STAT315B Homework 2

Problem 1:

Differentiating (10.11) w.r.t. β and setting the derivative to 0 gives

$$\left(e^{\beta_m} + e^{-\beta_m}\right) \sum_{i=1}^{N} \omega_i^{(m)} I(y_i \neq G(x_i)) - e^{-\beta_m} \sum_{i=1}^{N} \omega_i^{(m)} = 0$$

Solving for e^{β_m}, we obtain

$$e^{2\beta_m} = \frac{\sum_{i=1}^{N} \omega_i^{(m)} - \sum_{i=1}^{N} \omega_i^{(m)} I(y_i \neq G(x_i))}{\sum_{i=1}^{N} \omega_i^{(m)} I(y_i \neq G(x_i))}$$

If we divide both the numerator and denominator by $\sum_{i=1}^{N} \omega_i^{(m)}$, then it is equivalent to

$$e^{2\beta_m} = \frac{1 - err_m}{err_m}$$

which yields (10.12).

Problem 2:

Observe that $E_{Y|x}\left(e^{-Yf(x)}\right) = P(Y = 1|x)\, e^{-f(x)} + P(Y = -1|x)\, e^{f(x)}$. To find the minimizer, we differentiate this expression w.r.t. $f(x)$ and set the derivative equal to 0.

$$P(Y = -1|x)\, e^{f(x)} - P(Y = 1|x)\, e^{-f(x)} = 0$$

This gives exactly $f(x) = \frac{1}{2} \log \frac{P(Y = 1|x)}{P(Y = -1|x)}$ as in (10.16).

Problem 3:

For additive function, observe that

$$E_{X_C}\left[h_1(X_S) + h_2(X_C)\right] = h_1(X_S) + E_{X_C}\left[h_2(X_C)\right]$$

Since $E_{X_C}\left[h_2(X_C)\right]$ is constant, this is $h_1(X_S)$ up to an addition of some constant. Similarly,

$$E_{X_C}\left[h_1(X_S) \cdot h_2(X_C)\right] = h_1(X_S) \cdot E_{X_C}\left[h_2(X_C)\right]$$

This again is $h_1(X_S)$ up to a multiple of some constant. However, if using conditional expectation,

$$E\left[h_1(X_S) + h_2(X_C)|X_S\right] = h_1(X_S) + E\left[h_2(X_C)|X_S\right]$$

and

$$E\left[h_1(X_S) \cdot h_2(X_C)|X_S\right] = h_1(X_S) \cdot E\left[h_2(X_C)|X_S\right]$$

Since $E\left[h_2(X_C)|X_S\right]$ is a function of X_S and generally not constant, above two expressions are not $h_1(X_S)$ up to an addition/multiple of some constant unless X_C and X_S are independent of each other.

Problem 4:

Data set `california.data` is read into R and a `MART` model is then fit using the `R-MART` program. Then I update the model using `moremart` function to achieve better prediction error rate. After 1000 iterations, the test error still seems decreasing. But the it is pretty flat already (Figure 1). So I stop the updating after 1000 iterations. The test error rate I obtain is 0.3051.

```
> california <- read.table("california.data", sep=",")
> names(california) <- c("MedianValue", "MedianIncome", "HouseMedianAge",
                         "AveNoRoom","AveNoBdrm","Population",
                         "AveOccupancy","Latitude","Longitude")
> y <- california[, 1]; x <- as.matrix(california[, 2:9]); lx <- rep(1, 8)
> mart(x, y, lx)
   MART execution finished.
   iters       best    test abs(error)
    200         200      0.3379
> moremart()
  MART execution finished.
   iters       best    test abs(error)
   1000        1000      0.3051
```

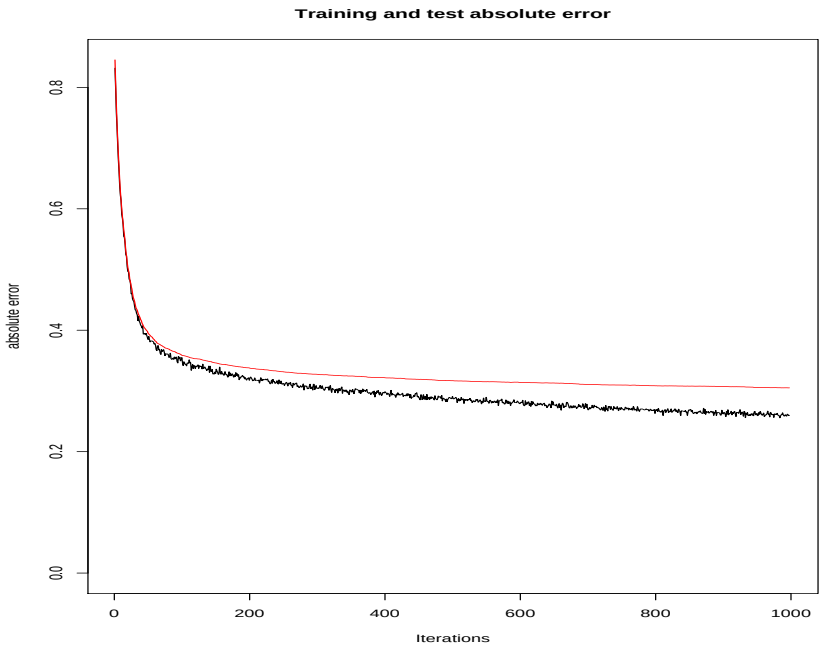

Figure 1: California Housing Data. Prediction error rate against iterations from the MART model. Red: test error curve; Black: training error rate.

I also plot the importance of each variable. From this plot (Figure 2), we can see that `MedianIncome` is the most important variable, followed by `Logitude`, `AveOccupancy`, `Latitude`. Then we study the dependence of the response on these most important variables using `singleplot` and `pairplot`. It is observed in the singleplot (Figure 3) that the Median House Value `MedianValue` is negatively correlated

with `Longitude`, `AveOccupancy`, `Latitude` and positively correlated with `MedianIncome`. But the curve does flatten out when income is large. The pairplot of `longitude` and `latitude` (Figure 4) reveals the strong geographic interaction between these two variables.

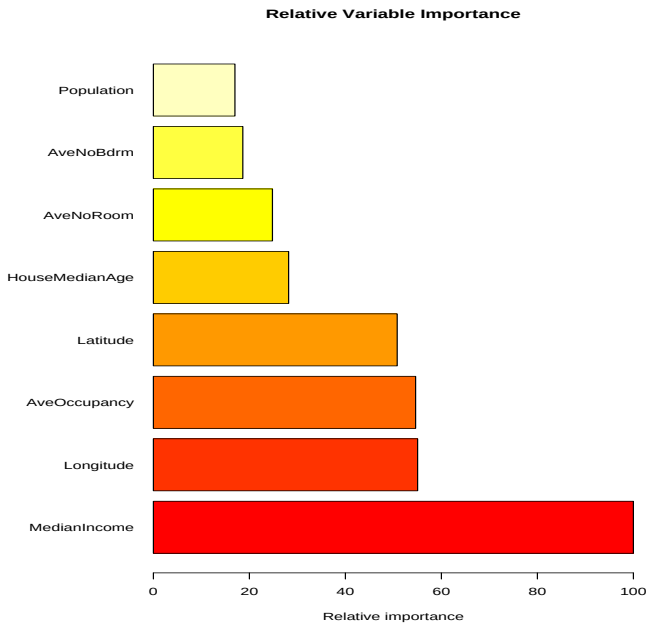

Figure 2: California Housing Data. Variables importance plot.

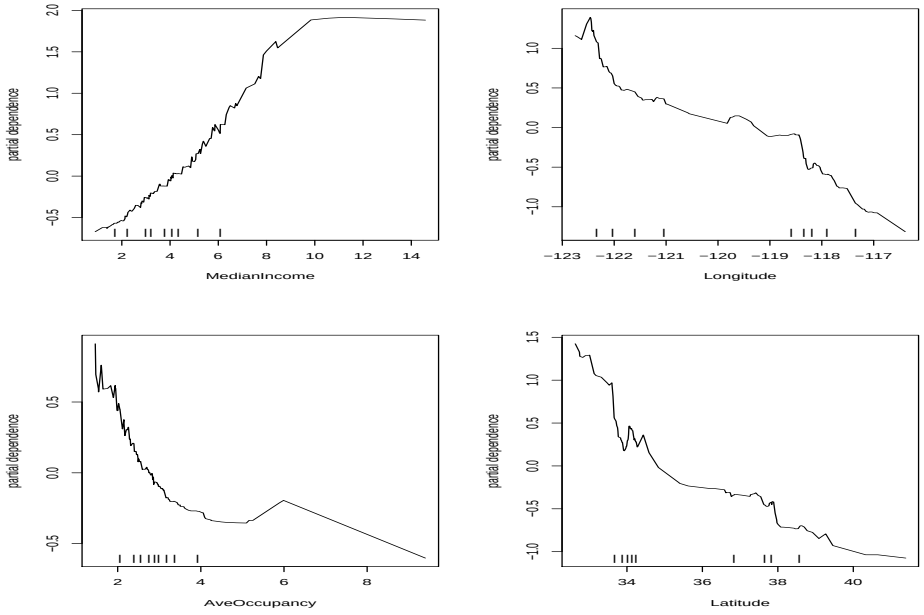

Figure 3: California Housing Data. Singleplot of the 4 most important variables.

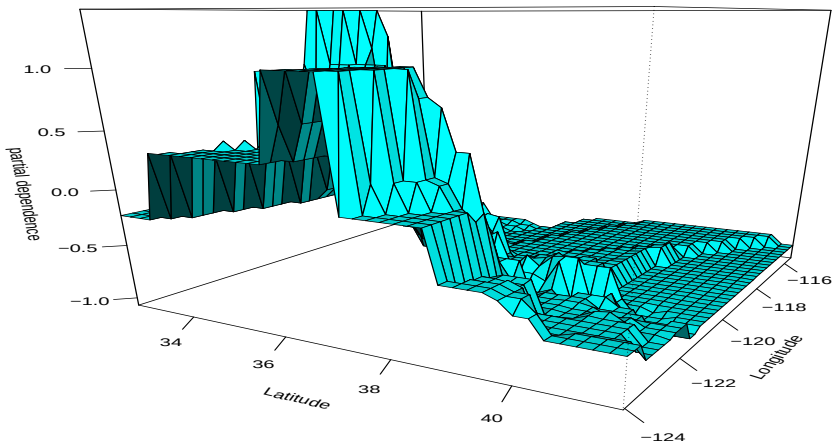

Figure 4: California Housing Data. Pairplot of `Longitude` and `Latitude`.

Problem 5:

To assess the performance of MART and compare with CART, I randomly split the original data set into training part and test part. The test set consists of 2000 observations (about 1/5 of the original one). I first build the MART and CART model based the training sample and then make prediction on the test set. The criterion I use to evaluate the prediction performance is the mean absolute error.

```
> income <- read.table("income.data", sep=",")
> x <- as.matrix(income[, 2:14]); y <- income[, 1]; lx <- c(rep(2,8), 1, 1, 2, 2, 2)
> test.set <- sample(c(1:8893), 2000)
> mart(x[-test.set,], y[-test.set], lx)
 MART execution finished.
   iters       best    test abs(error)
    200         200       1.480
 Warning message:
 x contains NA's - xmiss substituted. in: mart(x[-test.set, ], y[-test.set], lx)
> moremart()
 MART execution finished.
   iters       best    test abs(error)
    400         379       1.466
> yhat <- martpred(x[test.set, ])
Warning message:
 x contains NA's - xmiss substituted. in: martpred(x[test.set, ])
> mean(abs(income[test.set, 1]-yhat))
[1] 1.388786
```

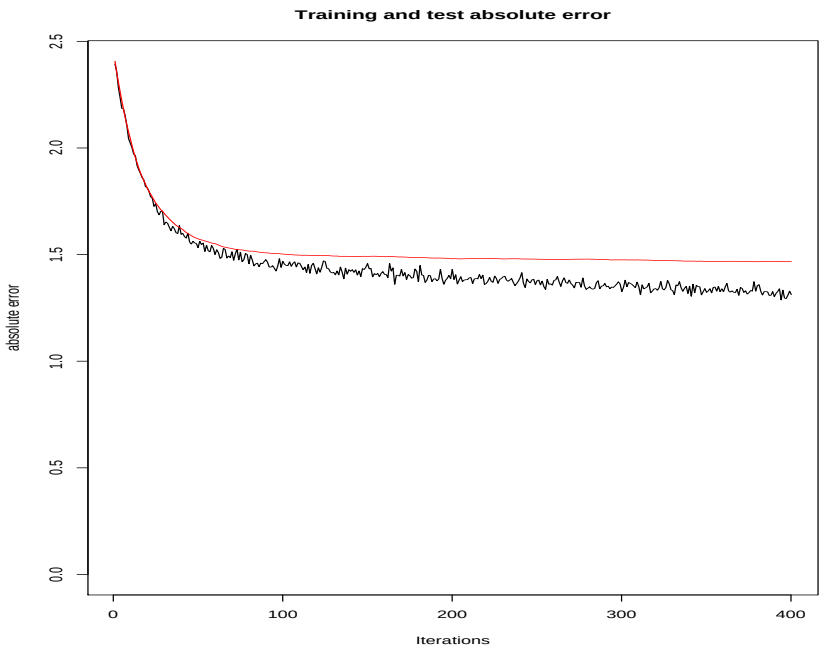

Figure 5: Income Data. Prediction error rate against iterations from the MART model. Red: test error curve; Black: training error rate.

As can be seen (Figure 5), the MART procedure flattens out very rapidly. After 400 runs, the associated average absolute error is 1.466. The average absolute error for predicting the test set using the trained MART model is 1.39. For the CART model, I treat `sex`, `marital status`, `occupation`, `dual income`, `householder status`, `home type`, `ethnic classification`, `language` as the class variables. The optimal tuning parameter based on the training sample is $CP \approx 0.002$ (Figure 6). The average absolute error of the test set based on the tree pruned using the optimal parameter is 1.60. It is seen that MART has a much better performance.

```
> short.name <- as.vector(apply(t(var.name), 1, substring, 1, 3))
> income <- read.table("income.data", sep=",", header=F, as.is=T,
+                      row.names=NULL, col.names=short.name)
> income$sex <- as.factor(income$sex)
> income$mar <- as.factor(income$mar)
> income$occ <- as.factor(income$occ)
> income$dua <- as.factor(income$dua)
> income$hou <- as.factor(income$hou)
> income$hom <- as.factor(income$hom)
> income$eth <- as.factor(income$eth)
> income$lan <- as.factor(income$lan)

> pruned.tree <- prune(income.tree, 0.0019)
> yhat <- predict(pruned.tree, income[test.set, ])
> mean(abs(income[test.set, 1]-yhat))
[1] 1.603234
```

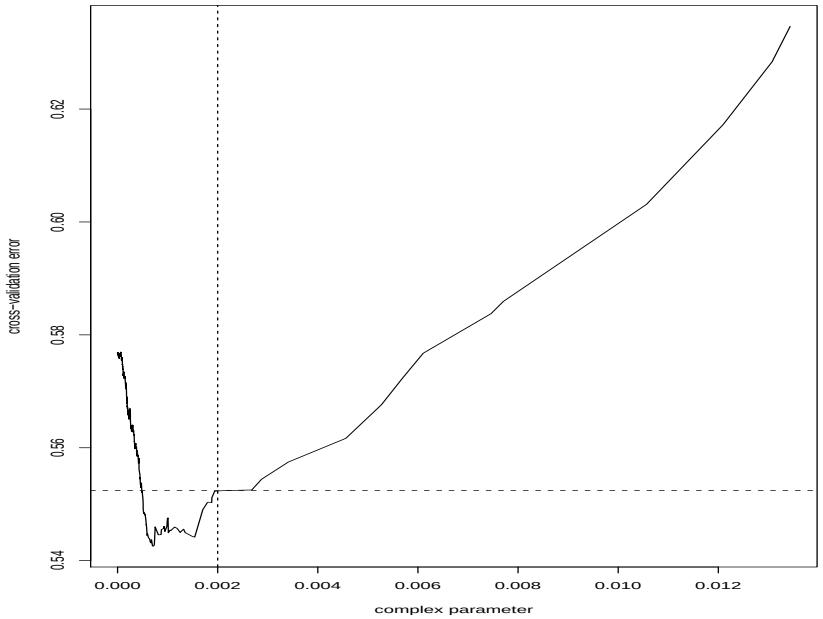

Figure 6: Income Data. Cross-validation error curve from CART model. The complexity parameter is around $CP = 0.002$.

Problem 6:

(a)

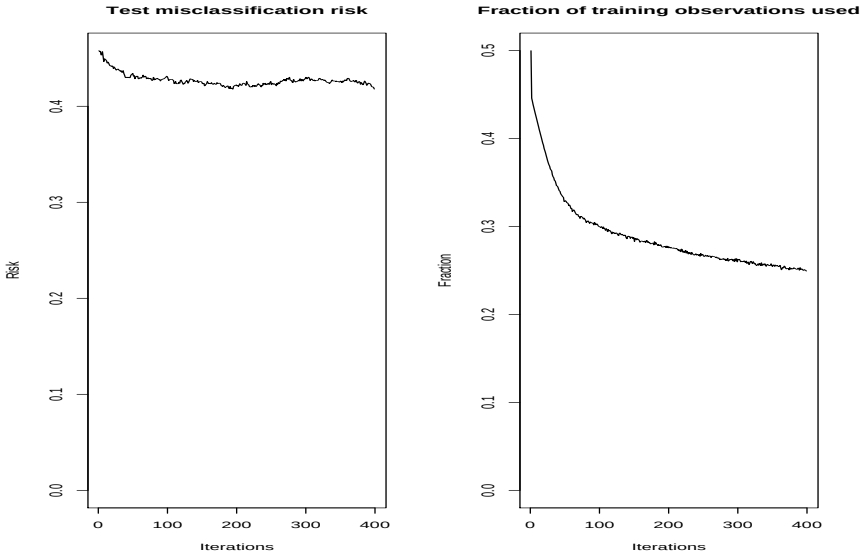

Figure 7: Occupation Data. Left: misclassification error risk versus iteration. Right: the fraction of training observations used.

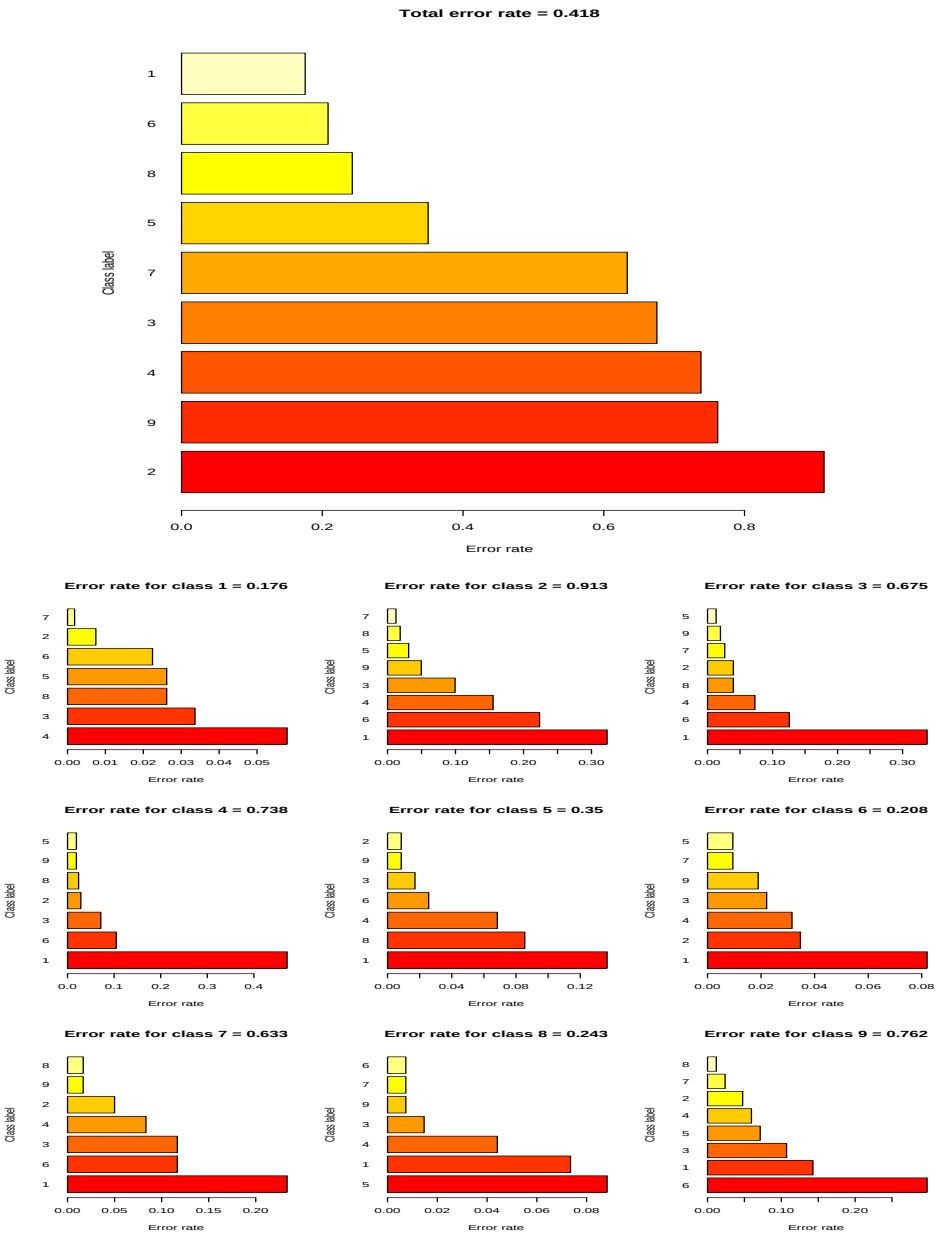

Figure 8: Occupation Data. Upper: misclassification error risk for each class label. Bottom: Detailed misclassification distribution for each class label.

In the `occupation` data set, I treat `home type`, `sex`, `marital status`, `dual income`, `householder status`, `ethnic classification`, `language` as the class variables. The MART procedure gives test misclassification error risk 0.4178 (Figure 7). I also plot the misclassification error for each class label (Figure 8). From the plot, we can see that class 2, 9, 4, 3, 7 are badly classified. The bottom of Figure 8 shows detailed distribution of misclassification in each class.

```
> x <- as.matrix(occupation[, 2:14])
> y <- occupation[, 1]
> lx <- c(2, 2, 2, 1, 1, 1, 1, 2, 1, 1, 2, 2, 2)
```

```
> mart(x, y, lx, martmode='class')
 MART execution finished.
   iters       best   test misclass risk
    200         195      0.4178
0 Warning message:
 x contains NA's - xmiss substituted. in: mart(x, y, lx, martmode = "class")
> classerrors()
```

(b)

The importance of each variable is identified in Figure 9. The upper plot shows the importance of each variable for all the classes while the bottom one shows the importance of each variable in each class.

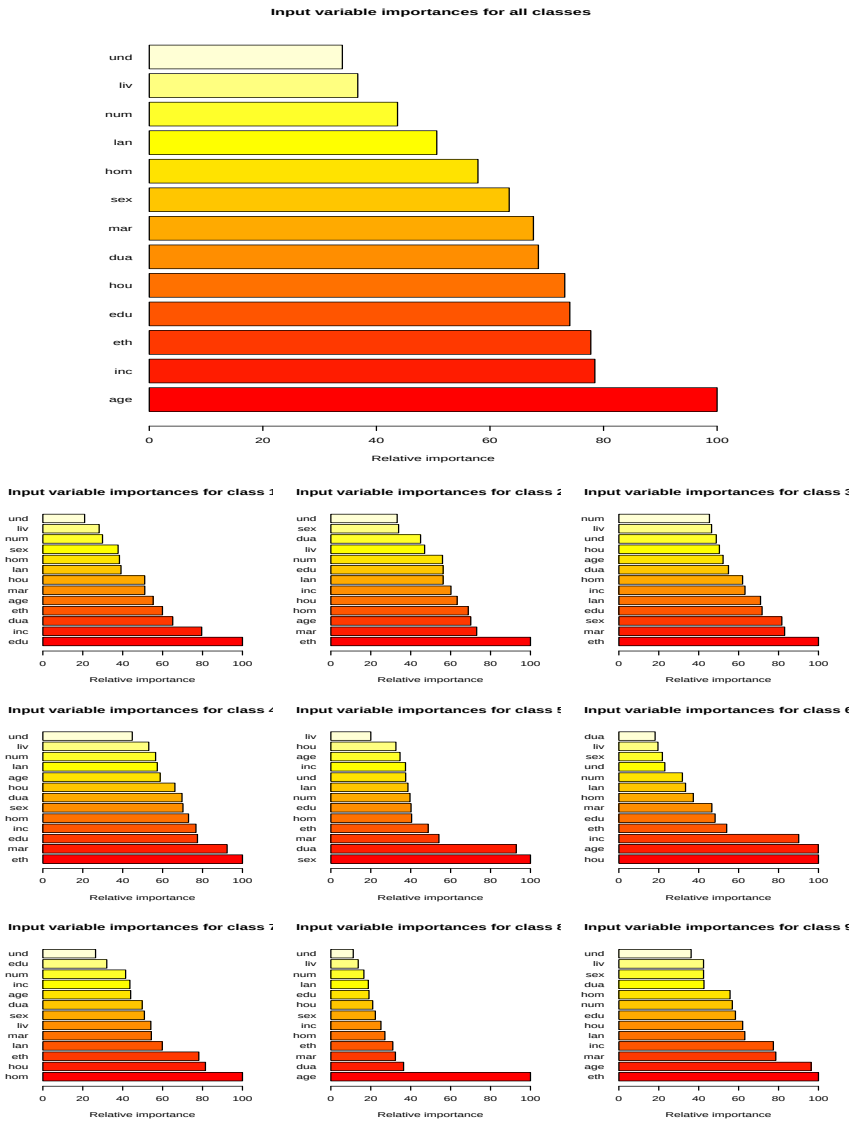

Figure 9: Occupation Data. Upper: the importance of each variable for all classes. Bottom: the importance of each variable in each class.

(c)

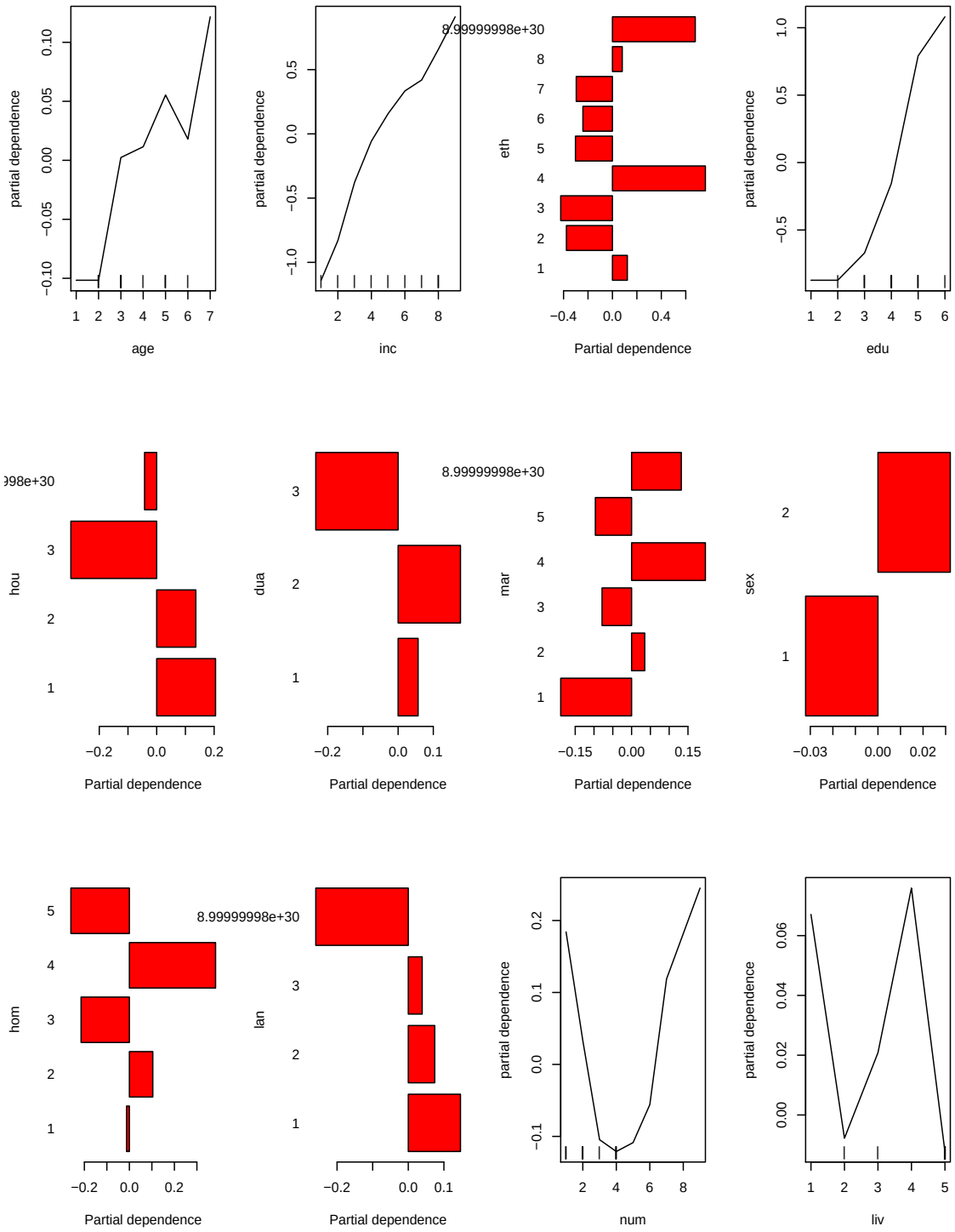

Figure 10: Occupation Data. Singleplots of variables in class 1.

In this part, I will explore the dependence of the response on each variable. Arbitrarily, I choose class 1 (Professional/Managerial) as demonstration. The partial dependence on each variable is shown in Figure 10. Note: I omit variable `und` (number of persons under 18) since it is the least important one as indicated in the first plot of the bottom part in Figure 9. The 4 singleplots in the first row of Figure 10 are the most important variables. Particularly, the dependence on `age`, `edu` (education) and `inc` (income) is positively correlated with these variables. Now, we plot the interaction between these variables. Figure 11 consists of the pairplots between numeric variables `age`, `edu` (education) and `inc` (income). These three plots show very strong additivity between these variables.

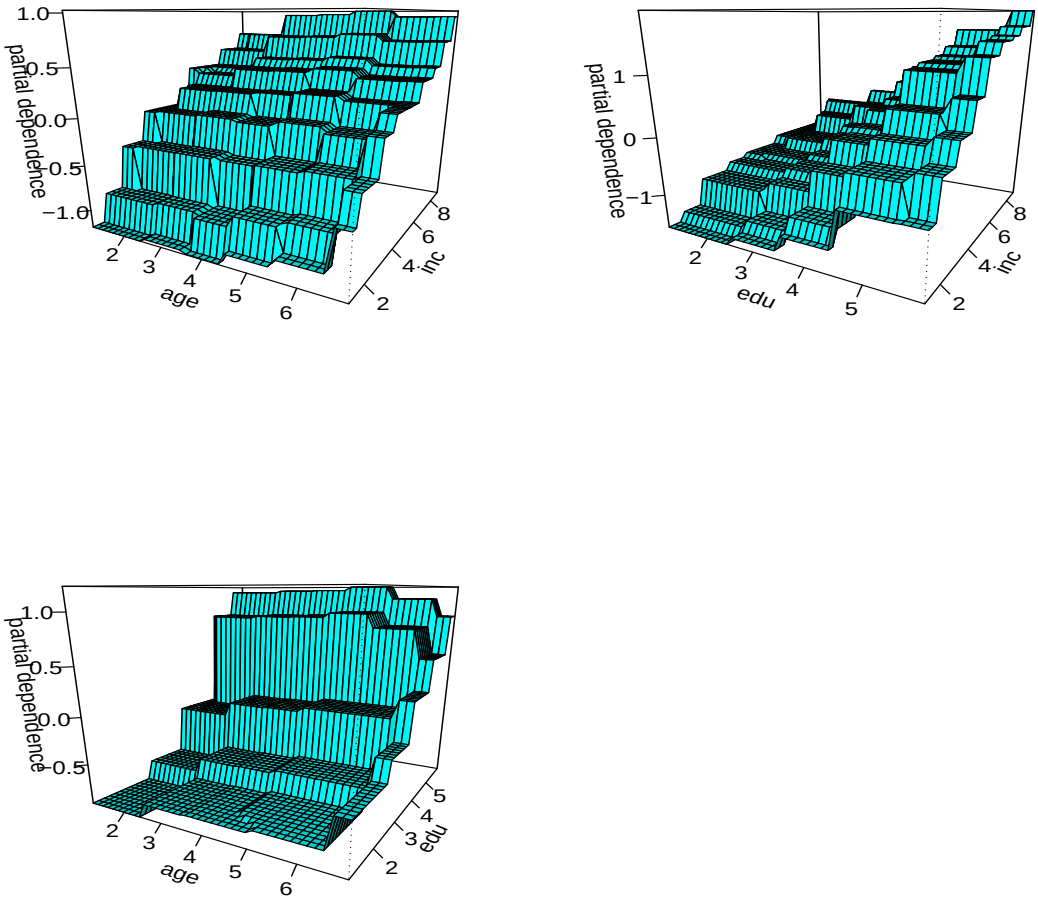

Figure 11: Occupation Data. Pairplots of variables `age`, `edu` and `inc` in class 1.

(d)

The bottom part of Figure 9 shows the variables that are most important in discriminating each class.

Problem 7:

(a)

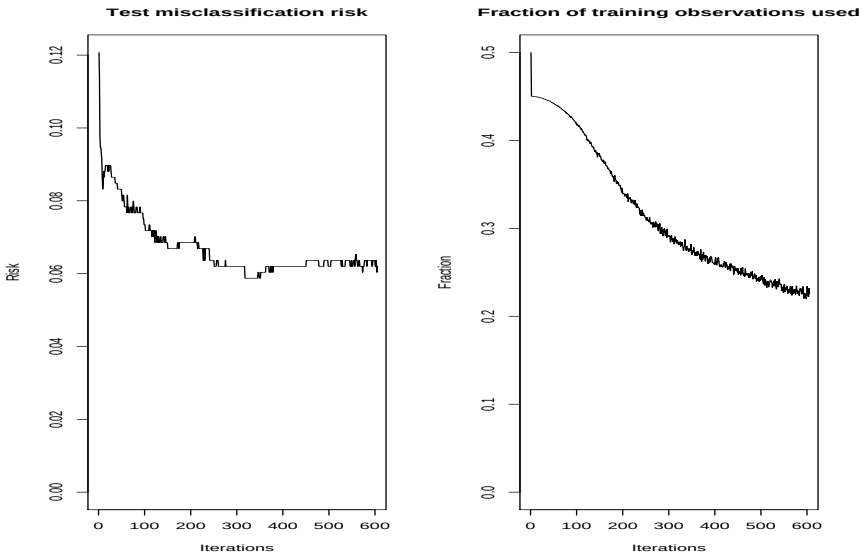

Figure 12: Spam Data. Left: misclassification error risk versus iteration. Right: the fraction of training observations used.

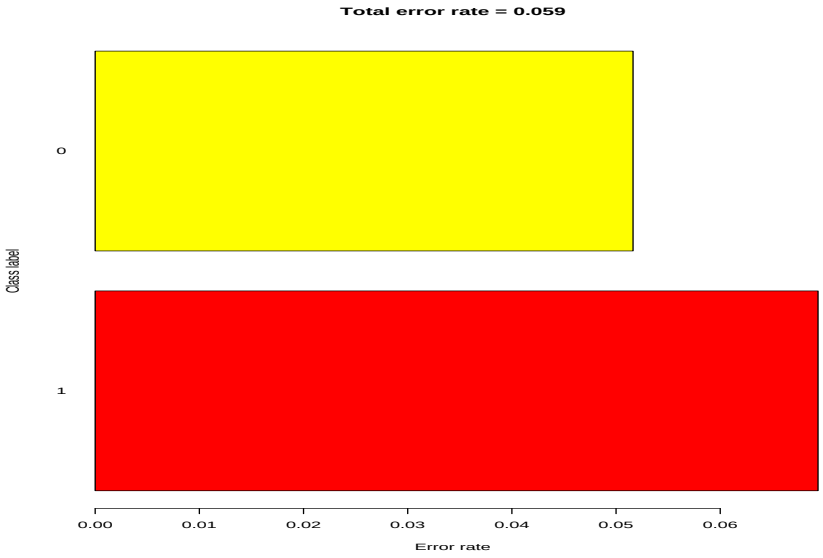

Figure 13: Spam Data. Misclassification error risk for each class.

I build MART model based on the training subset of the original `spam` data set. The indicator data set for training or test can be downloaded from the ESL textbook website:

http://www-stat.stanford.edu/ tibs/ElemStatLearn/datasets/spam.traintest

The misclassification error risk is 0.0587 (after 400 iterations, also see Figure 12). The misclassification risk for the test set (1536 observations) is 0.0553. Particularly, the misclassification risk for spam emails in the test set is 0.079 while the misclassification risk for the non-spam emails is 0.0404 (see Figure 13).

```
> spam <- read.table("spam.data", sep=",")
> index <- scan("spam.traintest")
> x <- as.matrix(spam[, 1:57])
> y <- spam[, 58]
> lx <- rep(1, 57)
> mart(x[index==0, ], y[index==0], lx, martmode='class')
 MART execution finished.
   iters       best   test misclass risk
    200         149      0.6688E-01
> moremart()
 MART execution finished.
   iters       best   test misclass risk
    400         315      0.5873E-01

> yhat <- martpred(x[index==1, ])
> sum(yhat!=y[index==1])/sum(index)
[1] 0.05533854
> y.spam <- y[(index==1) & (y==1)]
> y.nonspam <- y[(index==1) & (y==0)]
> yhat.spam <- martpred(x[(index==1) & (y==1), ])
> yhat.nonspam <- martpred(x[(index==1) & (y==0), ])
> sum(1-yhat.spam)/length(y.spam)
[1] 0.0789916
> sum(yhat.nonspam)/length(y.nonspam)
[1] 0.04038257
```

(b)

I set the cost matrix in `mart` procedure to be $\begin{pmatrix} 0 & 1 \\ k & 0 \end{pmatrix}$, where I search for the optimal k that controls the misclassification risk for non-spam emails to be less than 0.3% and minimizes the total misclassification risk. After some trials on the test set I used in (a), the best choice is identified as $k = 26$. The misclassification risk for spam and non-spam emails is 0.796 and 0 respectively and the total error risk is 0.318 (Figures 14). I then apply this MART model to the test set. The misclassification risk for spam and non-spam emails is 0.787 and 0 respectively and the total error risk is 0.305.

```
> k <- 26; spam.cost <- matrix(c(0, k, 1, 0), 2)
> mart(x[index==0, ], y[index==0], lx, martmode='class', cost.mtx=spam.cost)
 MART execution finished.
   iters       best   test misclass risk
    200         150      0.3181

> yhat <- martpred(x[index==1, ])
> sum(yhat!=y[index==1])/sum(index)
[1] 0.3046875
```

```
> y.spam <- y[(index==1) & (y==1)]
> y.nonspam <- y[(index==1) & (y==0)]
> yhat.spam <- martpred(x[(index==1) & (y==1), ])
> yhat.nonspam <- martpred(x[(index==1) & (y==0), ])
> sum(1-yhat.spam)/length(y.spam)
[1] 0.7865546
> sum(yhat.nonspam)/length(y.nonspam)
[1] 0
```

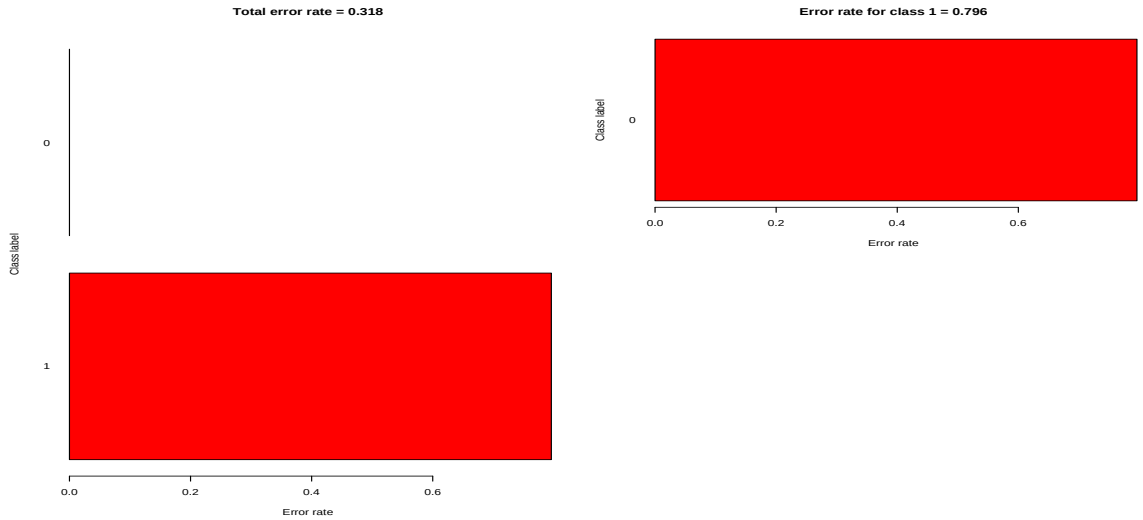

Figure 14: Spam Data. Misclassification error risk for each class with weight penalty adjustment. Left: total error risk. Right: error risk for spam emails.

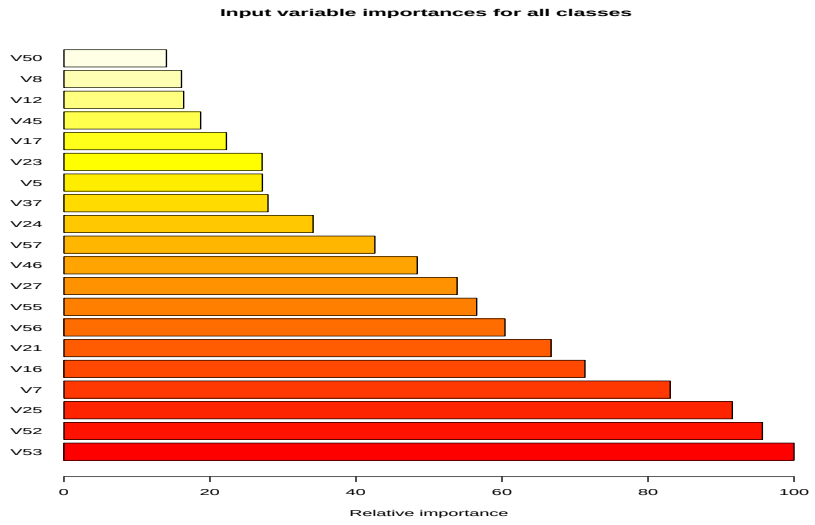

Figure 15: Spam Data. Misclassification error risk for each class with weight penalty adjustment.

Figure 15 shows the relative importance of each variable in discriminating good emails from spam. The four most important variables are the frequency of `remove`, `hp`, `!`, `$` in the emails respectively. Figures 16 and 17 show the dependence of response on these four most important variables when the response is good (non-spam) emails and spam respectively. It is seen that the probability of being good email decreases sharply with the frequency of all variables except for `hp`, where it increases sharply. Considering this data set comes from HP, it is not totally surprising to have such result. Also worth of noting is that Figures 16 and 17 are mirror reflections to each other since this is a 2-class classification problem.

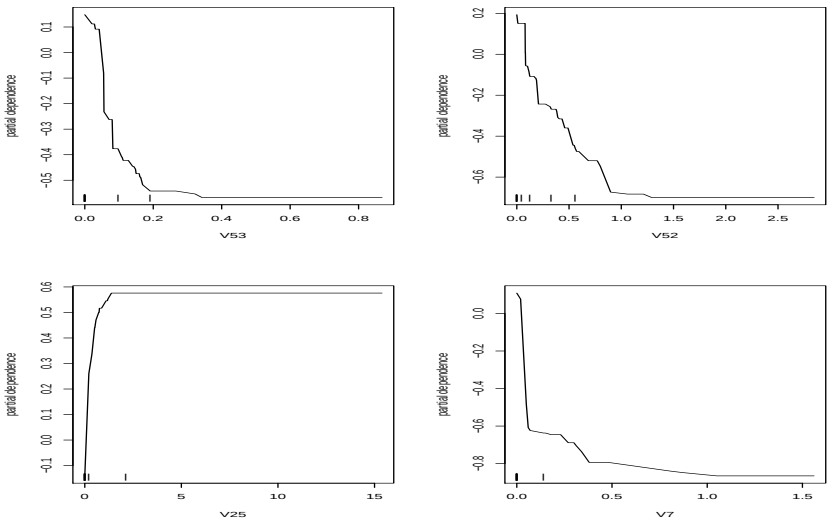

Figure 16: Spam Data. Dependece of response on `remove`, `hp`, `!`, `$` when the response is good email.

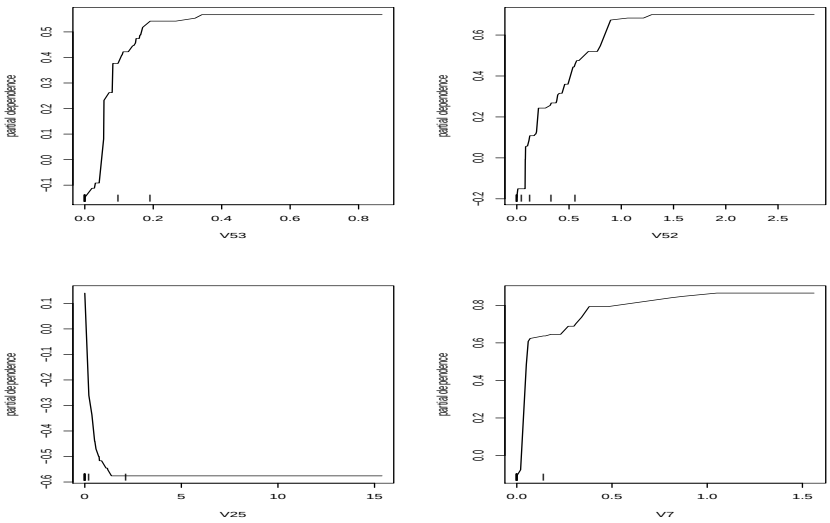

Figure 17: Spam Data. Dependece of response on `remove`, `hp`, `!`, `$` when the response is spam email.