Cancer is the second leading cause of deaths in the world. Breast cancer is the most commonly occurring cancer in women (WHO, 2018). In 2013 breast cancer had the highest incidence among women, with an estimated number of 1.8 million new cases, leading to around 471,000 deaths in the same year (Thompson, 2015).

The exact cause of the breast cancer remains unknown, early detection and diagnoses are considered highly important in determining the success of treatment and improving survival from breast cancer (ACS, 2015). Mammograms produce images of the inside of the breasts using low-dose X-rays and detect cancer in the preclinical phase when it is small and impalpable (NICE, 2017). Around 5-10% of the mammograms do not produce conclusive results (Kopans, 1992). Biopsy is the gold standard for pathological diagnosis of breast cancer (Zhang, 2013). However, only 15–30% of biopsies performed on suspicious masses found on mammogram prove to be malignant ([Hall, 1988](https://www-sciencedirect-com.libproxy.ucl.ac.uk/science/article/pii/S0957417410012054#b0040); Kopans, 1992).

To reduce the number of unnecessary biopsies and the associated cost and furthermore, the mental and physical discomfort caused to many patients, research should be focused on trying to find more accurate ways of identifying patients that should be referred for a biopsy. Data mining algorithms can be utilised to assist physicians in making decisions about whether to perform a breast biopsy on a suspicious lesions seen on a mammogram image or whether to offer a follow up examination instead (Mendelson, 2019).

The mammographic dataset investigated in this study is from the UCI Machine Learning Repository. The dataset includes BI-RADS attributes for 961 full field digital mammograms that were collected at the Institute of Radiology of the University Erlangen-Nuremberg between 2003 and 2006. Each instance has got a target label that offers information about whether the mass sample was classified as benign (0) or malignant (1). 516 or the reported cases were benign and 445 were malignant. The BI-RADS assessment rating is non-predictive and will be removed during the data-pre-processing stage (ACR, 2013). The descriptive statistics of each of the features are shown in Table 1. The correlation matrix reveals that an irregular mass shape has the highest correlation to the class label; age is also highly correlated to the class. The correlation matrix further indicates that a round mass shape is also highly correlated to the mass having a circumscribed margin. The 3D plot of the features reveals that malignant cases tend to have a lobular or irregular mass shape and an obscured, ill-defined or speculated mass margins as well as an increased density.

The mammographic mass dataset gives us information about the predictive BI-RADS attributes and the associated response represented as a class label for every patient. Having an associated response for every observation means that we can approach this as a supervised learning problem. The aim is to fit a model that accurately determines the relationship between the BI-RADS attributes and whether the mass is malignant or benign.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Attribute | Attribute type | Min value | Max  value | Mean value | Mode value | Original data type | Number of missing values |
| BI-RADS Assessment | Ordinal/ Non- predictive | 1 | 5 | 4.35 | 4 | Object | 2 |
| Age | Ordinal continuous | 18 | 96 | 55 | 59 | Integer | 5 |
| Mass shape | Nominal:   * Round - 1 * Oval - 2 * Lobular - 3 * Irregular - 4 | 1 | 4 | n/a as nominal feature | 4 | Object | 31 |
| Mass margin | Nominal:   * Circumscribed - 1 * Microlobulated - 2 * Obscured - 3 * ill-defined - 4 * speculated - 5 | 1 | 5 | n/a as nominal feature | 1 | Object | 48 |
| Mass density | Ordinal:   * Extremely dense -1 * Heterogeneously dense -2 * Scattered areas of fibroglandular density - 3 * Almost entirely fatty -4 | 1 | 4 | 2.91 | 3 | Object | 76 |

**Table 1.** Overview of the descriptive statistics of the predictive and non-predictive features.

Before applying a supervised learning model to the dataset, we need to ensure that any data quality issues are identified and addressed.

The mammographic mass dataset has 160 attribute values missing; this is about 16% of the data. Removing all rows with missing values will lead to losing 125 rows of valuable information. Imputation would allow any missing values to be replaced with plausible estimated values. The Patient age values were imputed with the mean age value. However, imputing can also introduce bias and reduce the power of the prediction, therefore, the other missing values will be removed (Azur, 2011).

Class distribution is balanced with around 51% of the tumours benign and 49% malignant, if the distribution was skewed, undersampling could be used to remove some of the overrepresented instances.

All input values are standardised to have a mean of zero and standard deviation of one. This allows all inputs to be treated equally in the regularization process, and appropriate starting weights to be selected by the classifier(Hastie et al., 2009; Raschka, 2017).

The data is split with 80% of the data used for building the model and selecting the optimal hyperparameters. 20% of the data is saved for independently assessing the individual models.

Both Support Vector Machine (SVM) and Multilayer Perceptron (MLP) are algebraic machine learning models; hence they require numerical input vectors. Categorical nominal variables have been assigned numerical labels from 1-4, hence the LabelEncoder does not need to be used. However, classifiers believe these labels to be ordinal, meaning that the classifier considers mass shape irregular (4) to be greater than mass shape round (1). Using dummy variables each categorical nominal value is converted into a new column and assigned 0 or 1 (False/True) value. This allows the values to be assigned appropriate weighting by the classifier.

The Learning curve for both SVC and MLP shows that the models have high variance and are therefore likely to overfit when trained on the available training samples.

Selecting only meaningful features helps reduce overfitting and improves the generalization capabilities of the classifier. This dataset has 11 predictive features. Using Random Forrest Classifier, 5 of the most discriminative features are selected for building the models, these are age, circumscribed and speculated mass margin, irregular and oval mass shape. Feature extraction is another way to reduce the complexity of a model. Principal component analysis (PCA) is a feature extraction technique that finds the direction of the maximum variance by projecting the data onto a lower dimensional feature space (Raschka, 2017). Plotting explained variance against the number of principal components indicates that the first and second principal components account for around 50% and 18 % of the variance, respectively. Plotting the accuracy against the number of principal components shows that choosing either 1 or 2 principal components produces the best accuracy with the SVM model, selecting 1 principal component procures best accuracy with MLP. However, a better accuracy is achieved by using feature selection alone without PCA.

**Support Vector Classifier**

The Support Vector Classifier (SVC) is a supervised classification algorithm. Support Vector Machine (SVM) is an extension of SVC and uses kernels to enlarge the feature space (Pseudocode 1) (James 2013; Fung, 2015). For non-linearly separable data, the SVM maps the data to a higher dimensional feature space using a non-linear kernel function. SVM algorithm finds the optimal hyperplane that can separate all objects of one class from those in the other class with the largest margin. The optimal hyperplane is found by solving a convex quadratic optimization problem (Hastie et al., 2009). The training samples that fall on the margins of the hyperplane are called support vectors (Mokhtar, 2013). The hyperplane is built by a set of these support vectors, when the algorithm is applied on the test data, it determines which side of the hyperplane they fall on and subsequently suggests a class label (Fung, 2015; Jain, 2017).

**Pseudocode 1.** Pseudocode for training an SVM (Pedersen, 2006).

**Require:** X and y loaded with training labelled data, α ⇐ 0 or α ⇐ partially trained SVM

1: C ⇐ some value (10 for example)

2: **repeat**

3: **for all** {xi, yi}, {xj, yj} **do**

4: Optimize αi and αj

5: **end for**

6: **until** no changes in α or other resource constraint criteria met

**Ensure:** Retain only the support vectors (αi > 0)

### Both SVM and MLP can be tuned with GridSearchCV. In nested cross-validation, the “inner” cross-validation uses GridSearchCV to perform an exhaustive search over the specified hyperparameter values to find the best combination of hyperparameters for the classifier. Subsequently, the “outer” cross-validation independently evaluates the performance of the model (Halder & Ozdemir, 2018).

The performance of each classification model is evaluated using the statistical measures shown in Table 2.

|  |  |  |
| --- | --- | --- |
| Performance measure metric | Description | Formula |
| Accuracy (ACC) | Accuracy (ACC) offers information about how many samples were correctly labelled. ACC is the sum of all correct predictions (TP & TN) to all predictions. |  |
| Precision (PRE) | Precision shows the number positively predicted cases that were correct. Optimising for precision means tuning the parameters so that there the number of True Positive predictions is maximised. |  |
| Recall (REC) also known as True Positive Rate | Recall (REC) allows us to determine the sensitivity of the classification model and indicates the total number of malignant cases that were correctly identified. This is a very important measure when it comes to the Mammographic Mass dataset. As the cost of False Negatives is very high, because it would mean that someone who potentially has a cancerous mass is not referred for further investigation and treatment. |  |
| False Positive Rate (Specificity) | False Positive Rate shows the proportion of negative data points that are mistakenly considered as positive, with respect to all False Positive and True Negative predictions. |  |
| F1 - score | F1 - score is the combination of Precision and Recall. |  |
| ****Receiver Operating Characteristics (****ROC) curve & Area under the curve (AUC) | The area under the ROC curve shows the overall performance of a classifier. The greater the area under curve the better the classifier. AUC shows the how well the model can separate between the two classes. Ideally the ROC curve should fall as close as possible to the top left corner (James et al., 2013). The ROC curve is a plot of the True Positive Rate or the classifier against the False Positive Rate. ROC curve takes into account the entire range of different thresholds and hence is a considered a good measure of the classifier’s performance. |  |

**Table 2.** Performance measure metrics used for assessing classifiers models.

The following parameters can be tuned to optimise the performance of SVM: C is the regularization parameter and is usually set between 1 and 10, increasing the value improves the prediction accuracy, this can also lead to over-fitting. γ is the kernel coefficient by increasing γ the flexibility of the model increases and this can help improve accuracy; C and γ a regulate the trade-off between training errors and misclassifications. Linear kernel is used for linearly separable data and radial basis kernel (rbf) for non-linearly separable data.

The best values for these parameters in terms of the accuracy are 0.01 for C and 0.1 for gamma with an ‘rbf’ kernel, which resulted in 82.8% training accuracy and 75% test accuracy with an AUC of 80%. Table 3 shows the hyperparameters that gave the best results for the performance metrics described in Table 2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Results optimised for: | Best relevant score  on training set | Best relevant score on test set | C | kernel | Gamma ( γ) |
| Accuracy | Accuracy  0.828 +/- 0.044 | Accuracy  0.75000 | 0.01 | rbf | 0.1 |
| Precision | Precision  0.810 +/- 0.051 | Precision  0.73810 | 0.1 | rbf | 10 |
| Recall | Recall  0.893 +/- 0.062 | Recall  0.74405 | 0.1 | linear | n/a |
| F1 score | F1 score  0.832 +/- 0.044 | F1 score  0.75000 | 0.01 | rbf | 0.1 |

**Table 3**. SVM hyperparameter values optimised for the best accuracy, precision, recall and F1 score .

Confusion matrix (Image 1) helps us understand the performance of a learning algorithm, it reports the number of True positive (TP), True negative (TN), False positive (FP) and False negative (FN) predictions of a classifier. The aim is to improve the classifier so that the TP and TN predictions are maximised and the number of false predictions are reduced.



**Image 1.** Confusion Matrix (Narkhede, 2018).

**Multi-layer perceptron**

Artificial Neural Network (ANN) is a powerful predictive two stage model that can be used for both regression and classification problems (Pseudocode 2,3 & 4; Table 4). Multilayer perceptron (MLP) is a feedforward artificial neural network with an input layer, one or more hidden layers and an output layer. As data moves through the network in a feed-forward fashion, it is influenced by the connection weights and the activation function. There are several activation functions that are in use including linear, sigmoid, tanh and Rectified Linear Unit Function (ReLu) (Mahmood, 2018). The choice of activation function depends on the model used, the hyperparameters used for optimisation and the properties of the feature vector. The values for weights are changed progressively as the algorithm finds the optimal solution that best fits the data with the backpropagation learning algorithm (Patterson, 2018). The bigger the weight the higher the correlation between the input signal and the outcome. Backpropagation uses gradient descent on the weights of the connections between layers to minimize the error on the output of the network (Patterson, 2018; Hastie et al., 2009). The weights are adjusted in the direction that most decreases the error. Output errors are propagated backwards to come up with the errors for the hidden layers. The gradient is computed by a forward and backward sweep over the whole network (Grus, 2005; Hastie et al., 2009).

##### **Pseudocode 2. General neural network training pseudocode (Patterson, 2018).**

function neural-network-learning (training-records) returns network

network <- initialize weights (randomly)

start loop

**for** each example in training-records **do**

network-output = neural-network-output (network, example)

actual-output = observed outcome associated with example

update weights in network based on

{example, network-output, actual-output}

end **for**

end loop when all examples correctly predicted or hit stopping conditions

**return** network

|  |  |
| --- | --- |
| Notation | Meaning |
| *i* | Index of artificial neuron |
| n i | Neuron at index *i* |
| *j* | Index of neuron in previous layer connecting to neuron *i* |
| a i | Activation value of neuron *i* (output of neuron *i*) |
| A*i* | Vector of activation values for the inputs into neuron *i* |
| *g* | Activation function |
| *g’* | Derivative of the activation function |
| E r r i | Difference between the network output and the actual output value for the training example |
| W*i* | Vector of weights leading into neuron *i* |
| W j, i | Weight on the incoming connection from previous layer neuron *j* to neuron *i* |
| i n p u t \_ sum i | Weighted sum of inputs to neuron *i* |
| i n p u t \_ sum j | Weighted sum of inputs for neuron *j* in previous layer (used in backpropagation) |
| α | Learning rate |
| Δ j | Error term for connected neuron *j* in previous layer |
| Δ i | Error term for neuron *i*;   = E r r i × g ' ( i n p u t \_ s u m i ) |

**Table 4.** Neural network notation for Pseudocode 3 (Patterson, 2018).

**Pseudocode 3.** Backpropagation algorithm for updating weights (Patterson, 2018).

function backpropagation-algorithm

(network, training-records, learning-rate) returns network

network <- initialize weights (randomly)

start loop

for each example in training-records do

  // compute the output for this input example

network-output <- neural-network-output (network, example)

// compute the error and the [delta] for neurons in the output layer

example\_err <- target-output - network-output

  // update the weights leading to the output layer

Wj,i ← Wj,i + α x aj × E r r i × g’ (input\_sum i)

for each subsequent-layer in network do

  // compute the error at each node

Δ j ← g’ (input\_sum j) Σ i W j,i Δ i

  // update the weights leading into the layer

W k, j ← W k, j + α × ak × Δ j

end for

end for

end loop when network has converged

return network

**Pseudocode 4.** Pseudocode for neural network training (Grus, 2005).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| def feed\_forward( neural\_network, input\_vector): | | | | | | | |
|  | #takes in a neural network (represented as a list of lists of weights) | | |
|  | and returns the output from forward-propagating the input | | |
|  |  | | |
|  | outputs = [] | | |
|  |  | | |
|  | #process one layer at a time  for layer in neural\_network: | | |
|  | | |  | | | | |
|  | | | | | | | |
| |  | | --- | | Inpu with\_bias = input\_vector + [1] # add a bias input | | output = [neuron\_output (neuron, input\_with\_bias) for neuron in layer] | | # compute the output for this layer | | outputs.append(output) # and remember it | |  | | # then the input to the next layer is the output of this one | | input\_vector = output | |  | | return outputs |   def backpropagate (network, input\_vector, target): | | | | | | | |
|  | | | | | | | |
|  | hidden\_outputs, outputs = feed\_forward (network, input\_vector) | | | |
|  |  | | | |
|  | # the output \* (1 - output) is from the derivative of sigmoid | | | |
|  | output\_deltas = [output \* (1 - output) \* (output - target[i]) | | | |
|  | for i, output in enumerate(outputs)] | | | |
|  |  | | | |
|  | # adjust weights for output layer (network[-1]) one neuron at a time | | | |
|  | for i, output\_neuron in enumerate(network[-1]):  #focus on the ith output layer neuron | | | |
|  | for j, hidden\_output in enumerate(hidden\_outputs + [1]): | | | |
|  | | | #adjusts the jth weight based on both this neuron’s delta and  its jth output  output\_neuron[j] -= output\_deltas[i] \* hidden\_output | | | | |
|  | | |  | | | | |
|  | # back-propagate errors to hidden layer | | | | |
|  | hidden\_deltas = [hidden\_output \* (1 - hidden\_output) \* | | | | |
|  | | | dot(output\_deltas, [n[i] for n in network[-1]]) | | | | |
|  | | | for i, hidden\_output in enumerate(hidden\_outputs)] | | | | |
|  | | |  | | | | |
|  | | # adjust weights for hidden layer (network[0]) one neuron at a time | | | | |
|  | | for i, hidden\_neuron in enumerate(network[0]): | | | | |
|  | | | for j, input in enumerate(input\_vector + [1]): | | | | |
|  | | | hidden\_neuron[j] -= hidden\_deltas[i] \* input | | | | |

The following hyperparameters can be tuned to improve the performance of the MLP Classifier:

Hidden layers capture the nonlinear properties of the data and increase the flexibility of the model. The number of hidden units usually falls between 5 and 100. More layers can be used for bigger datasets. Cross-validation and GridSearchCV can be used to estimate the optimal number of hidden layers at different levels of hidden layers. The hidden layer sizes of (50,50,50) gave the best results for accuracy.

Momentum for gradient descent prevents the learning algorithm from getting stuck thereby helping to continue the path towards local minimum. Increasing the momentum can help improve accuracy.

Learning rate controls the extent by which weights are adjusted during optimisation. A large learning rate coefficient (e.g. 1) makes big adjustments, whereas a small learning rate (e.g. 0.00001) ensures that smaller adjustments are made. Large leaps can save time but may also overshoot the local minimum. A smaller learning rate is likely to eventually reach the local minimum, but can take a long time to run.

Validation curve with differing values of initial learning rate and momentum shows that the best accuracy is achieved with an initial learning rate value of 0.0001 and momentum value of 0.3.

Learning rate can be kept constant at the rate that was initially specified. Adaptive learning rate keeps the learning rate constant as long as the training loss is reducing. Inv-scaling can be used to gradually decrease the learning rate. Constant learning rate produced the best accuracy score.

The solver refers to the method by which the weights are optimised, ‘sgd’ refers to stochastic gradient descent and ‘adam’ refers to a stochastic gradient-based optimizer proposed by Kingma, Diederik, and Jimmy Ba. The ‘adam’ solver produced the best accuracy, whereas the ‘sgd’ solver was better for optimising for precision, recall and F1.

Alpha is the L2 regularization coefficient; regularization helps prevent over fitting by regulating the extent to which the hyperparameters are adjusted over time. Alpha value of 1 produced the best results in terms of accuracy. Table 5 shows the hyperparameters that gave the best results for the performance metrics described in Table 2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Results optimised for: | Best relevant score  on training set | Best relevant score on test set | Activation function | alpha | Hidden layer sizes | Learning rate | Solver |
| Accuracy | Accuracy  0.822 +/- 0.048 | Accuracy  0.75000 | tanh | 1 | (50, 50, 50) | constant | adam |
| Precision | Precision  0.819 +/- 0.059 | Precision  0.74405 | tanh | 0.0001 | (100, 150, 100) | constant | sgd |
| Recall | Recall  0.715 +/- 0.085 | Recall  0.70833 | ReLu | 1 | (100,100,100) | constant | sgd |
| F1 score | F1 score  0.825 +/- 0.042 | F1 score  0.75000 | tanh | 0.0001 | (75, 75, 75) | constant | sgd |

**Table 5**. MLP hyperparameter values optimised for the best accuracy, precision, recall, f1 score and AUC.

Both the SVM and MLP both perform well with AUC values of 0.80 and 0.81 and training/test accuracy values of 82.8%/75% and 82.2%/75%, respectively. The confusion matrices for the best performing SVM and the MLP Classifiers are shown in Table 5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Classifier** |  | | **Actual values** | |
| **SVM** | **Predicted values** |  | **Benign (0)** | **Malignant (1)** |
| **Benign (0)** | 66 | 20 |
| **Malignant (1)** | 22 | 60 |
| **MLP** |  | **Benign (0)** | **Malignant (1)** |
| **Benign (0)** | 67 | 19 |
| **Malignant (1)** | 23 | 59 |

**Table 6.** Confusion matrixes for MLP Classifier and SVM Classifier.

Recent progress in the field of data mining has impelled the work on developing machine learning approaches that can assist clinicians in their decision making process. Two different classification models, Support Vector Machine and Multi-Layer Perceptron, were used to predict whether masses found on mammograms are malignant or benign based on the age of the patient, the density, shape and margins of the mass. The models were optimised using PCA, GridSearchCV and validation curves. Both of the optimised models made a high number of false negative and false positive predictions on the test set, the rate of which is unacceptable in a clinical setting. More work in needed to improve the performance of the model to a level where healthcare professionals would accept the use of these models in their decision making process. Using convoluted neural network or a multi-layer perceptron with a ‘poly’ kernel could also be explored to see if the accuracy of the model can be improved.

There are several studies that have applied various machine learning algorithms to the Mammographic Mass Dataset. Mokhtar et al. (2013) applied Decision Tree, Artificial Neural Network, and Support Vector Machine algorithms and were able to achieve training and test accuracies of 81.43%/78.13%, 81.13%/83.43% and 83.66%/81.25%, respectively. Keles et al. (2011) used neuro-fuzzy rules in developing an Ex-DBC expert system for predicting breast cancer using the same dataset and were able to achieve a positive predictive rate of 96% and specificity of 97%. Elsayad (2010) evaluated the performance of two Bayesian networks, the tree augmented Naïve Bayes (TAN) and Markov blanket estimation (MBE) as a classifier on this dataset and compared the outcome to a multi-layered neural network classifier. The classifiers achieved the following respective training and test accuracies 87.07%/84.72%, 89.45%/87.85% and 82.02%/82.64%. Guzel et al. (2013) imputed the missing values with the k Nearest Neighbours algorithm and Naïve Bayes. Subsequently, they applied the same learning algorithms to the updated dataset and were able to achieve training and test accuracies of 82.49% and 81.69%, respectively. Halawani et. al. (2012) assessed the performance of various clustering algorithms and were able to develop an Expectation Maximization Algorithm, Bagging Algorithm and Ahmad & Dey algorithm with respective test accuracies of 78.9%, 80.7% and 78.5%.

Mammogram images are also affected by noise, in addition low contrast and blurry contours can lead to some of the tumour characteristics being very hard to detect or be misclassified or undetected. Healthcare professionals use ultrasound and MRI imaging in addition to mammograms in their decision making process which improves the accuracy of their predictions. For the machine learning model to be accepted as a reliable tool, the reasoning process that the model uses should be easily interpretable. However, the intricate details in which the SVM and especially the MLP make their estimations and predictions are not easily understandable. This can lead to resistance from healthcare professionals to utilising these models in their practice. For conclusive evidence on the reliability and accuracy of these models further larger prospective clinical studies will need to be carried out.

**References:**

American Cancer Society (ACS). 2015. Breast Cancer Facts & Figures 2015-2016. [Online]. [Accessed 29 March 2019]. Available from: <https://www.cancer.org/content/dam/cancer-org/research/cancer-facts-and-statistics/breast-cancer-facts-and-figures/breast-cancer-facts-and-figures-2015-2016.pdf>

American College of Radiology (ACR). 2013. BI-RADS Atlas — Reporting System. [Online]. [Accessed 29 March 2019]. Available from: <https://www.acr.org/-/media/ACR/Files/RADS/BI-RADS/Mammography-Reporting.pdf>

Azur, M.J. et al., 2011. Multiple imputation by chained equations: what is it and how does it work? *International journal of methods in psychiatric research,* 20(1), pp.40–9.

Elsayad, A.M., 2010. Predicting the severity of breast masses using Bayesian networks. *2010 The 7th International Conference on Informatics and Systems (INFOS)*, pp.1–9.

Elter, M., Schulz-Wendtland, R. & Wittenberg, T., 2007. The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process. Medical Physics, 34(11), pp.4164–4172.

Fung, Germaine et al., 2015. Distinguishing bipolar and major depressive disorders by brain structural morphometry: a pilot study. *BMC Psychiatry*, 15(298), p.298.

Grus, J. 2015. *Data science from scratch : first principles with python*, Sebastopol: O'Reilly Media.

Güzel C., Kaya M. & Yıldız O. Hasan Ş.B. 2013. Breast Cancer Diagnosis Based on Naïve Bayes Machine Learning Classifier with KNN Missing Data Imputation. [Online]. [Accessed 18 April 2019]. Available from: [www.awer-center.org/pitcs](http://www.awer-center.org/pitcs)

Halawani, S.M., Alhaddad, M., Ahmad, M. 2012. A study of digital mammograms by using clustering algorithms. Journal of Scientific and Industrial Research. 71, pp.594- 600.

Halder, S. & Ozdemir, S. 2018. *Hands-On Machine Learning for Cybersecurity.* Birmingham: Packt Publishing

Hastie, T. et al. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* Second. New York: Springer.

James, G., Witten, D., Hastie, T., Tibshirani, R. 2013. *An introduction to statistical learning : with applications in R.* London: Springer.

Jain, R. 2017. *Simple Tutorial on SVM and Parameter Tuning in Python and R*. [Online]. [Accessed 04 April 2019]. Available from: <https://www.hackerearth.com/blog/machine-learning/simple-tutorial-svm-parameter-tuning-python-r/>

Keles, A., Keleş, A., Yavuz, U. 2011. Expert system based on neuro-fuzzy rules for diagnosis breast cancer. *Expert Systems With Applications*, 38(5), pp.5719–5726.

Kopans, D.B. 1992. The positive predictive value of mammography. *AJR. American journal of roentgenology*, 158(3), pp.521–6.

Mahmood, H. 2018. Activation Functions in Neural Networks. [Online]. [Accessed 04 April 2019]. Available from: <https://towardsdatascience.com/activation-functions-in-neural-networks-83ff7f46a6bd>

Narkhede, S. 2018. *Understanding Confusion Matrix.* [Online]. [Accessed 04 April 2019]. Available from: <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62>

Mendelson, E.B. 2019. Artificial Intelligence in Breast Imaging: Potentials and Limitations. *American journal of roentgenology*, 212(2), pp.293–299.

Mokhtar, S. & Elsayad, A. 2013. Predicting the Severity of Breast Masses with Data Mining Methods. *International Journal of Computer Science Issues (IJCSI)*, 10(2 Part 2), pp.160–168.

National Institute for Health and Care Excellence (NICE). 2017. Familial breast cancer: classification, care and managing breast cancer and related risks in people with a family history of breast cancer (update). [CG164]. London: National Institute for Health and Care Excellence.

Patterson, J. & Gibson, A. 2018. *Getting started with deep learning*. Sebastopol: O'Reilly Media, Inc.

Pedersen, R. & Schoeberl, M., 2006. An Embedded Support Vector Machine. 2006 International Workshop on Intelligent Solutions in Embedded Systems, pp.1–11.

Public Health England (PHE). 2016. NHS Breast Screening Programme. Clinical guidance for breast cancer screening assessment. [NHSBSP publication number 49]. London: Public Health England.

Qiao, M. et al. 2018. Breast Tumor Classification Based on a Computerized Breast Imaging Reporting and Data System Feature System. *Journal of Ultrasound in Medicine*, 37(2), pp.403–415.

Raschka, S. & Mirjalili, V. 2017. *Python machine learning : machine learning and deep learning with Python, scikit-learn, and TensorFlow*. 2nd ed. Birmingham: Packt Publishing Ltd.

Thompson, A. 2015. The Global Burden of Cancer 2013. *JAMA Oncology*, 314(10), p.1078.

World Health Organisation (WHO). 2018. Cancer: Key facts. [Online]. [Accessed 29 March 2019]. Available from: <https://www.who.int/news-room/fact-sheets/detail/cancer>

Zhang, Yan-Jun et al. 2013. Status quo and development trend of breast biopsy technology. *Gland surgery*, 2(1), pp.15–24.