CYOP report

Mathilde Couteyen Carpaye

Breast Cancer Wisconsin diagnosis

Introduction

The goal of the project was to train a machine learning algorithm on a chosen topic. We chose to analyze a Breast Cancer in Wisconsin data set. This data sets provides information on different patient whose breast tissue were diagnosed either malignant (M) or benign (B). We have various means and numbers on 30 features of breast cell nuclei for 569 patients. The goal is to train a machine learning algorithm that would predict the diagnosis of the breast cancer thanks to the analysis of these features for future patients. The data set is in tidy format. Each row represent all the features for one patient characterized by its id.

Methods

Creating test and train sets

To build a machine learning algorithm, we need to part the data set into train and test sets. At first we will use a test set containing 50% of the data and train set containing the other 50%.

Data exploration

Here we have 30 predictors. But some might not be as important as others. To know how much a predictor can influence the diagnosis, we can compute the standard deviation (sd) of each column. If the sd is high then it means that there is a high variability of the data and this might explain the different diagnosis. To compute the sds of each column we first have to convert our data frame into a matrix. Which we can easily do knowing that our inputs are numeric. We can visualize the sd variability of the data set in Figure 1.

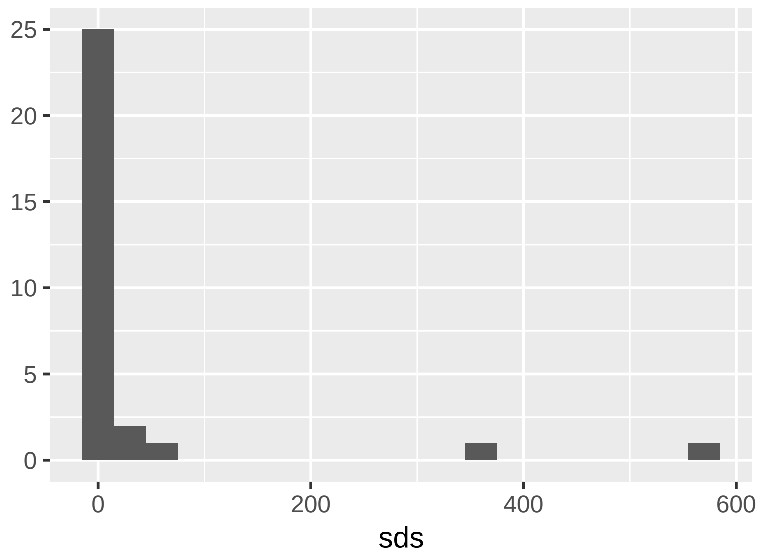

Fig 1. Histogram of the distributions of the sds of each columun

As we can see, only a few predictors are essential to determining the diagnosis. We could remove these predictors to go faster but because there are only 30 of them we didn't remove theme.

Distances and Principal Component Analysis

Each diagnosis can be considered as a point in 30 dimensions corresponding to the 30 features we have on each point. The distance between the predictors j for diagnosis $i = 1$ and $i = 2$ can be defined as

$$dist(1,2) = \sqrt{\sum_{j=1}^{30}(x_{i,j} - x_{2,j})^2}$$

With our matrix we can find a transformation that creates an other matrix that preserves the distance between rows but with the variance of the columns in decreasing order. The first columns represent the principal components (PC). They are the ones that contribute the most to the distance. When the variance of a column is very low we can consider the k previous column as PCs. If k is much smaller than 30 then we can achieve an efficient summary of our data. We can visualize the importance of each PC in Table 1.

Table 1.

```
## Importance of components:
##                             PC1      PC2      PC3     PC4     PC5     PC6
## Standard deviation     666.9805 72.22002 24.50549 7.31860 6.29009 1.60327
## Proportion of Variance   0.9869  0.01157  0.00133 0.00012 0.00009 0.00001
## Cumulative Proportion    0.9869  0.99845  0.99978 0.99990 0.99999 0.99999
##                          PC7    PC8    PC9   PC10   PC11    PC12    PC13
## Standard deviation     1.318 0.6283 0.3972 0.2713 0.1654 0.08426 0.05009
## Proportion of Variance 0.000 0.0000 0.0000 0.0000 0.0000 0.00000 0.00000
## Cumulative Proportion  1.000 1.0000 1.0000 1.0000 1.0000 1.00000 1.00000
##                           PC14    PC15    PC16    PC17    PC18    PC19
## Standard deviation     0.04501 0.03869 0.02496 0.01726 0.01445 0.01361
## Proportion of Variance 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
## Cumulative Proportion  1.00000 1.00000 1.00000 1.00000 1.00000 1.00000
##                          PC20     PC21     PC22     PC23    PC24     PC25
## Standard deviation     0.0125 0.008454 0.007053 0.005176 0.00486 0.003681
## Proportion of Variance 0.0000 0.000000 0.000000 0.000000 0.00000 0.000000
## Cumulative Proportion  1.0000 1.000000 1.000000 1.000000 1.00000 1.000000
##                            PC26     PC27     PC28     PC29      PC30
## Standard deviation     0.003335 0.001988 0.001686 0.001543 0.0007867
## Proportion of Variance 0.000000 0.000000 0.000000 0.000000 0.0000000
## Cumulative Proportion  1.000000 1.000000 1.000000 1.000000 1.0000000
```

PC1 and PC2 seems to account for 99% of the variability. This means that we could estimate pretty efficiently the distances with only these two PCs. If we plot the actual distance against the distance approximated with these two PCs we obtain Figure 2.

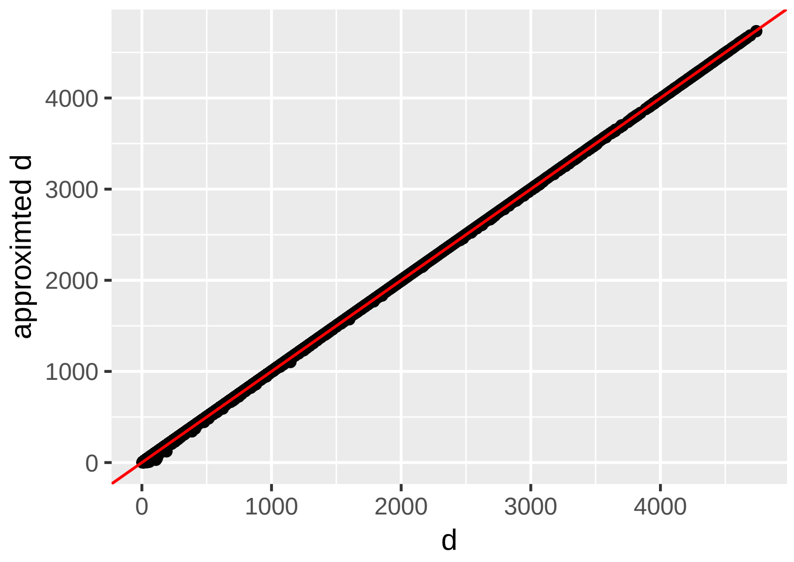

Fig 2. Plot of the true distances against the approximated distances

The approximation we made we these two predictors is pretty accurate as we obtain almost a linear model which means the distances are preserved.

If we now plot PC1 and PC2 with colors depending on the diagnosis (Figure 3) we can distinguish B and M diagnosis.

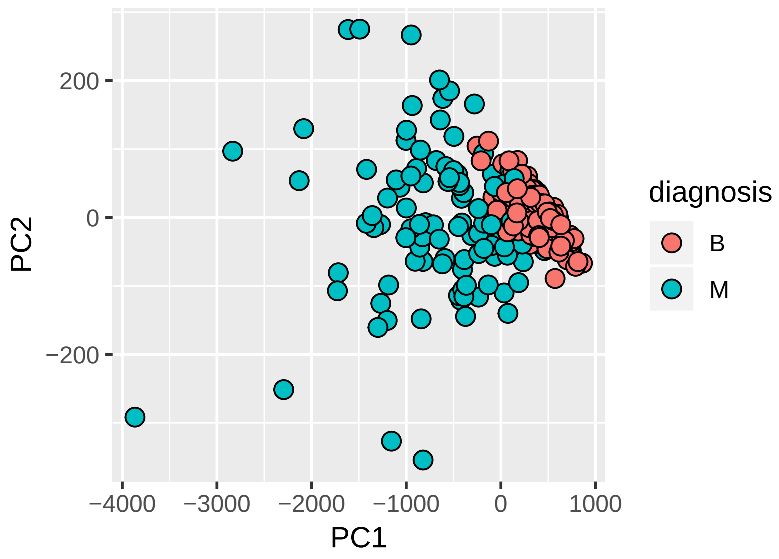

Fig 3. Plot of PC1 against PC2

Building the algorithm

k-nearest neighbors

For this analysis we focused on the k-nearest neighbor algorithm. To estimate $p(x_1, x_2)$ for any point (x_1, x_2) we look for the k nearest points to (x_1, x_2) and take the average of their observations. To implement the algorithm we used the knn3 function of the caret package with R.

First we fit the knn model on a train set containing only the two first PCAs. Then we have to transform the test set into a matrix with the PCA rotation. When it is done we can make our prediction on the test set and check our accuracy.

The tuning parameter we have to chose there is k. To select the best k we run our model with different ks from 1 to 50 with a span of 0.5.

Because we only have 30 parameters, we will run the same model but instead of using only PC1 and PC2 we will use all the predictors.

K-fold cross validation

To have as much data as possible to train we changed our training and tests sets. Our train set then comprise 80% of the data and our test set the other 20%. We divided our train set into $K = 10$ non-overlapping samples. We chose 10 so it would not take to much time.

K-fold cross validation consist into training 9 samples and use the one remaining as a validation set to chose the best parameters. This operation is repeated 10 times so each of the 10 sample serves as validation set. We then compare the accuracy and chose the parameter that returns us the best one.

Results

Figure 4 shows the accuracy obtained on both sets depending on k for our knn model on the PCs. We ended up using $k = 10.5$.

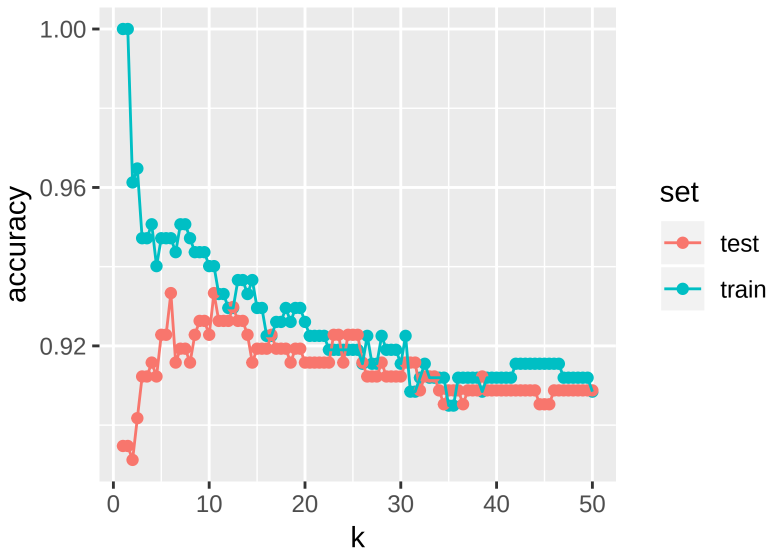

Fig 4. Evolution of the accuracy depending on k

Figure 5 shows the accuracy obtained on both sets depending on k for our knn model on all predictors. We ended up using $k = 11$.

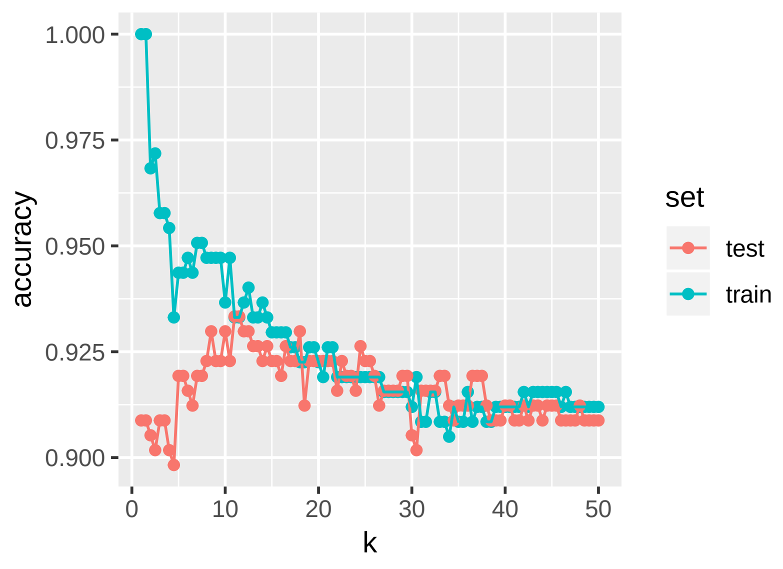

Fig 5. Evolution of the accuracy depending on k

Figure 6 shows the accuracy obtain with different ks for the K-fold cross validation. The k that maximizes the accuracy is 6.5.

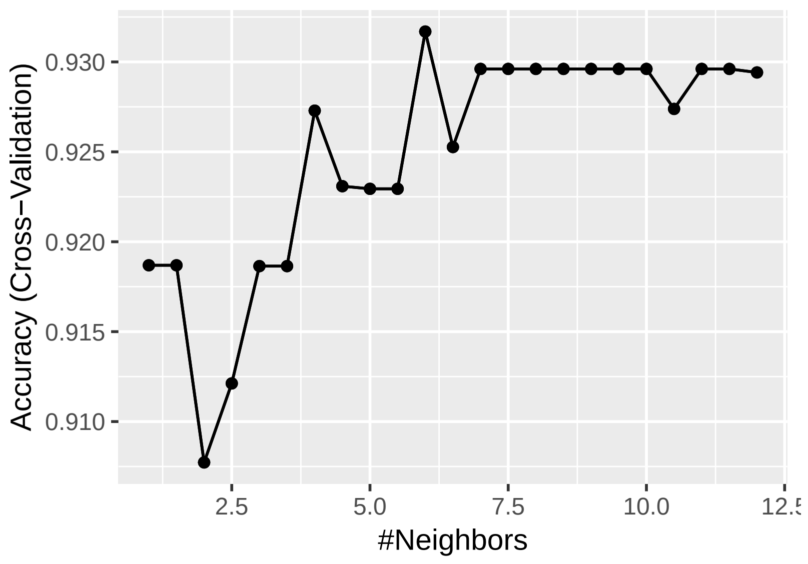

Fig 6. Evolution of the accuracy depending on k

The results obtained with the three methods are presented in Table 2.

Table 2. Accuracy results

Method	Accuracy
PCA	0.9333333
All predictors	0.9333333
K-fold cross validation	0.9478261

With the knn on the PCAs we get an accuracy of 93% as for the method with all the predictors.

When we improved our algorithm with K-fold cross validation, we get an accuracy of almost 95%.

Conclusion

To build our algorithm we relied on the analysis of the distances between predictors thanks to the knn function. Using all predictors or only the two first PCs returned us the same accuracy. This means that indeed with these only two PCs we can achieve a pretty good prediction algorithm. However our accuracy is of 93%. To improve our model we used K-fold cross validation. By using 6.5 nearest neighbor we managed to get an accuracy of about 95% which we consider satisfying.