Problem Set #3

Ian Bach

2024-09-07

1. (10 points) This question involves the Weekly dataset, which is included in the package ISLR2.

(a) Fit a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary() to print the results.
(b) For which of the predictors can you reject the null hypothesis H0 : j = 0?
(c) Compute the confusion matrix and overall fraction of correct predictions.
(d) Repeat (c) using LDA. Which method performs better on this data?

```
# Load necessary libraries
library(ISLR2)  # For the Weekly dataset
```

```
## Warning: package 'ISLR2' was built under R version 4.4.1
```

```
library(MASS)   # For LDA (Linear Discriminant Analysis)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:ISLR2':
##
##     Boston
```

```
# Inspect the Weekly dataset
head(Weekly)
```

```
##   Year   Lag1   Lag2   Lag3   Lag4   Lag5    Volume  Today Direction
## 1 1990  0.816  1.572 -3.936 -0.229 -3.484 0.1549760 -0.270      Down
## 2 1990 -0.270  0.816  1.572 -3.936 -0.229 0.1485740 -2.576      Down
## 3 1990 -2.576 -0.270  0.816  1.572 -3.936 0.1598375  3.514        Up
## 4 1990  3.514 -2.576 -0.270  0.816  1.572 0.1616300  0.712        Up
## 5 1990  0.712  3.514 -2.576 -0.270  0.816 0.1537280  1.178        Up
## 6 1990  1.178  0.712  3.514 -2.576 -0.270 0.1544440 -1.372      Down
```

```
# (1a) Fit a logistic regression model
logistic_model <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
                      data = Weekly, family = binomial)

# Print the summary of the logistic model
cat("(1a) Summary of Logistic Regression Model:\n")
```

```
## (1a) Summary of Logistic Regression Model:
```

```r
summary(logistic_model)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = Weekly)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

```r
# (1b) Identify predictors for which the null hypothesis (H0:  j = 0) can be rejected
# Based on the p-values from the logistic regression summary

cat("(1b) Predictors for which we can reject H0 ( j = 0):\n")
```

```
## (1b) Predictors for which we can reject H0 ( j = 0):
```

```r
# Extract p-values from the model summary and display significant ones
logistic_summary <- summary(logistic_model)
p_values <- coef(logistic_summary)[, 4]  # p-values are in the 4th column

# Print only the significant predictors (p-value < 0.05)
significant_predictors <- names(p_values[p_values < 0.05])
print(significant_predictors)
```

```
## [1] "(Intercept)" "Lag2"
```

```r
# (1c) Compute confusion matrix and overall fraction of correct predictions for Logistic Regression

# Predict the probabilities using logistic regression
predicted_probs <- predict(logistic_model, type = "response")
```

```r
# Convert probabilities to class labels (threshold of 0.5)
predicted_classes <- ifelse(predicted_probs > 0.5, "Up", "Down")

# Create confusion matrix for logistic regression
confusion_matrix <- table(Predicted = predicted_classes, Actual = Weekly$Direction)

cat("(1c) Confusion Matrix for Logistic Regression:\n")
```

```
## (1c) Confusion Matrix for Logistic Regression:
```

```r
print(confusion_matrix)
```

```
##          Actual
## Predicted Down  Up
##      Down   54  48
##      Up    430 557
```

```r
# Calculate the overall fraction of correct predictions (accuracy)
logistic_accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Logistic Regression Accuracy: ", logistic_accuracy, "\n")
```

```
## Logistic Regression Accuracy:  0.5610652
```

```r
# (1d) Perform Linear Discriminant Analysis (LDA) and repeat confusion matrix and accuracy calculation

# Fit the LDA model
lda_model <- lda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly)

# Make predictions using LDA
lda_predictions <- predict(lda_model)

# Create confusion matrix for LDA
lda_confusion_matrix <- table(Predicted = lda_predictions$class, Actual = Weekly$Direction)

cat("(1d) Confusion Matrix for LDA:\n")
```

```
## (1d) Confusion Matrix for LDA:
```

```r
print(lda_confusion_matrix)
```

```
##          Actual
## Predicted Down  Up
##      Down   52  46
##      Up    432 559
```

```r
# Calculate the overall fraction of correct predictions for LDA (accuracy)
lda_accuracy <- sum(diag(lda_confusion_matrix)) / sum(lda_confusion_matrix)
cat("LDA Accuracy: ", lda_accuracy, "\n")
```

```
## LDA Accuracy:  0.5610652
```

```r
# Compare the accuracy of Logistic Regression vs LDA
if (logistic_accuracy > lda_accuracy) {
    cat("Logistic Regression performs better with an accuracy of", logistic_accuracy, "\n")
} else {
    cat("LDA performs better with an accuracy of", lda_accuracy, "\n")
}
```

```
## LDA performs better with an accuracy of 0.5610652
```

Question #2

```r
# Load necessary libraries
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.4.1
```

```r
# Load the dataset
setwd("C:/Users/ibach/OneDrive - Terillium/Desktop/Purdue MSBA/Machine Learning/HW 3")
data <- read.csv("logit_data.csv")
y <- data$grade
x <- data$hours
```

2A):

Log-Likelihood Function

We take the natural logarithm of the likelihood function:

$$\ell(\beta \mid y) = \log L(\beta \mid y) = \sum_{i=1}^{n} \left(y_i \log(p(x_i;\beta)) + (1-y_i)\log(1-p(x_i;\beta))\right)$$

Now, substitute the expansion for $p(x_i;\beta)$ and $1-p(x_i;\beta)$:

$$\ell(\beta \mid y) = \sum_{i=1}^{n} \left(y_i \log\left(\frac{1+\exp(\beta_0+\beta_1 x_i)}{\exp(\beta_0+\beta_1 x_i)}\right) + (1-y_i)\log\left(\frac{1}{1+\exp(\beta_0+\beta_1 x_i)}\right)\right)$$

Simplified, the log-likelihood function becomes:

$$\ell(\beta \mid y) = \sum_{i=1}^{n} \left(y_i(\beta_0+\beta_1 x_i) - \log(1+\exp(\beta_0+\beta_1 x_i))\right)$$

Thus, the full log-likelihood function is written as:

$$\ell(\beta \mid y) = -\sum_{i=1}^{n} \log(1+\exp(\beta_0+\beta_1 x_i)) + \sum_{i=1}^{n} y_i(\beta_0+\beta_1 x_i)$$

```r
# (2a) Log-likelihood function
log_likelihood <- function(beta, y, x) {
  beta0 <- beta[1]
  beta1 <- beta[2]
  sum(-log(1 + exp(beta0 + beta1 * x)) + y * (beta0 + beta1 * x))
}
```

2B):

$$\ell(\beta \mid y) = \log L(\beta \mid y) = \sum_{i=1}^{n} (y_i \log(p(x_i;\beta)) + (1-y_i)\log(1-p(x_i;\beta)))$$

Now, substitute the expansion for $p(x_i;\beta)$ and $1-p(x_i;\beta)$:

$$\ell(\beta \mid y) = \sum_{i=1}^{n} \left(y_i \log\left(\frac{1+\exp(\beta_0+\beta_1 x_i)}{\exp(\beta_0+\beta_1 x_i)}\right) + (1-y_i)\log\left(\frac{1}{1+\exp(\beta_0+\beta_1 x_i)}\right)\right)$$

Simplified, the log-likelihood function becomes:

$$\ell(\beta \mid y) = \sum_{i=1}^{n} (y_i(\beta_0+\beta_1 x_i) - \log(1+\exp(\beta_0+\beta_1 x_i)))$$

Thus, the full log-likelihood function is written as:

$$\ell(\beta \mid y) = -\sum_{i=1}^{n} \log(1+\exp(\beta_0+\beta_1 x_i)) + \sum_{i=1}^{n} y_i(\beta_0+\beta_1 x_i)$$

```r
# (2b) Gradient of the log-likelihood
gradient <- function(beta, y, x) {
  beta0 <- beta[1]
  beta1 <- beta[2]
  p <- 1 / (1 + exp(-(beta0 + beta1 * x)))
  grad_beta0 <- sum(y - p)
  grad_beta1 <- sum((y - p) * x)
  return(c(grad_beta0, grad_beta1))
}
```

2C):

```r
# (2c) Maximum likelihood estimates using glm
model <- glm(grade ~ hours, family = binomial(link = "logit"), data = data)
summary(model)
```

```
##
## Call:
## glm(formula = grade ~ hours, family = binomial(link = "logit"),
##     data = data)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept) -3.09053    0.36962  -8.361   <2e-16 ***
## hours        0.77508    0.07806   9.930   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 566.69  on 499  degrees of freedom
## Residual deviance: 236.73  on 498  degrees of freedom
## AIC: 240.73
##
## Number of Fisher Scoring iterations: 7
```

2D):

```r
# (2d) Gradient ascent
gradient_ascent <- function(y, x, learning_rate = 0.001, epsilon = 1e-8, max_iter = 10000) {
  beta <- c(0, 0)  # Initial values
  beta_history <- matrix(0, nrow = max_iter, ncol = 2)
  for (i in 1:max_iter) {
    grad <- gradient(beta, y, x)
    beta_new <- beta + learning_rate * grad
    beta_history[i, ] <- beta_new
    if (sqrt(sum((beta_new - beta)^2)) < epsilon) {
      beta_history <- beta_history[1:i, ]
      break
    }
    beta <- beta_new
  }
  return(list(beta = beta, history = beta_history))
}

# Run gradient ascent
result <- gradient_ascent(y, x)
beta_estimates <- result$beta
beta_history <- result$history
```

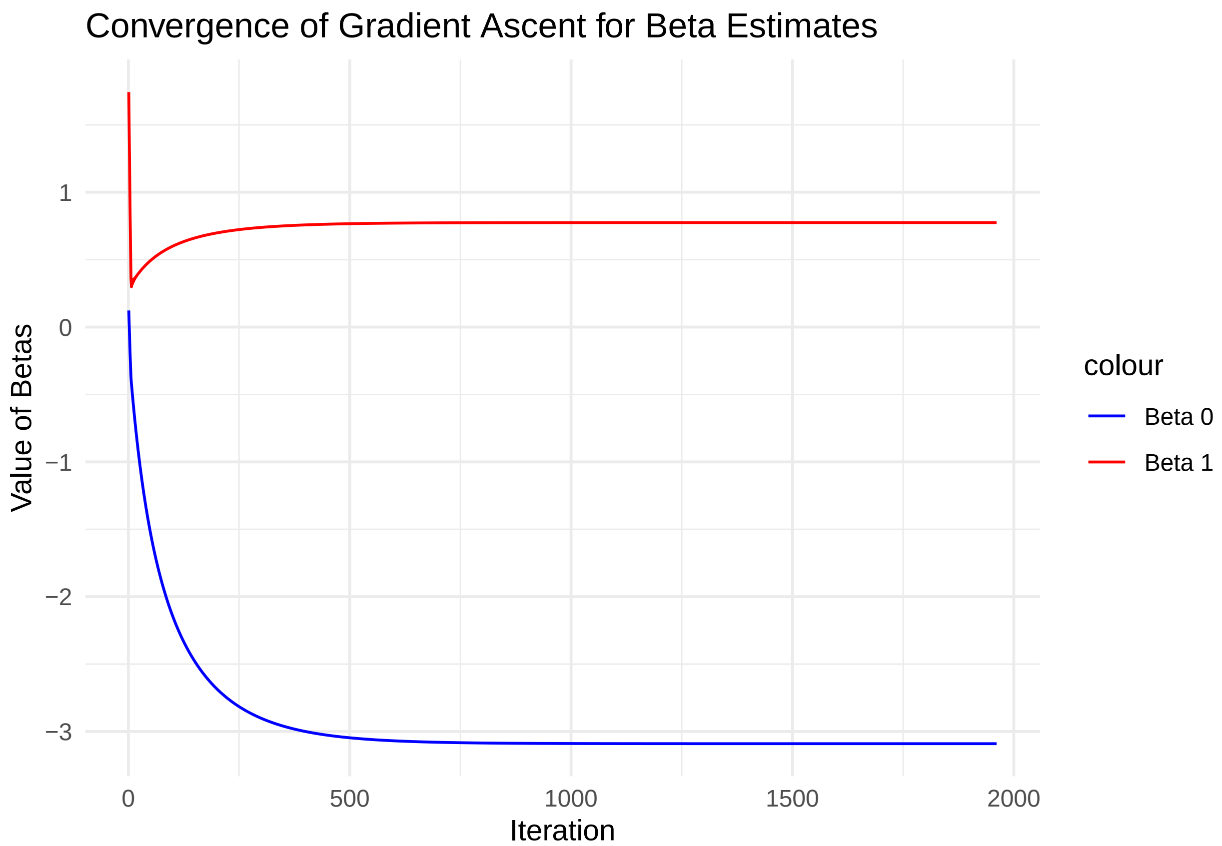
2E):

```r
# (2e) Predict probability of not getting an A for x = 10
x_new <- 10
p_getting_A <- 1 / (1 + exp(-(beta_estimates[1] + beta_estimates[2] * x_new)))
p_not_getting_A <- 1 - p_getting_A

# Scatter plot of beta values
beta_df <- data.frame(iteration = 1:nrow(beta_history),
                      beta0 = beta_history[, 1],
                      beta1 = beta_history[, 2])

ggplot(beta_df, aes(x = iteration)) +
  geom_line(aes(y = beta0, color = "Beta 0")) +
  geom_line(aes(y = beta1, color = "Beta 1")) +
  labs(title = "Convergence of Gradient Ascent for Beta Estimates",
```

```r
       x = "Iteration", y = "Value of Betas") +
  scale_color_manual(values = c("Beta 0" = "blue", "Beta 1" = "red")) +
  theme_minimal()
```




```r
# Output the probability that the student will not get an A
p_not_getting_A
```

```
## [1] 0.009374841
```