Breast Cancer Detection By Fine Needle Aspiration

Machine Learning Algorithms Using Ensembled Models to Detect Malignancy

L. Schoch, MD

3/1/2019

Introduction

Breast cancer is the most common cancer diagnosed among U.S. women (excluding skin cancers) - about 252,710 new cases were expected to be diagnosed in 2017. It is the second leading cause of cancer death among women in the U.S., after lung cancer, and 40,610 women in the U.S. were expected to die from this disease in 2017.[3] As with cancer in general, early diagnosis and intervention are key to long term survival. Currently, early diagnosis is best achieved by screening mammography and fine needle aspiration biopsy of detected lesions. This paper presents results of efforts to develop a machine learning algorithm that accurately discriminates between malignant and benign lesions using the Wisconsin Diagnostic Breast Cancer (WDBC) data set. This data set has been widely used for this purpose and accuracies approaching 100% have been achieved in the past.[1][5]

Data Exploration and Analysis

The Wisconsin Diagnostic Breast Cancer data set contains the data from a series of fine needle aspiration breast biopsies performed by Dr. William Wolberg, physician at the University of Wisconsin Hospital in Madison, Wisconsin, USA. It was donated in 1992 and is available to the public on the University of California Irvine Machine Learning Data Repository.[4]. To create the dataset Dr. Wolberg used fluid samples, taken from patients with solid breast masses, and an easy-to-use graphical computer program called Xcyt, which evaluates characteristics of cell nuclei present on a digital image. The dataset consists of 569 biopsies and there are 32 variables for each biopsy. Variables include a unique patient identifier and the target variable, diagnosis, a binary classifier with values of B (benign) or M (malignant). An additional 30 variables are obtained as follows bringing the total 32:

Ten real-valued variables are computed for each cell nucleus using the Xcyt software:
1. radius (mean of distances from center to points on the perimeter)
2. texture (standard deviation of gray-scale values)
3. perimeter
4. area
5. smoothness (local variation in radius lengths)
6. compactness (perimeter^2/area - 1.0)
7. concavity (severity of concave portions of the contour)
8. concave points (number of concave portions of the contour)
9. symmetry
10. fractal dimension ("coastline approximation" - 1)

The mean, standard error, and "worst" or largest (mean of the three largest values) of these variables are computed for each image, resulting in 30 variables. For instance variable 3 is mean radius, variable 13 is radius standard error (se) and variable 23 is worst radius.

As a first step in data exploration it is helpful to get an overview with the `glimpse` function:

```
Observations: 569
Variables: 32
$ ID                      <int> 842302, 842517, 84300903, 84348301, 84...
$ diagnosis               <fct> B, B, B, B, B, B, B, B, B, B, B, B, B,...
$ radius                  <dbl> 17.990, 20.570, 19.690, 11.420, 20.290...
$ texture                 <dbl> 10.38, 17.77, 21.25, 20.38, 14.34, 15....
$ perimeter               <dbl> 122.80, 132.90, 130.00, 77.58, 135.10,...
$ area                    <dbl> 1001.0, 1326.0, 1203.0, 386.1, 1297.0,...
$ smoothness              <dbl> 0.11840, 0.08474, 0.10960, 0.14250, 0....
$ compactness             <dbl> 0.27760, 0.07864, 0.15990, 0.28390, 0....
$ concavity               <dbl> 0.30010, 0.08690, 0.19740, 0.24140, 0....
$ concave_points          <dbl> 0.14710, 0.07017, 0.12790, 0.10520, 0....
$ symmetry                <dbl> 0.2419, 0.1812, 0.2069, 0.2597, 0.1809...
$ fractal_dimension       <dbl> 0.07871, 0.05667, 0.05999, 0.09744, 0....
$ radius_se               <dbl> 1.0950, 0.5435, 0.7456, 0.4956, 0.7572...
$ texture_se              <dbl> 0.9053, 0.7339, 0.7869, 1.1560, 0.7813...
$ perimeter_se            <dbl> 8.589, 3.398, 4.585, 3.445, 5.438, 2.2...
$ area_se                 <dbl> 153.40, 74.08, 94.03, 27.23, 94.44, 27...
$ smoothness_se           <dbl> 0.006399, 0.005225, 0.006150, 0.009110...
$ compactness_se          <dbl> 0.049040, 0.013080, 0.040060, 0.074580...
$ concavity_se            <dbl> 0.05373, 0.01860, 0.03832, 0.05661, 0....
$ concave_points_se       <dbl> 0.015870, 0.013400, 0.020580, 0.018670...
$ symmetry_se             <dbl> 0.03003, 0.01389, 0.02250, 0.05963, 0....
$ fractal_dimension_se    <dbl> 0.006193, 0.003532, 0.004571, 0.009208...
$ radius_worst            <dbl> 25.38, 24.99, 23.57, 14.91, 22.54, 15....
$ texture_worst           <dbl> 17.33, 23.41, 25.53, 26.50, 16.67, 23....
$ perimeter_worst         <dbl> 184.60, 158.80, 152.50, 98.87, 152.20,...
$ area_worst              <dbl> 2019.0, 1956.0, 1709.0, 567.7, 1575.0,...
$ smoothness_worst        <dbl> 0.1622, 0.1238, 0.1444, 0.2098, 0.1374...
$ compactness_worst       <dbl> 0.6656, 0.1866, 0.4245, 0.8663, 0.2050...
$ concavity_worst         <dbl> 0.71190, 0.24160, 0.45040, 0.68690, 0....
$ concave_points_worst    <dbl> 0.26540, 0.18600, 0.24300, 0.25750, 0....
$ symmetry_worst          <dbl> 0.4601, 0.2750, 0.3613, 0.6638, 0.2364...
$ fractal_dimension_worst <dbl> 0.11890, 0.08902, 0.08758, 0.17300, 0....
```

We can use the `anyNA` function to show that there are no missing values in our data set (wdbc_data):

```r
anyNA(wdbc_data)
```

```
[1] FALSE
```

Since this is a binary classification problem we need to evaluate the degree, if any, of imbalance in our target variable, diagnosis. Of the 569 biopsies in our dataset, 357 (62.7%) are malignant - so there *is* class imbalance (also shown graphically in Figure 1); however, since our goal is to detect malignancy and malignancy is the majority class, we do not need to concern ourselves with techniques for dealing with class imbalance such as cost-sensitive learning or sampling (oversampling the minority class or undersampling the majority class). (From https://towardsdatascience.com/dealing-with-imbalanced-classes-in-machine-learning-d43d6fa19d2)

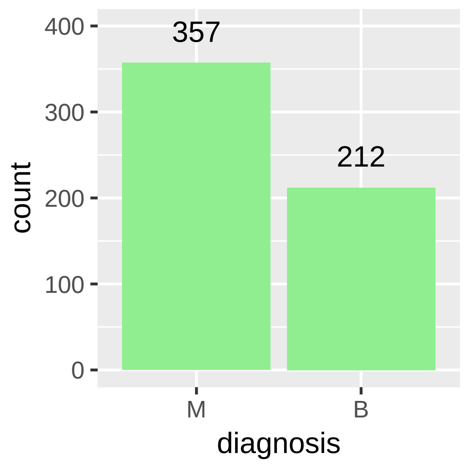

Figure 1: Diagnosis - class imbalance.

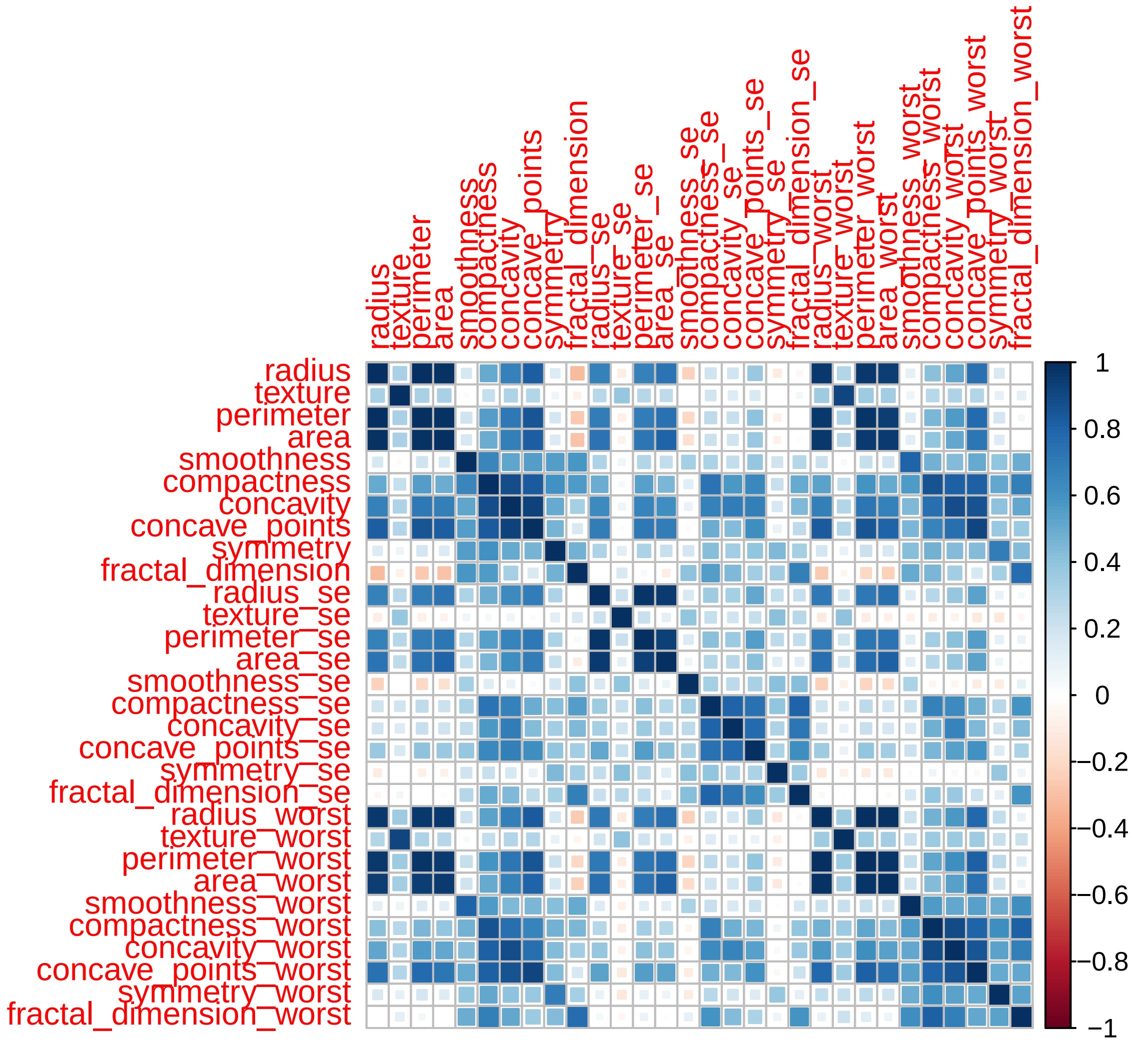

Figure 2: Correlation plot of predictor variables.

The degree of correlation between predictor variables is an important consideration in predictive modeling. Figure 2 is a correlation plot for the 30 predictor variables in our WDBC dataset and it does show some areas of potential concern. Simple geometry leads us to expect high correlation between radius, area and perimeter and this is confirmed in Figure 2. There is also high correltation between radius, area and perimeter and their corresponding "worst" categories, again, not unexpected. Finally, there is significant correlation between compactness, concavity and concave points and their corresponding "worst" categories.

Let's investigate our predictor variables further with principal component analysis (pca). Briefly, principal component analysis creates a set of linearly uncorrelated numeric variables - principal components - from a set of possibly correlated numeric variables using an orthogonal transformation. This transformation is defined in such a way that the first principal component accounts for as much of the variability in the data as possible, and each succeeding component, in turn, has the next highest variance possible under the constraint that it is orthogonal to the preceding components. If there are n observations with p variables, the number of distinct principal components is the lesser of *n-1* or *p*. (From https://en.wikipedia.org/wiki/Principal_component_analysis) Thus the number of principal components for our data is the number of numeric predictor variables, 30.

Figure 3 is a plot of the cumulative proportion of variance explained by the successive principal components of our data. As expected, the first component accounts for the largest variance (44.3%). 95.2% of the variance is explained by the first 10 components. Thus, principal component analysis of the WDBC data reveals that feature selection may be indicated when we develop our model. Fortunately, the caretEnsemble functions have pca built-in as a pre-processing option.

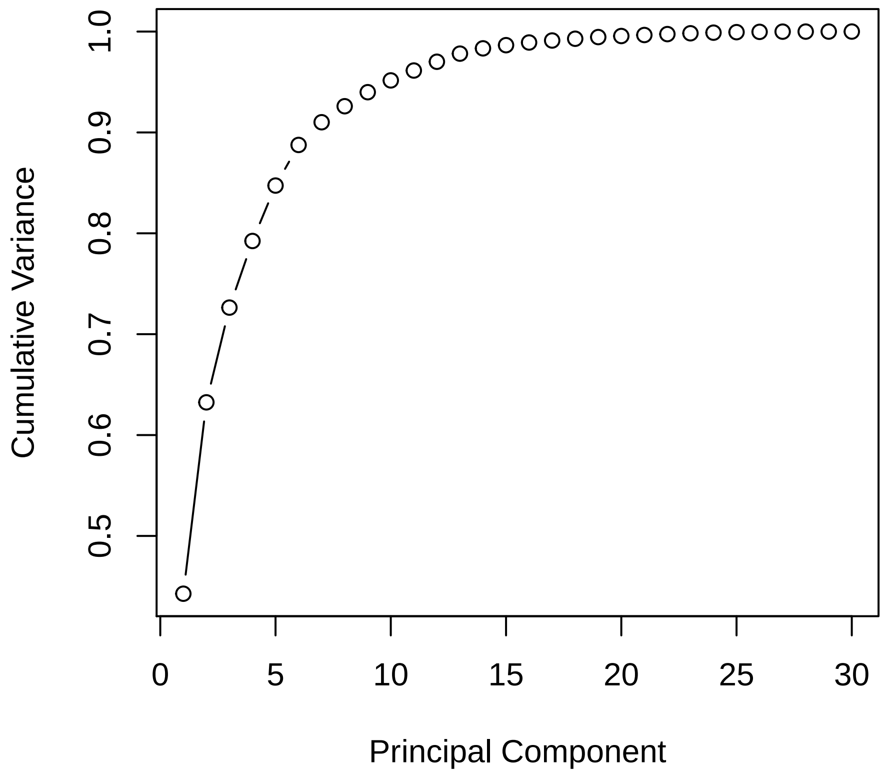

Figure 3: Cumulative proportion of variance.

Methods

The data were partitioned, 80% as a training set and 20% as a test set. Then, functions from the caretEnsemble package were used to create ensembles of models. The `caretList` function was used to train a list of six models: `lda`, `rpart`, `knn`, `svmRadial`, `Rborist`, and `nnet`. The default training parameters were used and normalization of the 30 predictor variables was done as part of the pre-processing. The effect of principal component analysis was tested using the `pca` pre-processing option.

The `caretEnsemble` function uses a `glm` to create a simple linear blend of models. Two ensembles were created with this function: one using the models that were trained without principal component analysis and the second using the models that were trained with principal component analysis.

Training of the models and ensembles was done with repeated cross validation, 30 folds repeated three times. The number 30 was chosen to produce enough resamples for meaningful p-value calculations.

Results

Table 1 shows the accuracies obtained by each of the six models in each of the 30 resamples. Values range from a low of 0.9006 to a high of 0.9884. On visual inspection, `svmRadial` appears to be the top performer.

lda	rpart	knn	svmRadial	Rborist	nnet
0.9573	0.9085	0.9634	0.9878	0.9695	0.9756
0.9578	0.9458	0.9759	0.9699	0.9398	0.9699
0.9636	0.9333	0.9394	0.9636	0.9455	0.9394
0.9702	0.9583	0.9405	0.9702	0.9643	0.9762
0.9441	0.9193	0.9379	0.9627	0.9130	0.9689
0.9489	0.9261	0.9773	0.9716	0.9545	0.9602
0.9415	0.9006	0.9766	0.9766	0.9591	0.9708
0.9756	0.9329	0.9573	0.9756	0.9695	0.9695
0.9593	0.9244	0.9419	0.9651	0.9128	0.9884
0.9527	0.9704	0.9467	0.9882	0.9704	0.9822
0.9506	0.9630	0.9568	0.9753	0.9753	0.9753
0.9217	0.9217	0.9398	0.9578	0.9458	0.9699
0.9684	0.9557	0.9620	0.9747	0.9557	0.9873
0.9524	0.9107	0.9524	0.9345	0.9524	0.9643
0.9538	0.9364	0.9711	0.9711	0.9538	0.9595
0.9408	0.9053	0.9527	0.9645	0.9231	0.9645
0.9709	0.9186	0.9651	0.9826	0.9651	0.9709
0.9701	0.9521	0.9521	0.9820	0.9521	0.9820
0.9500	0.9222	0.9667	0.9778	0.9556	0.9833
0.9412	0.9294	0.9471	0.9471	0.9471	0.9765
0.9524	0.9345	0.9702	0.9583	0.9464	0.9405
0.9264	0.9387	0.9632	0.9755	0.9387	0.9755
0.9755	0.9264	0.9693	0.9693	0.9571	0.9816
0.9371	0.9371	0.9657	0.9657	0.9314	0.9657
0.9268	0.9024	0.9146	0.9329	0.9085	0.9634
0.9253	0.9310	0.9483	0.9713	0.9598	0.9540
0.9695	0.9390	0.9573	0.9756	0.9756	0.9756
0.9578	0.9518	0.9578	0.9578	0.9337	0.9759
0.9538	0.9306	0.9595	0.9595	0.9595	0.9711
0.9515	0.9091	0.9818	0.9697	0.9697	0.9576

Table 1: Model accuracy for each resample.

The models were also trained using principal component analysis for the potential advantages of decreased complexity, decreased processing time and decreased risk of overfitting. Table 2 compares the accuracy averaged over the 30 resamples for each model, with and without pca. Paired t-testing was used to calculate the p-values. Based on the p-values, pca resulted in a significant decrease in accuracy for the `svmRadial` and `Rborist` models. Effect size (calculated manually) for `svmRadial` was large at 1.245 and effect size for `Rborist` was small at 0.472. Accuracies obtained by the other models were not significantly affected by principal component analysis.

A potential flaw in the p-value calculations is the inherent assumption with t-tests that the data are normally distributed. Normality of the data was not evaluated in this analysis beyond visual inspection of `qqplots` (not shown here). On the other hand, the Central Limit Theorem tells us that the sampling distribution will approach normality with a large enough sample size, and it is generally accepted that 30 or more samples is adequate to meet this criterion.

model	accuracy-no pca	accuracy-pca	p-value
lda	0.9522	0.9561	0.154
rpart	0.9312	0.9234	0.118
knn	0.9570	0.9596	0.150
svmRadial	0.9678	0.9506	0.000
Rborist	0.9502	0.9424	0.042
nnet	0.9698	0.9730	0.121

Table 2: Average model accuracy with and without pca.

Ensembles

Table 3 compares the accuracies obtained by ensembling. Two ensembles were created, one for the models trained without principal component analysis and one for the models trained with principal component analysis. Results obtained for both the training data and the test data are presented.

	no pca	pca
accuracy-train	0.9760	0.9767
accuracy-test	0.9826	0.9652

Table 3: Ensemble accuracy with and without pca.

Principal component analysis did not appear to affect the results obtained using the training data but there was a decrease in performance with the test data. Formal testing of the significance of this decrease is beyond the scope of this work; however, there is strong overlap of the 95% confidence intervals: 0.9386 to 0.9979 for the models trained without pca and 0.9133 to 0.9904 for the models trained with pca. This suggests that the difference is probably not signficant.

When attempting to detect cancer, emphasis is placed on minimizing the number of false negatives, even at the expense of increased false positives (better to call a benign lesion malignant than vice versa). Predicting from the test data, sensitivity for both ensembles (i.e., using the models trained with and without pca) was 100%, indicating that all malignant lesions were detected. The no-pca ensemble produced 2 false positives while the pca ensemble produced 4 false positives, reflective of the lower accuracy seen with the pca ensemble.

Was ensembling beneficial? Table 4 is a modification of Table 2 showing the average accuracy obtained for each model along with the accuracy obtained for the corresponding ensemble. For each case, either with or without pca, the ensemble does appear to perform better than any of the component models alone.

	accuracy-no pca	accuracy-pca
lda	0.9522	0.9561
rpart	0.9312	0.9234
knn	0.9570	0.9596
svmRadial	0.9678	0.9506
Rborist	0.9502	0.9424
nnet	0.9698	0.9730
ensemble	0.9760	0.9767

Table 4: Average accuracy for each model and the ensemble.

Conclusions

The caretEnsemble package is a powerful tool for simultaneously training a group of models on the same data set and it also provides functions for creating ensembles of the models. For this project, the `caretList` function was used to train a list of six models: `lda`, `rpart`, `knn`, `svmRadial`, `Rborist`, and `nnet` using the Wisconsin Diagnostic Breast Cancer data set. Two ensembles were created, one for the models trained without pca and one for the models trained with pca. In both cases, the ensemble performed better than any of the six models individually.

For the ensemble created from models that were trained without pca, accuracy of 0.976 was obtained using the training data and accuracy of 0.9826 was obtained using the test data. These results approach or exceed those reported by previous authors.[2][1][5]

Principal component analysis resulted in a significant decrease in peformance of two of the models, `svmRadial` and `Rborist` and the calculated effect size was large for the `svmRadial` model. Performance of the other models was not significantly affected by pca. For the ensembles, principal component analysis did not appear to affect the results obtained using the training data but there was a decrease in performance with the test data. The significance of this decrease was not formally evaluated so the question remains as to whether the potential advantages of pca such as decreased complexity and decreased risk of overfitting can be achieved without compromising accuracy.

References

1. Akay, Mehmet Fatih. 2009. "Support Vector Machines Combined with Feature Selection for Breast Cancer Diagnosis." Expert Systems with Applications 36(2). Elsevier: 3240–7.

2. Borges, Lucas Rodrigues. 1989. "Analysis of the Wisconsin Breast Cancer Dataset and Machine Learning for Breast Cancer Detection." Proceedings of XI Workshop de Visão Computacional - October 05th-07th, 2015 Group1(369).

3. DeSantis, Carol E, Jiemin Ma, Ann Goding Sauer, Lisa A Newman, and Ahmedin Jemal. 2017. "Breast Cancer Statistics, 2017, Racial Disparity in Mortality by State." CA: A Cancer Journal for Clinicians 67(6). Wiley Online Library: 439–48.

4. Dua, Dheeru, and Casey Graff. 2017. "UCI Machine Learning Repository." University of California, Irvine, School of Information; Computer Sciences. http://archive.ics.uci.edu/ml.

5. Şahan, Seral, Kemal Polat, Halife Kodaz, and Salih Güneş. 2007. “A New Hybrid Method Based on Fuzzy-Artificial Immune System and K-NN Algorithm for Breast Cancer Diagnosis.” Computers in Biology and Medicine 37(3). Elsevier: 415–23.