BUDA535Module4

## Problem 1 :

Revisit the Carseats Problem from homework 3. Under the same scenerio, use the elastic net (using binomial regression), lasso, and ridge regression, to find relevant variables. Discuss any differences you see in the results. Remember in these methods the choice of tuning parameters will dictate how large or small each each of these models are. You should use 5 or 10 fold cross validation to select your tuning parameters. You can use either cv.glment or caret to do the cross validation (caret may be easier). Make sure you discuss the final model and any inisghts you derive from it.

library(ISLR)  
library(caret)

## Warning: package 'caret' was built under R version 3.6.2

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.2

data("Carseats")  
  
Carseats$Sales\_factor <- as.factor(Carseats$Sales > median(Carseats$Sales))  
  
set.seed(7548)  
train\_in <- createDataPartition(Carseats$Sales\_factor,times=1,p=0.75,list=FALSE)  
train\_data <- Carseats[train\_in,]  
test\_data <- Carseats[-train\_in,]  
  
displaybesttuneresults = function(train\_model)  
{  
 bestaccuracy = which(rownames(train\_model$results) == rownames(train\_model$bestTune))  
 bestresults = train\_model$results[bestaccuracy,]  
 rownames(bestresults) = NULL  
 bestresults  
}  
  
#nrow(train\_data)  
#nrow(test\_data)  
#nrow(filter(test\_data,test\_data$Sales\_factor == "TRUE"))  
#nrow(filter(test\_data,test\_data$Sales\_factor == "FALSE"))  
  
fitControl <- trainControl(method="cv", number=10)  
  
# Lasso Regression  
set.seed(7548)  
lasso\_train\_model = train(Sales\_factor~.-Sales,data=train\_data,method = "glmnet", trControl=fitControl, family="binomial", tuneGrid=expand.grid(alpha= 1,lambda=seq(0.1,20,0.5)))  
besttuneResuts.lasso = displaybesttuneresults(lasso\_train\_model)  
besttuneResuts.lasso

## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 1 0.1 0.753871 0.5075052 0.06956444 0.1388098

# Coefficents  
coef(lasso\_train\_model$finalModel, lasso\_train\_model$bestTune$lambda)

## 12 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 1.9521046512  
## CompPrice .   
## Income .   
## Advertising 0.0110827299  
## Population .   
## Price -0.0194035810  
## ShelveLocGood 1.0550192545  
## ShelveLocMedium .   
## Age -0.0001559695  
## Education .   
## UrbanYes .   
## USYes .

# Prediction  
predict.lasso <- predict(lasso\_train\_model, test\_data)  
cm.lasso = confusionMatrix(predict.lasso, test\_data$Sales\_factor, positive = "TRUE")  
cm.lasso

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction FALSE TRUE  
## FALSE 37 13  
## TRUE 13 36  
##   
## Accuracy : 0.7374   
## 95% CI : (0.6393, 0.8207)  
## No Information Rate : 0.5051   
## P-Value [Acc > NIR] : 2.007e-06   
##   
## Kappa : 0.4747   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.7347   
## Specificity : 0.7400   
## Pos Pred Value : 0.7347   
## Neg Pred Value : 0.7400   
## Prevalence : 0.4949   
## Detection Rate : 0.3636   
## Detection Prevalence : 0.4949   
## Balanced Accuracy : 0.7373   
##   
## 'Positive' Class : TRUE   
##

# Ridge Regression  
set.seed(7548)  
ridge\_train\_model = train(Sales\_factor~.-Sales, data=train\_data, method="glmnet", trControl = fitControl, family="binomial", tuneGrid = expand.grid(alpha = 0, lambda=seq(0.1,20,0.5)))  
#ridge\_train\_model  
besttuneResuts.ridge = displaybesttuneresults(ridge\_train\_model)  
besttuneResuts.ridge

## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0 0.1 0.8504301 0.7007252 0.04797635 0.09588944

# Coefficients  
coef(ridge\_train\_model$finalModel,ridge\_train\_model$bestTune$lambda)

## 12 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 0.8360305861  
## CompPrice 0.0237992032  
## Income 0.0066367819  
## Advertising 0.0529745817  
## Population 0.0001156929  
## Price -0.0360512455  
## ShelveLocGood 1.6799265356  
## ShelveLocMedium 0.2451526472  
## Age -0.0198232342  
## Education -0.0012281815  
## UrbanYes 0.0806435083  
## USYes 0.1327719449

# Prediction  
predict.ridge <- predict(ridge\_train\_model,test\_data)  
cm.ridge = confusionMatrix(predict.ridge,test\_data$Sales\_factor, positive = "TRUE")  
cm.ridge

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction FALSE TRUE  
## FALSE 43 7  
## TRUE 7 42  
##   
## Accuracy : 0.8586   
## 95% CI : (0.7741, 0.9205)  
## No Information Rate : 0.5051   
## P-Value [Acc > NIR] : 1.461e-13   
##   
## Kappa : 0.7171   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.8571   
## Specificity : 0.8600   
## Pos Pred Value : 0.8571   
## Neg Pred Value : 0.8600   
## Prevalence : 0.4949   
## Detection Rate : 0.4242   
## Detection Prevalence : 0.4949   
## Balanced Accuracy : 0.8586   
##   
## 'Positive' Class : TRUE   
##

# Elastic net  
set.seed(7548)  
tuning\_grid = expand.grid(alpha=seq(0.0,1,0.025),lambda=seq(0.1,20,0.5))  
elastic\_train\_model = train(Sales\_factor~.-Sales,data=train\_data, method="glmnet",trControl = fitControl, family = "binomial", tuneGrid = tuning\_grid)  
#elastic\_train\_model  
besttuneResuts.elastic = displaybesttuneresults(elastic\_train\_model)  
besttuneResuts.elastic

## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0.025 0.1 0.8570968 0.7140586 0.04471087 0.08937623

# Coefficients  
coef(elastic\_train\_model$finalModel, elastic\_train\_model$bestTune$lambda)

## 12 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 9.076753e-01  
## CompPrice 2.323749e-02  
## Income 6.366991e-03  
## Advertising 5.288319e-02  
## Population 5.771109e-05  
## Price -3.578243e-02  
## ShelveLocGood 1.660185e+00  
## ShelveLocMedium 2.239012e-01  
## Age -1.940289e-02  
## Education .   
## UrbanYes 6.267906e-02  
## USYes 1.179159e-01

# Prediction  
predict.elastic <- predict(elastic\_train\_model,test\_data)  
cm.elastic = confusionMatrix(predict.elastic,test\_data$Sales\_factor,positive = "TRUE")  
cm.elastic

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction FALSE TRUE  
## FALSE 42 8  
## TRUE 8 41  
##   
## Accuracy : 0.8384   
## 95% CI : (0.7509, 0.9047)  
## No Information Rate : 0.5051   
## P-Value [Acc > NIR] : 4.317e-12   
##   
## Kappa : 0.6767   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.8367   
## Specificity : 0.8400   
## Pos Pred Value : 0.8367   
## Neg Pred Value : 0.8400   
## Prevalence : 0.4949   
## Detection Rate : 0.4141   
## Detection Prevalence : 0.4949   
## Balanced Accuracy : 0.8384   
##   
## 'Positive' Class : TRUE   
##

# Accuracy of best tunining parameters of train model  
list(title = "Accuracy of best tunining parameters of train model",  
 LassoModel = besttuneResuts.lasso ,  
 RidgeModel = besttuneResuts.ridge,  
 ElasticModel = besttuneResuts.elastic)

## $title  
## [1] "Accuracy of best tunining parameters of train model"  
##   
## $LassoModel  
## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 1 0.1 0.753871 0.5075052 0.06956444 0.1388098  
##   
## $RidgeModel  
## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0 0.1 0.8504301 0.7007252 0.04797635 0.09588944  
##   
## $ElasticModel  
## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0.025 0.1 0.8570968 0.7140586 0.04471087 0.08937623

# Accuracy of predicted models on test data  
list(title = "Accuracy of predicted models on test data",  
 LassoModel = cm.lasso$overall['Accuracy'],  
 RidgeModel = cm.ridge$overall['Accuracy'],  
 ElasticModel = cm.elastic$overall['Accuracy'])

## $title  
## [1] "Accuracy of predicted models on test data"  
##   
## $LassoModel  
## Accuracy   
## 0.7373737   
##   
## $RidgeModel  
## Accuracy   
## 0.8585859   
##   
## $ElasticModel  
## Accuracy   
## 0.8383838

### Interpretation :

**Data :** Created a new column with factor of Sales > median. Used the Sales factor as response to see the sales high and low from median. Data pratitioned into 75-25 for trained and test data sets.

Trained all three models on train dataset using caret package. Using 10 fold cross validations on these data.

**Lasso Regression (binomial):** Set the alpha value to 1(lasso penalty) and lambda value between 0.1 to 20. The train method will test all the lambda values and takes the best lambda value in final model whose accuracy is highest. The optimal lambda selected thru CV for final model is 0.1 and its accuracy is 75.38%. Looking at the coefficents of the final model, only 4 predictors are selected and 7 predictors are set to zero. Lasso model tries to shrink the small coefficents to zero which might cause the lose of predictive ability. Confusion matrix of prediction on test data gives 73.74% of accuracy.

**Ridge Regression (binomial):** Set the alpha value to 0(ridge penalty) and lambda value between 0.1 to 20. The trained model selected lambda as 0.1 and accuracy of model is 85.04% and is considered in the final model. In final model, all coefficents are selected here but few of them are close to zero. Ridge model shrink the coefficents towards zero but never become zero which may helps to improve prditive accuracy. Confusion matrix shows good prediction on test data with good accuracy.

**Elastic Net Regression (binomial)** For elastic net model, alpha and lambda, both are tuning parameters. Alpha takes values between 0 and 1 (as this model is a generalization of the ridge and lasso models) and lambda between 0.1. to 20. THe model tests a range of possible alpha and lambda values, then selects the best values for lambda and alpha, resulting to a final model. The trained model selected alpha = 0.025 and lambda = 0.1 and accuracy is 85.70%. The tuning parameters are more towards the ridge model.

**Comparisons of all three models output:** All three models selected the optimal lambda = 0.1 as best tuning parameter for the final model. Elastic net model selected alpha = 0.01 which is close to the ridge model. If we compare the accuracy, the final model of elastic net gives 85.70% accuracy which is greater than lasso and ridge and close to ridge.

In elastic net and ridge final models, all coeff are selected and 4 predictors are shrinked toward zero value while in lasso, only 4 coeff are selected and others are set to zero.

Elastic net model is performing good with prediction accuracy(84) which is almost same as ridge model.

Overall, elastic net model looks great as compared to other though the tuning parameters are towards the ridge penalty. Model Accuracy is almost higher as compared to lasso and ridge models. As the ridge and elastic net model statistics looks very much similar, elastic net is good for this dataset because elastic net model will enables regularization via the ridge penalty with the feature selection characteristics of the lasso penalty.

## Problem 2 :

The wbca data in the faraway package studies the type of tumor (malignant or benign) found in individuals with breast cancer based on measurements of those tumors. Use Class as the response and the rest of the variables as predictors. Use lasso, ridge and elastic net (binomial logistic regression) to find relevant variables and discuss your final model. You should use 5 or 10 fold cross validation to choose the tuning parameters (use caret or cv.glment). Make sure you discuss any inisghts you may have about the model, and relationship between the three models you find.

library(faraway)

## Warning: package 'faraway' was built under R version 3.6.2

##   
## Attaching package: 'faraway'

## The following object is masked from 'package:lattice':  
##   
## melanoma

library(caret)  
library(glmnet)

## Warning: package 'glmnet' was built under R version 3.6.3

## Loading required package: Matrix

## Loaded glmnet 3.0-2

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

data("wbca")  
#head(wbca)  
#str(wbca)  
  
wbca$Class <- as.factor(wbca$Class)  
  
set.seed(0909)  
train\_in1 = createDataPartition(wbca$Class,times = 1,list = FALSE, p=.75)  
  
train\_data1 <- wbca[train\_in1,]  
test\_data1 <- wbca[-train\_in1,]  
  
#nrow(filter(test\_data1,test\_data1$Class == "1"))  
#nrow(filter(test\_data1,test\_data1$Class == "0"))  
#nrow(test\_data1)  
#nrow(train\_data1)  
  
fitControl= trainControl(method="cv",number=10)  
  
# Lasso  
set.seed(0909)  
wbca\_lasso <- train(Class~.,data=train\_data1,method="glmnet",trControl=fitControl, tuneGrid = expand.grid(alpha= 1, lambda = seq(0.1,15,0.1)))  
besttuneresults.lasso.wbca = displaybesttuneresults(wbca\_lasso)  
besttuneresults.lasso.wbca

## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 1 0.1 0.9550151 0.8979721 0.02786384 0.06476714

# Coeff  
coefficients(wbca\_lasso$finalModel, wbca\_lasso$bestTune$lambda)

## 10 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 3.07005200  
## Adhes .   
## BNucl -0.21657804  
## Chrom -0.04531704  
## Epith .   
## Mitos .   
## NNucl -0.03889506  
## Thick -0.11117752  
## UShap -0.09316731  
## USize -0.13688980

# Prediction on test data  
predict.lasso.wbca <- predict(wbca\_lasso, test\_data1, type="raw")  
cm.lasso.wbca = confusionMatrix(predict.lasso.wbca, test\_data1$Class, positive = "1")  
cm.lasso.wbca

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 51 2  
## 1 8 108  
##   
## Accuracy : 0.9408   
## 95% CI : (0.8939, 0.9713)  
## No Information Rate : 0.6509   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.8667   
##   
## Mcnemar's Test P-Value : 0.1138   
##   
## Sensitivity : 0.9818   
## Specificity : 0.8644   
## Pos Pred Value : 0.9310   
## Neg Pred Value : 0.9623   
## Prevalence : 0.6509   
## Detection Rate : 0.6391   
## Detection Prevalence : 0.6864   
## Balanced Accuracy : 0.9231   
##   
## 'Positive' Class : 1   
##

## Ridge  
set.seed(0909)  
wbca\_ridge <- train(Class~.,data=train\_data1,method="glmnet",family="binomial",tuneGrid = expand.grid(alpha = 0, lambda=seq(0.1,15,0.1)), trControl = fitControl)  
besttuneResuts.ridge.wbca = displaybesttuneresults(wbca\_ridge)  
besttuneResuts.ridge.wbca

## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0 0.1 0.9726244 0.9395125 0.01375869 0.03033809

# coeff  
coefficients(wbca\_ridge$finalModel,wbca\_ridge$bestTune$lambda)

## 10 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 4.8392242  
## Adhes -0.1044009  
## BNucl -0.1618831  
## Chrom -0.1619404  
## Epith -0.1449909  
## Mitos -0.1222372  
## NNucl -0.1132699  
## Thick -0.1695196  
## UShap -0.1375598  
## USize -0.1222973

# Prediction on test data  
predict.ridge.wbca <- predict(wbca\_ridge, test\_data1)  
cm.ridge.wbca = confusionMatrix(predict.ridge.wbca,test\_data1$Class, positive = "1")  
cm.ridge.wbca

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 54 3  
## 1 5 107  
##   
## Accuracy : 0.9527   
## 95% CI : (0.9089, 0.9793)  
## No Information Rate : 0.6509   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.895   
##   
## Mcnemar's Test P-Value : 0.7237   
##   
## Sensitivity : 0.9727   
## Specificity : 0.9153   
## Pos Pred Value : 0.9554   
## Neg Pred Value : 0.9474   
## Prevalence : 0.6509   
## Detection Rate : 0.6331   
## Detection Prevalence : 0.6627   
## Balanced Accuracy : 0.9440   
##   
## 'Positive' Class : 1   
##

#Elastic net  
set.seed(0901)  
tuning\_grid1 = expand.grid(alpha=seq(0.01,1,0.025), lambda=seq(0.1,20,0.25))  
wbca\_elastic <- train(Class~.,data=train\_data1,family="binomial",method="glmnet",trControl=fitControl,tuneGrid=tuning\_grid1)  
besttuneResuts.elastic.wbca = displaybesttuneresults(wbca\_elastic)  
besttuneResuts.elastic.wbca

## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0.01 0.1 0.974736 0.9436993 0.02251384 0.05109104

# coeff  
coefficients(wbca\_elastic$finalModel,wbca\_elastic$bestTune$lambda)

## 10 x 1 sparse Matrix of class "dgCMatrix"  
## 1  
## (Intercept) 4.8193572  
## Adhes -0.1034161  
## BNucl -0.1619250  
## Chrom -0.1611561  
## Epith -0.1437646  
## Mitos -0.1195227  
## NNucl -0.1125823  
## Thick -0.1691547  
## UShap -0.1370879  
## USize -0.1217998

# Prediction on test data  
predict.elastic.wbca <- predict(wbca\_elastic,test\_data1)  
cm.elastic.wbca = confusionMatrix(predict.elastic.wbca,test\_data1$Class,positive = "1")  
cm.elastic.wbca

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 54 3  
## 1 5 107  
##   
## Accuracy : 0.9527   
## 95% CI : (0.9089, 0.9793)  
## No Information Rate : 0.6509   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.895   
##   
## Mcnemar's Test P-Value : 0.7237   
##   
## Sensitivity : 0.9727   
## Specificity : 0.9153   
## Pos Pred Value : 0.9554   
## Neg Pred Value : 0.9474   
## Prevalence : 0.6509   
## Detection Rate : 0.6331   
## Detection Prevalence : 0.6627   
## Balanced Accuracy : 0.9440   
##   
## 'Positive' Class : 1   
##

# best tune results of final model   
list(title = "Best tune results of final model",  
 lasso\_model = besttuneresults.lasso.wbca,  
 ridge\_model = besttuneResuts.ridge.wbca,  
 elastic\_model = besttuneResuts.elastic.wbca)

## $title  
## [1] "Best tune results of final model"  
##   
## $lasso\_model  
## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 1 0.1 0.9550151 0.8979721 0.02786384 0.06476714  
##   
## $ridge\_model  
## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0 0.1 0.9726244 0.9395125 0.01375869 0.03033809  
##   
## $elastic\_model  
## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0.01 0.1 0.974736 0.9436993 0.02251384 0.05109104

# Accruacy of prediction on test data  
list(title = "Accuracy of prediction on test data",  
 lasso\_pred = cm.lasso.wbca$overall['Accuracy'],  
 ridge\_pred = cm.ridge.wbca$overall['Accuracy'],  
 elastic\_pred = cm.elastic.wbca$overall['Accuracy'])

## $title  
## [1] "Accuracy of prediction on test data"  
##   
## $lasso\_pred  
## Accuracy   
## 0.9408284   
##   
## $ridge\_pred  
## Accuracy   
## 0.9526627   
##   
## $elastic\_pred  
## Accuracy   
## 0.9526627

# Prediction Eval  
list(title = "Prediction Eval",  
 lasso\_model = cm.lasso.wbca$table %>% prop.table() %>% round(3),  
 ridge\_model = cm.ridge.wbca$table %>% prop.table() %>% round(3),  
 elastic\_model = cm.elastic.wbca$table %>% prop.table() %>% round(3))

## $title  
## [1] "Prediction Eval"  
##   
## $lasso\_model  
## Reference  
## Prediction 0 1  
## 0 0.302 0.012  
## 1 0.047 0.639  
##   
## $ridge\_model  
## Reference  
## Prediction 0 1  
## 0 0.320 0.018  
## 1 0.030 0.633  
##   
## $elastic\_model  
## Reference  
## Prediction 0 1  
## 0 0.320 0.018  
## 1 0.030 0.633

### Interpretation :

Trained all three models on train dataset using caret package using 10 fold cross validations.

Looking at the output of lasso model (penalty alpha = 1), the best results uses lambda = 0.1 with accuracy of 95.5% for the final model. Train method of caret finds the value of lambda that gives the simplest model but also lies within one standard error of the optimal value of lambda. Model selected 6 predictors for the final model and setting other 3 variables to zero. Lasso tries to shrink the non-significant coeff to zero to reduce the complexity of the model which may decrease the accuracy. Predicting the trained model on test data gives 94% of accuracy. Looking at the confusion matrix and comparing this with actual test data, we see that true negatives and true positives numbers are very close to the actual test data.

As per the output of ridge model (penalty alpha = 0), the best results uses lambda = 0.1 with accuracy of 97.26%. Model tried to shrink less significant coefficents towards zero but it will not set to zero which may increase the accuracy as compared to lasso model. Looking at the confusion matrix, prediction accuracy of the model is 95%.

In elastic net regression model, alpha and beta are tuning parameters. Alpha takes values between 0 and 1 (as this model is a generalization of the ridge and lasso models) and lambda values varies from o to anyvalue (here 15). The model tests a range of possible alpha and lambda values, then selects the best values for lambda and alpha, resulting to a final model. Final values used for model are alpha = 0.01 and lambda = 0.1 which fits a model close to ridge. Prediction accuracy on test data of the model is 95.26% which is again similar to ridge.

**Comparisons :** Looking at the tuning parameters of all three models, optimal lambda value selected is 0.1 and alpha =0.01 for elastic net. Accuracy of ridge and elastic net is same and highter than lasso model. Lasso algorithm tried to shrink the 3 coeffs and set it to zero to reduce the complexity which caused the accuracy to reduce. Elastic net and Ridge coeff at pushed towards to zero vaulues which are not significant but needs to be considered at some point. Elastic net regression combines L1-norm and L2-norm introducing a mixing parameter aplha in conjunction with lambda. Prediction accuracy of elastic net and ridge are same which is 95.26% and higher than lasso model. In elasticnet and ridge models, 32% of the observations are predicted as true negatives (malignant) and about 63.3% are true positives(benign). 3% cases are not malignant but they actually are malignant and around 2% are benign but actually are not benign.