predict(glub, xtest)$class
}

table(test_sample.lasso$HeartDiseaseorAttack, health_lda_xval)
# PCC = 76.81
# Specificity = 74.63
# Sensitivity = 20.90
# Slightly better than full feature set


## QDA ##

health_qda <- qda(HeartDiseaseorAttack ~ . , CV = TRUE, data = new_train)

# LOO CV Confusion Matrix
table(new_train$HeartDiseaseorAttack, health_qda$class)
# PCC = 73.16
# Specificity = 88.23

# 10 fold CV
health_qda_xval_class <- rep(0, nrow(new_train))
health_qda_xval_posterior <- rep(0, nrow(new_train))
xvs <- rep(1:10, length = nrow(new_train))
xvs <- sample(xvs)
```

```r
for(i in 1:10){
  xtrain <- new_train[xvs != i, ]
  xtest <- new_train[xvs == i, ]
  glub <- qda(HeartDiseaseorAttack ~ . , data = xtrain)
  health_qda_xval_posterior[xvs == i] <- predict(glub, xtest)$posterior[, 2]
  health_qda_xval_class[xvs == i] <- predict(glub, xtest)$class
}

table(new_train$HeartDiseaseorAttack, health_qda_xval_class)
# PCC = 73.15
# Specificity = 88.25


# Using RF variables
health_qda_rf <- qda(HeartDiseaseorAttack ~ . , CV = TRUE, data = train_sample)

# LOO CV Confusion Matrix
table(train_sample$HeartDiseaseorAttack, health_qda_rf$class)
# PCC = 71.39
# Specificity = 89.98
# PCC slightly doum, specificity slightly up

# 10 fold CV
health_qda_xval_class <- rep(0, nrow(train_sample))
health_qda_xval_posterior <- rep(0, nrow(train_sample))
xvs <- rep(1:10, length = nrow(train_sample))
xvs <- sample(xvs)
for(i in 1:10){
  xtrain <- train_sample[xvs != i, ]
  xtest <- train_sample[xvs == i, ]
  glub <- qda(HeartDiseaseorAttack ~ . , data = xtrain)
  health_qda_xval_posterior[xvs == i] <- predict(glub, xtest)$posterior[, 2]
  health_qda_xval_class[xvs == i] <- predict(glub, xtest)$class
}

table(train_sample$HeartDiseaseorAttack, health_qda_xval_class)
# PCC = 71.40
# Specificity = 89.99
# PCC slightly doum, specificity slightly up


# Using LASSO variables
health_qda_lasso <- qda(HeartDiseaseorAttack ~ . , CV = TRUE, data = train_sample.lasso)

# LOO CV Confusion Matrix
table(train_sample.lasso$HeartDiseaseorAttack, health_qda_lasso$class)
# PCC = 72.15
# Specificity = 89.73
# PCC slightly down, specificity slightly up

# 10 fold CV
health_qda_xval_class <- rep(0, nrow(train_sample.lasso))
health_qda_xval_posterior <- rep(0, nrow(train_sample.lasso))
```

```r
xvs <- rep(1:10, length = nrow(train_sample.lasso))
xvs <- sample(xvs)
for(i in 1:10){
  xtrain <- train_sample.lasso[xvs != i, ]
  xtest <- train_sample.lasso[xvs == i, ]
  glub <- qda(HeartDiseaseorAttack ~ . , data = xtrain)
  health_qda_xval_posterior[xvs == i] <- predict(glub, xtest)$posterior[, 2]
  health_qda_xval_class[xvs == i] <- predict(glub, xtest)$class
}

table(train_sample.lasso$HeartDiseaseorAttack, health_qda_xval_class)
# PCC = 70.94
# Specificity = 52.63
# Sensitivity = 10.10
# PCC slightly down, specificity slightly up
```