# Evaluating and Analysing Breast Cancer Dataset for Machine Learning Algorithms

By Andy Davis

Student Number: 220491901

CSM010-01 Applied machine learning

Word Count: 1,656

## Preface

I just want to add a Preface to the report stating that I am a little uncertain if this was done correctly. What I mean is, I understand the individual parts and building up the code, but when it comes time to put it all together, I am unsure if that is correct. The bigger issue for me is the Feature Selection. It might just be the dataset that I am using, but using all the features provided is giving pretty good results. Each round of Feature Selection I have tested gives different features to use for the train/test split, thus confusing me what to do. In the end, I think I went with Pipeline and Feature Union which doesn't specifically say which features to eliminate but does it automatically for me. So, all the earlier code will be on the full dataset until the end with the union.

I also feel like explaining the different models/methods seems a bit redundant (since I am writing this for a class, I am assuming those looking at the assignment know what everything is and/or have seen explanations elsewhere and thus are burnt out by them at this point). This might make the references a little lighter than the usual report.

## Part 1: The Data

The data that is being used is the Breast Cancer Wisconsin (Original) Dataset donated in 1992 to the UCI Machine Learning Repository. Samples arrived periodically as Dr. Wolberg reported his clinical cases. The database therefore reflects this chronological grouping of the data [1]. The reason for choosing this dataset is personal for me since my mother has been battling breast cancer since I was 2 years old. That had also pushed my dad (a chemical engineer with no prior knowledge of cancer) to research new, less painful drugs to help future patients not go through the terrible treatments my mother went through (and is still going through 20+ years later).

The 1992 Dataset has 11 features: Sample Number (ID), Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli, Mitosis, and Class (target data). All the columns except ID and Class are on a scale of 1-10, and Class is labeled 2 for benign and 4 for malignant, thus rescaling the data does not seem necessary. Unfortunately, there is some missing data within the Bare Nuclei column that was removed [2]. This dropped the dataset total from 699 (Appendix Figure 1) instances to 683 (Appendix Figure 2) which is not a huge loss. Once the data was dropped, it had to be converted back to Int64 values [3]. Given there are twice as many benign entries as malignant (Appendix Figure 3), there may be a bias to false negatives given from this training.

## Part 2: Constructing and Selecting Features

To start, ID is not a needed feature since it has no correlation to the target variable, it is merely the identification of the trial. Thus, we can manually remove it from our dataset before we start the training. That leaves us with the 9 other features to select (not including the target variable Class). Already that is a small number to choose from, but we can test different wrapper and filter methods to see if they are all needed or not.

The two methods that I had decided to go with were the Recursive Feature Elimination (RFE) wrapper method and the Chi-Squared Feature Extraction filter method. RFE works by recursively removing attributes and building a model on those attributes that remain. It uses the model accuracy to identify which attributes (and combination of attributes) contribute the most to predicting the target attribute. Whereas the Chi-Squared method selects features according to the k highest scores using the chi-squared statistical test for non-negative features to select 4 of the best features [4]. Each method produced a different result for the features to keep or exclude. RFE went with the cell-shape, epithelial-size, nuclei, and mitosis features (Appendix Figure 4), whereas the chi-squared method went with the nuclei, cell-size, cell-shape, and nucleoli features (Appendix Figure 5) after cross checking the scores manually to the feature names. There really was not much of a benefit to the accuracies, thus all features are still going to be considered.

## Part 3: Building ML Algorithms

There wasn't really a clear motivation as to which candidate algorithm was going to be included, I just looked at a few papers and saw which ones were chosen for categorical based datasets. On top of that, the UCI Machine Learning Repository also lists algorithms used with their respective accuracy and precision ranges. I ended up trying out 5 different models: Logistic Regression (LR), Random Forest (RF), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Naive Bayes (NB). With each one, I split the dataset into train/test sets (with a test size of 0.33 and random seed of 7) as well as two different split methods (KFold and Shuffle Split, with N Splits of 10). The first method was just fit with the training set and then scored with the test set to get accuracies, but last two methods were there evaluated with cross validation to get both accuracies and standard deviation.

The Shuffle Split method gave the lowest accuracies, but also gave the lowest variances, whereas the non-method way of splitting the data gave the highest accuracies. Granted these differences were only 2-3% (95.929% being the lowest and 97.788% being the highest), so all in all each algorithm performed very well no matter the technique that was applied. Primarily, all of these were chosen through trial and

error (or testing everything and getting all comparisons without going through the hassle of trying to narrow things down, which may seem like more work). (Appendix Figure 6)

## Part 4: Evaluating Models and Analysing the Results

To evaluate the models, I fit each of them with the training data (that was not through KFold nor Shuffle Split) and then had a predicted variable from the X_test set. This was then run through both the classification report and the confusion matrix (with Y_test and predicted as the arguments). Again, each model did very well, having the greatest margin of error of 0.06 between predicting benign vs malignant results. All precisions, recalls, and f1-scores were 0.94 and above with an average of 0.97. Naive-Bayes had the greatest variance (0.06) across the board, whereas Random Forest was the tightest. But since all had an average of 0.97, the difference between each model is very minimal. (Appendix Figure 7)

The confusion matrix for each model performed just as positively. At the beginning, I thought that there would be a bias towards false negatives given that benign outweighed malignant, but the highest false negative result was 8 (max for false positive was 2). (Appendix Figure 7)

Since I had mostly run everything on the entire dataset (or with all the features opposed to selecting a few), I ran one final test using the Pipeline and Feature Union libraries having them choose 4 features to use. The accuracies and stand deviations were not far off from everything I had done prior, confirming my confidence in keeping all features. (Appendix Figure 8)

## Conclusion

I did not really look too deeply into which dataset I wanted to choose from the UCI site, I just wanted to try something that had a relative importance to myself. Thus, when I saw the option for the breast cancer dataset, I went for it. I was pretty lucky in the sense that this dataset had lots of work already done to it regarding pre-processing. There were very minor instances missing, all the data was scaled 1-10, and the features were small enough and relevant enough that everything could be included. I know that this may seem like I chose an easy route out, but as mentioned it was not intended. It did help since I am taking another class and a vacation (July 4$^{th}$ holiday for the United States, planned family trip a year ago before applying to the university and knowing the module schedules) inconveniently happened during these two assessment weeks, leaving me not very much time to work on this assignment.

This was a good introduction for me to start using my family's own medical data (and maybe other features) to help predict if either myself or my siblings will be susceptible to having cancer.

## Post Conclusion Thoughts

To be honest, I was not really expecting this for the Machine Learning class. It ended up more about Data Science, which is a reason I did not go to a university within the United States. I was hoping this class would feature more of the things that were mentioned in Topic 10 (computer vision) opposed to just a short topic. Everything combined soured my motivation towards the end of this course, and I am sad to say that it shows in my report. Just like in the Computer Systems class, once I hit a wall in my understanding, I did the best I could do, but at the same time I lost the motivation to go deeper. This term just ended poorly, which I am not happy with myself about. I am hoping this does not reflect poorly on my potential for these classes, but I think I am just a little burnt out for doing two classes at a time for the last few terms so it will be nice to only have one class to focus on for the next two terms (assuming grades go well from the previous terms). The other classes are also those that I need/use at work and thus will get my full attention, just like the first term did.

Sorry this did not end the way you might have expected, but I wanted to let you all know what I have been thinking and why this report may be lacking.

# Appendix

Figure 1 – Pandas data.info() on initial dataset

RangeIndex: 699 entries, 0 to 698

Data columns (total 11 columns):

| # | Column | Non-Null Count | Dtype |
|---|---|---|---|
| 0 | id | 699 non-null | int64 |
| 1 | thickness | 699 non-null | int64 |
| 2 | cell-size | 699 non-null | int64 |
| 3 | cell-shape | 699 non-null | int64 |
| 4 | adhesion | 699 non-null | int64 |
| 5 | epithelial-size | 699 non-null | int64 |
| 6 | nuclei | 699 non-null | object |
| 7 | chromatin | 699 non-null | int64 |
| 8 | nucleoli | 699 non-null | int64 |
| 9 | mitosis | 699 non-null | int64 |
| 10 | class | 699 non-null | int64 |

dtypes: int64(10), object(1)

Figure 2 - Pandas data.info() after removing null objects and converting back to int64

Index: 683 entries, 0 to 698

Data columns (total 11 columns):

| # | Column | Non-Null Count | Dtype |
|---|---|---|---|
| 0 | id | 683 non-null | int64 |
| 1 | thickness | 683 non-null | int64 |

| | | | |
|---|---|---|---|
| 2 | cell-size | 683 non-null | int64 |
| 3 | cell-shape | 683 non-null | int64 |
| 4 | adhesion | 683 non-null | int64 |
| 5 | epithelial-size | 683 non-null | int64 |
| 6 | nuclei | 683 non-null | int64 |
| 7 | chromatin | 683 non-null | int64 |
| 8 | nucleoli | 683 non-null | int64 |
| 9 | mitosis | 683 non-null | int64 |
| 10 | class | 683 non-null | int64 |

dtypes: int64(11)

Figure 3 – Class Counts

class

2 444

4 239

dtype: int64

Figure 4 – RFE feature analysis

Num Features: 4

The feature labels: ['thickness', 'cell-size', 'cell-shape', 'adhesion', 'epithelial-size', 'nuclei', 'chromatin', 'nucleoli', 'mitosis']

Selected Features: [False False True False False True True False True]

Feature Ranking: [2 5 1 4 6 1 1 3 1]

Figure 5 – Chi-Squared feature analysis

The feature labels: ['thickness', 'cell-size', 'cell-shape', 'adhesion', 'epithelial-size', 'nuclei', 'chromatin', 'nucleoli', 'mitosis']

Fit scores: [ 624.13570418 1370.06458731 1279.76770412  986.41787922  497.53676321

1729.0661744   682.97823856 1143.8667119   228.99434634]

Figure 6 – Test accuracies and deviations for splitting methods

Train/Test Accuracies:

LR: 96.903

RF: 97.788

KNN: 97.345

SVM: 96.903

NB: 97.788

KFold Accuracies:

LR: 96.488 (2.197)

RF: 96.922 (1.538)

KNN: 97.364 (1.708)

SVM: 96.777 (1.950)

NB: 96.334 (2.809)

ShuffleSplit Accuracies:

LR: 96.637 (0.889)

RF: 96.858 (1.161)

KNN: 97.212 (1.138)

SVM: 96.858 (0.873)

NB: 95.929 (1.478)

Figure 7 – Classification Reports and Confusion Matrices

LR Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 2 | 0.99 | 0.96 | 0.98 | 142 |
| 4 | 0.94 | 0.98 | 0.96 | 84 |
| accuracy | | | 0.97 | 226 |
| macro avg | 0.96 | 0.97 | 0.97 | 226 |
| weighted avg | 0.97 | 0.97 | 0.97 | 226 |

LR Confusion Matrix:

[[137 5]

[ 2 82]]

RF Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 2 | 0.99 | 0.96 | 0.98 | 142 |
| 4 | 0.94 | 0.99 | 0.97 | 84 |
| accuracy | | | 0.97 | 226 |
| macro avg | 0.97 | 0.98 | 0.97 | 226 |
| weighted avg | 0.97 | 0.97 | 0.97 | 226 |

RF Confusion Matrix:

[[137 5]

[ 1 83]]

KNN Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 2 | 0.99 | 0.96 | 0.98 | 142 |
| 4 | 0.94 | 0.99 | 0.97 | 84 |
| accuracy | | | 0.97 | 226 |
| macro avg | 0.97 | 0.98 | 0.97 | 226 |
| weighted avg | 0.97 | 0.97 | 0.97 | 226 |

KNN Confusion Matrix:

[[137 5]

[ 1 83]]

SVM Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 2 | 0.99 | 0.96 | 0.98 | 142 |
| 4 | 0.94 | 0.98 | 0.96 | 84 |
| accuracy | | | 0.97 | 226 |
| macro avg | 0.96 | 0.97 | 0.97 | 226 |
| weighted avg | 0.97 | 0.97 | 0.97 | 226 |

SVM Confusion Matrix:

[[137 5]

[ 2 82]]

NB Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 2 | 1.00 | 0.96 | 0.98 | 142 |
| 4 | 0.94 | 1.00 | 0.97 | 84 |
| accuracy | | | 0.98 | 226 |
| macro avg | 0.97 | 0.98 | 0.98 | 226 |
| weighted avg | 0.98 | 0.98 | 0.98 | 226 |

NB Confusion Matrix:

[[137 5]

[ 0 84]]

Figure 8 – Pipeline and Estimators accuracies

LR: 96.488 (2.197)

RF: 97.217 (1.389)

KNN: 97.364 (1.708)

SVM: 96.777 (1.950)

NB: 96.334 (2.809)

# References

[1] Wolberg,WIlliam. (1992). Breast Cancer Wisconsin (Original). UCI Machine Learning Repository. https://doi.org/10.24432/C5HP4Z.

[2] Gurav, S. (2022) 3 Ultimate Ways to Deal With Missing Values in Python, Towards Data Science. Available at: https://towardsdatascience.com/3-ultimate-ways-to-deal-with-missing-values-in-python-ac5a17c53787#:~:text=You%20can%20use%20pandas%20DataFrame,contain%20atleast%20one%20missing%20value. (Accessed: 28 June 2023).

[3] Admin. (2023) *Pandas Convert Column to Int in DataFrame*, *SparkBy{Examples}*. Available at: https://sparkbyexamples.com/pandas/pandas-convert-column-to-int/#:~:text=Convert%20Column%20to%20int%20(Integer,int64%20%2C%20numpy (Accessed: 06 July 2023).

[4] Harris, M. (2023) *Hands-on programming demo: feature selection, University of London Learning Portal.* Available at: https://learn.london.ac.uk/mod/lti/view.php?id=120056&forceview=1 (Accessed: 07 July 2023).