COMP-SCI 5565: Introduction to Statistical Learning Final Project

Possible points: 80 **Due: December 12, 2018**

Name: Maseerah Sarosh Muradabadi **ID No:** 16277877

Email Address: mmfg9@mail.umkc.edu

Please generate a report consisting of the following: (80 points)

1. Explain tree based method for classification. Please mention the major disadvantages of the tree-based method over Support Vector Machine classifiers? (20 points)

➢ Classification Trees:

- Classification decision trees, unlike regression trees predict a *qualitative* response rather than quantitative.
- For a classification tree we predict that each observation belongs to the *most commonly occurring class* of training observations in the region to which it belongs.
- For interpreting the results of a classification tree, we are interested in the *class corresponding to a particular terminal node region* and also in the *class proportions* among the training observations that fall into that region.
- We use binary splitting for growing a classification tree and *classification error rate* is used as a criterion for making the binary splits and is defined as the fraction of training observations in that region that do not belong to the most common class.

$$E = 1 - \max_k(\hat{p}_{mk}).$$

Proportion of training samples in the mth region that are from kth class

- However, it turns out that classification error is not sufficiently sensitive for tree growing, and in practice two other measures are preferable

i. Gini Index: a measure of total variance across the K classes

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

From the above equation we can observe that Gini Index takes a small value if all the $\hat{p}_{mk}$'s are close to zero or one. For this reason Gini index is referred to as node *purity*- a small value indicated that a node contains predominantly observations from a single class

ii. Entropy: An alternative to Gini index is entropy and is given by:

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}.$$

- Since 0<= $\hat{p}_{mk}$'s <=1, it follows that 0<= $-\hat{p}_{mk} \log \hat{p}_{mk}.$

- One can show that entropy can take a very small value near zero if $\hat{p}_{mk}$'s are all near zero or one. If mth node is pure both Gini index and Entropy

  Take a small value.

<u>Advantages of decision trees over Support vector machine classifiers:</u>

- ✓ Trees are easy to explain to people and have better interpretability than support vector machine classifiers
- ✓ For non-linear data, decision trees are able to capture true decision boundaries over other approaches where as support vectors machine classifiers are only for linear data that can be perfectly separable by the maximal margin M. If the data is non linear and we are not able to separate it perfectly, then we can apply the support vector machines approach that uses soft margin to separate data allowing some slack variables to violate the true classification rule.

<u>Disadvantages of decision trees over Support vector machine classifiers:</u>

- ✓ Decision trees do not have the same level of predictive accuracy as support vector machine classifier
- ✓ Decision trees are non-robust; small change in data can cause a large change in the final estimate. Whereas, support vector machines classifier are highly robust. And small changes in data samples do not affect the classifier, as only the representative samples(or support vectors, data samples lying near the margin) are taken into consideration for classification and not the entire data.

[Reference: ISL textbook]

2. Explain the various methods for mitigating the disadvantages of the tree. (20 points)

➢ Since trees have lower prediction accuracy by aggregating many decision trees, using methods like bagging, random forests, and boosting, the predictive performance of trees can be substantially improved.

1) Bagging/ Bootstrap Aggregation:

- Bootstrap aggregation, or bagging, is a general-purpose procedure for reducing the variance of a statistical learning method; we introduce it here because it is particularly useful and frequently used in the context of decision trees.
- Averaging a set of observations reduces variance.
- We can bootstrap, by taking repeated samples from the (single) training data set. In this approach we generate B different bootstrapped training data sets. We then train our method on the bth bootstrapped training set in order to get ˆ f∗b(x), and finally average all the predictions, to obtain

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x).$$

- This is called bagging.

- To apply bagging to regression trees, we simply construct B regression trees using B bootstrapped training sets, and average the resulting predictions.

- To apply bagging to classification trees, for a given test observation, we can record the class predicted by each of the B trees, and take a majority vote: the overall prediction is the most commonly occurring class among the B predictions.

2) Random Forest:

- Random forests provide an improvement over bagged trees by way of a small tweak that decorrelates the trees.

- As in bagging, we build a number of decision trees on bootstrapped training samples. But when building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors.

- The split is allowed to use only one of those m predictors. A fresh sample of m predictors is taken at each split, and typically we choose m ≈√p—that is, the number of predictors considered at each split is approximately equal to the square root of the total number of predictors.

- In other words, in building a random forest, at each split in the tree, the algorithm is not even allowed to consider a majority of the available predictors.

-The main difference between bagging and random forests is the choice of predictor subset size m. For instance, if a random forest is built using m = p, then this amounts simply to bagging.

3) Boosting:

- The trees are grown sequentially: each tree is grown using information from previously grown trees. Boosting does not involve bootstrap sampling; instead each tree is fit on a modified version of the original data set.

- Unlike fitting a single large decision tree to the data, which amounts to fitting the data hard and potentially overfitting, the boosting approach instead learns slowly.

- Given the current model, we fit a decision tree to the residuals from the model. That is, we fit a tree using the current residuals, rather than the outcome Y , as the response. We then add this new decision tree into the fitted function in order to update the residuals. Each of these trees can be rather small, with just a few terminal nodes, determined by the parameter d in the algorithm.

- Fitting small trees to the residuals, we slowly improve ˆ f in areas where it does not perform well. The shrinkage parameter λ slows the process down even further, allowing more and different shaped trees to attack the residuals. In general, statistical learning approaches that learn slowly tend to perform well. Note that in boosting, unlike in bagging, the construction of each tree depends strongly on the trees that have already been grown.

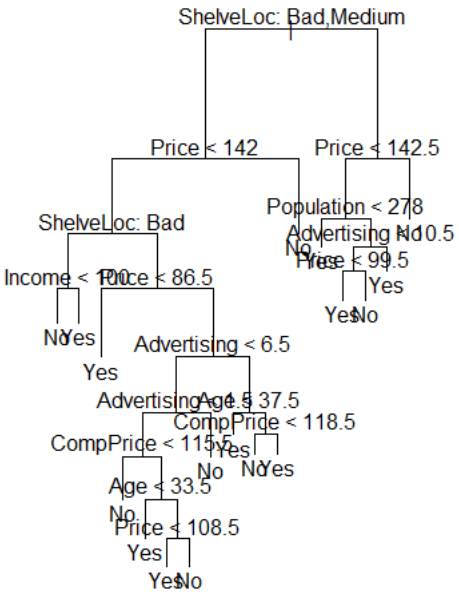
- Boosting has three tuning parameters:

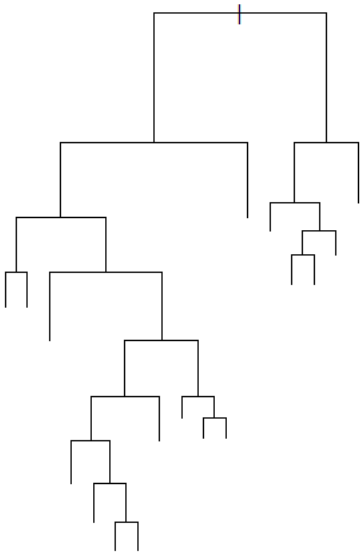
i. The number of trees B. Unlike bagging and random forests, boosting can overfit if B is too large, although this overfitting tends to occur slowly if at all. We use *cross-validation* to select B.
ii. The *shrinkage parameter λ,* a small positive number. This controls the rate at which boosting learns
iii. The *number d of splits in each tree*, which controls the complexity of the boosted ensemble. More generally d is the interaction depth, and controls the interaction order of the boosted model, since d splits can involve at most d variables

[Reference: ISL textbook]

3. Refer to Lab Exercise: Decision Tree (Chapter 8) on Carseats data. Select the best classification tree (refer to homework #3) of appropriate length. Plot the selected tree, confusion matrix and the accuracy of the classifier. (20 points)

- From the classification tree we find that the best tree obtained after applying *cost complexity pruning*, has the length of *node 9 with accuracy 77%* whereas the original accuracy without pruning was 71.5%.
- However, by increasing the value of node length using the 'best' command we can have a larger pruned tree which has the *length 15* but has lower *accuracy that is 74%*
- In my view, the accuracy of the node with nine nodes would be the best one to go for as the *deviance obtained for the node 9 from the code is only 50* which is the lowest when compared to other nodes.
- Following is the plot for the tree with length 9 and accuracy 77%





- The confusion matrix for the above classification tree is:

n= 200	Predicted No	Predicted Yes
Actual No	94 True Negative	22 False Positive
Actual Yes	22 False Negative	60 True Positive

- The accuracy can be calculated using the formula:

Accuracy = True Negative + True Positive = (94+60) / 200 = **0.77**

Total number of samples

[Reference: ISL textbook, Class Notes & HW #3]

4. Compare the performance of the selected classification decision tree (point c) with fine-tuned SVM classifier and SVM machine in terms of confusion matrix and accuracy on the Carseats data. Please also explain and plot the intermediate outputs related to summary of learned SVM model(s) (summary(svmfit)). Explain the difference in the performance of decision tree over SVM classifier/ machine. (20 points)

➢ The following points cover the answer to the above question:

i. **Performance of the Classification tree Vs Support vector Classifier & Support Vector Machine in terms of confusion matrix and accuracy on carseats data:**

- **Confusion Matrix for Classification Decision Tree:**

n= 200	Predicted No	Predicted Yes
Actual No	94 True Negative	22 False Positive
Actual Yes	22 False Negative	60 True Positive

➢ The accuracy can be calculated using the formula:

Accuracy = $\frac{\text{True Negative + True Positive}}{\text{Total number of samples}}$ = (94+60) / 200 = **0.77**

✓ Accuracy of Classification trees is 77% on Carseats data

- **Confusion Matrix for Support Vector Classifier:**

n= 20	Predicted No(-1)	Predicted Yes(1)
Actual No(-1)	112 True Negative	3 False Positive
Actual Yes(1)	3 False Negative	82 True Positive

Accuracy = $\frac{\text{True Negative + True Positive}}{\text{Total number of samples}}$ = (112+82) / 200 = **.97**

✓ The accuracy of support vector classifier on carseat data is 97%

- **Confusion Matrix for Support Vector machine:**

n= 100	Predicted No(1)	Predicted Yes(2)
Actual No(1)	104 True Negative	3 False Positive
Actual Yes(2)	11 False Negative	82 True Positive

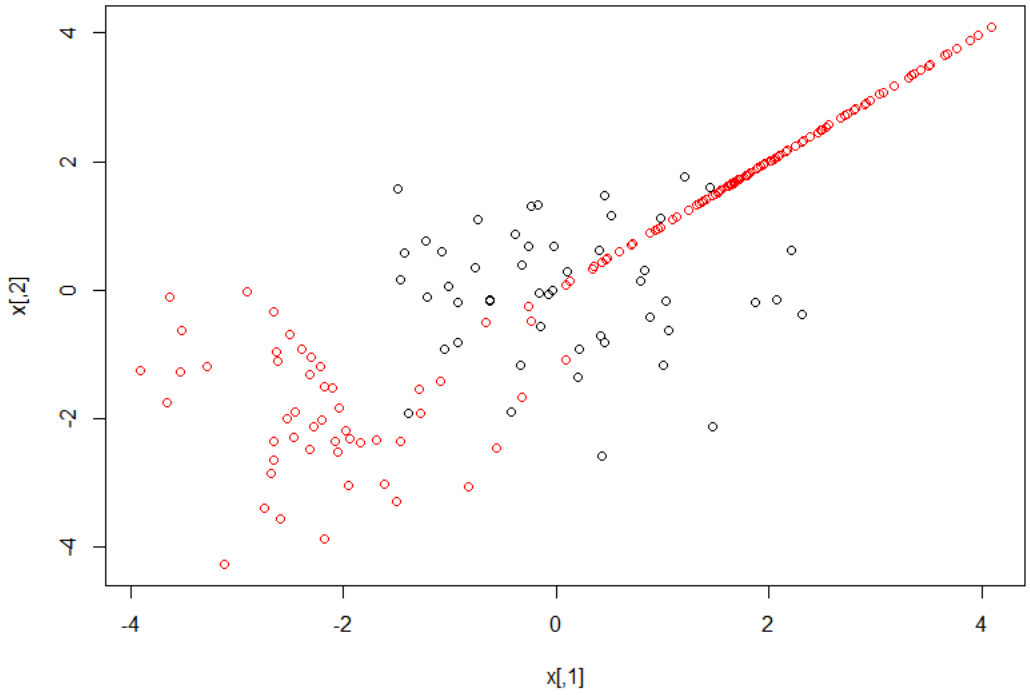
Accuracy = True Negative + True Positive = (104+82) / 200 = **0.93**

Total number of samples

✓ The accuracy of support vector classifier on carseat data is 93%

ii. Ouputs from the summary of learned model for Support Vector Classifier and Support Vector Machine:

➢ Support vector classifier for carseats data set:



```
> High=ifelse(Sales <=8,"No","Yes")
> Carseats$y = as.factor(High)
> train_dataset = sample(dim(Carseats)[1],200)
> dat.train = Carseats[train_dataset,]
> dat.test = Carseats[-train_dataset,]
> svmfit=svm(y~.,data=dat.train,kernel="linear",cost=10,scale = FALSE)
>
> svmfit$index
 [1]   1  13  42  58  85  74 148 156 168 174 187 200
> summary(svmfit)

Call:
svm(formula = y ~ ., data = dat.train, kernel = "linear", cost = 10,
    scale = FALSE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  10
      gamma:  0.07692308

Number of Support Vectors:  12

 ( 5 7 )


Number of Classes:  2

Levels:
 No Yes
```

-We can see that since our data is linear in nature we assign the type to the kernel function

-Also, the choice of “c” cost argument is chosen as 10, however a smaller c value like c=0.1 would result in wider margin and many support vectors violate the margin and conversely a higher value of “c” would result in a narrow margin with less violation of support vectrs

-The summary of the support vector classifier is shown above with a cost of 10

- Support vector machine for carseats data set:

```
> summary (tune.out)

Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost
    1

- best performance: 0.09

- Detailed performance results:
  cost error dispersion
1    1 0.090 0.09067647
2    5 0.110 0.09067647
3   10 0.120 0.10055402
4   50 0.125 0.09501462
5  100 0.125 0.09501462

> bestmod=tune.out$best.model
> summary(bestmod)

Call:
best.tune(method = svm, train.x = y ~ ., data = dat.train, ranges = list(cost = c(1,
    5, 10, 50, 100)), kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.07692308

Number of Support Vectors:  93

 ( 43 50 )
```

- For SVM we use kernel= radial or polynomial, and calculate gamma as well

- We find the model with lowest error and we can see that the best performance is having the error 0.09 with a cost 1. This is achieved after 10 fold cross validation.

iii. Difference in performance of decision trees over SVM classifier and machine:

Classification Trees	**Support Vector Classifier**	**Support Vector Machine**
Linear data	Linear data	Non-linear data
Optimization technique- cost complexity pruning	Cross Validation	Cross validation
Lowest accuracy-77%	Highest accuracy-97%	Second highest accuracy-93%
Used prune() to select the best node with low deviance	Used tune() to select best model with least cost argument "c"	Used tune() to select best model with least "gamma" and "c"
Higher Interpretability	low interpretability compared to trees but distinction of data can be found	low interpretability compared to trees but distinction of data can be found

[Reference: ISL textbook, Class Notes & R code Directory"F1"]