**The Classification Part**

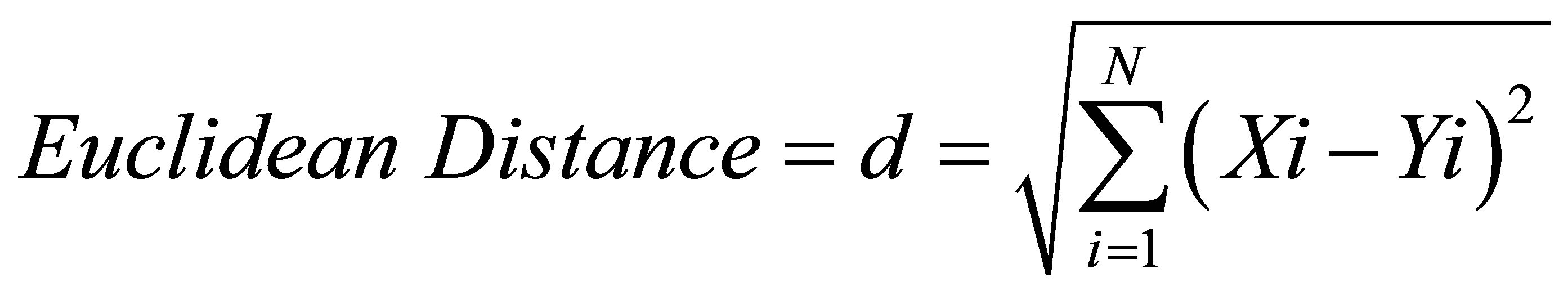
Then ,we will talk about the part of classification, since we have got training data and the training classes(labels) ,we are dealing with the supervised machine learning problem ,The idea of classification is to find the model that could fit and learn from the training data sets reasonably to make prediction on the classes or the labels of the incoming class-unidentified data.

1.This section will answer the models behind the two classification algorithms we chose and implemented

* 1. **First Model**

The first Model that has been implemented is the K-Nearest-Neighbors classifier, which is a non-probabilistic classifier and is generally used for prediction and classification for discrete valued classes. The basic idea for the algorithm is to find the K closest training points to each of the testing points and assume the class of that single testing point is the class of the majority training points within those k points around the testing point. The algorithm is implemented in CM-k-Nearest-Neighbors.file and called knn\_classifier.

Then, we will look into the procedure of the implementation to explain the mathematical details of this model. First of all, the function will take 4 parameters, the trainX(training data x with features),the traint(training class), the test\_data(single row of the testing data x with features) and the K value(a constant). Initially, we calculate the distance between the test\_data x which is one row of the whole testing data and the entire training data x using Euclidean distance which is one of the most frequently used distance metric, the reason why we use this approach is that it could generally provide accurate distance between 2 data and it is easy to use. The mathematical formula is represented as:



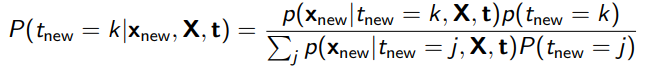
Thus, we could get the distance or the similarity of each training data x to the testing data x by doing subtraction on their features using:

**distances = np.sqrt(((trainX - test\_data)\*\*2).sum(axis=1))**

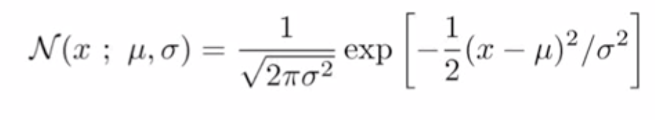
so that we can observe the similarity of two different data set based on their feature differences, which means that if their feature values are pretty similar, it is high likely that they are coming from same classes, then we sort the distances in ascend order to obtain the first K distance of training data x, after that, we count the training data x for each class and assign the greater counted class to the specific testing data x as well as calculating the predicted score for that testing data x. Finally, we combine the predicted class and predicted score into dictionary and return them.

**1.2. Second Model**

The second model has been applied is the Naïve Bayes classifier, which is a probabilistic based classifier relying on Bayes’s theorem and it is commonly utilized in text classification and grouping problem with multiple classes. The idea of this metric is that the class of the data points is determined based on the value of posterior probability where the posterior which is the probability that x belong to t=k is calculated by the normalized multiplication of the likelihood that x is sampled from t=k and the prior (The probability of the occurrence of the class k: in all of classes):



And the likelihood probability for calculating posterior probability is calculated using Gaussian Distribution function (here we assume the data is normally distributed) denoted as:



Now, the implementation will be explained to observe mathematic details, first of all, we calculate the mean μ and variance σ2 (applying on features) and prior of training data for each classes and calculate the likelihood probability using gaussian distribution provided above with mean and variance as well as testing data x (the higher the distance of testing data x to training data x ‘s mean, the lower the likelihood probability( testing data x is not likely from the class of training data x)) . afterwards, the prior is multiplied and whole computation is normalized to attain the posterior probability, finally, the training data class of the higher posterior probability is selected as the class for testing data x since the testing data x has lower distance to that training data x ‘s mean and the normalized probability itself is assigned as predicted score

**1.3. Benchmark for Two Models**

The differences that make Naïve Bayes approach different from K-Nearest-Neighbors method are:

1. It assumes that the presence of specific feature in the class is independent of other features within that class, so it could separate data using features based on this assumption. whereas KNN separate the data based on feature similarity of each pair data regardless the differences between distinct features, and Naïve Bayes make the assumption that the data is normally distributed.
2. Naïve Bayes classifier is linear classifier whereas KNN is contrary, it could process quick on big data

2.This section will demonstrate the experiment setup for training the classifiers and performance measurements

* 1. **For K-Nearest-Neighbors**

Firstly, For the experiment setup stage ,the training and testing data are loaded using numpy, then, the VarianceThreshold with threshold set as 0 is utilized to filter out the constant features for the training x and testing x data as the constant features does not help in classification. After eliminating the constant features, we apply StandardScaler to normalize the training and testing x data, the reason why we normalize the data is that we want the independent features to be in the same range and that is essential for distance based models.

Then, moving to the training and performance measuring stage, before moving to Repeated k-fold cross validation, we need to introduce the k-fold cross validation which is the process of separating the training data into k-folds and take N-1 as training data and 1 as testing data for validation to each fold, subsequently, The Repeated k-Fold cross validation technique with 10 times repetition and 10-Fold separation is used to cross validate on the data to measure the generalized performance and obtain the best k for KNN classification. The idea of Repeated K-Fold cross validation is to repeat certain times to testing on the accuracy (which in this case is calculated by 0/1 loss)of k-fold cross validation and measure the mean result of across all folds to improve the approximate performance of the model by reducing bias. In this case, if we cannot get a precise outcome of the model from k-fold cross validation, we will get relative accurate performance of the model by repeating it for several times. Thus, by performing Repeated k-fold cross validation on training data of all features, the best k which is **9** is found from the training procedure to ensure the average accuracy of this specific cross validation to **91.05%.** After executing this performance measure, sequence of other performance measure approaches is also applied to assisting confirming the performance of the model. They are:

(1).0/1 loss based cross validation with fixed random state, ROC (receiver operating curve) which is utilized to represent the trade-off between experimental sensitivity and specificity.

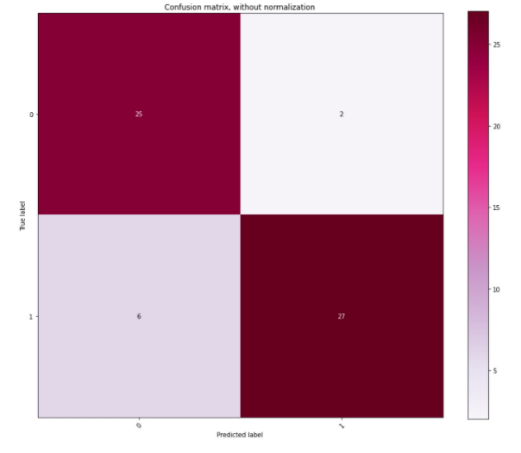
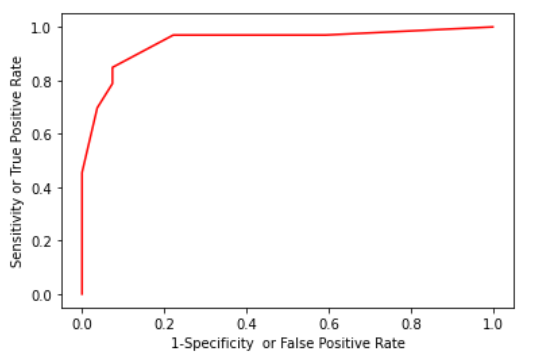
(2). AUC (Area under curve) which will display how true positive rate (recall) and false positive rate trade off,

(3). confusion matrix (present number of TP, TN, FP, FN of classification),

(4). Precision which refers to the proportion relevant of the result, recall gauges the fraction of overall relevant results accurately classified by model, and f1-score indicates the weighted average of the precision and recall.

Primarily, we perform cross validation to split training data and testing data into 7:3 ,then By applying Best K=9 to the model and fit the training data to predict on test data, consequently, we have got the Accuracy of **86.7% ,**AUC= **0.9438832772166106**,precison:class=1.0(epithelial)=**0.926**,precision:class=2.0(stromal)=**0.818**,recall:class=1.0(epithelial)=**0.806**,recall:class=2.0(stromal)=**0.913** and

f1-score:class=1.0(epithelial)=**0.862**, f1-score: class=2.0(stromal)=**0.871**, The ROC (left) and confusion matrix(right), those are the good quality values from prediction, and it indicates that our model is trained decently.

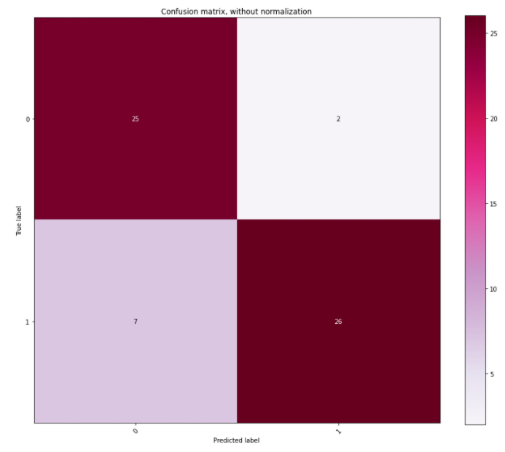
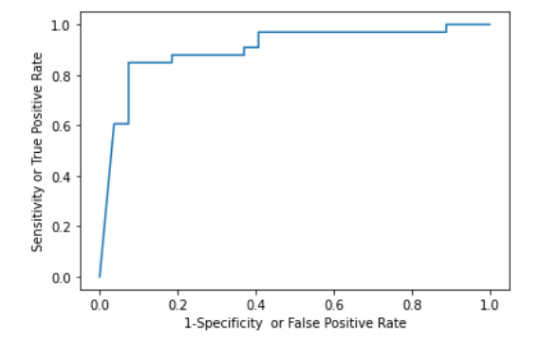


* 1. **For Naïve Bayes**

As the same experiment setup procedure as KNN for comparison intention, we first remove the constant features and normalize the data. then, for the training and performance measuring stage, we did Repeated k-fold cross validation with fixed randomness on training data with all features again to test the generalizability and overall accuracy of the prediction of the Naïve Bayes classifier, subsequently, the result has been collected ,the average accuracy of the repeated k-fold cross validation for this model is **83.1%**, afterwards, the same confirmation performance metrics that have been applied to measure the KNN model have been applied to Naïve Bayes classifier as well, which gives us **85%** (which is generally similar to the value produced by Repeated Cross Validation) accuracy for cross validation of fixed randomness with training and testing data divided with ratio 7:3 ,AUC of **0.9023569023569024,**

precison:class=1.0(epithelial)=**0.926**,precision:class=2.0(stromal)=**0.788**,recall:class=1.0(epithelial)=**0.781**,recall:class=2.0(stromal)=**0.929** and

f1-score:class=1.0(epithelial)=**0.847**, f1-score: class=2.0(stromal)=**0.852**, The ROC (left) and confusion matrix(right),



**3.Compare the performance of these two algorithms**

Before comparison, the metrics that are utilized to help with comparison will be introduced, they are accuracy validated by cross validation with data separated in ratio 7:3, ROC graph, AUC, confusion matrix, Precision, Recall and f1-score measurements.

Then, we introduce the some of the properties of those classification measuring method:

(1). The precision measures the correctness of the positive identification for model, higher the better

(1). The recall measures the correctness of actual positive identification, the higher the better.

(3). The confusion matrix provides the number of TP, TF, FP, FN of model.

(2). F1-score is used more than accuracy when the TN and TP has relative different amounts.

(3). As this is Binary classification, the AUC is therefore used to measure performance of classifier, the higher result represents better model.

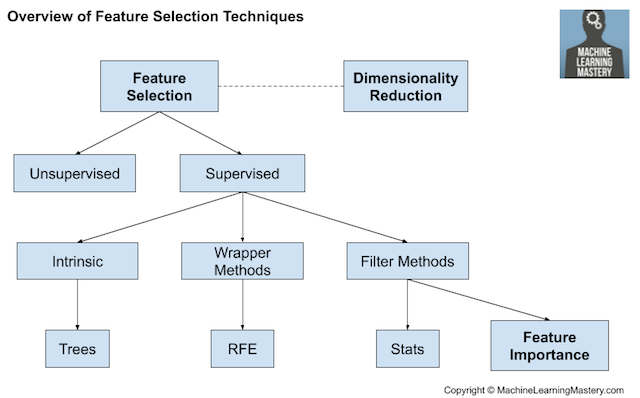
By observing the performance measurements listed above, we find that the overall performance of KNN is better than that of Naïve Bayes classifier as it generally provides higher accuracy, AUC, precision, recall and f1-score,The higher AUC means better tradeoff between recall of FP rate, the higher precision indicate higher relevant of the result, the higher recall return the higher entire relevant of results precisely classified by classifier, the higher f1-score indicate the relative higher harmonic average of the model’s precision and recall .Also KNN also obtain lesser FN by observing confusion matrix above, moreover, the ROC of the KNN offers better tradeoff between clinical sensitivity and specificity than Naïve bays model as we can see from graph below.

However, if we look into the results, we will find that the recall for label stromal=0.929 of Naïve Bayes classifier is slightly higher than that of KNN classifier which is 0.912,which give us the insights that Naïve Bayes classifier may find more corrected positive instance of stromal than KNN would do. In this case, if we are looking for more correct number of normal cells, Naïve Bayes might be more useful than KNN model.

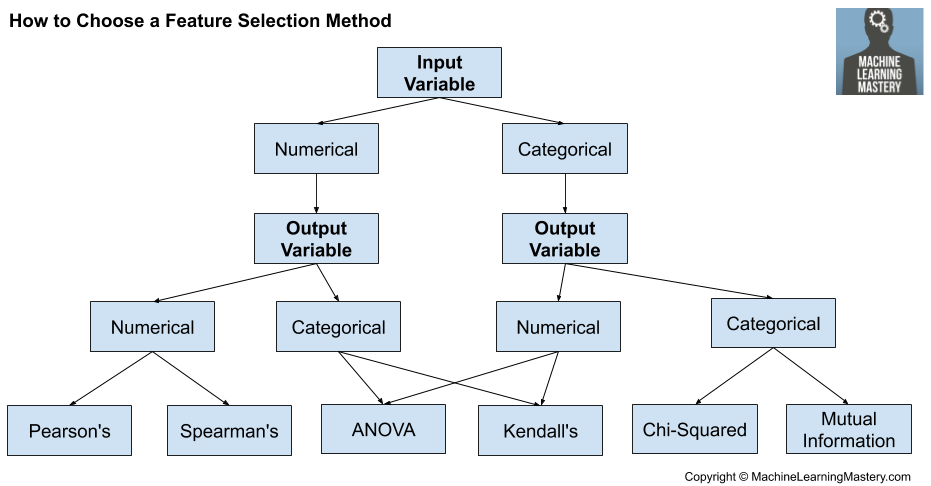
In conclusion, the overall performance of the KNN classifier is better than that of Naïve Bayes classifier when we deploy all the 112 features into classification. The reason could be that the Naïve Bayes classifier relays more on the independent property of the features to determine the labels for the data, whereas KNN relays on the similarity of the features to determine the labels. So if there’s correlated features involved in calculation, the result would not be optimistic, So, the solution would be that we deploy feature selection method to find subset of features to obtain the better performance for Naïve Bayes classifier than apply all of the features.

**4, This section will confirm that if we use feature selection approaches to obtain subset of features, we will make better performance of those 2 classifiers**

There are many feature selection methods exist, in order to perform the correct feature selection method, we need to first acknowledge what kind of learning we are performing (supervised or unsupervised). the selection would be demonstrated as figure below:



All our classification model is supervised; thus, we could use Intrinsic, Wrapper methods and Filter Methods. Our idea is to construct pipeline of feature extraction which involve the first stage of Filtering methods processing and second stage of Wrapping method processing. For the filter methods, the type of the input and output data of the model will take part in choosing the correlation statics for the method. The specific choosing strategy is shown as figure below:



Since the input data of our model is Numerical and the output of our model which is binary labels for cell type is categorical, we choose ANOVA as our statistic model, we can apply it along the sklearn library SelectBest(Select best k features) by importing f\_classif() from sklearn library.

For the Wrapper method, we will utilize RFE (Recursive Feature Elimination) which will examine for subset of features by removing useless features from the model by fitting the model several times. The determination of whether to keep feature will be applied using feature importance attribute of the model.

**4.1. For Naïve Bayes model**

We initially choose 50 features using SelectBest method and apply Wrapping method of RFE to recursively eliminate the features until last 20 top feature remains. Afterwards, the same process illustrated above in section 2 which utilize Repeated k-fold cross validation is again performed to test and validate our model using training data. After several arounds of training and choosing different number of top features, we find out that the top 10 of the features

**(which are'SkewnessLayer1', 'RatioLayer3’,'ContrasttoneighborpixelsLayer33',**

**'HSITransformationHueRLayer1GLayer2BLayer3’,'Areaofsubobjectsmean1Pxl',**

**'Areaofsubobjectsstddev1Pxl’,'GLCMHomogeneityquick811Layer1alldir',**

**'GLCMContrastquick811Layer1alldir’,'GLCMMeanquick811Layer1alldir',**

**'AreaofsubobjectsNucleus1Pxl’)**

bring the relative higher generalized accuracy (**90.15% (112 features 83.1%)**) than other attempts. Then, in order to confirm the prediction accuracy of our model, we then applied the same sequence of performance measurements methods illustrated above to help with validation. Firstly, the 0/1 loss based cross validation which partition training data into 7:3 is again performed and as the outcome, we have accomplished the accuracy of **90%** **(112 features 85%)** (which is generally similar to the value produced by Repeated Cross Validation) ,AUC of

**0.9719416386083053(112 features 0.9023569023569024)**,

precison:class=1.0(epithelial)=**0.963(112 features 0.926,)**,

precision:class=2.0(stromal)=**0.848**, **(112 features 0.788)**

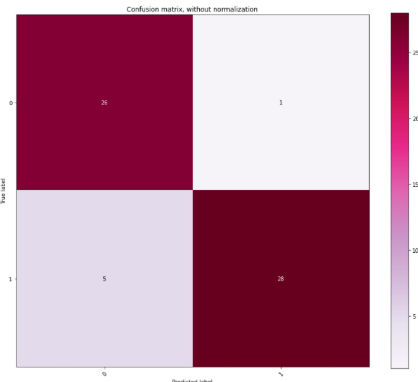
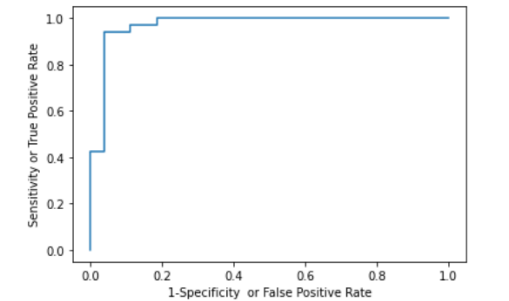
recall:class=1.0(epithelial)=**0.839(112 features 0.781)**,

recall:class=2.0(stromal)=**0.966** **(112 features 0.929)** and

f1-score:class=1.0(epithelial)=**0.897(112 features 0.847)**

f1-score: class=2.0(stromal)=**0.903(112 features 0.852)**,

The ROC (left) and confusion matrix (right):



The number of FN and FP are reduced from 7,2 to 5 and 1, In which is a huge improvement of the original analysis based on entire features.

**4.2 For K-Nearest-Neighbors model**

We applied the same strategy used in Naïve Bayes subset feature analysis to the KNN model and we therefore obtain the average accuracy of **93.3% (112 features 91.05%)** with best k as **3(112 features 9)** for Repeated k-fold cross validation, accuracy of **90%** **(112 features 86.7%)** for cross validation on training data splited with ratio 7:3, AUC of

**0.9377104377104377** (**112 features 0.9438832772166106**),

precison:class=1.0(epithelial)=**0.963(112 features 0.926**),

precision:class=2.0(stromal)=**0.848, (112 features 0.818)**

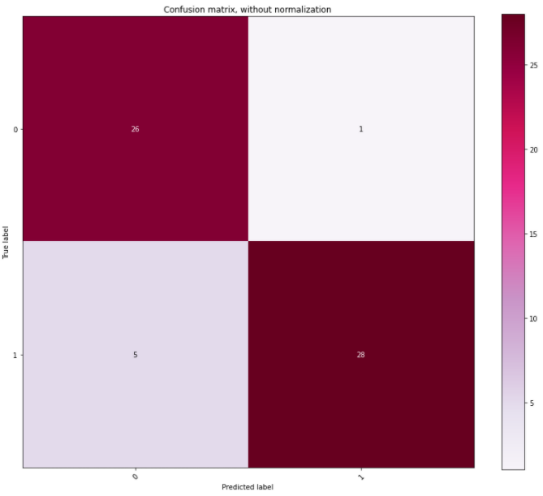
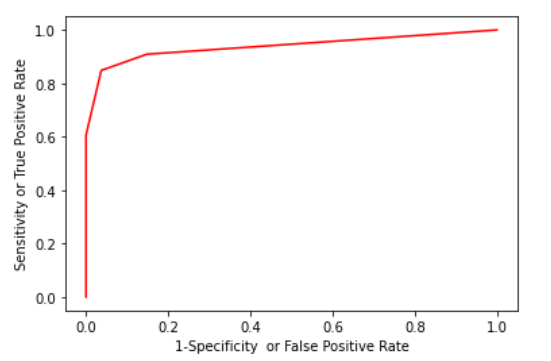
recall:class=1.0(epithelial)=**0.839(112 features 0.806)**,

recall:class=2.0(stromal)=**0.966 (112 features 0.931)** and

f1-score:class=1.0(epithelial)=**0.897(112 features 0.862)**

f1-score: class=2.0(stromal)=**0.903(112 features 0.871)**,

The ROC (left) and confusion matrix (right):



, The FN and FP are reduced from 6,2 to 5,1, in which it’s also a decent improvement of the previous entire feature model analysis.

In conclusion, with the evidence provided above, we could deduce that if we choose the subset of features reasonably using feature selection, we will get the better performance than the previous classification analysis operated on whole features.