Noah Saffie

COMP.5300

Section 204

Final Project Report

INTRODUCTION

The topic that I chose to solve was the classification of breast cancer into either benign or malignant. Cancer is one of the most serious illnesses in the world today, and breast cancer is one of the most common forms of cancer. While breast cancer is most prevalent in women, it can also occur in men, and should be watched for by people of all ages, genders, race, etc. For this project, my aim was just to look at the dataset and try out some basic machine learning models to see what kind of accuracy we could get. I chose to apply the following algorithms to the dataset, Naïve Bayes’ (Gaussian), Support Vector Machine, Multi-Layer Perceptron (Neural Network), and Decision Tree.

DATASET

The dataset I used was “Breast Cancer Wisconsin (Original) Data Set” by Dr. William H. Wolberg, published on the UCI Machine Learning Repository website. The dataset has nearly 700 instances, with 10 different attributes as follows: Sample code number, Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli, Mitoses, Class. The dataset consists entirely of integers, including eight of the attributes being in the discrete range [1-10]. This can be problematic because it can create a high degree of correlation since the range of potential values is much smaller, and this is the case for almost all of the attributes.

RELATED WORKS

One related work is “Multisurface method of pattern separation for medical diagnosis applied to breast cytology” by Wolberg, W.H., & Mangasarian, O.L. This paper used the method of multisurface pattern recognition on the dataset (an older version of the dataset to be exact). The multisurface pattern recognition worked by first plotting all the 9 (non-target) variables in a 9-dimensional space. A typical separation of the points would be by solving a single linear equation that creates a plane/hyperplane separating the points based on the target class. However, this can be ineffective since there are limitations on how janky the plane could be so that all points can be accurately classified. This is where the multisurface part of the method comes in, instead of just a single plane/hyperplane the decision boundary is constructed in a piece-wise manner, that is made up of many planes put together.

Given the dataset, I am not surprised that the authors of the paper found this method to be effective however, for most general machine learning problems this would not be effective enough, as you can have a lot more overlap between the classes, as well as becoming much harder to find the correct decision boundary when the values are real-valued instead of just finite integers. Since I am using several different algorithms, it is expected that the methods vary from that of this paper. The Naïve Bayes’ gaussian works in a somewhat similar way by trying to determine classification from a probabilistic perspective based on class distributions, which are assumed to be gaussian. However, the neural network (multi-layer perceptron) can depending on how you customize the number of layers, and activation functions vary wildly. Decision trees which work by having internal nodes that perform some kind of test, and each node depending on if answer to test is yes or no, splitting to a new node. A node can either be another test, or a terminal node that species a particular class. Finally, support vector machines transform the points in the plane into a new way such that there is an easy linear boundary to be calculated, this is done using a kernel machine.

METHODS

As discussed earlier in this paper, I used four different algorithms to compare the accuracies of each. For each of the algorithms I used the off-the-shelf version provided by the scikit-learn library. The first algorithm was a basic Naïve Bayes’ gaussian, followed by a generic Support Vector Machine, then a Decision Tree, and finally a Multi-Layer Perceptron Neural Network. Other than the simple application of these methods to the dataset, I used applied several visualization techniques some of which are shown in this paper, and all of which can be shown in the Juypter Notebook that corresponds with this paper. All of the visualizations are drawn from “The Art of Effective Visualization of Multi-dimensional Data” by Dipanjan (DJ) Sarkar, with some minor modifications to match my dataset, and desires.s

RESULTS

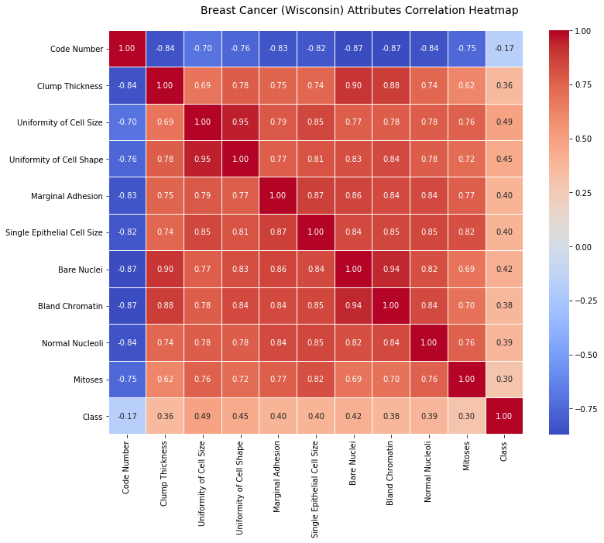
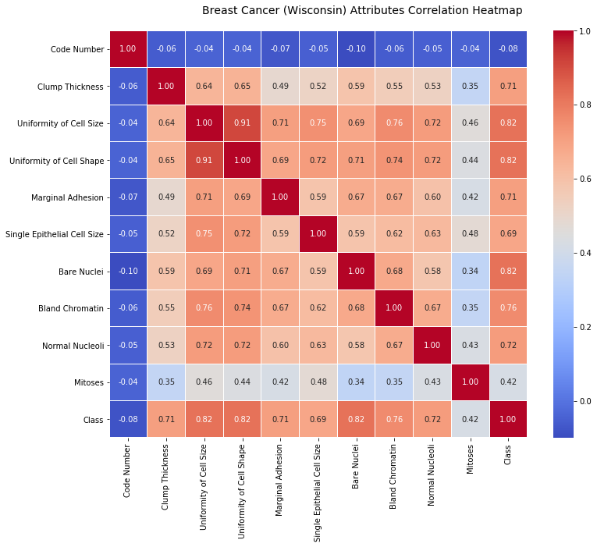
The results I got can be seen in the table below. As can be seen below decision tree performed the best in every single case, although Naïve Bayes’ performed fairly well overall as well. I originally expected that because the attributes Bland Chromatin and Bare Nuclei had such a high correlation with each other (94%) that removing Bare Nuclei might allow the training to perform better since the model can be less complex with less attributes to use. However, as can be seen for Naïve Bayes’ for training #1 and training #2, we see a drop in performance after removing the Bare Nuclei attribute.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Naïve Bayes’ Gaussian | Multi-Layer Perceptron | Support Vector Machine | Decision Tree |
| Training #1 – Basic Training | 89.05% | 64.96% | 64.96% | 93.43% |
| Training #2 – Basic Training without the Bare Nuclei attr. | 87.59% | 64.96% | 64.96% | 92.70% |
| Training #3 - Upsampling | 81.46% | 50% | 50% | 95.50% |
| Training #4 - Downsampling | 81.46% | 50% | 50% | 94.94% |

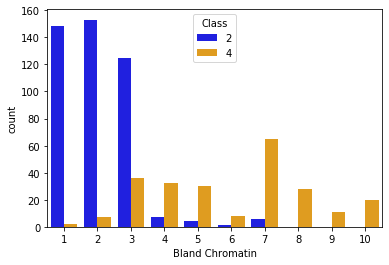
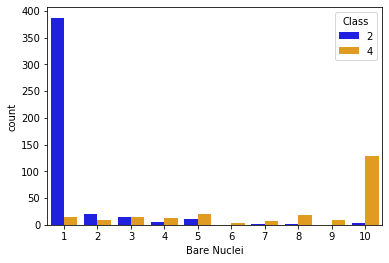
DISCUSSION

An interesting result that can be seen in the table is that both Multi-Layer Perceptron and Support Vector Machine have the exact same performance across all trainings. This is bizarre because the methods are quite different. There is a much more interesting conclusion that be drawn from the results of those two methods by seeing the two different results attained 64.96% and 50%. The 50% result is more obvious especially given it occur when upsampling or downsampling, that is when the classes have a perfectly balanced amount, so the 50% indicates that for both methods the algorithms are just consistently predicting a single class, that is to say that haven’t learned anything at all. The other result 64.96% is exactly the same, if we look at the distribution of the target class, we see 444 instances of benign, and 239 instances of malignant, if we were to predict benign every time, then we would obtain an accuracy of 64.96%, so once again it can be seen that those two models didn’t learn anything at all, but just predicted that everything was benign.

Since the goal of this project wasn’t just to test the dataset on various algorithms, it was to look into the data and see if anything interesting could be seen, I have some results regarding that as well. One interesting finding through the many visualizations of the data, is that the normalized dataset has a very high level of correlation across almost all of the variables, which in my past experience is quite uncommon. It can also be seen that there is a significant jump in correlation values when the data is normalized versus when it is used in its regular form, which can be seen in the heatmaps shown below. This can likely be attributed to that since all values are forced into the range [0-1] the values will have a smaller gap between values, so there is less “outliers” in the data, although proportions are still maintained so it is a bit odd to see such a serious jump.



Another interesting piece that can be seen from the data is that benign tends to heavily favor lower values for all the attributes (other than code number). This is demonstrated in the visualizations shown below. This is likely one of the big reasons that Decision Tree performed to favorably, because it could use simply tests such as if the value was with the first 5 values to favor it being benign. While the target class for malignant follows a somewhat similar pattern that it favors the higher values of each attribute, it doesn’t have quite as man extreme counts, and instead tends to be more evenly distributed, both in general but in particular among the higher 50% of the values. Keeping in mind that when creating these graphs I used the regular data, so there is an inherit class imbalance. However, this shouldn’t have much impact on the understanding on the graphs since we aren’t comparing the frequency of the classes to each other, but instead are looking at the general distributions of each class for each attribute. I believe that this could also lead to neural networks being highly effective since they rely on weights and activation functions, a pattern like that shown below could be exploited easily inside a neural network for good results.



FUTURE WORK

There are plenty of ways to improve upon the methods used in this paper. One of the most basic ways is to utilize the various parameters that each of the algorithms offer. Since using an algorithm right off the shelf with no tuning is prone to be inefficient, it would make sense to tune each of the algorithms, and the typical way to do so is utilizing parameters. For instance, picking a particular kernel machine for an SVM can have varying results, so a talented and experienced professional would try out various kernel machines that they believe to be correct based on looking at the data.

Another piece that could be looked into in the future is why both Multi-Layer Perceptron and Support Vector Machine didn’t seem to learn from the data at all but were instead just always predicting the first class. This is rather bizarre, and makes me wonder if it is an issue with my use of the algorithms right off the shelf, but for both algorithms I took directly from examples offered by scikit-learn so there is no reason that they should be breaking, especially without any kind of error. I would also like to take advantage of using K-Fold to ensure the accuracy better, since K-Fold will split the data K times, creating different test and training sets from it, then you can average the accuracies together to gain a better accuracy overall. I attempted to implement this into my program, in the same way that I have used it before but kept running into errors that I was unable to find a solution for, so it was abandoned.

CONCLUSION

Ultimately, I feel that I was successful, after all I was able to find an algorithm, Decision Tree, that obtained a very high accuracy of 95%, which even for medical uses is significant, and that is before any kind of tuning was performed. Similarly, I felt that finding that the distribution of benign correlated highly with lower half of values of almost all attributes, and the opposite was observed, although in a lesser way, for malignant distribution and the upper half of values for almost all attributes. I also feel that observing the correlation matrix’s large change in values after normalization is also significant. In conclusion I feel that I found some important information about the attributes of the data, and was also able to find a very good algorithm for this dataset.

WORKS CITED

Sarkar, Dipanjan. “The Art of Effective Visualization of Multi-Dimensional Data.” *Towards Data Science*, Medium, 11 Dec. 2018, towardsdatascience.com/the-art-of-effective-visualization-of-multi-dimensional-data-6c7202990c57.

“How to Handle Imbalanced Classes in Machine Learning.” *EliteDataScience*, EliteDataScience.com, 25 Jan. 2019, elitedatascience.com/imbalanced-classes.

Wolberg, W. H., and O. L. Mangasarian. “Multisurface Method of Pattern Separation for Medical Diagnosis Applied to Breast Cytology.” *Proceedings of the National Academy of Sciences*, vol. 87, no. 23, 1990, pp. 9193–9196., doi:10.1073/pnas.87.23.9193.

Wolberg, WIlliam H, and Olvi Mangasarian. “Breast Cancer Wisconsin (Original) Data Set.” *UCI Machine Learning Repository*, 15 July 1992, archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29.