Designing Ideal Classification Models of Malignant and Benign Breast Tumor Samples

Andrew Gillock, Jordan Toler

05/08/2022

Introduction

Breast cancer is a disease in which breast cells adopt abnormal cellular activities and develop into a cancerous tumor. Initially, these growths are "in situ" or remain in the cell line they originated in, but often spread to neighboring tissue and transition to a metastasis state (DeSantis et al., 2019). Today, breast cancer is the second leading cause of cancer death among women in the US, only surpassed by lung cancer. Globally, statistics from 2020 show that 2.3 million women were diagnosed with breast cancer, and of that group, 685,000 cases resulted in death (World Health Organization, 2021; DeSantis et al., 2019). The gravity of this issue has led to an increased emphasis on the development of diagnostic techniques that maximize the improvement in patient prognosis by identifying the stage of the disease and detecting it as early as possible (Bhushan et al., 2017).

Several "imaging modalities" are utilized in detecting, diagnosing, and clinical management of breast cancer and, broadly, cancer in general (Bhushan et al., 2017). Standard imaging methods such as ultrasound provide texture and morphological features interpreted by imaging specialists who classify a tumor as benign or malignant (Wei et al., 2020). However, characteristics of these imaging outputs, such as "noise and resolution," introduce ambiguity and convolute their interpretability (Wei et al., 2020). In addition, factors such as skill and experience differences amongst specialists can result in differing interpretations of tumor images and potentially incorrect tumor classifications regarding its benign or malignant status (Wei et al., 2020). Therefore, researchers have decided to develop computational methods to create "novel imaging techniques with sensitivity and accuracy" for tumor classification and "assist current imaging professionals with their interpretations" (Aruleba et al., 2022).

These approaches broadly fall under an area of machine learning that applies either a supervised or unsupervised learning technique to categorize data, known as classification. Supervised learning is a technique where labeled data is provided to the model and classifies the data based on the provided labels. In contrast, unsupervised learning provides unlabeled data to a model and allows it to detect and find patterns in the provided information (Seldon, 2021). Examples of classification models as it relates to classifying breast tumor samples have not only been developed but also have high accuracy. For example, Wei et al. (2020) utilized a combinatory approach of parametric and non-parametric classifiers to identify breast cancer tumors using texture and morphological features from ultrasounds with an accuracy of 91.11 percent. A study by Mohammad et al. (2022) explored the classification accuracies of the five most-used classification techniques: multi-layer perceptron (MLP), k-nearest neighbors (KNN), J48, Support Vector Machines (SVM), and decision-tree-based methods. They applied this comparative strategy to a well-known breast cancer dataset, the Wisconsin dataset, to determine which approaches performed the best regarding classification accuracy. The most formidable method was the SVM, with a classification accuracy of approximately 91 percent. Aruleba et al. (2020) explores this growing area of research in an extensive review that chronicles the evolution of computational applications in breast cancer diagnostics to research and illustrate its diversity of models and utility.

Our project analyzed the Breast Cancer Wisconsin Diagnostic Dataset, a multivariate dataset with 569 breast tumor instances and ten real-valued features. (Dua & Graff, 2019) From this dataset, we intend to understand which machine learning algorithm will be ideal for prediction and which variables play a focal role in modeling the prediction. We first observed which tumor attributes were the most critical in obtaining high classification accuracies by best subset selection. We then approached the process of identifying the most accurate classification model by using k-fold cross-validation and comparing several linear, non-linear, tree-based models and SVM based on their computed classification accuracies. We found that the most successful model in accurately classifying the breast tumor samples is the non-linear tree-based bagging method, and the most significant predictors used to achieve this were fourteen of the original thirty-two.

Research Questions

- Can we accurately predict if a tumor sample is malignant (1) or benign (0) using its characteristics?
- Which predictor variables most accurately predict the diagnosis of tumor samples?

Methods

Subset Selection (Best Subset Selection):

Subset selection is an approach that reduces the complexity of a model by removing variables that are less associated with the response variable compared to other variables with a higher association. (James et al., 2021, p. 227) As a result, the reduction in complexity increases the models' interpretability. For these reasons, in addition to the number of predictors in our dataset, we applied best subset selection. This method returns models combining different sets of predictors and their associated adjusted R squared statistic (Adjusted Rˆ2), Bayesian Information Criterion (BIC), and Mallow's Cp (Cp). Cp is an unbiased estimate of a model's test MSE or error rate and therefore takes on smaller values when the model's error is lower (James et al., 2021, p. 233). BIC takes on a smaller value for models with lower test MSE and penalizes larger models, favoring smaller models. The opposite is true for Cp (James et al., 2021, p. 234). Finally, the adjusted Rˆ2 is a way of measuring how well a set of predictors models the variation in a response variable and is penalized by having noise terms that do not aid in explaining the response variable's variation (James et al., 2021, p. 235) .The ideal model should maximize the adjusted Rˆ2 and minimize the BIC and Cp. We optimized all three of these measurements using 14 predictor variables, which were then used to generate a new simple probability model.

Model Selection

The response variable, tumor type, is binary, which guides our model selection to select those that can predict a qualitative response variable: Logistic regressions, Tree-based methods, generalized additive models (GAMs), and support vector machines (SVM). We excluded models that cannot accommodate qualitative response variables. A dichotomous response variable means that we must assess our model fit using classification accuracy, as opposed to calculating test error.

Logistic Regression (Linear & Non-linear): In a logistic regression, the relationship between the response and predictor variables is modeled using the logistic function and is interpreted as probabilities rather than discrete values. The model's fit uses a method called maximum likelihood and results in outputs between zero and one for all values that a predictor can take on. This makes for easy and sensible interpretation because the log odds (logit) of a regression model is linear in X, so you need only transform it to interpret both B0 and B1 (James et al., 2021, p. 133-135). However, suppose the relationship between the response and predictors does not appear to be linearly related. In that case, applying a non-linear regression method that models the data well is necessary, but not so much that it overfits it. Our application of a simple logistic probability model yielded a classification accuracy of 98.36% and 97.20% for the training and testing sets, respectively.

Tree-Based Methods: The positive aspects of the tree-based methods are that they are visually easy to interpret, making them readily explainable. They can also handle qualitative predictors without the need for the intermediate step of coding them as dummy variables. However, single trees tend to be non-robust, meaning that a change in the data can cause a significant change in the tree. Their predictive accuracy tends not to be as competitive as other classification or regression approaches (James et al., 2021, p. 339-340). The classification accuracy of our single tree was 96.5%. Pruning is used to reduce noise in the tree that appears to be non-predictive. Pruning our single tree did not change the classification accuracy. This metric can be improved by aggregating the results of many single trees through methods such as bootstrap aggregation (bagging), random forests, and boosting. The bagging approach takes the outputs of the B classification trees as qualitative labels (classes). On average, each bagged tree only uses around two-thirds of the original data, and the remaining third is called out-of-bag (OOB). The simplest and most popular averaging step with classification trees is to take a majority vote. We determine the most commonly occurring label from the outputs of these B classification trees and take that as the average. Bagging was performed with 100 trees using the training data, which had a classification accuracy of 98.6% and an OOB estimated error of 5.87%. The random forest approach decorrelates the subsequently generated trees by making their average less variable while also decreasing the importance of predictor variables. Our random forest was built using 800 trees and had a classification accuracy of 97.9%. Performing this on a sequence of test accuracies allowed us to visualize what number of predictors are most accurate when performing classification. When the decision trees are built, the total number of predictors is not allowed to be considered at each split in the tree, increasing the chance that more minor predictors are utilized in the splits in addition to stronger predictors (James et al., 2021, p. 343-345).

Generalized Additive Models (GAMs): Generalized additive models fit non-linear functions to each predictor, allowing non-linear relationships to be modeled that standard regressions cannot capture, potentially allowing for more accurate predictions for our response variable. The additive quality of the model allows the effect of each predictor to be the response variable to be seen as the other predictors are held constant (James et al., 2021, p. 309). We considered a standard logistic regression and created a classification GAM. The classification accuracy was equal to 97.2%.

Support Vector Machines (SVM): Support vector machines utilize a generalization of a classification technique called the maximal margin classifier. This method creates a separating hyperplane with the furthest minimum distance from the training observations. The assumption is that the hyperplane will have a large margin on the training and test data and therefore classify them accurately (James et al., 2021, p. 368-373). This method works well when the data is separable by a linear boundary. However, SVM accounts for this by enlarging the feature space with kernels when this is not the case (James et al., 2021, p. 380 - 383). Kernels quantify the similarity of two observations and determine the flexibility of the decision boundary based on its type: linear, radial, or polynomial. This feature of SVM makes them ideal for classification with not easily separable data and qualitative response variables (James et al., 2021, p. 380 - 383). We generated support vector classifier models for linear, radial and polynomial scenarios. Of the 3, the linear model yielded the highest classification accuracy of 96.5%. Due to computational constraints we did not perform cross-validation on the classifier model, which could have potentially increased the accuracy.

Analysis

```
#prepare data
data <- read.csv("data.csv")
data <- data %>% mutate(
  diagnosis = dplyr::recode(diagnosis,
                     "M" = 1,
                     "B" = 0
  )
) %>% select(-X)

dim(data) #check # of predictors
```

```
## [1] 569  32
```

```
sum(is.na(data))
```

```
## [1] 0
```

The data was prepared by re-coding the dichotomous diagnosis variable, where a malignant diagnosis is now equal to 1 and a benign diagnosis is equal to 0. The resulting dataset was composed of 569 observations and 32 variables.

```
#simple linear probability
set.seed(100)
subset <- sample(nrow(data),nrow(data)*0.75)

train <- data[subset, ]
test <- data[-subset, ]

my_full_model <- glm(diagnosis ~ ., data = train, family = "binomial")
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(my_full_model)
```

```
##
## Call:
## glm(formula = diagnosis ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -4.605e-04  -2.000e-08  -2.000e-08   2.000e-08   4.191e-04
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)              7.300e+02  1.860e+06   0.000    1.000
## id                       1.790e-07  1.003e-04   0.002    0.999
## radius_mean             -5.862e+02  5.769e+05  -0.001    0.999
## texture_mean             1.593e+01  8.445e+03   0.002    0.998
## perimeter_mean          -5.141e+00  1.069e+05   0.000    1.000
##  [ reached getOption("max.print") -- omitted 27 rows ]
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5.6590e+02  on 425  degrees of freedom
## Residual deviance: 2.0512e-06  on 394  degrees of freedom
## AIC: 64
##
## Number of Fisher Scoring iterations: 25
```

We began our analysis with a simple probability model. A training and testing set of data was made to assess the fit of each model considered in this study. The model created for all 31 predictors showed no significance.

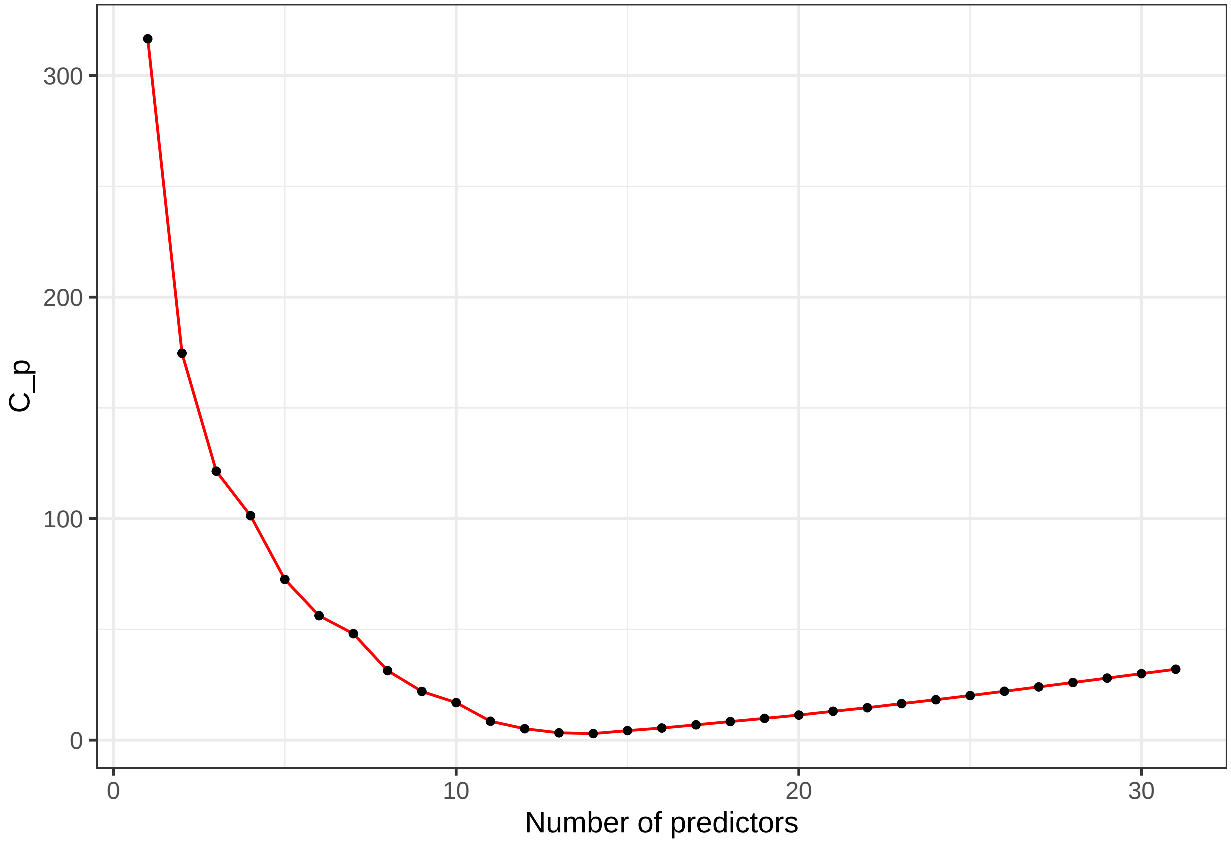
```
my_regsubset <- regsubsets(diagnosis ~ ., data = data, nvmax = 32)
summary(my_regsubset)
```

```
## Subset selection object
## Call: regsubsets.formula(diagnosis ~ ., data = data, nvmax = 32)
## 31 Variables  (and intercept)
##                         Forced in Forced out
## id                          FALSE      FALSE
## radius_mean                 FALSE      FALSE
## texture_mean                FALSE      FALSE
## perimeter_mean              FALSE      FALSE
## area_mean                   FALSE      FALSE
## smoothness_mean             FALSE      FALSE
## compactness_mean            FALSE      FALSE
## concavity_mean              FALSE      FALSE
## concave.points_mean         FALSE      FALSE
## symmetry_mean               FALSE      FALSE
##  [ reached getOption("max.print") -- omitted 21 rows ]
## 1 subsets of each size up to 31
## Selection Algorithm: exhaustive
##           id  radius_mean texture_mean perimeter_mean area_mean smoothness_mean
##           compactness_mean concavity_mean concave.points_mean symmetry_mean
##           fractal_dimension_mean radius_se texture_se perimeter_se area_se
##           smoothness_se compactness_se concavity_se concave.points_se
##           symmetry_se fractal_dimension_se radius_worst texture_worst
##           perimeter_worst area_worst smoothness_worst compactness_worst
##           concavity_worst concave.points_worst symmetry_worst
##           fractal_dimension_worst
##  [ reached getOption("max.print") -- omitted 31 rows ]
```

```
my_summary <- summary(my_regsubset)
names(my_summary)
```

```
## [1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
```

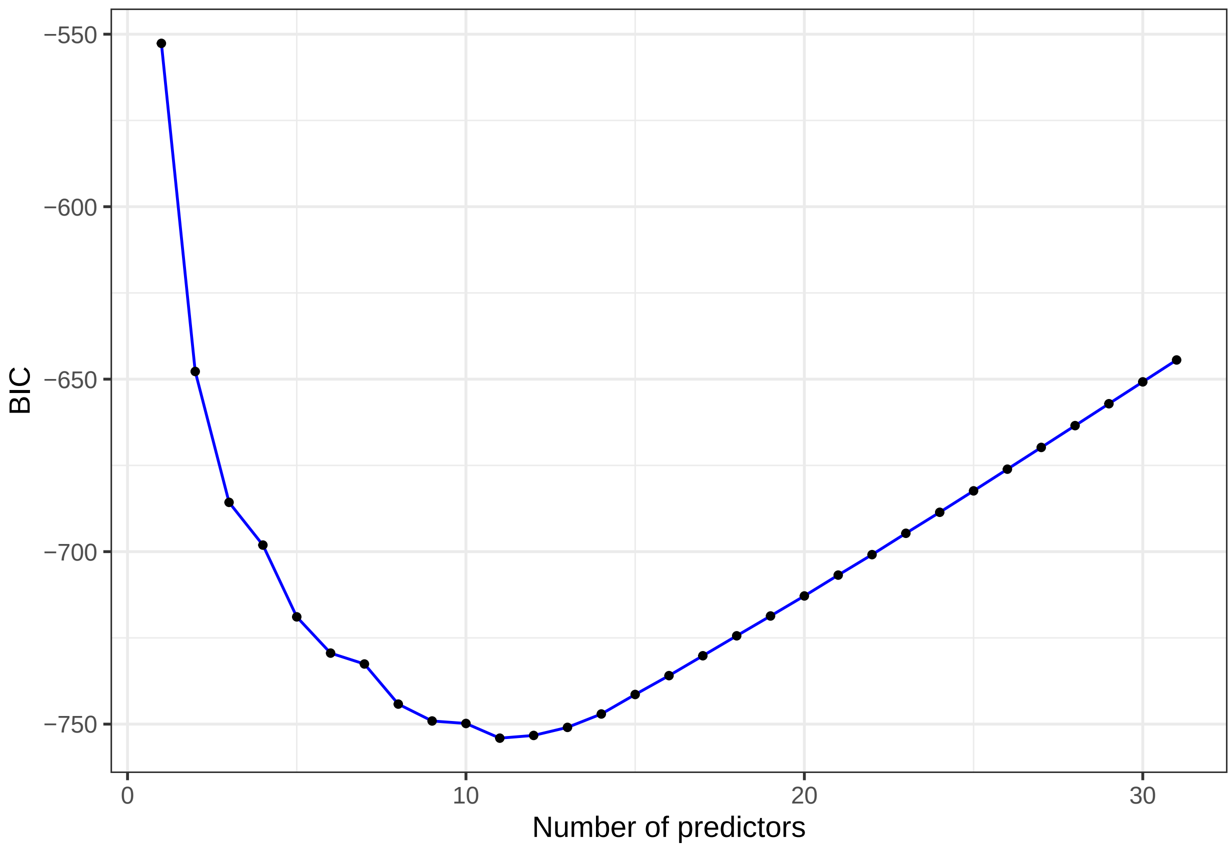
```
#check for lowest C_p
ggplot(as.data.frame(my_summary$cp),
       aes(x = seq(1:31), y = my_summary$cp)) +
  geom_line(color = "red") + geom_point(size = 1) +
  xlab("Number of predictors") + ylab("C_p") +
  theme_bw()
```



```
which.min(my_summary$cp)
```

```
## [1] 14
```

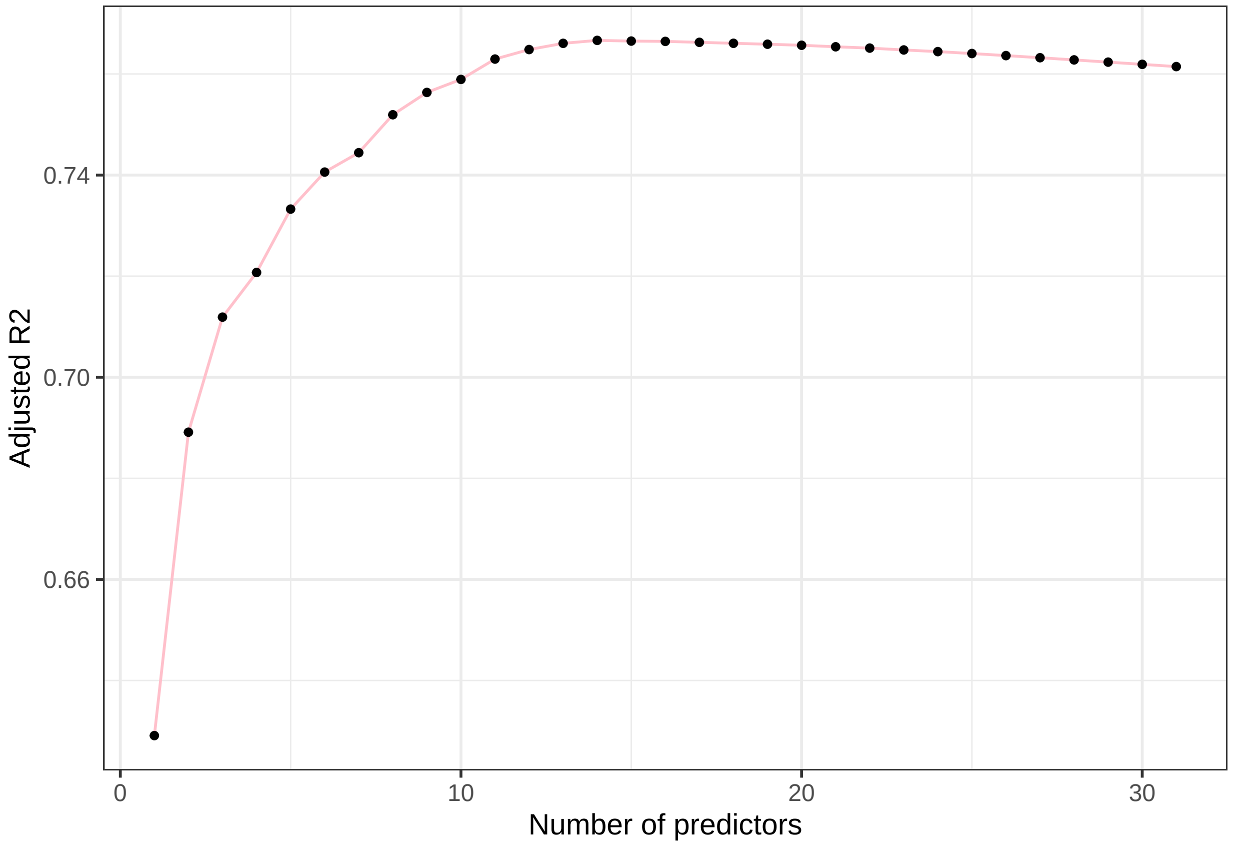
```
#check for lowest BIC
ggplot(as.data.frame(my_summary$bic),
       aes(x = seq(1:31), y = my_summary$bic)) +
  geom_line(color = "blue") + geom_point(size = 1) +
  xlab("Number of predictors") + ylab("BIC") +
  theme_bw()
```



```
which.min(my_summary$bic)
```

```
## [1] 11
```

```
#check for highest Adjusted R^2
ggplot(as.data.frame(my_summary$adjr2),
       aes(x = seq(1:31), y = my_summary$adjr2)) +
  geom_line(color = "pink") + geom_point(size = 1) +
  xlab("Number of predictors") + ylab("Adjusted R2") +
  theme_bw()
```



```
which.max(my_summary$adjr2)
```

```
## [1] 14
```

```
#select top 14 predictors
my_summary$which[14, ]
```

```
##            (Intercept)                     id            radius_mean 
##                   TRUE                  FALSE                   TRUE 
##           texture_mean         perimeter_mean              area_mean 
##                  FALSE                  FALSE                  FALSE 
##        smoothness_mean       compactness_mean         concavity_mean 
##                  FALSE                   TRUE                   TRUE 
##    concave.points_mean          symmetry_mean fractal_dimension_mean 
##                   TRUE                  FALSE                  FALSE 
##              radius_se             texture_se           perimeter_se 
##                   TRUE                  FALSE                  FALSE 
##                area_se          smoothness_se         compactness_se 
##                  FALSE                   TRUE                  FALSE 
##           concavity_se      concave.points_se 
##                   TRUE                   TRUE 
##  [ reached getOption("max.print") -- omitted 12 entries ]
```

Using best subset selection, we determined which predictors had the highest probability of correctly identifying a tumor sample. This was done by examining the graphs above for each subset. The subset with the highest adjusted R^2 and the lowest C_p and BIC overall contained 14 variables. Those predictors were then used to create another simple probability model.

```
#we now generate a logistic model using top predictors
my_glm <- glm(
  diagnosis ~ radius_mean + compactness_mean +
  concavity_mean + concave.points_mean + radius_se +
  concavity_se + smoothness_se + concave.points_se +
  radius_worst + texture_worst + area_worst + concavity_worst +
  symmetry_worst + fractal_dimension_worst,
  data = train, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(my_glm)
```

```
##
## Call:
## glm(formula = diagnosis ~ radius_mean + compactness_mean + concavity_mean +
##     concave.points_mean + radius_se + concavity_se + smoothness_se +
##     concave.points_se + radius_worst + texture_worst + area_worst +
##     concavity_worst + symmetry_worst + fractal_dimension_worst,
##     family = "binomial", data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.38017  -0.00139  -0.00004   0.00000   2.42006
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -22.15278   23.40460  -0.947  0.34389
## radius_mean               -5.81249    2.44897  -2.373  0.01762 *
## compactness_mean        -134.44211   59.82792  -2.247  0.02463 *
## concavity_mean            54.08794   52.75086   1.025  0.30520
## concave.points_mean      178.69110   93.52009   1.911  0.05604 .
## radius_se                 16.36221    6.04257   2.708  0.00677 **
## concavity_se            -209.43364   87.01465  -2.407  0.01609 *
## smoothness_se           -124.71588  253.53088  -0.492  0.62278
## concave.points_se        916.39770  409.33060   2.239  0.02517 *
## radius_worst               0.91631    2.97384   0.308  0.75799
## texture_worst              0.51777    0.17650   2.934  0.00335 **
## area_worst                 0.06229    0.03263   1.909  0.05628 .
## concavity_worst           25.34325   12.83256   1.975  0.04828 *
## symmetry_worst            21.55567   11.89479   1.812  0.06996 .
## fractal_dimension_worst   58.96946   87.98977   0.670  0.50274
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 565.900  on 425  degrees of freedom
## Residual deviance:  37.177  on 411  degrees of freedom
## AIC: 67.177
```

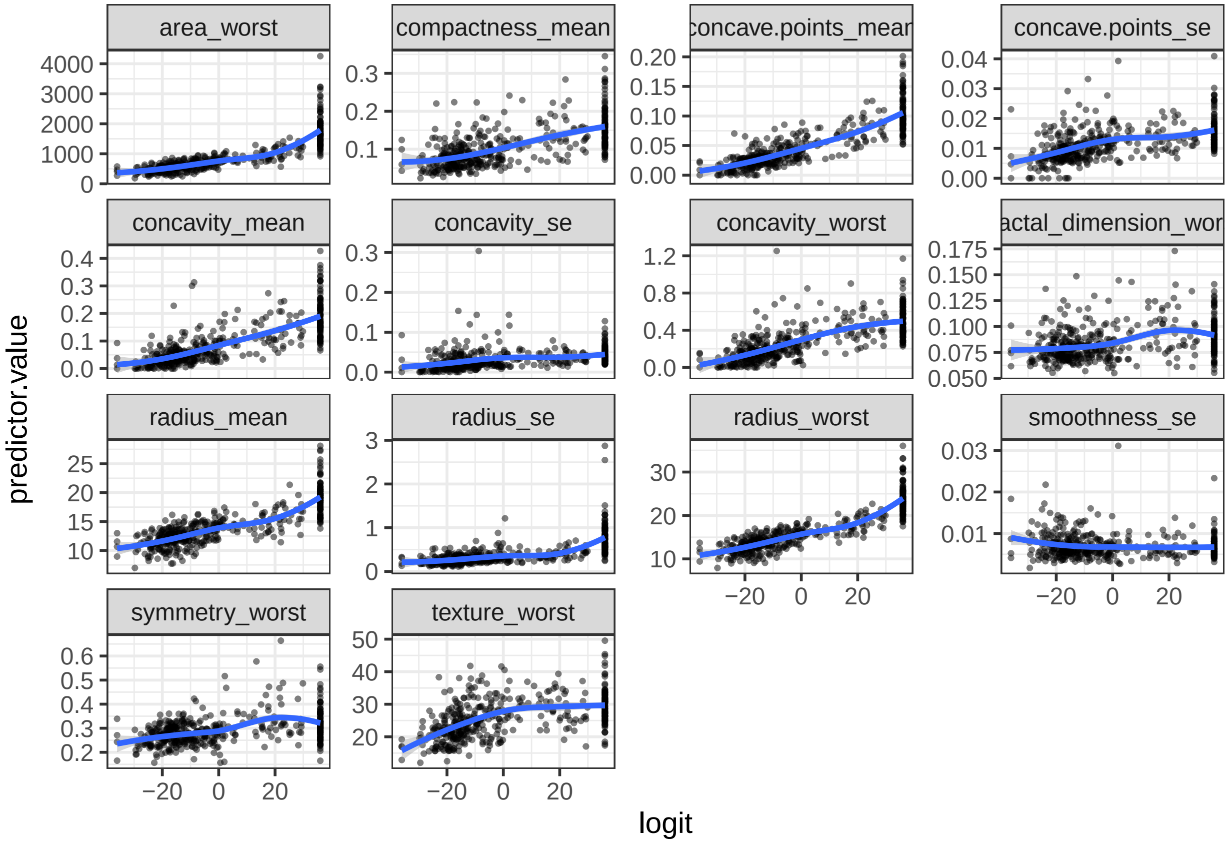
```
##
## Number of Fisher Scoring iterations: 12
```

```
#and check assumptions

#Linearity
probabilities <- predict(my_glm, type = "response")
predicted.classes <- ifelse(probabilities > 0.5, "pos", "neg")
head(predicted.classes)
```

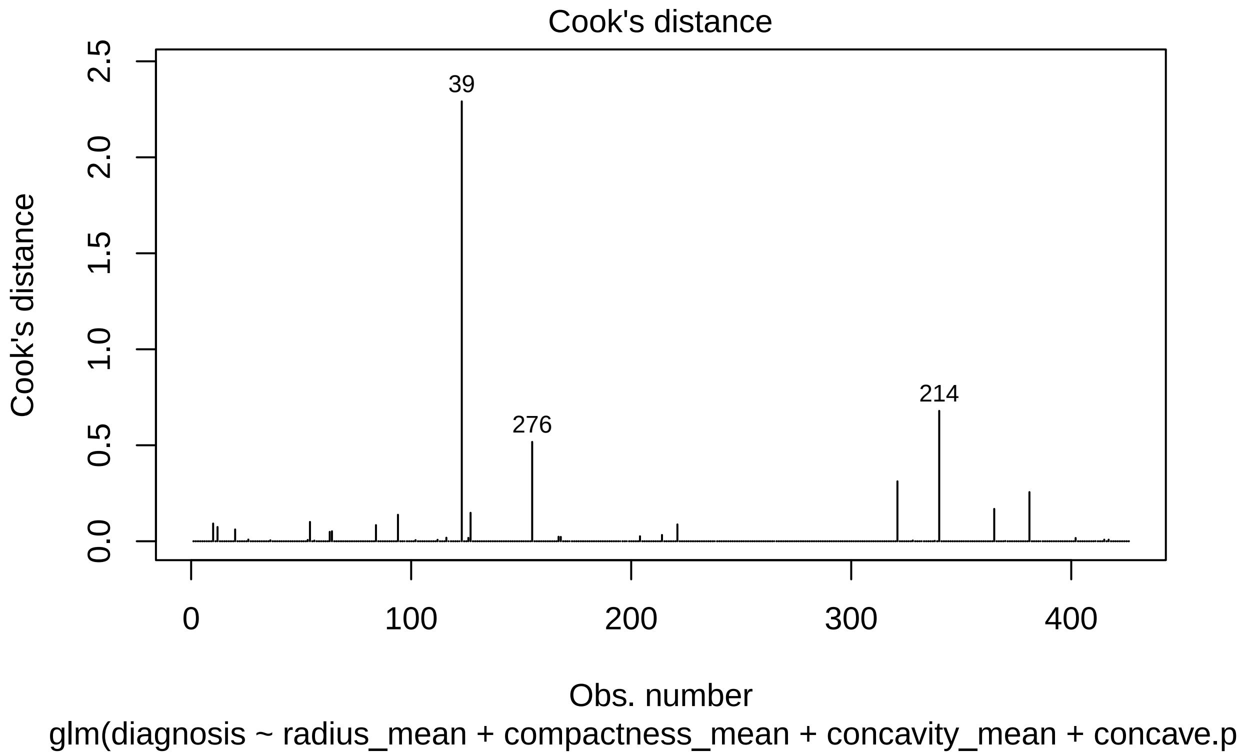
```
##   503   358   470   516    98     7
## "neg" "neg" "neg" "neg" "neg" "pos"
```

```
mydata <- train %>% select(
  radius_mean, compactness_mean, concavity_mean,
  concave.points_mean, radius_se, concavity_se, smoothness_se,
  concave.points_se, radius_worst, texture_worst, area_worst,
  concavity_worst, symmetry_worst, fractal_dimension_worst
) %>%
  mutate(logit = log(probabilities/(1 - probabilities))) %>%
  gather(key = "predictors", value = "predictor.value", -logit)

ggplot(mydata, aes(logit, predictor.value))+
  geom_point(size = 0.5, alpha = 0.5) +
  geom_smooth(method = "loess") +
  theme_bw() +
  facet_wrap(~predictors, scales = "free_y")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



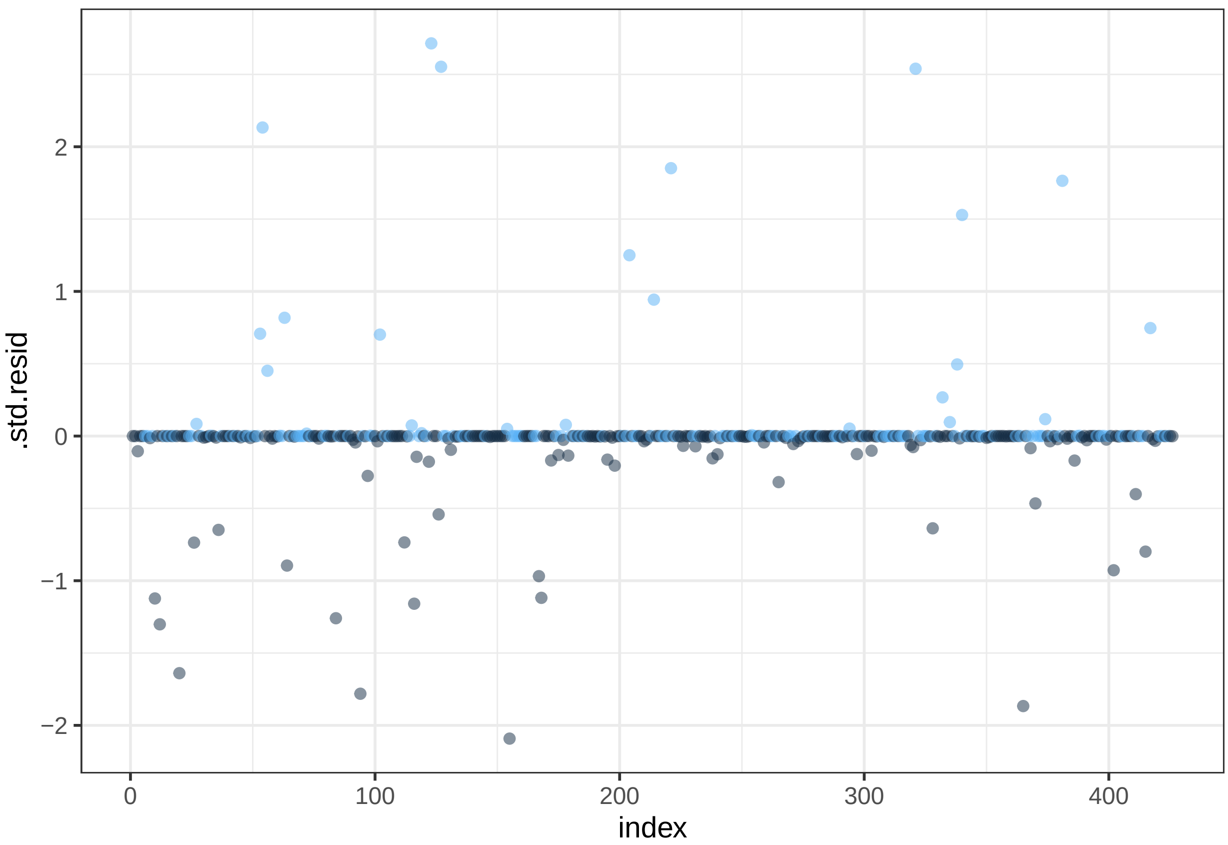
```
#Influential Values
plot(my_glm, which = 4, id.n = 3)
```



```
model.data <- augment(my_glm) %>%
  mutate(index = 1:n())

model.data %>% top_n(3, .cooksd)
```

```
## # A tibble: 3 x 23
##   .rownames diagnosis radius_mean compactness_mean concavity_mean
##   <chr>         <dbl>       <dbl>            <dbl>          <dbl>
## 1 39                1        15.0           0.0513         0.0240
## 2 276               0        11.9           0.0721         0.0593
## 3 214               1        17.4           0.115          0.168
## # ... with 18 more variables: concave.points_mean <dbl>, radius_se <dbl>,
## #   concavity_se <dbl>, smoothness_se <dbl>, concave.points_se <dbl>,
## #   radius_worst <dbl>, texture_worst <dbl>, area_worst <dbl>,
## #   concavity_worst <dbl>, symmetry_worst <dbl>, fractal_dimension_worst <dbl>,
## #   .fitted <dbl>, .resid <dbl>, .std.resid <dbl>, .hat <dbl>, .sigma <dbl>,
## #   .cooksd <dbl>, index <int>
```

```
ggplot(model.data, aes(index, .std.resid)) +
  geom_point(aes(color = diagnosis), alpha = .5) +
  theme_bw() + theme(legend.position = "none")
```



```
#Multicollinearity
vif(my_glm)
```

```
##              radius_mean        compactness_mean          concavity_mean
##                70.268171               32.810072               30.481216
##      concave.points_mean               radius_se            concavity_se
##                16.682959                9.836486               30.162268
##            smoothness_se       concave.points_se            radius_worst
##                 6.176541               31.224551              109.438339
##            texture_worst              area_worst         concavity_worst
##                 6.972530              119.951632               24.745424
##           symmetry_worst fractal_dimension_worst
##                 2.939657               13.076812
```

Assessing the performance of our new model, we see that we have now identified significant predictors for determining the diagnosis of tumor samples. We then checked the model for logistic assumptions. The predictors present in this model all had a relatively linear relationship with the logit of the outcome, which means the linearity assumption is met. Additionally, we checked for outliers present using Cook's distance, and mapped the standardized residuals. There appear to be no extreme influential values. Finally, the *vif()* function from the *car* library was used to assess multicollinearity. The high values suggest that our predictors have a high amount of collinearity. We keep this in mind while creating future models.

```
#compute classification accuracy using training set
predicted_glm_train <- predict(my_glm, train, type = "response")
yhat_predict_train <- ifelse(predicted_glm_train > 0.5, 1, 0)
```

```r
table_train <- table(y = train$diagnosis, yhat = yhat_predict_train)
table_train
```

```
##    yhat
## y     0   1
##   0 261   3
##   1   4 158
```

```r
accuracy_train <- sum(diag(table_train))/ sum(table_train)
accuracy_train #0.98356
```

```
## [1] 0.9835681
```

```r
#compute classification accuracy using test set
predicted_glm_test <- predict(my_glm, test, type = "response")
yhat_predict_test <- ifelse(predicted_glm_test > 0.5, 1, 0)

table_test <- table(y = test$diagnosis, yhat = yhat_predict_test)
table_test
```

```
##    yhat
## y    0  1
##   0 89  4
##   1  0 50
```

```r
accuracy_test <- sum(diag(table_test))/ sum(table_test)
accuracy_test
```

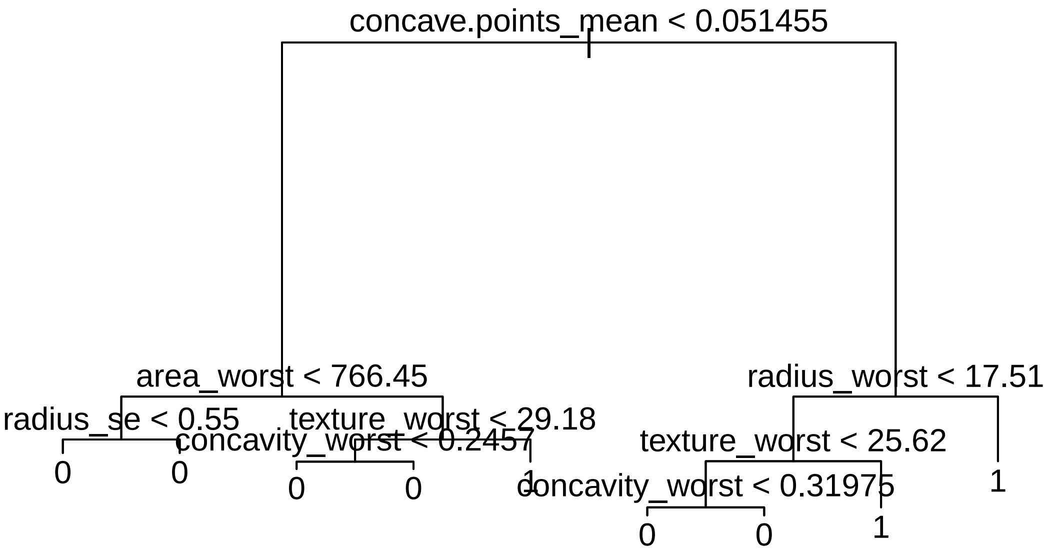
```
## [1] 0.972028
```

```r
train <- train %>% select(
  radius_mean, compactness_mean, concavity_mean,
  concave.points_mean, radius_se, concavity_se, smoothness_se,
  concave.points_se, radius_worst, texture_worst, area_worst,
  concavity_worst, symmetry_worst, fractal_dimension_worst, diagnosis
)

test <- test %>% select(
  radius_mean, compactness_mean, concavity_mean,
  concave.points_mean, radius_se, concavity_se, smoothness_se,
  concave.points_se, radius_worst, texture_worst, area_worst,
  concavity_worst, symmetry_worst, fractal_dimension_worst, diagnosis
)
```

```r
#create single tree
set.seed(100)
my_tree <- tree(as.factor(diagnosis) ~ ., data = train)
summary(my_tree)
```

```
##
## Classification tree:
## tree(formula = as.factor(diagnosis) ~ ., data = train)
## Variables actually used in tree construction:
## [1] "concave.points_mean" "area_worst"          "radius_se"
## [4] "texture_worst"       "concavity_worst"     "radius_worst"
## Number of terminal nodes:  9
## Residual mean deviance:  0.1478 = 61.64 / 417
## Misclassification error rate: 0.02817 = 12 / 426
```

```
plot(my_tree)
text(my_tree, pretty = 0)
```

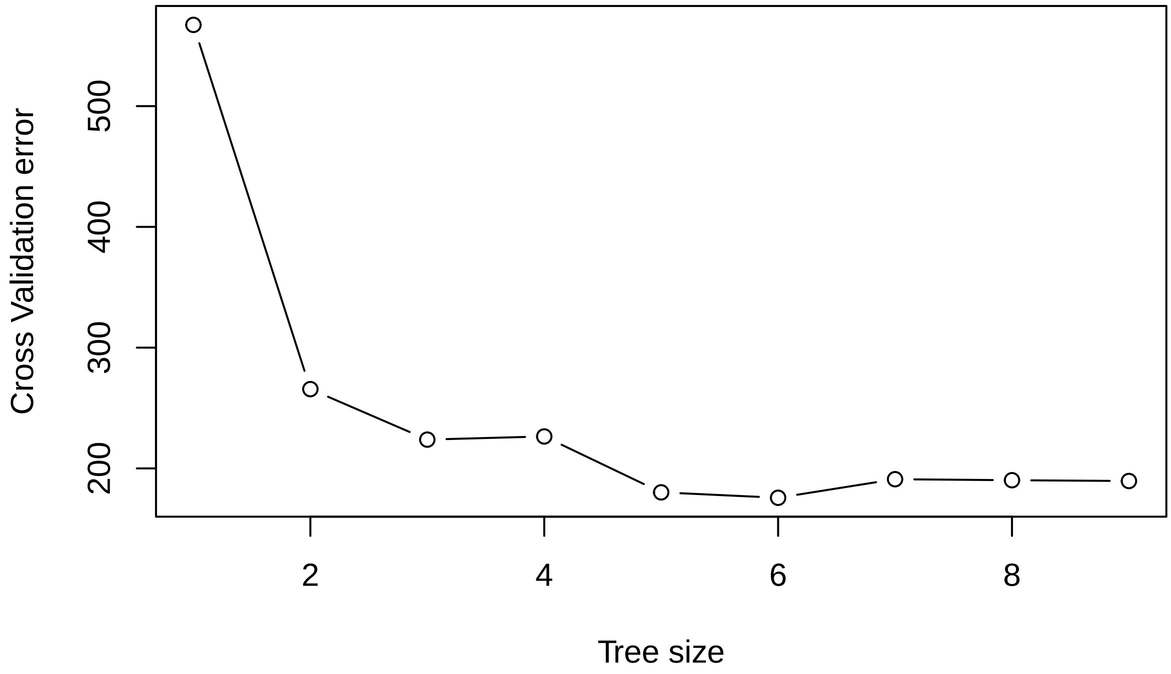


```
#compute classification accuracy of test set
tumor_pred <- predict(my_tree, newdata = test, type = "class")
table_tumor <- table(tumor_pred, test$diagnosis)

accuracy_tumor_test <- sum(diag(table_tumor))/ sum(table_tumor)
accuracy_tumor_test
```

```
## [1] 0.965035
```

```
#does pruning improve our tree?
set.seed(100)
```

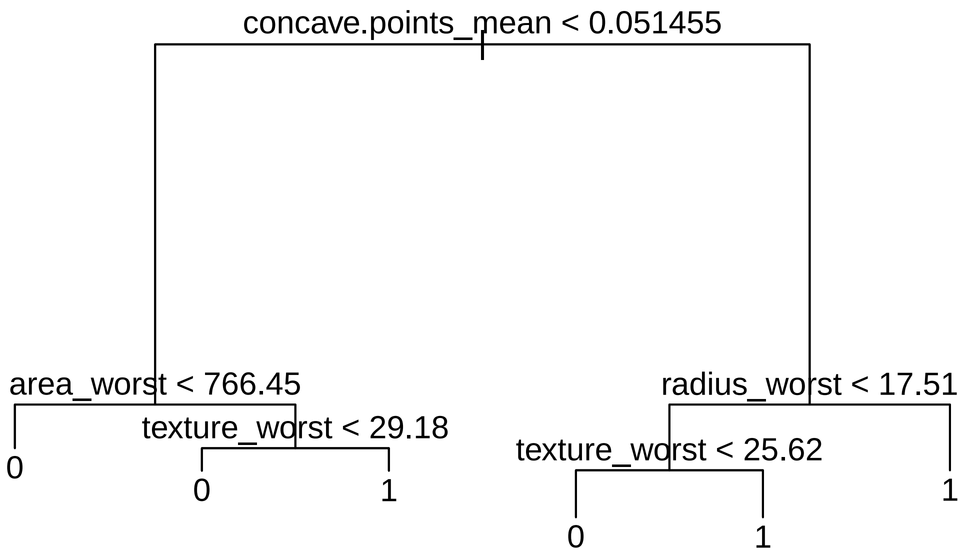
```
cv_data <- cv.tree(my_tree,  K = 10) #10-fold cross validation
plot(cv_data$size, cv_data$dev, type = 'b',
     xlab = 'Tree size', ylab = 'Cross Validation error')
```



```
#use K-fold cross-validation to determine best prune size
prune_cv <- prune.tree(my_tree, best = 6)
summary(prune_cv)
```

```
##
## Classification tree:
## snip.tree(tree = my_tree, nodes = c(10L, 12L, 4L))
## Variables actually used in tree construction:
## [1] "concave.points_mean" "area_worst"          "texture_worst"
## [4] "radius_worst"
## Number of terminal nodes:  6
## Residual mean deviance:  0.2089 = 87.75 / 420
## Misclassification error rate: 0.02817 = 12 / 426
```

```
plot(prune_cv)
text(prune_cv)
```



```r
#compute classification accuracy using test set
prune_pred <- predict(prune_cv, newdata = test, type = "class")
table_prune <- table(prune_pred, test$diagnosis)

accuracy_prune <- sum(diag(table_prune))/ sum(table_prune)
accuracy_prune
```

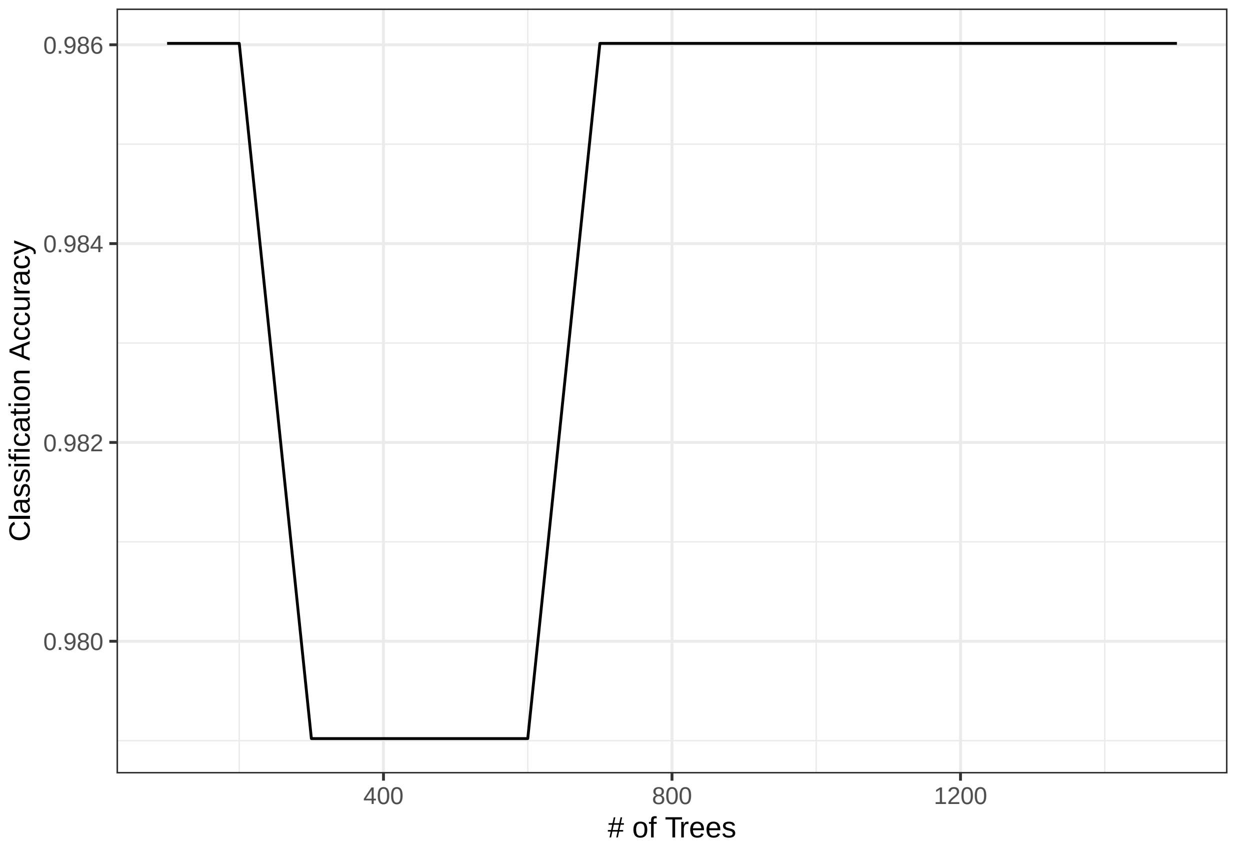
```
## [1] 0.965035
```

Pruning our model resulted in no change in classification accuracy.

```r
#bagging with all predictors
set.seed(100)
ncol <- ncol(train)

#start by bagging with 100 trees
my_bag <- randomForest(
  as.factor(diagnosis) ~ ., data = train, mtry = ncol - 1,
  ntree = 100, importance = TRUE)

#lets check classification accuracy
bag_accuracy <- predict(my_bag, newdata = test)
yhat_predict <- ifelse(bag_accuracy == 1, 1, 0)
table_bag <- table(y = test$diagnosis, yhat = yhat_predict)
accuracy <- sum(diag(table_bag))/ sum(table_bag)
accuracy
```

```
## [1] 0.986014
```

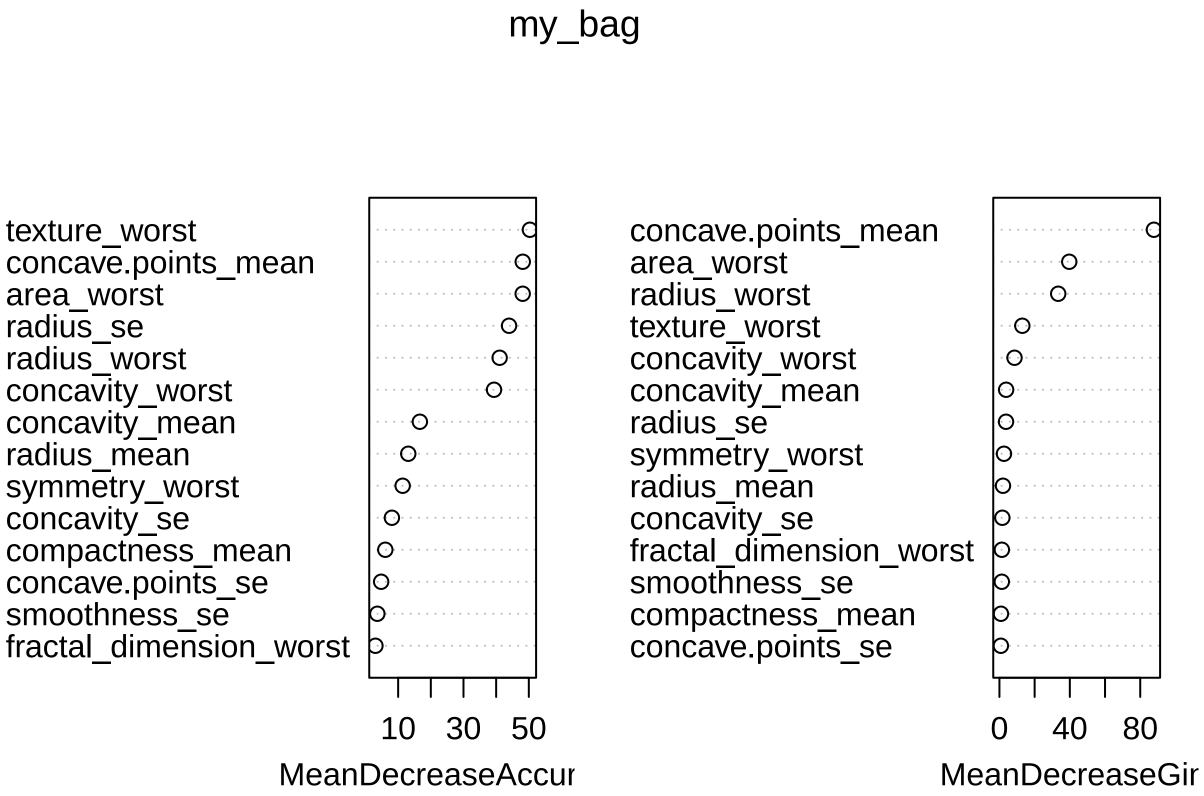
```r
#now we check what number of trees gives us the highest classification accuracy
set.seed(100)
sequence_tree <- seq(100, 1500, by = 100)

#we create a sequence of test accuracies with 0 values for each number of trees
accuracy <- rep(0, length(sequence_tree))

for (i in 1:length(sequence_tree))
{
  my_bag <- randomForest(
    as.factor(diagnosis) ~ ., data = train, mtry = ncol - 1,
    ntree = sequence_tree[i], importance = TRUE)
  bag_accuracy <- predict(my_bag, newdata = test)
  yhat_predict <- ifelse(bag_accuracy == 1, 1, 0)
  table_bag <- table(y = test$diagnosis, yhat = yhat_predict)
  accuracy[i] <- sum(diag(table_bag))/ sum(table_bag)
}

#plot classification accuracy
bag_accuracy_df <- as.data.frame(accuracy)
ggplot(data = bag_accuracy_df, aes(
  x = seq(100,1500, by = 100), y = accuracy
  )) + geom_line() +
  xlab("# of Trees") +
  ylab("Classification Accuracy") +
  theme_bw()
```



```
importance(my_bag)
```

```
##                                 0          1 MeanDecreaseAccuracy
## radius_mean             13.074471 -1.3242951            13.095010
## compactness_mean         4.865454  3.5634337             6.066557
## concavity_mean          12.989543 10.1981276            16.648019
## concave.points_mean     26.797932 39.2084151            48.146941
## radius_se               43.023460 13.1742875            43.948355
## concavity_se             6.340488  5.3353084             8.079104
## smoothness_se            1.592089  4.5685639             3.625945
## concave.points_se        1.835923  5.0243898             4.800909
## radius_worst            34.669759 22.6136177            41.073220
## texture_worst           32.603287 43.1923895            50.337697
## area_worst              31.819689 37.6534830            48.125140
## concavity_worst          7.456778 37.8584465            39.348771
## symmetry_worst           6.275544  9.7461843            11.377947
## fractal_dimension_worst  3.936671 -0.8392976             3.045492
##                         MeanDecreaseGini
## radius_mean                    2.0234503
## compactness_mean               0.8685765
## concavity_mean                 3.8052483
## concave.points_mean           87.7722714
## radius_se                      3.7359881
## concavity_se                   1.6487486
## smoothness_se                  1.3503125
## concave.points_se              0.8177911
## radius_worst                  33.3939957
## texture_worst                 12.9936117
## area_worst                    39.6743194
## concavity_worst                8.5541388
## symmetry_worst                 2.5742426
## fractal_dimension_worst        1.3732113
```

```
varImpPlot(my_bag)
```



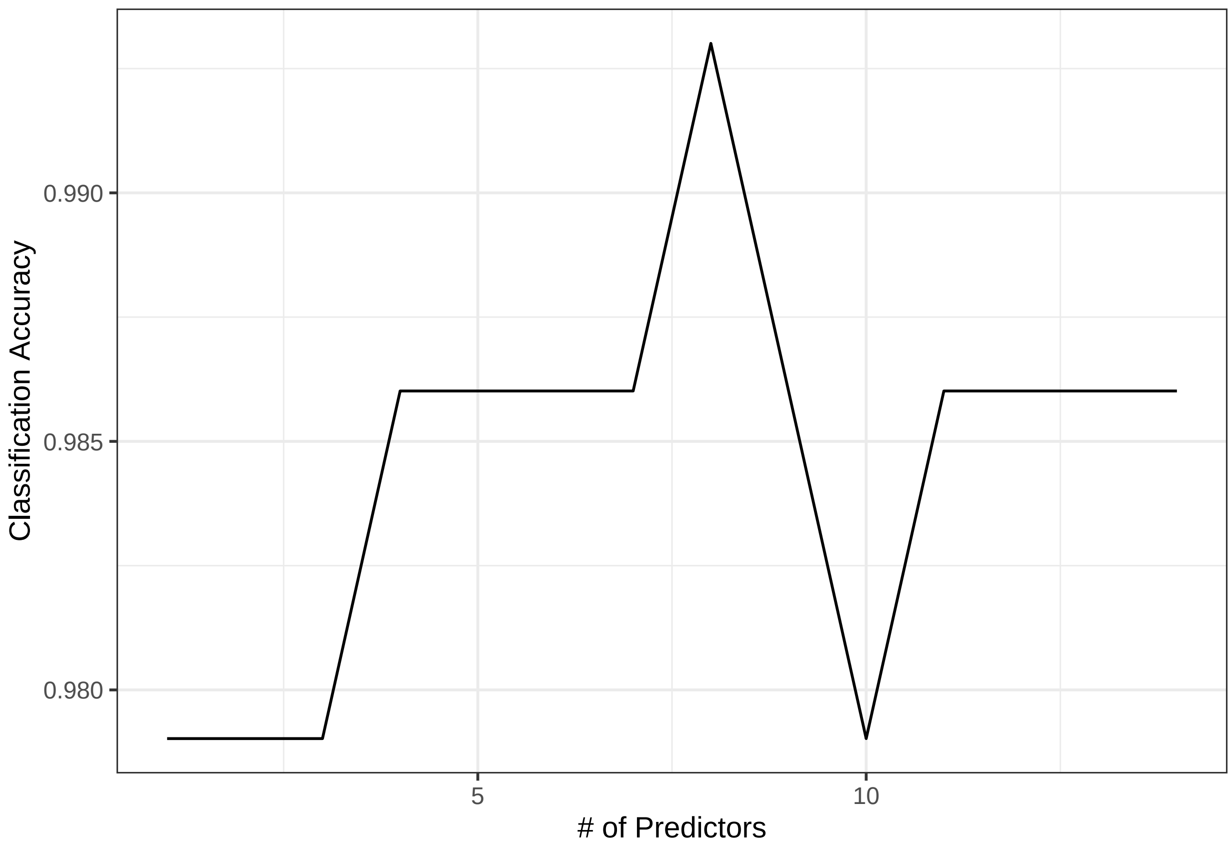
After bagging for a sequence of trees, we can see that the ideal number of trees would be around 800. Viewing the importance of our variables following bagging, it is clear that texture_worst, area_worst, concave.points_mean, radius_se, concavity_worst, and radius_worst are the most important predictors in the model. Our model suffers a large decrease in accuracy if any of these variables are removed from the model. We can try to decrease the importance of these variables by generating a random forest.

```
#perform random forest
set.seed(100)
my_forest <- randomForest(
  as.factor(diagnosis) ~ ., data = train,
  ntree = 800, importance = TRUE)

#now lets check classification accuracy
forest_accuracy <- predict(my_forest, newdata = test)
yhat_predict <- ifelse(forest_accuracy == 1, 1, 0)
table_forest <- table(y = test$diagnosis, yhat = yhat_predict)
```

```
accuracy <- sum(diag(table_forest))/ sum(table_forest)
accuracy #0.979021
```

```
## [1] 0.979021
```

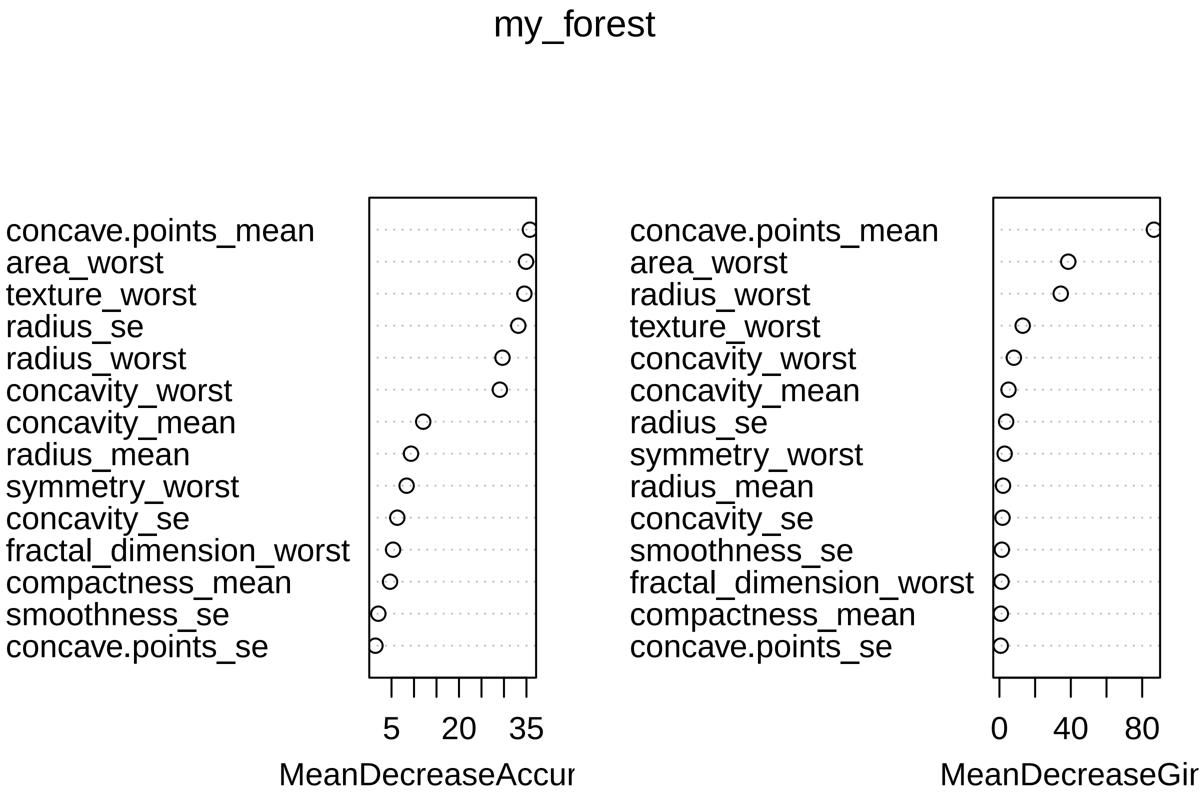
```
#now we consider the best number of predictors
set.seed(100)
sequence_predictors <- seq(1, ncol - 1, by = 1)

# We create a sequence of test accuracies with 0 values for each number of trees
accuracy <- rep(0, length(sequence_predictors))

for (i in 1:length(sequence_predictors))
{
  my_forest <- randomForest(
    as.factor(diagnosis) ~ ., data = train,
    mtry = sequence_predictors[i],
    ntree = 800, importance = TRUE)
  forest_accuracy <- predict(my_forest, newdata = test)
  yhat_predict <- ifelse(forest_accuracy == 1, 1, 0)
  table_forest <- table(y = test$diagnosis, yhat = yhat_predict)
  accuracy[i] <- sum(diag(table_forest))/ sum(table_forest)
}

#plot classification accuracy
forest_accuracy_df <- as.data.frame(accuracy)
ggplot(data = forest_accuracy_df, aes(
  x = seq(1,14, by = 1), y = accuracy
  )) + geom_line() +
  xlab("# of Predictors") +
  ylab("Classification Accuracy") +
  theme_bw()
```



```
importance(my_forest)
```

```
##                                0          1 MeanDecreaseAccuracy
## radius_mean             9.2628372 -0.7306048             9.332745
## compactness_mean        3.2984042  3.4790060             4.671567
## concavity_mean          9.3176499  7.2793036            12.045383
## concave.points_mean    18.6253236 29.5097813            35.769663
## radius_se              31.4790504 10.1473125            33.181962
## concavity_se            4.7683752  3.4250541             6.283441
## smoothness_se           0.3073673  3.4664566             2.072072
## concave.points_se      -0.1022692  2.2433630             1.422762
## radius_worst           25.3990268 16.9902305            29.671383
## texture_worst          21.5764590 30.5742759            34.542752
## area_worst             24.6479226 26.2838486            34.907909
## concavity_worst         5.1449280 28.1510312            29.082556
## symmetry_worst          4.2943010  7.1367146             8.358579
## fractal_dimension_worst  6.1287994 -0.6079823             5.347864
##                        MeanDecreaseGini
## radius_mean                   2.0132212
## compactness_mean              0.7957408
## concavity_mean                5.1029305
## concave.points_mean          86.5779046
## radius_se                     3.7495246
## concavity_se                  1.7334262
## smoothness_se                 1.3648591
## concave.points_se              0.7322662
## radius_worst                  34.3616040
## texture_worst                 13.0510611
## area_worst                    38.5737212
## concavity_worst                8.0870191
## symmetry_worst                 2.9312584
## fractal_dimension_worst        1.1841603
```

```
varImpPlot(my_forest)
```



We can conclude that using 6 predictors produces the highest classification accuracy in our model. Additionally, there is a decrease in importance among these variables, which suggests an improvement from bagging.

```
#perform classification gam
set.seed(100)

#first consider standard logistic regression
my_gam <- gam(as.factor(diagnosis) ~ ., family = binomial, data = train)
summary(my_gam)
```

```
##
## Call: gam(formula = as.factor(diagnosis) ~ ., family = binomial, data = train)
## Deviance Residuals:
##        Min         1Q     Median         3Q        Max
## -1.380e+00 -1.394e-03 -3.657e-05  2.107e-08  2.420e+00
```

```
##
## (Dispersion Parameter for binomial family taken to be 1)
##
##     Null Deviance: 565.9 on 425 degrees of freedom
## Residual Deviance: 37.1772 on 411 degrees of freedom
## AIC: 67.1772
##
## Number of Local Scoring Iterations: 12
##
## Anova for Parametric Effects
##                          Df Sum Sq Mean Sq F value    Pr(>F)
## radius_mean               1  1.532  1.5321 12.8996 0.0003684 ***
## compactness_mean          1  0.465  0.4651  3.9164 0.0484845 *
## concavity_mean            1  0.496  0.4962  4.1780 0.0415898 *
## concave.points_mean       1  0.019  0.0188  0.1582 0.6910476
## radius_se                 1  0.033  0.0331  0.2784 0.5980274
## concavity_se              1  0.174  0.1745  1.4691 0.2261832
## smoothness_se             1  0.554  0.5545  4.6687 0.0312944 *
## concave.points_se         1  0.045  0.0448  0.3770 0.5395760
## radius_worst              1  0.002  0.0018  0.0153 0.9017448
## texture_worst             1  0.509  0.5093  4.2882 0.0390016 *
## area_worst                1  0.651  0.6505  5.4772 0.0197438 *
## concavity_worst           1  6.243  6.2432 52.5664 2.080e-12 ***
## symmetry_worst            1  3.052  3.0523 25.6995 6.045e-07 ***
## fractal_dimension_worst   1  0.449  0.4491  3.7817 0.0524970 .
## Residuals               411 48.814  0.1188
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

```r
#compute classification accuracy on test set
gam_pred <- predict(my_gam, newdata = test, type = "response")
yhat_predict <- ifelse(gam_pred > 0.5, 1, 0)
table_gam <- table(y = test$diagnosis, yhat = yhat_predict)
accuracy <- sum(diag(table_gam))/ sum(table_gam)
accuracy
```

```
## [1] 0.972028
```

The results of the GAM are seen above. For a standard logistic regression, we have a classification accuracy of 97.20%.

```r
#support vector classifier - linear
set.seed(100)

my_svc <- svm(
  as.factor(diagnosis) ~ .,
  data = train, kernel = "linear",
  cost = 0.1, scale = FALSE)



svc_pred <- predict(my_svc, newdata = test)
table_svc <- table(svc_pred, test$diagnosis)
```

```r
accuracy <- sum(diag(table_svc)) / sum(table_svc)
accuracy
```

```
## [1] 0.965035
```

```r
#support vector classifier - radial
set.seed(100)

my_svc <- svm(
  as.factor(diagnosis) ~ .,
  data = train, kernel = "radial",
  cost = 0.1, scale = FALSE)


svc_pred <- predict(my_svc, newdata = test)
table_svc <- table(svc_pred, test$diagnosis)

accuracy <- sum(diag(table_svc)) / sum(table_svc)
accuracy
```

```
## [1] 0.6503497
```

```r
#support vector classifier - polynomial
set.seed(100)

my_svc <- svm(
  as.factor(diagnosis) ~ .,
  data = train, kernel = "polynomial",
  cost = 0.1, scale = FALSE)


svc_pred <- predict(my_svc, newdata = test)
table_svc <- table(svc_pred, test$diagnosis)

accuracy <- sum(diag(table_svc)) / sum(table_svc)
accuracy
```

```
## [1] 0.8531469
```

Conclusion / Future Work

Our results demonstrate that the model most successful at accurately classifying breast tumor samples was the non-linear tree-based method, bagging. However, it is of note that all models performed exceptionally well in classifying our dataset. Furthermore, of the thirty-two original predictors in our dataset, we determined that only fourteen were significant to keep in the eventually compared models. The fact that both the linear and non-linear methods yielded comparable accuracies suggests that the added complexity that non-linear methods contain is unnecessary to obtain optimal classification results on this particular data. Expanding on the idea of reducing complexity, the halving of our predictors after best subset selection demonstrates the importance of identifying essential versus non-essential predictors so that the most interpretable and straightforward computational approach is achieved. A limitation of our exploration is that computer hardware constraints prevented cross-validation from being applied to the SVM method. This could mean that the optimal combination of parameters used in our SVM may not have been used as we could not calculate the test error rates from cross-validation.

The implications of our research include a better understanding of breast tumor features that are important in discerning malignant breast tumors from benign ones. These findings will help imaging interpretation specialists regarding the characteristics to look for and the corresponding thresholds that differentiate the tumor classification. Even further, in ambiguous tumor cases, these models can be comfortably relied upon to classify the tumor with high accuracy, allowing healthcare professionals to develop a plan of action for their patients that fits their needs quickly. A modification of our approach would be to create the same models on different training, and test data splits to characterize the relationship between classification proficiency and the amount of training data.

An expansion of our current research would be to develop a model that can predict a person's likelihood of developing breast cancer based on lifestyle and genetic data. Considering this model in tandem with our current approach would allow a more holistic view of a patient's health and family history related to breast cancer, thereby allowing doctors and specialists to be more informed when making decisions so that their patients receive exceptional care. In addition, this synthesized strategy can also serve as an opportunity for patients to be better informed about their health, allowing for greater, empowered dialogue exchange between themselves and their doctor. As we have seen, computational models and methods have and will continue to shape the future of cancer diagnostics and treatment.

References

- Akram, M., Iqbal, M., Daniyal, M., & Khan, A. U. (2017). Awareness and current knowledge of breast cancer. Biological research, 50(1), 33. https://doi.org/10.1186/s40659-017-0140-9

- Aruleba, K., Obaido, G., Ogbuokiri, B., Fadaka, A. O., Klein, A., Adekiya, T. A., & Aruleba, R. T. (2020). Applications of Computational Methods in Biomedical Breast Cancer Imaging Diagnostics: A Review. Journal of imaging, 6(10), 105. https://doi.org/10.3390/jimaging6100105

- Bhushan, A., Gonsalves, A., & Menon, J. U. (2021). Current State of Breast Cancer Diagnosis, Treatment, and Theranostics. Pharmaceutics, 13(5), 723. https://doi.org/10.3390/pharmaceutics13050723

- DeSantis, C. E., Ma, J., Gaudet, M. M., Newman, L. A., Miller, K. D., Goding Sauer, A., Jemal, A., & Siegel, R. L. (2019). Breast cancer statistics, 2019. CA: A Cancer Journal for Clinicians, 69(6), 438–451. https://doi.org/10.3322/caac.21583

- Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

- Dubey, A.K., Gupta, U. & Jain, S. (2016). Analysis of k-means clustering approach on the breast cancer Wisconsin dataset. Int J CARS 11, 2033–2047. https://doi-org.ezproxy.lib.utexas.edu/10.1007/s11548-016-1437-9

- James, G., Witten, D., Hastie, T. & Tibshirani, R. (2021). An Introduction to Statistical Learning: with applications in R (2nd ed.). Springer Texts in Statistics. https://hastie.su.domains/ISLR2/ISLRv2_website.pdf

- Mohammad, Teete, R., Al-Aaraj, H., Rubbai, Y. S. Y., & Arabyat, M. M. (2022). Diagnosis of Breast Cancer Pathology on the Wisconsin Dataset with the Help of Data Mining Classification and Clustering Techniques. Applied Bionics and Biomechanics, 2022, 6187275–6187275. https://doi.org/10.1155/2022/6187275

- Seldon. (2021, October 16). Supervised vs unsupervised learning explained. Seldon. https://www.seldon.io/supervised-vs-unsupervised-learning-explained#:~:text=The%20main%20difference%20between%20supervise

- Wei, M., Du, Y., Wu, X., Su, Q., Zhu, J., Zheng, L., Lv, G., & Zhuang, J. (2020). A Benign and Malignant Breast Tumor Classification Method via Efficiently Combining Texture and Morphological Features on Ultrasound Images. Computational and mathematical methods in medicine, 2020, 5894010. https://doi.org/10.1155/2020/5894010

- World Health Organization. (2021). Breast cancer. World Health Organization. Retrieved May 7, 2022, from https://www.who.int/news-room/fact-sheets/detail/breast-cancer