ORF 350: Assignment 4

David Fan

4/10/2017

Collaborator: Brandon Tan

Question 1: Sentiment Analysis on Amazon Product Reviews (25 points)

```
setwd("/Users/dfan/Dropbox/School/Sophomore Year/Spring 2017/ORF 350/Assignments/HW4")
load("Amazon_SML.RData")  # loads object 'dat'

# 1a)
cat("Column names:", paste(names(dat), sep = ","))
```

```
## Column names: name review rating
```

```
paste("# of Reviews:", nrow(dat) - length(which(dat$review ==
    "")))
```

```
## [1] "# of Reviews: 1306"
```

```
paste("Unique products:", length(unique(dat$name)))
```

```
## [1] "Unique products: 20"
```

```
fivestar_ratings <- aggregate(rating ~ name, dat[which(dat$rating ==
    5), ], FUN = NROW)
paste("Product with most '5' ratings:", fivestar_ratings[order(fivestar_ratings$rating,
    decreasing = TRUE)[1], 1])
```

```
## [1] "Product with most '5' ratings: Vulli Sophie the Giraffe Teether"
```

```
paste("Number of '5' ratings for product:", fivestar_ratings[order(fivestar_ratings$rating,
    decreasing = TRUE)[1], 2])
```

```
## [1] "Number of '5' ratings for product: 526"
```

```
onestar_ratings <- aggregate(rating ~ name, dat[which(dat$rating ==
    1), ], FUN = NROW)
strwrap(paste("Product with most '1' ratings:", onestar_ratings[order(onestar_ratings$rating,
    decreasing = TRUE)[1], 1]))
```

```
## [1] "Product with most '1' ratings: Infant Optics DXR-5 2.4 GHz Digital"
## [2] "Video Baby Monitor with Night Vision"
```

```
paste("Number of '1' ratings for product:", onestar_ratings[order(onestar_ratings$rating,
    decreasing = TRUE)[1], 2])
```

```
## [1] "Number of '1' ratings for product: 68"
```

```
# 1b)
paste("Number of total '1' ratings:", length(which(dat$rating ==
    1)))
```

```
## [1] "Number of total '1' ratings: 656"
```

```r
paste("Number of total '5' ratings:", length(which(dat$rating ==
    5)))
```

```
## [1] "Number of total '5' ratings: 656"
```

```r
paste("The best performance of a constant classifer is 50%")
```

```
## [1] "The best performance of a constant classifer is 50%"
```

```r
# 1c) run tdMat.R and splitData.R first
source("tdMat.R")
source("splitData.R")
set.seed(10)
lambda <- exp(seq(-20, -1, length.out = 99))
model <- glmnet(train.x, train.y, family = "binomial")
cvfit <- cv.glmnet(train.x, train.y, family = "binomial", type.measure = "class",
    lambda = lambda)
model_coef <- data.frame(rownames(coef(model, s = cvfit$lambda.1se)),
    matrix(coef(model, s = cvfit$lambda.1se)))
names(model_coef) <- c("Variable", "Coefficient")
model_coef <- model_coef[order(model_coef$Coefficient, decreasing = TRUE),
    ]
paste("Twenty most positive coefficients:")
```

```
## [1] "Twenty most positive coefficients:"
```

```r
pos_coef <- model_coef[1:20, ]
pos_coef
```

```
##          Variable Coefficient
## 2203       wimper   3.4484919
## 617         endur   1.9268979
## 1364    overwhelm   1.6526359
## 1130         love   1.6247009
## 396      cordless   1.5848854
## 1941        teeth   1.5756916
## 651         excel   1.5435985
## 976        infanc   1.0479114
## 633          euro   1.0002475
## 1459     precious   0.9732243
## 478        definit   0.9085135
## 1265 neighborhood   0.8889463
## 1461     pregnant   0.8861453
## 1402      perfect   0.8775988
## 2255          yrs   0.8425592
## 2135         voic   0.8044794
## 892         haven   0.8010955
## 23           ador   0.7841694
## 1536      recharg   0.7605233
## 695       favorit   0.6840640
```

```r
model_coef <- model_coef[order(model_coef$Coefficient, decreasing = FALSE),
    ]
paste("Twenty most negative coefficients:")
```

```
## [1] "Twenty most negative coefficients:"
```

```r
neg_coef <- model_coef[1:20, ]
neg_coef
```

```
##        Variable Coefficient
## 1280 nightlight  -2.2077137
## 1774       solv  -1.8739763
## 187         bin  -1.8315534
## 1051       lame  -1.2349103
## 2077   unfortun  -1.0879947
## 2159       wast  -0.9929895
## 1597     return  -0.9884418
## 2107    useless  -0.9728764
## 2218        won  -0.9514425
## 543      doesnt  -0.9206300
## 141        bare  -0.8752857
## 1454      pound  -0.8650501
## 1915    swallow  -0.7820761
## 1863       stop  -0.7703378
## 673      fabric  -0.7167765
## 1323        off  -0.7134508
## 1779     someth  -0.6803976
## 1295        not  -0.6790863
## 1850       stay  -0.6246317
## 519  disappoint  -0.6095902
```

```r
# 1d)
most_pos <- ""
for (i in 1:nrow(pos_coef)) {
    if (sum(train.x[, as.character(pos_coef[i, 1])]) > 10) {
        most_pos <- pos_coef[i, 1]
        break
    }
}
paste("Most positive word in more than 10 docs:", most_pos)
```

```
## [1] "Most positive word in more than 10 docs: love"
```

```r
most_neg <- ""
for (i in 1:nrow(neg_coef)) {
    if (sum(train.x[, as.character(neg_coef[i, 1])]) > 10) {
        most_neg <- neg_coef[i, 1]
        break
    }
}
paste("Most negative word in more than 10 docs:", most_neg)
```

```
## [1] "Most negative word in more than 10 docs: unfortun"
```

```r
paste("Articles rated 1 with 'love':", length(which(train.x[,
    "love"] > 0 & train.y == 0)))
```

```
## [1] "Articles rated 1 with 'love': 61"
```

```r
paste("Articles rated 5 with 'love':", length(which(train.x[,
    "love"] > 0 & train.y == 1)))
```

```
## [1] "Articles rated 5 with 'love': 370"
```

```r
paste("Articles rated 1 with 'unfortun':", length(which(train.x[,
    "unfortun"] > 0 & train.y == 0)))
```

```
## [1] "Articles rated 1 with 'unfortun': 18"
```

```r
paste("Articles rated 5 with 'unfortun':", length(which(train.x[,
    "unfortun"] > 0 & train.y == 1)))
```

```
## [1] "Articles rated 5 with 'unfortun': 1"
```

```r
print("First article using 'love'")
```

```
## [1] "First article using 'love'"
```

```r
head(dat[as.numeric(train.tag[which(train.x[, "love"] > 0)])[1],
    "review"])
```

```
## [1] I found this one item to be an amazing value for the money.  The sling was helpful in the early
## 182643 Levels:  ...
```

```r
print("First article using'unfortunate'")
```

```
## [1] "First article using'unfortunate'"
```

```r
head(dat[as.numeric(train.tag[which(train.x[, "unfortun"] > 0)])[1],
    "review"])
```

```
## [1] Unfortunately, it doesn\\'t block the sun.
## 182643 Levels:  ...
```

```r
# 1e)
predictions <- as.numeric(predict(model, test.x, type = "class",
    s = cvfit$lambda.1se))
paste("Misclassification rate of 1:", length(intersect(which(predictions !=
    0), which(test.y == 0)))/length(which(test.y == 0)))
```

```
## [1] "Misclassification rate of 1: 0.0923076923076923"
```

```r
paste("Misclassification rate of 5:", length(intersect(which(predictions !=
    1), which(test.y == 1)))/length(which(test.y == 1)))
```

```
## [1] "Misclassification rate of 5: 0.119402985074627"
```

```r
paste("Total misclassification rate:", length(which(predictions !=
    test.y))/length(predictions))
```

```
## [1] "Total misclassification rate: 0.106060606060606"
```

Better than the constant classifier which at best has a misclassification rate of 50%!

Question 2: Non-existence of MLE for Logistic Regression (10 points)

2a)

$$L(\beta) = \prod_{i=1,y=1}^{n} \eta(x_i) \prod_{i=1,y=0}^{n} (1-\eta(x_i))$$
$$l(\beta) = log(\prod_{i=1,y=1}^{n} \eta(x_i) \prod_{i=1,y=0}^{n} (1-\eta(x_i)))$$
$$= \sum_{i=1}^{n} I(Y_i = 1) log(\eta(x_i)) + \sum_{i=1}^{n} I(Y_i = 0) log(1-\eta(x_i))$$

2b)

$$\frac{dl(\beta)}{d\beta} = \sum_{i=1}^{n} I(Y_i = 1) * \frac{(1+e^{\beta x_i})x_i\beta e^{\beta x_i} - e^{\beta x_i}(x_i\beta e^{\beta x_i})}{(1+e^{\beta x_i})^2} * \frac{1+e^{\beta x_i}}{e^{\beta x_i}} + \sum_{i=1}^{n} I(Y_i = 0) * \frac{1+e^{\beta x_i}}{1} * \frac{-e^{\beta x_i}}{(1+e^{\beta x_i})^2}$$
$$= \sum_{i=1}^{n} I(y_i = 1) * \frac{1}{1+e^{\beta x_i}} x_i - \sum_{i=1}^{n} I(y_i = 0) * \frac{e^{\beta x_i}}{1+e^{\beta x_i}} x_i$$

When y = 0, only the second sum contributes. $\frac{e^{\beta x_i}}{1+e^{\beta x_i}} > 0$ and since $x_i \leq 0$, the total sum is positive. When y = 1, only the first sum contributes. $\frac{1}{1+e^{\beta x_i}} > 0$ and since $x_i > 0$, the total sum is also positive. This means that for any given values of x_i and y, the derivative will be positive and so there is no single value of β that maximizes this likelihood function. That is, the MLE $\hat{\beta} = \infty$.

Question 3: LDA and QDA for Multivariate Gaussian (25 points)

3.1: Likelihood Model (5 points)

$$
\begin{aligned}
l(\mu_1, \mu_2, \eta, \sum) &= \sum_{i=1}^{n} logp(x_i, y_i) \\
&= \sum_{i=1}^{n} I(y_i = 1)log(p(x_i, y_i)) + \sum_{i=1}^{n} I(y_i = 2)log(p(x_i, y_i)) \\
&= \sum_{i=1}^{n} I(y_i = 1)log[p(x_i|y_i = 1) * p(y_i = 1)] + \sum_{i=1}^{n} I(y_i = 2)log[p(x_i|y_i = 2) * p(y_i = 2)]
\end{aligned}
$$

Let $\eta = P(y_i = 1)$ and $1 - \eta = P(Y_i = 2)$
$= \sum_{i=1}^{n} I(y_i = 1)log[\frac{1}{(2\pi)^{d/2}|\sum|^{1/2}} e^{-(x_i-\mu_1)^T \sum^{-1}(x_i-\mu_1)/2}] +$
$\sum_{i=1}^{n} I(y_i = 1) * log(\eta) + \sum_{i=1}^{n} I(y_i = 2)log[\frac{1}{(2\pi)^{d/2}|\sum|^{1/2}} e^{-(x_i-\mu_2)^T \sum^{-1}(x_i-\mu_2)/2}] + \sum_{i=1}^{n} I(y_i = 2) * log(1 - \eta)$
$= -\frac{nd}{2}log(2\pi) - \frac{n}{2}log|\sum| - \frac{1}{2}\sum_{i=1}^{n} I(y_i = 1)((x_i - \mu_1)^T \sum^{-1}(x_i - \mu_1)) - \frac{1}{2}\sum_{i=1}^{n} I(y_i = 2)((x_i - \mu_2)^T \sum^{-1}(x_i - \mu_2)) + n_1 log(\eta) + n_2 log(1 - \eta)$

3.2: MLE Estimation (10 points)

$$
\begin{aligned}
\frac{d(l(\mu_1, \mu_2, \eta, \sum))}{d\eta} &= \frac{n_1}{\eta} - \frac{n_2}{1-\eta} \qquad\qquad = 0 \\
\frac{n_1}{\eta} &= \frac{n_2}{1-\eta} \\
n_1 - n_1\eta - n_2\eta &= 0 \\
-\eta(n_1 + n_2) &= -n_1 \\
\hat{\eta} &= \frac{n_1}{n_1 + n_2}
\end{aligned}
$$

$$
\begin{aligned}
\frac{d(l(\mu_1, \mu_2, \eta, \sum))}{d\mu_1} &= \frac{d}{d\mu_1}[-\frac{1}{2}\sum_{i=1}^{n}(I(y_i = 1)(x_i - \mu_1)^T)\sum^{-1}(x_i - \mu_1)] \\
\frac{-1*2}{2}\sum_{i=1}^{n} I(y_i = 1)\sum^{-1}(x_i - \mu_1) &= 0 \\
\sum^{-1}(\sum_{i=1}^{n}[I(y_i = 1)x_i] - n_1\mu_1) &= 0 \\
\sum_{i=1}^{n} I(y_i = 1)x_i &= n_1\mu_1 \\
\hat{\mu_1} &= \frac{\sum_{i=1}^{n} I(y_i = 1)x_i}{n_1}
\end{aligned}
$$

In the same manner, we get $\hat{\mu_2} = \frac{\sum_{i=1}^{n} I(y_i=2)x_i}{n_2}$.

3.3: MLE Estimation (10 points)

Since $\sum^{-1}$ is symmetric, $\frac{\partial log|\sum|^{-1}}{\partial \sum^{-1}} = \sum$. Since $(x_i - \mu_1)^T \sum^{-1}(x_i - \mu_1)$ is scalar, we can write this to be equal to $Tr((x_i - \mu_1)^T \sum^{-1}(x_i - \mu_1))$.

$$\begin{aligned}
\frac{\partial((x_i - \mu_1)^T \sum^{-1}(x_i - \mu_1))}{\partial \sum^{-1}} &= \frac{\partial(Tr((x_i - \mu_1)^T \sum^{-1}(x_i - \mu_1)))}{\partial \sum^{-1}} \\
\textit{by hint 3} \quad &= \frac{\partial(Tr((x_i - \mu_1)(x_i - \mu_1)^T \sum^{-1}))}{\partial \sum^{-1}} \\
\textit{by hint 4} \quad &= ((x_i - \mu_1)(x_i - \mu_1)^T)^T \\
&= (x_i - \mu_1)(x_i - \mu_1)^T
\end{aligned}$$

In the same manner, we get $\frac{\partial((x_i - \mu_2)^T \sum^{-1}(x_i - \mu_2))}{\partial \sum^{-1}} = (x_i - \mu_2)(x_i - \mu_2)^T$

$$\frac{\partial(l(\mu_1, \mu_2, \eta, \sum^{-1}))}{\partial \sum^{-1}} = \frac{n}{2}\sum - \frac{1}{2}\sum_{i=1}^{n} I(y_i = 1)(x_i - \mu_1)(x_i - \mu_1)^T - \frac{1}{2}\sum_{i=1}^{n} I(y_i = 2)(x_i - \mu_2)(x_i - \mu_2)^T \quad = 0$$

$$\sum = \frac{1}{n}\sum_{i=1}^{n} I(y_i = 1)(x_i - \mu_1)(x_i - \mu_1)^T + \frac{1}{n}\sum_{i=1}^{n} I(y_i = 2)(x_i - \mu_2)(x_i - \mu_2)^T$$

Let $S_1 = \frac{1}{n_1}\sum_{i:y_i=1}^{n}(x_i - \mu_1)(x_i - \mu_1)^T$ and $S_2 = \frac{1}{n_2}\sum_{i:y_i=2}^{n}(x_i - \mu_2)(x_i - \mu_2)^T$.
Then, $\hat{\sum} = \frac{1}{n}[n_1 s_1 + n_2 s_2]$.

Question 4: Naive Bayes (25 points)

4.1 (10 points)

```
# 4.1
source("./SpamAssassin/readRawEmail.R")
mail <- readAllMessages(dirs = c("./SpamAssassin/easy_ham", "./SpamAssassin/spam"))

list_body <- numeric(0)
spam <- rep(0, length(mail))
for (i in 1:length(mail)) {
    tmp = mail[[i]]$body
    tmp2 = paste(tmp$text, collapse = "")
    tmp3 = gsub("\\b([[:punct:]|[:digit:]])*[a-zA-Z]*([[:punct:]|[:digit:]])+
                [a-zA-Z]*([[:punct:]|[:digit:]])*",
        " ", tmp2)
    tmp4 = gsub("[^A-Za-z]", " ", tmp3)
    list_body[[i]] <- tmp4
    spam[i] <- mail[[i]]$spam
}
res <- as.matrix(TermDocumentMatrix(Corpus(VectorSource(list_body)),
    control = list(removePunctuation = TRUE, stemming = TRUE,
        wordLengths = c(3, 20))))

JH_1 <- apply(res[, which(spam == 0)], 1, sum)/apply(res[, which(spam ==
    0)] > 0, 1, sum)
JH_2 <- apply(res[, which(spam == 0)] > 0, 1, sum)/length(which(spam ==
    0))
JS_1 <- apply(res[, which(spam == 1)], 1, sum)/apply(res[, which(spam ==
    1)] > 0, 1, sum)
JS_2 <- apply(res[, which(spam == 1)] > 0, 1, sum)/length(which(spam ==
    1))
sort(JH_1, decreasing = TRUE)[1:10]
```

```
##   ximian     gaim    powel  datapow dinosaur  dirksen   hextab    tribe
## 39.00000 38.00000 27.00000 23.50000 21.50000 21.00000 21.00000 21.00000
##  idefens      alb
## 19.00000 16.33333
```

```
sort(JH_2, decreasing = TRUE)[1:10]
```

```
##       the       and      that       for       com      this       you
## 0.9072212 0.8011883 0.7148080 0.6617916 0.6435101 0.5923218 0.5603291
##      with      have      http
## 0.5598720 0.5548446 0.5438757
```

```
sort(JS_1, decreasing = TRUE)[1:10]
```

```
##   enenkio      atol blockquot     freak      milf ebonylust    hermio
## 128.00000  91.00000  90.83333  58.00000  58.00000  57.00000  54.00000
##      king    option       kio
##  50.00000  48.64286  45.00000
```

```
sort(JS_2, decreasing = TRUE)[1:10]
```

```
##       the      this       you       and      http      your       for
```

```
## 0.6857430 0.6757028 0.6706827 0.6395582 0.6345382 0.6265060 0.6184739
##       com      from       are
## 0.5361446 0.4969880 0.4949799
```

4.2 (10 points)

MLE Derivations:

Let $P(s_j = 1) = \eta$.

$$
\begin{aligned}
logp(w_{1:m,1:n}, y_{1:m,1:n}, s_{1:n}) &= log \prod_{j=1}^{n} p(s_j) \prod_{i=1}^{m} p(w_{ij}, y_{ij}|s_j) \\
&= \sum_{j=1}^{n} [logp(s_j) + \sum_{i=1}^{m} logp(w_{ij}, y_{ij}|s_j)] \\
&= \sum_{j \in J_H} (log(1-\eta) + \sum_{i=1}^{m} logp^{-}(w_{ij}, y_{ij})) + \sum_{j \in J_S} (log(\eta) + \sum_{i=1}^{m} logp^{+}(w_{ij}, y_{ij}))
\end{aligned}
$$

Where $logp^{-}(w_{ij}, y_{ij}) = logp(w_{ij}, y_{ij}|s_j = 0) = w_{ij} log(\frac{\theta_i^{-} e^{-\lambda_i^{-}} \lambda_{i^{-}}^{y_{ij}-1}}{(y_{ij}-1)!}) + (1 - w_{ij}) log(1 - \theta_i^{-})$ and $logp^{+}(w_{ij}, y_{ij}) = logp(w_{ij}, y_{ij}|s_j = 1) = w_{ij} log(\frac{\theta_i^{+} e^{-\lambda_i^{+}} \lambda_{i^{+}}^{y_{ij}-1}}{(y_{ij}-1)!}) + (1 - w_{ij}) log(1 - \theta_i^{+})$.

Let $n_0 = |J_H|$ and $n_1 = |J_S|$.

$$
\begin{aligned}
\frac{d()}{d\eta} &= -\sum_{j \in J_H} \frac{1}{1-\eta} + \sum_{j \in J_S} \frac{1}{\eta} \qquad = 0 \\
-\frac{n_0}{1-\eta} + \frac{n_1}{\eta} &= 0 \\
\frac{n_0}{1-\eta} &= \frac{n_1}{\eta} \\
n_0 \eta &= n_1 - n_1 \eta \\
\eta(n_0 + n_1) &= n_1 \\
\hat{\eta} &= \frac{n_1}{n_1 + n_0}
\end{aligned}
$$

$$\frac{d()}{d\hat{\theta}_i^-} = \sum_{j \in J_H} \frac{w_{ij}}{\theta_i^-} + \sum_{j \in J_H} \frac{-(1 - w_{ij})}{1 - \theta_i^-} \qquad = 0$$

$$\frac{n_0}{\theta_i^-} \sum_{j \in J_H} w_{ij} - \frac{n_0}{1 - \theta_i^-} \sum_{j \in J_H} (1 - w_{ij}) = 0$$

$$\frac{1 - \theta_i^-}{\theta_i^-} = \frac{\sum\limits_{j \in J_H} (1 - w_{ij})}{\sum\limits_{j \in J_H} w_{ij}}$$

$$\frac{1}{\theta_i^-} = \frac{\sum\limits_{j \in J_H} 1}{\sum\limits_{j \in J_H} w_{ij}} = \frac{|J_H|}{\sum\limits_{j \in J_H} w_{ij}}$$

$$\hat{\theta_i^-} = \frac{\sum\limits_{j \in J_H} w_{ij}}{|J_H|}$$

Similarly, we get $\hat{\theta_i^+} = \frac{\sum\limits_{j \in J_S} w_{ij}}{|J_S|}$.

$$\frac{d()}{d(\lambda_i^-)} = \sum_{j \in J_H} \left(\frac{w_{ij}(y_{ij} - 1)}{\lambda_i^-} - w_{ij}\right) \qquad = 0$$

$$\frac{1}{\lambda_i^-} \sum_{j \in J_H} (w_{ij}(y_{ij} - 1)) = \sum_{j \in J_H} w_{ij}$$

$$\hat{\lambda_i^-} = \frac{\sum\limits_{j \in J_H} (w_{ij}(y_{ij} - 1))}{\sum\limits_{j \in J_H} w_{ij}}$$

Similarly, we get $\hat{\lambda_i^+} = \frac{\sum\limits_{j \in J_S} (w_{ij}(y_{ij}-1))}{\sum\limits_{j \in J_S} w_{ij}}$.

```
# 4.2
set.seed(1)
testingidx = sample(1:ncol(res), 100)
trainingidx = 1:ncol(res)
trainingidx = trainingidx[-testingidx]

res_train <- res[, trainingidx]
spam_train <- spam[trainingidx]
res_test <- res[, testingidx]
spam_test <- spam[testingidx]

# Calculate MLEs eta hat
n1 <- length(which(spam_train == 1))
n0 <- length(which(spam_train == 0))
eta_hat <- n1/(n0 + n1)

# theta hat and theta hat prime wi,j = 1 if res[i,j] > 0 , =
# 0 otherwise sj = 1 if document j is spam, 0 otherwise
```

```r
theta_hat <- apply(res_train[, spam_train] > 0, 1, sum)/n1
theta_hat_p <- apply(res_train[, !spam_train] > 0, 1, sum)/n0

# lambda hat and lambda hat prime
denom <- (theta_hat * n1)
denom[which(denom == 0)] <- denom[which(denom == 0)] + 1e-04  # pertubation to avoid division by 0
lambda_hat <- apply((res_train[, spam_train] > 0) * (res_train[,
    spam_train] - 1), 1, sum)/denom
denom <- (theta_hat_p * n0)
denom[which(denom == 0)] <- denom[which(denom == 0)] + 1e-04  # pertubation to avoid division by 0
lambda_hat_p <- apply((res_train[, !spam_train] > 0) * (res_train[,
    !spam_train] - 1), 1, sum)/denom

# prediction accuracy on testing data
log_perturb <- function(x) {
    return(ifelse(x == 0, log(x + 1e-04), log(x)))
}

test_predictions <- rep(0, ncol(res_test))
log_prob1 <- rep(0, ncol(res_test))
log_prob2 <- rep(0, ncol(res_test))

for (j in 1:ncol(res_test)) {
    log_prob1[j] <- sum((res_test[, j] > 0) * (log_perturb(theta_hat) -
        lambda_hat + (res_test[, j] - 1) * log_perturb(lambda_hat)),
        na.rm = TRUE)
    log_prob1[j] <- log_prob1[j] + sum((res_test[, j] == 0) *
        (log_perturb(1 - theta_hat)), na.rm = TRUE)
    log_prob2[j] <- sum((res_test[, j] > 0) * (log(theta_hat_p) -
        lambda_hat_p + (res_test[, j] - 1) * log_perturb(lambda_hat_p)),
        na.rm = TRUE)
    log_prob2[j] <- log_prob2[j] + sum((res_test[, j] == 0) *
        log_perturb(1 - theta_hat_p), na.rm = TRUE)
    p_spam <- sum(spam_train)/length(spam_train)
    test_predictions[j] <- (log_prob1[j] - log_prob2[j] + log(p_spam/(1 -
        p_spam)) > 0)
}
paste("Prediction accuracy on test data:", mean(test_predictions ==
    spam_test))
```

```
## [1] "Prediction accuracy on test data: 0.95"
```

```r
# 4.3
source("./SpamAssassin/readRawEmail.R")
mail <- readAllMessages(dirs = c("./SpamAssassin/easy_ham", "./SpamAssassin/spam"))

list_body <- numeric(0)
spam <- rep(0, length(mail))
for (i in 1:length(mail)) {
    tmp = mail[[i]]$body
    tmp2 = paste(tmp$text, collapse = "")
    tmp3 = gsub("\\b([[:punct:]])", "", tmp2)
    # tmp3 =
    # gsub('\\b([[:punct:]]|[:digit:]])*[a-zA-Z]*([[:punct:]]|[:digit:]])+
```

```r
    # [a-zA-Z]*([[:punct:]|[:digit:]])*',' ',tmp2)
    tmp4 = gsub("[^A-Za-z]", " ", tmp3)
    list_body[[i]] <- tmp4
    spam[i] <- mail[[i]]$spam
}
res <- as.matrix(TermDocumentMatrix(Corpus(VectorSource(list_body)),
    control = list(removePunctuation = TRUE, stemming = TRUE,
        wordLengths = c(3, 20))))

set.seed(1)
testingidx = sample(1:ncol(res), 100)
trainingidx = 1:ncol(res)
trainingidx = trainingidx[-testingidx]

res_train <- res[, trainingidx]
spam_train <- spam[trainingidx]
res_test <- res[, testingidx]
spam_test <- spam[testingidx]

# Calculate MLEs eta hat
n1 <- length(which(spam_train == 1))
n0 <- length(which(spam_train == 0))
eta_hat <- n1/(n0 + n1)

# theta hat and theta hat prime wi,j = 1 if res[i,j] > 0 , =
# 0 otherwise sj = 1 if document j is spam, 0 otherwise
theta_hat <- apply(res_train[, spam_train] > 0, 1, sum)/n1
theta_hat_p <- apply(res_train[, !spam_train] > 0, 1, sum)/n0

# lambda hat and lambda hat prime
denom <- (theta_hat * n1)
denom[which(denom == 0)] <- denom[which(denom == 0)] + 1e-04  # pertubation to avoid division by 0
lambda_hat <- apply((res_train[, spam_train] > 0) * (res_train[,
    spam_train] - 1), 1, sum)/denom
denom <- (theta_hat_p * n0)
denom[which(denom == 0)] <- denom[which(denom == 0)] + 1e-04  # pertubation to avoid division by 0
lambda_hat_p <- apply((res_train[, !spam_train] > 0) * (res_train[,
    !spam_train] - 1), 1, sum)/denom

# prediction accuracy on testing data
log_perturb <- function(x) {
    return(ifelse(x == 0, log(x + 1e-04), log(x)))
}

test_predictions <- rep(0, ncol(res_test))
log_prob1 <- rep(0, ncol(res_test))
log_prob2 <- rep(0, ncol(res_test))

for (j in 1:ncol(res_test)) {
    log_prob1[j] <- sum((res_test[, j] > 0) * (log_perturb(theta_hat) -
        lambda_hat + (res_test[, j] - 1) * log_perturb(lambda_hat)),
        na.rm = TRUE)
    log_prob1[j] <- log_prob1[j] + sum((res_test[, j] == 0) *
```

```
        (log_perturb(1 - theta_hat)), na.rm = TRUE)
    log_prob2[j] <- sum((res_test[, j] > 0) * (log(theta_hat_p) -
        lambda_hat_p + (res_test[, j] - 1) * log_perturb(lambda_hat_p)),
        na.rm = TRUE)
    log_prob2[j] <- log_prob2[j] + sum((res_test[, j] == 0) *
        log_perturb(1 - theta_hat_p), na.rm = TRUE)
    p_spam <- sum(spam_train)/length(spam_train)
    test_predictions[j] <- (log_prob1[j] - log_prob2[j] + log(p_spam/(1 -
        p_spam)) > 0)
}
paste("Prediction accuracy on testing data:", mean(test_predictions ==
    spam_test))
```

```
## [1] "Prediction accuracy on testing data: 0.92"
```

My intuition was that removing the punctuation does most of the heavy-lifting, since punctuation doesn't really mean anything with regards to classifying an email as spam or not. Including it in the model just creates uneccessary noise and distracts the model from what is more important, like looking at the start of each word to see if it's a number of not (maybe indicating a price or sales ad) or removing repetitive character sequences. I simplified regex #3 such that it only removes punctuation. As I expected, the prediction accuracy only dropped to 92%.

Question 5: The Bayes Rule (15 points)

5.1 (10 points)

P(x = 1) = P(x = 1 | y = 0)P(y = 0) + P(x = 1 | y = 1)P(y = 1) = $\frac{1}{3} * \frac{1}{2} + 0 * \frac{1}{2} = \frac{1}{6}$.
P(x = 2) = P(x = 2 | y = 0)P(y = 0) + P(x = 2 | y = 1)P(y = 1) = $\frac{2}{3} * \frac{1}{2} + \frac{1}{3} * \frac{1}{2} = \frac{1}{2}$.
P(x = 3) = P(x = 3 | y = 0)P(y = 0) + P(x = 3 | y = 1)P(y = 1) = $0 * \frac{1}{2} + \frac{2}{3} * \frac{1}{2} = \frac{1}{3}$.

Bayes rule: $h^*(x) = 1$ if P(Y=1|X=x) > P(Y=0|X=x), 0 otherwise.

$h^*(1):$
$P(Y = 1|X = 1) = \frac{0*1/2}{0*\frac{1}{2}+\frac{1}{2}*\frac{1}{2}} = 0$
$P(Y = 0|X = 1) = \frac{\frac{1}{3}*\frac{1}{2}}{0*\frac{1}{2}+\frac{1}{3}*\frac{1}{2}} = 1$
So, $h^*(1) = 0$.

$h^*(2):$
$P(Y = 1|X = 2) = \frac{\frac{1}{3}*\frac{1}{2}}{\frac{1}{3}*\frac{1}{2}+\frac{2}{3}*\frac{1}{2}} = \frac{1}{3}$
$P(Y = 0|X = 2) = \frac{\frac{2}{3}*\frac{1}{2}}{\frac{1}{3}*\frac{1}{2}+\frac{2}{3}*\frac{1}{2}} = \frac{2}{3}$
So, $h^*(2) = 0$.

$h^*(3):$
$P(Y = 1|X = 3) = \frac{\frac{2}{3}*\frac{1}{2}}{\frac{2}{3}*\frac{1}{2}+0*\frac{1}{2}} = 1$
$P(Y = 0|X = 3) = \frac{0*\frac{1}{2}}{\frac{2}{3}*\frac{1}{2}+0*\frac{1}{2}} = 0$
So, $h^*(3) = 1$.

Baye's risk:

$$\begin{aligned}
P(Y \neq h(X)) &= E_X[P(Y \neq h(X)|X)] \\
&= \sum_{k=1}^{3} P(Y \neq h(X)|X = k)P(X = k) \\
&= P(Y = 1|X = 1)P(X = 1) + P(Y = 1|X = 2)P(X = 2) + P(Y = 0|X = 3)P(X = 3) \\
&= 0 * \frac{1}{6} + \frac{1}{3} * \frac{1}{2} + 0 * \frac{1}{3} \\
&= \frac{1}{6}
\end{aligned}$$

5.2 (5 points)

The joint-conditional likelihood is $P(X_1, X_2|Y)$ and the marginal distribution is $P(Y)$.

(a) With the Naive Bayes classifier, we assume that the parameters are independent, so $P(X_1, X_2|Y) = P(X_1|Y)P(X_2|Y)$. Y is binary so there are two realizations of Y (0 and 1). We calculate P(Y=+1), and $P(X_1 = 1|Y = 1), P(X_2 = 1|Y = 1), P(X_1 = 1|Y = -1), P(X_2 = 1|Y = -1)$. That is five parameters.

(b) Without the Naive Bayes classifer, we don't assume the parameters are independent. We calculate P(Y = 1). We also calculate $P(X_1 = \pm 1, X_2 = \pm 1|Y = 1)$ which is 4 quantities, but we only need 3 to specify these since the sum of the probabilities equals 1. Similarly, we calculate $P(X_1 = \pm 1, X_2 = \pm 1|Y = -1)$ which is another 3 quantities. Thus, there are 1 + 3 + 3 = 7 quantities total.