ISYE6501 HW4

Keh-Harng Feng

June 6, 2017

0.1 Preface

This is a reproducible report with most of the codes doing the heavy lifting hidden in the background. Someone wondered why I don't show my code on my report. The reason is simple: the code chunks are often VERY long and the information you need as a reviewer is in the report, not the code. As always, it is still available for completeness and reproducibility. You can download the source code of the report by clicking here.

1 Question 1

Using the same crime data set as in Homework 3 Question 4, apply Principal Component Analysis and then create a regression model using the first 4 principal components. Specify your new model in terms of the original variables (not the principal components), and compare its quality to that of your solution to Homework 3 Question 4. You can use the R function prcomp for PCA. (Note that to first scale the data, you can include scale. = TRUE to scale as part of the PCA function.)

1.1 Model Consturction

The data set is split into a test (~10%) and training (~90%) with the same random seed used in my HW3 report. Principle components are computed for all of the predictors **except** `So` because it is categorical.

Figure 1 shows the cumulative amount of variance explained by the first four principle components. Without `So` they already explain over 99% of the variance.

The final model is trained using the first four principle components plus `So` as the predictors. The final model using principle components contains the following coefficients:

```
## (Intercept)         PC1         PC2         PC3         PC4 factor(So)1
##      714.99     -159.64       22.11      -15.04     -141.60      508.84
```

Define a 4x1 column vector $\bar{M}$ containing the model coefficients *except* the intercept and the coefficient for `So`. The equation of the line constructed by linear regression can be written in terms of the original predictors as follows:

$$\bar{Y} = (Intercept) + \bar{X} \times \hat{P} \times \bar{M} + X_{So} M_{So} \tag{1}$$

where

$\bar{X}$: 1x14 vector of centered & scaled predictors used in the principle components.

$\hat{P}$: 14x4 rotation matrix of the first 4 principle components.

X_{So}: The `So` predictor.

M_{So}: scalar model `So` coefficient.

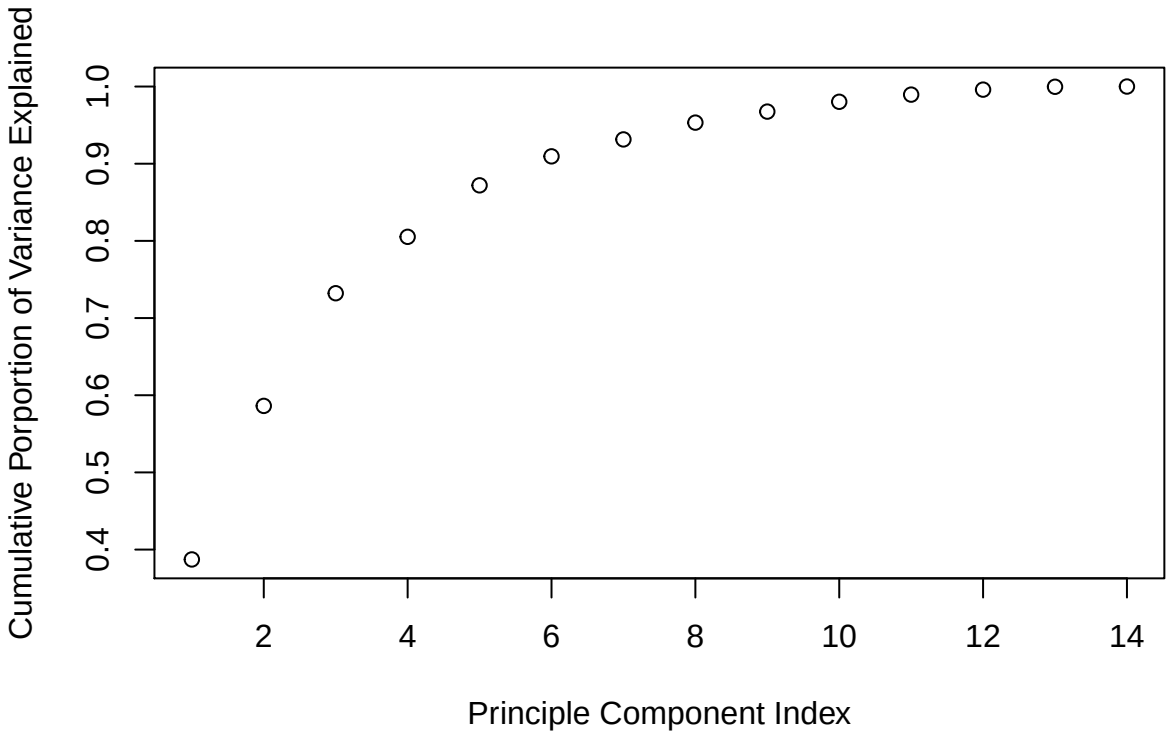

Figure 1: Cumulative porportion of variance explained by the principle components.

Writing out the model using the original predictors, we have:

$Crime = 714.988666953386 - 8.52629201506481 * (M - 13.9418604651163) + 46.0169692963876 * (Ed - 10.4860465116279) + 14.7864835410493 * (Po1 - 8.43023255813953) + 14.6266792679287 * (Po2 - 7.96511627906977) + 1616.10194560122 * (LF - 0.559418604651163) + 19.0080910663664 * (M.F - 98.3232558139535) + 0.984098065207822 * (Pop - 36.1395348837209) + -4.03686171479741 * (NW - 10.7162790697674) + 1492.62643870274 * (U1 - 0.095046511627907) + 25.5249041324182 * (U2 - 3.4093023255814) + 0.0561447608001619 * (Wealth - 5198.13953488372) + -9.34831382074279 * (Ineq - 19.5813953488372) + -5399.58251854536 * (Prob - 0.0480371860465116) + 11.5225629087635 * (Time - 26.4791069767442) + 508.842957562774 * So$

Table 1 shows the results computed from the above equation using regular predictors directly from the test set and the predictions using `predict()` on the PCA model with the predictors in the test set transformed accordingly. The equation is indeed correct.

1.2 Model Performance

The model using the first four principle components achieved a MSE of 7.9558×10^4 on the test set. The model I built in HW3 using the same training set was able to achieve a MSE of 1.3309e4 on the same test set. **The PC model has worse performance.** It should be noted however that my HW3 model was built with extensive feature selection along with data transformation assisted by exploratory data analysis. It is also possible that my HW3 model was overfitted (especially since the sample was so small and there was a bit of test set contamination due to transformation being carried out prior to training/test split - a mistake that won't be repeated here.)

Table 1: Prediction values using direct equation vs prediction values using PCA model.

Equation	PCA
910.6	910.6
1107.4	1107.4
1035.8	1035.8
896.1	896.1

2 Question 2

Using the same crime data set as in Homework 3 Question 4, find the best model you can using (a) a regression tree model, and (b) a random forest model. In R, you can use the tree package or the rpart package, and the randomForest package. For each model, describe one or two qualitative takeaways you get from analyzing the results (i.e., don't just stop when you have a good model, but interpret it too).

2.1 Part A: Regression Tree

As usual, the data is split into the same training set and test set as before. A regression tree model is built using `rpart` in conjunction with `train()` from `caret` on the training set. Automatic tuning of the `cp` parameter was done using cross-validation. No transformation on the data was done prior to model training since tree algorithims are generally not sensitive to things such as predictor/response distributions. Model information is shown below:

```
## CART
##
## 43 samples
## 15 predictors
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 39, 39, 38, 38, 39, 40, ...
## Resampling results across tuning parameters:
##
##   cp       RMSE   Rsquared
##   0.05267  327.1  0.4292
##   0.19033  322.7  0.4228
##   0.37957  370.1  0.2758
##
## RMSE was used to select the optimal model using  the smallest value.
## The final value used for the model was cp = 0.1903.
```

A graphical representation of the tree is shown in Figure 2

Incredibly the final tree model only makes use of ONE predictor, `Po2`. It can only make TWO different predictions: 666 or 1154 for the crime rate. Decisions are made by checking the `Po2` predictor values against 7.2. Its performance will be evaluated later on the test set.

2.2 Part B: Random Forest

A bit of parameter tuning is done to create a model random forest. The out-of-sample MSE is estimated using 10-fold CV for forests ranging from 50 trees to 5000 in increment of 50. The resulting average MSE is plotted in Figure 3.

While overall there is no discernible decrease to the average of estimated MSE as the number of trees increases, the variance in MSE does become smaller. This seems to indicate that the more trees a random forest has, the more stable it is in making prediction. The highest number of trees tested, ntrees = 5000, is chosen to build the final random forest model.

Unfortunately, unlike regression trees there are no simple ways to visualize the forest effectively. Interpretation of random forest models is always a difficult if not down right impossible task due to the aggregation of many

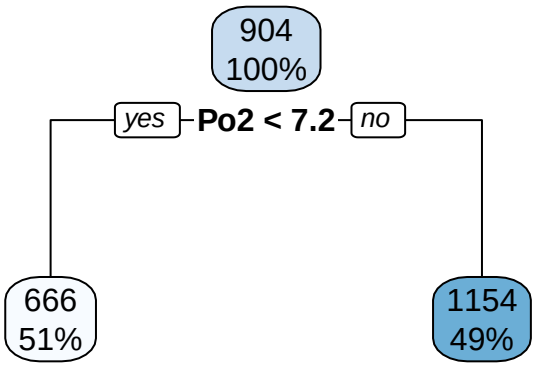

Figure 2: The regression tree model.

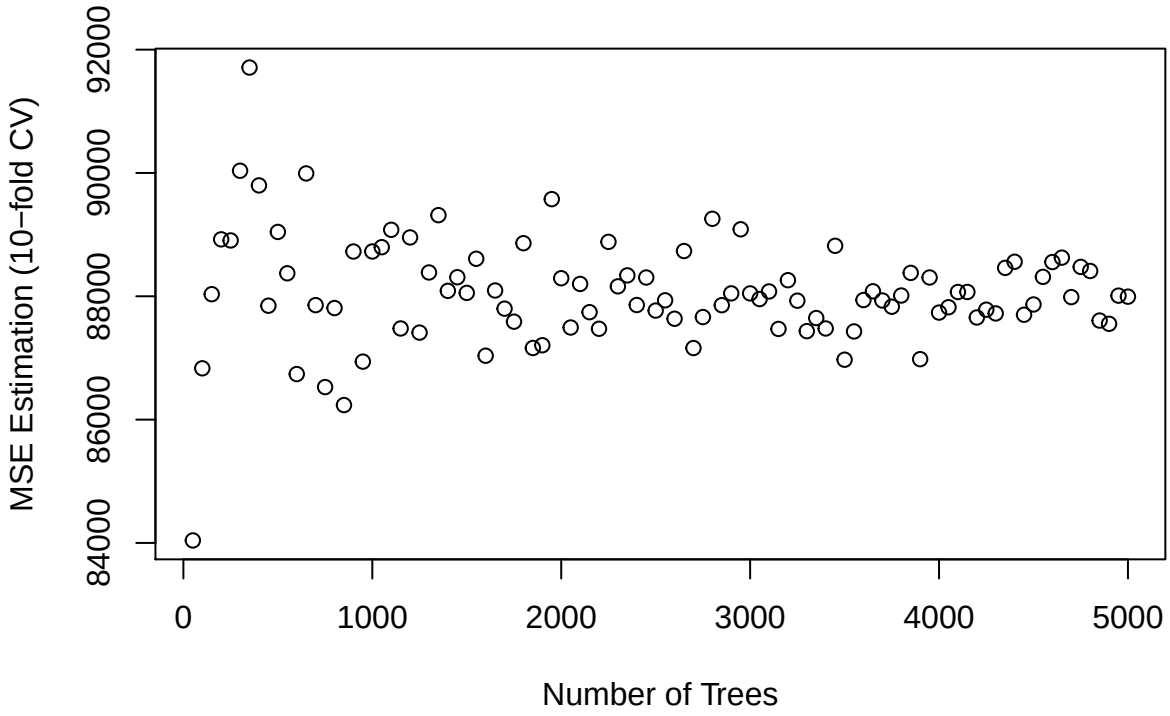

Figure 3: Estimated OOS MSE vs Number of Trees in Forest

trees together. Whille it is possible to graphically show a single tree in the forest like in Figure 2, it is not a statistically meaningful representation of the forest.

2.3 Performance Evaluation

The RMSE of the four different models I have built so far are shown in Table 2. All models are trained on the same training set and tested on the same test set. It is clear that the predictor transformation and selection algorithm I came up with for HW3 either paid off or got lucky with this test set - the simple multivariate linear regression model is the best performer out of the bunch. This is followed not so closely by random forest, then regression tree and finally the PCA MLR model. The performance of the regression tree surprised me since it is only capable of making two prediction values yet it did better than PCA MLR. Perhaps it's caused by the distribution of the response or the specific test set used.

Table 2: RMSE of the 4 different models trained for the uscrime data.

Model	RMSE
HW3 MLR	115.4
Q1 PCA MLR	282.1
RPART Tree	237.2
Random Forest	193.3

3 Question 3

Describe a situation or problem from your job, everyday life, current events, etc., for which a logistic regression model would be appropriate. List some (up to 5) predictors that you might use.

With the ISYE6501 mid-term approaching, many people are no doubt wondering if they will pass the test or not. A logistic regression model can be constructed to predict the outcome. The response is the success/fail of a student on his midterm test. Some of the predictors (assuming they can be systematically quantified) can be:

`Grades on Homework Assignments`

`Level of Participation in Office Hours`

`Grades on the Sample Quiz (before reading through the answers)`

`Time Spent on Preparation`

4 Question 4

4.1 Part 1

Using the GermanCredit data set at http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german / (description at http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29), use logistic regression to find a good predictive model for whether credit applicants are good credit risks or not. Show your model (factors used and their coefficients), the software output, and the quality of fit. You can use the glm function in R. To get a logistic regression (logit) model on data where the response is either zero or one, use family=binomial(link="logit") in your glm function call.

The data is loaded and column 21 is identified as the categorical response. A bit of data manipulation is done to redefine "Bad" as 0 and "Good" as 1. The response is then converted to a factor.

As usual the data is then split into a training (80%) and test (20%) set. A logistic regression model using all predictors is first constructed to allow backwards stepwise feature selection using `stepAIC()` from the `MASS` library. The final model statistics is shown below:

```
##
## Call:
## glm(formula = V21 ~ V1 + V2 + V3 + V4 + V5 + V6 + V8 + V9 + V10 +
##     V14 + V15 + V19 + V20, family = binomial(link = "logit"),
##     data = q4_data.train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.876  -0.734   0.371   0.699   2.251
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.00e-01   7.46e-01   -0.54  0.59232
## V1A12        5.26e-01   2.44e-01    2.15  0.03126 *
## V1A13        1.59e+00   4.29e-01    3.70  0.00021 ***
## V1A14        1.80e+00   2.55e-01    7.05  1.8e-12 ***
## V2          -2.74e-02   1.00e-02   -2.73  0.00630 **
## V3A31       -3.41e-01   5.96e-01   -0.57  0.56728
```

```
## V3A32        2.80e-01   4.73e-01    0.59  0.55434
## V3A33        9.59e-03   5.38e-01    0.02  0.98578
## V3A34        9.36e-01   4.97e-01    1.88  0.05982 .
## V4A41        1.65e+00   4.03e-01    4.10  4.2e-05 ***
## V4A410       1.11e+00   8.43e-01    1.31  0.18944
## V4A42        6.35e-01   2.82e-01    2.26  0.02410 *
## V4A43        8.69e-01   2.75e-01    3.16  0.00160 **
## V4A44        1.30e+00   9.58e-01    1.35  0.17562
## V4A45        5.84e-02   6.14e-01    0.10  0.92425
## V4A46        3.36e-01   4.44e-01    0.76  0.44992
## V4A48        2.16e+00   1.28e+00    1.69  0.09152 .
## V4A49        4.16e-01   3.67e-01    1.13  0.25659
## V5          -1.62e-04   4.76e-05   -3.41  0.00066 ***
## V6A62        3.68e-01   3.20e-01    1.15  0.24963
## V6A63        5.55e-01   4.35e-01    1.28  0.20195
## V6A64        1.98e+00   6.65e-01    2.98  0.00287 **
## V6A65        1.08e+00   2.83e-01    3.84  0.00012 ***
## V8          -3.29e-01   9.68e-02   -3.40  0.00067 ***
## V9A92       -2.66e-01   4.18e-01   -0.64  0.52505
## V9A93        5.32e-01   4.09e-01    1.30  0.19333
## V9A94       -9.55e-02   4.99e-01   -0.19  0.84809
## V10A102     -3.19e-01   4.58e-01   -0.70  0.48623
## V10A103      1.07e+00   4.85e-01    2.20  0.02763 *
## V14A142     -1.40e-01   4.52e-01   -0.31  0.75641
## V14A143      5.70e-01   2.63e-01    2.17  0.03036 *
## V15A152      5.27e-01   2.46e-01    2.14  0.03197 *
## V15A153      3.82e-01   3.65e-01    1.05  0.29448
## V19A192      4.77e-01   2.09e-01    2.28  0.02240 *
## V20A202      1.25e+00   6.44e-01    1.95  0.05146 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 990.56  on 799  degrees of freedom
## Residual deviance: 722.05  on 765  degrees of freedom
## AIC: 792.1
##
## Number of Fisher Scoring iterations: 5
```

It should be noted that some of the selected features display bad P-values. However they are all dummy variables created from the factors they belong to and as a whole one or more other levels in that particular factor are always found to be statistically significant. The final model can be written as:

$P = -0.399720453032344 + 0.525603732999066 * V1A12 + 1.58822345267653 * V1A13 + 1.79558235500307 * V1A14 + -0.0273860923925631 * V2 + -0.340876406598862 * V3A31 + 0.279507964933386 * V3A32 + 0.00958697912220112 * V3A33 + 0.935526502150113 * V3A34 + 1.65076571887356 * V4A41 + 1.10568599546412 * V4A410 + 0.634970660847489 * V4A42 + 0.868950955082708 * V4A43 + 1.29734545968168 * V4A44 + 0.0583734392382542 * V4A45 + 0.335770299624084 * V4A46 + 2.16302564228504 * V4A48 + 0.41614205841517 * V4A49 + -0.000162050415615924 * V5 + 0.368381074600474 * V6A62 + 0.555051882413809 * V6A63 + 1.98226184114851 * V6A64 + 1.08498289964085 * V6A65 + -0.329293580378144 * V8 + -0.265938588050582 * V9A92 + 0.532053130718026 * V9A93 + -0.0955446937108228 * V9A94 + -0.318578936947573 * V10A102 + 1.06931213138456 * V10A103 + -0.140176180378961 * V14A142 + 0.570114814683447 * V14A143 + 0.527229061390038 * V15A152 + 0.382361100362198 * V15A153 + 0.476968866043074 * V19A192 + 1.25411565631081 * V20A202$

Notice that the model computes the *probability* of the response being positive (ie: credit should be classfied as "Good"), rather than the actual response labels themselves.

4.1.1 Performance Evaluation

Predictions are made using the logistic regression model on the test set. To convert predicted probabilities to the proper response labels, a sample cutoff is set at 0.5 (ie: P > 0.5 -> classify as "Good"). The resulting confusion matrix statistics are shown below:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Bad Good
##       Bad   24   24
##       Good  28  124
##
##                Accuracy : 0.74
##                  95% CI : (0.673, 0.799)
##     No Information Rate : 0.74
##     P-Value [Acc > NIR] : 0.537
##
##                   Kappa : 0.307
##  Mcnemar's Test P-Value : 0.677
##
##             Sensitivity : 0.838
##             Specificity : 0.462
##          Pos Pred Value : 0.816
##          Neg Pred Value : 0.500
##              Prevalence : 0.740
##          Detection Rate : 0.620
##    Detection Prevalence : 0.760
##       Balanced Accuracy : 0.650
##
##        'Positive' Class : Good
##
```

4.2 Part 2

Because the model gives a result between 0 and 1, it requires setting a threshold probability to separate between "good" and "bad" answers. In this data set, they estimate that incorrectly identifying a bad customer as good, is 5 times worse than incorrectly classifying a good customer as bad. Determine a good threshold probability based on your model.

A 2-nested grid search is carried out with 1000 intervals on each level to find the optimal threshold probability level that minimizes the cost on the test set. Figure 4 shows the cost as a function of threshold.

The probability threshold P = 0.6872 results in the minimum cost 103.

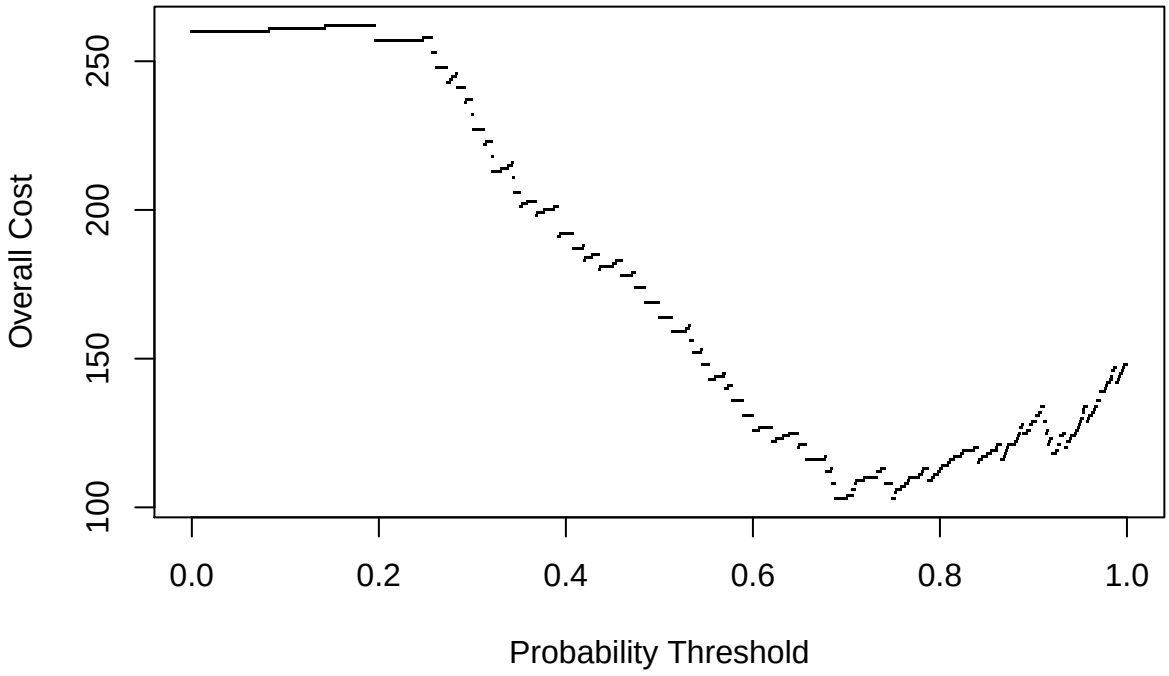

Figure 4: Cost vs Probability Threshold