Breast Cancer Diagnosis Prediction

HarvardX Data Science Professional Certificate: PH125.9x, Choose Your Own

Mahaman Sani SAHIROU ADAMOU

18 septembre, 2022

Contents

1 Introduction

Breast cancer is a malignant tumor of the mammary gland. In other words, it is a cancer that is born in the cellular units whose function is to secrete milk, the ducto-lobular units of the breast, essentially in women. Eight out of ten breast cancers occur after the age of 50. First cancer in the world, it affects, in 2016, 1.8 million women per year in the world. One in eight women is expected to be diagnosed with breast cancer in her lifetime.

A tumor does not mean cancer, it can be benign (not cancerous), pre-malignant (pre-cancerous), or malignant (cancerous). It is now possible to detect tumors very early when they are still very small. This reduces the risk of mortality and allows lighter and less traumatic treatments.

Tests such as MRI, mammogram, ultrasound and biopsy are commonly used to diagnose breast cancer. Also, machine learning based predictive models promise earlier detection techniques for breast cancer diagnosis. However, making an evaluation for models that efficiently diagnose cancer is still challenging.

In this project, we proposed data exploratory techniques and developed different predictive models to improve machine learning based breast cancer diagnostic accuracy.

2 Getting data

Our models were implemented on the **W**isconsin **D**iagnostic **B**reast **C**ancer (WDBC) dataset which consists of 10 features (Table 1) of breast tumor, and the result in the data were taken from 569 patients, samples of malignant and benign tumor cells. These features were computed from digitized images of breast mass **F**ine **N**eedle **A**spiration (FNA) and describe characteristics of cells nuclei.

Table 1: Description of nucleus features

Feature	Description
Radius	Mean of distances from center to points on the perimeter for each cell nucleus
Texture	Standard deviation of gray-scale values
Perimeter	Perimeter length of each cell nucleus
Area	Area of each cell nucleus
Smoothness	Local variation in radius lengths
Compactness	Calculated using the formula 'perimeter^2 / area - 1.0'
Concavity	Severity of concave portions of the contour
Concave Points	Number of concave portions of the contour
Symmetry	Symmetry of the nucleus as measured by length differences between lines perpendicular to the major axis and the cell boundary
Fractal Dimension	Based on 'coastline approximation' - 1

The mean, standard error (SE) and worst or largest (mean of the three largest values) of these features were computed, resulting in 30 features in total for each sample, to which an **ID column** was added to differentiate samples. Finally, the diagnosis result of each sample, which consisted of malignant (M) and benign (B), was also added. In conclusion, the dataset contained 32 attributes (ID, diagnosis, and 30 input features) and 569 instances.

The first column of the dataset, **«id_number»**, was not considered when building our models. The second column, **«diagnosis»**, is the target of the study, the **outcome** or **dependent variable** that will be predicted by our models. The third to the thirtyth-second columns are the **predictors**.

3 Analysis

3.1 Exploratory data analysis (EDA)

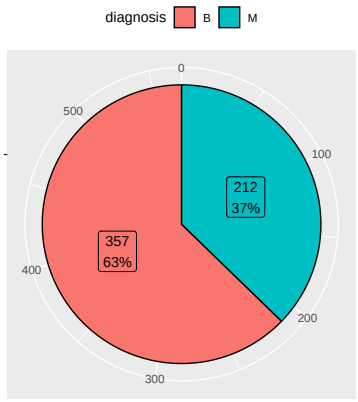

Figure 1: Distribution of «diagnosis»

EDA help to understand the nature of the dataset, to identify the outliers or correlated variables that are more accessible. To analyze the features and understand their predictive value for diagnosis, we applied feature distribution, correlation coefficient, and so on.

First of all, we discover that the dataset is a bit unbalanced in its proportions, as we can see on Figure 1.

This imbalance between benign and malignant samples means that we should probably look beyond overall accuracy in evaluating our models.

3.1.1 Descriptive statistics

There is no missing values (NAs) in the WDBC dataset.

For each of the predictors, higher values generally indicate a higher likelihood of malignancy as they reflect larger cells and/or more irregular shapes.

Tables **2, 3 and 4** summarise the numeric data for each of the features included in the dataset, showing that the range and magnitude of values for each feature vary considerably and would benefit from Z-Score normalization prior to further visualization and use in developing predictive algorithms.

Table 2: Statistics for the 'mean' sufixed features

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	symmetry_mean	fractal_dimension_mean
Min.	6.98100	9.71000	43.79000	143.5000	0.0526300	0.019380	0.0000000	0.0000000	0.1060000	0.0499600
1st Qu.	11.70000	16.17000	75.17000	420.3000	0.0863700	0.064920	0.0295600	0.0203100	0.1619000	0.0577000
Median	13.37000	18.84000	86.24000	551.1000	0.0958700	0.092630	0.0615400	0.0335000	0.1792000	0.0615400
Mean	14.12729	19.28965	91.96903	654.8891	0.0963603	0.104341	0.0887993	0.0489191	0.1811619	0.0627976
3rd Qu.	15.78000	21.80000	104.10000	782.7000	0.1053000	0.130400	0.1307000	0.0740000	0.1957000	0.0661200
Max.	28.11000	39.28000	188.50000	2501.0000	0.1634000	0.345400	0.4268000	0.2012000	0.3040000	0.0974400

Table 3: Statistics for the 'standard error (se)' sufixed features

	radius_se	texture_se	perimeter_se	area_se	smoothness_se	compactness_se	concavity_se	concave_points_se	symmetry_se	fractal_dimension_se
Min.	0.1115000	0.360200	0.757000	6.80200	0.001713	0.0022520	0.0000000	0.0000000	0.0078820	0.0008948
1st Qu.	0.2324000	0.833900	1.606000	17.85000	0.005169	0.0130800	0.0150900	0.0076380	0.0151600	0.0022480
Median	0.3242000	1.108000	2.287000	24.53000	0.006380	0.0204500	0.0258900	0.0109300	0.0187300	0.0031870
Mean	0.4051721	1.216853	2.866059	40.33708	0.007041	0.0254781	0.0318937	0.0117961	0.0205423	0.0037949
3rd Qu.	0.4789000	1.474000	3.357000	45.19000	0.008146	0.0324500	0.0420500	0.0147100	0.0234800	0.0045580
Max.	2.8730000	4.885000	21.980000	542.20000	0.031130	0.1354000	0.3960000	0.0527900	0.0789500	0.0298400

Table 4: Statistics for the 'worst' sufixed features

	radius_worst	texture_worst	perimeter_worst	area_worst	smoothness_worst	compactness_worst	concavity_worst	concave_points_worst	symmetry_worst	fractal_dimension_worst
Min.	7.93000	12.02000	50.4100	185.2000	0.0711700	0.027290	0.0000000	0.0000000	0.1565000	0.0550400
1st Qu.	13.01000	21.08000	84.1100	515.3000	0.1166000	0.147200	0.1145000	0.0649300	0.2504000	0.0714600
Median	14.97000	25.41000	97.6600	686.5000	0.1313000	0.211900	0.2267000	0.0999300	0.2822000	0.0800400
Mean	16.26919	25.67722	107.2612	880.5831	0.1323686	0.254265	0.2721885	0.1146062	0.2900756	0.0839458
3rd Qu.	18.79000	29.72000	125.4000	1084.0000	0.1460000	0.339100	0.3829000	0.1614000	0.3179000	0.0920800
Max.	36.04000	49.54000	251.2000	4254.0000	0.2226000	1.058000	1.2520000	0.2910000	0.6638000	0.2075000

3.1.2 Feature Distribution

From Figure 2 and Figure 3, we can see that the data are relatively normally distributed and don't require transformations. More precisely:

- concavity, concavity_point, perimeter, radius, area and compactness attributes may have an exponential distribution,
- texture, smooth and symmetry attributes may have a Gaussian or nearly Gaussian distribution,
- malignant samples are, on average, larger in size and more abnormal in shape, than benign samples,
- malignant samples have a greater variance in data than benign samples,
- in any of the histograms there are no noticeable large outliers that warrants further cleanup.

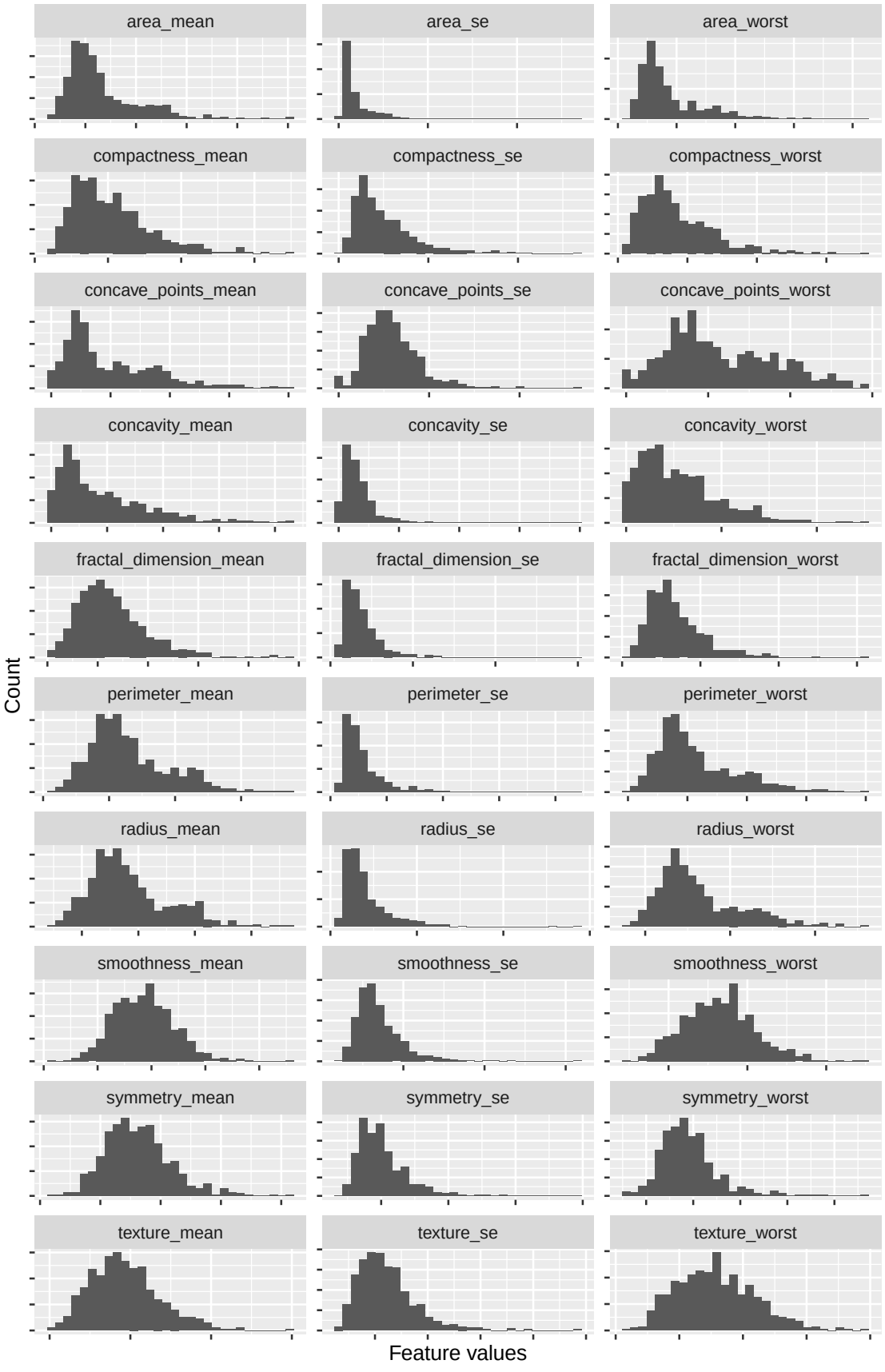

Figure 2: Histogram for each feature

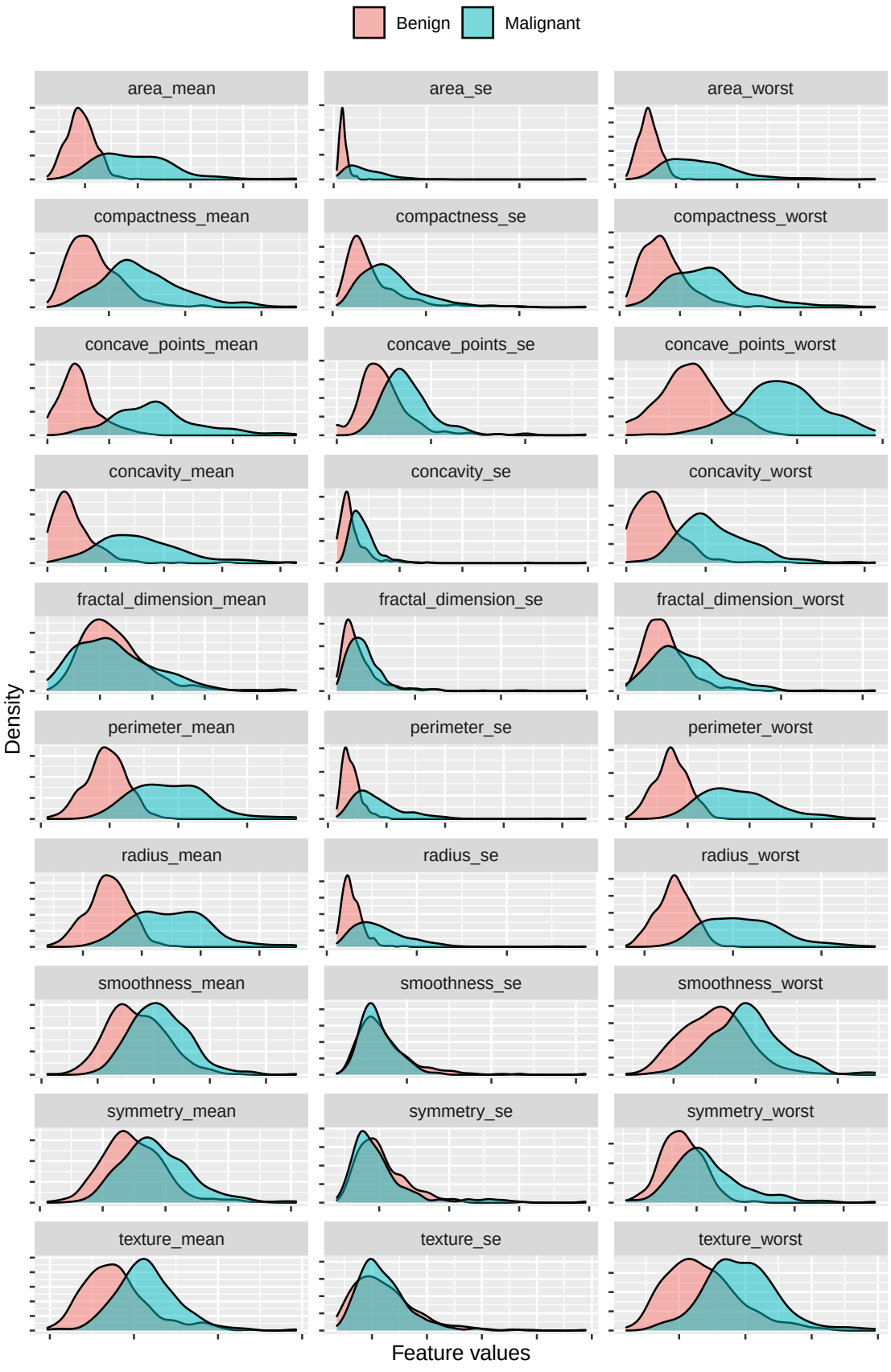

Figure 3: Feature density plots by diagnosis

3.1.3 Feature Correlation

The Pearson Correlation Coefficient ($-1 \leq r \leq +1$) calculates the correlation coefficient between two features. Then, the relationship between two features can be determined by categorizing them into three groups:

- positively correlated features if $r \to +1$ and so the features move in the same direction,
- negatively correlated features if $r \to -1$ and so the features move in the opposite directions,
- uncorrelated features otherwise $r \to 0$, there is no true relationship between them.

We can see from Figure 4 that:

- the mean area of the tissue nucleus has a strong positive correlation with mean values of radius and parameter (as expected),
- some parameters are moderately positively correlated (r between 0.5 and 0.75) : concavity and area, concavity and perimeter, and so one,
- likewise, we see some strong negative correlation between fractal_dimension with radius, texture, parameter mean values.

It is helpful to understand the degree of correlation between features before deciding which features ton include in the development of a model. In particular, unsupervised methods for predictive modelling can perform better if features that are highly correlated with each other are excluded.

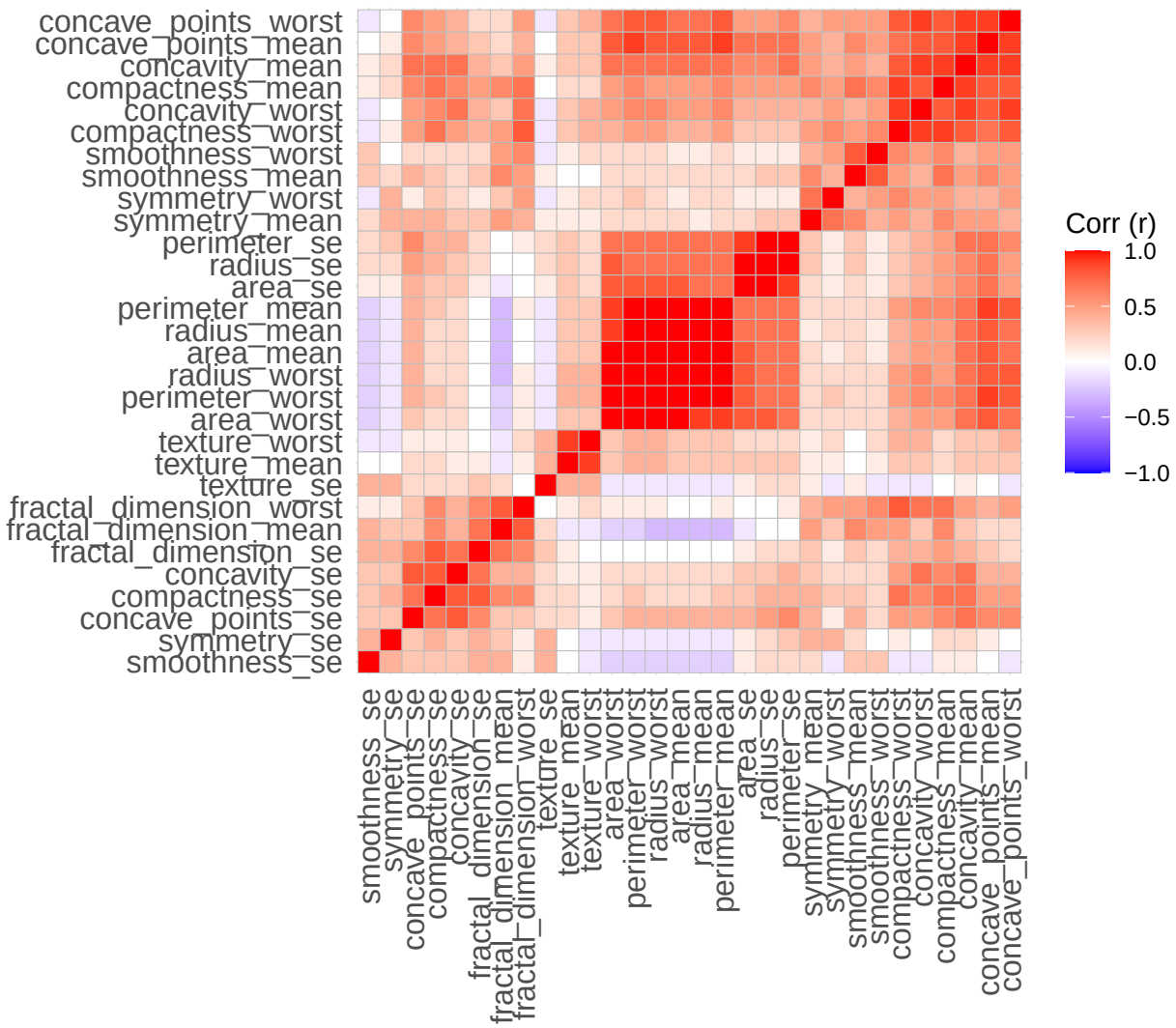

Figure 4: Correlation Heatmap

3.2 Pre-Processing the data

Here we prepare our data in such way to best expose the structure of the problem to the machine learning algorithms that we intend to use.This involves a number of steps such as:

- label encoding,
- taking care of missing data,
- split data into training and test sets,
- feature scaling,
- dimensionality reduction

In this section, we will use tools to find the most predictive features of the data and filter it so it will enhance the predictive power of the analytics models.

First of all, **id_number** is an unique attribute for each patient and it is irrelevant to the process of classification, so it can be removed.

3.2.1 Label encoding

We have only one categorical variable in the data set, the dependent variable **diagnosis** which is encoded as **factor** with 2 levels, one for benign masses (B) and the other for malignant masses (M).

3.2.2 Taking care of missing data

As seen in previous sections, there is no missing values (NAs) in the WDBC dataset.

3.2.3 Split data into training and test sets

The simplest method to evaluate the performance of a machine learning algorithm is to use different training and testing datasets. Here we will:

- split the available data into a training set and a testing set,
- train the algorithm on the first part (training set),
- make predictions on the second part (testing set), and evaluate the predictions against the expected results.

The size of the split can depend on the size and specifics of the dataset, although it is common to use 80% of the data for training and the remaining 20% for testing.

The balance of classes was consistent between the training sets (malignant = 37.2%) and test set (malignant = 37.4%). Further data exploration was conducted with the training set only.

3.2.4 Feature scaling

We have seen over **EDA** sections that our raw data has differing distributions which may have an impact on the most machine learning algorithms. Most machine learning and optimization algorithms behave much better if features are on the same scale.

So, scaling is a useful technique to transform features to a standard Gaussian distribution with a mean of 0 and a standard deviation of 1.

When we center and scale a variable in the training data using the mean and sd of that variable calculated on the training data, we are essentially creating a brand-new variable. Then we train our models on that brand new variable.To use that new variable to predict for the validation and/or

test datasets, we have to create the same variable in those data sets, by subtracting the same means and dividing by the same standard deviations calculated from the training set.

3.2.5 Exploring training set malignant(M) and benign(B) samples

To measure the variance between samples, we can for example calculate the **euclidean distance**. The average distance between all samples included in the training set was **6.99**. **Benign** samples were closer to each other (**6.39**) than **malignant** samples were from each other (**8.02**), indicating greater variance in the measured features in **malignant** cells. **Benign** samples were also closer to each other than to **malignant** samples (**8.53**), indicating that samples are relatively well clustered by class.

3.2.6 Exploring training set features

In this section, we re-explored correlation between features and also explored Principal Component Analysis (PCA).

3.2.6.1 Correlation between training set features Here, we tried to understand the degree of correlation between features before deciding which features ton include in the development of a model. In particular, unsupervised methods for predictive modelling can benefit from excluding features that are highly correlated with each other from the training set.

Overall, there is a low level of correlation between features. Indeed, the mean correlation coefficient of features in the train set is **0.43**. **10** features have a correlation of **0.9** or more, namely **concavity_mean, concave_points_mean, perimeter_worst, radius_worst, perimeter_mean, area_worst, radius_mean, perimeter_se, area_se, and texture_mean**. Excluding these features from unsupervised methods of developing the predictive algorithm may be beneficial.

3.2.6.2 Principal Component Analysis (PCA) **PCA** is a technique for transforming data sets in order to reduce dimensionality without reducing the number of features by identifying the principal components which explain as much of the data variance as possible. PCA reduces the dimensions of a dataset by projecting the data onto a lower-dimensional subspace. For example, a 2D dataset could be reduced by projecting the points onto a line. Each instance in the dataset would then be represented by a single value, rather than a pair of values. In a similar way, a 3D dataset could be reduced to two dimensions by projecting variables onto a plane. PCA has the following utilities:

- Compress the data while minimizing the information lost at the same time and, potentially, improve the predictive accuracy of classification models,
- principal components will be further utilized in the next stage of supervised learning, in Logistic regression, QDA, and so on,
- understanding the structure of data with hundreds of dimensions can be difficult, hence, by reducing the dimensions to 2D or 3D, observations can be visualized easily.

Table 5 shows the standard deviation, proportion of variance and cumulative proportion of variance for the first 10 principal components. The first principal component (PC1) accounts for 45.4% of the total variance within the dataset, the first two components account for almost 63.9% of the cumulative variance and the first 10 principal components account for more than 95.4% of the cumulative variance within the data set.

Table 5: First 10 Principal Components

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10
Standard deviation	3.691824	2.353774	1.653652	1.406342	1.243307	1.102831	0.8442692	0.7059938	0.6157372	0.6116064
Proportion of Variance	0.454320	0.184680	0.091150	0.065930	0.051530	0.040540	0.0237600	0.0166100	0.0126400	0.0124700
Cumulative Proportion	0.454320	0.638990	0.730150	0.796070	0.847600	0.888140	0.9119000	0.9285100	0.9411500	0.9536200

Figure **5** is a series of box plots for each of the first 10 principal components grouped by diagnosis. In most cases the spread is greater for malignant masses than for benign masses. PC1 is the only component for which the interquartile ranges do not overlap. Principal component analysis does not take into account the classification of data, in this case the diagnosis assigned to each sample.

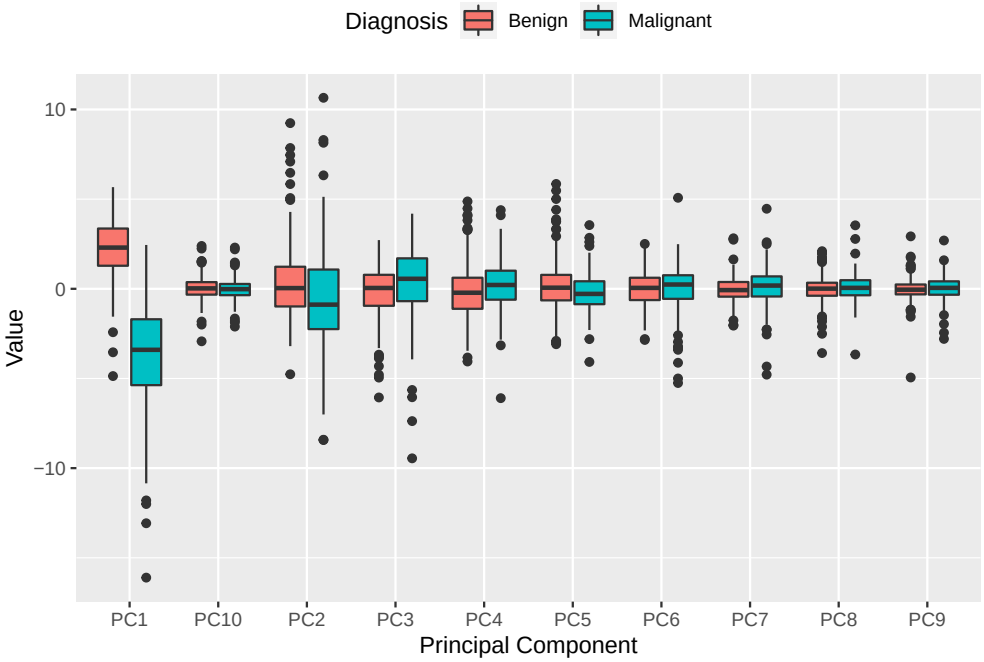

Figure 5: Box plots of first 10 Principal Components by diagnosis

Figure **6** is a two-dimensional scatter plot of the first two principal components. The graph shows that the malignant data-points are more spread out than the benign data-points and that more of the variance can be accounted for on the x-axis (PC1) than on the y-axis (PC2). Ellipses help to visualise this even better, firstly with a larger ellipse for malignant data-points than for benign data-points and considerable separation of data by classification, despite some overlap. This analysis support the use of PCA in algorithm development to predict diagnosis from this dataset.

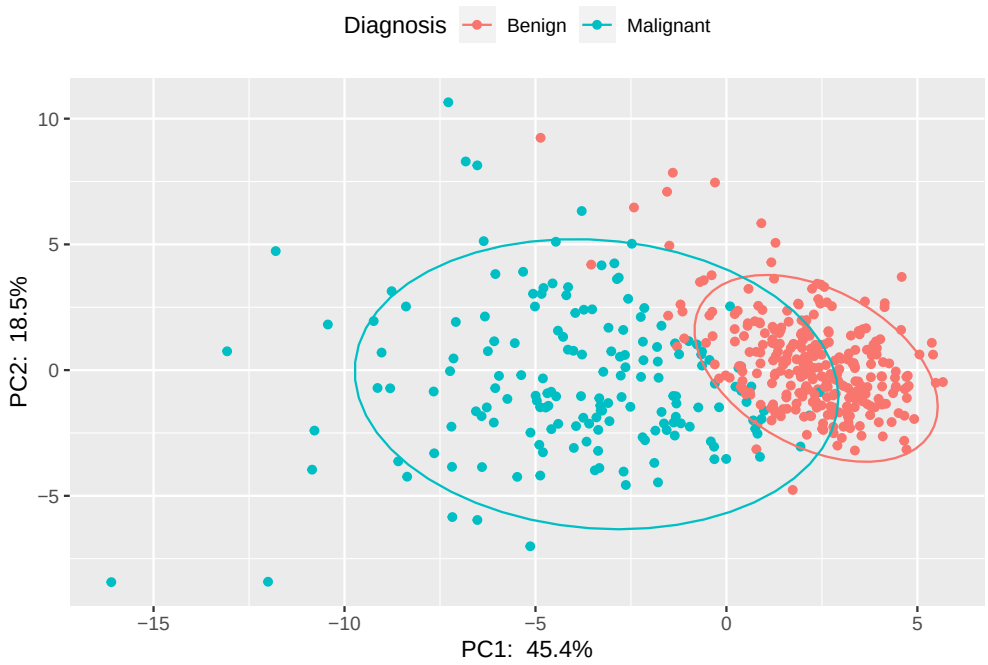

Figure 6: Scatter plot of PC1 and PC2 by diagnosis

4 Model building & evaluation

Several models are built starting with the simplest. The following key performance metrics for each model developed are stored in a specific data frame :

Table 6: Key performance metrics definitions

Metric	Definition
True Positive (TP)	Prediction correctly indicates the presence of a condition or characteristic
True Negative (TN)	Prediction correctly indicates the absence of a condition or characteristic
False Positive (FP)	Prediction wrongly indicates that a particular condition or attribute is present
False Negative (FN)	Prediction wrongly indicates that a particular condition or attribute is absent
Accuracy	Overall Accuracy, ratio of number of correct predictions to the total number of input samples
Sensitivity	True Positive Rate, proportion of Positive data points that are correctly considered as Positive, with respect to all Positive data points
Specificity	True Negative Rate, proportion of Negative data points that are correctly considered as Negative, with respect to all Negative data points
False Positive Rate (FPR)	Ratio between the number of Negative events wrongly categorized as Positive and the total number of actual Negative events
False Negative Rate (FNR)	Ratio between the number of Postive events wrongly categorized as Negative and the total number of actual Positive events
F1 Score (F1)	Harmonic mean of precision (or positive predictive value) and Sensitivity

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{3}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} = 1 \text{ - Specificity} \tag{4}$$

$$\text{FNR} = \frac{\text{FN}}{\text{FN} + \text{TP}} = 1 \text{ - Sensitivity} \tag{5}$$

$$\text{F1 Score} = \frac{2(\text{TP})}{2(\text{TP}) + \text{FP} + \text{FN}} \tag{6}$$

We used cross-validation, a way of ensuring robustness in the model at the expense of computation. In the ordinary modeling methodology, a model is developed on train data and evaluated on test data. In some extreme cases, train and test might not have been homogeneously selected and some unseen extreme cases might appear in the test data, which will drag down the performance of the model. On the other hand, in cross-validation methodology, data was divided into equal parts and training performed on all the other parts of the data except one part, on which performance will be evaluated. This process repeated as many parts user has chosen. For example in five-fold cross-validation, data will be divided into five parts, subsequently trained on four parts of the data, and tested on the one part of the data. This process will run five times, in order to cover all points in the data. Finally, the error calculated will be the average of all the errors. Cross-validation thus minimizes the risk of overfitting. It is also a useful technique for tuning parameters for those models that require it (for example, to tune the number of neighbours k to include in a k-nearest neighbours model). The caret package provides a convenient method for cross-validation that can be defined in advance, using the **trainControl** function and applied to each model as required. Here, the train control parameters were set to apply 10-fold cross-validation, repeated 10 times.

4.1 Random sampling

4.1.1 Simple Random Sample

The first and simplest model to be tested involved random sampling with equal probability for each outcome, exactly like a perfect coin toss. This method is straightforward and easy to implement, but not very suitable for cases like ours with imbalance in prevalence of benign and malignant samples.

4.1.2 Weighted Random Sample

So, we built a weighted random sample model where the prevalence of each class of samples was used to define the probability of each outcome within the random sample.

4.2 Unsupervised Learning

Similar to the teacher-student analogy, in which the instructor does not present and provide feedback to the student and who needs to prepare on his/her own.

4.2.1 *k*-Means Clustering

k-Means Clustering is the most commonly used unsupervised machine learning algorithm for partitioning a given data set into a set of k groups (k = 2 clusters in our case), where k represents the number of groups pre-specified by the analyst. It classifies objects in multiple groups, such that objects within the same cluster are as similar as possible (high intra-class similarity), whereas objects from different clusters are as dissimilar as possible (low inter-class similarity). In k-means clustering, each cluster is represented by its center (centroid) which corresponds to the mean of points assigned to the cluster.

We can compute k-means in R with the **stats::kmeans** function. Here we grouped the data into two clusters (centers = 2). The kmeans function also has an **nstart** option that attempts multiple initial configurations and reports on the best one. For example, adding **nstart = 25** will generate 25 initial configurations. This approach is often recommended, and helps to introduce stability to the model outcome given that it can be sensitive to the fact that the initial centre is chosen at random.

In practical applications, usually business should be able to provide what would be approximate number of clusters they are looking for. But what if we don't know the number of clusters? Preferably we would like to use the optimal number of clusters. **Elbow Method** is one of the most popular methods for determining these optimal clusters. The basic idea behind cluster partitioning methods, such as k-means clustering, is to define clusters such that the total intra-cluster variation (known as total within-cluster variation or total within-cluster sum of square) is minimized. Thus, we can use the following algorithm to define the optimal clusters:

- compute clustering algorithm (k-means clustering) for different values of k. For instance, by varying k from 1 to 20 clusters,
- for each k, calculate the total within-cluster sum of square (wss),
- plot the curve of wss according to the number of clusters k,
- the location of a bend (knee) in the plot is generally considered as an indicator of the appropriate number of clusters.

The results suggest that **2** is as expected the optimal number of clusters, as it appears to be the bend in the knee (or elbow).

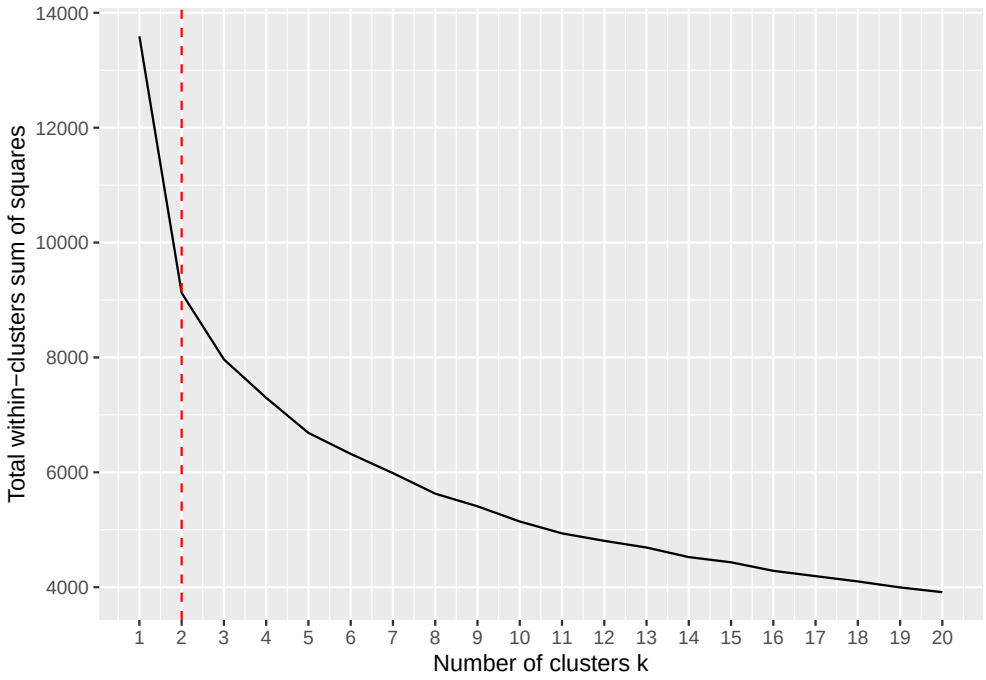

Figure 7: Elbow Method

Two version of the k-means model were developed. The first used the normalized data from the full training dataset. The second selected out those features which were highly correlated, 10 features

where the correlation coefficient exceeded the 0.9 cutoff defined in the exploratory analysis.

4.3 Supervised learning

This is where an instructor provides feedback to a student on whether they have performed well in an examination or not. In which target variable do present and models do get tune to achieve it. Many machine learning methods fall in to this category. Supervised machine learning models can be classified into **discriminative** and **generative** models. In simple words, a discriminative model makes predictions based on conditional probability and is either used for classification or regression. On the other hand, a generative model revolves around the distribution of a dataset to return a probability for a given example.

4.3.1 Generative modelling

Generative models are supervised machine learning techniques that model how the entire dataset, including both predictors and outcomes are distributed and use the joint probability distribution in order to predict the conditional probability of one outcome or another. The most general generative model is the Naive Bayes model which is based on the Bayes rule, where $f_{\mathbf{X}|Y=1}$ and $f_{\mathbf{X}|Y=0}$ are the distribution functions of the predictor $\mathbf{X}$ with binary outcomes, $Y = 1$ and $Y = 0$.

$$p(\mathbf{x}) = \Pr(Y = 1|\mathbf{X} = \mathbf{x}) = \frac{f_{\mathbf{X}|Y=1}(\mathbf{x}) \Pr(Y = 1)}{f_{\mathbf{X}|Y=0}(\mathbf{x}) \Pr(Y = 0) + f_{\mathbf{X}|Y=1}(\mathbf{x}) \Pr(Y = 1)} \tag{7}$$

The Naive Bayes model assumes that all features within the dataset are equally important and independent. Whilst this is a naive assumption that is unlikely to be true for a given set of data, it is typically good enough for the purposes of classification. The **nb** method provided in the caret package includes three tuning parameters each of which was tuned using resampling during cross-validation of this model: laplace smoothing (0 or 1), kernel distribution (true or false) and adjustment (0 or 1).

Other generative models include Linear Discriminative Analysis (LDA) and Quadratic Discriminative Analysis (QDA). LDA has the benefit of serving to reduce the dimensionality of the data (similarly to PCA) and to classify the data for predictive purposes. LDA assumes that the data are normally distributed and that the correlation structure is the same for all classes.

On the other hand QDA assumes that the distributions are multivariate normal, and cannot be used for dimension reduction but is more useful than LDA where different classes are known to exhibit distinct co-variances. Each of these models were developed to train the normalised train data and predict the outcome using the normalized test data. In addition, the QDA model was also run incorporating the outputs from PCA via caret's pre-processing functionality.

4.3.2 Discriminative modelling

The discriminative model is used particularly for supervised machine learning. Also called a conditional model, it learns the boundaries between classes or labels in a dataset. It creates new instances using probability estimates and maximum likelihood. However, they are not capable of generating new data points. The ultimate goal of discriminative models is to separate one class from another.

4.3.2.1 Logistic Regression Model Logistic regression is one of the most popular Supervised Learning technique which is used for predicting a categorical dependent variable using a given set of independent variables. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, True or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

Logistic regression is the most commonly used form of generalised linear model (GLM), used for solving classification problems. In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1). The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, etc.

4.3.2.2 k-Nearest Neighbors Model k-Nearest Neighbors is one of the simplest Machine Learning algorithms which assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. kNN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using kNN algorithm.

The k in kNN is a parameter that refers to the number of nearest neighbors to include in the majority of the voting process. There is no particular way to determine the best value for k, so we need to try some values to find the best out of them. A very low value for k such as k=1 or k=2, can be noisy and lead to the effects of outliers in the model. As with the use of bins in smoothing, larger values of k result in smoother estimates. k is a tuning parameter within the train function for the kNN model, and cross-validation within the training set was used to tune a value for k between 1 and 30 in increments of 1 to optimise the model before using it to predict outcome in the testing set.

4.3.2.3 Random Forest Model Many Machine Leaning algorithms suffer from diminished performance due to multidimensionality of data. As seen in previous sections, PCA can be useful to reduce the number of dimensions required as part of pre-processing of data prior to training an algorithm. Decisions trees are another way to address this issue, effectively partitioning the data such that final predictions can be made on a smaller subset of predictors.

Random Forest is a supervised Machine Learning algorithm that is constructed from decision tree algorithms, to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems.

Random Forest consists of many decision trees. The **Forest** generated is trained through bagging or bootstrap aggregating.

Random Forest establishes the outcome based on the predictions of the decision trees. It predicts by taking the average or mean of the output from various trees. Increasing the number of trees increases the precision of the outcome.

Random Forest eradicates the limitations of a decision tree algorithm. It reduces the overfitting of datasets and increases precision. The **rf** method within the caret package was used and the number of randomly selected predictors to include in each decision tree was tuned within the train set using the **mtry** tuning parameter via cross-validation. Other random forest methods also allow for tuning of the minimum number of data points to include in each decision tree but were not utilised here.

4.3.2.4 Neural Networks Model Another option for dealing with multidimensional data, and particularly with non-linearity, is Neural Networks Modelling. As such, this type of algorithm has found particular application in dealing with complex machine learning tasks such as image processing, particularly with the multi-layer Neural Networks. The more layered the network, however, the more computational resource it requires and the greater the risk of over-fitting to a training data set. The simplest forms of Neural Network, known as single-layer Neural Networks, are only equipped to deal with linear data. These algorithms take multidimensional inputs (x_i), apply a weighting (w_i) before summing them in order to classify the output y_i.

$$y_i = \sum_i w_i x_i \tag{8}$$

We built our model using the single hidden layer Neural Network method, **nnet** that is available within the caret package. Unlike the simplest Neural Network this model has three layers and, consequently, can handle non-linear data. Whilst tuning parameters to optimise the number of hidden units (size) and weight decay (decay) are available with this method, they were not deployed here.

5 Results

The table below provides the key performance metrics for each of the models tested in the project.

Table 7: Key performance metrics per model

Model	Accuracy	Sensitivity	Specificity	F1	FNR	FPR
Random Sampling						
Simple Random Sample	0.565	0.628	0.528	0.519	37.2%	47.2%
Weighted Random Sample	0.496	0.140	0.708	0.171	86.0%	29.2%
Unsupervised Learning						
K-Means Clustering	0.887	0.860	0.903	0.851	14.0%	9.7%
K-Means Clustering Without Highly Correlated Features	0.243	0.372	0.167	0.269	62.8%	83.3%
Supervised Learming (Generative Models)						
Naive Bayes	0.922	0.930	0.917	0.899	7.0%	8.3%
Linear Discriminant Analysis	0.957	0.884	1.000	0.938	11.6%	0.0%
Quadratic Discriminant Analysis	0.965	0.977	0.958	0.955	2.3%	4.2%
Quadratic Discriminant Analysis with PCA	0.974	0.953	0.986	0.965	4.7%	1.4%
Supervised Learming (Discriminative Models)						
Logistic Regression	0.965	0.907	1.000	0.951	9.3%	0.0%
Logistic Regression With PCA	0.991	0.977	1.000	0.988	2.3%	0.0%
K-Nearest Neighbors	0.974	0.930	1.000	0.964	7.0%	0.0%
Random Forest	0.974	0.953	0.986	0.965	4.7%	1.4%
Neural Networks	0.991	0.977	1.000	0.988	2.3%	0.0%

Overall, specificity will be easier to achieve with this data than sensitivity. **5** models achieved a specificity of **1**, but only **0** of these achieved the same level of sensitivity.

With the worst performance, **Random Sampling (Simple and Weighted)** is definitely not an effective method for predicting diagnosis. The results from these two models reinforce the importance of the F1 Score in evaluating performance of models dealing with imbalanced datasets. Simple Random sampling and Weighted Random Sampling resulted in F1 Scores of 0.519 and 0.171 respectively.

k-Means Clustering improved accuracy, with an F1 Score of 0.851 but still not to an acceptable level, with a False Negative Rate of 14% and a False Positive Rate of 10%. Of note, removing the highly correlated features (those with a correlation coefficient above 0.9) from the dataset reduced the accuracy of the k-Means Clustering model, yielding a reduced F1 Score of 0.269 and reducing both the specificity and, in particular, the sensitivity of the model.

Supervised Learning techniques substantially improved the accuracy of predictions, with each subsequent model achieving an overall accuracy of more than **0.90**. Amongst the generative models, **Naive Bayes** achieved an overall accuracy of and an F1 Score of with a balance of sensitivity and specificity. Of note, the Naive Bayes model performed best when the usekernel parameter was set to TRUE and the associated adjustment was set to 1, which indicates that a normal Gaussian distribution is not the best way to estimate the conditional probabilities (Figure 8)

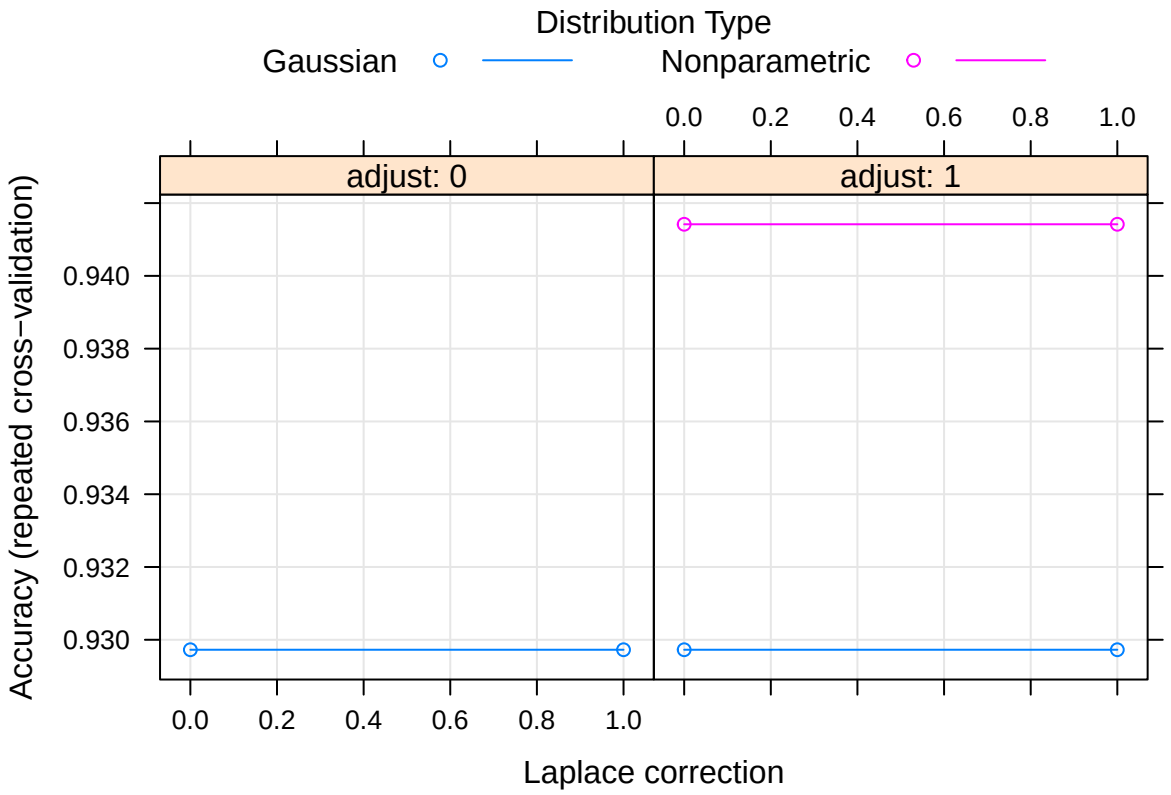

Figure 8: Tuning results for Naive Bayes Model during cross-validation

Linear Discriminant Analysis and Quadratic Discriminant Analysis improved on the performance of the Naive Bayes model, with F1 Scores of 0.938 and 0.955 respectively. The LDA model achieved a specificity of 1 (FPR of 0%) but this was offset by reduced sensitivity (0.884), an unacceptable FNR of 12%.

The QDA model delivered a better balance between FNR (2%) and FPR (4%). Of note, dimension reduction through pre-processing the training data with PCA improved the specificity of the QDA model, reducing the FPR to 1% but it reduced the sensitivity, increased the FNR to 5%.

The discriminative models of supervised learning were the best performing models with this dataset. Logistic Regression achieved an overall accuracy of 0.965. This was improved further to 0.991 with dimension reduction using PCA by improving the sensitivity of the model, achieving FNR and FPR of only 2% and 0% respectively.

k**NN** model performed best when the number of neighbours, k, was defined as 3 (see Figure 9). On this basis, the overall accuracy of 0.974 but with lower sensitivity (0.93).

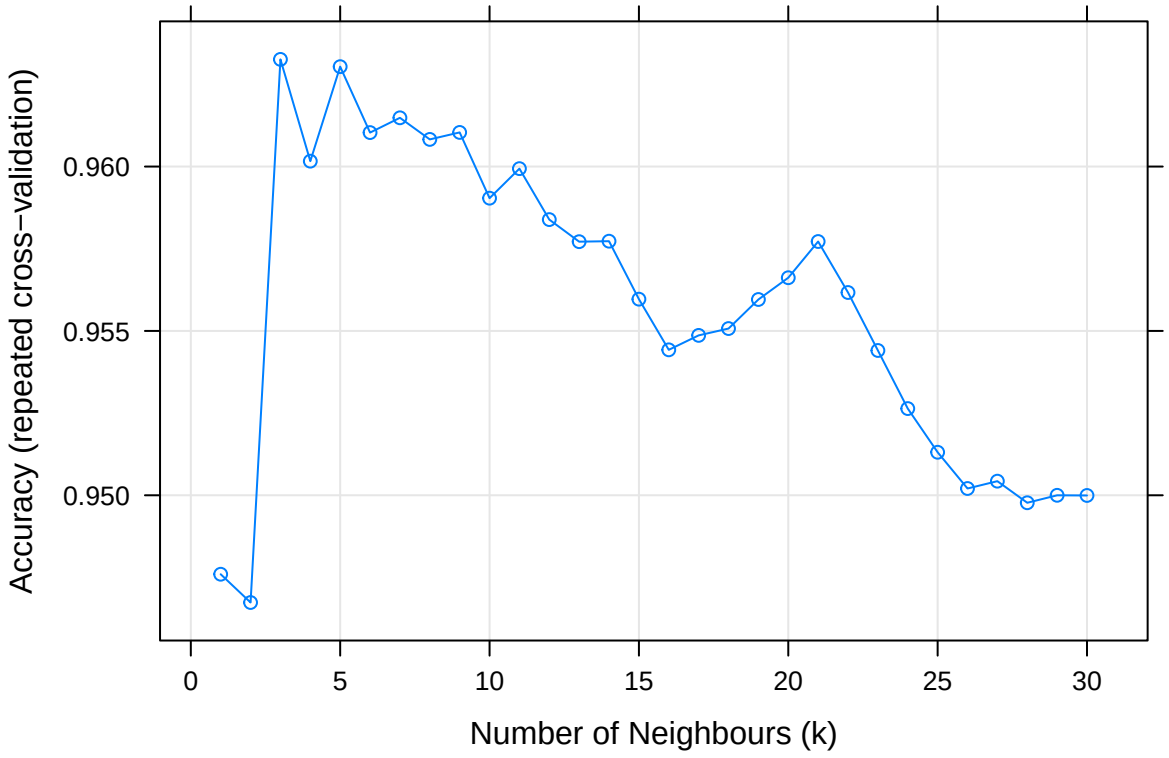

Figure 9: Tuning results for KNN Model during cross-validation

Random Forest model was not very sensitive to the number of randomly selected predictors included in each decision tree it was tuned for, but performed marginally best when mtry was 15 (see Figure 10), with overall accuracy of 0.974, sensitivity of 0.953 and specificity of 0.986.

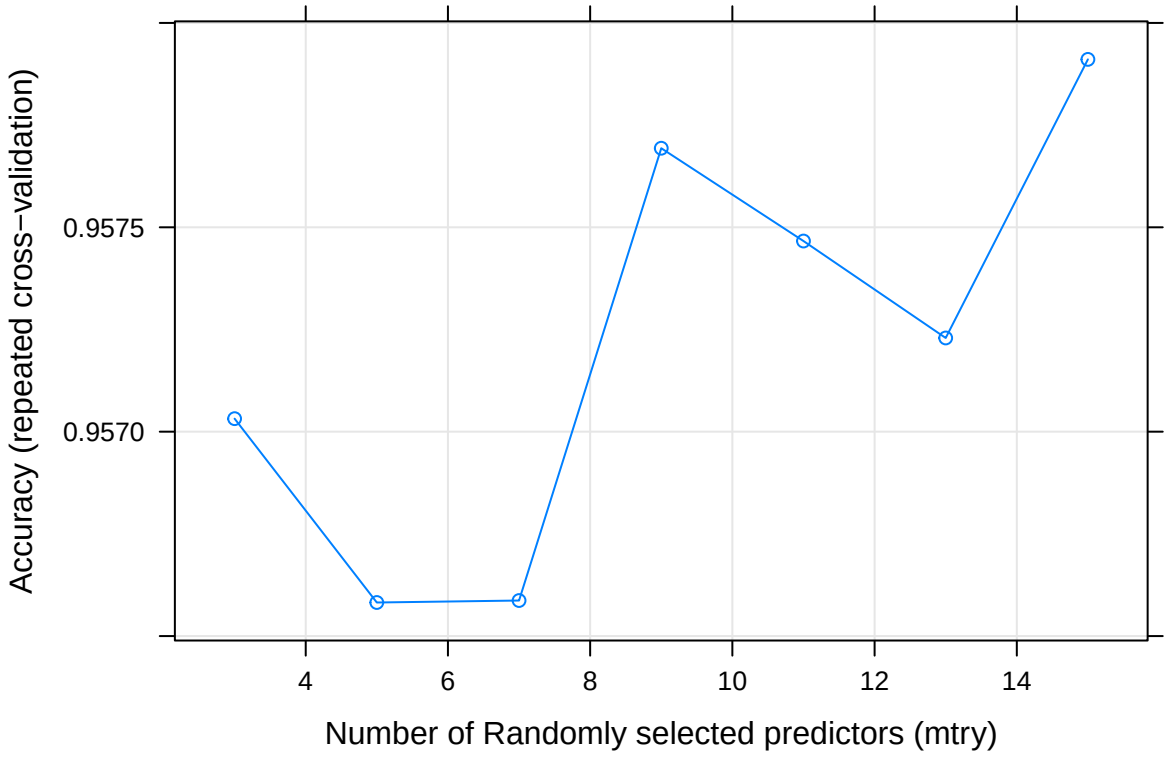

Figure 10: Tuning results for random forest model during cross-validation

The final individual model to be trained and tested was the **Neural Networks**. This performed very well with an overall accuracy of 0.991 and an F1 Score of 0.988.

6 Conclusion

An accurate and timely diagnosis of breast cancer is still a major problem for proper treatment in the healthcare field. The precise analysis of cancer features is still a time-consuming and challenging task. In this project, several Machine Learning predictive models were developped to detect breast cancer tumors and classify them into benign and malignant tumors. We tried to improve the prediction model's performance with a maximum F1 Score and a highest accuracy score possible.

We selected the optimal model by selecting a high accuracy/F1 Score level combinated with a low rate of false negatives. **Neural Networks** and **Logistic Regression With PCA** both come first with the optimal results (Accuracy = 0.991, F1 Score = 0.988, FNR = 2.3%). These results confirm the potential of Machine Learning to accurately predict diagnosis of breast cancer using samples obtained via FNA biopsy, with high levels of both sensitivity and specificity.

7 Appendix - Environment

```
## [1] "Operating System:"
```

```
##                _
## platform       x86_64-w64-mingw32
## arch           x86_64
## os             mingw32
## crt            ucrt
## system         x86_64, mingw32
## status
## major          4
## minor          2.1
## year           2022
## month          06
## day            23
## svn rev        82513
## language       R
## version.string R version 4.2.1 (2022-06-23 ucrt)
## nickname       Funny-Looking Kid
```

8 References

Corinne Hahn, **S M** 2016 *Méthodes statistiques appliquées au management. Eyrolles.* Available at https://www.eyrolles.com/Sciences/Livre/methodes-statistiques-appliquees-au-management-9782326001336/

Dangeti, **P** 2017 *Statistics for machine learning : Build supervised, unsupervised, and reinforcement learning models using both python and r. Packt Publishing Ltd.* Available at https://www.packtpub.com/product/statistics-for-machine-learning/9781788295758

Irizarry, **R A** 2022 *Introduction to data science: Data analysis and prediction algorithms with r. Github Sites.* Available at https://rafalab.github.io/dsbook/

Kuhn, **M** 2019 The caret package. *Github Sites.* Available at https://topepo.github.io/caret/index.html

Wikipedia 2022 Cancer du sein. *Wikipedia.* Available at https://fr.wikipedia.org/wiki/Cancer_du_sein