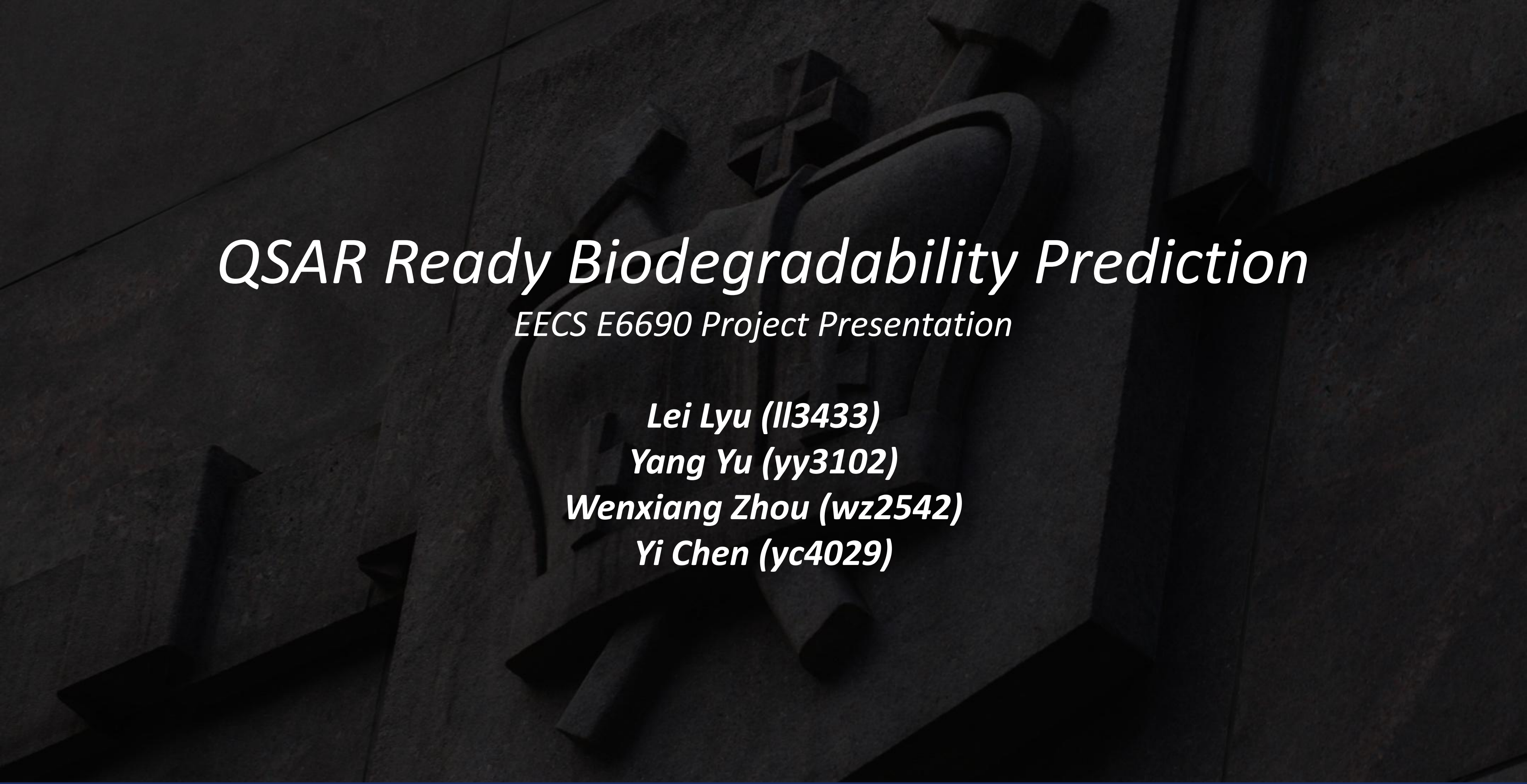


QSAR Ready Biodegradability Prediction

EEСS E6690 Project Presentation

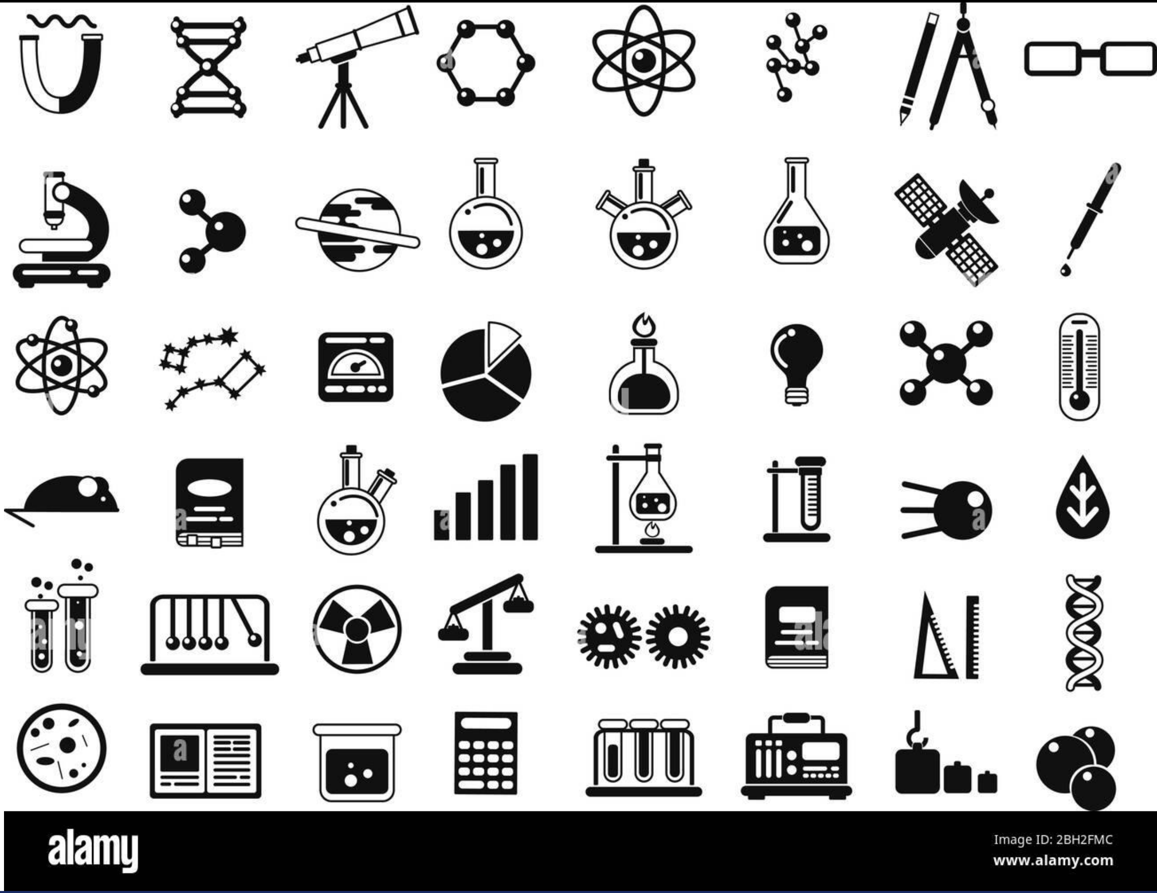
Lei Lyu (ll3433)
Yang Yu (yy3102)
Wenxiang Zhou (wz2542)
Yi Chen (yc4029)

Part 1: Dataset description

QSAR biodegradation Data Set

Dataset Description

- Various kinds of chemicals are left in the environment by industry
- Many of the chemicals are not biodegradable
- Information about biodegradability of chemicals are not yet abundant



Dataset Description

- QSAR (Quantitative Structure-Activity Relationships) is used to predict the biodegradability of chemicals

- QSAR biodegradation data set was built to develop QSAR models for studying the relationship between chemical structure and biodegradability of molecules

- Experimental values collected from webpage of the National Institute of Technology and Evaluation of Japan (NITE)

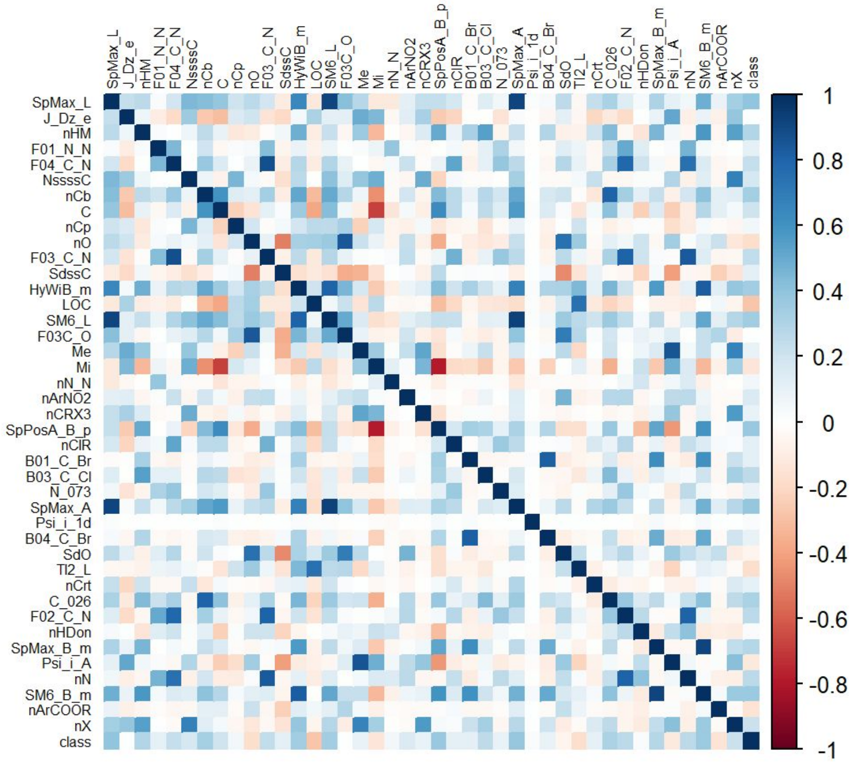
Dataset Description

- Number of Instances: 1055
  where 356 molecules are ready biodegradable (RB) and 699 are not ready biodegradable (NRB)

- Number of Attributes: 41
  selected using many classification modeling methods combined with genetic algorithms

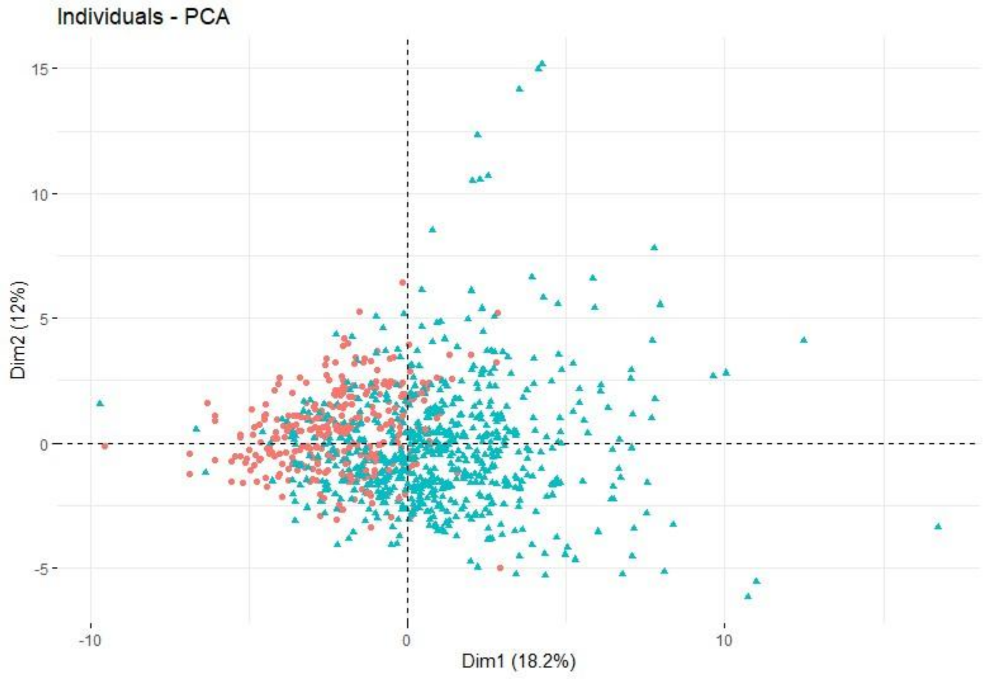
- Correlations between descriptors:

Heatmap of variable correlations

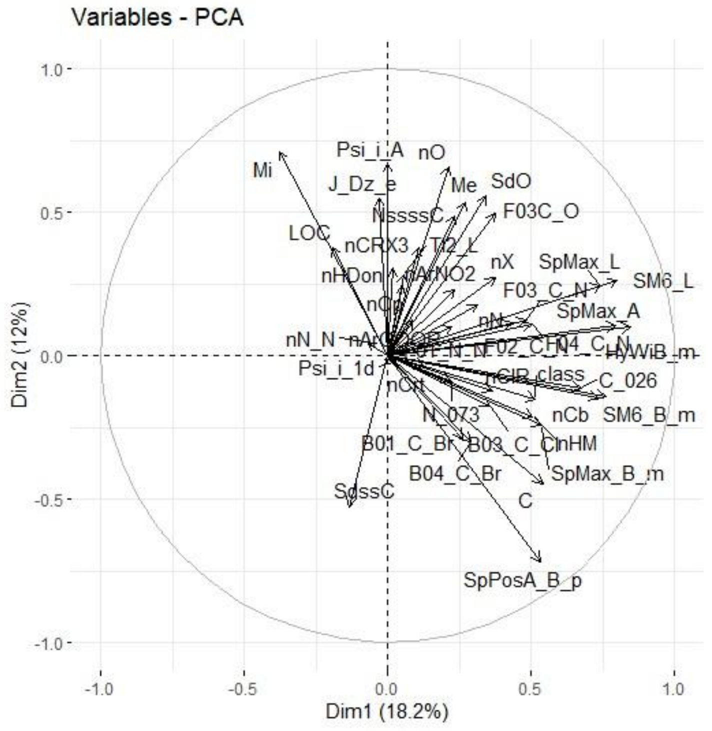


All Descriptors PCA

Score Plot

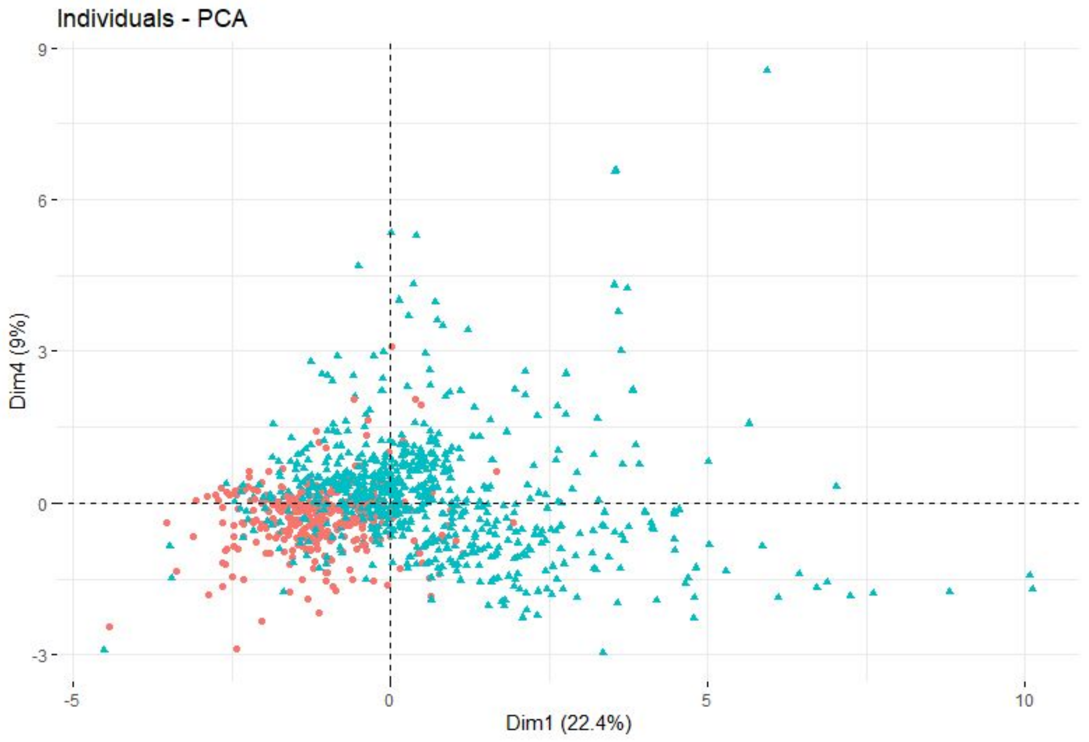


Loading Plot

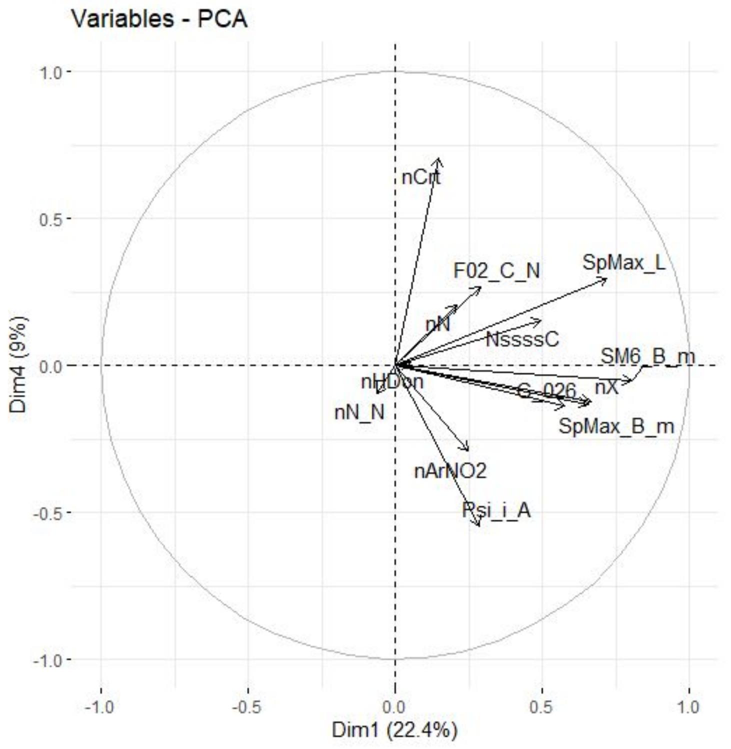


Paper’s SVM Descriptors PCA

Score Plot

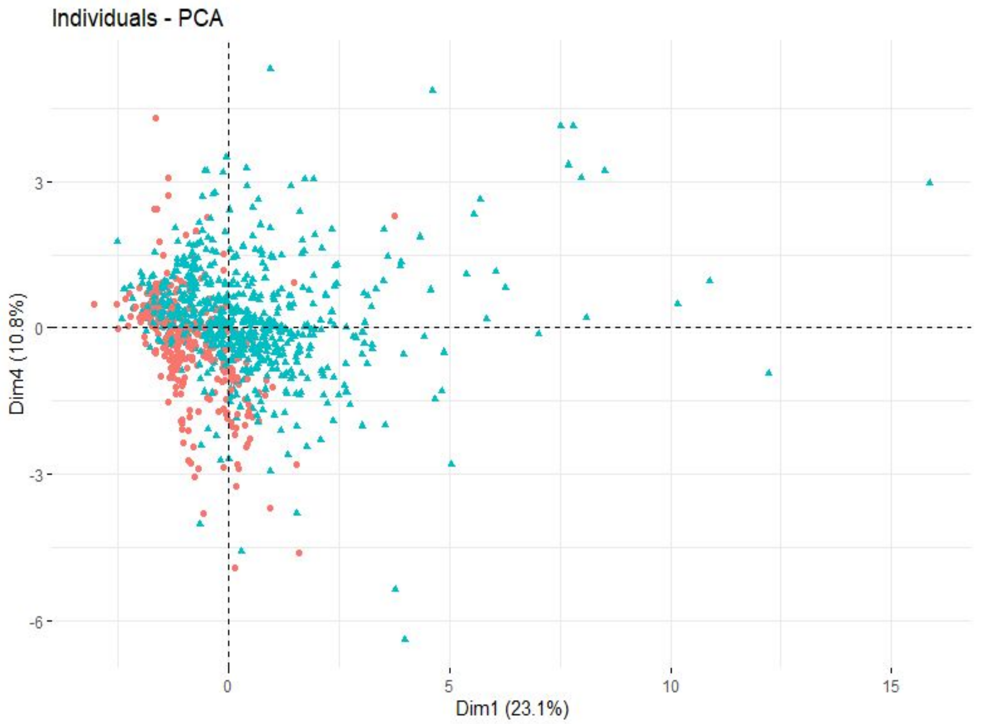


Loading Plot

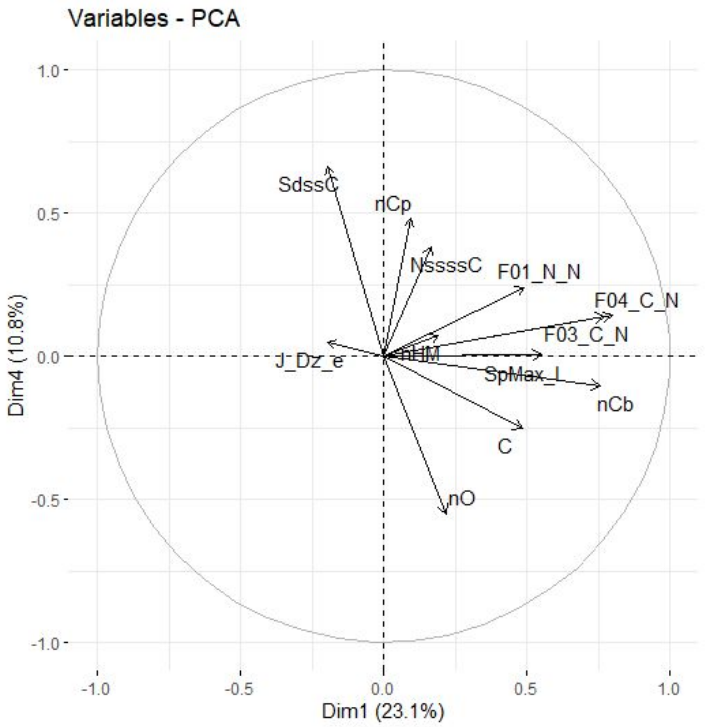


Paper’s kNN Descriptors PCA

Score Plot



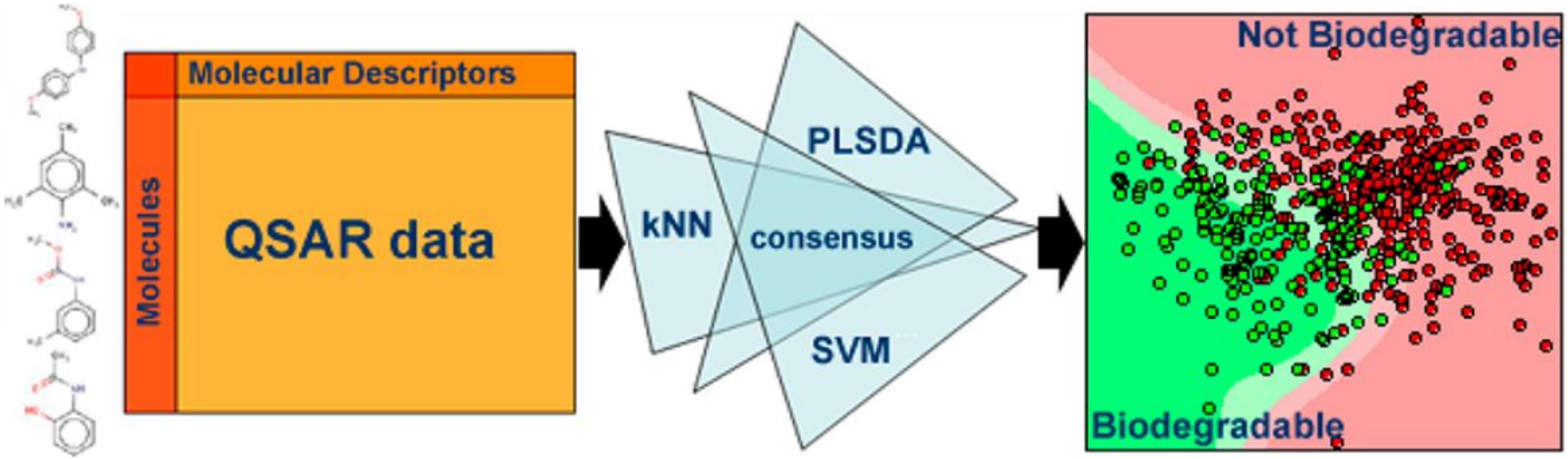
Loading Plot



Part 2: Paper detail and reproduce

KNN, PLSDA, SVM

Quick View



Data set dividing:

41 Attributes, 1055 instances

training set: 837 test set: 218

5-fold cross-validation

2 Class:

RB: ready biodegradable

NRB: not ready biodegradable

Model Validation

- **Specificity:**

the ability to correctly predict RB molecules

$$Sp = \frac{TN}{TN + FP}$$

TN: # true negatives
FP: # false positives

- **Sensitivity:**

the ability to correctly predict NRB molecules

$$Sn = \frac{TP}{TP + FN}$$

TP: # true positives
FN: # false negatives

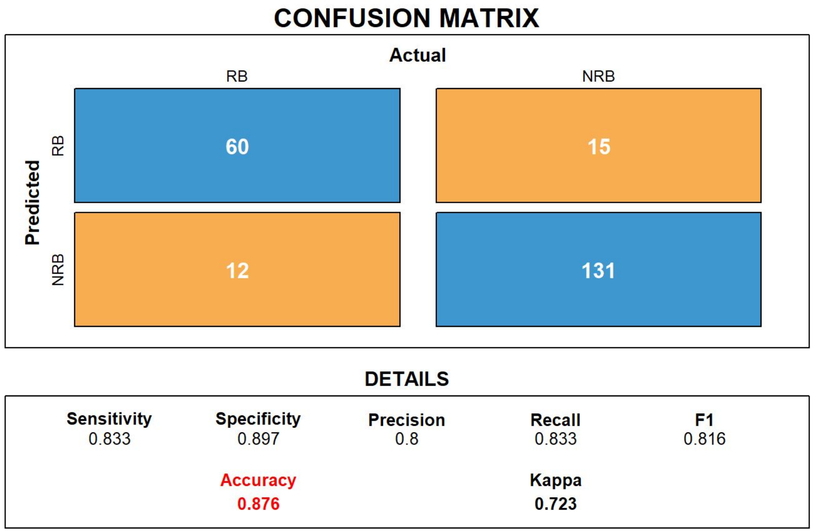
- **ER:**

the classification error rate

$$ER = 1 - \frac{Sp + Sn}{2}$$

1. k Nearest Neighbors (kNN)

Our reproduce result:


CONFUSION MATRIX

Predicted \ Actual	RB	NRB
RB	60	15
NRB	12	131

DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.833	0.897	0.8	0.833	0.816

Accuracy 0.876

Kappa 0.723

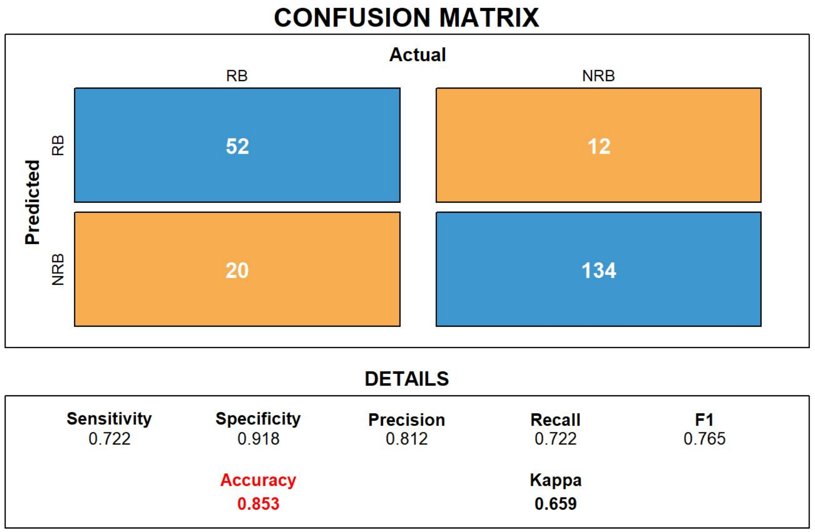
12 descriptors selected

Euclidean distance

Techniques	ER	Sp	Sn
KNN (paper)	0.15	0.90	0.81
KNN (reproduced)	0.12	0.90	0.83

2. Partial Least Squares Discriminant Analysis (PLSDA)

Our reproduce result:


CONFUSION MATRIX

Predicted \ Actual	RB	NRB
RB	52	12
NRB	20	134

DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.722	0.918	0.812	0.722	0.765

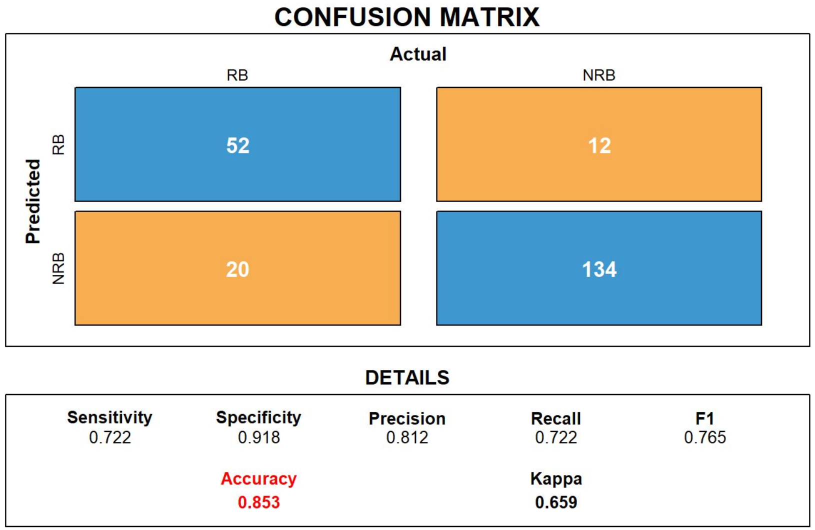
Accuracy	Kappa
0.853	0.659

23 descriptors selected

Techniques	ER	Sp	Sn
PLSDA (paper)	0.15	0.87	0.83
PLSDA (reproduced)	0.15	0.92	0.72

3. support vector machines (SVM)

Our reproduce result:



14 descriptors selected

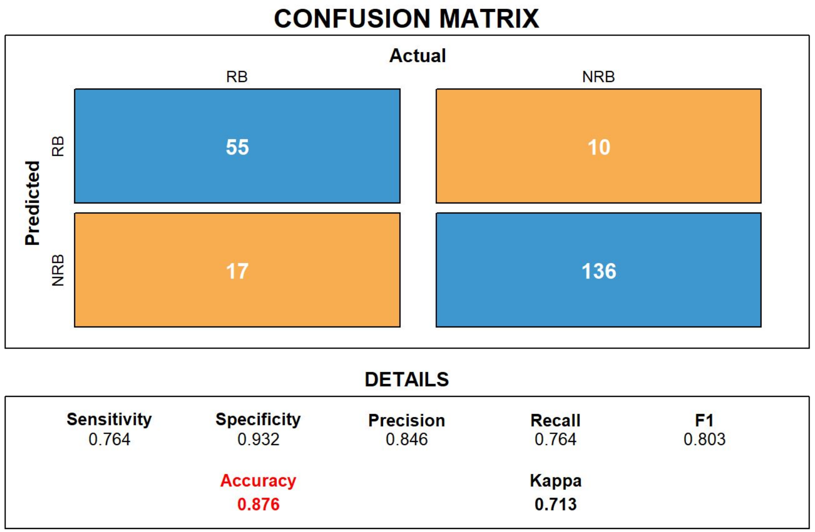
Techniques	ER	Sp	Sn
SVM (paper)	0.14	0.91	0.82
SVM (reproduced)	0.15	0.92	0.72

SVM-improved

Our reproduce result:

all descriptors

5-fold cross-validation ==>7-fold



Sensitivity	Specificity	Precision	Recall	F1
0.764	0.932	0.846	0.764	0.803

Accuracy	Kappa
0.876	0.713

Techniques	ER	Sp	Sn
SVM (paper)	0.14	0.91	0.82
SVM (reproduced)	0.15	0.92	0.72
SVM (improved)	0.12	0.93	0.76

Part 3: Other Techniques Implementation

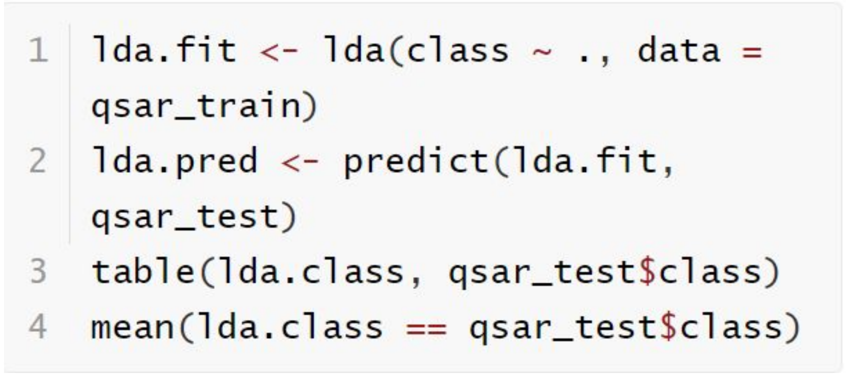
Overview

Methods

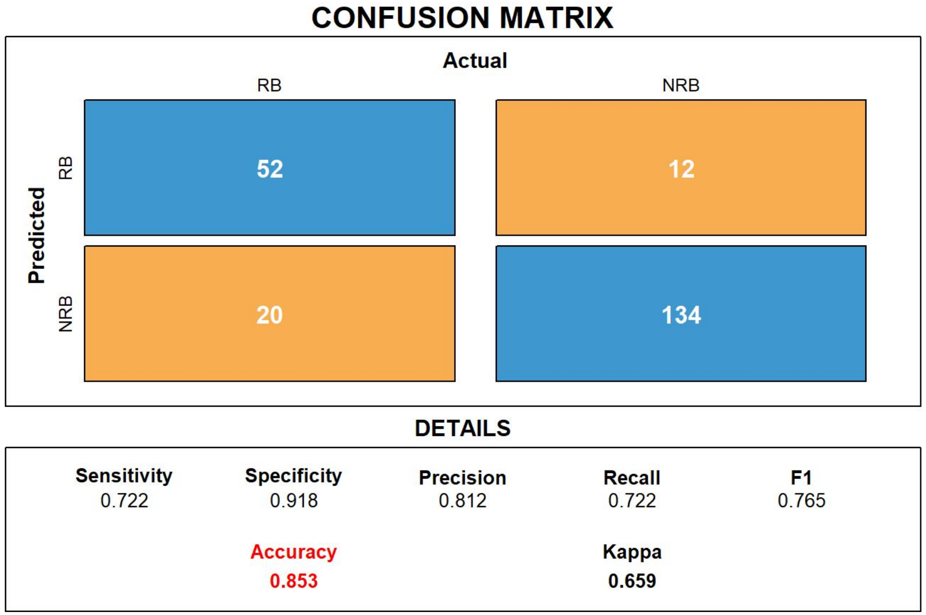
- LDA
- Naive Bayes
- Decision Tree
- Bagging
- Random Forest
- Adaboost
- Neural Network

LDA(Linear Discriminant Analysis)

R Code



```
lda.fit <- lda(class ~ ., data =
qsar_train)
lda.pred <- predict(lda.fit,
qsar_test)
table(lda.class, qsar_test$class)
mean(lda.class == qsar_test$class)
```



CONFUSION MATRIX

Predicted \ Actual	RB	NRB
RB	52	12
NRB	20	134

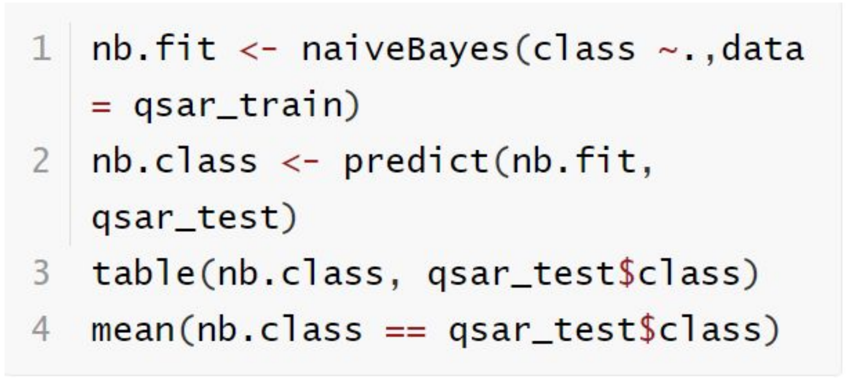
DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.722	0.918	0.812	0.722	0.765

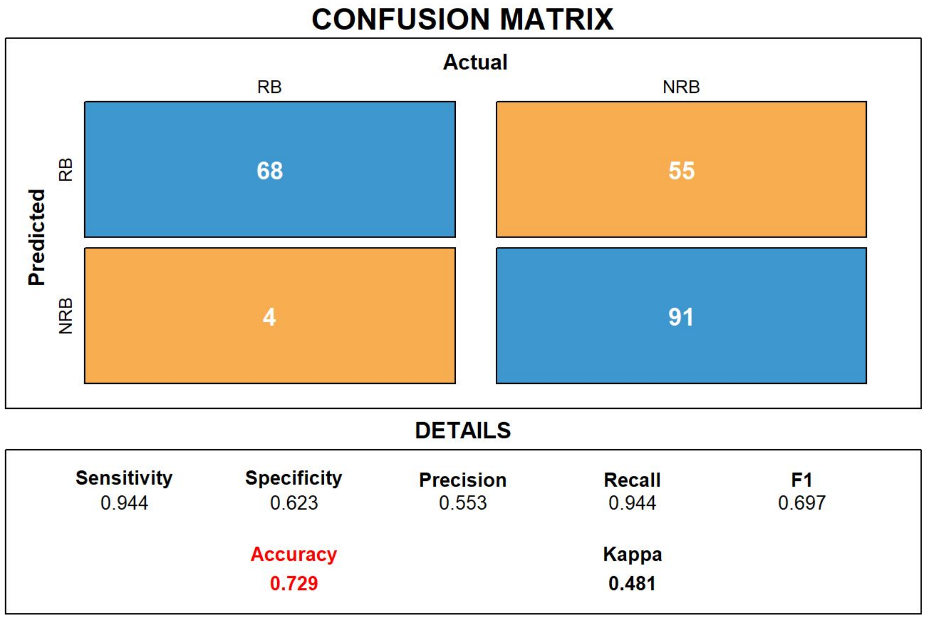
Accuracy	Kappa
0.853	0.659

Naive Bayes

R Code

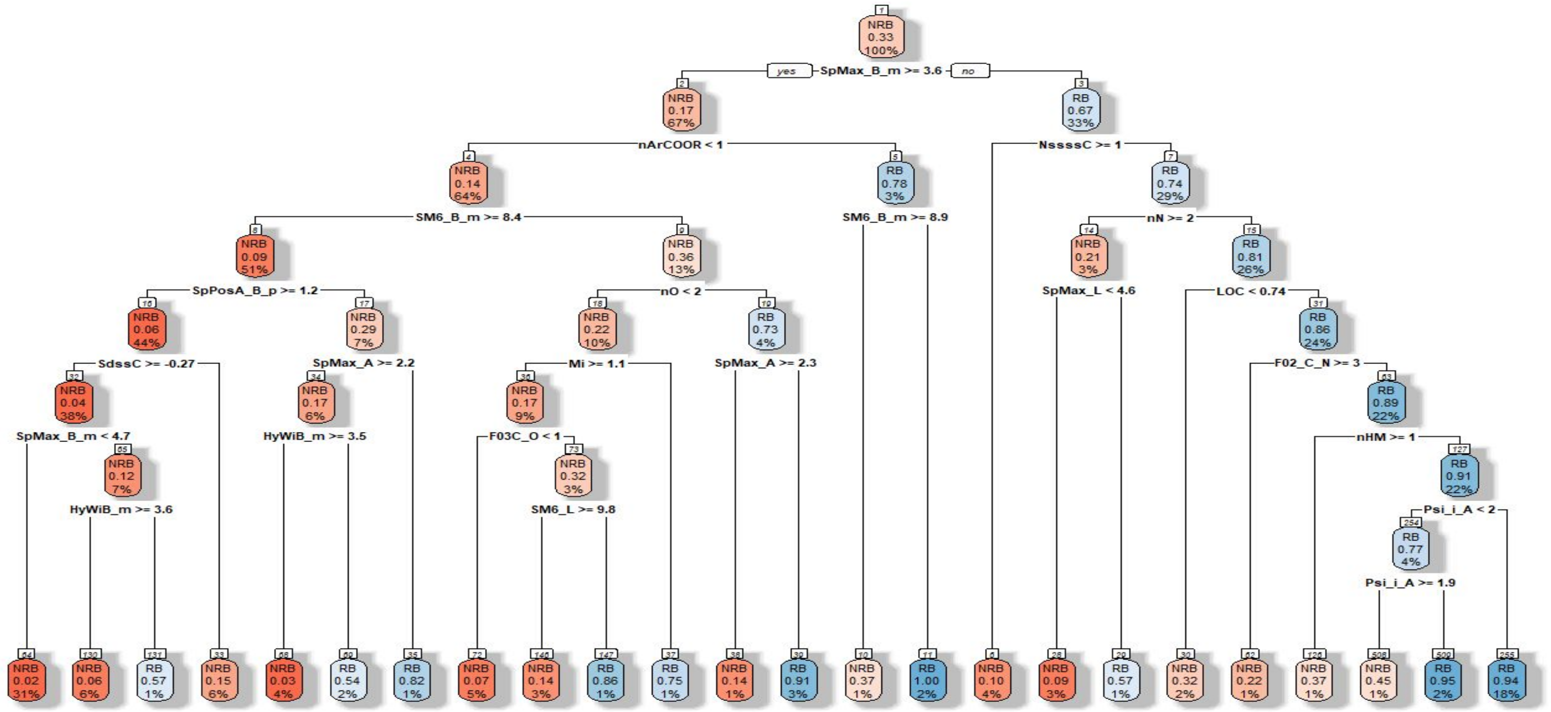


```r
nb.fit <- naiveBayes(class ~.,data
= qsar_train)
nb.class <- predict(nb.fit,
qsar_test)
table(nb.class, qsar_test$class)
mean(nb.class == qsar_test$class)
```

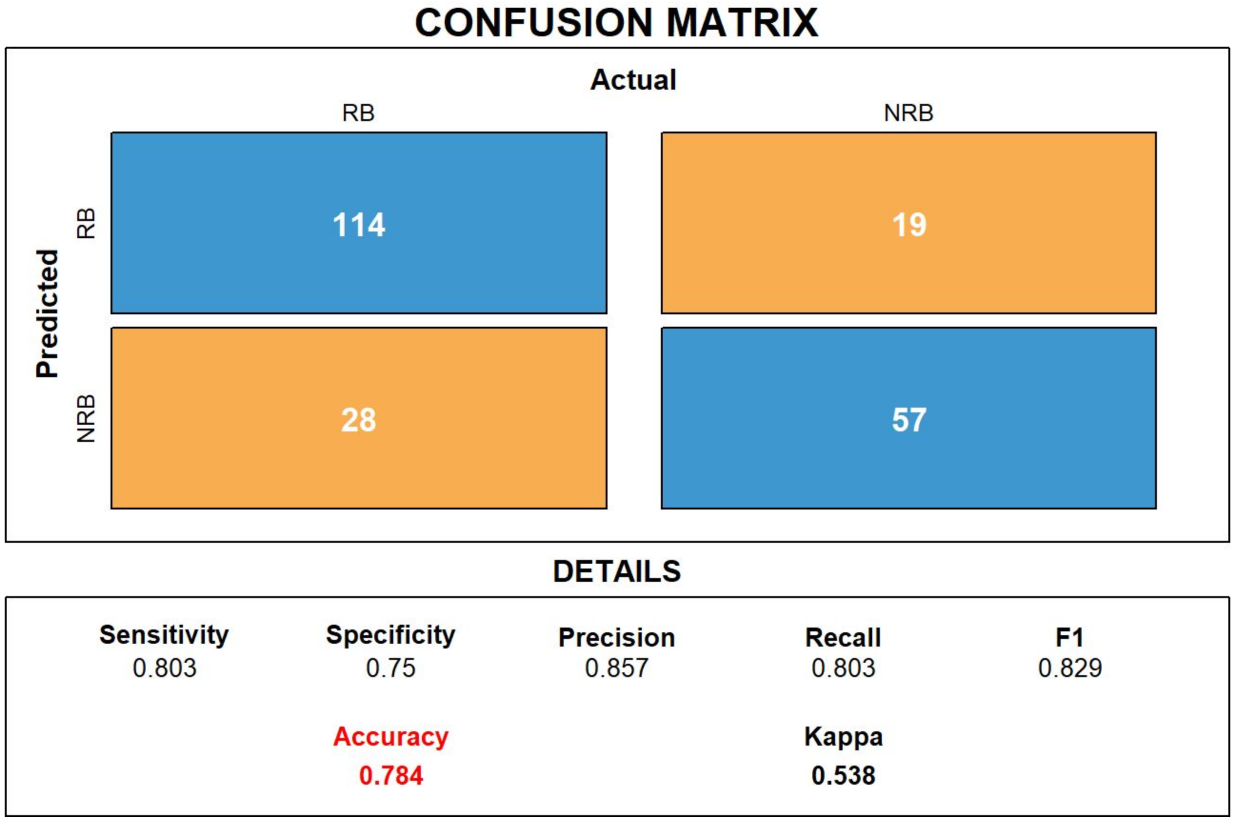


Decision Tree(Without pruning)

```r
q.rpart <- rpart(class~., data=q_train, method = 'class', cp=0)
```



Decision Tree(Without pruning)



CONFUSION MATRIX

Predicted \ Actual	RB	NRB
RB	114	19
NRB	28	57

DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.803	0.75	0.857	0.803	0.829

Accuracy	Kappa
0.784	0.538

Decision Tree(Pruning)

- For each α:

$$\min_{T \subseteq T_0} \left\{ \sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T| \right\},$$

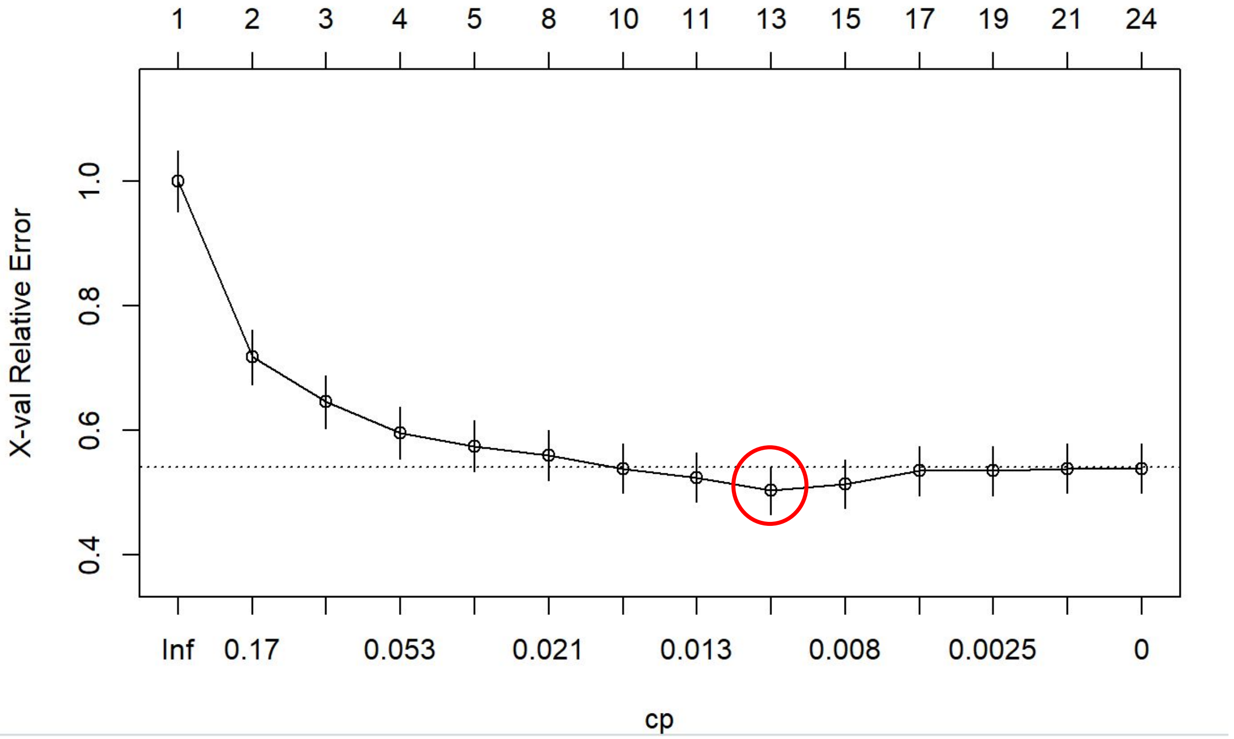
CP(Complexity Parameter) in rpart

$$\sum_{Terminal\ Nodes} Misclass_i + \lambda * (Splits)$$

Decision Tree(Pruning)

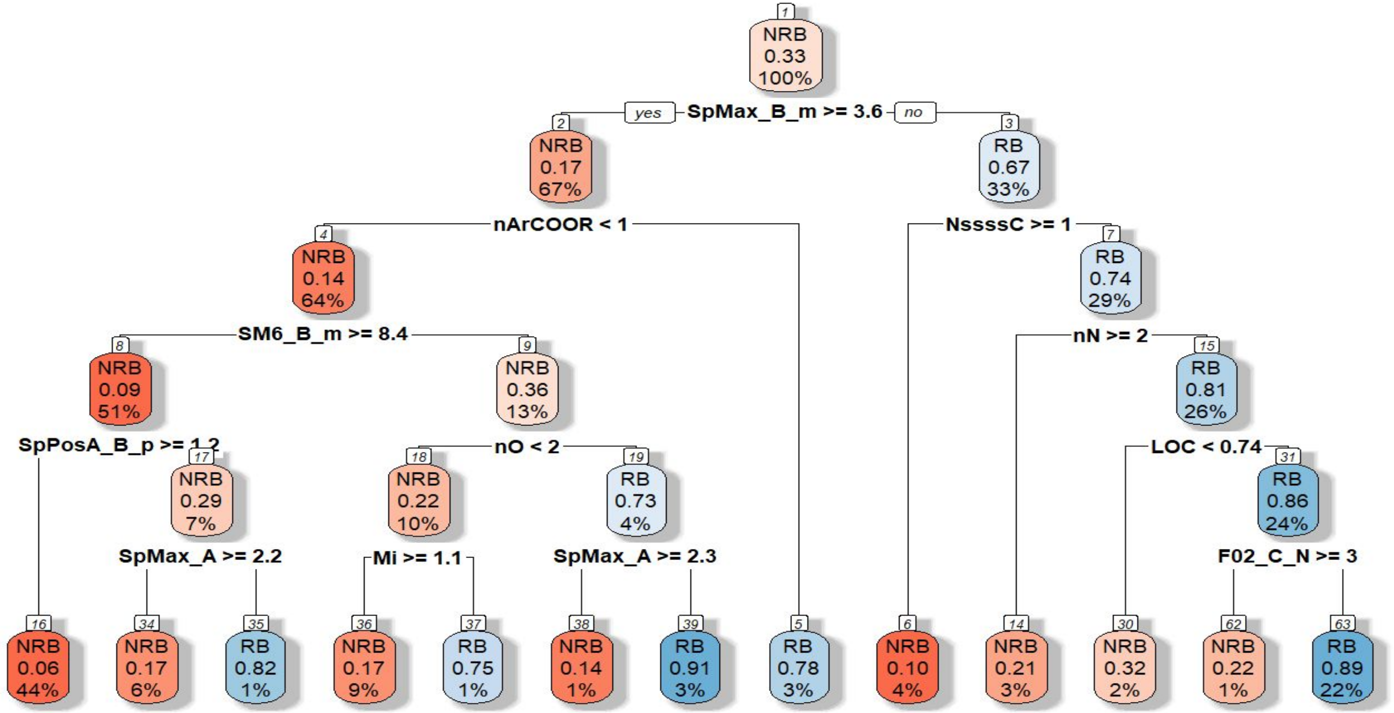
R Code

```
printcp(q.rpart)
plotcp(q.rpart)
```

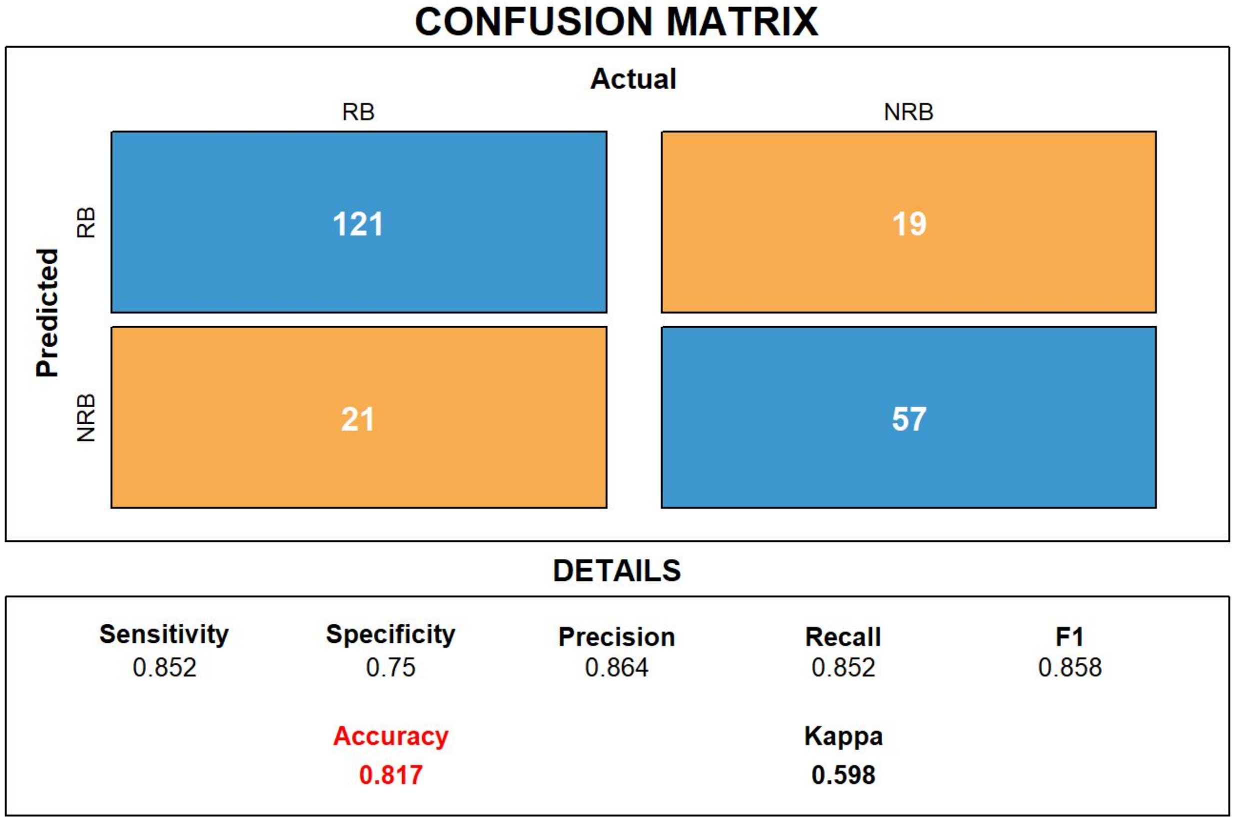


Decision Tree(Pruning)

```r
q.prune <- rpart(class~., data=q_train, method = 'class', cp=0.0089286)
```



Decision Tree(Pruning)



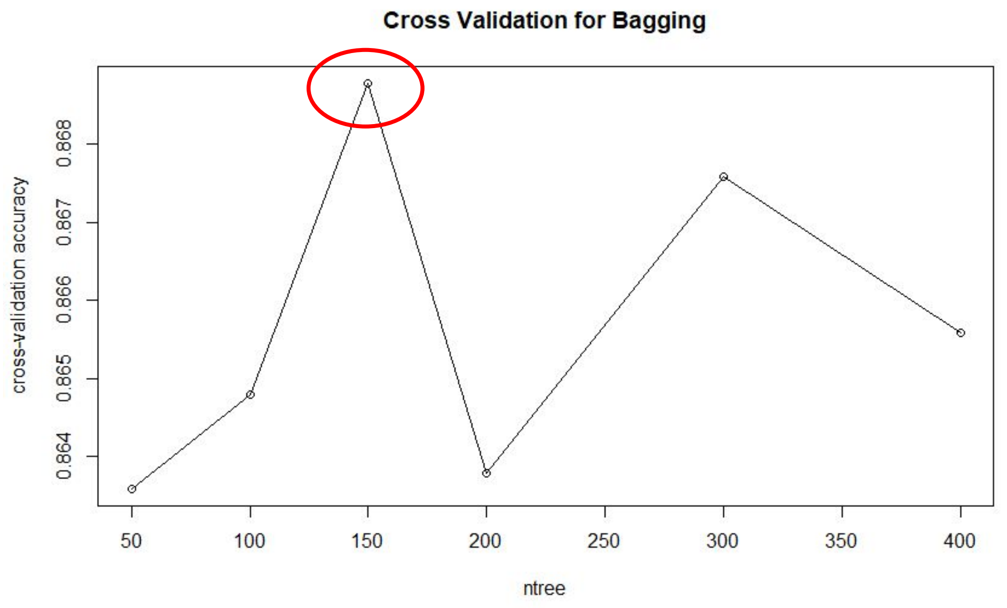
Sensitivity	Specificity	Precision	Recall	F1
0.852	0.75	0.864	0.852	0.858

Accuracy	Kappa
0.817	0.598

Bagging

Cross Validation

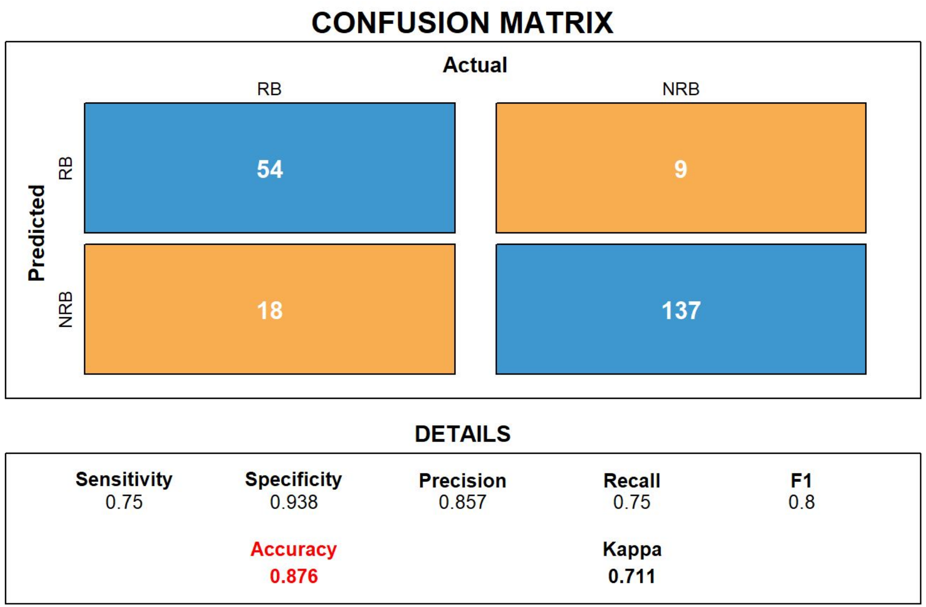
```r
num_trees<-c(50, 100, 150, 200, 300, 400)
bagging_cv_acc<-seq(0, 0, length = 6)
for (epoch in 1:6) {
  cv_index<-sample(837, 837)
  for (i in 1:6) {
    for (k in 1:5) {
      cv_test_index<-cv_index[(1+round(167.4*(k-1))):round(167.4*k)]
      qsar.bag.cv<-randomForest(as.factor(class)~.,data = qsar_train[-cv_test_index, ],
                               ntree = num_trees[i], mtry = 41, importance = TRUE)
      bag.cv.pred <- predict(qsar.bag.cv, newdata = qsar_train[cv_test_index, ])
      cv.acc<-mean(bag.cv.pred == qsar_train[cv_test_index, ]$class)
      bagging_cv_acc[i]<-bagging_cv_acc[i] + cv.acc
    }
  }
}
num_tree = num_trees[which.max(bagging_cv_acc)]
```



Bagging

R Code

```r
qsar.bagging <-
randomForest(as.factor(class) ~.,data =
qsar_train, ntree = num_tree, mtry = 41,
importance = TRUE)
bag.pred <- predict(qsar.bagging, newdata =
qsar_test)
mean(bag.pred == qsar_test$class)
```


CONFUSION MATRIX

		Actual	
		RB	NRB
Predicted	RB	54	9
	NRB	18	137

DETAILS

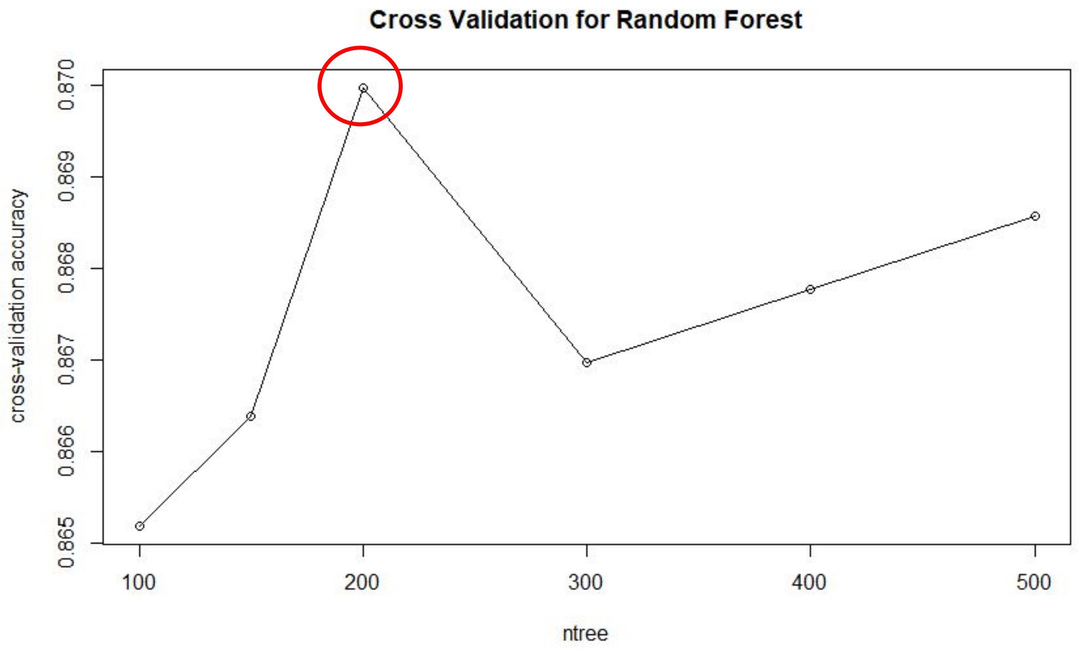
Sensitivity	Specificity	Precision	Recall	F1
0.75	0.938	0.857	0.75	0.8

Accuracy	Kappa
0.876	0.711

Random Forest

Cross Validation

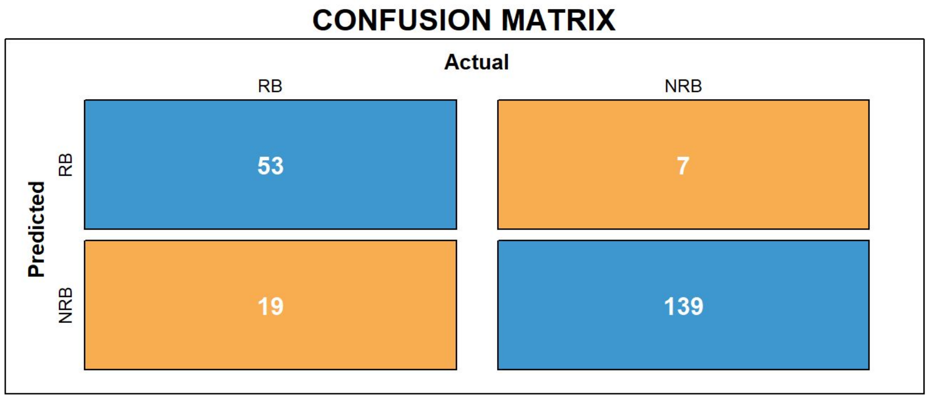
```
num_trees_rf<-c(100, 150, 200, 300, 400, 500)
rf_cv_acc<-seq(0, 0, length = 6)
for (epoch in 1:6) {
  cv_index<-sample(837, 837)
  for (i in 1:6) {
    for (k in 1:5) {
      cv_test_index<-cv_index[(1+round(167.4*(k-1))):round(167.4*k)]
      qsar.rf.cv<-randomForest(as.factor(class) ~.,data = qsar_train[-cv_test_index, ],
                               ntree = num_trees_rf[i], importance = TRUE)
      rf.cv.pred <- predict(qsar.rf.cv, newdata = qsar_train[cv_test_index, ])
      cv.acc<-mean(rf.cv.pred == qsar_train[cv_test_index, ]$class)
      rf_cv_acc[i]<-rf_cv_acc[i] + cv.acc
    }
  }
}
num_tree_rf = num_trees_rf[which.max(rf_cv_acc)]
```



Random Forest

R Code

```r
qsar.rf <- randomForest(as.factor(class)
~.,data = qsar_train, ntree = num_tree_rf,
importance = TRUE)
rf.pred <- predict(qsar.rf, newdata =
qsar_test)
mean(rf.pred == qsar_test$class)
```

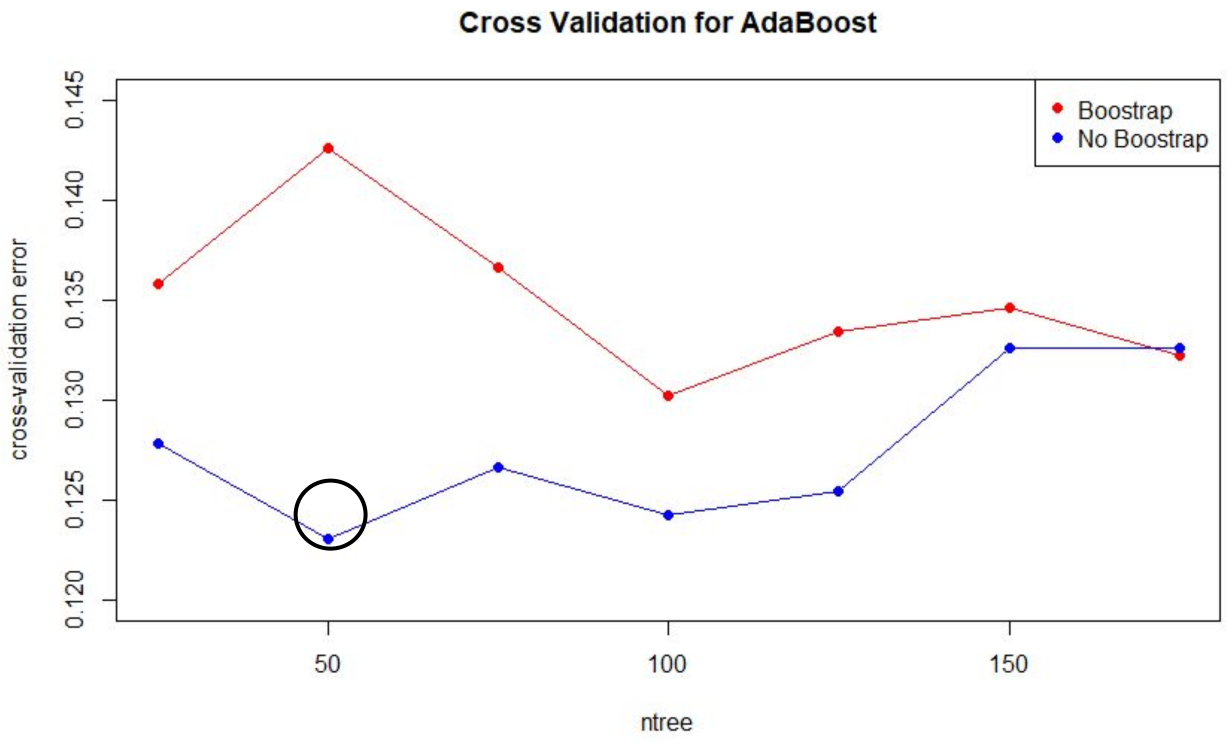


DETAILS

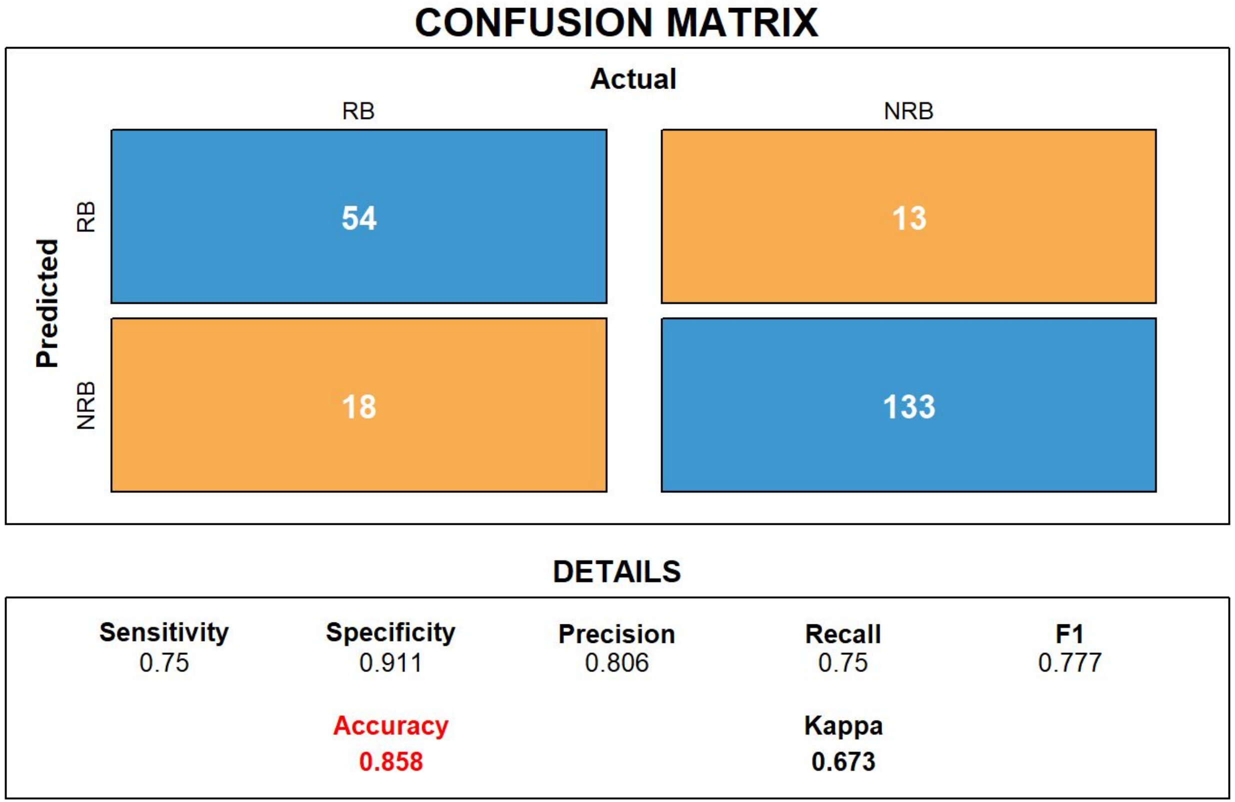
Sensitivity	Specificity	Precision	Recall	F1
0.736	0.952	0.883	0.736	0.803

Accuracy	Kappa
0.881	**0.719**

AdaBoost (with Freund coefficient)



AdaBoost (with Freund coefficient)



CONFUSION MATRIX

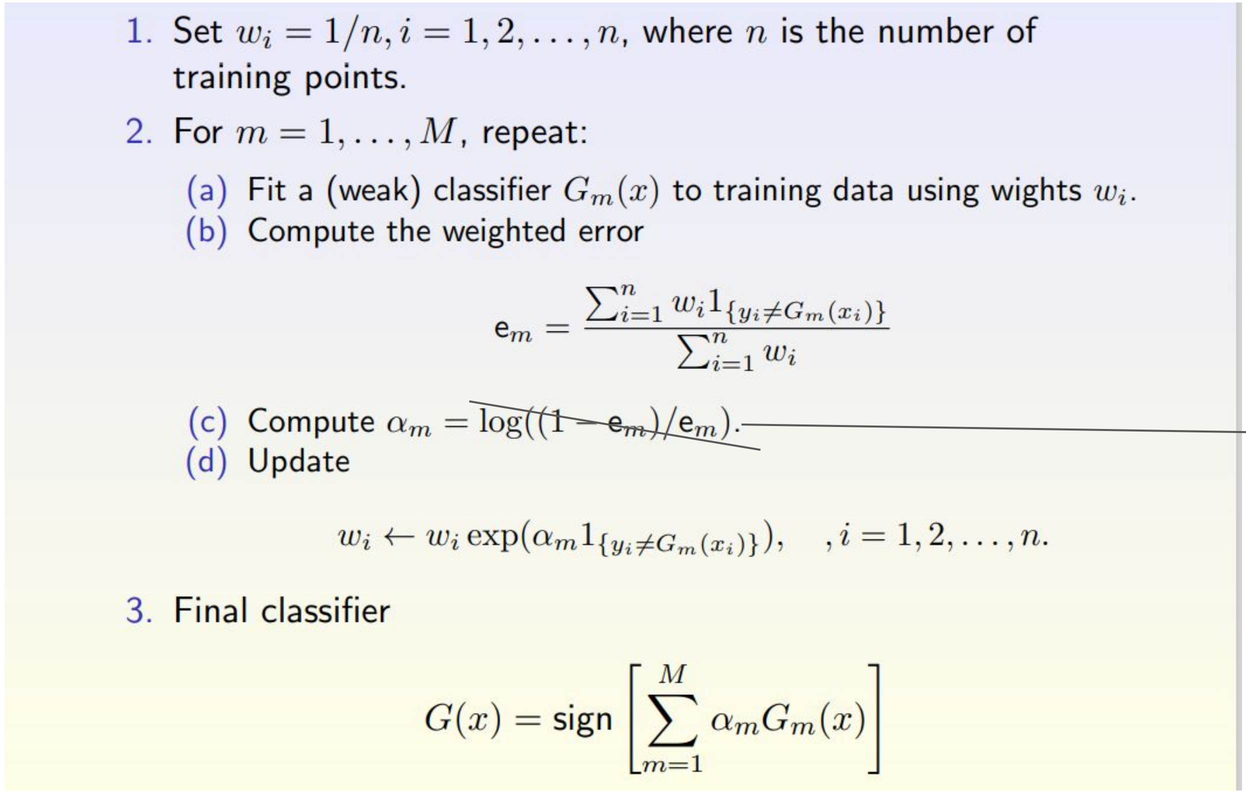
Predicted \ Actual	RB	NRB
RB	54	13
NRB	18	133

DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.75	0.911	0.806	0.75	0.777

Accuracy	Kappa
0.858	0.673

AdaBoost with Breiman coefficient



1. Set $w_i = 1/n, i = 1, 2, \ldots, n$, where n is the number of training points.
2. For $m = 1, \ldots, M$, repeat:
   (a) Fit a (weak) classifier $G_m(x)$ to training data using wights w_i.
   (b) Compute the weighted error

$$\mathrm{e}_m = \frac{\sum_{i=1}^{n} w_i 1_{\{y_i \neq G_m(x_i)\}}}{\sum_{i=1}^{n} w_i}$$

   (c) Compute $\alpha_m =$ ~~$\log((1 - \mathrm{e}_m)/\mathrm{e}_m)$~~.
   (d) Update

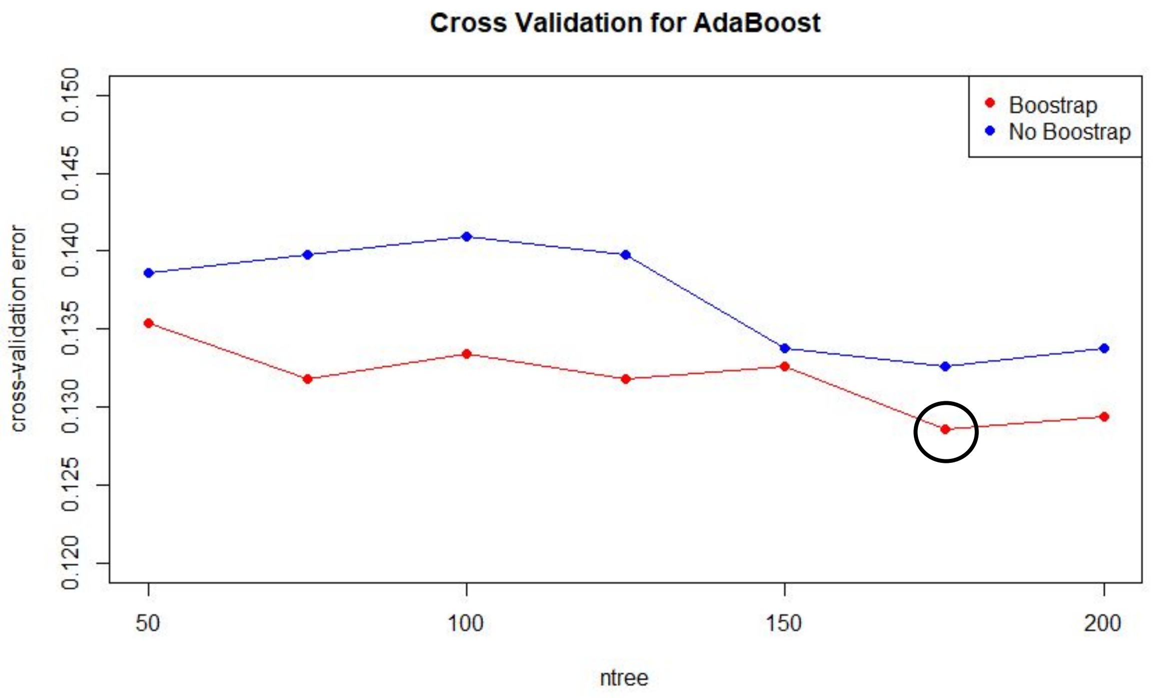
$$w_i \leftarrow w_i \exp(\alpha_m 1_{\{y_i \neq G_m(x_i)\}}), \quad , i = 1, 2, \ldots, n.$$

3. Final classifier

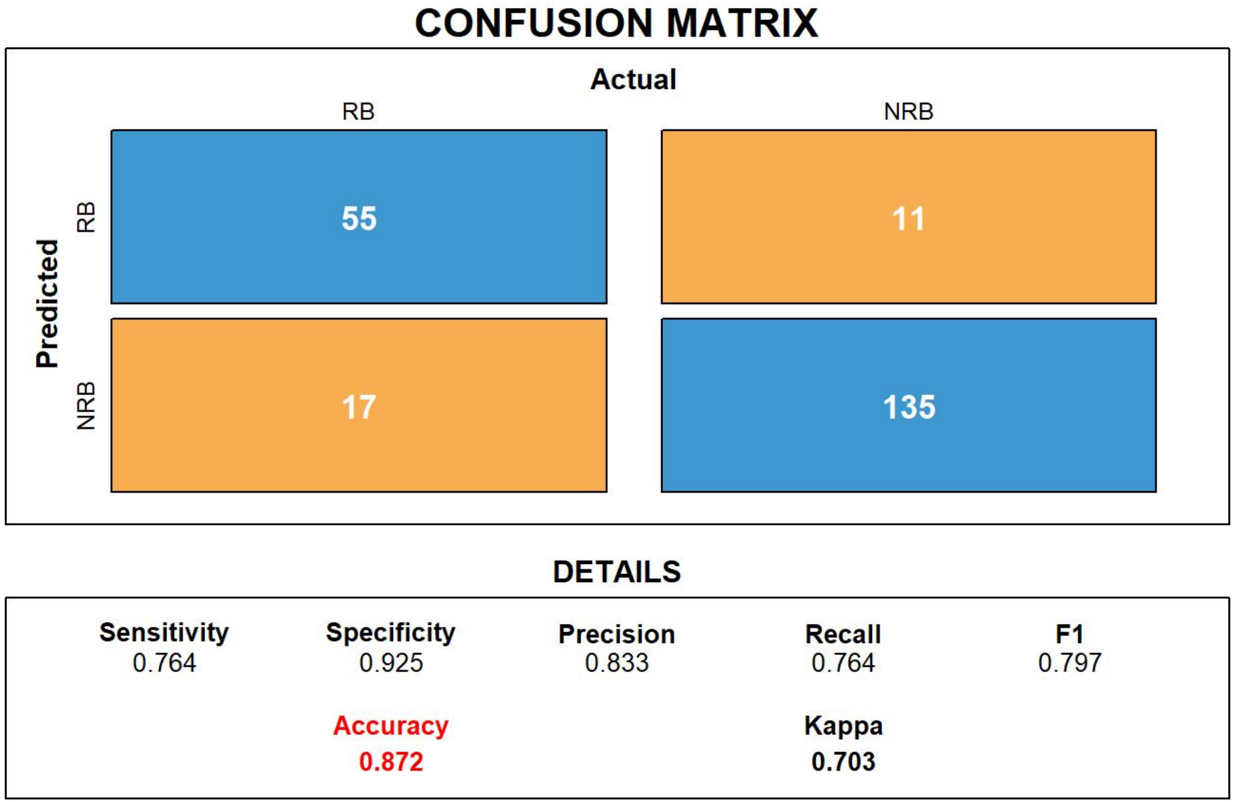
$$G(x) = \mathrm{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$$

$$\alpha_m = \frac{1}{2} \ln \frac{1 - e_m}{e_m}$$

AdaBoost with Breiman coefficient



AdaBoost with Breiman coefficient



CONFUSION MATRIX

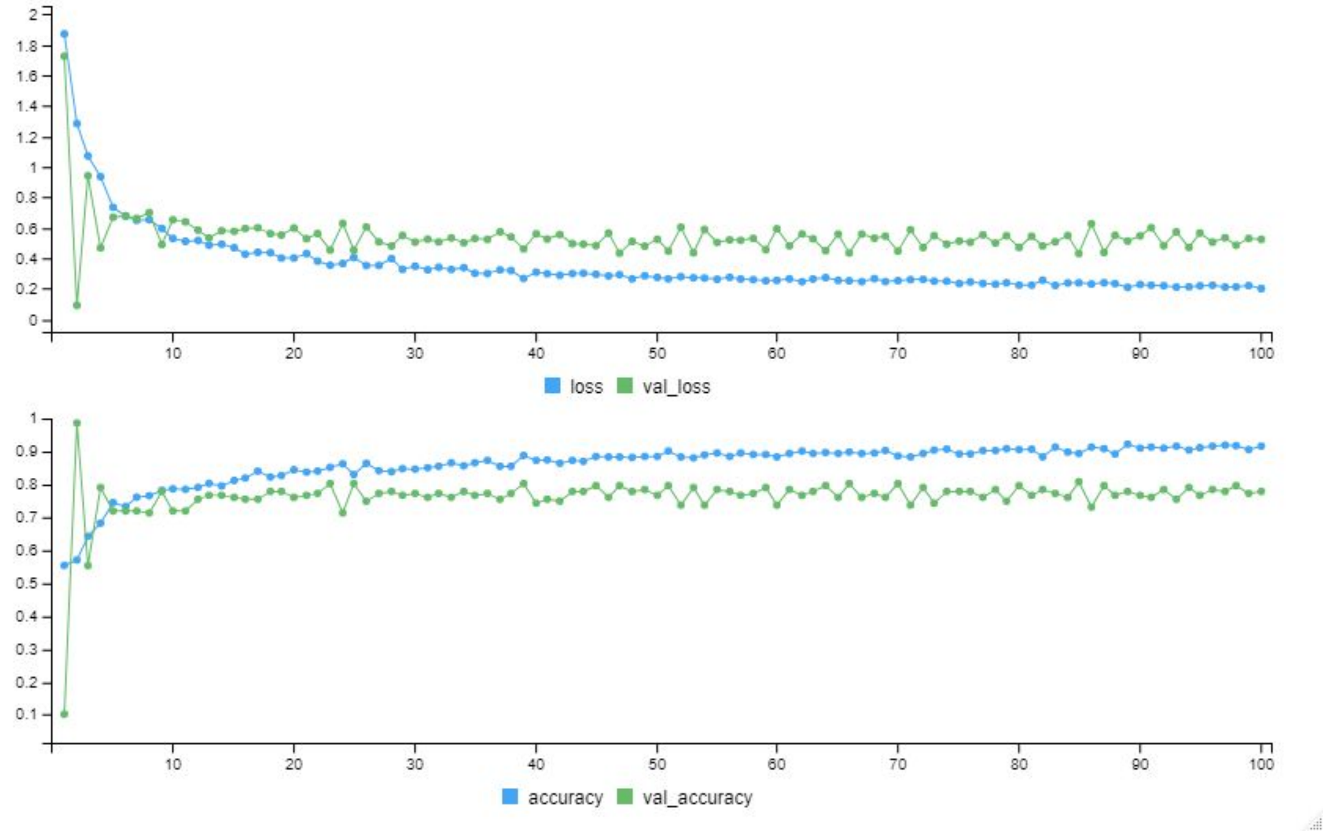
$$\begin{array}{c|cc} & \text{Actual RB} & \text{Actual NRB} \\ \hline \text{Predicted RB} & 55 & 11 \\ \text{Predicted NRB} & 17 & 135 \end{array}$$

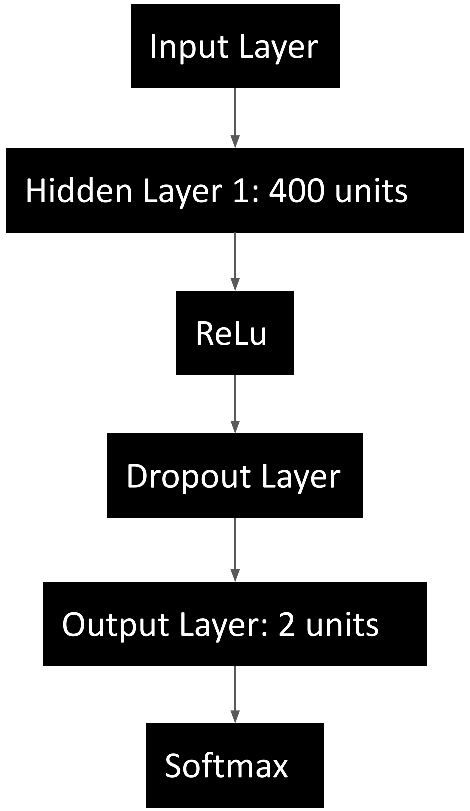
DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.764	0.925	0.833	0.764	0.797

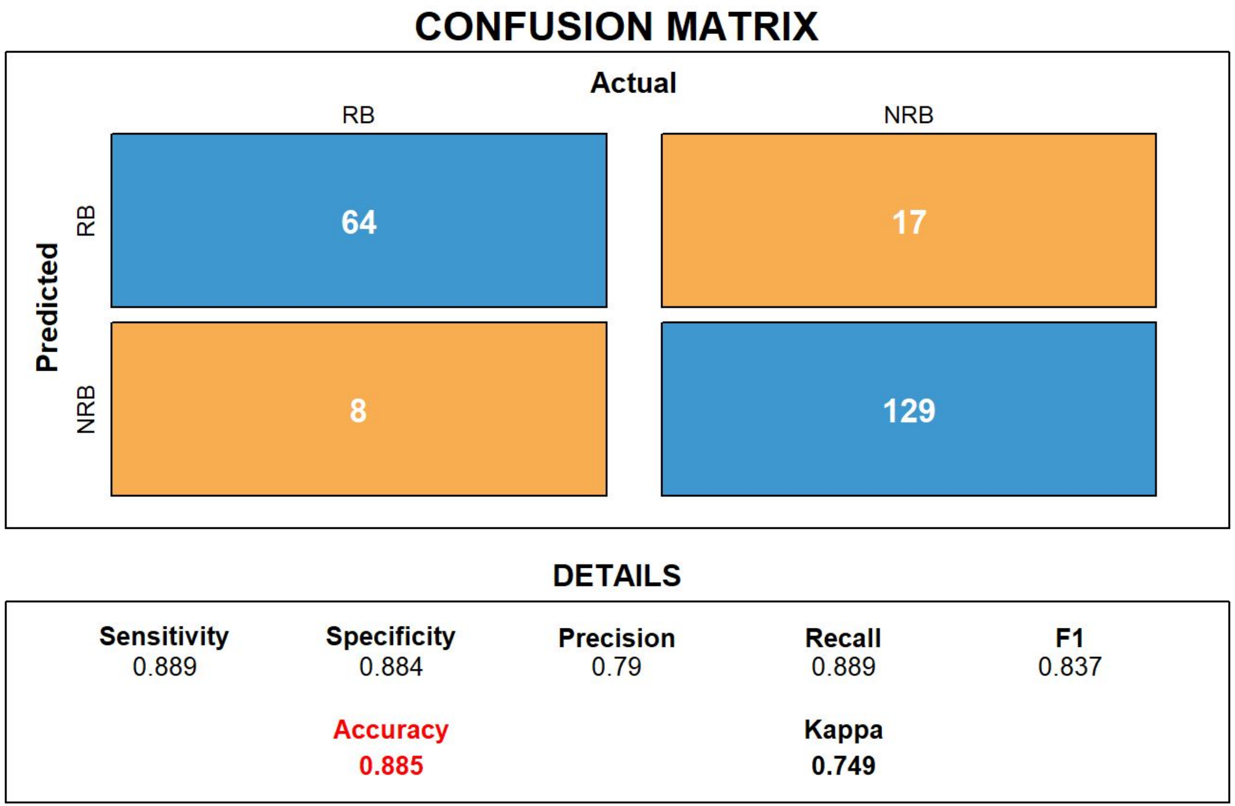
Accuracy	Kappa
0.872	0.703

Neural Network (NN): 1 hidden layer





Neural Network (NN): 1 hidden layer



CONFUSION MATRIX

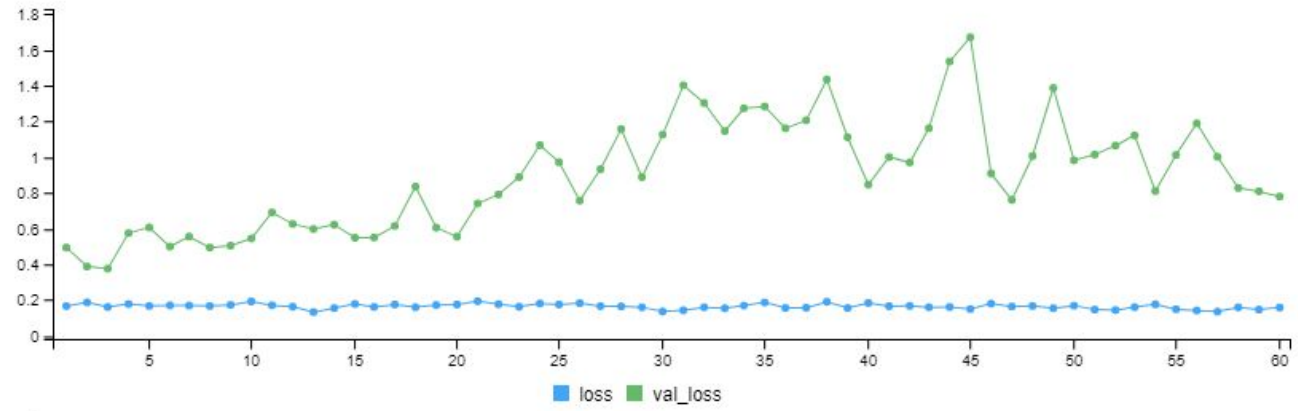
Predicted \ Actual	RB	NRB
RB	64	17
NRB	8	129

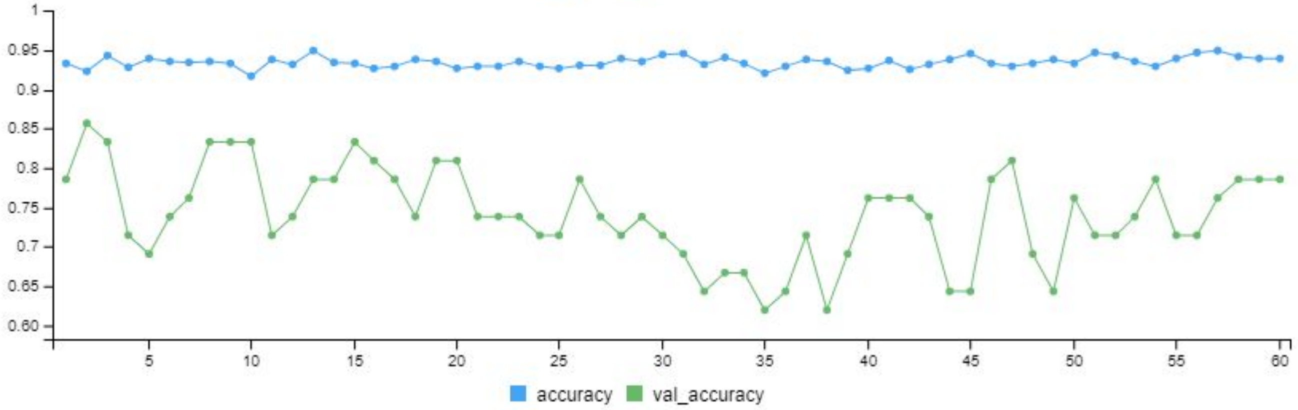
DETAILS

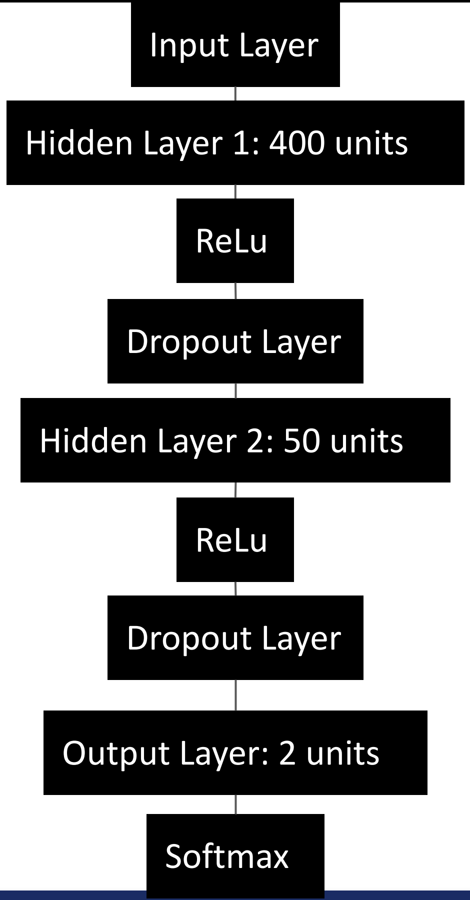
Sensitivity	Specificity	Precision	Recall	F1
0.889	0.884	0.79	0.889	0.837

Accuracy	Kappa
0.885	0.749

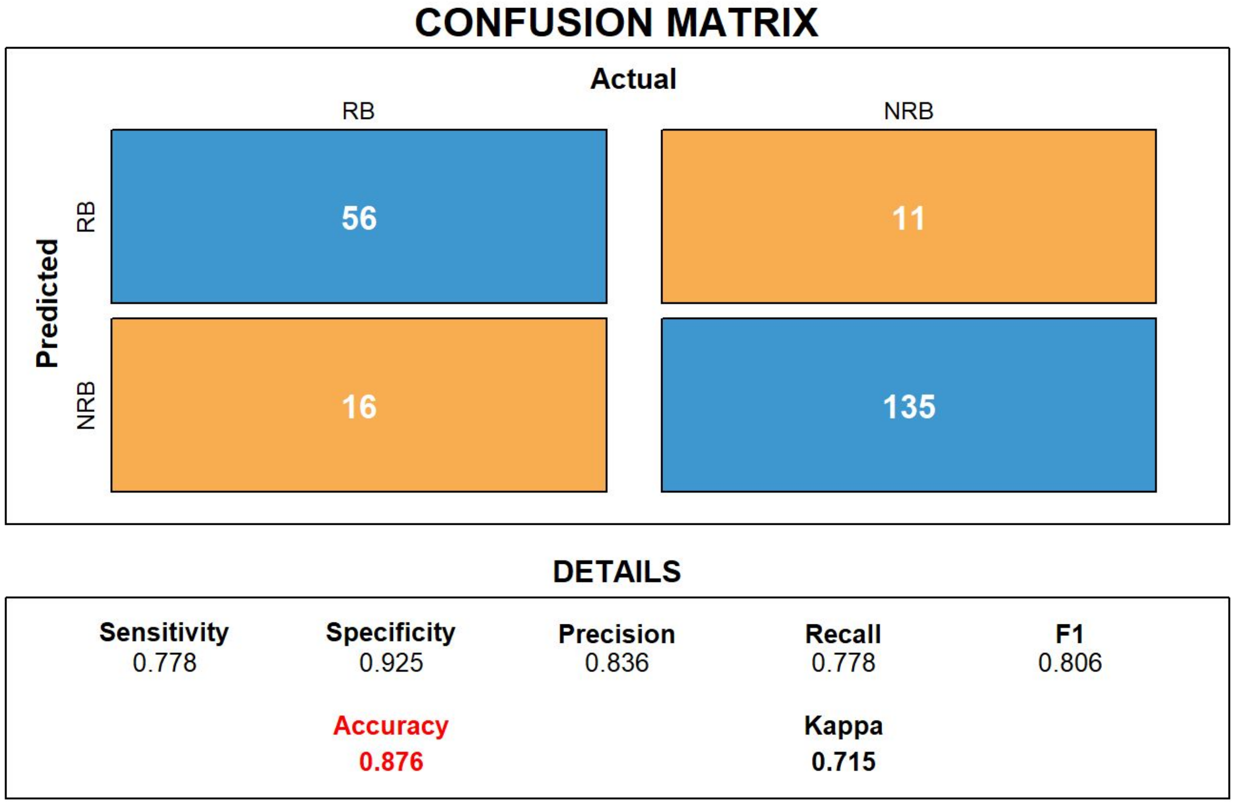
Deep Neural Network (DNN): 2 hidden layers







Deep Neural Network (DNN): 2 hidden layers



CONFUSION MATRIX

Predicted \ Actual	RB	NRB
RB	56	11
NRB	16	135

DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.778	0.925	0.836	0.778	0.806

Accuracy	Kappa
0.876	0.715

Part 4: Conclusion

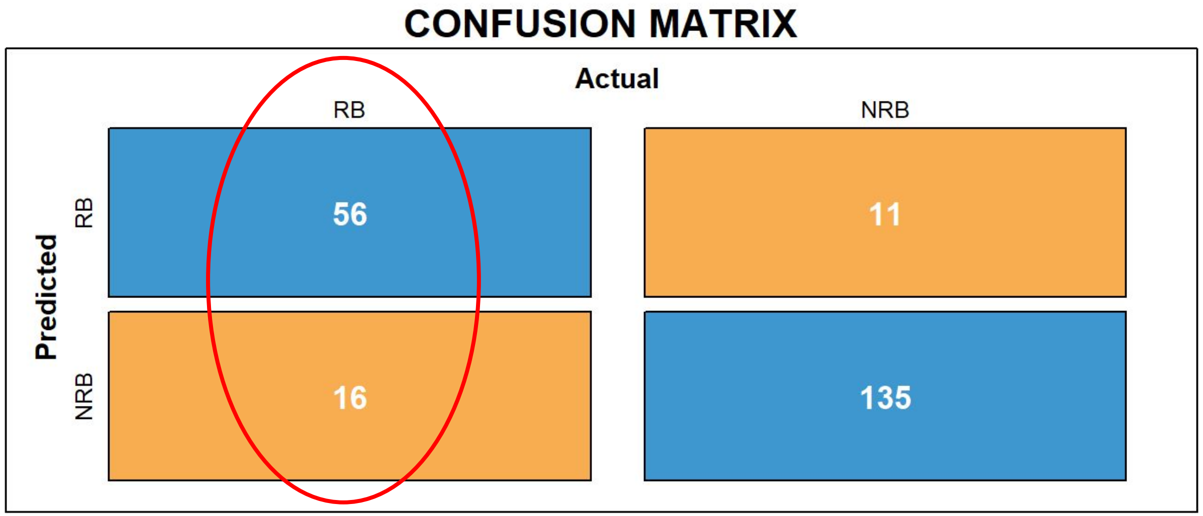
Individual Model Summary

D.S. = descriptors selection

Techniques	Accuracy	Specificity	Sensitivity
SVM+D.S.	85.3%	91.8%	72.2%
KNN+D.S.	87.6%	89.7%	83.3%
PLSDA+D.S.	85.3%	91.8%	72.2%
LDA	85.3%	91.8%	72.2%
Naive Bayes	72.9%	62.3%	94.4%
Tree	78.4%	75%	80.3%
Pruning Tree	81.7%	75%	85.2%
SVM	87.6%	93.2%	76.4%

Techniques	Accuracy	Specificity	Sensitivity
Bagging	87.6%	93.8%	75.0%
Random Forest	88.1%	95.2%	73.6%
Adaboost (with Breiman coef.)	87.2%	92.5%	76.4%
Adaboost (with Freud coef.)	85.8%	91.1%	75.0%
Neural Network (NN)	88.5%	88.4%	88.9%
DNN	87.6%	92.5%	77.8%

Individual Model Summary



DETAILS

Sensitivity 0.778	**Specificity** 0.925	**Precision** 0.836	**Recall** 0.778	**F1** 0.806
	Accuracy **0.876**		**Kappa** **0.715**	

Consensus Model

Consensus Model	Accuracy	Specificity	Sensitivity
C1: (SVM+D.S.) + (KNN+D.S.) + (PLSDA+D.S.)	87.61%	93.15%	76.39%
C2: NN + SVM + (KNN+D.S.)	88.99%	92.47%	81.94%
C3: NN + DNN + (KNN+D.S.)	89.91%	92.47%	84.72%
C4: NN + Adaboost (with Breiman coef.) + (KNN+D.S.)	90.37%	93.15%	84.72%
C5: NN + Random Forest + (KNN+D.S.)	89.91%	93.15%	83.33%
C6: NN + Bagging + (KNN+D.S.)	88.99%	91.78%	83.33%
C7: NN + Adaboost (with Freund coef.) + (KNN+D.S.)	90.37%	93.84%	83.33%
C8: NN * 2 + Adaboost (with Freud coef.) + Adaboost (with Breiman coef.) + (KNN+D.S.)	90.83%	93.84%	84.72%

Thanks For Listening

Lei Lyu (ll3433)
Yang Yu (yy3102)
Wenxiang Zhou (wz2542)
Yi Chen (yc4029)