Breast Cancer Prediction with Random Forest and Support Vector Machine

Sudan Zhang, Yi Wei, Longfei Li
Department of Computer Science/Data Informatics
University of Southern California
sudanzha@usc.edu, wei256@usc.edu, longfeil@usc.edu

Abstract

In order to gain deeper understanding of the threats of breast cancers and to predict whether the cancer is benign or malignant for faster treatments, we decide to apply machine learning techniques: Decision Tree with Random Forest and Support Vector Machine, to a set of Breast Cancer Diagnostic data set from University of Wisconsin as training data set and use it to test and to generate predictions for breast cancer.

Introduction

Breast cancer, according to the American Cancer Society, is caused by cells in the breast begin to grow out of control and usually, forming a tumor that can often be seen on an x-ray or felt as a lump. The tumor can be benign if it does not spread outside of the breast, on the other hand, the tumor is malignant if the cells can grow into surrounding tissues or spread to distant areas of the body. Given a valuable dataset for breast cell nuclei characteristics, breast cancer prediction can be done using two very popular machine learning techniques, one is called Decision Tree with Random Forest, the other is called Support Vector Machine (SVM). We decided with these two because both techniques are for supervised learning models, hence the data set has labels (classes). In our training data set from University of Wisconsin, the data are given attribute labels (classes) along with an outcome, benign (b) or malignant (m). For the learning algorithm, we utilized Scikit-Learn's Random Forest and SVM library functions, which could save us more time from implementing the algorithm and allow us to focus on the analysis of the data.

Our objective is to train our models with the training data set and eventually, they can output prediction outcomes when supplied with only values for each of the attribute. Before we analyze the data, we calculated the correlations amongst data attributes, filtering out the data who correlates most to other data points, thus increase efficiency or accuracy. The reason for us to choose decision tree with random forest is that we would like to predict or to classify the value of a target variable, in our case, the characteristic of the cancer, from several input variables. Then to form a better approximation of the prediction, we use random forest to generate multitude of decision trees, thus allowing a data to be run through all the trees and generating an average or weighted average of all the terminal nodes those are reached, hence the result prediction is more accurate based on the majority outcome.

To further consolidate our prediction for breast cancer, we decided to use Support Vector Machine (SVM) to classify the data into the two target classes. Since SVM can distinguish a margin which separates two classes of data, we then can apply a kernel method to the model and map the data into higher dimensional spaces to find a hyperplane that divides two class of data. Eventually, after identifying the margin that clearly classifies two classes of data from the training set, we can then map the test data into the same space and arrive at a class prediction base on which side of the margin it falls into. We also applied Principal Component Analysis (PCA) to reduce the dimensions of the data and test whether it is linear separable using SVM.

Dataset and Features

We collected our data from University of Wisconsin's machine learning database. The data is very rich in features, since the attributes are computed and collected from a digitized image of a fine needle aspirate (FNA) of a breast mass, in which the attributes describe the characteristics of cell nuclei in the image.

The attribute is consist of the following features:

1) ID number
2) Diagnosis (M = malignant, B = benign)

3-32) Ten real-valued features are computed for each cell nucleus:

a) radius (mean of distances from center to points on the perimeter)
b) texture (standard deviation of gray-scale values)
c) perimeter
d) area
e) smoothness (local variation in radius lengths)
f) compactness (perimeter^2 / area - 1.0)
g) concavity (severity of concave portions of the contour)
h) concave points (number of concave portions of the contour)
i) symmetry
j) fractal dimension ("coastline approximation" - 1)

The mean, standard error and worst or largest of these features were computed for each image. For instance,

field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius. All feature values are recorded with four significant digits. There are no missing attributes and the final class distribution is 357 benign and 212 malignant.

Initial Analysis

The dataset has three groups of data: mean, se (standard error), worst. Each group has 10 features: 'radius', 'texture', 'perimeter', 'area', 'smoothness', 'compactness', 'concavity', 'concave points', 'symmetry', 'fractal dimension'.

Mean feature group (Lists of features):

['radius_mean', 'texture_mean', 'perimeter_mean', 'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean', 'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean']

Standard error feature group (Lists of features):

['radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se', 'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se', 'fractal_dimension_se']

Worst feature group (Lists of features):

['radius_worst', 'texture_worst', 'perimeter_worst', 'area_worst', 'smoothness_worst', 'compactness_worst', 'concavity_worst', 'concave points_worst', 'symmetry_worst', 'fractal_dimension_worst']

For the most accurate model, we compared data set after correlation filtering with the original data set, then picked out attribute that will likely to contribute the most to the decision tree algorithm.

Analysis of correlations between features

First, we analyzed the correlation between every feature using heatmap. If the value of cell is bigger than 0.8, then these two features of the cell can be recognized correlated.

1) The heatmap of features of mean data :

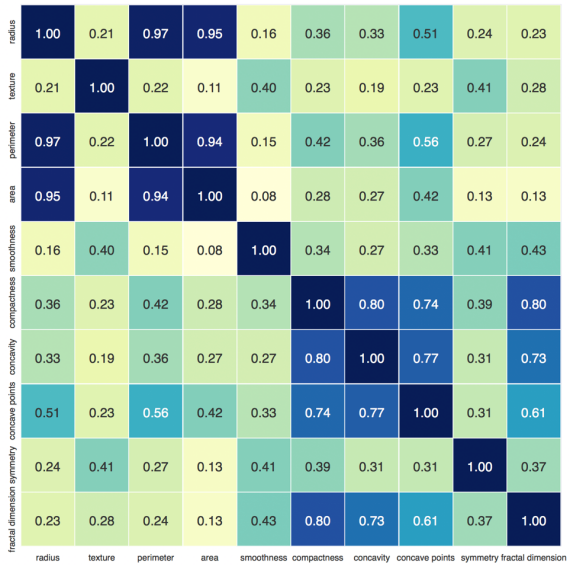
correlation<radius, perimeter>=1.00
correlation<radius, area>=0.99
correlation<perimeter, area>=0.99.

Here, radius, perimeter and area are highly correlated. So we will take either one attribute of these three attributes to represent all.

correlation<compactness, concavity>=0.88
correlation<compactness, concave points>=0.83
correlation<concavity, concave points>=0.92

Again, compactness, concavity and concave points are highly correlated. So we will take either one attribute of these three attributes to represent all. In this research, we will use 'radius', 'texture', 'smoothness', 'compactness', 'symmetry', 'fractal dimension' of mean to build random forest.

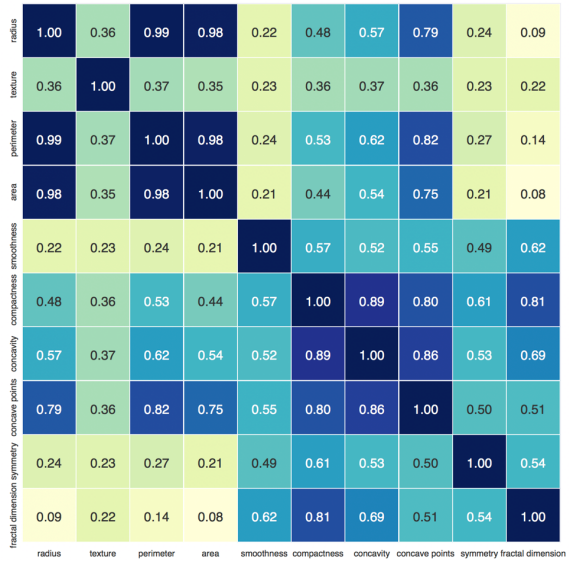
2) The heatmap of features of se data:



correlation<radius,perimeter>=0.97
correlation<radius,area>=0.95
correlation<perimeter,area>=0.94

Thus, radius, perimeter, area are highly correlated. So we will take either one attribute of these three attributes to represent all. In this research, we will use 'radius', 'texture', 'smoothness', 'compactness', 'concavity', 'concave points', 'symmetry', 'fractal_dimension' of standard error to build random forest.

3) the heatmap of features of worst data:



correlation<radius, perimeter> =0.99
correlation<radius, area>=0.98
correlation<perimeter, area>=0.98

Here, radius, perimeter, area are highly correlated. So we will take either one attribute of these three attributes to represent all.

correlation<compactness, concavity>=0.89
correlation<compactness, concave points>=0.80
correlation<concavity, concave points>=0.86

Again, compactness, concavity, concave points are highly correlated. So we will take either one attribute of these three attributes to represent all. In this research, we will use 'radius', 'texture', 'smoothness', 'compactness', 'symmetry', 'fractal dimension' of worst to build random forest.

Implementation of Random Forest

We used train_test_split() function from sklearn to get training data and test data for the random forest.
In this research we take test_size as 0.3: train, test = train_test_split(data, test_size = 0.3)

First, we get the instance model of random forest using RandomForestClassifier(). Then, we use fit(*X*, *y*, *sample_weight=None*) to build a forest of trees from training set(X,y). We use predict(X) to get the predicted class. The predicted class of an input sample is a vote by the trees in the forest, weighted by their probability estimates. That is, the predicted class is the one with highest mean probability estimate across the trees. Finally, we calculate the accuracy rate using accuracy_score(*y_true,y_pred,normalize=True,sample_weight=Non*)

Output:

```
Accuracy of Random Forest model built with filtered dataset:
mean features: 0.93567251462
standard error features: 0.842105263158
worst features: 0.964912280702
----------------------------------------------------------
```

Thus, worst data set is more reliable using random forest algorithm.

Accuracy of each random forest model produced with each group:

```
Accuracy of Random Forest model built with original dataset:
mean features: 0.941520467836
standard error features: 0.900584795322
worst features: 0.947368421053
----------------------------------------------------------
```

In producing Random Forest model, every feature will contribute differently. With the scikit learn library, we can get the importance of each feature by using feature_importances_ method.

```
f_importance_m=pd.Series(rf_m.feature_importances_,index=features_mean).sort_values(ascending="False")
f_importance_se=pd.Series(rf_se.feature_importances_,index=features_se).sort_values(ascending="False")
f_importance_w=pd.Series(rf_w.feature_importances_,index=features_worst).sort_values(ascending="False")
```

Then, we used pandas library to save them into a more readable format. It can be printed out like:

The importance of mean features

```
fractal_dimension_mean    0.016227
symmetry_mean             0.016990
smoothness_mean           0.025476
compactness_mean          0.031523
texture_mean              0.066012
radius_mean               0.137341
area_mean                 0.149438
perimeter_mean            0.156678
concavity_mean            0.158432
concave points_mean       0.241883
```

The importance of standard error features

```
compactness_se            0.029518
texture_se                0.039654
smoothness_se             0.041200
fractal_dimension_se      0.046565
symmetry_se               0.052744
concave points_se         0.059544
concavity_se              0.065917
perimeter_se              0.151115
radius_se                 0.196242
area_se                   0.317501
```

The importance of worst features

```
fractal_dimension_worst   0.019872
symmetry_worst            0.025055
smoothness_worst          0.027878
compactness_worst         0.037688
texture_worst             0.038904
concavity_worst           0.065616
radius_worst              0.154100
concave points_worst      0.154691
area_worst                0.235526
perimeter_worst           0.240670
```

We want to create a new feature set by combining the top three important features in each group, trying to see if the most important features can produce a more accurate model. Thus we put the names of the top 3 important features in each group into a new list, and use it to produce a new random forest model and compute the accuracy.

New list consists of top three important features of mean, standard error and worst group:
['radius_mean', 'concavity_mean', 'concave points_mean', 'radius_se', 'perimeter_se', 'area_se', 'concave points_worst', 'radius_worst', 'perimeter_worst']

Accuracy of each group and new list of features:

```
Accuracy of Random Forest model built with original dataset and new dataset:
mean features: 0.941520467836
standard error features: 0.900584795322
worst features: 0.947368421053
top 3 features of each group: 0.93567251462
-----------------------------------------------------
```

But as we can see from the result, the accuracy of new model is not higher than the accuracy of model which made with the worst group. All in all, the worst feature group is the most efficient feature set for producing a random forest model, as well as the most accurate attribute set to predict whether the tumor is benign.

Implementation of Support Vector Machine

Aside from using Decision Tree algorithm and Random Forest algorithm, we also applied Support Vector Machine (SVM) for classification in this research. For producing a svm model, we can import the svm package from sklearn, and get the instance of svm using svm.SVC(). Then we use fit(*X*, *y*, *sample_weight=None*) to fit the SVM model according to the given training data. Finally we use predict(*X*) to perform classification on test data.

```
from sklearn import svm
model = svm.SVC()
model.fit(train_X, train_Y)
prediction = model.predict(test_X)
```

We first calculate the accuracy rate for correlation filtered data set using **accuracy_score**(*y_true*, *y_pred*, *normalize=True*, *sample_weight=Non*), and the output is the following:

```
Accuracy of SVM model built with filtered dataset:
mean features: 0.906432748538
standard error features: 0.771929824561
worst features: 0.970760233918
-----------------------------------------------------
```

Thus, worst data set is more reliable for SVM classification after correlation filtering.

Then we input the original data set for accuracy calculation and the output is:

```
Accuracy of SVM model built with original dataset:
mean features: 0.672514619883
standard error features: 0.883040935673
worst features: 0.625730994152
-----------------------------------------------------
```

In contrast to the result of random forest, the accuracy of svm model using standard error feature group is the highest one, and it is much higher than the others. But the average of accuracies of svm models is less than the average of random forest models. The cause of this result might be the different principle of these two algorithms (Decision Tree algorithm and Support Vector Machine algorithm). SVM algorithm is to find a line or a hyperplane that can distinct two classes, so when the data points are not linearly separable (existing outliers or serious not linearly separable in this space), the svm classifier might be less reliable. But we can apply kernel function to svm, to map

the data points into a space with higher dimension. And the points might be linearly separable in this new space.

With sklearn library, we can use linear kernel function and radial basis function (rbf) kernel, and we also can use a custom python function as kernel function. So I plan to use rbf kernel and linear kernel to improve the svm classifier and compare the result of each model.

```
clf_svm_linear=svm.SVC(kernel="linear")
clf_svm_rbf=svm.SVC(kernel="rbf")
```

The accuracy of svm with rbf kernel:

```
Accuracy of SVM model(with rbf kernel) built with filtered dataset:
mean features: 0.906432748538
standard error features: 0.771929824561
worst features: 0.970760233918
------------------------------------------------------------

Accuracy of SVM model(with rbf kernel) built with original dataset:
mean features: 0.672514619883
standard error features: 0.883040935673
worst features: 0.625730994152
------------------------------------------------------------
```

The accuracy of svm with linear kernel:

```
Accuracy of SVM model(with linear kernel) built with filtered dataset:
mean features: 0.900584795322
standard error features: 0.795321637427
worst features: 0.976608187135
------------------------------------------------------------

Accuracy of SVM model(with linear kernel) built with original dataset:
mean features: 0.93567251462
standard error features: 0.883040935673
worst features: 0.982456140351
------------------------------------------------------------
```

Compare with the svm classifier built without kernel, applying rbf kernel does not heavily influence the accuracy of model, while applying linear kernel apparently improve the accuracy of model made with the worst feature group and mean feature group. Thus we can simply conclude that build svm classifier with linear kernel can improve the performance of this model, and will make the worst feature group became the most efficient feature group for producing the model.

Implementation of Principal Component Analysis

We use the Principal Component Analysis (PCA) algorithm to transform the data into two dimensional space and then use the SVM to make the classification.

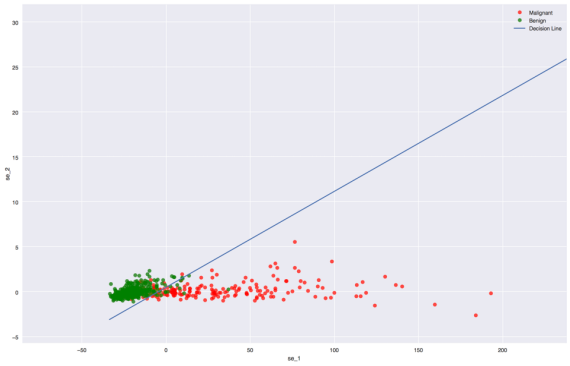
First, using to get a instance of pca for two dimensional space. Then using fit(*X*, *y=None*) to fit the model with X. Then transform(*X*, *y=None*) to Apply dimensionality reduction to X and return a array with two columns, each element of the array is a new coordinate after reduction in dimensionality. Finally apply the SVM to the new data.

For each kind of Data we draw a picture of data points after PCA and the decision line of SVM(using linear kernel) to separate data.

1) Features_mean

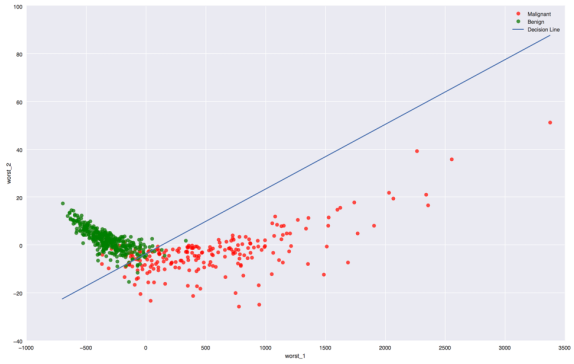
The accuracy is 0.888888888889

2) Features_se



The accuracy is 0.87134502924

3) Features_worst



The accuracy is 0.923976608187

Thus, for all three kinds of data the output is

```
Accuracy of SVM model(with linear kernel) after using pca built with original dataset:
mean features: 0.888888888889
standard error features: 0.87134502924
worst features: 0.923976608187
------------------------------------------------------------
```

So, the worst features is more desirable to make the classification using SVM (with linear kernel) after PCA algorithm.

Conclusion

In our paper, we proposed the experiment of Breast Cancer Diagnostic data using Decision Tree with Random Forest and Support Vector Machine with some Principal Component Analysis. From our experiment of Random Forest, we concluded that using "worst" attribute group values from correlation filtered data set produces the most accurate training model. Whereas, mean attribute and standard error attribute group values produces less accurate training models.

For Support Vector Machine (SVM) model without kernel function, we also found that using "worst" attribute group values from correlation filtered data set produces the most accurate training model out of the three attribute groups, but when using original data set, mean attribute and standard error attribute group values produces training models those are not desirable. It can be concluded that correlation filtering effects SVM models' grouping dramatically.

When using a linear kernel function for SVM, the "worst" attribute group values produces the most accurate training models, and this is also the most accurate models out of all of the experiments we conducted. Since radial basis function (rbf) kernel's accuracy is not so desirable, the outcome is negligible.

After all, from the data set that we used, the "worst" attribute produces accurate prediction and it should be used in future predictions for patients. Since the "worst" attribute is the mean of three of the largest values from the data set, we recommend the patients to go to hospital and have doctors perform checks on a regular basis in order to acquire more data about his/her conditions and to produce more accurate prediction.

Contribution

Our group member Sudan Zhang focused mainly on the Support Vector Machine (SVM) experiment with the data set and reporting of her findings. Yi Wei focused on the correlational filtering, Decision Tree with Random Forest and Principal Component Analysis. Yi also reported his findings with graphs. Longfei Li collected all the data, graphs and reports from other two members, and collaboratively, he was able to write up this report with the help from other two teammates. Longfei also did extensive research on breast cancer prediction and how machine learning could help the patients.

References

1.4. Support Vector Machines. Scikit-learn developers , n.d. Web. 2 May 2017. <http://scikit-learn.org/stable/modules/svm.html>.

3.2.4.3.1. sklearn.ensemble.RandomForestClassifier. Scikit-learn developers, n.d. Web. 2 May 2017. <http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn-ensemble-randomforestclassifier>.

Benyamin, Dan. "A Gentle Introduction to Random Forests, Ensembles, and Performance Metrics in a Commercial System." *CitizenNet*. N.p., 9 Nov. 2012. Web. 1 May 2017. <http://blog.citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics>.

What is Breast Cancer? The American Cancer Society medical and editorial content team, 18 Aug. 2016. Web. 1 May 2017. <https://www.cancer.org/cancer/breast-cancer.html>.

Wolberg, William H., Dr, Nick Street, and Olvi L. Mangasarian. *Breast Cancer Wisconsin (Diagnostic) Data Set* . University of Wisconsin, 1 Nov. 1995. Web. 2 May 2017. <https://archive.ics.uci.edu/ml/datasets/Breast Cancer Wisconsin %28Diagnostic%29>.