Homework #3

Spencer Pease

2/22/2021

Question 1

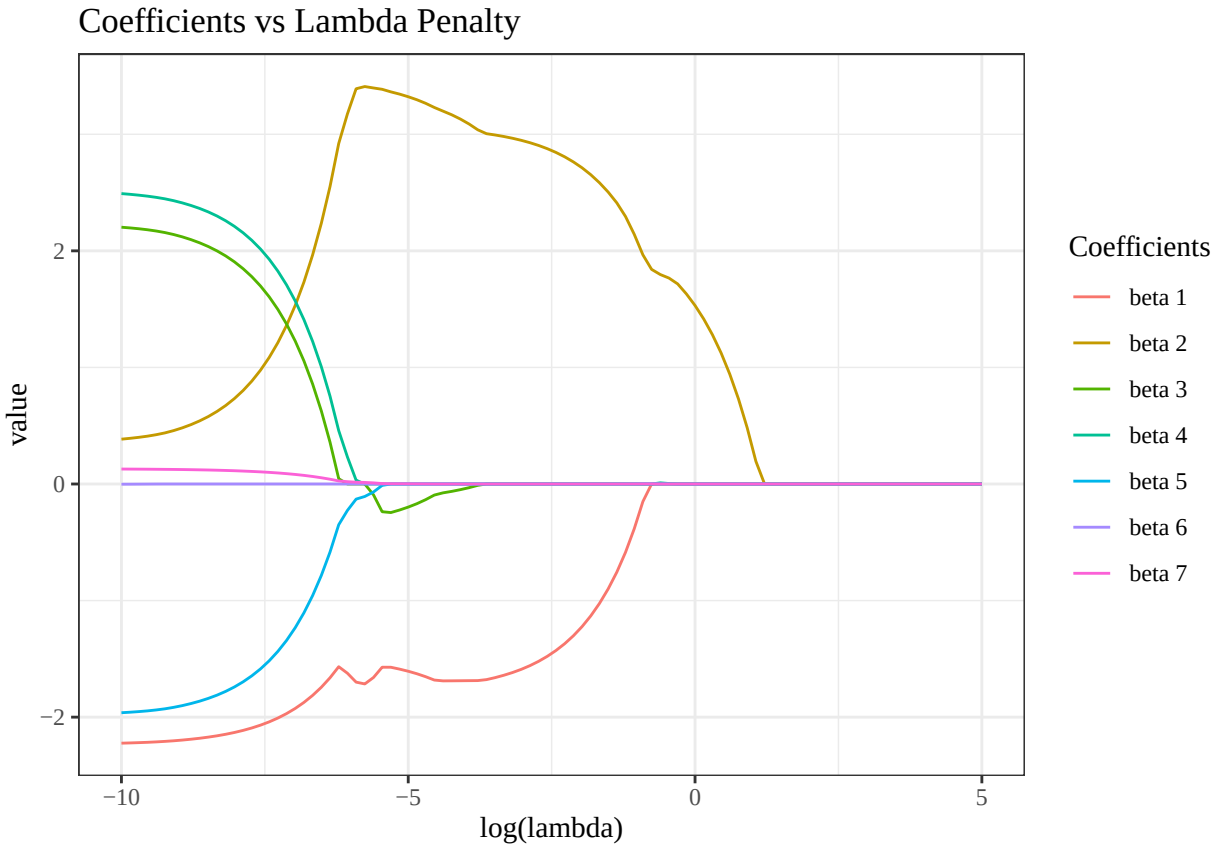
Part (a, b)

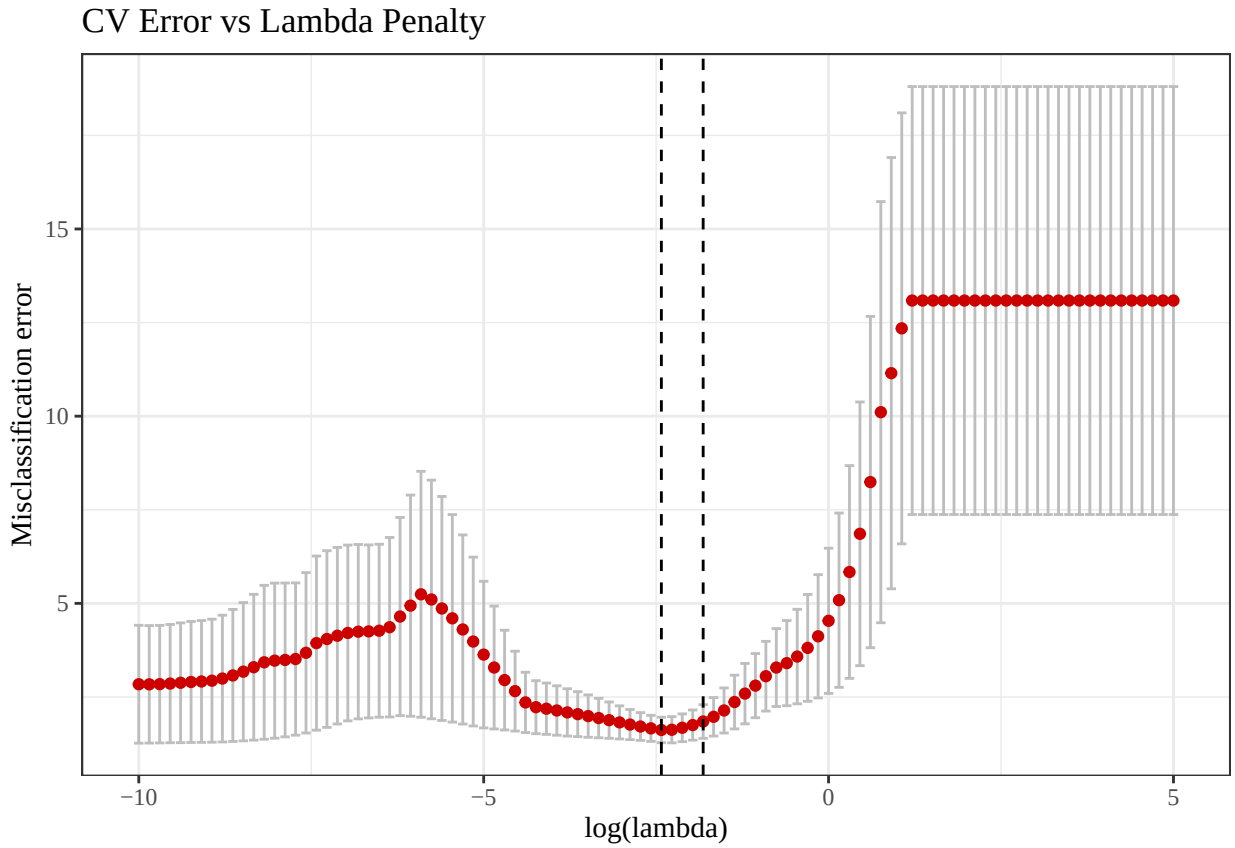
X and ϵ are each 30 random samples from the standard normal distribution ($\mathcal{N}(\mu = 0, \sigma = 1)$) used to construct the response:

$$Y = 3 - 2X + 3X^2 + \epsilon$$

Part (c)

A lasso model of the form $Y \sim \text{poly}(X, 7)$ is then fit to the generated data X and Y.





Performing cross-validation shows that a lambda value of 0.089 minimizes the misclassification error, making it the optimal value for the tuning parameter.

Table 1: Optimal model coefficients

X^1	X^2	X^3	X^4	X^5	X^6	X^7
-1.428	2.844	0	0	0	0	0

Looking at the coefficients of the "optimal" model, we see that the lasso model is able to mostly recover the original response vector, selecting only the first and second degree terms with coefficients similar in direction and value to the actual coefficients.

Part (d)

Applying the optimal model to a set of 1000 new observations generated in the same manner yields a prediction mean squared error of 1.644.

Question 2

Part(a)

Table 2: Observations by diagnosis class

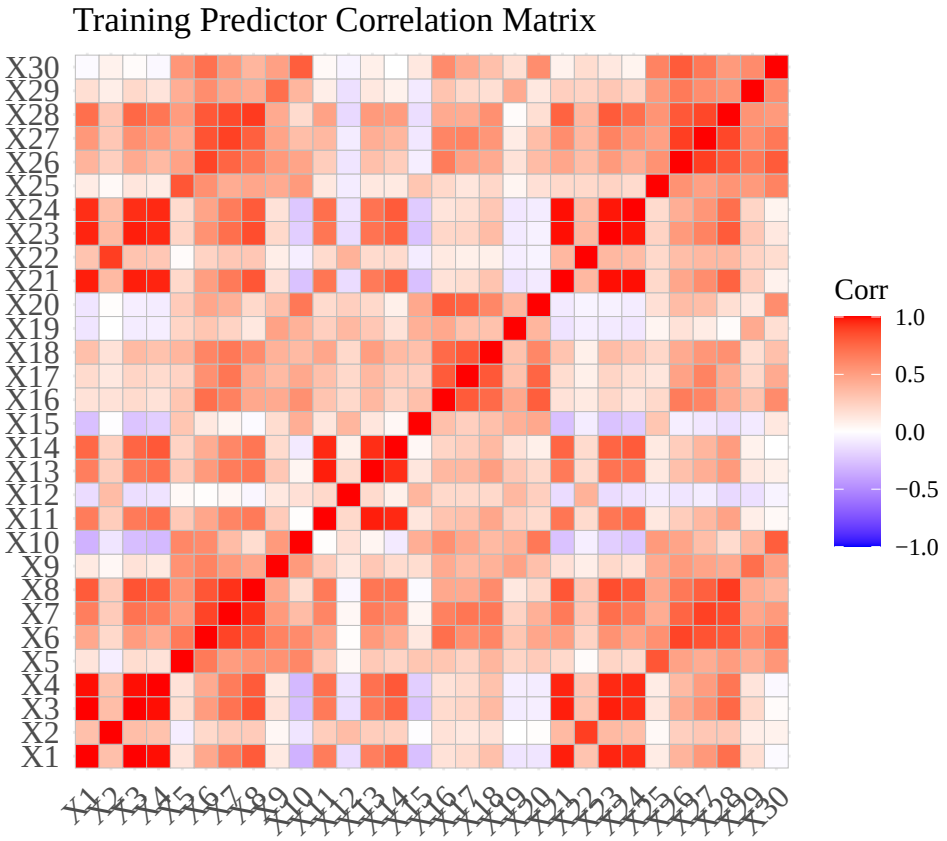
diagnosis	observations
benign	357
malignant	212

The *wdbc* dataset has a sample size n of 569 and 30 predictors p. *Table 2* shows a summary of the number of observations in each class.

Part (b, c)

This dataset is randomly split into a training set of 400 observations and a test set of 169 observations. Each split is then normalized separately to prevent observations in one dataset from influencing the distribution characteristics of the other dataset. Put another way, we don't want any information from the test set affect how the training data is normalized, because then the test data will not be completely new to any model fit on the training data.

Part (d)



The correlation matrix shows some predictor pairs with near perfect correlation. This introduces collinearity to models including these predictor pairs, which can reduce the stability of other predictors' estimated coefficients.

Part (e)

Table 3: Simple logistic regression training coefficients

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	92.46	28281.57	0.00	1.00
X1	-863.00	462270.50	0.00	1.00
X2	-3.80	13416.45	0.00	1.00
X3	989.13	544553.11	0.00	1.00
X4	-189.77	100538.36	0.00	1.00
X5	117.78	15574.74	0.01	0.99
X6	-302.08	41791.18	-0.01	0.99
X7	-8.09	39076.94	0.00	1.00
X8	265.44	49259.17	0.01	1.00
X9	-122.56	22339.35	-0.01	1.00
X10	96.25	28237.70	0.00	1.00
X11	559.97	61288.54	0.01	0.99
X12	3.35	9628.51	0.00	1.00
X13	-110.48	70063.53	0.00	1.00
X14	-540.24	107218.22	-0.01	1.00
X15	31.71	12943.54	0.00	1.00
X16	73.75	13649.61	0.01	1.00
X17	52.34	57193.23	0.00	1.00
X18	90.04	20354.30	0.00	1.00
X19	-93.14	24992.86	0.00	1.00
X20	-262.42	65894.49	0.00	1.00
X21	-1030.88	195279.90	-0.01	1.00
X22	105.94	15834.64	0.01	0.99
X23	-502.59	175318.35	0.00	1.00
X24	2156.15	279421.09	0.01	0.99
X25	-97.37	20531.90	0.00	1.00
X26	5.47	30882.30	0.00	1.00
X27	106.24	52714.25	0.00	1.00
X28	162.49	25575.21	0.01	0.99
X29	241.66	25846.59	0.01	0.99
X30	-39.36	62867.79	0.00	1.00

The correlation between X_1 and X_3 in the simple logistic model is -0.977.

$\hat{\beta}_1$ and $\hat{\beta}_3$ are nearly equal and opposite of each other, with magnitudes greater than the coefficients of other predictors that aren't colinear with another. This confirms that the inclusion of colinear predictors leads to a less stable model with inflated effects of single predictors.

Part (f)

Table 4: Confusion table of predicted outcomes for training data

	pred. benign	pred. malignant
benign	255	0
malignant	0	145

Table 5: Confusion table of predicted outcomes for test data

	pred. benign	pred. malignant
benign	99	3
malignant	7	60

- Training accuracy: 1
- Test accuracy: 0.9408

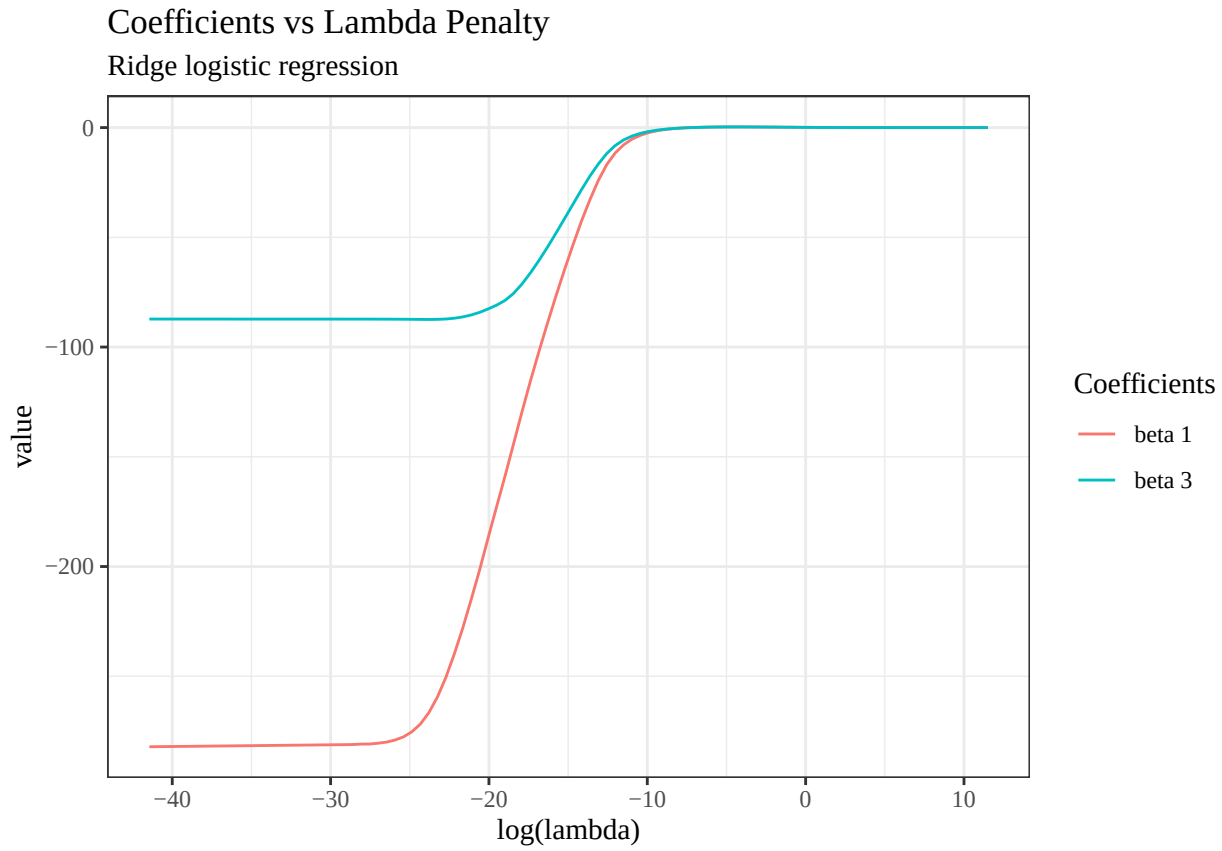
No misclassifications in the training data shows that the model is likely overfit to that data. Overfitting has the effect of reducing test accuracy, which is also observed in this case.

Question 3

Part (a)

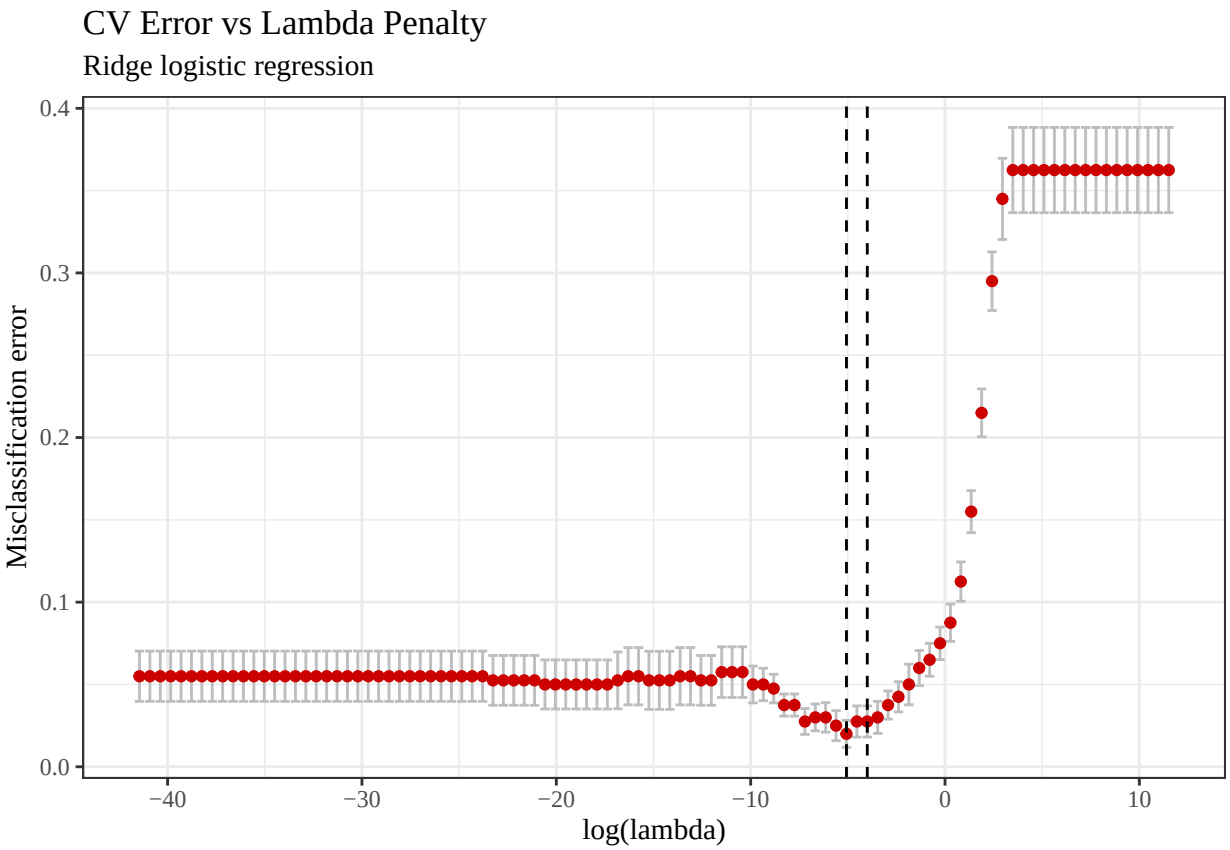
The normalized training and test data is converted into matrices for use in `glmnet` models.

Part (b, c)



Under a ridge logistic regression,the coefficient β_1 experiences a more extreme penalization as λ increases, with β_1 and β_3 converging to near-zero around $\log(\lambda) = -10$.

Part (d)



Optimal λ: 0.006

Part (e)

Using the optimal value of λ, there are 30 non-zero model coefficients. This is expected for a ridge regression model, which shrinks coefficients towards zero as λ increases, but only requires them to be zero when $\lambda = \infty$.

Part (f)

Table 6: Confusion table of predicted outcomes for training data

	pred. benign	pred. malignant
benign	254	1
malignant	3	142

Table 7: Confusion table of predicted outcomes for test data

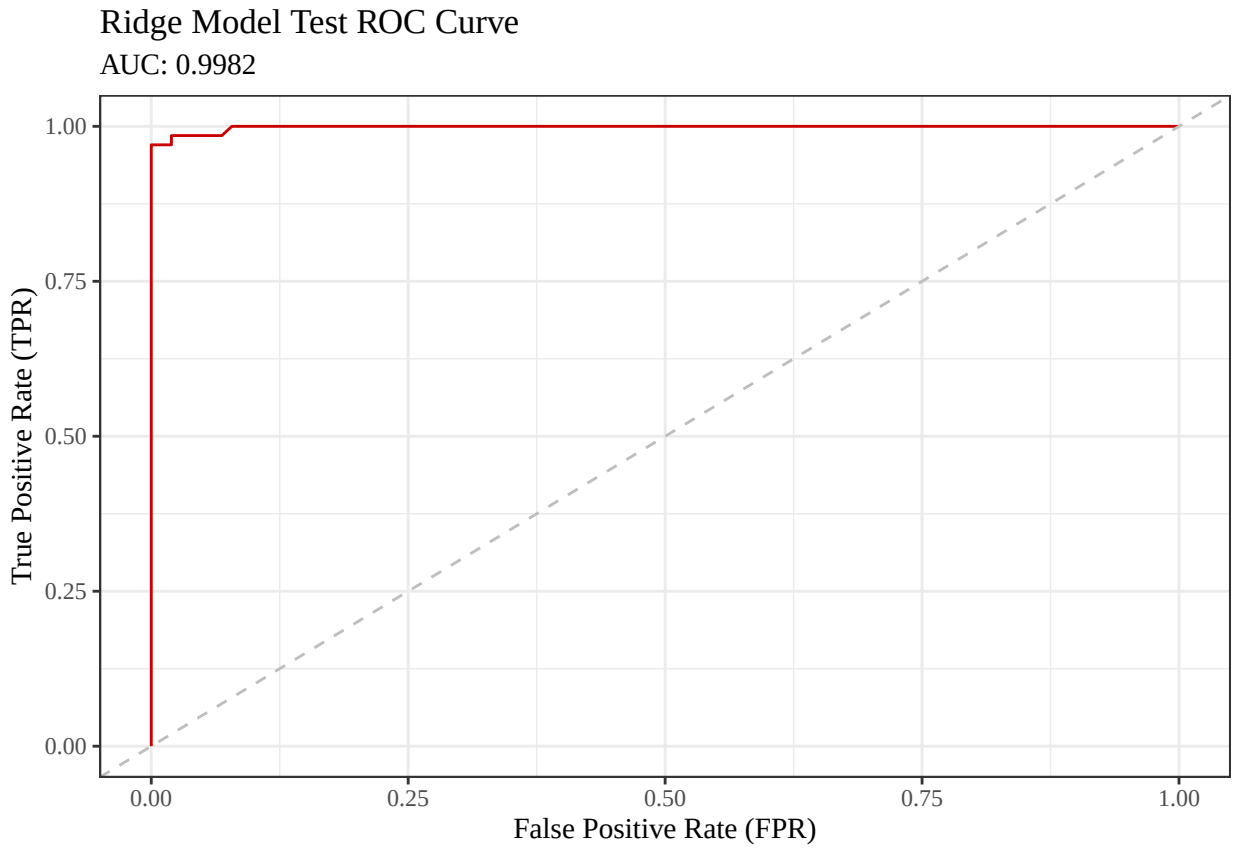
	pred. benign	pred. malignant
benign	102	0

	pred. benign	pred. malignant
malignant	2	65

- Training accuracy: 0.99
- Test accuracy: 0.9882

The training accuracy is slightly higher than the test accuracy, which is expected. The similarity between the two accuracies suggests that the model is not overfit to the training data, or the test data has a similar distribution to the training data.

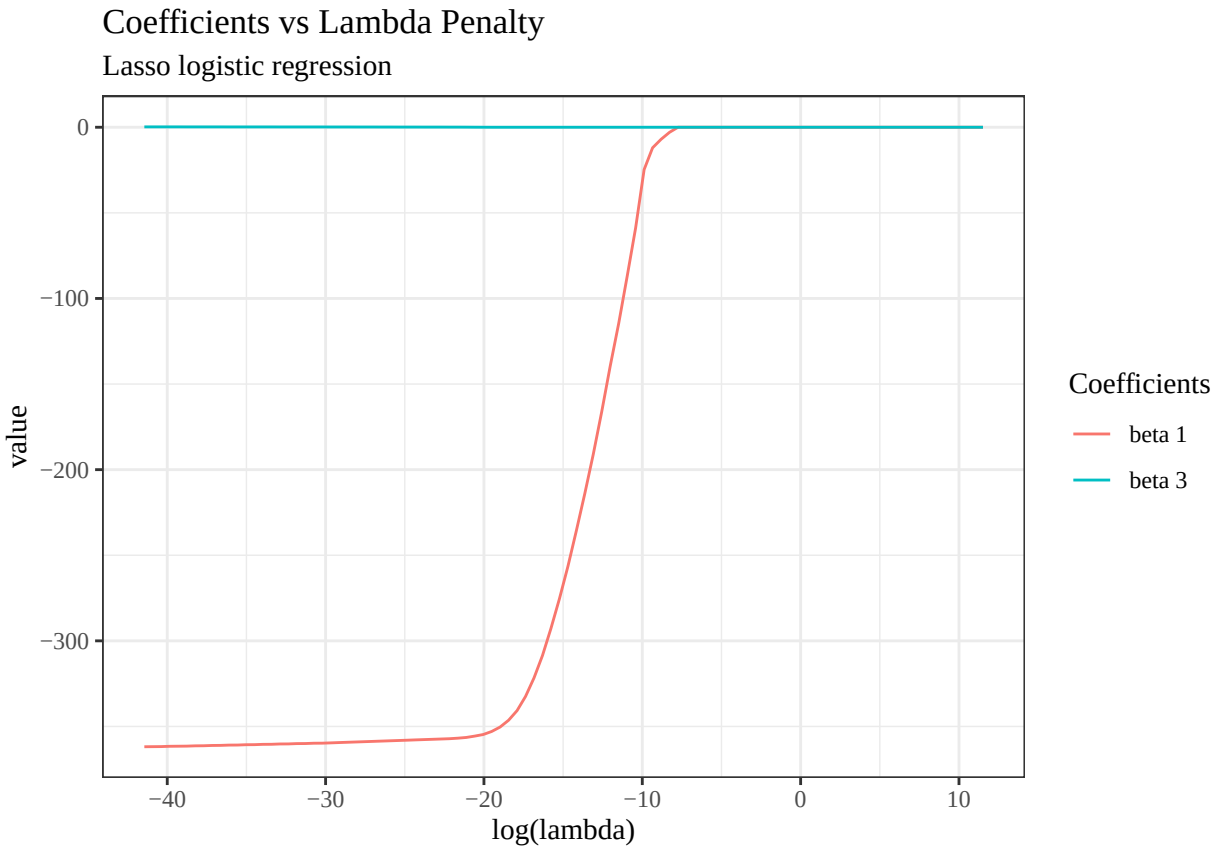
Part (g, h)



The area under the ROC curve is 0.998.

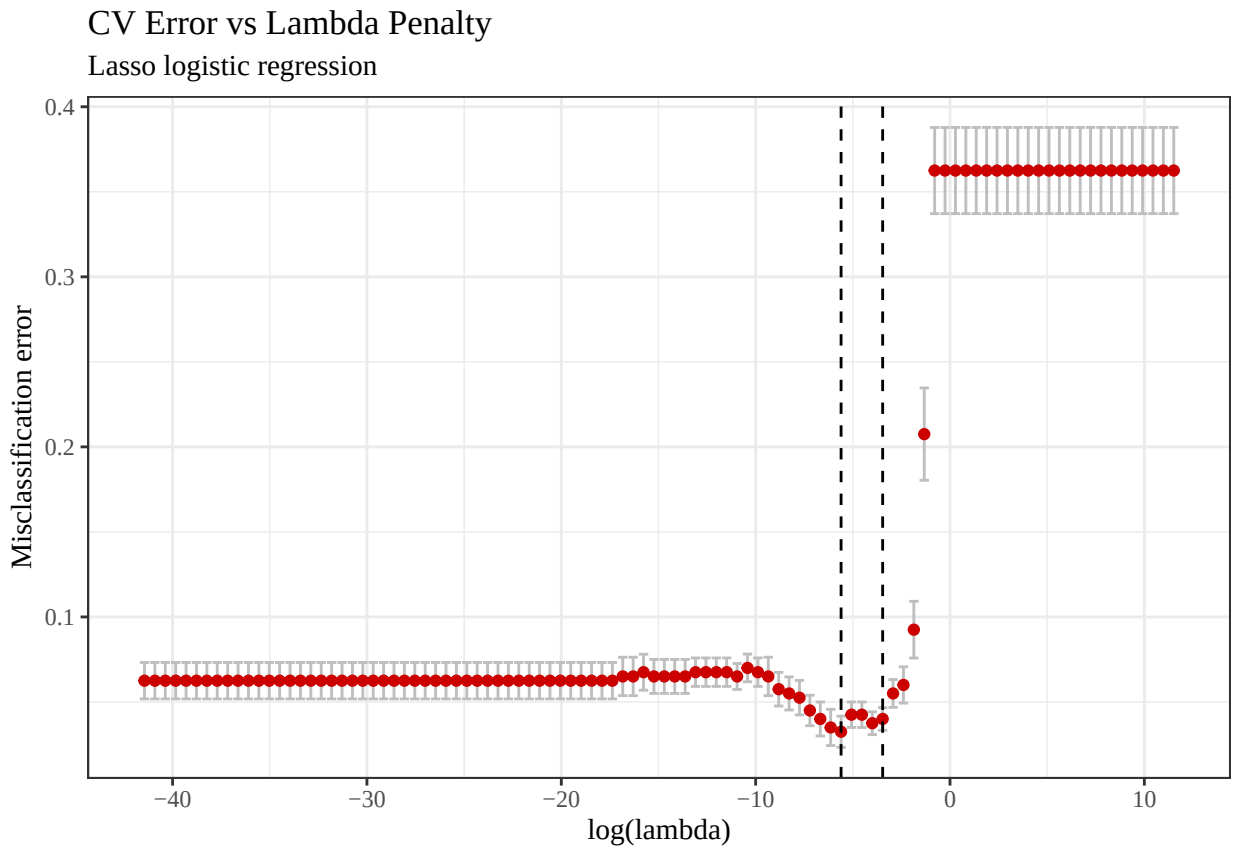
Question 4

Part (b, c)



Under a lasso logistic regression,the coefficient β_3 is reduced to zero for all values of λ, while begins to sharply decrease in magnitude around $\log(\lambda) = -20$, reaching zero around $\log(\lambda) = -7$.

Part (d)



Optimal λ: 0.004

Part (e)

Using the optimal value of λ, there are 15 non-zero model coefficients. This is expected for a lasso regression model, which will penalize coefficients to zero for intermediate values of λ.

Part (f)

Table 8: Confusion table of predicted outcomes for training data

	pred. benign	pred. malignant
benign	254	1
malignant	4	141

Table 9: Confusion table of predicted outcomes for test data

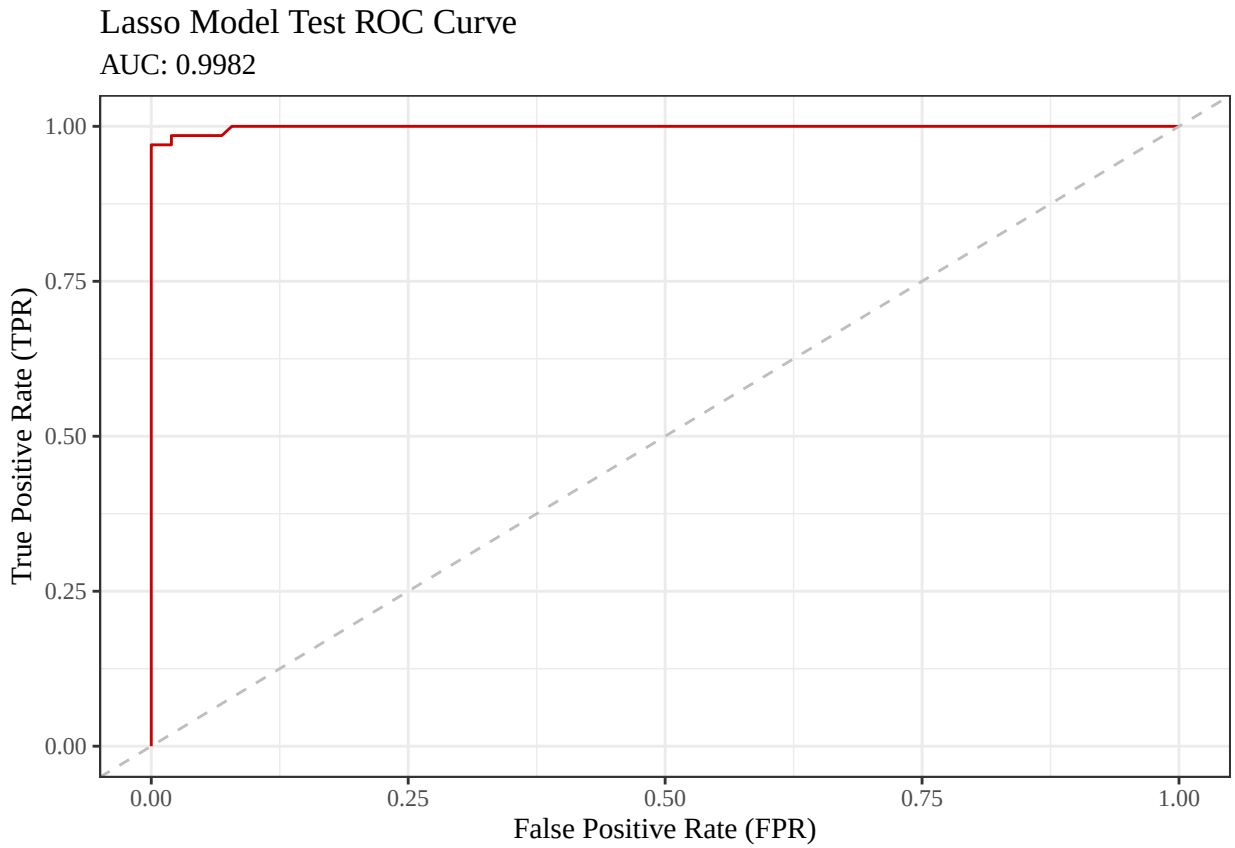
	pred. benign	pred. malignant
benign	102	0

	pred. benign	pred. malignant
malignant	2	65

- Training accuracy: 0.9875
- Test accuracy: 0.9882

The test accuracy is slightly higher than the training accuracy, which is likely an artifact of the random data sampling. Considering them to be nearly equal shows that fitting a simpler model reduces overfitting and leads to more robust performance on unobserved data.

Part (g, h)



The area under the ROC curve is 0.998.

Question 5

Table 10: Summary of test model performance

Model	Accuracy	TPR	FPR	Non-zero coefs.
simple	0.9408	0.8955	0.0294	30
ridge	0.9882	0.9701	0.0000	30
lasso	0.9882	0.9701	0.0000	15

The simple logistic model performs worse than the ridge and lasso models on all performance metrics. While the ridge and lasso models have identical performance, the lasso model has the advantage of being a much simpler model, requiring half the predictors to get the same performance. This simplicity makes the lasso model more interpretable than any of the alternatives.

Appendix

Analysis

```
# Prep work ------------------------------------------------------------------

library(dplyr)
library(ggplot2)
library(glmnet)
library(ggcorrplot)

source("functions/predict_bayes.R")
source("functions/performance_metrics.R")
source("functions/plots.R")
source("functions/calculate_roc.R")

df_wdbc <- readr::read_csv("data/wdbc.data", col_names = FALSE) %>%
  select(2:32) %>%
  setNames(c("diagnosis", paste0("X", 1:30))) %>%
  mutate(
    diagnosis = factor(
      diagnosis, levels = c("B", "M"), labels = c("benign", "malignant")
    )
  )


set.seed(123456)


# Question 1a,b --------------------------------------------------------------

gen_rnorm_data <- function(n) {

  X <- rnorm(n)
  eps <- rnorm(n)
  Y <- 3 - (2 * X) + (3 * X^2) + eps

  return(list(X = X, Y = Y))

}

data_rnorm <- gen_rnorm_data(30)


# Question 1c ----------------------------------------------------------------

lambda_grid <- exp(seq(5, -10,length = 100))
X_poly7 <- poly(data_rnorm$X, degree = 7, raw = TRUE)

model_rnorm_lasso <-  glmnet(
  X_poly7, data_rnorm$Y,
  alpha = 1,
  lambda = lambda_grid,
```

```
  thresh = 1e-8
)

model_rnorm_lasso_cv <- cv.glmnet(
  X_poly7, data_rnorm$Y,
  alpha = 1,
  lambda = lambda_grid,
  thresh = 1e-8,
  nfolds = 10
)

model_rnorm_lasso_optimal <- glmnet(
  X_poly7, data_rnorm$Y,
  alpha = 1,
  lambda = model_rnorm_lasso_cv$lambda.min,
  thresh = 1e-8
)

coef_rnorm_lasso_optimal <- model_rnorm_lasso_optimal$beta[, 1]

plot_rnorm_lasso <- plot_lambda_coef(model_rnorm_lasso, 1:7)
plot_rnorm_lasso_cv <- plot_cv_error(model_rnorm_lasso_cv)


# Question 1d ------------------------------------------------------------

data_rnorm2 <- gen_rnorm_data(1000)

pred_Y2 <- predict(
  model_rnorm_lasso_optimal,
  poly(data_rnorm2$X, degree = 7, raw = TRUE)
)

mse_Y2 <- mean((data_rnorm2$Y - pred_Y2)^2)


# Question 2a ------------------------------------------------------------

obs_by_class <- df_wdbc %>%
  group_by(diagnosis) %>%
  summarise(observations = n())


# Question 2b,c ----------------------------------------------------------

train_obs <- 400

set.seed(2)

df_shuffled <- df_wdbc[sample(nrow(df_wdbc)), ]

df_train <- df_shuffled %>%
  slice(1:train_obs) %>%
```

```r
  mutate(across(-diagnosis, scale))

df_test <- df_shuffled %>%
  slice(-(1:train_obs)) %>%
  mutate(across(-diagnosis, scale))


# Question 2d ------------------------------------------------------------

cor_wdbc_train <- df_train %>% select(-diagnosis) %>% cor()
plot_cor_wdbc_train <- ggcorrplot(cor_wdbc_train) +
  theme(text = element_text(family = "serif")) +
  labs(title = "Training Predictor Correlation Matrix")


# Question 2e ------------------------------------------------------------

model_wdbc_simple <- glm(diagnosis ~ ., family = "binomial", data = df_train)
model_wdbc_simple_summary <- summary(model_wdbc_simple, correlation = TRUE)


# Question 2f ------------------------------------------------------------

pred_wdbc_simple <- list(
  train = predict_bayes(model_wdbc_simple, df_train),
  test = predict_bayes(model_wdbc_simple, df_test)
)

confusion_wdbc_simple <- list(
  train = confusion_table(df_train$diagnosis, pred_wdbc_simple$train),
  test = confusion_table(df_test$diagnosis, pred_wdbc_simple$test)
)

acc_wdbc_simple <- list(
  train = performance_table(df_train$diagnosis, pred_wdbc_simple$train),
  test = performance_table(df_test$diagnosis, pred_wdbc_simple$test)
)



# Question 3a ------------------------------------------------------------

X_train <- df_train %>% select(-diagnosis) %>% as.matrix()
X_test <- df_test %>% select(-diagnosis) %>% as.matrix()

Y_train <- df_train$diagnosis
Y_test <- df_test$diagnosis


# Question 3b ------------------------------------------------------------

lambda_grid <- 10^seq(5, -18, length = 100)
```

```r
model_wdbc_ridge <- glmnet(
  X_train, Y_train,
  alpha = 0,
  lambda = lambda_grid,
  family = "binomial",
  thresh = 1e-8
)


# Question 3c ------------------------------------------------------------

plot_wdbc_ridge <- plot_lambda_coef(model_wdbc_ridge, c(1, 3)) +
  labs(subtitle = "Ridge logistic regression")


# Question 3d ------------------------------------------------------------

model_wdbc_ridge_cv <- cv.glmnet(
  X_train, Y_train,
  alpha = 0,
  lambda = lambda_grid,
  family = "binomial",
  type.measure = "class",
  nfolds = 10,
  thresh = 1e-8
)

plot_wdbc_ridge_cv <- plot_cv_error(model_wdbc_ridge_cv) +
  labs(subtitle = "Ridge logistic regression")


# Question 3e ------------------------------------------------------------

nzero_wdbc_ridge <- with(model_wdbc_ridge_cv, nzero[index["min", ]])


# Question 3f ------------------------------------------------------------

model_wdbc_ridge_optimal <- glmnet(
  X_train, Y_train,
  alpha = 0,
  lambda = model_wdbc_ridge_cv$lambda.min,
  family = "binomial",
  thresh = 1e-8
)

pred_wdbc_ridge <- list(
  train = predict_bayes(model_wdbc_ridge_optimal, X_train),
  test = predict_bayes(model_wdbc_ridge_optimal, X_test)
)

confusion_wdbc_ridge <- list(
  train = confusion_table(Y_train, pred_wdbc_ridge$train),
```

```
  test = confusion_table(Y_test, pred_wdbc_ridge$test)
)

acc_wdbc_ridge <- list(
  train = performance_table(Y_train, pred_wdbc_ridge$train),
  test = performance_table(Y_test, pred_wdbc_ridge$test)
)


# Question 3g,h ----------------------------------------------------------

roc_wdbc_ridge <- calculate_roc(model_wdbc_ridge_optimal, X_test, Y_test)
auc_wdbc_ridge <- roc_wdbc_ridge$AUC$value


# Question 4b ----------------------------------------------------------

model_wdbc_lasso <- glmnet(
  X_train, Y_train,
  alpha = 1,
  lambda = lambda_grid,
  family = "binomial",
  thresh = 1e-8
)


# Question 4c ----------------------------------------------------------

plot_wdbc_lasso <- plot_lambda_coef(model_wdbc_lasso, c(1, 3)) +
  labs(subtitle = "Lasso logistic regression")


# Question 4d ----------------------------------------------------------

model_wdbc_lasso_cv <- cv.glmnet(
  X_train, Y_train,
  alpha = 1,
  lambda = lambda_grid,
  family = "binomial",
  type.measure = "class",
  nfolds = 10,
  thresh = 1e-8
)

plot_wdbc_lasso_cv <- plot_cv_error(model_wdbc_lasso_cv) +
  labs(subtitle = "Lasso logistic regression")


# Question 4e ----------------------------------------------------------

nzero_wdbc_lasso <- with(model_wdbc_lasso_cv, nzero[index["min", ]])
```

```r
# Question 4f ------------------------------------------------------------

model_wdbc_lasso_optimal <- glmnet(
  X_train, Y_train,
  alpha = 1,
  lambda = model_wdbc_lasso_cv$lambda.min,
  family = "binomial",
  thresh = 1e-8
)

pred_wdbc_lasso <- list(
  train = predict_bayes(model_wdbc_lasso_optimal, X_train),
  test = predict_bayes(model_wdbc_lasso_optimal, X_test)
)

confusion_wdbc_lasso <- list(
  train = confusion_table(Y_train, pred_wdbc_lasso$train),
  test = confusion_table(Y_test, pred_wdbc_lasso$test)
)

acc_wdbc_lasso <- list(
  train = performance_table(Y_train, pred_wdbc_lasso$train),
  test = performance_table(Y_test, pred_wdbc_lasso$test)
)


# Question 4g,h ----------------------------------------------------------

roc_wdbc_lasso <- calculate_roc(model_wdbc_lasso_optimal, X_test, Y_test)
auc_wdbc_lasso <- roc_wdbc_lasso$AUC$value


# Question 5 -------------------------------------------------------------

performance_summary_tbl <-
  list(
    simple = acc_wdbc_simple$test,
    ridge = acc_wdbc_ridge$test,
    lasso = acc_wdbc_lasso$test
  ) %>%
  bind_rows(.id = "model") %>%
  mutate(nzero = c(30, nzero_wdbc_ridge, nzero_wdbc_lasso))
```

Helper Functions

```r
predict_bayes <- function(model, data, threshold = 0.5) {

  factor(
    predict(model, data, type = "response") > threshold,
    levels = c(FALSE, TRUE),
    labels = c("benign", "malignant")
```

```
  )

}
```

```
confusion_table <- function(obs, pred) {

  cf_table <- table(obs, pred)
  names(dimnames(cf_table)) <- c("observed", "predicted")

  return(cf_table)

}

performance_table <- function(obs, pred) {

  cf_tbl <- confusion_table(obs, pred)

  P <- sum(cf_tbl["malignant",])
  N <- sum(cf_tbl["benign",])
  TP <- cf_tbl["malignant", "malignant"]
  TN <- cf_tbl["benign", "benign"]

  tibble(
    acc = mean(obs == pred),
    tpr = TP/P,
    fpr = 1 - TN/N
  )

}
```

```
calculate_roc <- function(model, X, Y, thresholds = seq(0, 1, by = .01)) {

  roc_data <- thresholds %>%
    purrr::map(~predict_bayes(model, X, threshold = .x)) %>%
    purrr::map_dfr(~performance_table(Y, .x), .id = "index")

  roc_func <- with(roc_data, approxfun(x = fpr, y = tpr))
  auc <- integrate(roc_func, 0, 1)

  roc_plot <- ggplot(roc_data, aes(x = fpr, y = tpr)) +
    geom_path(color = "red3") +
    geom_abline(slope = 1, intercept = 0, lty = "dashed", color = "gray") +
    theme_bw(base_family = "serif") +
    labs(
      title = "ROC Curve",
      subtitle = sprintf("AUC: %0.4f", auc$value),
      x = "False Positive Rate (FPR)",
      y = "True Positive Rate (TPR)"
    )

  list(data = roc_data, plot = roc_plot, AUC = auc)

}
```

```r
plot_lambda_coef <- function(model, beta_indices) {

  data <- model$beta[beta_indices, ] %>%
    as.matrix() %>%
    t() %>%
    as_tibble() %>%
    setNames(paste("beta", beta_indices)) %>%
    mutate(lambda = model$lambda) %>%
    tidyr::pivot_longer(-lambda, names_to = "Coefficients")

  ggplot(data, aes(x = log(lambda), y = value, color = Coefficients)) +
    geom_line() +
    theme_bw(base_family = "serif") +
    labs(title = "Coefficients vs Lambda Penalty")

}


plot_cv_error <- function(model) {

  data <- tibble(
    lambda = model$lambda,
    cv_err_mean = model$cvm,
    cv_err_lo = model$cvlo,
    cv_err_up = model$cvup
  )

  ggplot(data, aes(x = log(lambda), y = cv_err_mean)) +
    geom_errorbar(aes(ymin = cv_err_lo, ymax = cv_err_up), color = "gray") +
    geom_point(color = "red3") +
    geom_vline(
      xintercept = log(c(model$lambda.min, model$lambda.1se)),
      lty = "dashed"
    ) +
    theme_bw(base_family = "serif") +
    labs(
      title = "CV Error vs Lambda Penalty",
      x = "log(lambda)",
      y = "Misclassification error"
    )

}
```