COMS 4721 - Homework 1

Jeff Hudson (jdh2182)

Wednesday, January 28, 2015

Problem 1 (Maximum Likelihood)

Part 1 Imagine we have a sequence of N observations $(x_i, \ldots, x_N)$, where each $x_i \in 0, 1$. We model this sequence as i.i.d. Bernoulli random variables, where $p(x_i, \ldots, x_N|\pi) = \pi$ and π is unknown.

(a) What is the joint likelihood of the data.

$\mathcal{L}(\pi|x) = \prod p(x_i|\pi) = \pi^{\Sigma x_i} \times (1-\pi)^{N-\Sigma x_i}$

(b) Derive the maximum likelihood estimate $\hat{\pi}_{ML}$ for π.

Joint Likelihood: $\prod p(x_i|\pi) = \pi^{\Sigma x_i} \times (1-\pi)^{N-\Sigma x_i}$
Log Likelihood: $\sum \log p(x_i|\pi) = \sum x_i \log \pi + (N - \sum x_i) \log(1-\pi)$
Derivative: $\Delta_\pi [\sum x_i \log \pi + (N - \sum x_i) \log(1-\pi)] = \frac{\sum x_i}{\pi} + \frac{-(N - \sum x_i)}{1-\pi} = 0$
$\frac{\sum x_i}{\pi} = \frac{(N - \sum x_i)}{1-\pi}$
$\sum x_i - \pi \sum x_i = N\pi - \pi \sum x_i$
MLE: $\hat{\pi}_{ML} = \frac{\sum x_i}{N}$

(c) Explain why this maximum likelihood makes intuitive sense.

The maximum likelihood estimator for π corresponds to the empirical proportion of 1s (or successes) observed in $(x_i, \ldots, x_N)$. This makes good intuitive sense because if the estimator were any lower we would expect fewer 1s in the data, and conversely, if it were higher, we would expect more. Thus, the most likely estimate is the one directly supported by our empirical observation of the exact proportion in our data $(x_i, \ldots, x_N)$.

Part 2 Now imagine another sequence of N observations $(x_i, \ldots, x_N)$, where each $x_i \in 0, 1, 2, \ldots$. We model this sequence as i.i.d. Poisson random variables with unknown parameter λ. The following questions follow exactly from Part 1.

(a) What is the joint likelihood of the data.

$\mathcal{L}(\lambda|x) = \prod p(x_i|\lambda) = \prod \frac{\lambda^x}{x!} e^{-\lambda} = \frac{\lambda^{\Sigma x_i}}{\Pi(x_i!)} e^{-N\lambda}$

(b) Derive the maximum likelihood estimate $\hat{\lambda}_{ML}$ for λ.

Joint Likelihood: $\prod p(x_i|\lambda) = \frac{\lambda^{\Sigma x_i}}{\Pi(x_i!)} e^{-N\lambda}$
Log-Likelihood: $\sum x_i \log \lambda - \log \prod (x_i!) - N\lambda$
Derivative: $\frac{\Sigma x_i}{\lambda} - N = 0$
MLE: $\hat{\lambda}_{ML} = \frac{\Sigma x_i}{N}$

(c) Explain why this maximum likelihood makes intuitive sense.

This is the empirical mean of the data, since the expectation of a Poisson distribution is λ, it makes intuitive sense that the empirical mean is the maximum likelihood estimator for its expectation.

Problem 2 (Bayes Rule)

This problem builds on Part 2 of Problem 1. Again imagine we have a sequence of N non-negative integer-valued observations $(x_i, \dots, x_N)$, which we model as i.i.d. Poisson random variables with unknown parameter λ. We place a gamma prior distribution on λ written $\lambda \sim Gam(\lambda|a,b)$, where $Gam(\lambda|a,b) = \frac{b^a}{\Gamma(a)}\lambda^{a-1}e^{-b\lambda}$

(a) Use Bayes Rule to derive the posterior distribution of λ and identify the name of this distribution.
$p(\lambda|x_1, \dots, x_N) \propto p(x_1, \dots, x_N|\lambda)p(\lambda)$
$\frac{\lambda^{\Sigma x_i}}{\Pi x_i!}e^{-N\lambda}\frac{b^a}{\Gamma(a)}\lambda^{a-1}e^{-b\lambda} \to \frac{b^a}{\Gamma(a)\Pi x_i!}\lambda^{\Sigma x_i+a-1}e^{-\lambda(N+b)}$
Posterior: $\lambda \sim Gam(\Sigma x_i + a, N + b)$

(b) What is the mean and variance of λ under this posterior? Discuss how this relates to your solution to Part 2 of Problem 1.
Since λ is gamma distributed, its mean/expectation is $\frac{\Sigma x_i+a}{N+b}$ and the variance is $\frac{\Sigma x_i+a}{(N+b)^2}$
The expectation of λ looks similar to the maximum likelihood estimate derived in Part 2 of Problem 1, except with an added a in the numerator and b in the denominator which pulls it closer to the expected value of λ under our prior $(\frac{a}{b})$

Problem 3 (Linear Regression)

In this problem you will implement and analyze results using least squares linear regression. The goal is to predict the miles per gallon a car will get using six quantities about that car. The data can be found on the course website and on Courseworks. The data set contains 392 instances of different car models, each containing a 6-dimensional feature vector (plus a dimension of 1's for the intercept) contained in X, and its average miles per gallon which we treat as the output contained in y.

Remember to include all original code with your homework .pdf submission.

Part 1 First, randomly split the data set into 20 testing examples and 372 training examples. Using the training data only, solve a linear regression model of the form $y \approx w_0 + \sum_{j=1}^{6} x_j w_j$ using least squares

```r
# Set a seed to ensure consistent results
set.seed(11)

# Read in the data
X <- as.matrix(read.csv("X.txt", F))
Y <- as.matrix(read.csv("Y.txt", F))

# Create a vector representing the testing set
testset <- sample(nrow(X), 20)

# Create separate test and training matrices
Xtest <- X[testset,]
Ytest <- Y[testset,]

Xtrain <- X[-testset,]
Ytrain <- Y[-testset,]

# Compute X transpose
XT <- t(Xtrain)

# Use equation from class to generate Least Squares solution
w <- (solve(XT %*% Xtrain) %*% XT) %*% Ytrain
```

(a) Print the numbers you obtain for the vector $\hat{w}_{ML}$. Using the labels of each dimension contained in the readme file, explain what the sign of each value in $\hat{w}_{ML}$ says about the relationship of the inputs to the output.

```r
colnames(w) <- "Coefficient"
rownames(w) <- c("Intercept","Cylinders","Displacement","Horsepower",
                  "Weight","Acceleration","Year")
print(w)
```

```
##              Coefficient
## Intercept    23.44581028
## Cylinders    -0.66597593
## Displacement  0.92239463
## Horsepower   -0.03835564
## Weight       -5.77804380
## Acceleration  0.27924054
## Year          2.70868825
```

The negative values indicate that vehicles with a larger number/amount of Cylinders, Horsepower and Weight will, on average, have lower MPG (the response variable), while the positive signs for Displacement, Acceleration and Year indicate that higher values of these variables will be associated with increased MPG.

(b) Use the least squares solution to predict the outputs for each of the 20 testing examples.

```r
# Use solution from Least Squares to predict test set
Ypred <- Xtest %*% w
colnames(Ypred) <- "Prediction"
print(Ypred)
```

```
##       Prediction
##  [1,]  25.705179
##  [2,]  15.089612
##  [3,]  19.265695
##  [4,]  10.201428
##  [5,]   8.117139
##  [6,]  34.451414
##  [7,]  16.704204
##  [8,]  26.051398
##  [9,]  30.972574
## [10,]  18.881035
## [11,]   9.776743
## [12,]  23.334084
## [13,]  33.312981
## [14,]  32.323647
## [15,]  30.531435
## [16,]  31.949614
## [17,]  28.658669
## [18,]  16.434985
## [19,]  26.013631
## [20,]  30.682964
```

Repeat this process of randomly splitting into training and testing sets 1000 times. Each time, calculate the mean absolute error of the resulting predictions, MAE $= \frac{1}{20}\sum_{i=1}^{20} |y_i^{test} - y_i^{pred}|$ What is the mean and standard deviation of the MAE for these 1000 tests?

```
# Capture OLS testing/training in function for repeated use
#  This time it returns the mean absolute error instead of w
fitandpredict <- function(X,Y){
  testset <- sample(nrow(X), 20)

  Xtest <- X[testset,]
  Ytest <- Y[testset,]

  Xtrain <- X[-testset,]
  Ytrain <- Y[-testset,]

  XT <- t(Xtrain)

  w <- (solve(XT %*% Xtrain) %*% XT) %*% Ytrain

  Ypred <- Xtest %*% w
  error <- Ytest - Ypred

  return(mean(abs(error)))
}

# Test 1000 times
MAEs <- replicate(1000,fitandpredict(X,Y))

# Compute mean and standard deviation of the 1000 MAEs generated
Results <- c(mean(MAEs),sd(MAEs))
as.data.frame(Results,row.names=c("Mean","Standard Deviation"))
```

```
##                      Results
## Mean               2.6817312
## Standard Deviation 0.4800622
```

Part 2 Using exactly the same training/testing setup as in *Part 1*, fit a p-th order polynomial regression model using least squares for $p = 1, 2, 3, 4$. (Note that $p = 1$ is equivalent to *Part 1*.) For each value of p run 1000 experiments on randomly partitioned training/testing sets using 20 testing and 372 training examples. For each experiment calculate the root mean squared error,

RMSE $= \sqrt{\frac{1}{20}\sum_{i=1}^{20}(y_i^{test} - y_i^{pred})^2}$

```
# Add pth order polynomial features to X
for(i in 2:4){
  X <- cbind(X,X[,2:7]^i)
}

# New function first omits higher order polynomial features for p < 4
#  Additionally, it now returns Root Mean Squared Error (instead of MAE) for each run
polyfitandpredict <- function(X,Y,p){

  ind <- 1 + 6*p
  X <- X[,1:ind]

  testset <- sample(nrow(X), 20)
```

```r
  Xtest <- X[testset,]
  Ytest <- Y[testset,]

  Xtrain <- X[-testset,]
  Ytrain <- Y[-testset,]

  XT <- t(Xtrain)

  w <- (solve(XT %*% Xtrain) %*% XT) %*% Ytrain

  Ypred <- Xtest %*% w
  error <- Ytest - Ypred

  return(sqrt(mean(error^2)))
}

RMSEs <- NULL
# Run 1000 times for each of p = 1,2,3,4, and store results in a matrix
for (i in 1:4) {
    RMSEs <- cbind(RMSEs,replicate(1000,polyfitandpredict(X,Y,i), simplify = "matrix"))
}
```

(a) In a table, print the mean and standard deviation of the RMSE as a function of p. Using these numbers argue for which value of p is the best.

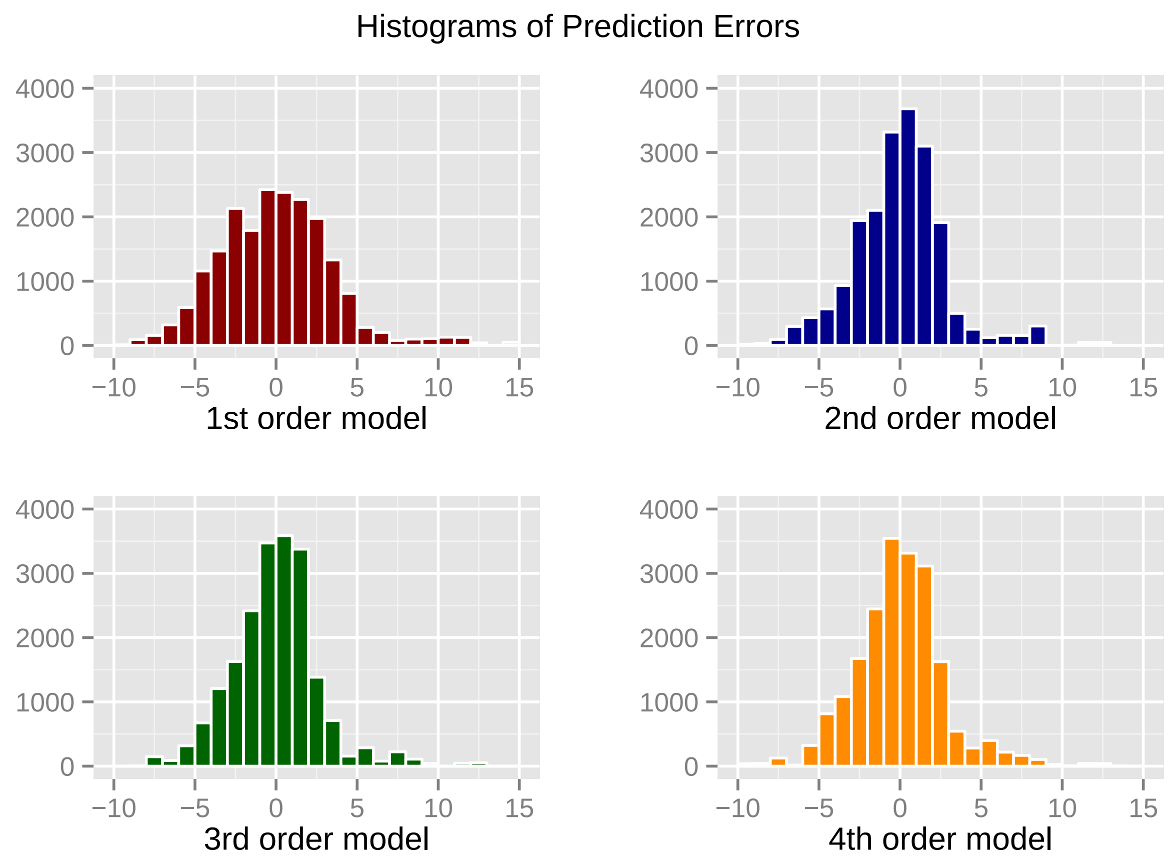
```r
colnames(RMSEs) <- c("p=1","p=2","p=3","p=4")
as.data.frame(rbind(colMeans(RMSEs),
                    apply(RMSEs,2,sd)),
              row.names=c("Mean","S.D."))
```

```
##            p=1       p=2       p=3       p=4
## Mean 3.4436903 2.7826294 2.6455919 2.7068384
## S.D. 0.6872588 0.6509125 0.6040772 0.6142272
```

The 3rd order polynomial model has the lowest mean RMSE and lowest standard deviation in the RMSE, however they are relatively close to the values for 2nd and 4th order polynomials. If our goal is only to get the lowest RMSE, the third order polynomial model is the best option, however, if we are also interested in minimizing model complexity, the 2nd order polynomial model may suffice.

(b) For each value of p, collect $y^{test} - y^{pred}$ for each test example. (Observe that this number can be negative, and there are 20×1000 in total.) Plot a histogram of these errors for each p.

```r
# Loading libraries necessary for plotting the histograms
library(ggplot2)
library(grid)
library(gridExtra)

# Updated function returns list of prediction errors
polyfitandpredictb <- function(X,Y,p){

  ind <- 1 + 6*p
  X <- X[,1:ind]
  testset <- sample(nrow(X), 20)

  Xtest <- X[testset,]
  Ytest <- Y[testset,]

  Xtrain <- X[-testset,]
  Ytrain <- Y[-testset,]

  XT <- t(Xtrain)

  w <- (solve(XT %*% Xtrain) %*% XT) %*% Ytrain

  Ypred <- Xtest %*% w
  error <- Ytest - Ypred

  return(error)
}

# Test the model 1000 times for each value of p = 1,...,4
# Collect the 20 errors for each of the 4000 runs
errors <- NULL
for (i in 1:4) {
  errors <- cbind(errors,as.vector(replicate(1000,polyfitandpredictb(X,Y,i),
                                          simplify = "matrix")))
}

# Aesthetics for the histograms
binwidth = 1
outline = "white"
ylab = ""
xlab = c("1st","2nd","3rd","4th")
xlim = c(-10,15)
ylim = c(0,4000)
fill = c("darkred","darkblue","darkgreen","darkorange")
errors.df <- as.data.frame(errors)

# Build histograms of prediction errors in ggplot
for (i in 1:4){
  assign(paste0("g",i),
       ggplot(data=errors.df,
              aes_string(names(errors.df)[i])) +
        geom_histogram(binwidth = binwidth,
                       color=outline,
                       fill=fill[i]) +
        xlab(paste0(xlab[i]," order model")) +
        ylab(ylab) +
        xlim(xlim) +
        ylim(ylim)
       )
}

# Display histograms in 2x2 frames
grid.arrange(g1,g2,g3,g4,nrow=2, ncol=2, main="Histograms of Prediction Errors")
```



(c) For each p, use maximum likelihood to fit a univariate Gaussian to the 20,000 errors from *Part 2(b)*. Describe how you calculated the maximum likelihood values for the mean and variance (this is a univariate case of what we did in class, so no need to re-derive it). What is the log likelihood of these empirical errors using the maximum likelihood values for the mean and variance? Show this as a function of p and discuss how this agrees/disagrees with your conclusion in *Part 2(a)*. What assumptions are best satisfied by the optimal value of p using this approach?

As discussed in class, the maximum likelihood estimates for the mean and variance of these (hypothetically) normally distributed errors are the empirical mean and variance from the sample data.

Log likelihood for normal distribution:

$\ln \mathcal{L}(\mu, \sigma^2|x) = \ln \prod \frac{1}{\sqrt{2\sigma^2\pi}} \exp\{-\frac{x_i-\mu}{2\sigma^2}\}$

$= \sum \ln \frac{1}{\sqrt{2\sigma^2\pi}} - \frac{x_i-\mu}{2\sigma^2}$

$= \ln\{2\sigma^2\pi\}^{-\frac{n}{2}} - \frac{\sum x_i-\mu}{2\sigma^2}$

$= -\frac{n}{2} \ln \sqrt{2\sigma^2\pi} - \frac{\sum x_i-\mu}{2\sigma^2}$

```r
library(plyr)

#  I am defining a "MLEvar" function because R's built-in "var" function
#    uses the unbiased estimator with n-1 in the denominator.
#    The MLE variance should have n in the denominator instead

MLEvar <- function(x){
  mean <- mean(x)
  n <- length(x)
  var <- sum((x - mean)^2)/n
  return(var)
}

#  As discussed in class, the Maximum Likelihood Estimates for mean and variance
#   are the empirical mean and variance of the sample.
setNames(as.data.frame(rbind(colMeans(errors),colwise(MLEvar)(errors.df)),
                       row.names=c("Mean","Variance")),
         c("p=1","p=2","p=3","p=4"))
```

```
##                   p=1         p=2         p=3         p=4
## Mean     -0.004685523 -0.01422569 -0.01434787 0.008616707
## Variance 12.005684533  7.92595395  7.25572508 7.660513579
```

```
# Computes log likelihood of the parameters given the data
#  using the function derived above
normLogLikelihood <- function(x){
  mean <- mean(x)
  var <- MLEvar(x)
  n <- length(x)
  return(((-n/2)*log(var*2*pi))-
         (sum((x-mean)^2)/(2*var)))
}
# Table of log likelihoods for the four models
setNames(colwise(normLogLikelihood)(errors.df),c("p=1","p=2","p=3","p=4"))
```

```
##         p=1      p=2       p=3       p=4
## 1 -53232.57 -49080.2 -48196.68 -48739.56
```

The third order polynomial model has the highest log likelihood. This suggests that, out of the four models, its errors are the closest to normally distributed. This aligns nicely with our finding in *Part 2(a)* that this model also has the smallest root mean squared error. Thus using $p = 3$ best represents the assumption of ordinary least squares linear regression that the error terms are normally distributed about a mean of zero.