STOR 565 Final Report

2023-12-06

KENNY ZHANG

ZIYING SONG

MATHEW HUGHES

FEIHONG HE

Summary

In our project, we trained models using logistic regression, KNN classification, LDA, QDA, SVM, Random Forest, XGBoost, and neural networks, and used the testing set to find the best model for classifying two types of breast cancer. After fitting the data with different classification techniques, we used ROC curves to determine performance and decide which method is the best for predicting the type of breast cancer a patient has.

Based on the visualization of MDS and PCA, and the comparatively worse performance of LDA and SVM with polynomial kernel, the distribution of the two kinds of cancers seems linearly separable but not completely independent from each other.

For model evaluation, three metrics are considered: AUC, accuracy, and sensitivity. The AUC measures the entire two-dimensional area underneath the entire ROC curve from (0,0) to (1,1). It provides an aggregate measure of performance across all possible classification thresholds, allowing us to evaluate different models on the same page. Sensitivity is particularly valued in breast cancer classification because the cost of a false negative rate is unaffordable; it could potentially affect future treatment and possibly harm the patient's health in the long term. Nearly all the best-performing models of each method perform very well on the testing set, with overall accuracy above 95%, sensitivity above 91%, and AUC above 95%. This indicates a strong relationship between the condition of the breast mass and the two types of breast cancer and shows that the FNA technique can provide critical information for breast cancer diagnosis.

Our best model is KNN using MDS with Canberra distance, which achieves 99.30% accuracy and 98% sensitivity. This is sufficient to be a strong reference for doctors to identify these two kinds of breast cancer in practice.

Our project also identifies the most important factors by applying Bagging, Random Forest, and XGBoost. **Area_worst** is the top influencer in all three methods, leading to the largest mean Gini decrease in Bagging and Random Forest, indicating it should be a key factor when distinguishing between the two types of cancer. Identifying the second most important factor is challenging since different models yield varying results.

Contribution

Kenny Zhang: EDA, MDS, PCA, LDA, QDA, Random Forest, Neural Network

Ziying Song: Logistic Regression, Bagging, and XGBoost

Mathew Hughes: Data set description, KNN, and Boosting

Feihong He: L1 logistic Regulation, SVM and Summary

AI Disclaimer: No AI has been used for the creation of this project.

Introduction

Problem of Interest

Breast cancer, the most common cancer among women worldwide, is a major global health concern. The impact of breast cancer extends beyond its prevalence, affecting individuals physically, emotionally, and socially. Early detection and accurate diagnosis can play a critical role in improving survival rates and treatment efficacy of the patients. It also relieves unnecessary anxiety of the individuals who have benign tumors. Beyond individual health, accurate breast cancer classification contributes to public health efforts by facilitating targeted healthcare strategies. Thus, the distinction between malignant and benign breast cancer is pivotal in optimizing patient care, deciding treatment plans, and positively impacting both individual and public health outcomes.

Data Description

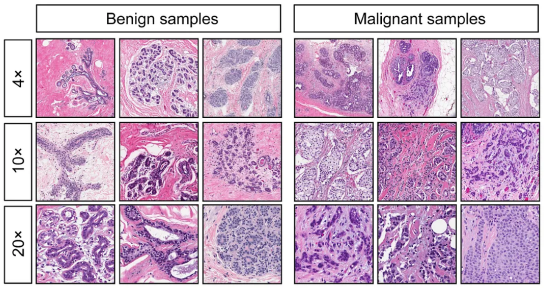


Figure 1: Images of cell nuclei

The data set consists of features describing 569 digitized images of fine needle aspirates (FNA) of breast masses by the University of Wisconsin. These features were captured and computed across three dimensions to more accurately characterize the mass. These features include:

ID	ID Number
Diagnosis	Captured from benign or malignant tumor
Radius	Mean of distances from center to points on the perimeter
Texture	Standard deviation of gray-scale values
Perimeter	Perimeter of the mass
Area	Area of the mass
Smoothness	Local variation in radius lengths
Compactness	Perimeter^2 / (area - 1.0)
Concavity	Severity of concave portions of the contour
Concave Points	Number of concave portions of the contour
Symmetry	Symmetry of the mass
Fractal Dimension	Value describing the variance in the mass over small distances

Data Exploratory Analysis

There are 33 variables with 569 observations in the original data set. The last variable only contains NAs so we will remove that variable. After that, we will convert our binary variable of interest *diagnosis* into 0 and 1 where 0 is the benign class and 1 is the malignant class.

The variable *id* contains information to identify each individual patient, and there is no duplicates meaning each observation in the data is for a different patient. Since *id* is not useful for classifying the breast cancer type, we will remove it from the data set.

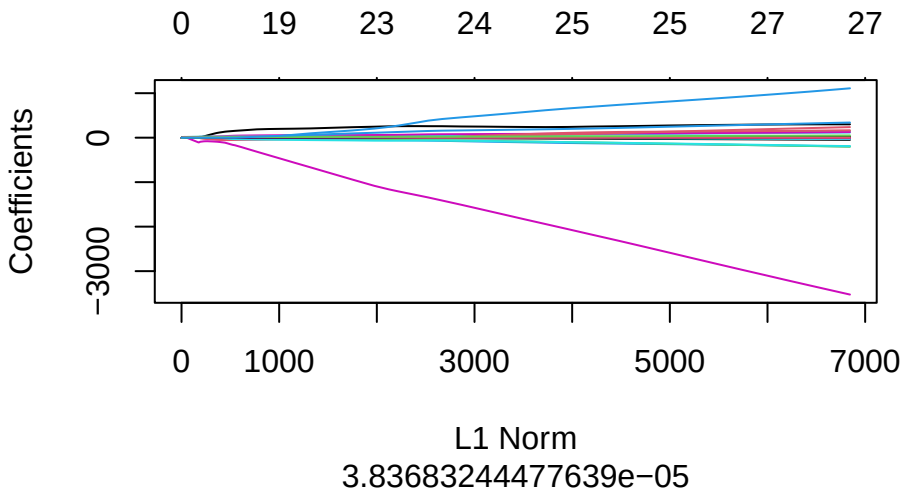
Hence, we are left with 30 covariate variables and 1 response variable.

L1 Logistic

Incorporating L1 regularization into logistic regression helps in feature selection. This is because L1 regularization tends to shrink some coefficients to exactly zero, effectively removing those features from the model. This can be particularly useful when dealing with high-dimensional data where feature selection is crucial.

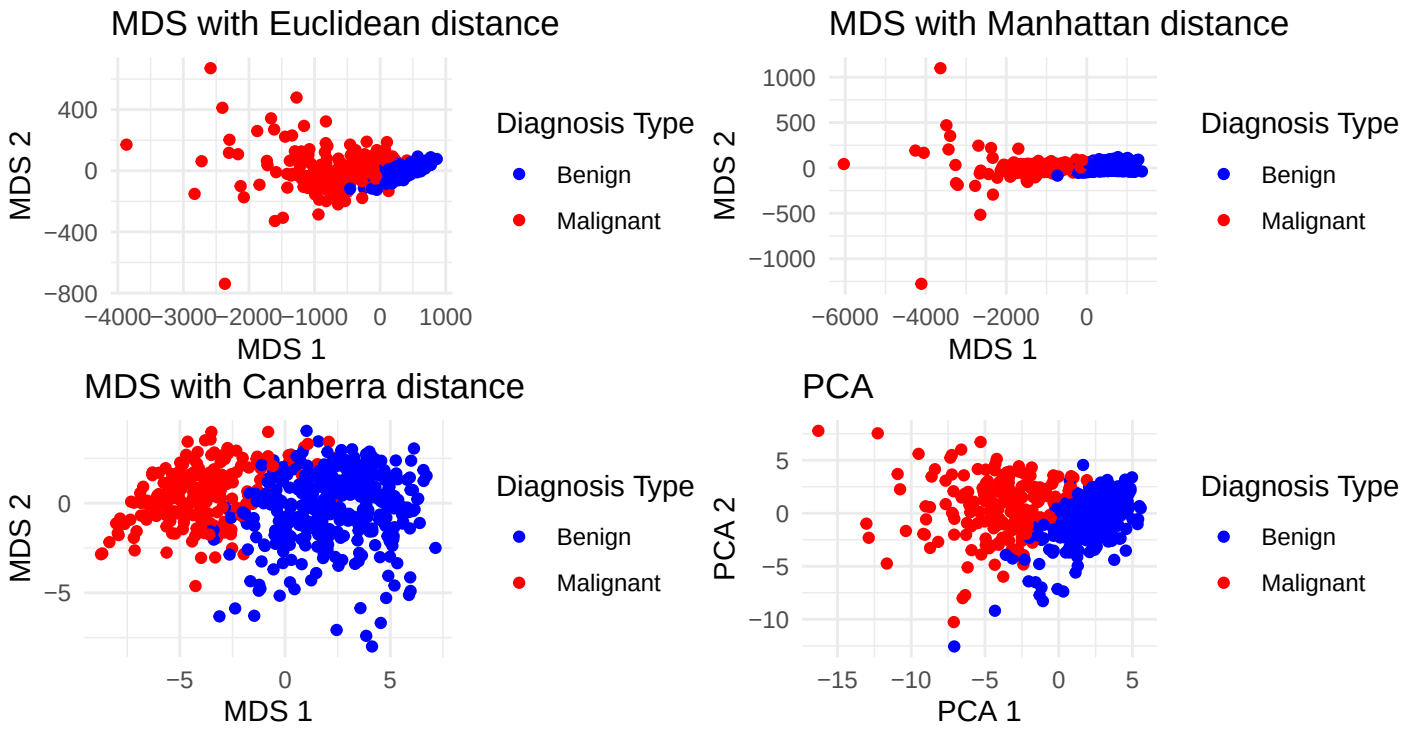
We use cross-validation to tune the optimal penalty parameter to find the logistic regression resulting in minimized loss. In our best model, three variables are shrunk to zero: area_mean, perimeter_mean, and concave_points_worst. However, when we utilize the new dataset without these three variables, it does not boost the result in either accuracy or sensitivity. As a result, we no longer consider them in later Exploratory Data Analysis.

```
fit <- glmnet(X_l1, y_l1, family = "binomial", alpha = 1)
#coef(fit, s = min(fit$lambda))
plot(fit,s=min(fit$lambda))
```



Multidimensional scaling

Although our data's dimensionality is low, meaning there are much more observations (n) than variables (p), we still want to see if dimension reduction methods like Multidimensional scaling (MDS) and Principal Component Analysis (PCA) can increase the performance of our models.



We decided to run MDS with three different metrics, *Euclidean Distance*, *Manhattan Distance*, and *Canberra Distance*. We can see clearly from the plots of MDS with different metrics above, *Euclidean Distance* and *Manhattan Distance* result in many observations from the two classes overlapping, but *Canberra Distance* result in a decent separation for the two classes despite some overlaps still.

Principal Component Analysis

Followed by MDS, we also decide to use PCA to reduce the dimensionality and obtain the best low-dimensional representation of the original data.

As evident from the lower-right plot labled "PCA", the two classes are also separated well with only the first two PCs, and they explain 0.6324321 of the total variability of the original data.

Hence, we will run all of our classification methods with three different data set: original data, data with MDS transformation, and data with PCA transformation.

Classification Methods

For all of our methods, we split the all three data sets into training set and test set. 75% of the data are randomly selected for the training set, and 25% are chosen for the test set. We will evaluate all our methods based on the test set.

Logistic Regression

The **Logistic Regression** is a widely-used binary classification method that predicts the probability of an observation belonging to one of two classes. The model's output is expressed in terms of log-odds, representing the logarithm of the odds ratio between the probabilities of the two classes. The log-odds have a linear relationship with the predictors, and the model is trained to estimate coefficients for each predictor.

The Logistic Regression is especially valuable when the dependent variable is categorical and binary, such as 'malignant' or 'benign' in our prediction.

Logistic Regression establishes a decision boundary in the feature space, separating instances into different classes based on a chosen threshold. Here we first use cross validation to train and fit the Logistic Regression to the original data set with a decision boundary value of 0.5.

```
glm.fits <- glm(diagnosis ~ ., data = training, family = binomial)
glm.predictions <- predict(glm.fits, newdata = test,  type="response")
glm.predictions <- as.factor(if_else(glm.predictions>0.5, 1, 0))
```

We then train and fit the Logistic Regression on MDS using Canberra metric data set and the PCA data set.

Based on the presented table, it is evident that logistic regression performs admirably in predicting breast cancer based on our data. The application of MDS with Logistic Regression yields marginally improved results, particularly in terms of AUC and Accuracy. However, it is noteworthy that the sensitivity of MDS using Logistic Regression is slightly lower compared to the other datasets.

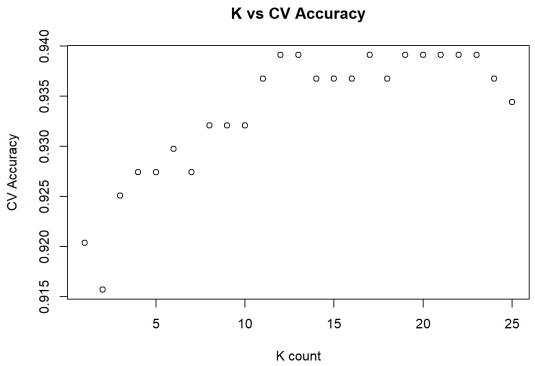
In our context, a higher sensitivity value is desired for predictions. This preference is rooted in the fact that a higher sensitivity translates to a lower false negative rate—an outcome where we aim to minimize the risk of misclassifying patients with malignant breast cancer as negative. Achieving a balance between overall predictive accuracy and the sensitivity of the model remains crucial in ensuring effective and reliable predictions.

	Original	MDS_Canberra	PCA
Area under ROC curve	0.9396552	0.9509442	0.9396552
Accuracy	0.9507042	0.9577465	0.9507042
Sensitivity	1.0000000	0.9880952	1.0000000

K-Nearest Neighbors

K-Nearest Neighbors is a classification technique known best for its simplicity and interpretability. For this type of model, new data points are compared to the previous training set and the k-nearest neighbors are found using some distance metric such as Euclidean or Canberra. In regression models, the predicted value of each point is averaged to the find the predicted value of the new point while in classification models, the new prediction is decided by majority vote.

For our model, the value of K was chosen by preforming cross-validation on the training set and choosing the K that gave the best accuracy. As demonstrated in the below graph, the optimal K for this model was 12.



Once the K had been chosen, KNN was preformed on the original test set, the test set passed through MDS with Canberra distance and the the first two dimensions of PCA. Interestingly, KNN preformed on the original data set had the lowest accuracy and sensitivity of all the models however, KNN preformed on the MDS set with Canberra distance had the best accuracy and sensitivity. What this likely means is that non-relevant information in the original training set resulted in over-fitting while the dimension reduction of MDS resulted in more clumped data with similar diagnosis leading to a more robust KNN model. PCA also preformed much better than the original dataset enforcing the idea that dimension reduction was responsible for the dramatic increases in accuracy and sensitivity.

	Original	MDS.Canberra.	PCA
Area under ROC curve	0.8760	0.9914	0.9364
Accuracy	0.8944	0.9930	0.9437
Sensitivity	0.7759	0.9828	0.8966

Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a classification method for finding a linear boundary between the two classes. However, there are a few assumptions that needs to be met in order to achieve good prediction performance.

The first assumption is that it assumes the two classes have Gaussian density, and the second assumption is that all classes share a common variance-covariance matrix. The two assumptions are clearly not met for our data set. However, we will still implement LDA as a way to compare the performance between different methods.

```
cancer.lda <- lda(diagnosis~.,data = training)
lda.pred <- predict(cancer.lda,test)$class
table_results <- table(lda.pred, test$diagnosis)
```

	Original	MDS_Canberra	PCA
Area under ROC curve	0.9569000	0.9682000	0.9052000
Accuracy	0.9647887	0.9718310	0.9225352
Sensitivity	0.9137931	0.9482759	0.8103448

For LDA with the original data, the prediction performance is good considering the assumptions are not met for LDA.

The performance of LDA with MDS using Canberra Distance is slightly better than with original data, where accuracy, sensitivity, and AUC all have better values.

LDA for data with PCA transformation yields the worst performance out of the three data sets with the lowest values for accuracy, sensitivity, and AUC.

Quadratic Discriminant Analysis

Quadratic Discriminant Analysis (QDA) tends to find the non-linear (quadratic) boundary between the two classes for this data set. The assumption is relaxed in QDA comparing to LDA where we no longer assumes all classes share a common variance-covariance matrix.

```
cancer.qda <- qda(diagnosis~.,data = training)
qda.pred <- predict(cancer.qda,test)$class
table_results <- table(qda.pred, test$diagnosis)
```

	Original	MDS.Canberra.	PCA
Area under ROC curve	0.9596000	0.9596000	0.9278000
Accuracy	0.9647887	0.9647887	0.9366197
Sensitivity	0.9310345	0.9310345	0.8793103

We can see that even though the accuracy stayed the same for QDA with the original data comparing to LDA, the sensitivity and AUC actually increased, meaning it has a better prediction performance than LDA as expected.

The prediction performance using QDA with MDS actually decreased comparing to LDA where all three evaluation metric has lower values. This suggests that the true boundary between the two classes with MDS transformation is closer to linear rather than quadratic.

The prediction performance increased with QDA using PCA transformation comparing to LDA, which suggests that the true boundary with PCA transformation is closer to quadratic rather than linear.

Support Vector Machine

For SVM, we also trained our model on the original, MDS, and PCA datasets respectively. We achieve the best result with the model trained on the original dataset, attaining 97.88% accuracy, 94.82% sensitivity, and 94.23% AUC. Models trained on other datasets achieve slightly lower but comparable results. However, when we train the SVM with a polynomial kernel, all results drop significantly, which implies to some degree that the distribution of these two kinds of cancers is linearly separable in the high-dimensional space.

	original_data	MDS	PCA	polynomial_kernel
Area under AUC Curve	0.9741379	0.9449918	0.9337028	0.8448276
Accuracy	0.9788732	0.9507042	0.9436620	0.8732394
Sensitivity	0.9482759	0.9137931	0.8793103	0.6896552

Bagging

Bagging, short for Bootstrap Aggregation, helps make classification trees more reliable by combining their predictions through majority voting. Because it lowers both bias and variance, making the model more robust, Bagging can be a solid choice for improving our predictions.

```
bag <- randomForest(diagnosis~., mtry = ncol(training)-1, data = trainBag)
varImpPlot(bag, n.var = 10)
```

Above is the variable importance plot generated by the original data set. We can see that 'area worst' is being the most important variable with the Bagging method in our original data set, and it is followed by 'concave.points_worst'. Thus, they play relatively crucial role in model's decision makeing process.

The presented table highlights that employing the bagging method on our PCA dataset yields superior performance, characterized by larger values in both AUC and sensitivity metrics. Despite bagging demonstrating commendable overall performance with consistently high accuracies, it falls slightly short of the satisfaction derived from other methods explored earlier.

This observation hints at the possible existence of strong predictors within the dataset, potentially resulting in highly correlated trees. In such scenarios, the application of Random Forest presents itself as a promising alternative, while it can offer improved performance by decorrelating the trees, thereby reducing variance.

	Original	MDS.Canberra.	PCA
Area under ROC curve	0.9423235	0.9304187	0.9536125
Accuracy	0.9507042	0.9366197	0.9577465
Sensitivity	0.8965517	0.8965517	0.9310345

Random Forest

Random forest is a special case of Bagging. Bagging may not given significance reduction in variance when there is a few strong predictor in the data set, along with a number of moderately strong ones.

In random forest, only m predictors which are chosen randomly are available for the tree from each bootstrap sample. Not only does this help with variance reduction, it is also less computational expensive. When $m = p$, i.e. all predictors are available, it becomes the usual bagging again.

For our data sets, we choose m relative small to p where $m = \sqrt{p}$.

```
set.seed(565)
p <- ncol(training)-1
cancer.randomF<- randomForest(as.factor(diagnosis) ~ ., data=training,mtry = sqrt(p))
rf.pred <- predict(cancer.randomF, newdata=test)
table_results <- table(rf.pred, test$diagnosis)
```

Comparing to Bagging with the original data set, random forest has the same accuracy, but sensitivity and AUC has increased, suggesting that there might be some predictors stronger than others.

	Original	MDS.Canberra.	PCA
Area under ROC curve	0.9536000	0.9304000	0.9536000
Accuracy	0.9577465	0.9366197	0.9577465
Sensitivity	0.9310345	0.8965517	0.8965517

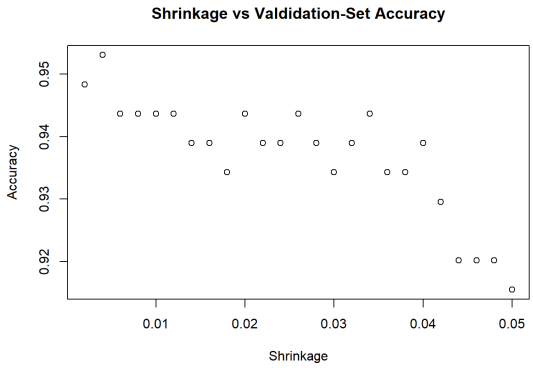
The prediction performance of random forest with MDS transformation is exactly the same as bagging because the data only have two predictors. Therefore, the results from bagging and random forest will be extremely similar, in this case completely the same.

The prediction accuracy for random forest with PCA transformation is the same as bagging, but the sensitivity and AUC is lower than bagging. PCA has the same property as MDS in this case there they all have two predictors. Since bagging has a better performance, it suggests that the two predictors of PCA are equally strong in terms of classification.

Boosting

Like Bagging and Random Forests, Boosting is a decision tree based classification method which allows a decision to move along a path and take different routes according to its values, similar to how an ant might walk along a tree branch and choose which path to take as it encounters splits. Unlike Bagging and Random Forests however, Boosting does not train each tree using independent bootstrap samples but rather trains an initial tree on the entire training data set and creates sequential trees off of the branches of the preceding tree. What this means is that rather than fitting each tree on a data set, each tree is fit on the residuals of the preceding tree. Resulting in more complex decisions that gradually improves on the weaknesses of its predecessors rather than attempting to find a solution all at once.

The speed at which each new tree grows is modified by a value known as the shrinkage parameter where lower values result in more complex models while higher values result in less complex but more interpretable models. The increased complexity is good for fitting non-linear solutions but can often lead to over-fitting on the training set. To solve this we used a validation set and tested models with various shrinkage parameters trained by the training set on the new validation set. A shrinkage parameter of .004 preformed the best as shown in the table plot below.

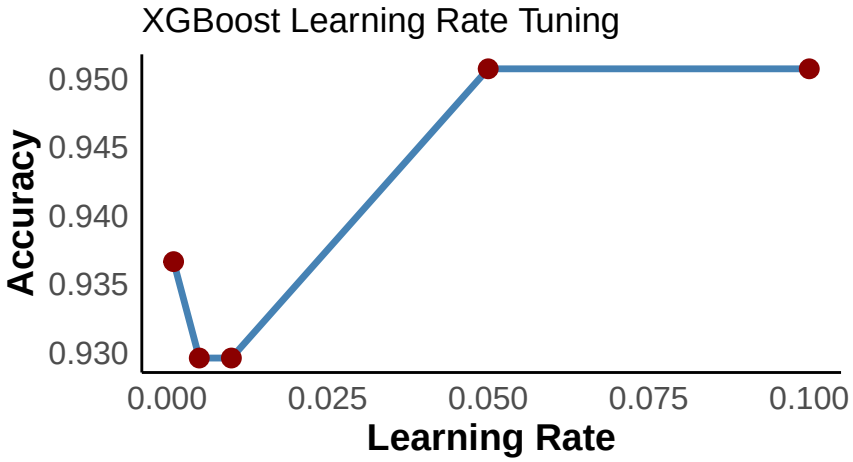


The model preformed fairly well on the test set with an accuracy of 96.48% on the original data set, the MDS data set, and the PCA set. The similarity in accuracy between the sets implies that the model was effective at finding the ideal boundary but was unable to optimally represent it due to the nature of decision trees struggling to perfectly represent boundaries that are not perfectly vertical or horizontal.

	Original	MDS.Canberra.	PCA
Area under ROC curve	0.9569	0.9569	0.9569
Accuracy	0.9648	0.9648	0.9648
Sensitivity	0.9138	0.9138	0.9138

Extreme Gradient Boosting

XGBoost, or eXtreme Gradient Boosting, is a machine learning algorithm renowned for its exceptional predictive capabilities. Similar to Boosting, XGBoost decision trees are created in sequential form, which allows the later trees to correct the mistakes made by earlier trees. Both have final predictions that come from the weighted sum of the individual model predictions, and both methods ensemble individual classifiers / predictors to give a strong and precise model. However, the difference between XGBoost and boosting is that XGBoost introduces regularization terms and it prune the individual trees by using parameters like 'max_depth' and gamma.



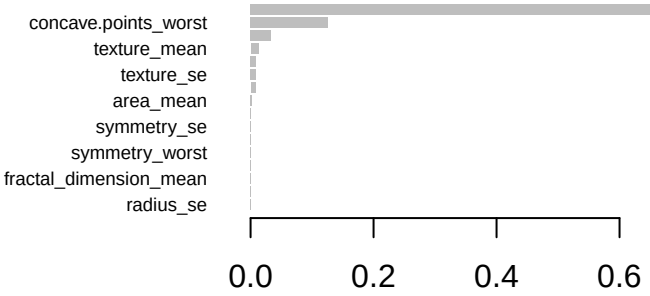
We firstly applied cross validation on a range of learning rates ('eta'), and found out the learning rate value that gives the best accuracy is 0.05 for the original data set. Since max_depth and n_estimators parameters are key hyperparameters in XGBoost that influence the complexity and size of the ensemble, we tried to set them on different values, but the CV result didn't get changed much.

```
xgb_model <- xgboost(
  data = as.matrix(xgb_training[, -1]),
  label = as.numeric(xgb_training$diagnosis) - 1,
  nrounds = 10,
  objective = "binary:logistic",
  eval_metric = "logloss",
  verbose = 0,
  eta = 0.05,
  max_depth = 5,

)
predictions_xgb <- predict(xgb_model, as.matrix(test[, -1]))
predictions_xgb_class <- ifelse(predictions_xgb > 0.5, 1, 0)
```

From the table below, we can conclude that XGBoost with PCA out performs XGBoost with original dataset and MDS with a significant higher AUC, accuracy and sensitivity. This happens might indicate that the original dataset for using XGBoost has a high dimensionality with potentially redundant features or high multicollinearity.

	Original	MDS.Canberra.	PCA
Area under ROC curve	0.9423235	0.9158456	0.9622332
Accuracy	0.9507042	0.9225352	0.9647887
Sensitivity	0.9880952	0.9523810	0.9761905



The variable importance plot of the original data set indicates that the most important value here is 'area_worst', which matches to the most important variable in bagging. It is also used for the first split of the first tree in the model trained by the original data set.

Neural Network

Neural Network is a method inspired by the structure and function of the human brain. It composed of layers of interconnected nodes, called neurons, that process and transmit information. It is used for image and speech recognition, NLP, and autonomous vehicles. Since it is well suite for pattern recognition and classification, we will use it to classify different breast cancer type.

Each layer is connected with lines called channels, each channel is assigned with a weight value determine how important the input from the previous node is. The neural network is able to train itself by identifying what it did wrong in the prediction, and work backward to adjust the weights of each channel to achieve better prediction performance.

Since neural network is very sensitive to predictors' scales, where larger values might have a more significant impact on the model causing the prediction performance to decrease. Therefore, we will use min-max scale to scale the data first where each value with be within 0 and 1. The intuition behind the min-max scale is as following:

$$X_{scaled} = \frac{X - min(X)}{max(X) - min(X)}$$

It is important to select the proper value for number of hidden layers and their corresponding neurons. For a low dimensional data like this, one layer is usually enough. However, after testing different value of neurons $\{N : 0 < N < p\}$, we found out that having a two hidden layer neural network with fewer neurons for each layer has a better prediction performance.

After doing some research, a neural network with 2 layers and fewer nodes not only can learn more about the features of the data set, and more computational efficient, but also is less likely to overfitting. Therefore, we are set on a neural network with 2 layers, where the first layer has 6 neurons, and the second layer has 3 neurons.

```
set.seed(565)
nn <- neuralnet(diagnosis~.,data = training_nn,hidden = c(6,3))
nn_pred <- compute(nn,test_nn)
nn_results <- ifelse(nn_pred$net.result > 0.5, 1,0)
```

The neural network we ended up is shown as below:

	Original	MDS.Canberra.	PCA
Area under ROC curve	0.9828000	0.9622000	0.9741000
Accuracy	0.9859155	0.9647887	0.9788732
Sensitivity	0.9655172	0.9482759	0.9482759

The neural network with the original data set yield very good performance, only second to KNN with MDS transformation. It has high accuracy, sensitivity, and a large AUC.

Neural network with MDS transformation also has good result, but not as good as neural network with the original data set. We believe it is because there is only two predictors for MDS transformation, causing the neural network not being as wide and deep as the previous one.

The performance for PCA transformation is slightly worse than MDS transformation for the same reason as MDS transformation. There is only two predictors, and the neural network doesn't get as wide and as deep as the original data set, causing it not able to learn more features.

Conclusion

In conclusion, our project tackled breast cancer classification, a significant global health issue, particularly prevalent among women. Recognizing the crucial role of early and accurate diagnosis in enhancing treatment efficacy and survival rates, we applied a range of models including logistic regression, KNN, LDA, QDA, SVM, Random Forest, XGBoost, and neural networks. Our evaluations using ROC curves showed most models excelling with over 95% accuracy, sensitivity, and AUC. The standout was the KNN model using MDS with Canberra distance, achieving 99.14% accuracy and 98% sensitivity, marking it as highly effective for clinical use.

We identified area_worst as a key differentiator in cancer types, a finding crucial for personalized treatment and public health strategies. This project enhances breast cancer classification, significantly impacting patient care and public health, emphasizing the need for accurate diagnosis in breast cancer management.