Detecting Breast Cancer

Abstract

We attempt to build a model to predict if a breast tumor is benign or malignant using past data. We found that using principal component analysis greatly increased the accuracy of the standard logistic regression for classification by adding just two additional columns; this implies that using PCA on breast cancer patients before performing analysis can lead to significant improvements while only adding little run time. We also analyze which components made a tumor significantly more or less likely to be malignant.

Introduction

Whether a tumor is benign or malignant plays a large role in the treatments that should be afforded, including in the case of metastasis. We attempt to determine the factors that would make breast tumor more likely to be malignant. As breast cancer is a leading cause of death among women, detecting malign tumors accurately is of paramount importance.

Methods

We use a dataset (n=569) on breast tumor diagnoses from the University of Wisconsin, which contains the status of each case, and the characteristics, including the radius and the compactness. The dataset is available via Kaggle, and can be downloaded in CSV format: https://www.kaggle.com/uciml/breast-cancer-wisconsin-data/data.
We use logistic regression as well as k-nearest neighbors for classification. In each model, the diagnosis is the dependent variable, and all other variables are independent. For logistic regression, we run two models: We use only the data given for the first model. In order to interpret the coefficients more efficiently, we scale the independent variables to a [0, 1] scale. For the second model, we initially perform PCA and use the PCA results as one of our variables, to detect improvement in results due to PCA. We use a small sample of our variables in order to observe correlation and to gauge the effect of preprocessing using PCA. For k-nearest neighbors, we decided to use k=13 based on a chart of error vs k value. We use all independent variables except the perimeter (which are linearly related to the radius) in our kNN model.

Results

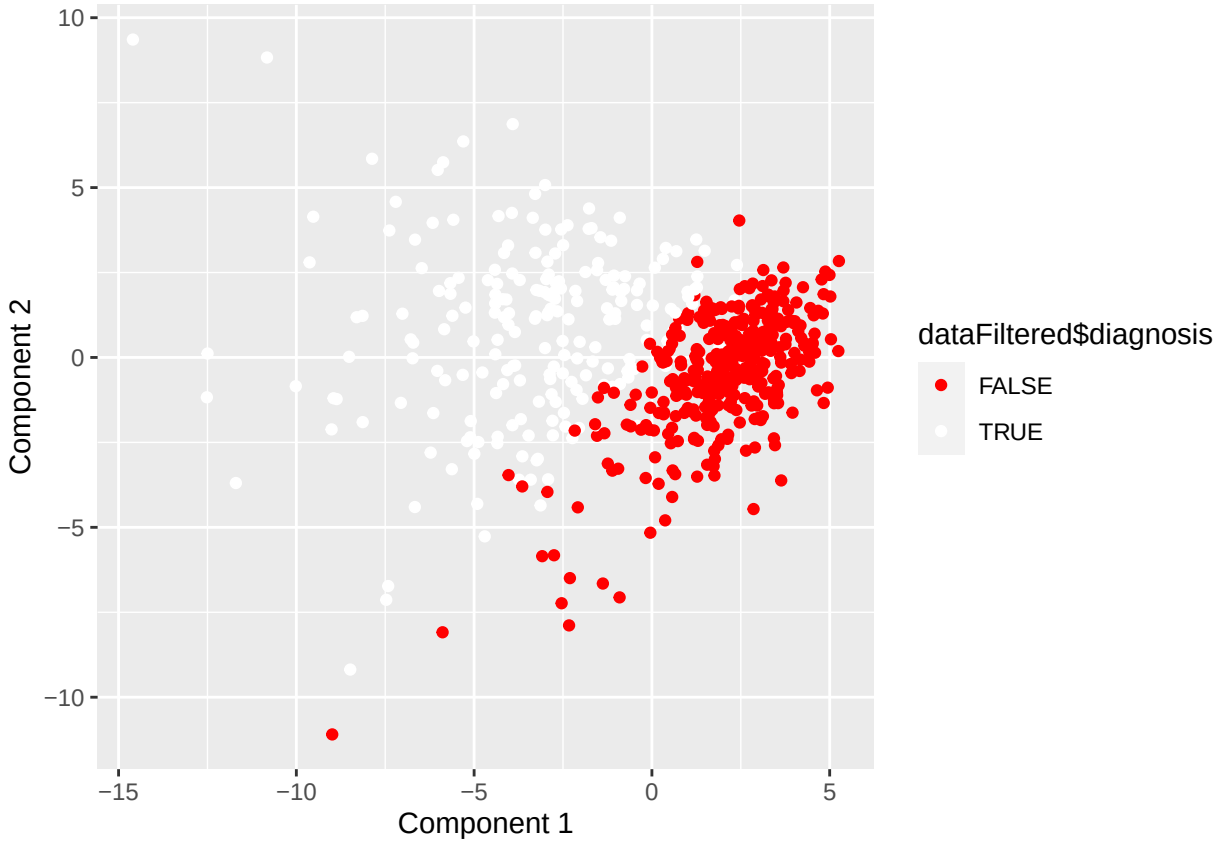
```
knitr::opts_chunk$set(echo = TRUE)
options(warn=-1)
library(tibble)
library(ggplot2)
library(FNN)
library(mosaicData)
library(foreach)
library(cluster)
data <- read.csv("~/Documents/SDS323Assignments/breastcancerdata.csv")
data$diagnosis = (data$diagnosis == 'M')
xData = subset(data, select = -c(X,id,diagnosis))
```

```
dataFiltered = subset(data, select = -c(X, id, perimeter_mean, perimeter_worst))
sum(dataFiltered$diagnosis)/length(dataFiltered$diagnosis)
```

```
## [1] 0.3725835
```

We noted that the radius and perimeter have virtually perfect correlation. (See the Appendix) This is expected as the perimeter of a circle is a constant multiple of its radius. We therefore decided to exclude the perimeter and only include the radius for our logistic model. We also note that a model which predicts False (benign) every time will get 63% accuracy (100-37). This is our null model and our benchmark. We first run PCA for dimensionality reduction:

```
pc2 = prcomp(dataFiltered[ ,!(colnames(dataFiltered) == c('diagnosis'))], scale=TRUE, rank=2)
scores = pc2$x
qplot(scores[,1], scores[,2], color=dataFiltered$diagnosis, title = 'Figure 1', xlab='Component 1', yla
```



We first run logistic regression without using our PCA, in order to get insight on which factors influence the malignant/benign nature the most. When we used all variables, we noticed very large coefficients, making them hard to interpret. We therefore only take a subset of the dependent variables. We also scale the variables we are using to make the results easier to interpret:

```
n = nrow(dataFiltered)
n_train = round(0.8*n)
n_test = n - n_train
FP = 0
FN = 0
TP = 0
TN = 0
dataFiltered$radius_mean = scale(dataFiltered$radius_mean)
dataFiltered$texture_mean = scale(dataFiltered$texture_mean)
```

```
dataFiltered$smoothness_mean = scale(dataFiltered$smoothness_mean)
dataFiltered$compactness_mean = scale(dataFiltered$compactness_mean)
dataFiltered$concavity_mean = scale(dataFiltered$concavity_mean)
dataFiltered$symmetry_mean = scale(dataFiltered$symmetry_mean)
for(i in 1:100)
{
train_cases = sample.int(n, n_train, replace=FALSE)
test_cases = setdiff(1:n, train_cases)
data_train = dataFiltered[train_cases,]
data_test = dataFiltered[test_cases,]
logit = glm(diagnosis ~ radius_mean + texture_mean + smoothness_mean + compactness_mean + concavity_mean
yhat_test = predict(logit, data_test)
pred = (yhat_test > -1)
actual = data_test$diagnosis
TP = TP + (sum(actual & pred)) / n_test
TN = TN + (sum(!actual & !pred)) / n_test
FP = FP + (sum(!actual & pred)) / n_test
FN = FN + (sum(actual & !pred)) / n_test
}
TP
```

```
## [1] 34.25439
```

```
TN
```

```
## [1] 58.21053
```

```
FP
```

```
## [1] 5.201754
```

```
FN
```

```
## [1] 2.333333
```

```
TP + TN
```

```
## [1] 92.46491
```

```
logit["coefficients"]
```

```
## $coefficients
##      (Intercept)      radius_mean     texture_mean  smoothness_mean
##       -0.8036485        4.8956879        1.8108481        1.5822166
## compactness_mean   concavity_mean    symmetry_mean
##       -0.4991713        1.8812710        0.3311077
```

It appears that higher radius and concavity significantly increases the probability of the tumor being malignant, while fractal dimension decreases it. We obtained an accuracy around 94%, but notably falsely diagnose 3.8% of the cases as benign. Because the consequences of misdiagnosing a malignant tumor as benign is so great (i.e. untreated growth), we try setting a lower threshold than 50%:

```
n = nrow(dataFiltered)
n_train = round(0.8*n)
n_test = n - n_train
FP = 0
FN = 0
TP = 0
TN = 0
```

```
for(i in 1:100)
{
train_cases = sample.int(n, n_train, replace=FALSE)
test_cases = setdiff(1:n, train_cases)
data_train = dataFiltered[train_cases,]
data_test = dataFiltered[test_cases,]
logit = glm(diagnosis ~ radius_mean + texture_mean + smoothness_mean + compactness_mean + concavity_mean
yhat_test = predict(logit, data_test)
pred = (yhat_test > -3)
actual = data_test$diagnosis
TP = TP + (sum(actual & pred)) / n_test
TN = TN + (sum(!actual & !pred)) / n_test
FP = FP + (sum(!actual & pred)) / n_test
FN = FN + (sum(actual & !pred)) / n_test
}
TP
```

```
## [1] 37.46491
```

```
TN
```

```
## [1] 47.14912
```

```
FP
```

```
## [1] 14.71053
```

```
FN
```

```
## [1] 0.6754386
```

```
TP + TN
```

```
## [1] 84.61404
```

We now have many fewer cases of undetected malignant tumors, but more false positives, which could lead to more unnecessary and potentially invasive treatments. Which threshold is "better" is a matter of subjective opinion. We noticed above that the PCA graph seemed to separate the malignant and benign tumors fairly well. We try running the regression using the PCA values:

```
n = nrow(dataFiltered)
n_train = round(0.8*n)
n_test = n - n_train
FP = 0
FN = 0
TP = 0
TN = 0
dataFiltered$pca1 = scores[,1]
dataFiltered$pca2 = scores[,2]
for(i in 1:100)
{
train_cases = sample.int(n, n_train, replace=FALSE)
test_cases = setdiff(1:n, train_cases)
data_train = dataFiltered[train_cases,]
data_test = dataFiltered[test_cases,]
logit = glm(diagnosis ~ radius_mean + texture_mean + smoothness_mean + compactness_mean + concavity_mean
yhat_test = predict(logit, data_test)
pred = (yhat_test > 0)
actual = data_test$diagnosis
```

```
TP = TP + (sum(actual & pred)) / n_test
TN = TN + (sum(!actual & !pred)) / n_test
FP = FP + (sum(!actual & pred)) / n_test
FN = FN + (sum(actual & !pred)) / n_test
}
TP
```

```
## [1] 35.50877
```

```
TN
```

```
## [1] 60.94737
```

```
FP
```

```
## [1] 1.701754
```

```
FN
```

```
## [1] 1.842105
```

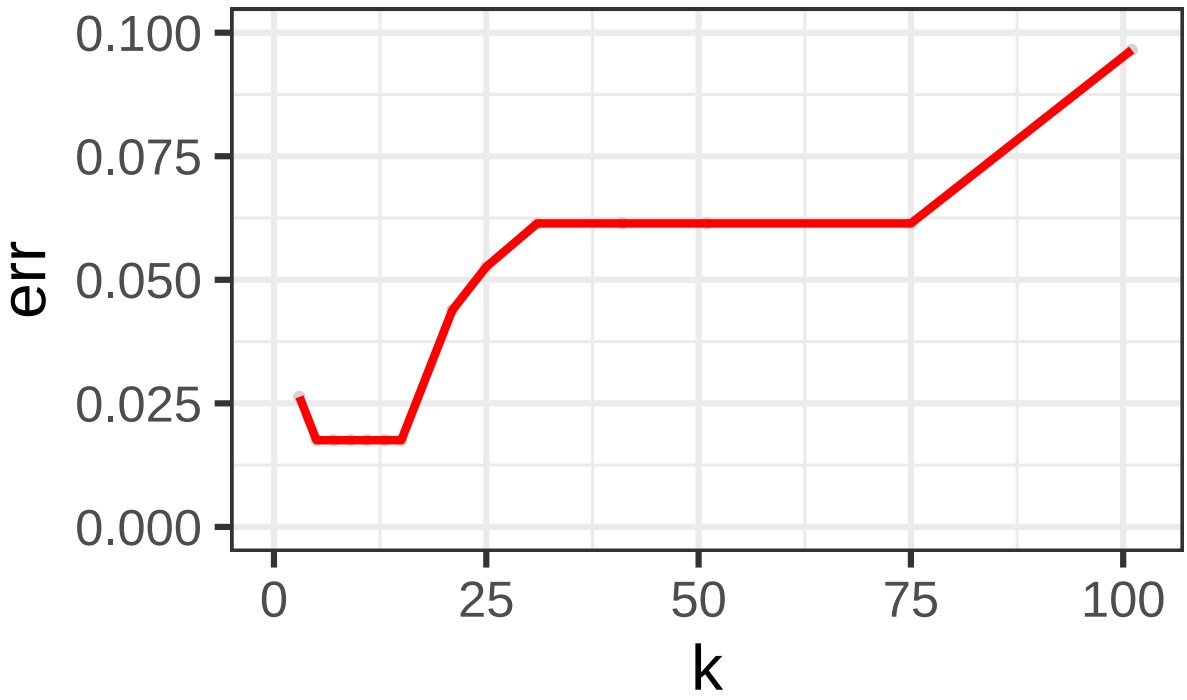
```
TP + TN
```

```
## [1] 96.45614
```

We notice an improvement in both the accuracy and the the false negative rate. We now run k-nearest neighbors:

```
dataFilteredScaled = scale(dataFiltered[ ,!(colnames(dataFiltered) == "diagnosis")], center=TRUE, scale=
n = length(dataFiltered$diagnosis)
n_train = round(0.8*n)
n_test = n - n_train
train_ind = sample.int(n, n_train)
kValues = c(3, 5, 7, 9, 11, 13, 15, 21, 25, 31, 41, 51, 75, 101)
y = dataFiltered$diagnosis
X = dataFilteredScaled
X_train = X[train_ind,]
X_test = X[-train_ind,]
y_train = y[train_ind]
y_test = y[-train_ind]
actual = y_test
df = data.frame(k=integer(0),err=numeric(0))
for(i in kValues)
{
  knn = class::knn(train = X_train, test = X_test, cl = y_train, k = i)
  pred = knn['pred']
  err = sum(knn != y_test) / n_test
  df = add_row(df, k = i, err = err)
}
p_train = ggplot(data = df) +
  geom_point(mapping = aes(x = k, y = err), color='lightgrey') +
  theme_bw(base_size=24) +
  ylim(0, 0.1) + xlim(0,102)
p_train = p_train + geom_path(mapping = aes(x = k, y=err), color='red', size=1.5) + ggtitle("Figure 2")
p_train
```

Figure 2



There doesn't seem to be a clear optimal value for k, except k<20. We choose k=13.

```
dataFilteredScaled = scale(dataFiltered[ ,!(colnames(dataFiltered) == "diagnosis")], center=TRUE, scale=
n = length(dataFiltered$diagnosis)
n_train = round(0.8*n)
n_test = n - n_train
kValues = c(3, 5, 7, 9, 11, 13, 15, 21, 25, 31, 41, 51, 75, 101)
y = dataFiltered$diagnosis
X = dataFilteredScaled
actual = y_test
df = data.frame(k=integer(0),err=numeric(0))
acc = 0
FN = 0
for(i in 1:100)
{
  train_ind = sample.int(n, n_train)
  X_train = X[train_ind,]
  X_test = X[-train_ind,]
  y_train = y[train_ind]
  y_test = y[-train_ind]
  knn = class::knn(train = X_train, test = X_test, cl = y_train, k = 13)
  acc = acc + (sum(knn == y_test) / n_test)
  FN = FN + (sum((knn != y_test) & (y_test)) / n_test)
}
acc
```

```
## [1] 96.42982
```

```
FN
```

```
## [1] 3.105263
```

Our accuracy is comparable to that of logistic regression with PCA, but out false negative rate is higher. Thus, we can argue that logistic regression with PCA is the more effective model.

Conclusion

To summarize our results, PCA with 2 components differentiates malignant and benign tumors fairly efficiently based on Figure 1. Indeed, when running logistic regression, just by adding the two variables (the two components) we noticed a substantial increase in accuracy. This result can be helpful when attempting to classify future cases, especially when larger datasets are used, since we can substantially improve the model by adding just two more columns. We also found that, all other factors equal, the scaled radius and concavity tend to significantly increase the probability of a tumor being malignant. For k-nearest neighbors, we find that the value of k doesn't have a lot of impact except that k generally shouldn't exceed 20 (Figure 2). The result had comparable accuracy to the logistic regression with PCA, but a higher false negative rate. False negatives are very problematic when diagnosing cancer, as they lead to exacerbation of the condition; it should be taken into account separately along with accuracy.

Appendix

We display the correlations between the means of each variable.

```
cor(xData[c('radius_mean', 'texture_mean', 'perimeter_mean', 'area_mean', 'smoothness_mean', 'compactnes
```

```
##                         radius_mean texture_mean perimeter_mean  area_mean
## radius_mean               1.0000000   0.32378189      0.9978553  0.9873572
## texture_mean              0.3237819   1.00000000      0.3295331  0.3210857
## perimeter_mean            0.9978553   0.32953306      1.0000000  0.9865068
## area_mean                 0.9873572   0.32108570      0.9865068  1.0000000
## smoothness_mean           0.1705812  -0.02338852      0.2072782  0.1770284
## compactness_mean          0.5061236   0.23670222      0.5569362  0.4985017
## concavity_mean            0.6767636   0.30241783      0.7161357  0.6859828
## concave.points_mean       0.8225285   0.29346405      0.8509770  0.8232689
## symmetry_mean             0.1477412   0.07140098      0.1830272  0.1512931
## fractal_dimension_mean   -0.3116308  -0.07643718     -0.2614769 -0.2831098
##                         smoothness_mean compactness_mean concavity_mean
## radius_mean                  0.17058119        0.5061236      0.6767636
## texture_mean                -0.02338852        0.2367022      0.3024178
## perimeter_mean               0.20727816        0.5569362      0.7161357
## area_mean                    0.17702838        0.4985017      0.6859828
## smoothness_mean              1.00000000        0.6591232      0.5219838
## compactness_mean             0.65912322        1.0000000      0.8831207
## concavity_mean               0.52198377        0.8831207      1.0000000
## concave.points_mean          0.55369517        0.8311350      0.9213910
## symmetry_mean                0.55777479        0.6026410      0.5006666
## fractal_dimension_mean       0.58479200        0.5653687      0.3367834
##                         concave.points_mean symmetry_mean fractal_dimension_mean
## radius_mean                       0.8225285    0.14774124            -0.31163083
## texture_mean                      0.2934641    0.07140098            -0.07643718
## perimeter_mean                    0.8509770    0.18302721            -0.26147691
## area_mean                         0.8232689    0.15129308            -0.28310981
## smoothness_mean                   0.5536952    0.55777479             0.58479200
## compactness_mean                  0.8311350    0.60264105             0.56536866
## concavity_mean                    0.9213910    0.50066662             0.33678336
## concave.points_mean               1.0000000    0.46249739             0.16691738
```

```
## symmetry_mean                  0.4624974    1.00000000             0.47992133
## fractal_dimension_mean         0.1669174    0.47992133             1.00000000
```