STA304XS - Assignment 3: Generalised Linear models

Jing Yeh
yhxjin001@myuct.ac.za

Saurav Sathnarayan
sthsau001@myuct.ac.za

2025-10-23

Abstract

This report explores the relationship between Ridge Logistic Regression and Bayesian Maximum A Posteriori (MAP) estimation with Gaussian priors. The theoretical connection between the ridge penalty and the prior precision is derived, showing how regularisation arises naturally from Bayesian assumptions. A modified Iteratively Weighted Least Squares (IWLS) algorithm for ridge-penalised logistic regression is derived step-by-step and implemented in R. The performance of the custom Ridge-IWLS implementation is compared to standard software (e.g., glmnet), evaluating coefficient shrinkage, predictive accuracy, and computational stability. The analysis highlights the role of regularisation in improving generalisation and mitigating overfitting in binary classification problems.

Keywords: Ridge Logistic Regression, Bayesian MAP Estimation, IWLS Algorithm, Regularisation, Penalised Likelihood, Coefficient Shrinkage, Logistic Regression, Generalisation.

Department of Statistical Sciences Plagiarism Declaration form

A copy of this form, completed and signed, to be attached to all coursework submissions to the Statistical Sciences Department. Submissions without this form will not be marked.

COURSE:	**STA3043/7/8S**
TITLE:	STA304X ASSIGNMENT 3
STUDENT NAME:	JING YEH, SAURAV SATHNARAYAN
STUDENT NUMBER:	YHXJIN001, STHSAU001
TUTOR'S NAME:	
TUT. GROUP #:	

PLAGIARISM DECLARATION

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used a generally accepted citation and referencing style. Each contribution to, and quotation in, this tutorial/report/project from the work(s) of other people has been attributed, and has been cited and referenced.
3. This tutorial/report/project is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.
5. I acknowledge that copying someone else's assignment or essay, or part of it, is wrong, and declare that this is my own work.

Note that agreement to this statement does not exonerate you from the University's plagiarism rules (http://www.uct.ac.za/uct/policies/plagiarism_students.pdf).

Signature: [illegible] [illegible]

Date: 23/10/2025

Contents

1 Question 1 - Bayesian Interpretation

1.1 (a)

We start with the logistic regression model:

$$\Pr(Y_i = 1 \mid x_i) = \text{logit}^{-1}(x_i^\top \beta) = \frac{1}{1+\exp(-x_i^\top \beta)}.$$

The log-likelihood for all n observations is

$$l(\beta) = \sum_{i=1}^{n} \left[y_i x_i^\top \beta - \log(1 + e^{x_i^\top \beta}) \right].$$

Assume independent Normal priors for the coefficients:

$$\beta_j \sim N(0, \tau^2), \quad j = 1, \ldots, p.$$

Hence, the prior density is

$$\pi(\beta) = \prod_{j=1}^{p} \frac{1}{\sqrt{2\pi\tau^2}} \exp\left(-\frac{\beta_j^2}{2\tau^2} \right)$$

Taking logs, we obtain the log-prior:

$$\log \pi(\beta) = -\frac{p}{2} \log(2\pi\tau^2) - \frac{1}{2\tau^2} \sum_{j=1}^{p} \beta_j^2$$

Now, by Bayes' theorem,

$$\pi(\beta \mid y) = \frac{\pi(y \mid \beta)\, \pi(\beta)}{\pi(y)}$$

Taking logs of both sides gives

$$\log \pi(\beta \mid y) = \log \pi(y \mid \beta) + \log \pi(\beta) - \log \pi(y)$$

The term $\log \pi(y)$ is a normalising constant that does not depend on β, so when maximising over β, it can be ignored. Therefore,

$$\log \pi(\beta \mid y) \propto \log \pi(y \mid \beta) + \log \pi(\beta)$$

Substituting the expressions for $\log p(y \mid \beta)$ and $\log p(\beta)$, we have

$$\log p(\beta \mid y) \propto l(\beta) - \frac{1}{2\tau^2} \sum_{j=1}^{p} \beta_j^2$$

This is the expression for the log-posterior up to a constant.

To obtain the maximum a posteriori (MAP) estimate, we maximise $\log p(\beta \mid y)$ with respect to β. Equivalently, we minimise the negative log-posterior:

$$\widehat{\beta}_{MAP} = \arg\min_{\beta} \left[-l(\beta) + \frac{1}{2\tau^2} \sum_{j=1}^{p} \beta_j^2 \right]$$

If we define $\lambda = \dfrac{1}{2\tau^2}$, then the optimisation problem becomes

$$\boxed{\widehat{\beta}_{MAP} = \arg\min_{\beta} \left[-l(\beta) + \lambda \|\beta\|_2^2 \right]}$$

This shows that the MAP estimator under a Normal prior is equivalent to the ridge-regularised logistic regression estimator, where the penalty parameter λ corresponds to the precision of the prior.

1.2 (b)

Role of λ and τ: λ controls the amount of shrinkage or regularisation applied to the coefficients.A **large** λ (small τ^2) implies a strong prior belief that coefficients should be close to zero, resulting in **greater shrinkage**. A **small** λ (large τ^2) implies a weak prior, approaching the unpenalised **maximum likelihood estimate (MLE)**.

Effect of this regularisation: It helps prevent **overfitting** by penalising large coefficient values.

Bayesian vs. MLE perspective: The MLE only uses the data and can overfit when the sample size is small. The Bayesian (MAP) approach incorporates **prior information** through τ^2, providing a **probabilistic justification** for regularisation. Hence, the Bayesian view explains **why** regularisation arises naturally as a consequence of imposing a Gaussian prior.

2 Question 2 - Deriving Ridge - IWLS

2.1 (a)

In IWLS, the weight for observation i is given by

$$w_i^{(t)} = \frac{1}{\mathrm{Var}(Y_i)} \left(\frac{\partial \mu_i}{\partial \eta_i} \right)^2 .$$

For logistic regression, $\mathrm{Var}(Y_i) = p_i(1 - p_i)$ and $\frac{\partial \mu_i}{\partial \eta_i} = p_i(1 - p_i)$, so

$$w_i^{(t)} = \frac{\left(p_i^{(t)}(1 - p_i^{(t)}) \right)^2}{p_i^{(t)}(1 - p_i^{(t)})} = p_i^{(t)}(1 - p_i^{(t)}).$$

These are exactly the diagonal entries of the weight matrix:

$$W^{(t)} = \operatorname{diag}\left(p_1^{(t)}(1-p_1^{(t)}), \ldots, p_n^{(t)}(1-p_n^{(t)})\right).$$

and $p^{(t)} = (p_1^{(t)}, \ldots, p_n^{(t)})^\top$ are the predicted probabilities at $\beta^{(t)}$.
and for $z^{(t)}$

$$z_i^{(t)} = \eta_i^{(t)} + \frac{y_i - \mu_i^{(t)}}{\frac{\partial \mu_i}{\partial \eta_i}}$$

where

$$\eta_i^{(t)} = x_i^\top \beta^{(t)}, \quad \mu_i^{(t)} = \sigma(\eta_i^{(t)}) = p_i^{(t)}, \quad \frac{\partial \mu_i}{\partial \eta_i} = p_i^{(t)}(1-p_i^{(t)}).$$

Substituting the derivative for logistic regression gives

$$z_i^{(t)} = x_i^\top \beta^{(t)} + \frac{y_i - p_i^{(t)}}{p_i^{(t)}(1-p_i^{(t)})}.$$

In matrix form, for all n observations:

$$z^{(t)} = X\beta^{(t)} + (W^{(t)})^{-1}(y - p^{(t)}),$$

From our IWLS formula, we have:

$$(\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}\beta^{(t+1)} = \mathbf{X}^T\mathbf{W}^{(t)}z^{(t)}$$

Thus,

$$\boxed{\beta^{(t+1)} = (\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^{(t)}z^{(t)}}$$

(b)

We are asked to derive the modified IWLS update formula for the Ridge-penalised objective function.

1. Objective Function

We start from the objective function to be minimised:

$$F(\boldsymbol{\beta}) = -\ell(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_2^2 = -\ell(\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^\top\boldsymbol{\beta}$$

The Newton-Raphson update rule for minimising $F(\boldsymbol{\beta})$ is:

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - [H_F(\boldsymbol{\beta}^{(t)})]^{-1}U_F(\boldsymbol{\beta}^{(t)})$$

We need the gradient U_F and the Hessian H_F of our new objective function F.

3. Gradient (U_F) and Hessian (H_F)

From part a, we know:

- Standard Score Vector: $U_\ell = \frac{\partial \ell}{\partial \boldsymbol{\beta}} = \mathbf{X}^\top(\mathbf{y} - \mathbf{p})$
- Standard Hessian: $H_\ell = \frac{\partial^2 \ell}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} = -\mathbf{X}^\top \mathbf{W} \mathbf{X}$

Now, we find the derivatives for our penalised objective F:
Gradient (U_F):

$$U_F = \frac{\partial F}{\partial \boldsymbol{\beta}} = \frac{\partial}{\partial \boldsymbol{\beta}}(-\ell(\boldsymbol{\beta})) + \frac{\partial}{\partial \boldsymbol{\beta}}(\lambda \boldsymbol{\beta}^\top \boldsymbol{\beta})$$

Using denominator layout for matrix calculus, we have:

$$U_p = \frac{\partial f_p}{\partial \boldsymbol{\beta}} = \begin{bmatrix} \frac{\partial f_p}{\partial \beta_1} \\ \frac{\partial f_p}{\partial \beta_2} \\ \vdots \\ \frac{\partial f_p}{\partial \beta_p} \end{bmatrix} = \begin{bmatrix} 2\lambda\beta_1 \\ 2\lambda\beta_2 \\ \vdots \\ 2\lambda\beta_p \end{bmatrix} = 2\lambda\boldsymbol{\beta}$$

And thus:

$$U_F = -U_\ell + 2\lambda\boldsymbol{\beta} = -\mathbf{X}^\top(\mathbf{y} - \mathbf{p}) + 2\lambda\boldsymbol{\beta}$$

Hessian (H_F):

$$H_F = \frac{\partial^2 F}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} = \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top}(-\ell(\boldsymbol{\beta})) + \frac{\partial^2}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top}(\lambda \boldsymbol{\beta}^\top \boldsymbol{\beta})$$

Similarly, using matrix calculus, we get:

$$H_p = \frac{\partial U_p}{\partial \boldsymbol{\beta}^\top} = \frac{\partial}{\partial \boldsymbol{\beta}^\top} \begin{bmatrix} 2\lambda\beta_1 \\ \vdots \\ 2\lambda\beta_p \end{bmatrix} = \begin{bmatrix} \frac{\partial(2\lambda\beta_1)}{\partial\beta_1} & \dots & \frac{\partial(2\lambda\beta_1)}{\partial\beta_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial(2\lambda\beta_p)}{\partial\beta_1} & \dots & \frac{\partial(2\lambda\beta_p)}{\partial\beta_p} \end{bmatrix} = \begin{bmatrix} 2\lambda & 0 & \dots & 0 \\ 0 & 2\lambda & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 2\lambda \end{bmatrix} = 2\lambda\mathbf{I}$$

Therefore, we have:

$$H_F = -H_\ell + 2\lambda\mathbf{I} = (\mathbf{X}^\top \mathbf{W} \mathbf{X}) + 2\lambda\mathbf{I}$$

4. Algebraic Simplification

We substitute in the $\mathbf{H_F}$ and $\mathbf{U_F}$ we've derived.

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - [H_F^{(t)}]^{-1} U_F^{(t)}$$

Move $\boldsymbol{\beta}^{(t)}$ to the left and pre-multiply both sides by $H_F^{(t)}$:

$$H_F^{(t)}(\boldsymbol{\beta}^{(t+1)} - \boldsymbol{\beta}^{(t)}) = -U_F^{(t)}$$

Expand the left side:

$$H_F^{(t)}\boldsymbol{\beta}^{(t+1)} - H_F^{(t)}\boldsymbol{\beta}^{(t)} = -U_F^{(t)}$$

Isolate the $\boldsymbol{\beta}^{(t+1)}$ term:

$$H_F^{(t)}\boldsymbol{\beta}^{(t+1)} = H_F^{(t)}\boldsymbol{\beta}^{(t)} - U_F^{(t)}$$

Now, substitute the concrete expressions for $H_F^{(t)}$ and $U_F^{(t)}$ from Step 3:

$$[\mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{X} + 2\lambda\mathbf{I}]\boldsymbol{\beta}^{(t+1)} = [\mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{X} + 2\lambda\mathbf{I}]\boldsymbol{\beta}^{(t)} - \left[-\mathbf{X}^\top(\mathbf{y} - \mathbf{p}^{(t)}) + 2\lambda\boldsymbol{\beta}^{(t)}\right]$$

Distribute the negative sign on the right-hand side and then cancel out the $2\lambda\boldsymbol{\beta}^{(t)}$ terms:

$$[\mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{X} + 2\lambda\mathbf{I}]\boldsymbol{\beta}^{(t+1)} = (\mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{X})\boldsymbol{\beta}^{(t)} + \mathbf{X}^\top(\mathbf{y} - \mathbf{p}^{(t)})$$

We recognise the right-hand side as the core expression from the standard IWLS update, which simplifies to $\mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{z}^{(t)}$, where $\mathbf{z}^{(t)} = \mathbf{X}\boldsymbol{\beta}^{(t)} + (\mathbf{W}^{(t)})^{-1}(\mathbf{y} - \mathbf{p}^{(t)})$.

$$[\mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{X} + 2\lambda\mathbf{I}]\boldsymbol{\beta}^{(t+1)} = \mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{z}^{(t)}$$

Finally, isolate $\boldsymbol{\beta}^{(t+1)}$ to get the desired formula:

$$\boldsymbol{\beta}^{(t+1)} = (\mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{X} + 2\lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{W}^{(t)}\mathbf{z}^{(t)}$$

3 Question 3 - Implementation and Evaluation

3.1 (a)

```
glm_fit = function(Y, X, lambda = 1, eps = 1e-6, K = 100) {
  beta = rep(0, p)
  for (iter in 1:K) {

    # Linear predictor
    eta = as.vector(X %*% beta)

    # For logistic regression (binomial), use:
    mu = 1 / (1 + exp(-eta))

    # Weights and z
    W = diag(as.vector(mu * (1 - mu)), n, n)
    z = eta + (Y - mu) / (mu * (1 - mu))

    # Lambda Scaling
    P = diag(rep(lambda / n, p))

    # Ridge-IWLS updated
    XtWX = t(X) %*% W %*% X
    XtWz = t(X) %*% W %*% z
    beta_new = solve(XtWX + 2 * lambda * diag(p), XtWz)

    # Check convergence
    if (sqrt(sum((beta_new - beta)^2)) < eps) break
    beta = beta_new
  }
  return = list(coefficients = beta, iterations = iter, converged = (iter < K))
}
```

We used $\lambda = 1$ and our convergence criteria is checking if the differences between out beta values are less than 1eˆ-16 or not.

3.2 (b) & (c)

3.2.1 Estimates

Looking at our estimates, we see that our intercepts are very different from each other. This is due to the fact that,in IWLS, our X matrix includes a column of ones and no centering/standardization is done. However, the glmnet package fits an intercept separately and standardises for us, even with standardize = FALSE, the intercept optimization is treated separately. Other differences in the magnitudes of coefficients are small.

Table 1: Ridge IWLS - Coefficient estimates.

	Estimate
1	0.0576661
Measurement1	0.1642729
Measurement2	0.6732748
Measurement3	-1.2440750

Table 2: Glmnet coefficient estimates.

	Estimate
(Intercept)	17.9690080
Measurement1	0.0527774
Measurement2	0.7371886
Measurement3	-1.2873998

3.2.2 Prediction Performance

Table 3: Predicted probabilities: Ridge-IWLS vs. glmnet

Obs	Actual	IWLS_Pred	Glmnet_Pred
1	1	0.9701	0.9427
2	0	0.0002	0.0000
3	0	0.2312	0.2208
4	1	0.9969	0.9963
5	0	0.0040	0.0023
6	0	0.0009	0.0002
7	0	0.0001	0.0000
8	0	0.0110	0.0038
9	0	0.0027	0.0030
10	1	0.9470	0.9283

Despite different intercepts, the predicted probabilities are similar. Small differences in slopes and intercept can slightly shift the probabilities, but classification accuracy or nearly identical. In our tables of predicted probalities we have an ‘Actual’ Column which tells us the disease status of our observations.

Table 4: Model performance comparison (MSE)

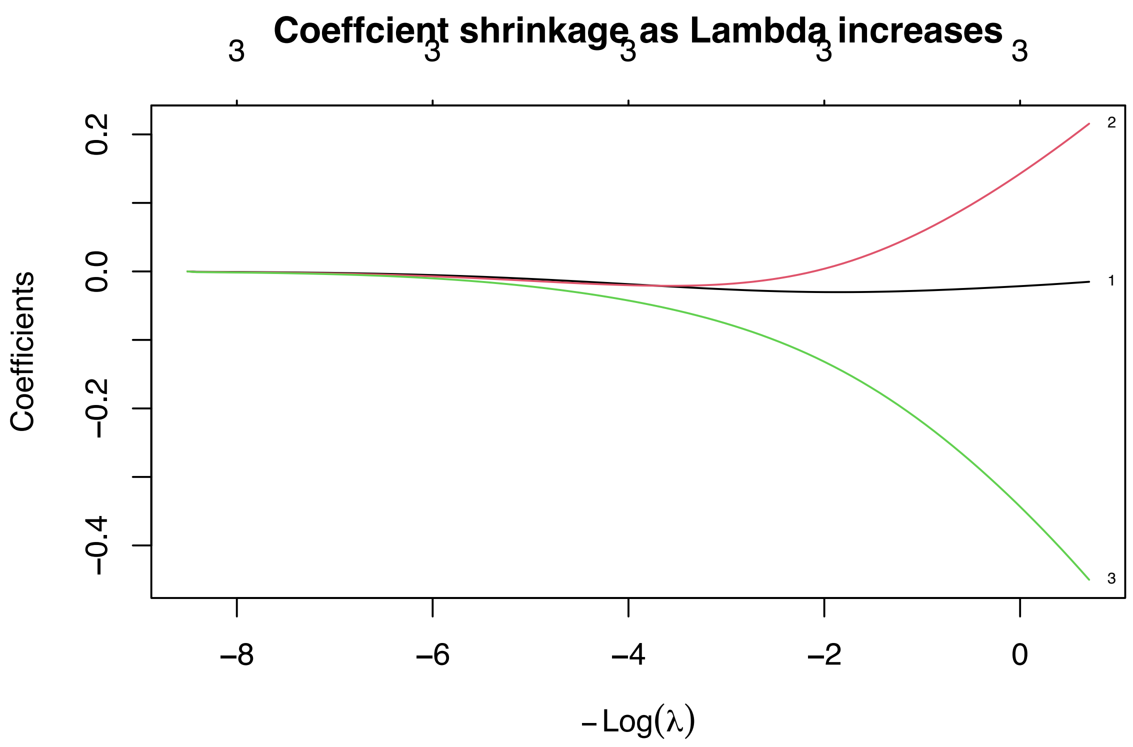
Model	MSE
Ridge-IWLS	0.019529
glmnet	0.017164

Model performance is roughly the same, with the glmnet package performing slightly better.

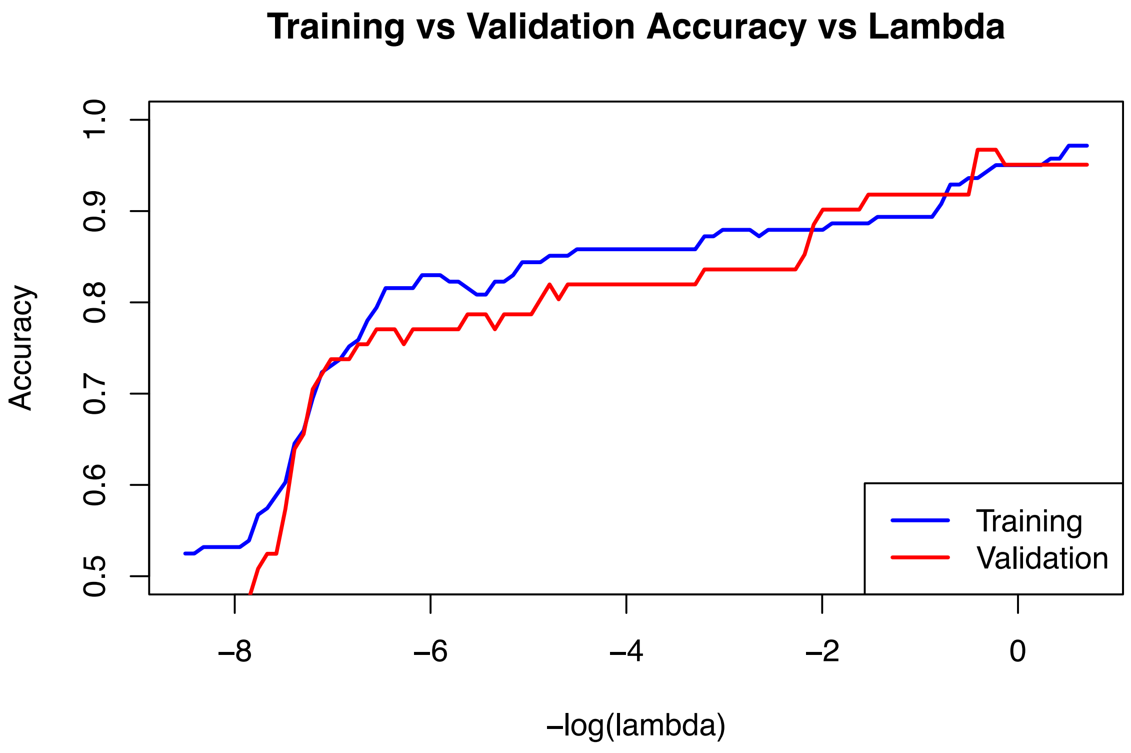
3.2.3 Algorithmic Stability and Limitations of IWLS

IWLS is slower for large p or n, because each iteration requires computing t(X) %% *W* %% X. While, glmnet uses coordinate descent, which is much faster and more memory-efficient for large datasets. IWLS is good small problems, but for large datasets, glmnet is preferable. Differences in intercept values are normal and expected due to centering, scaling, and optimization differences. With proper scaling and intercept handling, IWLS can approximate glmnet closely.

4 Question 4 - Bonus: Visualisation and Analysis



A smaller -log(λ) value indicates a larger penalty, therefore our coefficients reduce to zero, as -log(λ) increases, our coefficients move to their true values.



To evaluate the effect of the regularization parameter λ on model performance, we computed predicted probabilities from both the Ridge-IWLS model and glmnet logistic regression across a sequence of λ values. For each λ, the predicted probabilities were converted into binary class predictions using a threshold of 0.5, and accuracy was calculated separately on the training and validation sets. This procedure allows us to observe how increasing λ (stronger regularization) shrinks the coefficients, reducing variance and preventing overfitting. Typically, training accuracy decreases as λ increases due to the penalty constraining the model, while validation accuracy initially rises to a peak at an optimal λ before declining as over-regularization induces underfitting. This analysis illustrates the bias–variance tradeoff and demonstrates how ridge regularization improves generalization compared to unpenalized logistic regression.

4.1 Appendix

```
# Global options for the document
knitr::opts_chunk$set(
  echo = FALSE,
  warning = FALSE,
  message = FALSE,
  fig.pos = '!ht', # =- Change this line
  out.width = '100%',
  dpi = 300
)

# Load all required packages
library(knitr)
library(kableExtra)
library(glmnet)
dat = read.table('~/Desktop/template/Assignment_Disease.txt', header = TRUE)
attach(dat)
Y = dat$DiseaseStatus
X = as.matrix(cbind(1, dat[, c("Measurement1", "Measurement2", "Measurement3")]))
n = length(Y)
p = ncol(X)
glm_fit = function(Y, X, lambda = 1, eps = 1e-6, K = 100) {
  beta = rep(0, p)
  for (iter in 1:K) {

    # Linear predictor
```

```r
    eta = as.vector(X %*% beta)

    # For logistic regression (binomial), use:
    mu = 1 / (1 + exp(-eta))

    # Weights and z
    W = diag(as.vector(mu * (1 - mu)), n, n)
    z = eta + (Y - mu) / (mu * (1 - mu))

    # Lambda Scaling
    P = diag(rep(lambda / n, p))

    # Ridge-IWLS updated
    XtWX = t(X) %*% W %*% X
    XtWz = t(X) %*% W %*% z
    beta_new = solve(XtWX + 2 * lambda * diag(p), XtWz)

    # Check convergence
    if (sqrt(sum((beta_new - beta)^2)) < eps) break
    beta = beta_new
  }
  return = list(coefficients = beta, iterations = iter, converged = (iter < K))
}
fit = glm_fit(Y, X, lambda = 1)
fit_glmnet = glmnet(X[,-1], Y, family = "binomial", alpha = 0, lambda = 1/n, intercep
#Ridge IWLS coefficients
df <- data.frame(
  Estimate = fit$coefficients
)
kable(df, format = "latex", booktabs = FALSE, caption = "Ridge IWLS - Coefficient est
  kable_styling(latex_options = "hold_position")

# glmnet coefficients
coefs <- as.matrix(coef(fit_glmnet))
df2 <- data.frame(
  Estimate = coefs[, 1]
)
kable(df2, format = "latex", booktabs = FALSE, caption = "Glmnet coefficient estimate
  kable_styling(latex_options = "hold_position")

# IWLS predictions
eta_iwls = X %*% fit$coefficients
p_iwls = 1 / (1 + exp(-eta_iwls))


# glmnet predictions
p_glmnet = predict(fit_glmnet,X[,-1], type = "response", s = 1/n)

#mean squared error
mse_iwls = mean((Y - p_iwls)^2)
mse_glmnet = mean((Y - p_glmnet)^2)


pred_df = data.frame(
  Obs = 1:10,
  Actual = Y[1:10],
  IWLS_Pred = round(p_iwls[1:10], 4),
  Glmnet_Pred = round(p_glmnet[1:10], 4)
)

kable(pred_df, format = "latex", booktabs = FALSE, caption = "Predicted probabilities
  kable_styling(latex_options = "hold_position", position = "center")
perf <- data.frame(
  Model = c("Ridge-IWLS", "glmnet"),
  MSE = c(round(mse_iwls, 6), round(mse_glmnet, 6)))
  kable(perf, format = "latex", booktabs = TRUE, caption = "Model performance compari
  kable_styling(latex_options = "hold_position", position = "center")
fit = glmnet(X[,-1], Y, family = "binomial",alpha = 0, intercept = TRUE, standardize
plot(fit, xvar = "lambda", label = TRUE, main = 'Coeffcient shrinkage as Lambda incre

set.seed(2025)

# Split into training and validation sets
index = sample(1:n, size = 0.7*n)
X_train = X[index, ]
Y_train = Y[index]
```

```r
X_val = X[-index, ]
Y_val = Y[-index]

# Fit Ridge (alpha = 0) over a sequence of lambda values
fit = glmnet(X_train, Y_train, family = "binomial", alpha = 0, standardize = FALSE)

# Lambda sequence
lambda_seq = fit$lambda
n_lambda = length(lambda_seq)

# Initialize accuracy vectors
train_acc = numeric(n_lambda)
val_acc = numeric(n_lambda)

# Compute training and validation accuracy for each lambda
for (i in 1:n_lambda) {
  pred_train = predict(fit, newx = X_train, s = lambda_seq[i], type = "response")
  pred_val = predict(fit, newx = X_val, s = lambda_seq[i], type = "response")

  train_acc[i] = mean((pred_train > 0.5) == Y_train)
  val_acc[i] = mean((pred_val > 0.5) == Y_val)
}

# Plot
plot(-log(lambda_seq), train_acc, type = "l", col = "blue", lwd = 2,
     xlab = "-log(lambda)", ylab = "Accuracy", ylim = c(0.5, 1),
     main = "Training vs Validation Accuracy vs Lambda")
lines(-log(lambda_seq), val_acc, col = "red", lwd = 2)
legend("bottomright", legend = c("Training", "Validation"),
       col = c("blue", "red"), lwd = 2)
```