Hwk #2: Classification methods and Penalization

For this homework we will use NHANES data that exists in a package for R.

NHANES consists of survey data collected by the US National Center for Health Statistics (NCHS) which has conducted a series of health and nutrition surveys since the early 1960's. Since 1999 approximately 5,000 individuals of all ages are interviewed in their homes every year and complete the health examination component of the survey. The health examination is conducted in a mobile examination center (MEC).

Note that there is the following warning on the NHANES website: "For NHANES datasets, the use of sampling weights and sample design variables is recommended for all analyses because the sample design is a clustered design and incorporates differential probabilities of selection. If you fail to account for the sampling parameters, you may obtain biased estimates and overstate significance levels."

For this homework, please ignore this warning and just apply our analyses to the data as if they were randomly sampled! We will be using the data called `NHANESraw`.

For questions that ask for your comments, it suffices to answer with one or two sentences in each case.

Data Preparation

1. Install the package `NHANES` into R, load the `NHANES` package, and then run the command `data(NHANES)` which will load the NHANES data. Type `?NHANES` and read about the dataset.

```
library(NHANES)
data('NHANES')
?NHANES
```

```
## starting httpd help server ... done
```

2. Make an object `nhanes` that is a subset version `NHANESraw` that does not include any missing data for `Diabetes`, `BPSysAve`, `BPDiaAve`, or `Age`.

```
nhanes <- subset(NHANESraw, !is.na(NHANESraw$Diabetes) & !is.na(NHANESraw$BPSysAve) & !is.na(NHA
NESraw$BPDiaAve) & !is.na(NHANESraw$Age))
```

3. (1 point) Further subset the data such the observations with `BPDiaAve` equal to zero are removed.

```
nhanes_sub <- subset(nhanes, nhanes$BPDiaAve!=0)
```

4. (1 point) Make an object `nhanes09` that is a subset of `nhanes` to only the 2009_10 data. This will be your training dataset. Also make an object `nhanes11` that is a subset of `nhanes` to only the 2011_12 data. This will be your test dataset.

```
nhanes09 <- subset(nhanes, nhanes$SurveyYr=='2009_10')
nhanes11 <- subset(nhanes, nhanes$SurveyYr=='2011_12')
```

Logistic regression

5. (2 point) Fit a logistic regression model (call it `glm1`) using the `nhanes09` dataset. Use `Diabetes` as the outcome and averaged systolic blood pressure (`BPSysAve`) as a single predictor. Use the summary command to examine the fitted model. Generate the 95% confidence intervals for the `BPSysAve` coefficient.

```
glm1 <- glm(Diabetes ~ BPSysAve , data=nhanes09, family = 'binomial')
summary(glm1)
```

```
##
## Call:
## glm(formula = Diabetes ~ BPSysAve, family = "binomial", data = nhanes09)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.65798    0.22316  -25.35   <2e-16 ***
## BPSysAve     0.02901    0.00175   16.57   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5307.0  on 7810  degrees of freedom
## Residual deviance: 5040.1  on 7809  degrees of freedom
## AIC: 5044.1
##
## Number of Fisher Scoring iterations: 5
```

```
confint(glm1, level=0.95)
```

```
## Waiting for profiling to be done...
```

```
##                 2.5 %      97.5 %
## (Intercept) -6.097706 -5.22264299
## BPSysAve     0.025585  0.03244903
```

6. (1 point) Generate the estimate and 95% confidence interval for the odds-ratio associated with BPSysAve. Summarize the result.

```
OR_est <- exp(coef(glm1)['BPSysAve'])
OR_ci <- exp(confint(glm1, 'BPSysAve'))
```

```
## Waiting for profiling to be done...
```

```
cat("Odds Ratio is", OR_est, "\n")
```

```
## Odds Ratio is 1.029436
```

```
cat("95%CI is", OR_ci, "\n")
```

```
## 95%CI is 1.025915 1.032981
```

7. (1 point) Predict the probabilities of diabetes associated with each of the training observations of `BPSysAve`. Make a vector of predictions for diabetes based on whether the predictions are above or below 0.5.

```
prob <- predict(glm1, type = 'response')
predictions <- ifelse(prob>0.5, 1, 0)
```

8. (1 point) Generate a confusion matrix that shows the number of false positives, false negatives, true positives, and true negatives in the training data. The rows should correspond to the true diabetes status and the columns should correspond to the predicted values.

```
table(actual = nhanes09$Diabetes, predictions)
```

```
##       predictions
## actual    0    1
##    No  6961   16
##    Yes  827    7
```

9. (1 point) Find the proportion of correctly classified observations in the training data.

```
(6961+7)/(6961+16+827+7)
```

```
## [1] 0.8920753
```

10. (2 points) Now repeat questions 7 to 9 but for predicting the test dataset.

```
prob_test <- predict(glm1, newdata=nhanes11, type = "response")
predict_test <- ifelse(prob_test>0.5,1,0)
# Confusion Matrix
table(actual=nhanes11$Diabetes, predict_test)
```

```
##       predict_test
## actual    0    1
##    No  6269   13
##    Yes  763    4
```

```
# The proportion of correctly classified observations
(6269+4)/(6269+13+763+4)
```

```
## [1] 0.8899135
```

11. (1 point) Comment on the difference in results between the training and test prediction tables and classification accuracies.

The accuracies in test data is a little bit lower than the test data one. It might be caused by some outliers and the difference of distribution between the two data.

12. (1 point) Manually calculate the sensitivity and specificity estimates for the test dataset based on the 0.5 threshold.

```
# Sensitivity
4/(4+763)
```

```
## [1] 0.005215124
```

```
# Specificity
6269/(6269+13)
```

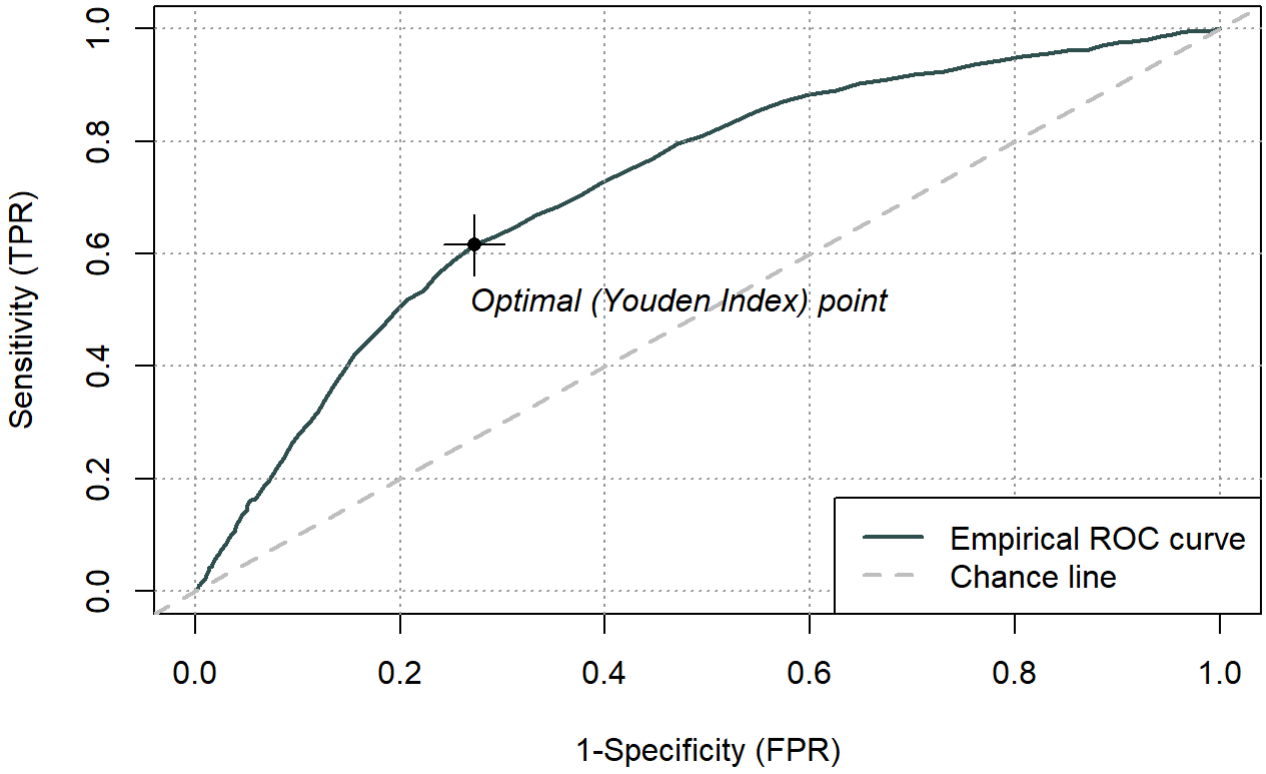
```
## [1] 0.9979306
```

13. (2 points) Generate an ROC curve using the test data. What is the AUC and its 95% confidence interval?

```
library(ROCit)
```

```
## Warning: package 'ROCit' was built under R version 4.3.2
```

```
roc <- rocit(score = prob_test, class = nhanes11$Diabetes)
plot(roc)
```




```
ciAUC(roc,level=0.95)
```

```
##
##    estimated AUC : 0.720726153281639
##    AUC estimation method : empirical
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.699532555254025     upper = 0.741919751309252
```

14. What value can you use to threshold the predicted probability to achieve a sensitivity of at least 0.6 and a specificity of at least 0.7?

```
sen <- roc$TPR
spe <- 1-roc$FPR
cutoff <- roc$Cutoff

thresholds <- cutoff[sen >= 0.6 & spe >= 0.7]
thresholds
```

```
## [1] 0.1159204 0.1129803
```

15. (2 points) Comment on the results of the analyses for the different thresholds in terms of the tables, classification accuracies, and sensitivity and specificity. Under what circumstances might you prefer each of the thresholds?

```
# Accuracy at the threshold=0.11
predict_test <- ifelse(prob_test>0.11,1,0)
table(actual=nhanes11$Diabetes, predict_test)
```

```
##       predict_test
## actual    0    1
##    No  4320 1962
##    Yes  270  497
```

```
(4320+497)/(1962+270+4320+497)
```

```
## [1] 0.6833593
```

When the threshold is set to 0.5, the model emphasizes specificity, resulting in lower sensitivity. The accuracy is higher at this threshold, which can be attributed to the fact that the original data contains a larger proportion of non-disease cases, leading the model to correctly identify these cases more frequently. On the other hand, when the threshold is lowered to 0.1, there is a better balance between sensitivity and specificity. This balance can also be inferred from the AUC curve. The higher accuracy at a threshold of 0.5 is due to the model's focus on correctly predicting the more prevalent non-disease outcomes.

16. (2 points) Fit a multiple predictor logistic regression (call it `glm2`) with `Diabetes` as outcome and predictors: `BPSysAve`, `BPDiaAve`, and `Age`. Use the `summary` command to examine the fitted model and determine the estimated coefficients, odds-ratios, and 95% confidence intervals thereof.

```
glm2 <- glm(Diabetes ~ BPSysAve + BPDiaAve + Age, data=nhanes09, family = 'binomial')
summary(glm2)
```

```
##
## Call:
## glm(formula = Diabetes ~ BPSysAve + BPDiaAve + Age, family = "binomial",
##     data = nhanes09)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.061135   0.274094 -18.465   <2e-16 ***
## BPSysAve     0.004576   0.002224   2.057   0.0397 *
## BPDiaAve    -0.002893   0.002767  -1.045   0.2958
## Age          0.051445   0.002386  21.557   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5307.0  on 7810  degrees of freedom
## Residual deviance: 4452.6  on 7807  degrees of freedom
## AIC: 4460.6
##
## Number of Fisher Scoring iterations: 6
```

```
OR_est <- exp(coef(glm2)['BPSysAve'])
OR_ci <- exp(confint(glm2, 'BPSysAve'))
```

```
## Waiting for profiling to be done...
```

```
cat("Odds Ratio is", OR_est, "\n")
```

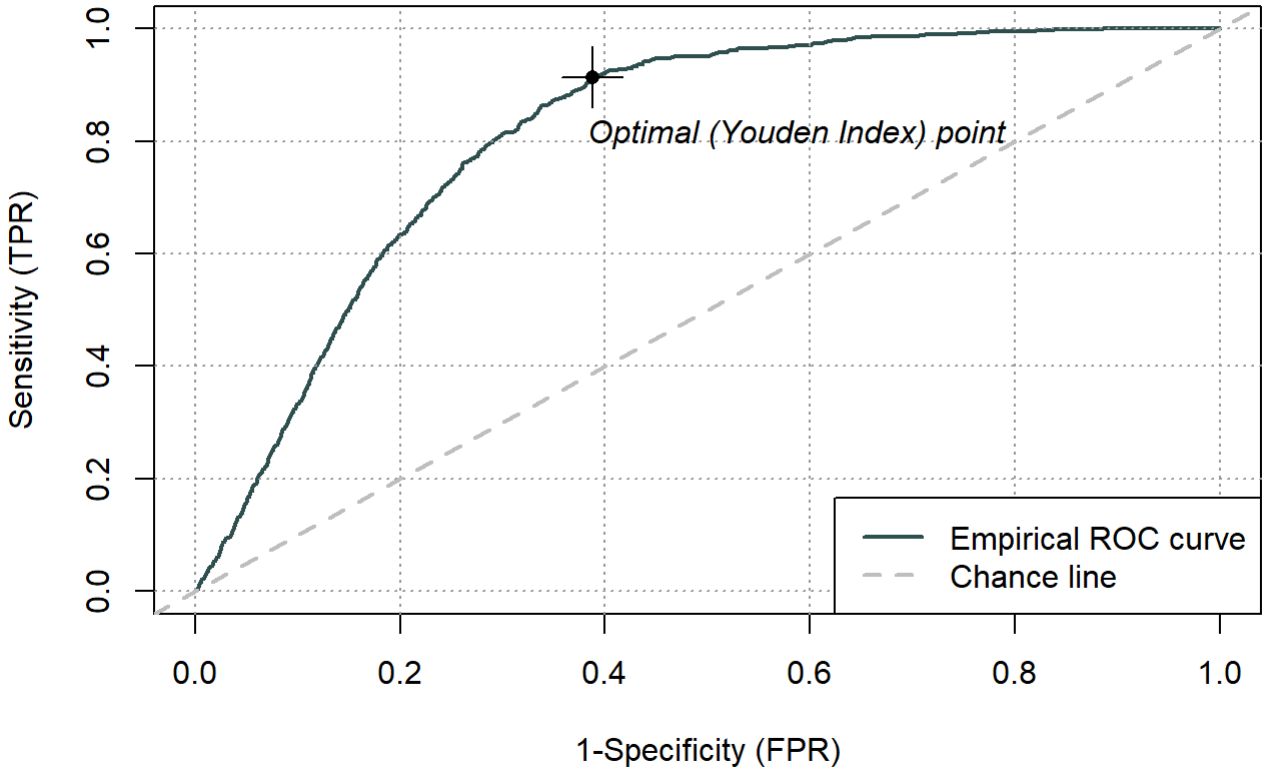
```
## Odds Ratio is 1.004587
```

```
cat("95%CI is", OR_ci, "\n")
```

```
## 95%CI is 1.000201 1.008964
```

17. (2 points) Generate an ROC curve for the `glm2` model using the test data. What is the AUC and its 95% confidence interval?

```
prob_test <- predict(glm2, newdata=nhanes11, type = "response")
roc <- rocit(score = prob_test, class = nhanes11$Diabetes)
plot(roc)
```




```
ciAUC(roc,level=0.95)
```

```
##
##    estimated AUC : 0.813645970959846
##    AUC estimation method : empirical
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.79479653699808     upper = 0.832495404921612
```

18. (1 point) What is the maximum sensitivity level you can achieve if we require the specificity to be at least 0.7?

```
sen <- roc$TPR
spe <- 1-roc$FPR
cutoff <- roc$Cutoff

max_sen <- max(sen[spe >= 0.7])
max_sen
```

```
## [1] 0.8135593
```

19. (1 point) Would you prefer the single predictor or multiple predictor model if your objective was to maximize classification accuracy, and which threshold level would you choose? Comment on the reason for your choices.

I prefer the multiple predictor morel to the single predictor morel because the AUC is bigger than the single predictor model's. I also chose a higher threshold to increase the specificity and the accuracy because the original data contains a larger proportion of non-disease cases.

Linear discriminant analysis

20. (2 points) Fit a linear discriminant analysis (`lda1`) with `Diabetes` as outcome and predictors of `BPSysAve` , `BPDiaAve` , and `Age` in the training dataset. Examine the fit by typing `lda1` .

```
lda1 <- lda(Diabetes ~ BPSysAve + BPDiaAve + Age, data=nhanes09)
lda1
```

```
## Call:
## lda(Diabetes ~ BPSysAve + BPDiaAve + Age, data = nhanes09)
##
## Prior probabilities of groups:
##        No       Yes
## 0.8932275 0.1067725
##
## Group means:
##     BPSysAve BPDiaAve      Age
## No  116.4810 64.96560 37.67866
## Yes 128.3321 66.06355 60.84053
##
## Coefficients of linear discriminants:
##                    LD1
## BPSysAve  0.009533227
## BPDiaAve -0.015657491
## Age       0.044706982
```

21. (2 points) Generate the confusion matrix for `lda1` using the test set. Compute the classification accuracy, sensitivity, and specificity.

```
ldapred <- predict(lda1, newdata=nhanes11)
table(class=nhanes11$Diabetes, pred=ldapred$class)
```

```
##      pred
## class   No  Yes
##   No  6239   43
##   Yes  748   19
```

```
# Classification accuracy
mean(ldapred$class == nhanes11$Diabetes)
```

```
## [1] 0.8877855
```

```
# Sensitivity
19/(748+19)
```

```
## [1] 0.02477184
```

```
# Specifiity
6239/(6239+43)
```

```
## [1] 0.993155
```

22. How do these measures compare with that of the logistic regression model with these predictors and 0.5 threshold?

To compare these models, we use sensitivity, specificity, and precision. The specificity and precision are almost the same as the glm model, but the sensitivity of the lda model is better than the glm model.

23. (3 points) Redo question 21 but with prior probabilities set to 0.5 for diabetes.

```
lda2 <- lda(Diabetes ~ BPSysAve + BPDiaAve + Age, data=nhanes11, prior=c(0.5, 0.5))

ldapred <- predict(lda2, newdata=nhanes11)
table(class=nhanes11$Diabetes, pred=ldapred$class)
```

```
##       pred
## class   No  Yes
##   No  4592 1690
##   Yes  178  589
```

```
# Classification accuracy
mean(ldapred$class == nhanes11$Diabetes)
```

```
## [1] 0.7349979
```

```
# Sensitivity
589/(589+178)
```

```
## [1] 0.767927
```

```
# Specifiity
4592/(4592+1690)
```

```
## [1] 0.7309774
```

24. (2 points) Comment on how LDA's performance changed when we changed the prior probabilities.

Changing the prior probability, the performance is well-balanced. This might be because there is a disparity of the outcome in the training data and the previous model is built based on the imbalanced probability. Accuracy, sensitivity, and specificity are affected by the frequency of the outcome's occurrence, therefore, these values are changed by setting the probability.

Penalized regression

26. Read in the dementia data "dementia.csv" into a data frame called `dementia_dat` . This dataset contains measurements obtained from MRI brain scans and whether or not the patient has dementia. We'll try to build a prediction model for diagnosing dementia based on these derived measurements. How many observations are in this dataset? How many predictors are in this dataset?

```
dementia_dat <- read.csv('dementia.csv')
# Observations
nrow(dementia_dat)
```

```
## [1] 660
```

```
# Predictors
ncol(dementia_dat)
```

```
## [1] 142
```

27. Load the `glmnet` and `caret` packages.

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.3.2
```

```
## Loading required package: Matrix
```

```
## Warning: package 'Matrix' was built under R version 4.3.2
```

```
## Loaded glmnet 4.1-8
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
## Loading required package: lattice
```

28. (1 point) Set the random seed to 4 and then split the data into 2 sets (400 train, and 260 test)

```
set.seed(4)
nrow_train <- sample(nrow(dementia_dat), 400, replace=FALSE)
dementia_train <- dementia_dat[nrow_train, ]
dementia_test <- dementia_dat[-nrow_train, ]
```

29. (4 points) Perform cross-validated lasso in the training data to select the optimal penalty parameter lambda. Use 5 folds and search over the range $\lambda = 10^3$ to $\lambda = 10^{-3}$. Set `Dementia` as outcome with all other variables as predictors. Use the `caret` package to do CV.

```
train_control <- trainControl(method="cv", number=5)
caret_grid <- data.frame("lambda" = 10^seq(3, -3), "alpha"= 1)
cv_model <- train(as.factor(Dementia) ~., data=dementia_train, trControl=train_control, method
="glmnet", tuneGrid=caret_grid)
cv_model$results
```

```
##   alpha lambda Accuracy     Kappa AccuracySD    KappaSD
## 1     1  1e-03   0.8750 0.7491072 0.04419417 0.08724037
## 2     1  1e-02   0.8950 0.7894603 0.04383919 0.08736771
## 3     1  1e-01   0.8375 0.6721172 0.04050463 0.08363492
## 4     1  1e+00   0.5500 0.0000000 0.00000000 0.00000000
## 5     1  1e+01   0.5500 0.0000000 0.00000000 0.00000000
## 6     1  1e+02   0.5500 0.0000000 0.00000000 0.00000000
## 7     1  1e+03   0.5500 0.0000000 0.00000000 0.00000000
```

30. (1 point) What is the optimal value of lambda?

```
cv_model$bestTune
```

```
##   alpha lambda
## 2     1   0.01
```

31. (1 point) Generate the confusion matrix for this final model.

```
predictions <- predict(cv_model, newdata = dementia_test)

table(class=dementia_test$Dementia, pred=predictions)
```

```
##      pred
## class  No Yes
##   No  122  13
##   Yes  21 104
```

32. (1 point) How many non-zero coefficients are in the final model?

```
coefficients <- coef(cv_model$finalModel, s = cv_model$bestTune$lambda)
sum(coefficients != 0)
```

```
## [1] 36
```