Group 25 :Assignment 2.5

Code

Melissa Tan Joy Li 200249191 | Zhuoyang Li 480164337 | Biji George 470346121 | Joseph Tagudin 470542996 | Sergio Kulikovsky 480322960

31 十月, 2018

STAT5003 Computational Statistic Final Project 1

1 Introduction

Below is the problem statement issued from the precisionFDA website that forms part of this assignment.

“In biomedical research, sample mislabeling (accidental swapping of patient samples) or data mislabeling (accidental swapping of patient omics data) has been a long-standing problem that contributes to irreproducible results and invalid conclusions. These problems are particularly prevalent in large scale multi-omics studies, in which multiple different omics experiments are carried out at different time periods and/or in different labs. Human errors could arise during sample transferring, sample tracking, large-scale data generation, and data sharing/management. Thus, there is a pressing need to identify and correct sample and data mislabeling events to ensure the right data for the right patient. Simultaneous use of multiple types of omics platforms to characterize a large set of biological samples, as utilized in The Cancer Genome Atlas (TCGA) and the Clinical Proteomic Tumor Analysis Consortium (CPTAC) projects, has been demonstrated as a powerful approach to understanding the molecular basis of diseases and speeding the translation of new discoveries to patient care. Comprehensive multi-omics data obtained on the same patient sample can also add value in pinpointing and correcting mislabeling problems that can be encountered in the process. The FDA and NCI-CPTAC have joined forces to launch this challenge to encourage the development and evaluation of computational algorithms that can accurately detect and correct mislabeled samples using rich multi-omics datasets (Boja et al. 2018)

Paired proteomics data were generated for each of the 162 tumor samples. Protein quantification was based on spectral counting and mRNA quantification was based on Fragments Per Kilobase of transcript per Million mapped reads (FPKM). For both proteomics and RNA-Seq data, genes with more than 50% missing values were removed, except for genes located in X or Y chromosomes, which were retained even if they were missed in more than 50% of the samples. The proteomics data was then normalized using quantile normalization followed by batch correction using ComBat, whereas the RNA-Seq data was normalized using the trimmed mean of M-values normalization method (TMM) followed by batch correction using ComBat“”

2 Data Preparation

- **train|test_cli** - Contains clinical information such as gender and Microsatellite instability (MSI) status for the 80 training samples.
- **train|test_pro** - Proteomics data from the 80 training samples. Each row represents a protein and each column represents a training sample.
- **train|test_key** - Mislabelling information for the training samples. 0 indicates match i.e. clinical and proteomics data are from the same sample. 1 indicates that are not from the sample.

The gender & msi columns in the train_cli and test_cli.csv are combined and stored as factor.

Code

	ZNF638 <dbl>	ZNF706 <dbl>	ZPR1 <dbl>	ZW10 <dbl>	ZYX <dbl>	ZZEF1 <dbl>
Training_1	1.007971	1.005565	1.016313	1.859706	3.834986	*NA*
Training_2	1.516790	1.009703	*NA*	1.465056	4.156957	*NA*
Training_3	*NA*	1.009703	1.019879	1.932920	4.303349	*NA*
Training_4	1.024392	*NA*	1.589198	1.569086	5.218254	*NA*
Training_5	1.134580	1.120853	1.129109	1.205099	4.190499	1.756951

5 rows

Code

	ZNF638 <dbl>	ZNF706 <dbl>	ZPR1 <dbl>	ZW10 <dbl>	ZYX <dbl>	ZZEF1 <dbl>
Testing_1	*NA*	1.058798	1.848274	2.565798	3.597116	1.636396
Testing_2	1.733361	*NA*	0.997150	2.265862	3.967766	1.802938
Testing_3	1.126579	*NA*	1.120824	2.944166	3.779964	1.756282
Testing_4	1.081373	1.780610	1.074043	*NA*	3.918465	*NA*
Testing_5	*NA*	1.721321	*NA*	1.795269	3.535849	*NA*

5 rows

Code

3 Data Preprocessing

> We will begin by analysing the data and understand the challenges. 10 kold cross validation is repeated 5 times is used in all approaches.

The following were provided * 10% labelling errors were introduced to the samples for the proteomics data. and 5% to the clinical information table. Sample labelling errors were not shared across different types of data (i.e., for each sample, a mislabelling error only occurs in, at most, one type of data), so that all three data types can be used to identify the sources of the error. * For proteomics there are three error types: * Sample duplication (B to A', where A' is a duplicate of A), sample swapping (A to B and B to A), * Sample shifting (A to B, B to C, and C to D). * Duplicated proteomic samples came from technical replications (outputs from independent proteomics experiments of the same biological samples). The swapped samples were required to have different gender or MSI status. * For clinical data, swapping (A to B and B to A) between gender inconsistent samples.

3.1 Data Imputation

Data imputation is done to remove or substitute missing values in the dataset. Missing data has to be addressed, otherwise it will bring in a lot of vagueness and undermine the validity the data. In our project, we combined both the training and test set for data imputation to avoid variance in the features being imputed. After the imputation, the samples were separated, back again, to the training and testing samples respectively. For e.g. in the Proteomic training dataset, feature “ATP7A” has 79 out of 80 rows as NA. As such, those features that had more than 90% missing data had been removed before the data imputation was done. [2,3]

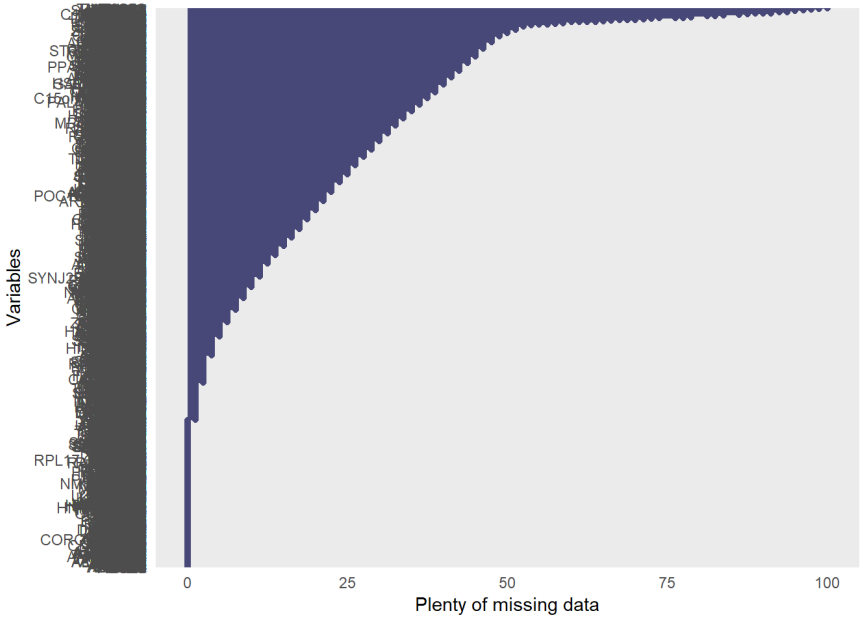
In the following sub-sections, the single imputation approaches taken for this project are described, and only the best result of training the imputed data was further taken for evaluating the prediction models. The KNN algorithm was used to benchmark the results for each of the imputation approaches, and further selection, by evaluating the metrics such as the F1 score, accuracy and confusion matrix.

Note: MICE – Multiple Imputation by Chained Equations (MICE) is an iterative algorithm based on chained equations that used an imputation model specified separately for each variable and involving the other variables as predictors. This approach was abandoned as it was resource intensive and the proteomic training dataset had 4,118 features to begin with.

3.1.1 Summary

Analyse the dataset to see how much NA we are dealing with.

Code



Code

3.1.2 Mean Imputation

Mean imputation consists of replacing the missing data for a given feature by the mean of all known values of that feature. The Hmisc R package was used for this imputation. The main advantages of mean imputation are that it is simple to apply and understand, it does not reduce the sample size in any form and, if the missing values are completely random, then this imputation approach is totally unbiased. Disadvantages are that it can be biased towards the multivariate estimates such as correlation or regression coefficients as well as towards the standard variance and error. [3]

As each chuck are run against a set of files, code to upsampled the data is also included to reduce file I/O.

Code

3.1.3 Median Imputation

Median imputation consists of replacing the missing data for a given feature by the median of all known values of that feature. The Hmisc R package was used for this imputation. This approach has similar advantages and disadvantages as the mean imputation. It has one major advantage over mean imputation is that it is preferable when the dataset is skewed, and this approach is outlier robust. [4]

Code

3.1.4 KNN-Imputation

The kNN imputation approach is based on the kNN algorithm. These values are obtained by using similarity-based methods that rely on distance metrics such as Euclidean distance, Jaccard similarity, Minkowski norm etc. They can be used to predict both discrete and continuous attributes. The main disadvantage of using kNN imputation is that it becomes time-consuming when analysing large datasets because it searches for similar instances throughout the dataset. Note that an important criterion while using kNN imputation is selecting the optimal value for the number of neighbours (k).

Code

3.1.5 missMDA

missMDA performs principal component methods with missing values and is also used to impute data with PC methods. To achieve this goal, the missing values are predicted using the iterative PCA algorithm for a predefined number of dimensions. Then, PCA is performed on the imputed dataset. The single imputation step requires tuning of the number of dimensions used to impute the data.

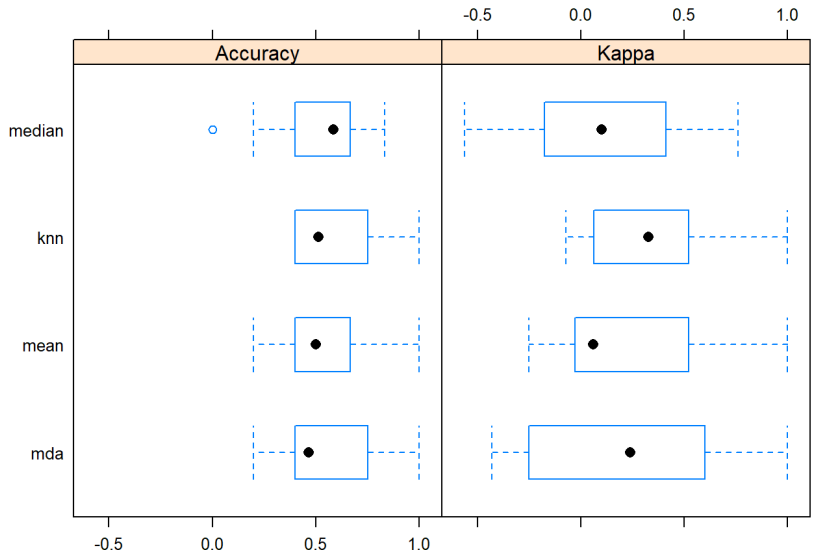
This imputation approach takes quite a while to run. Therefore, this chunk was disabled and executed manually to generate a once-off csv file which was read-in directly for any future processing.

Code

Code

3.1.6 Result

Code



Code


```
## [1] "cm_mean"
##                          Sensitivity Specificity Precision Recall
## Class: Female MSI-High           0.5   0.8823529 0.3333333    0.5
## Class: Female MSI-Low/MSS        0.9   0.6666667 0.7500000    0.9
## Class: Male MSI-High             1.0   1.0000000 1.0000000    1.0
## Class: Male MSI-Low/MSS          0.5   1.0000000 1.0000000    0.5
##                                    F1 Balanced Accuracy
## Class: Female MSI-High    0.4000000          0.6911765
## Class: Female MSI-Low/MSS 0.8181818          0.7833333
## Class: Male MSI-High      1.0000000          1.0000000
## Class: Male MSI-Low/MSS   0.6666667          0.7500000
## [1] "cm_median"
##                          Sensitivity Specificity Precision Recall
## Class: Female MSI-High           0.5   0.9411765 0.5000000    0.5
## Class: Female MSI-Low/MSS        0.9   0.5555556 0.6923077    0.9
## Class: Male MSI-High             0.0   1.0000000        NA    0.0
## Class: Male MSI-Low/MSS          0.5   0.9230769 0.7500000    0.5
##                                    F1 Balanced Accuracy
## Class: Female MSI-High    0.5000000          0.7205882
## Class: Female MSI-Low/MSS 0.7826087          0.7277778
## Class: Male MSI-High             NA          0.5000000
## Class: Male MSI-Low/MSS   0.6000000          0.7115385
## [1] "cm_knn"
##                          Sensitivity Specificity Precision    Recall
## Class: Female MSI-High     0.5000000   0.9411765 0.5000000 0.5000000
## Class: Female MSI-Low/MSS  0.8000000   0.6666667 0.7272727 0.8000000
## Class: Male MSI-High       0.0000000   1.0000000        NA 0.0000000
## Class: Male MSI-Low/MSS    0.6666667   0.8461538 0.6666667 0.6666667
##                                    F1 Balanced Accuracy
## Class: Female MSI-High    0.5000000          0.7205882
## Class: Female MSI-Low/MSS 0.7619048          0.7333333
## Class: Male MSI-High             NA          0.5000000
## Class: Male MSI-Low/MSS   0.6666667          0.7564103
## [1] "cm_mda"
##                          Sensitivity Specificity Precision    Recall
## Class: Female MSI-High     0.5000000   0.9411765 0.5000000 0.5000000
## Class: Female MSI-Low/MSS  0.9000000   0.4444444 0.6428571 0.9000000
## Class: Male MSI-High       0.0000000   1.0000000        NA 0.0000000
## Class: Male MSI-Low/MSS    0.3333333   0.9230769 0.6666667 0.3333333
##                                    F1 Balanced Accuracy
## Class: Female MSI-High    0.5000000          0.7205882
## Class: Female MSI-Low/MSS 0.7500000          0.6722222
## Class: Male MSI-High             NA          0.5000000
## Class: Male MSI-Low/MSS   0.4444444          0.6282051
```

3.2 Class Imbalance

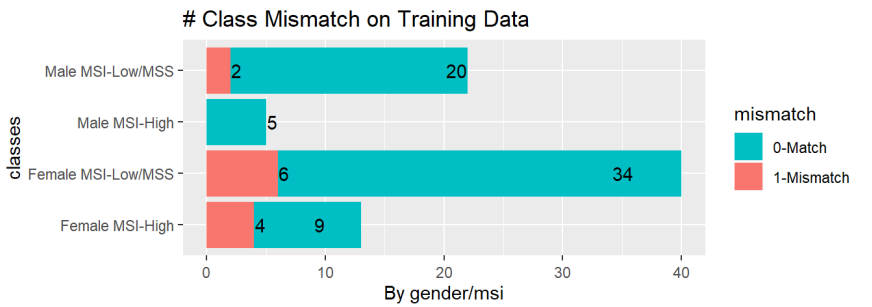
Class Imbalance is a scenario that happens quite often in machine learning where certain classes have less occurrence in comparison to other classes in the data. For e.g.: in the FDA dataset exploratory charts below, there are mismatch labels for class “3-Male/MSI-High”. Also, there was an imbalance in the number of mismatches versus the match rows present in the dataset i.e. 68 versus 12 respectively. [1]

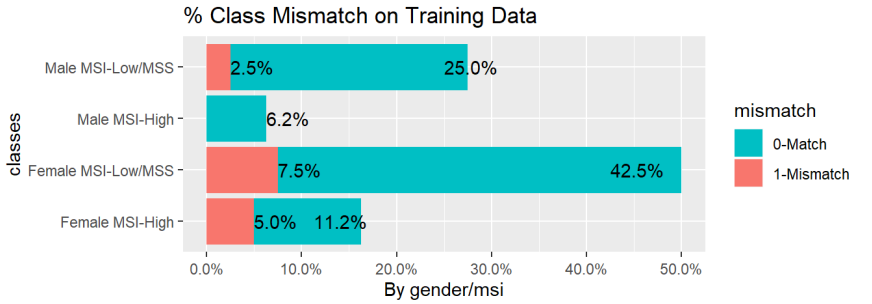
In R, this problem can be tackled by using the weighted, or sampling, approach. For the project, we decided to go with up-sampling as the dataset only has 80 samples and it couldn’t be further reduced. Also, the model training to be done downstream only uses the labels that matched, which

further brought the overall sample size down to only 68 samples.

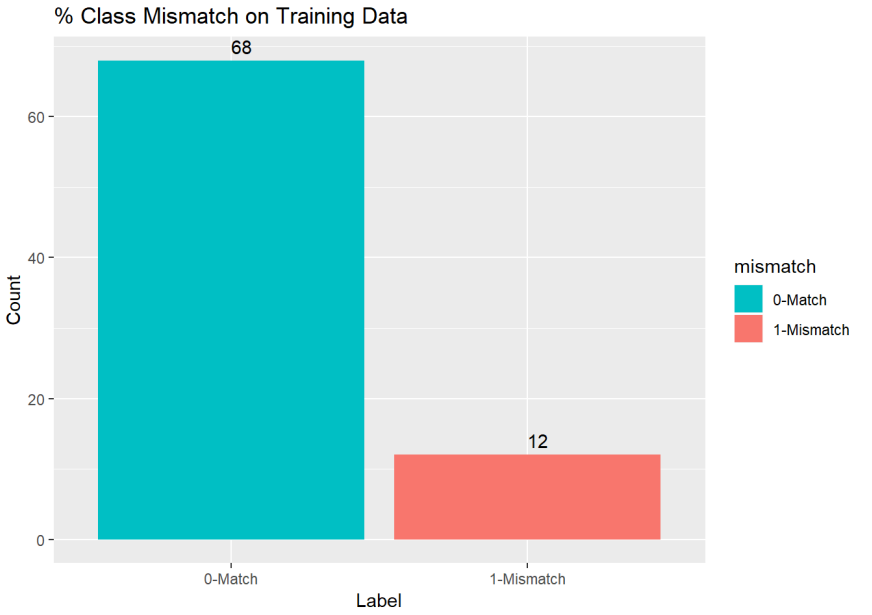
In the up-sampling approach, it is vital that we do it only for the training set. Also, the approach is to sample, with replacements, to make the class distributions equal.

Code





Code

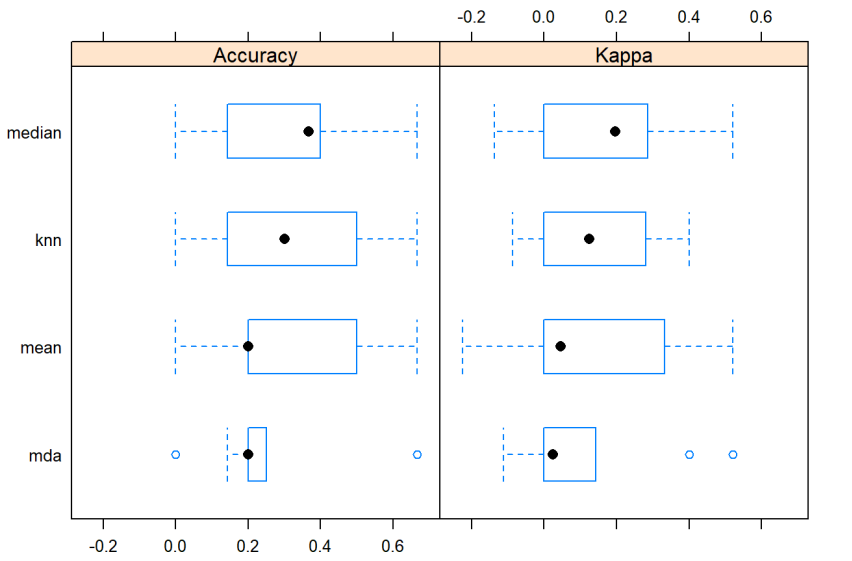


3.2.1 Result

> Result seems worst off

The initial preliminary analysis is slotted in here to show that by upsampling the data, we have overfitted the data. Performance against test was poor.

Code



Code

```
## [1] "cm_mean"
##                          Sensitivity Specificity Precision Recall
## Class: Female MSI-High           0.5   0.8823529 0.3333333    0.5
## Class: Female MSI-Low/MSS        0.9   0.6666667 0.7500000    0.9
## Class: Male MSI-High             1.0   1.0000000 1.0000000    1.0
## Class: Male MSI-Low/MSS          0.5   1.0000000 1.0000000    0.5
##                                F1 Balanced Accuracy
## Class: Female MSI-High    0.4000000         0.6911765
## Class: Female MSI-Low/MSS 0.8181818         0.7833333
## Class: Male MSI-High      1.0000000         1.0000000
## Class: Male MSI-Low/MSS   0.6666667         0.7500000
## [1] "cm_median"
##                          Sensitivity Specificity Precision Recall
## Class: Female MSI-High           0.5   0.9411765 0.5000000    0.5
## Class: Female MSI-Low/MSS        0.9   0.5555556 0.6923077    0.9
## Class: Male MSI-High             0.0   1.0000000        NA    0.0
## Class: Male MSI-Low/MSS          0.5   0.9230769 0.7500000    0.5
##                                F1 Balanced Accuracy
## Class: Female MSI-High    0.5000000         0.7205882
## Class: Female MSI-Low/MSS 0.7826087         0.7277778
## Class: Male MSI-High             NA         0.5000000
## Class: Male MSI-Low/MSS   0.6000000         0.7115385
## [1] "cm_knn"
##                          Sensitivity Specificity Precision    Recall
## Class: Female MSI-High     0.5000000   0.9411765 0.5000000 0.5000000
## Class: Female MSI-Low/MSS  0.8000000   0.6666667 0.7272727 0.8000000
## Class: Male MSI-High       0.0000000   1.0000000        NA 0.0000000
## Class: Male MSI-Low/MSS    0.6666667   0.8461538 0.6666667 0.6666667
##                                F1 Balanced Accuracy
## Class: Female MSI-High    0.5000000         0.7205882
## Class: Female MSI-Low/MSS 0.7619048         0.7333333
## Class: Male MSI-High             NA         0.5000000
## Class: Male MSI-Low/MSS   0.6666667         0.7564103
## [1] "cm_mda"
##                          Sensitivity Specificity Precision    Recall
## Class: Female MSI-High     0.5000000   0.9411765 0.5000000 0.5000000
## Class: Female MSI-Low/MSS  0.9000000   0.4444444 0.6428571 0.9000000
## Class: Male MSI-High       0.0000000   1.0000000        NA 0.0000000
## Class: Male MSI-Low/MSS    0.3333333   0.9230769 0.6666667 0.3333333
##                                F1 Balanced Accuracy
## Class: Female MSI-High    0.5000000         0.7205882
## Class: Female MSI-Low/MSS 0.7500000         0.6722222
## Class: Male MSI-High             NA         0.5000000
## Class: Male MSI-Low/MSS   0.4444444         0.6282051
```

Conclusion : From the result, can observed that the accuracy and kappa have dropped. The chunk below also shows that the the accuracy dropped after the data has been resampled. The is likely due overfitting as

Code


```
## Confusion Matrix and Statistics
##
##                     Reference
## Prediction           Female MSI-High Female MSI-Low/MSS Male MSI-High
##   Female MSI-High                  1                  0             1
##   Female MSI-Low/MSS               1                  8             0
##   Male MSI-High                    0                  0             0
##   Male MSI-Low/MSS                 0                  2             0
##                     Reference
## Prediction           Male MSI-Low/MSS
##   Female MSI-High                  0
##   Female MSI-Low/MSS               2
##   Male MSI-High                    0
##   Male MSI-Low/MSS                 4
##
## Overall Statistics
##
##                Accuracy : 0.6842
##                  95% CI : (0.4345, 0.8742)
##     No Information Rate : 0.5263
##     P-Value [Acc > NIR] : 0.1248
##
##                   Kappa : 0.4597
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Female MSI-High Class: Female MSI-Low/MSS
## Sensitivity                         0.50000                     0.8000
## Specificity                         0.94118                     0.6667
## Pos Pred Value                      0.50000                     0.7273
## Neg Pred Value                      0.94118                     0.7500
## Prevalence                          0.10526                     0.5263
## Detection Rate                      0.05263                     0.4211
## Detection Prevalence                0.10526                     0.5789
## Balanced Accuracy                   0.72059                     0.7333
##                      Class: Male MSI-High Class: Male MSI-Low/MSS
## Sensitivity                       0.00000                  0.6667
## Specificity                       1.00000                  0.8462
## Pos Pred Value                        NaN                  0.6667
## Neg Pred Value                    0.94737                  0.8462
## Prevalence                        0.05263                  0.3158
## Detection Rate                    0.00000                  0.2105
## Detection Prevalence              0.00000                  0.3158
## Balanced Accuracy                 0.50000                  0.7564
```

Code

```
## Confusion Matrix and Statistics
##
##                     Reference
## Prediction           Female MSI-High Female MSI-Low/MSS Male MSI-High
##   Female MSI-High                  1                  1             0
##   Female MSI-Low/MSS               0                  6             0
##   Male MSI-High                    1                  0             1
##   Male MSI-Low/MSS                 0                  3             0
##                     Reference
## Prediction           Male MSI-Low/MSS
##   Female MSI-High                   1
##   Female MSI-Low/MSS                0
##   Male MSI-High                     2
##   Male MSI-Low/MSS                  3
##
## Overall Statistics
##
##               Accuracy : 0.5789
##                 95% CI : (0.335, 0.7975)
##    No Information Rate : 0.5263
##    P-Value [Acc > NIR] : 0.4111
##
##                  Kappa : 0.4039
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Female MSI-High Class: Female MSI-Low/MSS
## Sensitivity                         0.50000                    0.6000
## Specificity                         0.88235                    1.0000
## Pos Pred Value                      0.33333                    1.0000
## Neg Pred Value                      0.93750                    0.6923
## Prevalence                          0.10526                    0.5263
## Detection Rate                      0.05263                    0.3158
## Detection Prevalence                0.15789                    0.3158
## Balanced Accuracy                   0.69118                    0.8000
##                      Class: Male MSI-High Class: Male MSI-Low/MSS
## Sensitivity                       1.00000                  0.5000
## Specificity                       0.83333                  0.7692
## Pos Pred Value                    0.25000                  0.5000
## Neg Pred Value                    1.00000                  0.7692
## Prevalence                        0.05263                  0.3158
## Detection Rate                    0.05263                  0.1579
## Detection Prevalence              0.21053                  0.3158
## Balanced Accuracy                 0.91667                  0.6346
```

Code


```
## Confusion Matrix and Statistics
##
##                     Reference
## Prediction           Female MSI-High Female MSI-Low/MSS Male MSI-High
##   Female MSI-High                  3                  2             4
##   Female MSI-Low/MSS               0                 11             3
##   Male MSI-High                    0                  0             0
##   Male MSI-Low/MSS                 3                 12             1
##                     Reference
## Prediction           Male MSI-Low/MSS
##   Female MSI-High                   2
##   Female MSI-Low/MSS               21
##   Male MSI-High                     1
##   Male MSI-Low/MSS                 17
##
## Overall Statistics
##
##               Accuracy : 0.3875
##                 95% CI : (0.2806, 0.503)
##    No Information Rate : 0.5125
##    P-Value [Acc > NIR] : 0.99075
##
##                  Kappa : 0.0434
##  Mcnemar's Test P-Value : 0.07013
##
## Statistics by Class:
##
##                      Class: Female MSI-High Class: Female MSI-Low/MSS
## Sensitivity                         0.5000                     0.4400
## Specificity                         0.8919                     0.5636
## Pos Pred Value                      0.2727                     0.3143
## Neg Pred Value                      0.9565                     0.6889
## Prevalence                          0.0750                     0.3125
## Detection Rate                      0.0375                     0.1375
## Detection Prevalence                0.1375                     0.4375
## Balanced Accuracy                   0.6959                     0.5018
##                      Class: Male MSI-High Class: Male MSI-Low/MSS
## Sensitivity                        0.0000                  0.4146
## Specificity                        0.9861                  0.5897
## Pos Pred Value                     0.0000                  0.5152
## Neg Pred Value                     0.8987                  0.4894
## Prevalence                         0.1000                  0.5125
## Detection Rate                     0.0000                  0.2125
## Detection Prevalence               0.0125                  0.4125
## Balanced Accuracy                  0.4931                  0.5022
```

Code

```
## Confusion Matrix and Statistics
##
##                     Reference
## Prediction           Female MSI-High Female MSI-Low/MSS Male MSI-High
##   Female MSI-High                  2                  5             5
##   Female MSI-Low/MSS               0                  5             0
##   Male MSI-High                    3                  8             3
##   Male MSI-Low/MSS                 1                  7             0
##                     Reference
## Prediction           Male MSI-Low/MSS
##   Female MSI-High                   7
##   Female MSI-Low/MSS                6
##   Male MSI-High                    10
##   Male MSI-Low/MSS                 18
##
## Overall Statistics
##
##                Accuracy : 0.35
##                  95% CI : (0.2467, 0.4648)
##     No Information Rate : 0.5125
##     P-Value [Acc > NIR] : 0.9988
##
##                   Kappa : 0.1248
##  Mcnemar's Test P-Value : 9.088e-05
##
## Statistics by Class:
##
##                      Class: Female MSI-High Class: Female MSI-Low/MSS
## Sensitivity                          0.3333                    0.2000
## Specificity                          0.7703                    0.8909
## Pos Pred Value                       0.1053                    0.4545
## Neg Pred Value                       0.9344                    0.7101
## Prevalence                           0.0750                    0.3125
## Detection Rate                       0.0250                    0.0625
## Detection Prevalence                 0.2375                    0.1375
## Balanced Accuracy                    0.5518                    0.5455
##                      Class: Male MSI-High Class: Male MSI-Low/MSS
## Sensitivity                        0.3750                  0.4390
## Specificity                        0.7083                  0.7949
## Pos Pred Value                     0.1250                  0.6923
## Neg Pred Value                     0.9107                  0.5741
## Prevalence                         0.1000                  0.5125
## Detection Rate                     0.0375                  0.2250
## Detection Prevalence               0.3000                  0.3250
## Balanced Accuracy                  0.5417                  0.6169
```

3.3 Feature Selection

This is the procedure used to narrow down a subset of features, or attributes, for use in predictive modelling. Feature selection is useful for several reasons: * it provides the best ammunition to use against the Curse of Dimensionality; * it can shorten training times overall; * and it provides a solid buttress against overfitting, which increases the ability of the model to generalise.

A common view of feature selection contemplates that the variables most used by various machine learning algorithms are to be regarded as the most important. Depending on how the machine learning algorithm learns the relationship between Xs and Y, different machine learning algorithms

may sometimes end up using different variables (but mostly common variables) to various degrees. For example, the variables that have been demonstrated to be useful in a tree-based algorithm like rpart, may result in being less useful in a regression-based model.Thus, all variables need not be equally useful to all algorithms. One method used to discover the variable importance for a selected machine learning algorithm is to train the desired model using the caret package, then to use varImp() to determine the different level of importance of each feature. The feature importance of each feature of a dataset can also be found by using the feature importance property of the model, particularly for a classifier which is tree-based.

Feature importance provides a score for each feature of your data. The higher the score, the more important or relevant the feature is in achieving your desired output variable. Random forests are a commonly used method for ranking features, being so simple to apply. They usually require a minimum of feature engineering and parameter tuning, and the mean decrease impurity is readily revealed in the majority of random forest libraries. However, caution is advised, as they come with inherent traps for beginners, especially when data interpretation is involved. For example, when features are correlated, the strong features can appear to have low scores; and this method can often be biased towards variables that comprise many categories.

3.3.1 Correlated Predictors

When you are dealing with a model which assumes that dependent variables have a linear relationship the linear relationship between them, the correlation will assist in providing a first-pass, or basic, importance list. This list can also work as an initial draft for models that are nonlinear. The concept here is that features having a high correlation with the dependent variable, are also strong predictors when utilized in a model [6].

In performing regression or classification, it can be seen that some models will perform better when the highly correlated attributes are ignored.

While some models perform well on correlated predictors, other models benefit from reducing or removing the level of correlation between the predictors. Removing correlated predictors is quite useful as they normally contribute quite similarly to the actual prediction. Issues may arise when their dimensions are very different, so it's always wise to normalize before the correlated predictors are deleted.

The main parameter for removing a correlated predictor is the cut-off or minimum correlation between predictors. In our present implementation, below, it can be observed that by using a minimum correlation of 0.8, 67 predictors were removed (or 1.6% of the predictors). Note, that as the minimum correlation increases, that the above figure will quickly decrease. The findCorrelation function, provided by the caret package, can find the attributes that are highly correlated with each other. In this experiment, we demonstrated how highly correlated features are found by using the caret package.

Code

```
## [1] "Variables set to be remove using cutoff 0.8"
```

Code

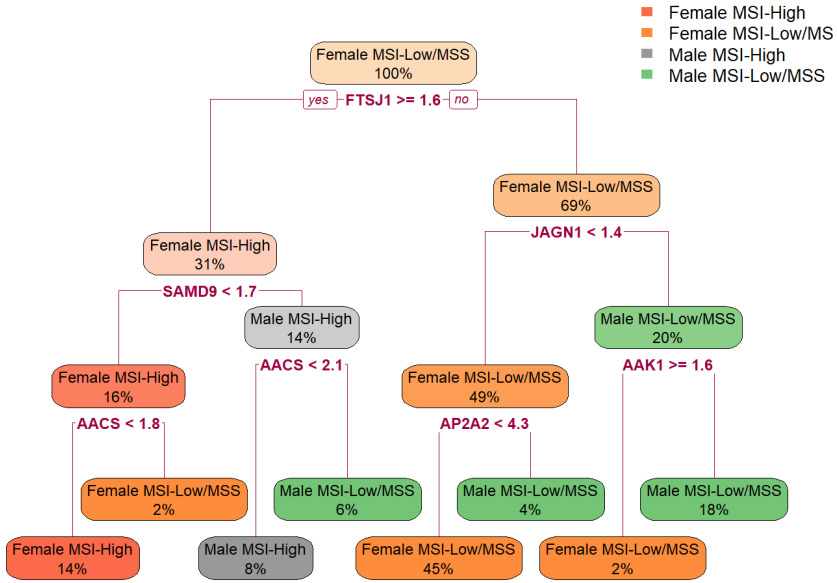
```
## [1] 0.9444444
```

3.3.2 Feature Importance Rpart

Decision trees and random forests have long been a fundamental part of the machine learning toolbox, especially for their accuracy and robustness. As well as having powerful predictive accuracy, they come with feature importance measures that are commonly utilised in applications where model interpretability is a primary necessity. The importance scores are used for model selection. Predictors with high-ranking scores may be selected for further analysis, or for building a more frugal model. Rpart uses a greedy feature-selection algorithm, which trains a decision tree and clips the features from the tree, beginning at the root and moving toward the leaves. It builds a new tree without the original features, and the best features from this tree are similarly removed. This process is reiterated until sufficient features have been discovered or until the tree cannot split the data any further [7].

Recursive Partitioning and Regression Trees: Rpart, is a package which implements classification trees one by one, between growing and pruning trees. As the trees are being fitted, Rpart accepts the discovery of feature importance and helps to decides how many are important in predicting each of the multiple classes.Rpart is a very welcome implementation, for its interpretability and the easy way it provides the visualisation of predictors and their importance. In the figure, below, we plot the classification tree. This can be easily understood with respect to its main predictors.

Code



Code

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

3.3.3 Feature Selection (RF)

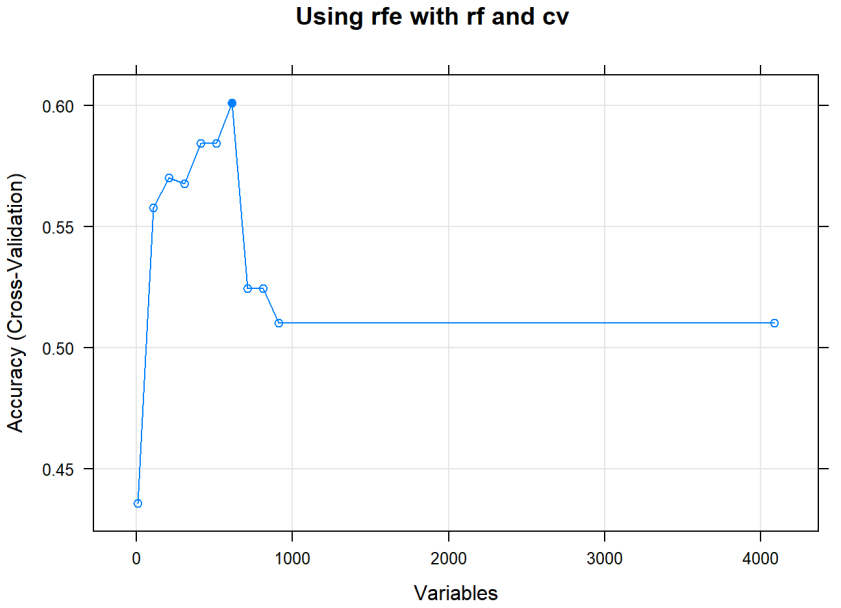
Random Forest is often used for prediction. However, when looking at feature importances you can get the impression of which variables have the greatest impact on the models. New features can be created from this information, as well as the ability to eliminate features with the appearance of noise, or just to inform one to continue building models [8].

A number of decision trees comprises Random forest. Each decision tree node is a condition of a single feature. This is planned to divide the dataset into two parts with like response values appearing in the same set. 'Impurity' is a measure based on the way locally optimal condition is chosen.

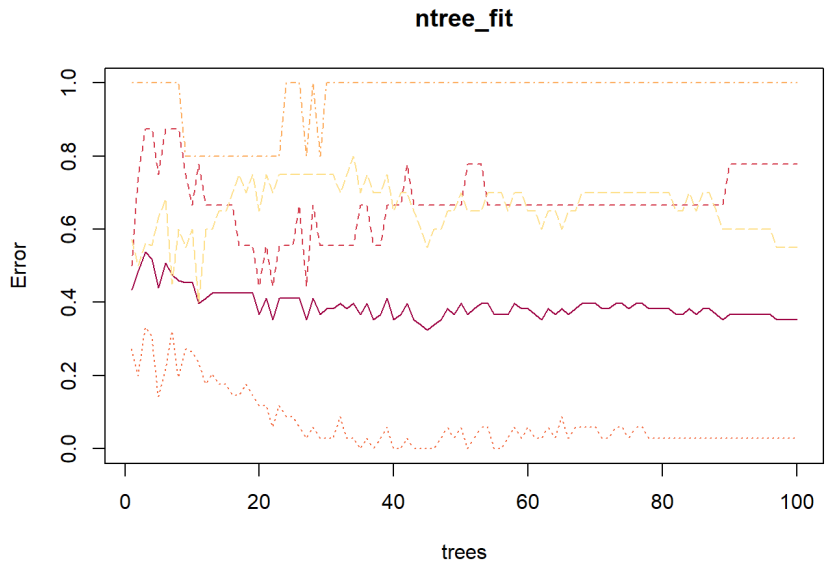
Code

```
## The number of features selected by wrapper method is: 610
```

Code



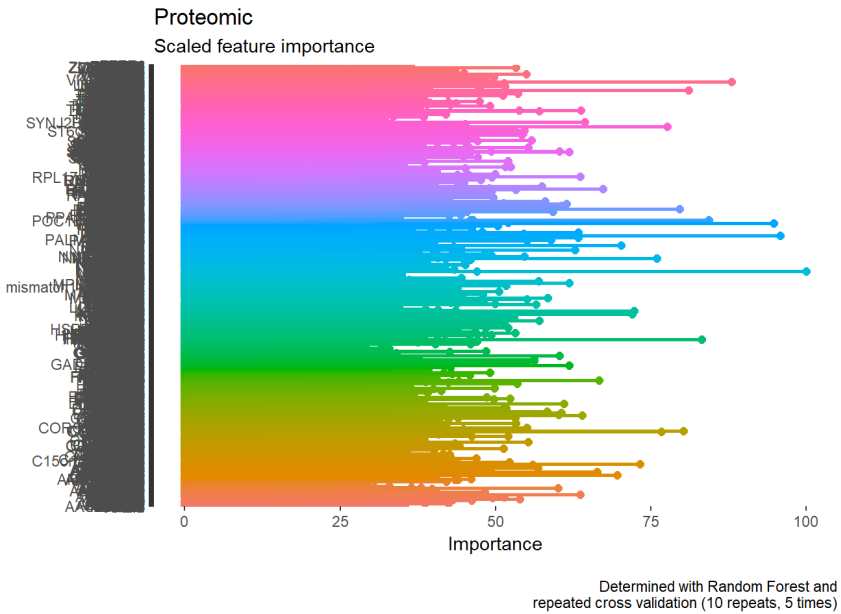
Code



Code

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

Code



3.3.4 sPLS-DA

Using a single stage process sPLS-DA makes variable selection and classification. To permit variable selection sPLS-DA may be regarded as a limited aspect of sparse PLS. Note that variables are selected in a supervised framework, and only in the X data set. That is, choosing of X-variables done in respect of samples of different classes [10].

The classification performance of sPLS-DA is along the same lines as wrapper and sparse discriminant analysis techniques found in SNP data sets and on the public microarray. Of key importance, sPLS-DA is superior in terms of interpretability of the results via valuable graphical outputs, as well as being is quite competitive in terms of computational efficiency.sPLS-DA is to be found in the R package mixOmics, which is committed to the analysis of large biological datasets [11].

Code

```
## Warning: package 'mixOmics' was built under R version 3.5.1
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

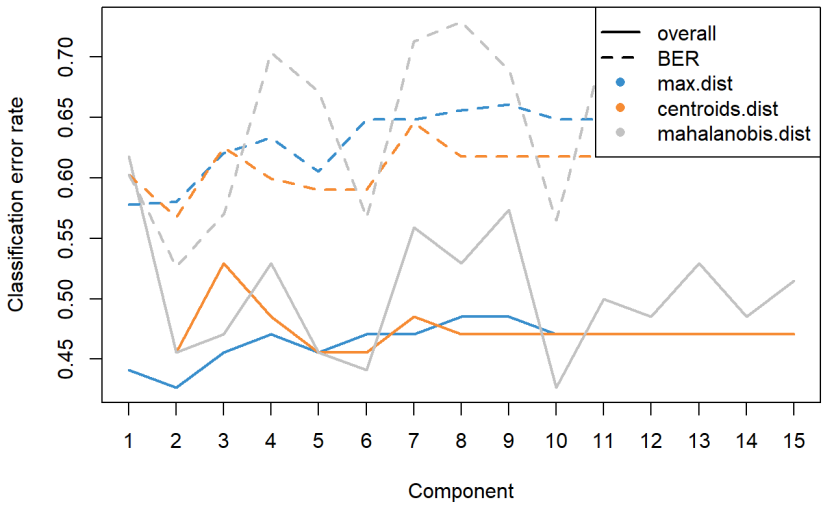
```
##
## Loaded mixOmics 6.3.2
##
## Thank you for using mixOmics!
##
## How to apply our methods: http://www.mixOmics.org for some examples.
## Questions or comments: email us at mixomics[at]math.univ-toulouse.fr
## Any bugs? https://bitbucket.org/klecao/package-mixomics/issues
## Cite us:  citation('mixOmics')
```

```
##
## Attaching package: 'mixOmics'
```

```
## The following objects are masked from 'package:caret':
##
##     nearZeroVar, plsda, splsda
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

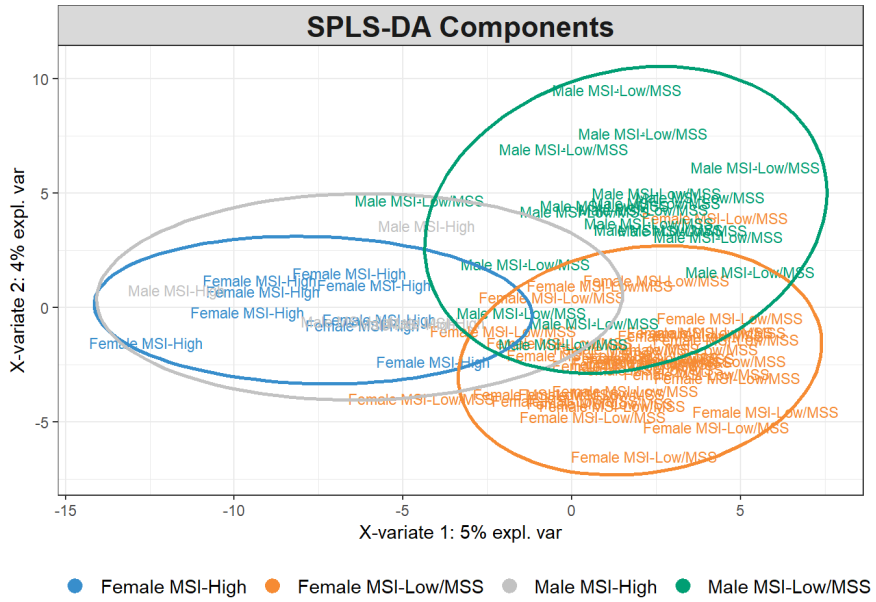
Code



Code

```
## comp1 comp2
##    70    90
```

Code



Code


```
## [1] 0.9444444
```

3.3.5 PCA

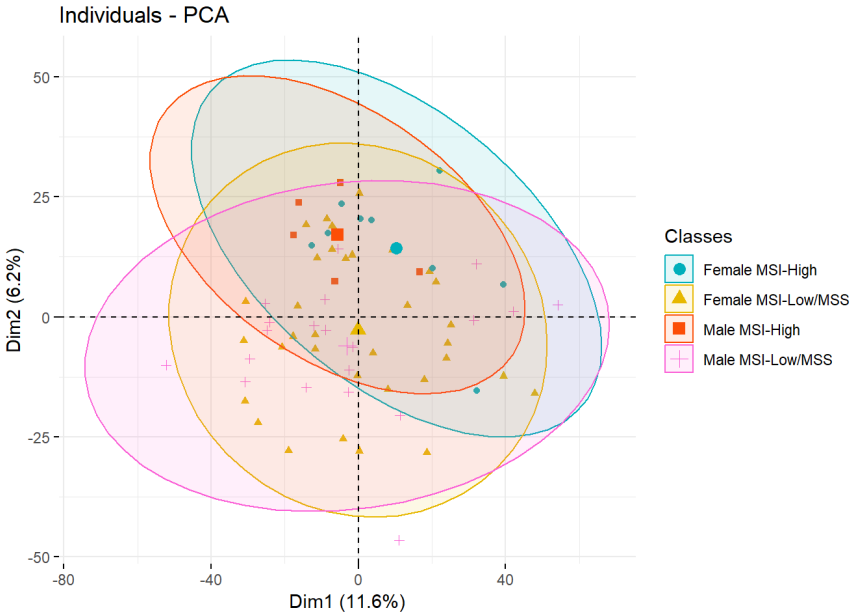
To get an idea about the dimensionality and variance of the datasets, PCA is run agaist the updampled knn-imputed data. The first two principal components (PCs) show the two components that explain the majority of variation in the data. PCA reduces the dimensionality while explaining most of the variability, but there is a more technical method for measuring exactly what percentage of the variance was retained in these principal components. the proportion of variance explained (PVE) by the mth principal component is calculated using the equation:

$$PVE = \frac{\sum_{i=1}^{n}(\sum_{j=1}^{p}\phi_{jm}x_{ij})^2}{\sum_{j=1}^{p}\sum_{i=1}^{n}x_{ij}^2}$$

The most common technique for determining how many principal components to keep is eyeballing the scree plot, which is the left-hand plot shown above and stored in the ggplot object PVEplot. To determine the number of components, we look for the “elbow point”, where the PVE significantly drops off.

PCA is one of the most powerful dimensionality reduction algorithms as it takes care of a lot of the issues found above: * Each principal component is the best principal component. So it can achieve the best model with the least number of features. * Even though its principal components cannot be reinterpretable to the original dimensions, the fact that to each component there is an “explanation quotient” makes it nice to understand model compression.

Code



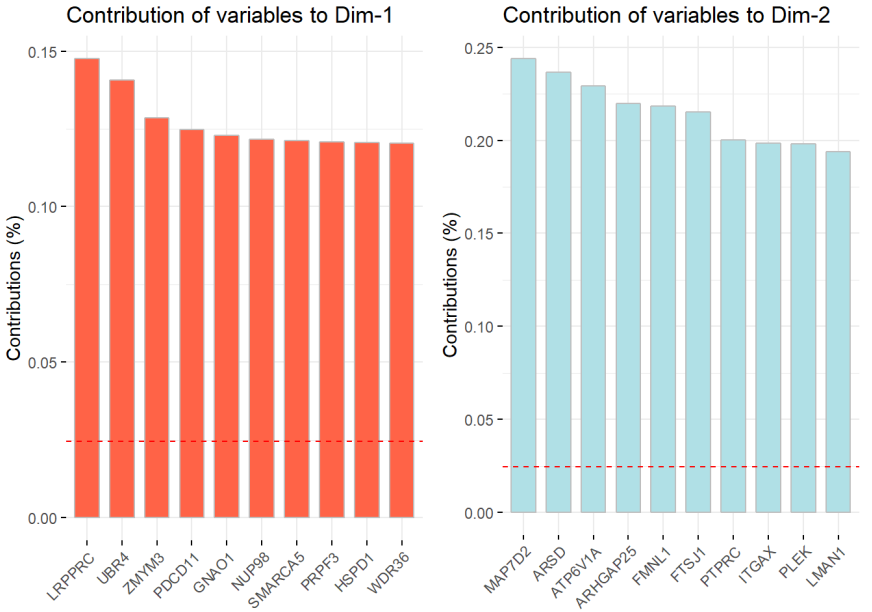
Code

```
## Importance of components:
##                           PC1       PC2       PC3       PC4       PC5
## Standard deviation     21.7966 15.85297 14.69657 12.73246 11.59867
## Proportion of Variance  0.1163  0.06152  0.05287  0.03969  0.03293
## Cumulative Proportion   0.1163  0.17782  0.23070  0.27038  0.30332
##                           PC6       PC7       PC8       PC9      PC10      PC11
## Standard deviation     10.87893 9.73236 9.55563 9.01777 8.75245 8.61418
## Proportion of Variance  0.02897 0.02319 0.02235 0.01991 0.01875 0.01817
## Cumulative Proportion   0.33229 0.35547 0.37783 0.39773 0.41649 0.43465
##                           PC12      PC13      PC14      PC15      PC16      PC17
## Standard deviation     8.33046 8.21692 8.11454 7.95943 7.8791 7.74712
## Proportion of Variance 0.01699 0.01653 0.01612 0.01551 0.0152 0.01469
## Cumulative Proportion  0.45164 0.46817 0.48429 0.49980 0.5150 0.52968
##                           PC18      PC19      PC20      PC21      PC22      PC23
## Standard deviation     7.55435 7.41503 7.33386 7.26292 7.17128 7.13567
## Proportion of Variance 0.01397 0.01346 0.01317 0.01291 0.01259 0.01246
## Cumulative Proportion  0.54366 0.55711 0.57028 0.58319 0.59578 0.60825
##                           PC24      PC25      PC26      PC27      PC28      PC29
## Standard deviation     7.05563 7.04317 6.92901 6.90578 6.87159 6.80123
## Proportion of Variance 0.01219 0.01214 0.01175 0.01167 0.01156 0.01132
## Cumulative Proportion  0.62043 0.63258 0.64433 0.65601 0.66756 0.67889
##                           PC30      PC31      PC32      PC33      PC34      PC35
## Standard deviation     6.71492 6.69855 6.63239 6.55180 6.5192 6.4538
## Proportion of Variance 0.01104 0.01098 0.01077 0.01051 0.0104 0.0102
## Cumulative Proportion  0.68993 0.70091 0.71168 0.72219 0.7326 0.7428
##                           PC36      PC37      PC38      PC39      PC40      PC41
## Standard deviation     6.39930 6.34858 6.3262 6.30564 6.22737 6.20026
## Proportion of Variance 0.01002 0.00987 0.0098 0.00973 0.00949 0.00941
## Cumulative Proportion  0.75281 0.76268 0.7725 0.78221 0.79170 0.80111
##                           PC42      PC43      PC44      PC45      PC46      PC47
## Standard deviation     6.13318 6.09361 6.06678 6.02080 5.9257 5.86682
## Proportion of Variance 0.00921 0.00909 0.00901 0.00887 0.0086 0.00843
## Cumulative Proportion  0.81032 0.81941 0.82842 0.83730 0.8459 0.85432
##                           PC48      PC49      PC50      PC51      PC52      PC53
## Standard deviation     5.81512 5.79208 5.77984 5.77604 5.72674 5.66317
## Proportion of Variance 0.00828 0.00821 0.00818 0.00817 0.00803 0.00785
## Cumulative Proportion  0.86259 0.87081 0.87899 0.88715 0.89518 0.90303
##                           PC54      PC55      PC56      PC57      PC58      PC59
## Standard deviation     5.61626 5.60256 5.54182 5.50071 5.47023 5.40727
## Proportion of Variance 0.00772 0.00768 0.00752 0.00741 0.00733 0.00716
## Cumulative Proportion  0.91075 0.91844 0.92596 0.93336 0.94069 0.94784
##                           PC60      PC61      PC62      PC63      PC64      PC65
## Standard deviation     5.37289 5.35741 5.26630 5.24636 5.17231 5.1147
## Proportion of Variance 0.00707 0.00703 0.00679 0.00674 0.00655 0.0064
## Cumulative Proportion  0.95491 0.96194 0.96873 0.97546 0.98201 0.9884
##                           PC66      PC67       PC68
## Standard deviation     4.96648 4.75893 2.359e-14
## Proportion of Variance 0.00604 0.00554 0.000e+00
## Cumulative Proportion  0.99446 1.00000 1.000e+00
```

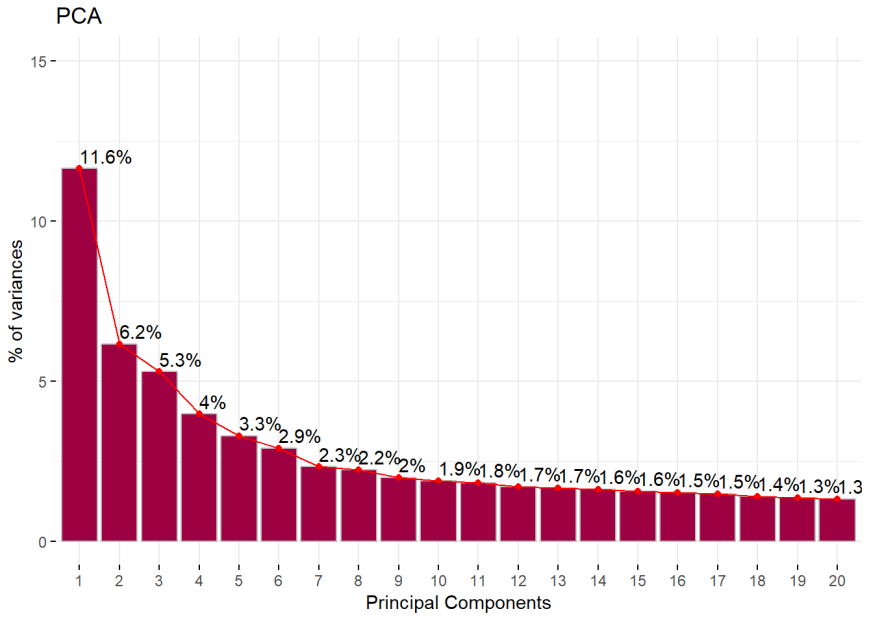
Code


```
## Principal Component Analysis Results for variables
##  ===================================================
##   Name       Description
## 1 "$coord"   "Coordinates for the variables"
## 2 "$cor"     "Correlations between variables and dimensions"
## 3 "$cos2"    "Cos2 for the variables"
## 4 "$contrib" "contributions of the variables"
```

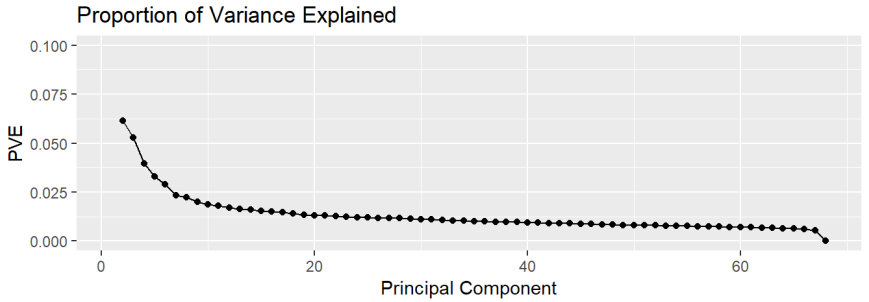
Code

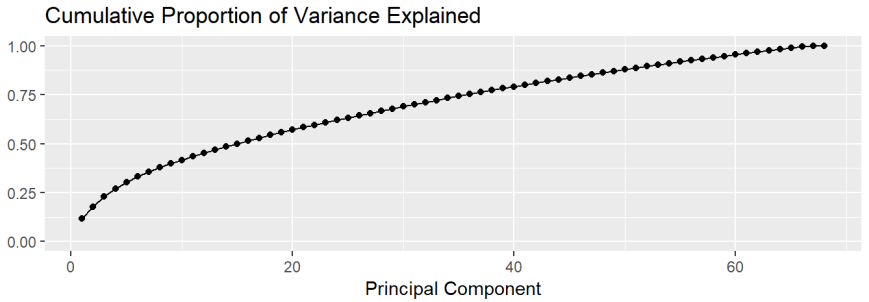


Code



Code



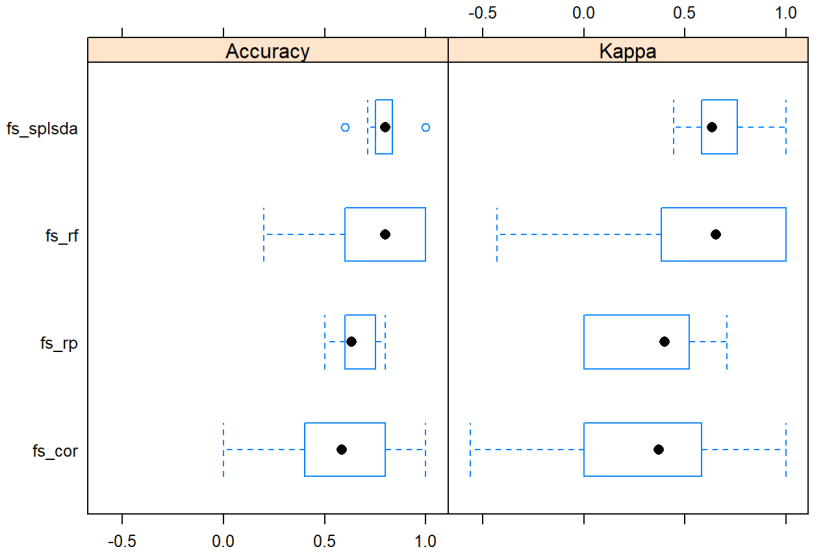


Code

3.3.6 Result

Random Forest Feature Select improves the prediction

Code



Code

```
## [1] "cm_fs_cor"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     0.5000000   1.0000000      1.0000000
## Class: Female MSI-Low/MSS  0.8000000   0.7777778      0.8000000
## Class: Male MSI-High       1.0000000   0.9444444      0.5000000
## Class: Male MSI-Low/MSS    0.6666667   0.8461538      0.6666667
##                          Neg Pred Value Precision    Recall        F1
## Class: Female MSI-High        0.9444444 1.0000000 0.5000000 0.6666667
## Class: Female MSI-Low/MSS     0.7777778 0.8000000 0.8000000 0.8000000
## Class: Male MSI-High          1.0000000 0.5000000 1.0000000 0.6666667
## Class: Male MSI-Low/MSS       0.8461538 0.6666667 0.6666667 0.6666667
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316      0.05263158           0.05263158
## Class: Female MSI-Low/MSS 0.52631579      0.42105263           0.52631579
## Class: Male MSI-High      0.05263158      0.05263158           0.10526316
## Class: Male MSI-Low/MSS   0.31578947      0.21052632           0.31578947
##                          Balanced Accuracy
## Class: Female MSI-High           0.7500000
## Class: Female MSI-Low/MSS        0.7888889
## Class: Male MSI-High             0.9722222
## Class: Male MSI-Low/MSS          0.7564103
## [1] "cm_fs_rp"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     0.5000000   0.8823529      0.3333333
## Class: Female MSI-Low/MSS  0.9000000   0.4444444      0.6428571
## Class: Male MSI-High       0.0000000   1.0000000            NaN
## Class: Male MSI-Low/MSS    0.3333333   1.0000000      1.0000000
##                          Neg Pred Value Precision    Recall   F1
## Class: Female MSI-High        0.9375000 0.3333333 0.5000000 0.40
## Class: Female MSI-Low/MSS     0.8000000 0.6428571 0.9000000 0.75
## Class: Male MSI-High          0.9473684        NA 0.0000000   NA
## Class: Male MSI-Low/MSS       0.7647059 1.0000000 0.3333333 0.50
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316      0.05263158           0.1578947
## Class: Female MSI-Low/MSS 0.52631579      0.47368421           0.7368421
## Class: Male MSI-High      0.05263158      0.00000000           0.0000000
## Class: Male MSI-Low/MSS   0.31578947      0.10526316           0.1052632
##                          Balanced Accuracy
## Class: Female MSI-High           0.6911765
## Class: Female MSI-Low/MSS        0.6722222
## Class: Male MSI-High             0.5000000
## Class: Male MSI-Low/MSS          0.6666667
## [1] "cm_fs_rf"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High           0.5   0.9411765           0.50
## Class: Female MSI-Low/MSS        0.9   0.6666667           0.75
## Class: Male MSI-High             0.0   1.0000000            NaN
## Class: Male MSI-Low/MSS          0.5   0.8461538           0.60
##                          Neg Pred Value Precision Recall        F1
## Class: Female MSI-High        0.9411765       0.50    0.5 0.5000000
## Class: Female MSI-Low/MSS     0.8571429       0.75    0.9 0.8181818
## Class: Male MSI-High          0.9473684         NA    0.0        NA
## Class: Male MSI-Low/MSS       0.7857143       0.60    0.5 0.5454545
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316      0.05263158           0.1052632
## Class: Female MSI-Low/MSS 0.52631579      0.47368421           0.6315789
## Class: Male MSI-High      0.05263158      0.00000000           0.0000000
```



```
## Class: Male MSI-Low/MSS   0.31578947      0.15789474           0.2631579
##                          Balanced Accuracy
## Class: Female MSI-High           0.7205882
## Class: Female MSI-Low/MSS        0.7833333
## Class: Male MSI-High             0.5000000
## Class: Male MSI-Low/MSS          0.6730769
## [1] "cm_fs_splsda"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     1.0000000   0.9411765      0.6666667
## Class: Female MSI-Low/MSS  1.0000000   0.6666667      0.7692308
## Class: Male MSI-High       0.0000000   0.9444444      0.0000000
## Class: Male MSI-Low/MSS    0.3333333   1.0000000      1.0000000
##                          Neg Pred Value Precision    Recall        F1
## Class: Female MSI-High        1.0000000 0.6666667 1.0000000 0.8000000
## Class: Female MSI-Low/MSS     1.0000000 0.7692308 1.0000000 0.8695652
## Class: Male MSI-High          0.9444444 0.0000000 0.0000000       NaN
## Class: Male MSI-Low/MSS       0.7647059 1.0000000 0.3333333 0.5000000
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316      0.1052632           0.15789474
## Class: Female MSI-Low/MSS 0.52631579      0.5263158           0.68421053
## Class: Male MSI-High      0.05263158      0.0000000           0.05263158
## Class: Male MSI-Low/MSS   0.31578947      0.1052632           0.10526316
##                          Balanced Accuracy
## Class: Female MSI-High           0.9705882
## Class: Female MSI-Low/MSS        0.8333333
## Class: Male MSI-High             0.4722222
## Class: Male MSI-Low/MSS          0.6666667
```

4 Train Model

Insert code to preload saved preprocessed data. Depending on the outcome from step 4, we will pick the best imputation algorithm that gives the best F1 result to train the model.

Code

4.1 rpart

Code

```
## CART
##
##  68 samples
## 610 predictors
##   4 classes: 'Female MSI-High', 'Female MSI-Low/MSS', 'Male MSI-High', 'Male MSI-Low/MS
S'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 62, 61, 61, 62, 61, 62, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.4297619   0.072023751
##   0.02614379  0.4297619   0.072023751
##   0.05228758  0.4297619   0.062023751
##   0.07843137  0.4464286   0.083023751
##   0.10457516  0.4464286   0.066242493
##   0.13071895  0.4464286   0.066242493
##   0.15686275  0.4214286   0.036648058
##   0.18300654  0.4196429  -0.004155564
##   0.20915033  0.4607143   0.007443182
##   0.23529412  0.4750000  -0.023333333
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.2352941.
```

Code

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

Code


```
## CART
##
##  68 samples
## 160 predictors
##   4 classes: 'Female MSI-High', 'Female MSI-Low/MSS', 'Male MSI-High', 'Male MSI-Low/MS
S'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 62, 61, 61, 62, 61, 62, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.3452381  -0.057191286
##   0.02614379  0.3452381  -0.057191286
##   0.05228758  0.3452381  -0.062039771
##   0.07843137  0.3452381  -0.062039771
##   0.10457516  0.3452381  -0.076325485
##   0.13071895  0.3595238  -0.057094716
##   0.15686275  0.3928571  -0.007094716
##   0.18300654  0.4196429   0.004352219
##   0.20915033  0.4464286  -0.039379975
##   0.23529412  0.4750000  -0.033333333
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.2352941.
```

Code

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

4.2 Random Forest

Code

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

Code

```
## Random Forest
##
##  68 samples
## 160 predictors
##   4 classes: 'Female MSI-High', 'Female MSI-Low/MSS', 'Male MSI-High', 'Male MSI-Low/MS
S'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 62, 61, 61, 62, 61, 62, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.7833333  0.6292413
##    4    0.7666667  0.5965746
##    6    0.7523810  0.5763912
##    8    0.7541667  0.5712788
##   10    0.7541667  0.5695520
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

Code

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

4.3 KNN

Code

```
## k-Nearest Neighbors
##
##  68 samples
## 610 predictors
##   4 classes: 'Female MSI-High', 'Female MSI-Low/MSS', 'Male MSI-High', 'Male MSI-Low/MS
S'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 62, 61, 61, 62, 61, 62, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.7107143  0.5427866
##   7  0.7785714  0.6305469
##   9  0.7619048  0.6019754
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 7.
```

Code

```
## [1] 0.9722222
```

Code


```
## k-Nearest Neighbors
##
##  68 samples
## 160 predictors
##   4 classes: 'Female MSI-High', 'Female MSI-Low/MSS', 'Male MSI-High', 'Male MSI-Low/MS
S'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 62, 61, 61, 62, 61, 62, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.7190476  0.5589802
##   7  0.7065476  0.5362403
##   9  0.7809524  0.6434440
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

Code

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

4.4 SVM

Code

```
## Support Vector Machines with Radial Basis Function Kernel
##
##  68 samples
## 610 predictors
##   4 classes: 'Female MSI-High', 'Female MSI-Low/MSS', 'Male MSI-High', 'Male MSI-Low/MS
S'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 62, 61, 61, 62, 61, 62, ...
## Resampling results across tuning parameters:
##
##   C     Accuracy  Kappa
##   0.25  0.500000  0.0000000
##   0.50  0.500000  0.0000000
##   1.00  0.685119  0.4307587
##
## Tuning parameter 'sigma' was held constant at a value of 0.0008899526
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.0008899526 and C = 1.
```

Code

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

Code

```
## k-Nearest Neighbors
##
##  68 samples
## 160 predictors
##   4 classes: 'Female MSI-High', 'Female MSI-Low/MSS', 'Male MSI-High', 'Male MSI-Low/MS
S'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 62, 61, 61, 62, 61, 62, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.7190476  0.5589802
##   7  0.7065476  0.5362403
##   9  0.7809524  0.6434440
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

Code

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

5 Result

5.1 Random Forest Variable Importance

Code

```
##
## Call:
## resamples.default(x = model_list)
##
## Models: KNN, SVM, RF, RP
## Number of resamples: 10
## Performance metrics: Accuracy, Kappa
## Time estimates for: everything, final model fit
```

Code

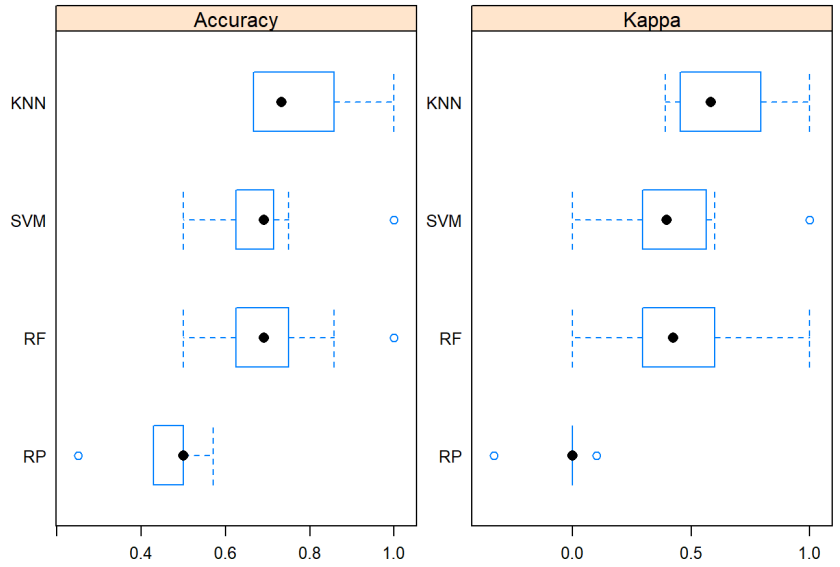

```
## [1] "cm_fs_cor"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     0.5000000   1.0000000       1.0000000
## Class: Female MSI-Low/MSS  0.8000000   0.7777778       0.8000000
## Class: Male MSI-High       1.0000000   0.9444444       0.5000000
## Class: Male MSI-Low/MSS    0.6666667   0.8461538       0.6666667
##                          Neg Pred Value Precision    Recall        F1
## Class: Female MSI-High        0.9444444 1.0000000 0.5000000 0.6666667
## Class: Female MSI-Low/MSS     0.7777778 0.8000000 0.8000000 0.8000000
## Class: Male MSI-High          1.0000000 0.5000000 1.0000000 0.6666667
## Class: Male MSI-Low/MSS       0.8461538 0.6666667 0.6666667 0.6666667
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316     0.05263158            0.05263158
## Class: Female MSI-Low/MSS 0.52631579     0.42105263            0.52631579
## Class: Male MSI-High      0.05263158     0.05263158            0.10526316
## Class: Male MSI-Low/MSS   0.31578947     0.21052632            0.31578947
##                          Balanced Accuracy
## Class: Female MSI-High           0.7500000
## Class: Female MSI-Low/MSS        0.7888889
## Class: Male MSI-High             0.9722222
## Class: Male MSI-Low/MSS          0.7564103
## [1] "cm_fs_rp"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     0.5000000   0.8823529       0.3333333
## Class: Female MSI-Low/MSS  0.9000000   0.4444444       0.6428571
## Class: Male MSI-High       0.0000000   1.0000000             NaN
## Class: Male MSI-Low/MSS    0.3333333   1.0000000       1.0000000
##                          Neg Pred Value Precision    Recall   F1
## Class: Female MSI-High        0.9375000 0.3333333 0.5000000 0.40
## Class: Female MSI-Low/MSS     0.8000000 0.6428571 0.9000000 0.75
## Class: Male MSI-High          0.9473684        NA 0.0000000   NA
## Class: Male MSI-Low/MSS       0.7647059 1.0000000 0.3333333 0.50
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316     0.05263158            0.1578947
## Class: Female MSI-Low/MSS 0.52631579     0.47368421            0.7368421
## Class: Male MSI-High      0.05263158     0.00000000            0.0000000
## Class: Male MSI-Low/MSS   0.31578947     0.10526316            0.1052632
##                          Balanced Accuracy
## Class: Female MSI-High           0.6911765
## Class: Female MSI-Low/MSS        0.6722222
## Class: Male MSI-High             0.5000000
## Class: Male MSI-Low/MSS          0.6666667
## [1] "cm_fs_rf"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High           0.5   0.9411765            0.50
## Class: Female MSI-Low/MSS        0.9   0.6666667            0.75
## Class: Male MSI-High             0.0   1.0000000             NaN
## Class: Male MSI-Low/MSS          0.5   0.8461538            0.60
##                          Neg Pred Value Precision Recall        F1
## Class: Female MSI-High        0.9411765       0.50    0.5 0.5000000
## Class: Female MSI-Low/MSS     0.8571429       0.75    0.9 0.8181818
## Class: Male MSI-High          0.9473684         NA    0.0        NA
## Class: Male MSI-Low/MSS       0.7857143       0.60    0.5 0.5454545
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316     0.05263158            0.1052632
## Class: Female MSI-Low/MSS 0.52631579     0.47368421            0.6315789
## Class: Male MSI-High      0.05263158     0.00000000            0.0000000
```

```
## Class: Male MSI-Low/MSS   0.31578947      0.15789474             0.2631579
##                          Balanced Accuracy
## Class: Female MSI-High             0.7205882
## Class: Female MSI-Low/MSS          0.7833333
## Class: Male MSI-High               0.5000000
## Class: Male MSI-Low/MSS            0.6730769
## [1] "cm_fs_splsda"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     1.0000000   0.9411765      0.6666667
## Class: Female MSI-Low/MSS  1.0000000   0.6666667      0.7692308
## Class: Male MSI-High       0.0000000   0.9444444      0.0000000
## Class: Male MSI-Low/MSS    0.3333333   1.0000000      1.0000000
##                          Neg Pred Value Precision    Recall        F1
## Class: Female MSI-High        1.0000000 0.6666667 1.0000000 0.8000000
## Class: Female MSI-Low/MSS     1.0000000 0.7692308 1.0000000 0.8695652
## Class: Male MSI-High          0.9444444 0.0000000 0.0000000       NaN
## Class: Male MSI-Low/MSS       0.7647059 1.0000000 0.3333333 0.5000000
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High   0.10526316       0.1052632            0.15789474
## Class: Female MSI-Low/MSS 0.52631579       0.5263158            0.68421053
## Class: Male MSI-High      0.05263158       0.0000000            0.05263158
## Class: Male MSI-Low/MSS   0.31578947       0.1052632            0.10526316
##                          Balanced Accuracy
## Class: Female MSI-High             0.9705882
## Class: Female MSI-Low/MSS          0.8333333
## Class: Male MSI-High               0.4722222
## Class: Male MSI-Low/MSS            0.6666667
```

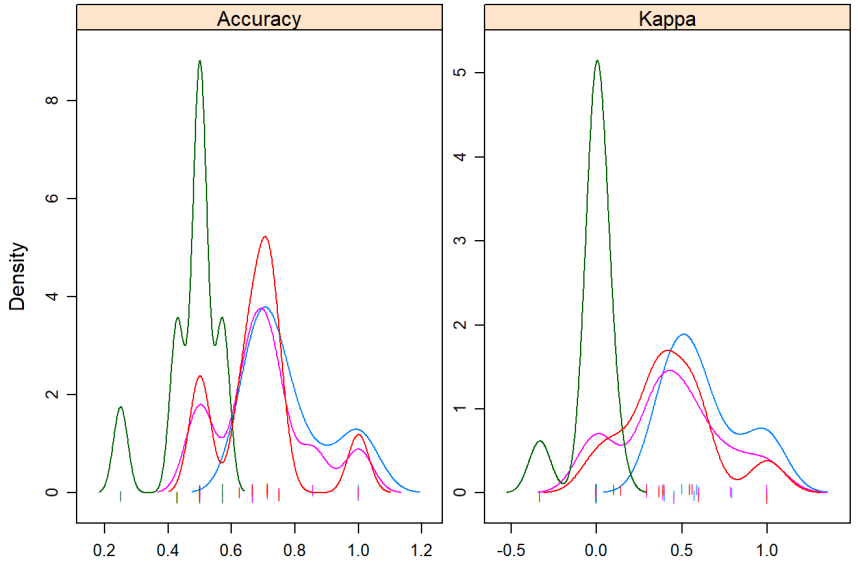
Code

```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: KNN, SVM, RF, RP
## Number of resamples: 10
##
## Accuracy
##          Min.    1st Qu.     Median      Mean   3rd Qu.      Max. NA's
## KNN 0.6666667 0.6785714 0.7321429 0.7785714 0.8303571 1.0000000    0
## SVM 0.5000000 0.6354167 0.6904762 0.6851190 0.7142857 1.0000000    0
## RF  0.5000000 0.6354167 0.6904762 0.6994048 0.7410714 1.0000000    0
## RP  0.2500000 0.4464286 0.5000000 0.4750000 0.5000000 0.5714286    0
##
## Kappa
##           Min.    1st Qu.     Median        Mean   3rd Qu. Max. NA's
## KNN  0.3913043 0.4659091 0.5827506  0.63054686 0.7455882  1.0    0
## SVM  0.0000000 0.3126935 0.3956522  0.43075873 0.5589718  1.0    0
## RF   0.0000000 0.3126935 0.4229249  0.44446544 0.5870968  1.0    0
## RP  -0.3333333 0.0000000 0.0000000 -0.02333333 0.0000000  0.1    0
```

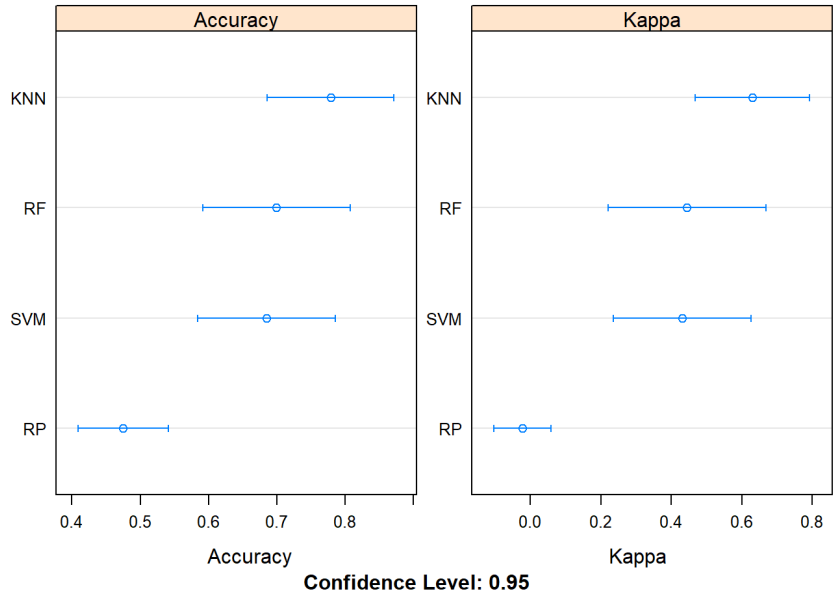
Code



Code



Code



Code

```
##
## Call:
## summary.diff.resamples(object = diffs)
##
## p-value adjustment: bonferroni
## Upper diagonal: estimates of the difference
## Lower diagonal: p-value for H0: difference = 0
##
## Accuracy
##     KNN        SVM        RF         RP
## KNN             0.09345    0.07917    0.30357
## SVM 0.1695718              -0.01429    0.21012
## RF  0.9141620 1.0000000               0.22440
## RP  0.0006353 0.0073113 0.0090220
##
## Kappa
##     KNN        SVM       RF         RP
## KNN             0.19979   0.18608   0.65388
## SVM 0.088894              -0.01371   0.45409
## RF  0.584909   1.000000             0.46780
## RP  5.268e-05 0.003771 0.009530
```

Code

5.2 sPLSDA

Code


```
##
## Call:
## resamples.default(x = model_list)
##
## Models: KNN, SVM, RF, RP
## Number of resamples: 10
## Performance metrics: Accuracy, Kappa
## Time estimates for: everything, final model fit
```

Code

```
## [1] "cm_fs_cor"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     0.5000000   1.0000000      1.0000000
## Class: Female MSI-Low/MSS  0.8000000   0.7777778      0.8000000
## Class: Male MSI-High       1.0000000   0.9444444      0.5000000
## Class: Male MSI-Low/MSS    0.6666667   0.8461538      0.6666667
##                          Neg Pred Value Precision    Recall        F1
## Class: Female MSI-High        0.9444444 1.0000000 0.5000000 0.6666667
## Class: Female MSI-Low/MSS     0.7777778 0.8000000 0.8000000 0.8000000
## Class: Male MSI-High          1.0000000 0.5000000 1.0000000 0.6666667
## Class: Male MSI-Low/MSS       0.8461538 0.6666667 0.6666667 0.6666667
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316      0.05263158           0.05263158
## Class: Female MSI-Low/MSS 0.52631579      0.42105263           0.52631579
## Class: Male MSI-High      0.05263158      0.05263158           0.10526316
## Class: Male MSI-Low/MSS   0.31578947      0.21052632           0.31578947
##                          Balanced Accuracy
## Class: Female MSI-High           0.7500000
## Class: Female MSI-Low/MSS        0.7888889
## Class: Male MSI-High             0.9722222
## Class: Male MSI-Low/MSS          0.7564103
## [1] "cm_fs_rp"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     0.5000000   0.8823529      0.3333333
## Class: Female MSI-Low/MSS  0.9000000   0.4444444      0.6428571
## Class: Male MSI-High       0.0000000   1.0000000            NaN
## Class: Male MSI-Low/MSS    0.3333333   1.0000000      1.0000000
##                          Neg Pred Value Precision    Recall   F1
## Class: Female MSI-High        0.9375000 0.3333333 0.5000000 0.40
## Class: Female MSI-Low/MSS     0.8000000 0.6428571 0.9000000 0.75
## Class: Male MSI-High          0.9473684        NA 0.0000000   NA
## Class: Male MSI-Low/MSS       0.7647059 1.0000000 0.3333333 0.50
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316      0.05263158           0.1578947
## Class: Female MSI-Low/MSS 0.52631579      0.47368421           0.7368421
## Class: Male MSI-High      0.05263158      0.00000000           0.0000000
## Class: Male MSI-Low/MSS   0.31578947      0.10526316           0.1052632
##                          Balanced Accuracy
## Class: Female MSI-High           0.6911765
## Class: Female MSI-Low/MSS        0.6722222
## Class: Male MSI-High             0.5000000
## Class: Male MSI-Low/MSS          0.6666667
## [1] "cm_fs_rf"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High           0.5   0.9411765           0.50
## Class: Female MSI-Low/MSS        0.9   0.6666667           0.75
## Class: Male MSI-High             0.0   1.0000000            NaN
## Class: Male MSI-Low/MSS          0.5   0.8461538           0.60
##                          Neg Pred Value Precision Recall        F1
## Class: Female MSI-High        0.9411765       0.50    0.5 0.5000000
## Class: Female MSI-Low/MSS     0.8571429       0.75    0.9 0.8181818
## Class: Male MSI-High          0.9473684         NA    0.0        NA
## Class: Male MSI-Low/MSS       0.7857143       0.60    0.5 0.5454545
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316      0.05263158           0.1052632
## Class: Female MSI-Low/MSS 0.52631579      0.47368421           0.6315789
## Class: Male MSI-High      0.05263158      0.00000000           0.0000000
```

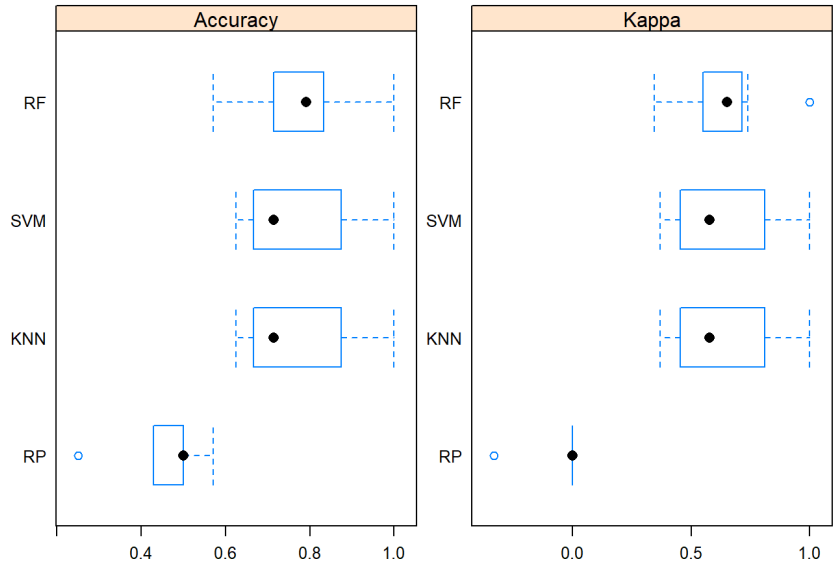


```
## Class: Male MSI-Low/MSS   0.31578947      0.15789474           0.2631579
##                          Balanced Accuracy
## Class: Female MSI-High           0.7205882
## Class: Female MSI-Low/MSS        0.7833333
## Class: Male MSI-High             0.5000000
## Class: Male MSI-Low/MSS          0.6730769
## [1] "cm_fs_splsda"
##                          Sensitivity Specificity Pos Pred Value
## Class: Female MSI-High     1.0000000   0.9411765      0.6666667
## Class: Female MSI-Low/MSS  1.0000000   0.6666667      0.7692308
## Class: Male MSI-High       0.0000000   0.9444444      0.0000000
## Class: Male MSI-Low/MSS    0.3333333   1.0000000      1.0000000
##                          Neg Pred Value Precision    Recall        F1
## Class: Female MSI-High        1.0000000 0.6666667 1.0000000 0.8000000
## Class: Female MSI-Low/MSS     1.0000000 0.7692308 1.0000000 0.8695652
## Class: Male MSI-High          0.9444444 0.0000000 0.0000000       NaN
## Class: Male MSI-Low/MSS       0.7647059 1.0000000 0.3333333 0.5000000
##                          Prevalence Detection Rate Detection Prevalence
## Class: Female MSI-High    0.10526316      0.1052632           0.15789474
## Class: Female MSI-Low/MSS 0.52631579      0.5263158           0.68421053
## Class: Male MSI-High      0.05263158      0.0000000           0.05263158
## Class: Male MSI-Low/MSS   0.31578947      0.1052632           0.10526316
##                          Balanced Accuracy
## Class: Female MSI-High           0.9705882
## Class: Female MSI-Low/MSS        0.8333333
## Class: Male MSI-High             0.4722222
## Class: Male MSI-Low/MSS          0.6666667
```

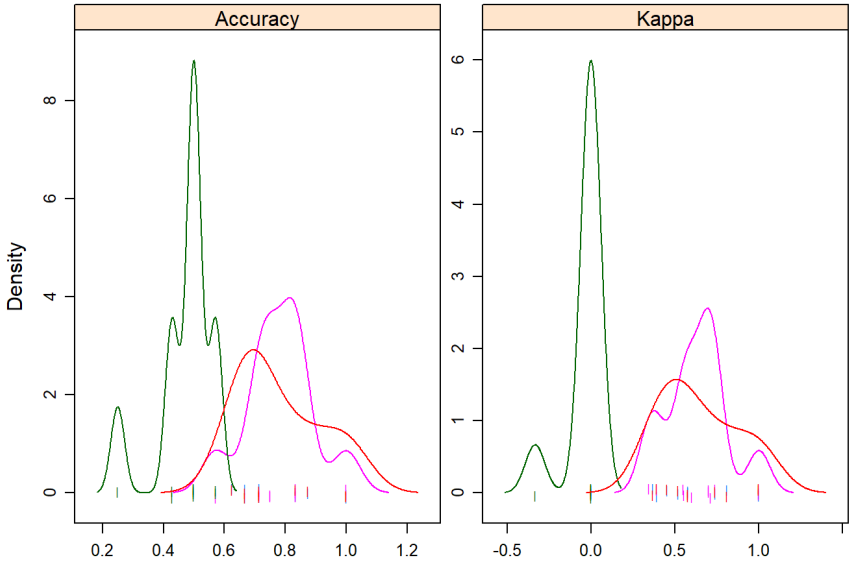
Code

```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: KNN, SVM, RF, RP
## Number of resamples: 10
##
## Accuracy
##          Min.    1st Qu.     Median       Mean    3rd Qu.       Max. NA's
## KNN 0.6250000 0.6785714 0.7142857 0.7809524 0.8645833 1.0000000    0
## SVM 0.6250000 0.6785714 0.7142857 0.7809524 0.8645833 1.0000000    0
## RF  0.5714286 0.7232143 0.7916667 0.7833333 0.8333333 1.0000000    0
## RP  0.2500000 0.4464286 0.5000000 0.4750000 0.5000000 0.5714286    0
##
## Kappa
##           Min.    1st Qu.     Median        Mean    3rd Qu.  Max. NA's
## KNN  0.3684211 0.4709091 0.5757576  0.64344403 0.7919255     1    0
## SVM  0.3684211 0.4709091 0.5757576  0.64344403 0.7919255     1    0
## RF   0.3437500 0.5501792 0.6500000  0.62924131 0.7107143     1    0
## RP  -0.3333333 0.0000000 0.0000000 -0.03333333 0.0000000     0    0
```

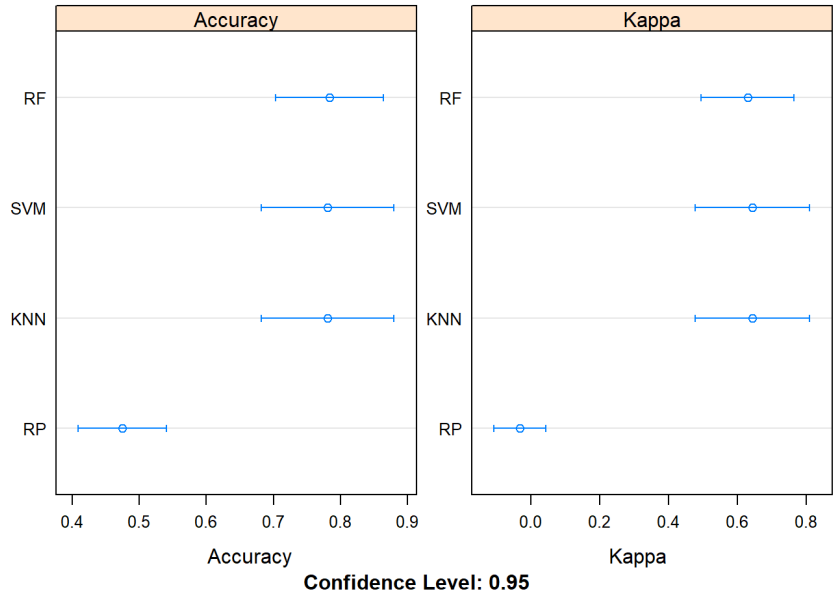
Code



Code



Code



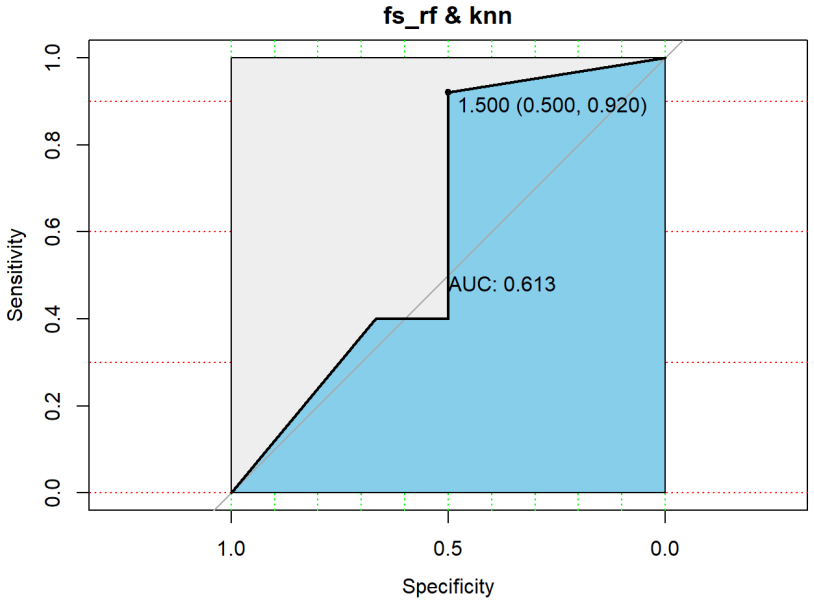
Code

```
##
## Call:
## summary.diff.resamples(object = diffs)
##
## p-value adjustment: bonferroni
## Upper diagonal: estimates of the difference
## Lower diagonal: p-value for H0: difference = 0
##
## Accuracy
##      KNN       SVM       RF         RP
## KNN            0.000000 -0.002381  0.305952
## SVM NA                  -0.002381  0.305952
## RF   1.000000 1.000000             0.308333
## RP   0.001369 0.001369  5.494e-05
##
## Kappa
##      KNN       SVM       RF         RP
## KNN            0.0000    0.0142     0.6768
## SVM NA                   0.0142     0.6768
## RF   1.000000 1.000000             0.6626
## RP   0.000155 0.000155 1.339e-05
```

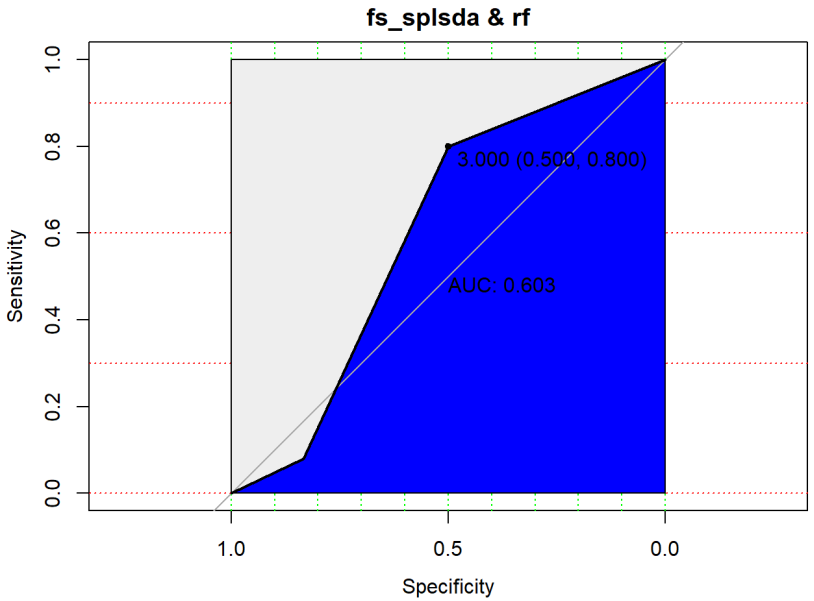
Code

5.3 AUC

Code



Code



Code

KNN(with Random Forest Feature Selection) and RF(with sPLSDA Feature Selection) show promising result,and charts show that KNN(AUC:0.587) is better than RF(AUC:0.537).

6 Others

6.1 Mismatch Labels

The all the above analysis was run using 0-match label. This section will run the same code but on all 80 dataset and try to predict class

Code

```
## [1]  80 162
```

Code

```
## [1]  80 161
```

Code

```
## k-Nearest Neighbors
##
##  80 samples
## 160 predictors
##   4 classes: 'Female MSI-High', 'Female MSI-Low/MSS', 'Male MSI-High', 'Male MSI-Low/MS
S'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 1 times)
## Summary of sample sizes: 73, 71, 71, 71, 72, 73, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.6333333  0.3872695
##   7  0.6962302  0.5023897
##   9  0.6948413  0.4860601
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 7.
```

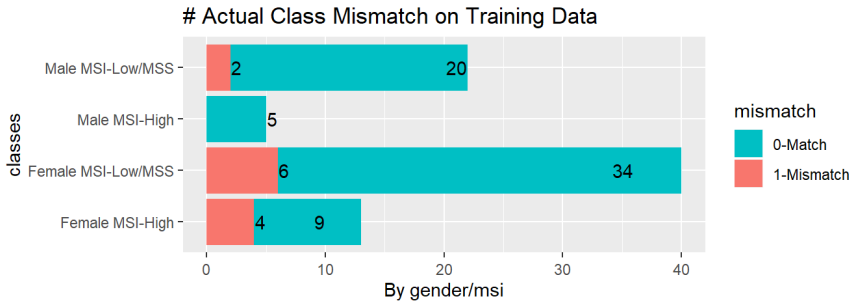
Code

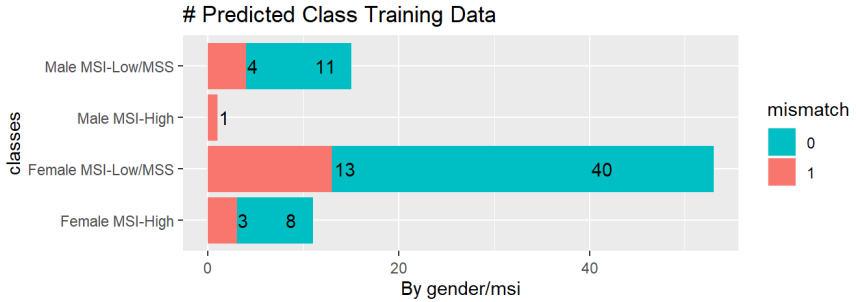
As expected, accuracy dropped slightyly.

6.2 Output files

1. Probability of each sample been mislabelled

Code





7 Evaluation Criteria

The evaluation of any machine learning model or algorithm forms a critical key to achieving a viable outcome for any serious study. The model may provide satisfying results when evaluated using a metric such as accuracy score but may provide unhelpful results when other metrics, such as logarithmic loss or similar, are used. While classification accuracy is commonly used to quantify the performance of our model, it is insufficient to realistically evaluate our model with a high level of confidence.

For this study, we used a variety of different evaluation metrics from the types that were available to us. Classification accuracy is usually what we actually mean when we use accuracy as a shortcut term. It is defined as the ratio of the number of correct predictions to the total number of input samples. This only works well if you have an equal number of samples comprised in each class.

Classification accuracy appears fine but can often give the incorrect impression that a high level of accuracy has been recorded. A significant difficulty arises, when the negative effect of misclassifying of minor class samples peaks. To illustrate the point: when dealing with a rare but usually terminal cancer, the cost of failing to diagnose the precise type of cancer is much higher than the cost undertaking the full range of necessary diagnostic tests. Confusion matrix, as the name indicates, gave us a matrix as output which describe the complete performance of our model. These four important terms are commonly used in confusion matrix -

- True Positives (TP)
- True Negatives (TN)
- False Positives (FP)
- False Negatives (FN)

Area Under Curve (AUC) is a very commonly used evaluation metric, especially for problems with binary classification. The AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example. In defining

the AUC, we need to be aware of these two basic terms: * True Positive Rate (Sensitivity): The True Positive Rate is defined as TP/ (FN+TP). This corresponds with the proportion of positive data points that may be correctly considered as positive, with respect to all of the positive data points. * False Positive Rate (Specificity): The False Positive Rate is defined as FP / (FP+TN). This corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all of the negative data points. Confusion matrix can also be a basis for metrics of other types.

The F1 Score is the Harmonic Mean between precision and recall. The range for the F1 Score is [0,1]. This lets you know the accuracy and precision of your classifier (how many instances are correctly classified), and how solid and robust it is (a significant number of instances are not ignored).

Having high precision but lower recall gives you an extremely accurate result. However, it can then miss very many instances that were difficult to classify. The higher the F1 Score, the greater the performance of our model. F1 Score aims to find a balance between precision and recall, defined as follows: Precision: The number of correct positive results divided by the number of positive results, as predicted by the classifier. Recall: The number of correct positive results divided by the number of all relevant samples (that is, all samples that should have been identified as positive).

8 Conclusion

For this study, we were required to identify a dataset that was potentially mis-labelled. We observed and evaluated a variety of data imputation methods, feature selections and algorithms, and identified 2 classifiers and compare the combination of feature selection method.

We demonstrate that by up-sampling a dataset, as a converse to imbalance, that distribution accuracy and F1 has decreased. Commonly used algorithms do not allow for data distribution and are often found to be biased in favour of the majority class. Thus, it is essential that an imbalanced dataset gets special attention to ensure that both minority and majority classes are properly represented. Initial speculation is upsampling on this datset creates an equal distribution of the four classes which may have resulted in over fitting.

When developing classification models, we frequently need to go beyond just accuracy. A thorough understanding and comprehension of recall, precision, F1 will allow us to better assess classification models to facilitate a healthy scepticism where there is an over-hyped model that focusses only on one metric and fails to emphasize the need for a model to be able to discover all the relevant cases within a dataset, especially where problems are imbalanced. We observed that KNN with its low model complexity produces good result followed by SVM.

It was noted that each stage of this study precision, recall, F1 and accuracy of results, where the best result will proceed to the next section. Importantly, it was found that when Random Forest Feature Selection was able to remove highly correlated variables, the performance of classifiers was significantly improved.

9 Contribution

- $+$ indicates percentage of coding
- o documentation & analysis

Section	Melissa	Zhuoyang	Josh	Biji	Sergio
Introduction					

Section	Melissa	Zhuoyang	Josh	Biji	Sergio
Data Load	+	+	+	+	+
Data Imputation					
-Mean			+o	o	
-Median				+o	
-KNN Inputation		+o			
-Missknn	+o				
Class Imbalance	+		o	o	
Feature Selection					
-Correlated Predictor	+		o	o	o
-Feature Importance rpart	+		o	o	o
-Feature Importance RF		+	o	o	o
-PCA	+	+			
-SPLSDA		+o	o		
Train Model					
Rpart	+				
Random Forest		+			
KNN			+		
SVM				+	
Fine Tune Parameter	+	+	+	+	
Predict Class	+	+			
Conclusion	o	o	o	o	
Analysis/Powerpoint		o	o	o	o

10 Reference

[1] Wicked Good Data, Handling Class Imbalance with R and Caret – An Introduction, accessed 28 Oct 2018, https://www.r-bloggers.com/handling-class-imbalance-with-r-and-caret-an-introduction/ (https://www.r-bloggers.com/handling-class-imbalance-with-r-and-caret-an-introduction/)

[2] Jonathan A C Sterne, Michael Sprat, James R Carpenter, Jun 2009, Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls, BMJ 2009;338:b2393, accessed 28 Oct 2018, https://www.bmj.com/content/338/bmj.b2393.full.pdf+html (https://www.bmj.com/content/338/bmj.b2393.full.pdf+html)

[3] Mandel J, S. (2015). A Comparison of Six Methods for Missing Data Imputation. Journal Of Biometrics & Biostatistics, 06(01). doi: 10.4172/2155-6180.1000224

[4] Joachim, Statistical Programming, accessed 28 Oct 2018, https://statistical-programming.com/mean-imputation-for-missing-data/ (https://statistical-programming.com/mean-imputation-for-missing-data/)

[5] Wale Akinfaderin, Missing Data Conundrum: Exploration and Imputation Techniques, accessed 28 Oct 2018, https://medium.com/ibm-data-science-experience/missing-data-conundrum-exploration-and-imputation-techniques-9f40abe0fd87 (https://medium.com/ibm-data-science-experience/missing-data-conundrum-exploration-and-imputation-techniques-9f40abe0fd87)

[6] Sagar, C. (2018). Feature selection techniques with R. Retrieved from http://dataaspirant.com/2018/01/15/feature-selection-techniques-r/ (http://dataaspirant.com/2018/01/15/feature-selection-techniques-r/)

[7] Kazemitabar, J. et. al. (2018). Retrieved from https://www.cs.cmu.edu/~atalwalk/dstump_nips17.pdf (https://www.cs.cmu.edu/~atalwalk/dstump_nips17.pdf)

[8] Gluck, C. (2018). Running Random Forests? Inspect the feature importances with this code. Retrieved from https://towardsdatascience.com/running-random-forests-inspect-the-feature-importances-with-this-code-2b00dd72b92e (https://towardsdatascience.com/running-random-forests-inspect-the-feature-importances-with-this-code-2b00dd72b92e)

[9] Saabas, A. (2018). Selecting good features – Part IV: stability selection, RFE and everything side by side | Diving into data. Retrieved from https://blog.datadive.net/selecting-good-features-part-iv-stability-selection-rfe-and-everything-side-by-side/ (https://blog.datadive.net/selecting-good-features-part-iv-stability-selection-rfe-and-everything-side-by-side/)

[10] PLS-DA | mixOmics. (2018). Retrieved from http://mixomics.org/methods/pls-da/ (http://mixomics.org/methods/pls-da/) [11] Lê Cao, K., Boitard, S., & Besse, P. (2018). Sparse PLS discriminant analysis: biologically relevant feature selection and graphical displays for multiclass problems.