Introduction

For this project we wanted to investigate the leading causes for heart disease and if it can be predicted based on risk factors in the models. America is one of the least healthy countries in the world with heart disease being the leading cause of death and 1 in 4 American adults dying from heart disease (Nina). Although these heart problems are staggering, there is still hope to counteract these statistics. 80 percent of all cardiovascular problems can be prevented, so through proper analysis we hope to determine the leading risk factors for heart disease so that the average American can be more knowledgeable on the topic and actively live a life that is healthy and heart friendly (Nina).

The major group of people that we look to assist are doctors and physicians. By better understanding risk factors for heart disease we can hope to better inform doctors of patients that could be at risk. If at risk patients can be identified sooner than preventative measures can be taken sooner and hopefully save the lives of American adults all across the country.

In addition the investigation of this problem is beneficial to health insurance companies across the country. With more knowledge on risk factors for heart disease, insurance companies will have better knowledge on who to charge higher rates or to not insure at all. As we make our way to adulthood ourselves, it is also important for people our age to know the facts on heart disease and how they can live a healthier lifestyle in order to avoid suffering from the detrimental effects of heart disease.

Problem Description

The question that we specifically investigated for our models is, “What factors are most likely to influence heart disease?” By better understanding the risk factors for heart disease we can better understand how to combat it and actively live lifestyles that counteract the development of heart problems.

Approach (Classification)

For this problem we decided that the best approach was classification as we are trying to identify if someone will have heart disease or not based on the risk factors in our models. The three models that we conducted for the project were, KNN classification, logistic regression, and a classification tree. The outcome variable or predictor was the variable “HasHeartDisease” and the 23 input variables are as follows; BMI, IsSmoker, IsDrinker, HadStroke, PhysicalHealth, MentalHealth, HasDiffWalking, IsMale, Is40-44, is45-49, is50-54, is55-59, is60-64, is65-69, is70-74, is75-79, is80+, isDiabetic, isPhyscialyActive, SleepTime, hasAsthma, hasKidneyDisease, hasSkinCancer.

The data set originally had 96 columns and over 319,000 rows. In order to better fit our models we narrowed the columns down to the 14 that we found to be the most statistically significant. From there we created dummy variables for all of the variables that had different units in order to standardise the data. This included separating the age variable into 8

different binary dummy variables where 1 indicates that they fall into that age bracket and 0 indicates that they don't. For all other variables listed above that include “is” or “had/has” they were also converted to categorical binary variables that indicate 1 for having or is the specific variable and 0 for not having or being the variable.

This leaves us with BMI, PhysicalHealth, MentalHealth and SleepTime as the only numerical variables in the model. BMI is represented simply by a number indicating what an individual's BMI is. Physical health and mental health are represented by survey respondents indicating out of 30 days in a month how many days they struggle with physical or mental health. Sleep time is simply represented by the average hours of sleep per night that respondents get.

Initial Problems:

We initially encountered significant challenges in model predictions due to the extreme imbalance in our dataset. Over 90% of the responses indicated no presence of heart disease, making it difficult for our models to accurately identify individuals with heart disease. This imbalance led to a strong bias toward predicting the majority class, reducing the overall effectiveness of our models.

To address this issue, we applied the Synthetic Minority Oversampling Technique (SMOTE). Through external research, we identified SMOTE as an effective method for handling class imbalances by artificially generating examples for the minority class. Instead of simply copying existing instances, SMOTE creates new synthetic samples by adding between existing minority class observations.

SMOTE works by selecting a random instance from the minority class and identifying its k-nearest neighbors. A synthetic data point is then generated by selecting a random point along the line segment between the chosen instance and one of its neighbors. This insertion process introduces more variation into the minority class rather than simply oversampling existing cases, making the dataset more balanced and improving the model’s ability to generalize. By generating synthetic data instead of copying minority samples, SMOTE reduces overfitting and helps the model learn better patterns associated with heart disease.

By applying SMOTE to our dataset, we created additional synthetic records of heart disease cases, making our model more capable of identifying true positive cases. This approach significantly improved the model’s ability to distinguish between individuals with and without heart disease, leading to better generalization and a more robust classification performance.

Data Analysis

KNN Classification

The first model that we ran was the KNN-Classification model. This model was found to be the least accurate of the three models that we ran and provided little insight into the problem that we were trying to address. For this model we used the minimum required 30,000 rows of data rather than the full 319,000 that were included in the data set so that the model would be able to run more smoothly. Initially we attempted to use the full 319,000 rows but as KNN is a model that utilises nearest neighbors to group together data and make a prediction, it became increasingly difficult for the RStudio software to run the model on so much data. By tapering our data down we were able to get things to run in an effective manner.

The first step in this model was determining the correct level of K to use. I began by running the accuracy for the first 60 levels of K, we found that K=1 provided the highest accuracy but this level of K runs the risk of overfitting the data. The level of accuracy for K=1 was 89.08% and the accuracy for the following levels dropped as K increased. In the future, to run a better KNN model we would need a data set that is less unbalanced and contains a smaller amount of rows so that we could use the entire set without having to discard rows.

The outputs for the KNN model are as follows, the sensitivity was 91.56%, the specificity was 22.8% and the kappa was 0.1366. These metrics indicate that this model was effective at predicting whether or not an individual would not have heart disease, but was not effective at predicting if someone would have heart disease. Our kappa was very low and indicates a slight improvement over a random guess, not an ideal output for this model. We would have hoped to see a kappa that was at least over 0.5 and a specificity closer to 50%. For future models we would hope to improve these metrics.

Logistic Regression

The next model that we ran was the logistic regression model. For this model we utilised the full breadth of the data set and included all 319,000 rows. This model was able to provide some very valuable insights and was the top performing model out of the three that we ran for analysis.

For this model we used a seed of 123 and split our training and validation data into a 70/30 split. This model proved to have a prediction accuracy of 88.93%, performing only slightly better than the KNN model. The sensitivity for this model was 94.01%, coming in slightly higher than the KNN model again, meaning that this model was better at predicting if someone would not have heart disease. The specificity for this model came in at 34.72%, showing a marginal increase over the KNN model once again, indicating that this model is much better at predicting if an individual will have heart disease. The kappa for this model came in at 0.2772, indicating that this model is slightly better than a random guess.

In future models we would hope to see a kappa that is closer to 0.5 and a specificity at or around 50%. Although these metrics were lower then we had hoped, the logistic regression was able to provide some very valuable insights through the regression output and the odds ratios.

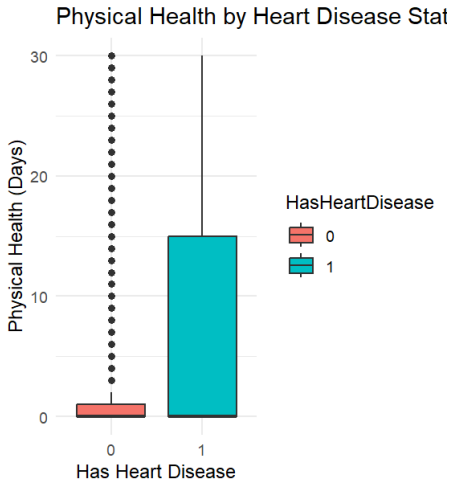
	Estimate	Std. Error	z value	Pr(>|z|)	
(Intercept)	-8.3856077	0.0767853	-109.208	<2e-16	***
BMI	0.0190972	0.0009149	20.873	<2e-16	***
IsSmoker	0.5037042	0.0113631	44.328	<2e-16	***
IsDrinker	-0.5174745	0.0281880	-18.358	<2e-16	***
HadStroke	1.1895999	0.0216771	54.878	<2e-16	***
PhysicalHealth	0.0245292	0.0006325	38.780	<2e-16	***
MentalHealth	0.0137462	0.0007135	19.265	<2e-16	***
HasDiffWalking	0.3742675	0.0151294	24.738	<2e-16	***
IsMale	0.8064995	0.0116715	69.100	<2e-16	***
`is40-44`	0.4377200	0.0423983	10.324	<2e-16	***
`is45-49`	0.7379102	0.0383406	19.246	<2e-16	***
`is50-54`	1.2028983	0.0331193	36.320	<2e-16	***
`is55-59`	1.5265768	0.0304266	50.172	<2e-16	***
`is60-64`	1.8776636	0.0287851	65.230	<2e-16	***
`is65-69`	2.1258257	0.0283672	74.940	<2e-16	***
`is70-74`	2.4678639	0.0283513	87.046	<2e-16	***
`is75-79`	2.7146848	0.0296291	91.622	<2e-16	***
`is80+`	3.0836451	0.0293236	105.159	<2e-16	***
isDiabetic	0.5855067	0.0139360	42.014	<2e-16	***
isPhysicalyActive	-0.0306866	0.0131915	-2.326	0.020	*
SleepTime	-0.0389757	0.0035921	-10.850	<2e-16	***
hasAsthma	0.2774095	0.0162646	17.056	<2e-16	***
hasKidneyDisease	0.6239585	0.0227273	27.454	<2e-16	***
hasSkinCancer	0.0231096	0.0163639	1.412	0.158	

As seen from the regression output, almost every variable besides “isPhysicalyActive” and “hasSkinCancer” indicate statistical significance for correlation to the outcome variable of “HasHeartDisease”. Meaning that all of these variables in someway or another indicate that they would be potential risk factors for an individual to have heart disease.

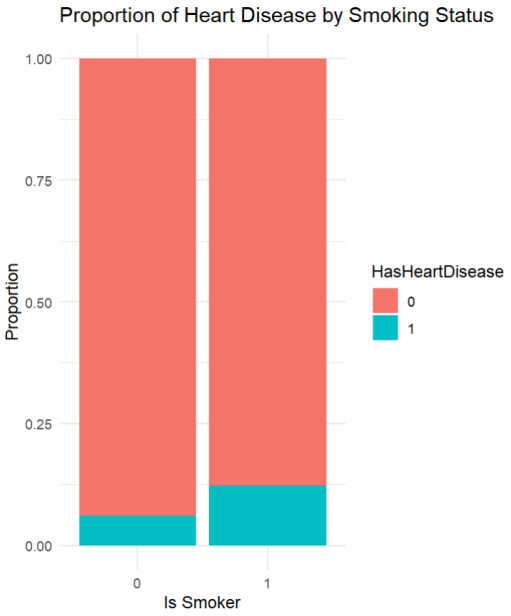
Variable	Odds Ratio
BMI	1.019
IsSmoker	1.654
IsDrinker	0.596
HadStroke	3.285
PhysicalHealth	1.022
MentalHealth	1.01
HasDiffWalking	1.481
IsMale	2.241
Is40-44	1.549
is45-49	2.092
is50-54	3.331
is55-59	4.602
Is60-64	6.538
is65-69	8.379
is70-74	11.797
is75-79	15.099
Is80+	21.837
IsDiabetic	1.796
IsPhyicalyActive	0.969
SleepTime	0.962
HasAshtma	1.444
HasKindeyDisease	1.996
HasSkinCancer	1.023

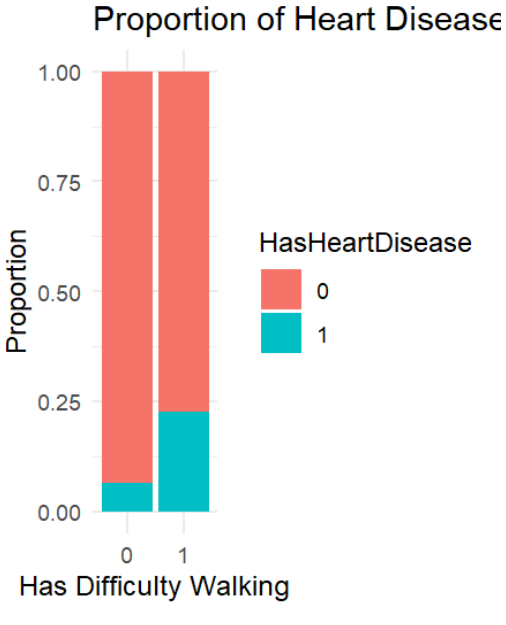
Pictured above is a table that indicates the odds ratio for each of the input variables in the regression model. The odds ratio is a statistical measure that indicates the strength of association between two events. Seen from the highlighted rows, “HadStroke”, “IsMale”, and “is80+” all show an odds ratio that is higher than the rest of the input variables.

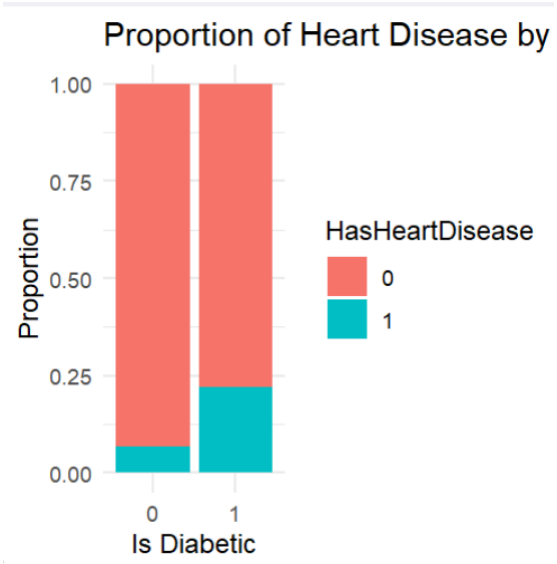
It is clear to see that age has the most striking impact on indicating if someone will have heart disease or not, followed by having a stroke and being a man. Below are some graphs that help better visualise the impact of some of the variables that may have not scored high on the odds ratio but are still statistically significant.



Before any models were prepared there were some variables that stuck out to us as variables that we assumed would play a big impact on if someone would have heart disease or not. Variables that indicate if someone is a smoker, is diabetic or has difficulty walking where all variables that we thought would be high indicators for heart disease. Pictured below are three bar graphs created in R that help to visualise the connection between heart disease and these variables.







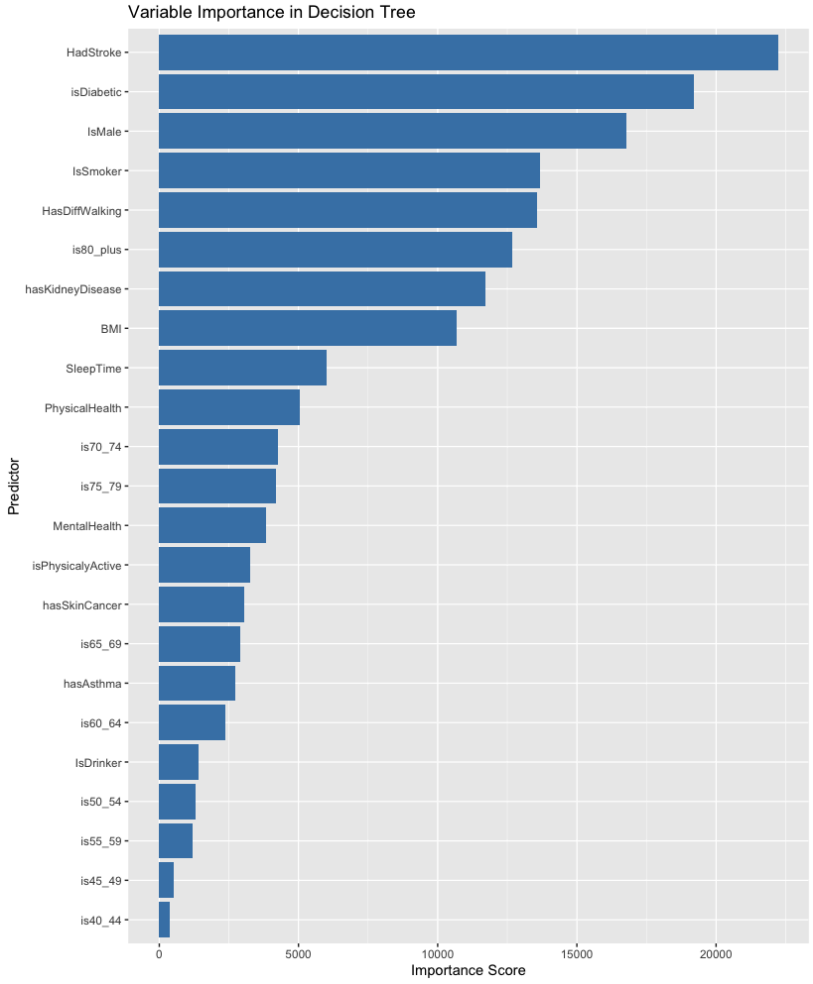
For all three of these variables it is very clear to see that those who are diabetic, have difficulty walking and are smokers all indicate more instances of individuals who have heart disease. Although those who are diabetic or have difficulty walking may not be able to change those aspects of their lives, individuals who are smokers can quit at any given time, indicating an actionable, measurable way to decrease heart disease by eliminating this risk factor.

Classification Tree

For our third and final model, we used a classification tree model. We used the entirety of the dataset, and we split it into a 70% training and 30% validation set, using a random seed of 123 to ensure reproducibility. Additionally, SMOTE was applied to balance the dataset and address class imbalances.

This model achieved an accuracy of 88.89%, which suggests strong predictive performance. However, the model displayed high sensitivity (95.50%) and low specificity (18.48%), meaning it was very good at predicting individuals without heart disease, but struggled in correctly identifying those who do have heart disease. The kappa statistic of 0.1648 indicates that while better than random guessing, the agreement between predictions and true labels remains low.

The variable importance plot below revealed key insights from the model:



The most important factors contributing to heart disease are as follows: HadStroke, IsDiabetic, IsMale, IsSmoker, Has Difficulty Walking, Age groups (especially 80+), BMI, and Sleep Time.These features significantly impacted the decision-making process of the model, highlighting the strong correlation between cardiovascular disease and lifestyle-related factors.

While the decision tree demonstrated solid accuracy, its low specificity resulted in a high number of false positives, which could be problematic in medical applications. Future improvements could involve fine-tuning the model through feature selection or exploring alternative models such as Random Forest to enhance both sensitivity and specificity.

Results

The findings from our three classification models presented us with some very interesting results.

Model	Prediction	Sensitivity	Specificity	Kappa
KNN	89.08%	92%	22.80%	0.1366
Logistic Regr	88.93%	94%	35%	0.2772
Classificatio	88.89%	95.50%	18.48%	0.1648

Inference

Among the models tested, logistic regression demonstrated the most balanced performance, achieving the highest specificity and kappa score. This suggests that logistic regression was the most effective at distinguishing individuals who are likely to develop heart disease, making it the best choice for this particular analysis. Its ability to correctly classify both those with and without heart disease provides a stronger foundation for predictive accuracy in real-world applications. Given its superior specificity, this model is particularly valuable in healthcare settings where reducing false positives is essential for preventing unnecessary medical interventions.

Conclusion

Our analysis highlights that males over the age of 55 who have previously suffered a stroke are at the greatest risk of developing heart disease. The combination of age, gender, and prior cardiovascular events significantly increases susceptibility, making these individuals a high-priority group for medical monitoring and intervention.

Recommendations

Given these insights, healthcare professionals should prioritize preventative healthcare for males over the age of 55 who have experienced a stroke. This could involve proactive measures such as regular cardiovascular screenings, personalized treatment plans, and lifestyle interventions focused on promoting heart health. Encouraging patients to adopt healthier eating habits, engage in regular physical activity, and manage other risk factors such as smoking and diabetes could significantly reduce the likelihood of developing heart disease.

From an insurance perspective, providers may need to consider adjusting premium rates for individuals within this high-risk category. Since males over 55 with a history of stroke are more prone to costly health complications, including hospitalizations and long-term care needs, insurers may implement higher premiums to account for the increased financial risk. Additionally, incentivizing policyholders to participate in wellness programs or maintain

healthy behaviors could help mitigate long-term healthcare expenses and improve overall patient outcomes.

Logistic Regression (Model by Sam Shafkowitz)

Code:

```
rm(list = ls())

install.packages("smotefamily")

library("tidyverse")
library("caret")
library("readxl")
library("smotefamily")
library("readxl")

data <- read_csv("455ProjectDataSet.csv")

data$HasHeartDisease <- as.factor(data$HasHeartDisease)

categorical_vars <- c("IsSmoker", "IsDrinker", "HadStroke", "HasDiffWalking",
                      "IsMale", "isDiabetic", "isPhysicalyActive",
                      "hasAsthma", "hasKidneyDisease", "hasSkinCancer")

data[categorical_vars] <- lapply(data[categorical_vars], function(x) as.numeric(as.factor(x)))

set.seed(123)
sample_size <- floor(0.7 * nrow(data))
train_indices <- sample(seq_len(nrow(data)), size = sample_size)
train_data <- data[train_indices, ]
test_data <- data[-train_indices, ]

smote_data <- SMOTE(X = train_data %>% select(-HasHeartDisease),
                    target = train_data$HasHeartDisease, K = 5, dup_size = 2)

train_data <- smote_data$data
train_data$HasHeartDisease <- as.factor(train_data$class)
train_data$class <- NULL
```

```
smote_data <- SMOTE(X = train_data %>% select(-HasHeartDisease),
                    target = train_data$HasHeartDisease, K = 5, dup_size = 2)

train_data <- smote_data$data
train_data$HasHeartDisease <- as.factor(train_data$class)
train_data$class <- NULL

logistic_model <- glm(HasHeartDisease ~ ., data = train_data, family = binomial)

logistic_predictions <- predict(logistic_model, test_data, type = "response")

logistic_predictions <- factor(ifelse(logistic_predictions >= 0.5, 1, 0), levels = c(0, 1))

confusionMatrix(logistic_predictions, test_data$HasHeartDisease)

summary(logistic_model)

odds_ratios = exp(coef(logistic_model))
print(odds_ratios)
# Print odds ratios without scientific notation
print(format(odds_ratios, scientific = FALSE))
```

Output:

```
Warning message:
package 'caret' was built under R version 4.4.3
> library("readxl")
Warning message:
package 'readxl' was built under R version 4.4.3
> library("smotefamily")
Warning message:
package 'smotefamily' was built under R version 4.4.3
> library("readxl")
>
> data <- read_excel("455ProjectDataSet.csv")
Error:  `path` does not exist: '455ProjectDataSet.csv'
> data <- read_csv("455ProjectDataSet.csv")
Error: '455ProjectDataSet.csv' does not exist in current working directory ('C:/Users/sshaf/OneDrive/Desk
top/oba_455_555_ddpm_r/rproject').
> setwd("C:/Users/sshaf/OneDrive/Desktop/oba_455_555_ddpm_r/rproject/n.cluster")
> data <- read_csv("455ProjectDataSet.csv")
Rows: 319795 Columns: 24
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
dbl (24): HasHeartDisease, BMI, IsSmoker, IsDrinker, HadStroke, PhysicalHealth, MentalHealth, Has...

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
> data$HasHeartDisease <- as.factor(data$HasHeartDisease)
>
> categorical_vars <- c("IsSmoker", "IsDrinker", "HadStroke", "HasDiffWalking",
+                       "IsMale", "isDiabetic", "isPhysicalyActive",
+                       "hasAsthma", "hasKidneyDisease", "hasSkinCancer")
>
> data[categorical_vars] <- lapply(data[categorical_vars], function(x) as.numeric(as.factor(x)))
>
```

```
ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
> data$HasHeartDisease <- as.factor(data$HasHeartDisease)
>
> categorical_vars <- c("IsSmoker", "IsDrinker", "HadStroke", "HasDiffWalking",
+                       "IsMale", "isDiabetic", "isPhysicalyActive",
+                       "hasAsthma", "hasKidneyDisease", "hasSkinCancer")
>
> data[categorical_vars] <- lapply(data[categorical_vars], function(x) as.numeric(as.factor(x)))
>
> set.seed(123)
> sample_size <- floor(0.7 * nrow(data))
> train_indices <- sample(seq_len(nrow(data)), size = sample_size)
> train_data <- data[train_indices, ]
> test_data <- data[-train_indices, ]
>
> smote_data <- SMOTE(X = train_data %>% select(-HasHeartDisease),
+                     target = train_data$HasHeartDisease, K = 5, dup_size = 2)
>
> train_data <- smote_data$data
> train_data$HasHeartDisease <- as.factor(train_data$class)
> train_data$class <- NULL
>
> logistic_model <- glm(HasHeartDisease ~ ., data = train_data, family = binomial)
>
> logistic_predictions <- predict(logistic_model, test_data, type = "response")
>
> logistic_predictions <- factor(ifelse(logistic_predictions >= 0.5, 1, 0), levels = c(0, 1))
>
> confusionMatrix(logistic_predictions, test_data$HasHeartDisease)
Confusion Matrix and Statistics
```

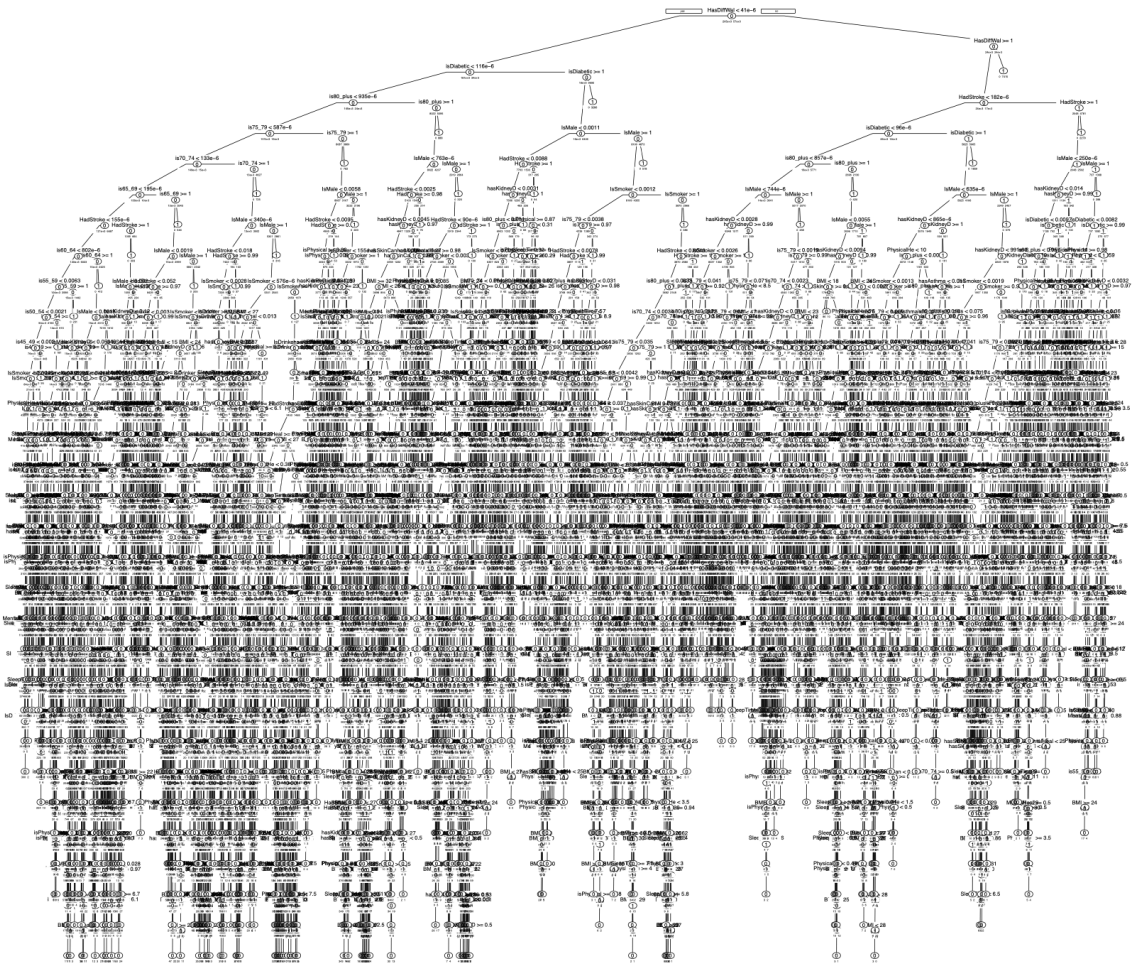
```
> confusionMatrix(logistic_predictions, test_data$HasH
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 82459  5369
         1  5256  2855

               Accuracy : 0.8893
                 95% CI : (0.8872, 0.8912)
    No Information Rate : 0.9143
    P-Value [Acc > NIR] : 1.0000

                  Kappa : 0.289

 Mcnemar's Test P-Value : 0.2772

            Sensitivity : 0.9401
            Specificity : 0.3472
         Pos Pred Value : 0.9389
         Neg Pred Value : 0.3520
             Prevalence : 0.9143
         Detection Rate : 0.8595
   Detection Prevalence : 0.9155
      Balanced Accuracy : 0.6436

       'Positive' Class : 0

> summary(logistic_model)


Coefficients:
                  Estimate Std. Error  z value Pr(>|z|)
(Intercept)      -8.3856077  0.0767853 -109.208   <2e-16 ***
BMI               0.0190972  0.0009149   20.873   <2e-16 ***
IsSmoker          0.5037042  0.0113631   44.328   <2e-16 ***
IsDrinker        -0.5174745  0.0281880  -18.358   <2e-16 ***
HadStroke         1.1895999  0.0216771   54.878   <2e-16 ***
PhysicalHealth    0.0245292  0.0006325   38.780   <2e-16 ***
MentalHealth      0.0137462  0.0007135   19.265   <2e-16 ***
HasDiffWalking    0.3742675  0.0151294   24.738   <2e-16 ***
IsMale            0.8064995  0.0116715   69.100   <2e-16 ***
`is40-44`         0.4377200  0.0423983   10.324   <2e-16 ***
`is45-49`         0.7379102  0.0383406   19.246   <2e-16 ***
`is50-54`         1.2028983  0.0331193   36.320   <2e-16 ***
`is55-59`         1.5265768  0.0304266   50.172   <2e-16 ***
`is60-64`         1.8776636  0.0287851   65.230   <2e-16 ***
`is65-69`         2.1258257  0.0283672   74.940   <2e-16 ***
`is70-74`         2.4678639  0.0283513   87.046   <2e-16 ***
`is75-79`         2.7146848  0.0296291   91.622   <2e-16 ***
`is80+`           3.0836451  0.0293236  105.159   <2e-16 ***
isDiabetic        0.5855067  0.0139360   42.014   <2e-16 ***
isPhysicalyActive -0.0306866  0.0131915   -2.326    0.020 *
SleepTime         -0.0389757  0.0035921  -10.850   <2e-16 ***
hasAsthma         0.2774095  0.0162646   17.056   <2e-16 ***
hasKidneyDisease  0.6239585  0.0227273   27.454   <2e-16 ***
hasSkinCancer     0.0231096  0.0163639    1.412    0.158
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)


Number of Fisher Scoring iterations: 6

>
> odds_ratios = exp(coef(logistic_model))
> print(odds_ratios)
      (Intercept)               BMI          IsSmoker         IsDrinker         HadStroke
     2.281271e-04      1.019281e+00      1.654840e+00      5.960239e-01      3.285766e+00
   PhysicalHealth      MentalHealth    HasDiffWalking            IsMale         `is40-44`
     1.024832e+00      1.013841e+00      1.453926e+00      2.240053e+00      1.549171e+00
         `is45-49`         `is50-54`         `is55-59`         `is60-64`         `is65-69`
     2.091560e+00      3.329753e+00      4.602395e+00      6.538211e+00      8.379814e+00
         `is70-74`         `is75-79`           `is80+`       isDiabetic isPhysicalyActive
     1.179722e+01      1.509985e+01      2.183786e+01      1.795901e+00      9.697794e-01
        SleepTime         hasAsthma  hasKidneyDisease     hasSkinCancer
     9.617740e-01      1.319707e+00      1.866301e+00      1.023379e+00
> # Print odds ratios without scientific notation
> print(format(odds_ratios, scientific = FALSE))
      (Intercept)               BMI          IsSmoker         IsDrinker         HadStroke
  " 0.0002281271"   " 1.0192807621"   " 1.6548398712"   " 0.5960238837"   " 3.2857661700"
   PhysicalHealth      MentalHealth    HasDiffWalking            IsMale         `is40-44`
  " 1.0248324663"   " 1.0138411554"   " 1.4539260045"   " 2.2400528404"   " 1.5491710884"
         `is45-49`         `is50-54`         `is55-59`         `is60-64`         `is65-69`
  " 2.0915600844"   " 3.3297534853"   " 4.6023949597"   " 6.5382110783"   " 8.3798142573"
         `is70-74`         `is75-79`           `is80+`       isDiabetic isPhysicalyActive
  "11.7972204549"  "15.0998495658"   "21.8378596202"   " 1.7959006779"   " 0.9697794177"
        SleepTime         hasAsthma  hasKidneyDisease     hasSkinCancer
  " 0.9617740447"   " 1.3197066725"   " 1.8663012821"   " 1.0233786738"
>
```

Classification Tree (Model by Luke Snellback)

Code:

```r
library("tidyverse")
library("rpart")
library("rpart.plot")
library("caret")
library("readxl")
library("smotefamily")
library("ggplot2")

df = read_excel("455ProjectDataSet.xlsx")

df = df %>%
  rename(is40_44 = `is40-44`, is45_49 = `is45-49`, is50_54 = `is50-54`,
         is55_59 = `is55-59`, is60_64 = `is60-64`, is65_69 = `is65-69`,
         is70_74 = `is70-74`, is75_79 = `is75-79`, is80_plus = `is80+`)


df$HasHeartDisease <- as.factor(df$HasHeartDisease)

set.seed(123)

train_index = sample(seq_len(nrow(df)), size = 0.7 * nrow(df))
train = df[train_index, ]
validation = df[-train_index, ]

smote_data = SMOTE(X = train %>% select(-HasHeartDisease),
                   target = train$HasHeartDisease,
                   K = 5, dup_size = 2)
train = smote_data$data
train$HasHeartDisease = as.factor(train$class)
train$class = NULL

heart_tree = rpart(HasHeartDisease ~ ., data = train, method = "class")

prp(heart_tree, type = 1, extra = 1, under = TRUE, split.font = 1, varlen = -10)

heart_tree_pruned = rpart(HasHeartDisease ~ ., data = train, method = "class",
                          cp = 0.00001, minsplit = 10, xval = 10)

plotcp(heart_tree_pruned)

cp_table = as_tibble(heart_tree_pruned$cptable)
optimal_cp = cp_table %>% filter(nsplit == 10)

pruned_tree = prune(heart_tree_pruned, cp = optimal_cp$CP)

prp(pruned_tree, type = 1, extra = 1, under = TRUE, split.font = 1, varlen = -10)

pruned_predictions = predict(pruned_tree, validation, type = "class")

validation = validation %>% mutate(Predicted = pruned_predictions)
confusionMatrix(validation$Predicted, validation$HasHeartDisease, positive = "0")


var_importance = varImp(heart_tree_pruned, scale = FALSE)
print(var_importance)


ggplot(var_importance, aes(x = reorder(rownames(var_importance), Overall), y = Overall)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +
  labs(title = "Variable Importance in Decision Tree", x = "Predictor", y = "Importance Score")
```

Output:



```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 83764  6704
         1  3951  1520

               Accuracy : 0.8889
                 95% CI : (0.8869, 0.8909)
    No Information Rate : 0.9143
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1648

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.9550
            Specificity : 0.1848
         Pos Pred Value : 0.9259
         Neg Pred Value : 0.2778
             Prevalence : 0.9143
         Detection Rate : 0.8731
   Detection Prevalence : 0.9430
      Balanced Accuracy : 0.5699

       'Positive' Class : 0
```

KNN Classification Code: (Model by Luke Snellback):

```r
rm(list = ls())

library(caret)
library(tidyverse)
library(smotefamily)
library(readxl)

data = read_csv("455ProjectDataSet.csv")

data = data[1:30000, ]

data$HasHeartDisease = as.factor(data$HasHeartDisease)

preProcValues = preProcess(data[, -1], method = c("center", "scale"))
dataScaled = predict(preProcValues, data)

set.seed(123)
trainIndex = createDataPartition(dataScaled$HasHeartDisease, p = 0.7, list = FALSE)
trainData = dataScaled[trainIndex, ]
testData = dataScaled[-trainIndex, ]

smote_data = SMOTE(X = trainData %>% select(-HasHeartDisease),
                   target = trainData$HasHeartDisease,
                   K = 5, dup_size = 2)

trainData = smote_data$data
trainData$HasHeartDisease = as.factor(trainData$class)
trainData$class = NULL

control = trainControl(method = "repeatedcv", number = 10, repeats = 3)
```

```r
customGrid = expand.grid(k = 1:10)

knnModel = train(HasHeartDisease ~ ., data = trainData, method = "knn",
                 trControl = control, tuneGrid = customGrid)

print(knnModel$results[, c("k", "Accuracy")])

predictions = predict(knnModel, newdata = testData)

confMatrix = confusionMatrix(predictions, testData$HasHeartDisease)

print(confMatrix)
```

Output:

```r
> data = data[1:30000, ]
>
> data$HasHeartDisease = as.factor(data$HasHeartDisease)
>
> preProcValues = preProcess(data[, -1], method = c("center", "scale"))
> dataScaled = predict(preProcValues, data)
>
> set.seed(123)
> trainIndex = createDataPartition(dataScaled$HasHeartDisease, p = 0.7, list = FALSE)
> trainData = dataScaled[trainIndex, ]
> testData = dataScaled[-trainIndex, ]
>
> smote_data = SMOTE(X = trainData %>% select(-HasHeartDisease),
+                    target = trainData$HasHeartDisease,
+                    K = 5, dup_size = 2)
>
> trainData = smote_data$data
> trainData$HasHeartDisease = as.factor(trainData$class)
> trainData$class = NULL
>
> control = trainControl(method = "repeatedcv", number = 10, repeats = 3)
>
> customGrid = expand.grid(k = 1:10)
>
> knnModel = train(HasHeartDisease ~ ., data = trainData, method = "knn",
+                  trControl = control, tuneGrid = customGrid)
>
>
> print(knnModel$results[, c("k", "Accuracy")])
    k  Accuracy
1   1 0.8900788
2   2 0.8590666
3   3 0.8582552
4   4 0.8440062
5   5 0.8402756
6   6 0.8323535
7   7 0.8342595
8   8 0.8293925
9   9 0.8293522
10 10 0.8259860
>
> predictions = predict(knnModel, newdata = testData)
>
> confMatrix = confusionMatrix(predictions, testData$HasHeartDisease)
>
```

```
> print(confmat.rf)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 7529  565
         1  687  218

               Accuracy : 0.8609
                 95% CI : (0.8536, 0.868)
    No Information Rate : 0.913
    P-Value [Acc > NIR] : 1.000000

                  Kappa : 0.182

 Mcnemar's Test P-Value : 0.000627

            Sensitivity : 0.9164
            Specificity : 0.2784
         Pos Pred Value : 0.9302
         Neg Pred Value : 0.2409
             Prevalence : 0.9130
         Detection Rate : 0.8366
   Detection Prevalence : 0.8994
      Balanced Accuracy : 0.5974

       'Positive' Class : 0
```

Works Cited

Julia, Nina. “Heart Disease Statistics in the US (2024 Update).” *CFAH*, 11 Jan. 2024, cfah.org/heart-disease-statistics/.

Pytlak, Kamil. “Indicators of Heart Disease (2022 Update).” *Kaggle*, 12 Oct. 2023, www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease/data.

“How to Use Smote for Imbalanced Data in R.” *GeeksforGeeks*, GeeksforGeeks, 26 July 2024, www.geeksforgeeks.org/how-to-use-smote-for-imbalanced-data-in-r/.

Otten, Neri Van. “Smote Oversampling & Tutorial on How to Implement in Python and R.” *Spot Intelligence*, 31 Oct. 2023, spotintelligence.com/2023/02/17/smote-oversampling-python-r/#:~:text=The%20algorithm%20explained,works%20on%20a%20test%20set.

Link to excel data: 455ProjectDataSet.xlsx