Penalized Logistic Regression

Jina Ryu, Julie Seo, Michael Giudice, Nestor Mendoza

April 3, 2025

Overview

1. Define Linear Regression, Penalized Regression and Penalized Logistic Regression.
2. Define Lasso, Ridge and Elastic Net Regression.
3. Comment on the advantages/disadvantages of each type of Penalized Regression.
4. See the implementation of Lasso, Ridge and Elastic Net Regression using R packages.

Linear Regression

Linear regression is defined as a model that summarizes data by the linear equation

$$y = \beta_0 + \sum_{i=1}^{n} x_i \beta_i + \varepsilon.$$

Where the β_i are fixed coefficients that are found by a minimizing process.

Penalized Regression

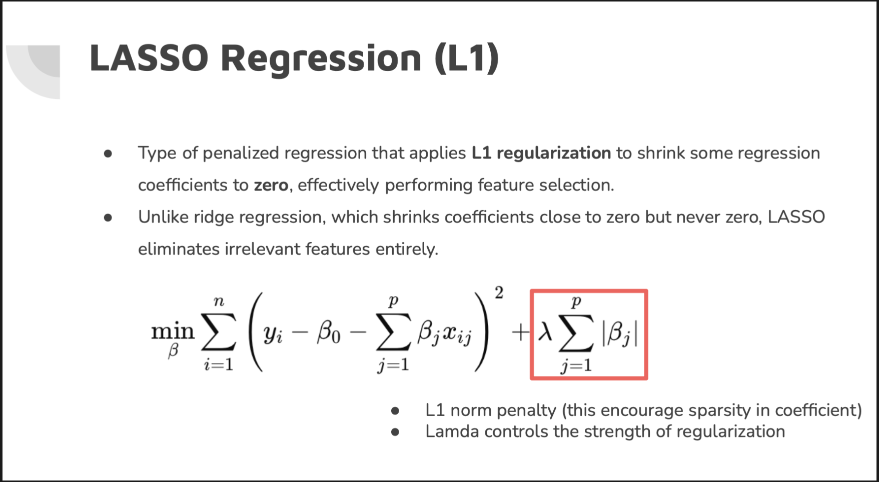
Penalized Regression is a type of Regression that adds a penalty term to the minimizing process used to find the fixed coefficients.

The purpose of this penalization is to shrink the fixed coefficients.

Types of Penalized Regression

There are 3 main types of Penalized Regression: Lasso Regression, Ridge Regression and Elastic Net Regression,

Lasso L1 Regression



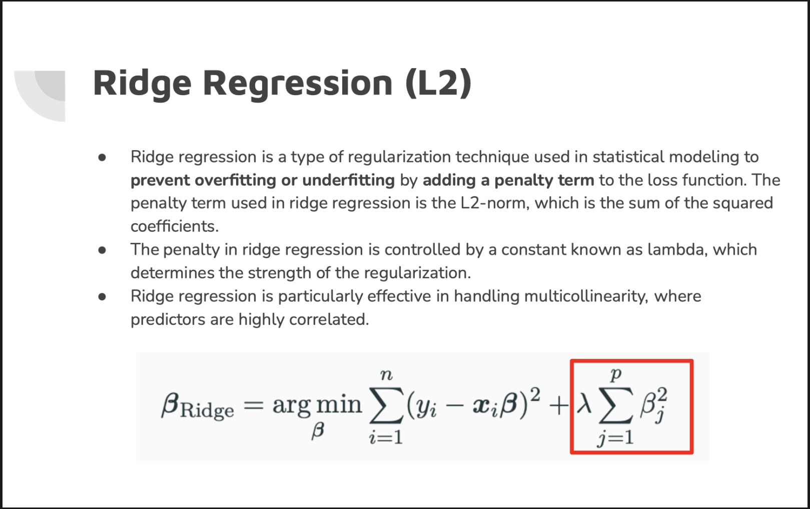
LASSO Regression (L1)

- Type of penalized regression that applies **L1 regularization** to shrink some regression coefficients to **zero**, effectively performing feature selection.
- Unlike ridge regression, which shrinks coefficients close to zero but never zero, LASSO eliminates irrelevant features entirely.

$$\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \boxed{\lambda \sum_{j=1}^{p} |\beta_j|}$$

- L1 norm penalty (this encourage sparsity in coefficient)
- Lamda controls the strength of regularization

Ridge L2 Regression

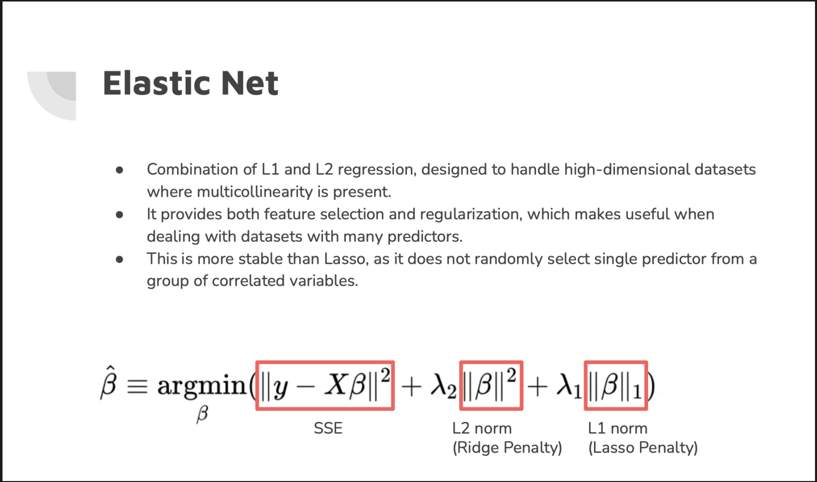


Ridge Regression (L2)

- Ridge regression is a type of regularization technique used in statistical modeling to **prevent overfitting or underfitting** by **adding a penalty term** to the loss function. The penalty term used in ridge regression is the L2-norm, which is the sum of the squared coefficients.
- The penalty in ridge regression is controlled by a constant known as lambda, which determines the strength of the regularization.
- Ridge regression is particularly effective in handling multicollinearity, where predictors are highly correlated.

$$\boldsymbol{\beta}_{\text{Ridge}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i \boldsymbol{\beta})^2 + \boxed{\lambda \sum_{j=1}^{p} \beta_j^2}$$

Elastic Net Regression



Elastic Net

- Combination of L1 and L2 regression, designed to handle high-dimensional datasets where multicollinearity is present.
- It provides both feature selection and regularization, which makes useful when dealing with datasets with many predictors.
- This is more stable than Lasso, as it does not randomly select single predictor from a group of correlated variables.

$$\hat{\beta} \equiv \underset{\beta}{\operatorname{argmin}}(\underbrace{\boxed{||y - X\beta||^2}}_{\text{SSE}} + \lambda_2 \underbrace{\boxed{||\beta||^2}}_{\text{L2 norm (Ridge Penalty)}} + \lambda_1 \underbrace{\boxed{||\beta||_1}}_{\text{L1 norm (Lasso Penalty)}})$$

Logistic Regression

Logistic Regression is a type of Regression in which we look to fit the linear model

$$\text{logit}(p_{x_1,\ldots,x_k}) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i.$$

Logistic Regression

The coefficients β_i are calculated by maximizing the Log Likelihood function:

$$\begin{aligned}\ln(L) &= \ln(\Pi_{i=1}^{n} P(D_i = 1|X = X_i)P(X = X_i)) \\ &= \sum_{i=1}^{n} \ln(P(D_i = 1|X = X_i)P(X = X_i))\end{aligned}$$

Logistic Penalized Regression

Logistic Penalized Regression is a regression where we fit a model of the form

$$\text{logit}(p_{x_1,\ldots,x_k}) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i.$$

However, we find the coefficients by minimizing a function called the Negative Penalized Log Likelihood Function rather than maximizing the Log Likelihood function.

Logistic Penalized Regression: Negative Penalized Log Likelihood Function

The Penalized Log Likelihood function is defined as

$$J(\beta,\lambda) = -\sum_{i=1}^{m}[Y_i \ln(P(Y_i|X_i) + (1-Y_i)\ln(1-P(Y_i|X_i))] + \lambda R(\beta).$$

Where $R(\beta)$ is called the regularization term, β is the vector of the unknown β parameters and λ is a parameter that is found through a process named Cross Validation.

Logistic Penalized Regression: Regularization Term

The Regularization term determines the type of penalized logistic regression. In particular:

Lasso $\implies R(\beta) = \sum_{j=1}^{n} |\beta_j|$

Ridge $\implies R(\beta) = \sum_{j=1}^{n} \beta_j^2$

Elastic Net $\implies R(\beta) = \alpha \sum_{j=1}^{n} |\beta_j| + (1-\alpha) \sum_{j=1}^{n} \beta_j^2$ for some $\alpha \in (0,1)$.

Patterns in the coefficients of Penalized Regression.

A key pattern in penalized regression is that:

In Lasso regression the coefficients may be reduced to zero.

In Ridge regression the coefficients cannot be reduced to zero.

In Elastic Net regression the coefficients may or may not be reduced to zero.

Patterns in the coefficients of Penalized Regression.

According to a book named Statistical Learning with Sparsity the reason for this pattern is due to the norm used in the Regularization term.

Since the Regularization term in Lasso Regression is of the form

$$R(\beta) = \sum_{j=1}^{n} |\beta_j|.$$

Then, the minimizing process can be described by geometry under the L1 norm which causes the pattern.

Patterns in the coefficients of Penalized Regression.

Similarly, since the Regularizing term in Ridge regression is of the form

$$R(\beta) = \sum_{i=1}^{n} \beta_j^2.$$

Then, the minimizing process can be described by geometry under the L2 norm which causes the pattern.

Advantages/Disadvantages of Lasso Regression:

Advantages:

Feature Selection: Lasso regression helps to create a simpler and more interpretable model by keeping only the most relevant features.

Disadvantages:
Problems with correlated variables: Lasso regression can under perform when features are correlated. If there are strongly correlated variables, Lasso Regression may arbitrarily select one variable while ignoring others.

Computationally Expensive: Lasso Regression be slower to compute, especially for large datasets.

Advantages/Disadvantages of Ridge Regression:

Advantages:

Prevents Overfitting: Reduces the impact of less significant features by shrinking their coefficients.

Handles Multicollinearity Well: Performs well when independent variables are highly correlated.

Disadvantages:

Does Not Perform Feature Selection: Unlike Lasso, Ridge does not shrink coefficients to zero, meaning it retains all predictors in the model.

Harder Interpretability: Since all variables remain in the model, it may be harder to interpret than a Lasso Regression.

Advantages/Disadvantages of Elastic Net Regression

Advantages:

Combines Benefits of Ridge and Lasso: Provides both regularization (Ridge) and feature selection (Lasso).

Handles Multicollinearity Better Than Lasso: It does not arbitrarily pick one correlated feature but distributes the coefficient weights among them.

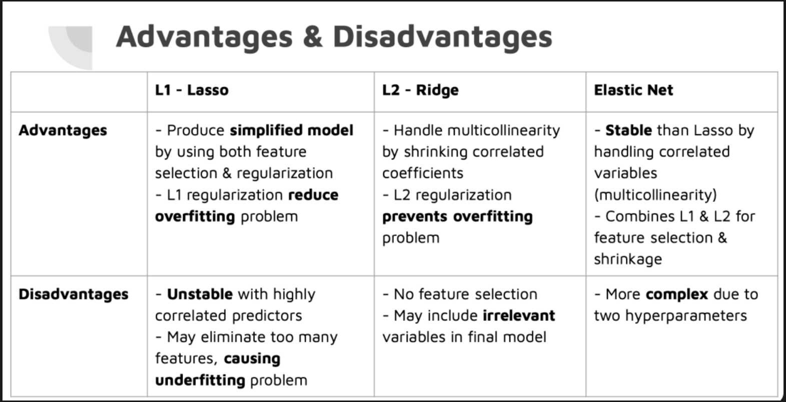
Disadvantages:

Requires Tuning of Two Parameters (α and λ).

Not Always Needed: If data is purely high-dimensional with no strong feature correlation, Ridge or Lasso alone may be sufficient.

Overall Advantages and Disadvantages

Overall we can summarize the advantages and disadvantages of each type of penalized regression by:

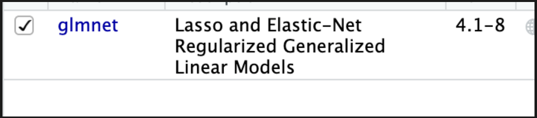


Advantages & Disadvantages

	L1 - Lasso	**L2 - Ridge**	**Elastic Net**
Advantages	- Produce **simplified model** by using both feature selection & regularization - L1 regularization **reduce overfitting** problem	- Handle multicollinearity by shrinking correlated coefficients - L2 regularization **prevents overfitting** problem	- **Stable** than Lasso by handling correlated variables (multicollinearity) - Combines L1 & L2 for feature selection & shrinkage
Disadvantages	- **Unstable** with highly correlated predictors - May eliminate too many features, **causing underfitting** problem	- No feature selection - May include **irrelevant** variables in final model	- More **complex** due to two hyperparameters

R Packages

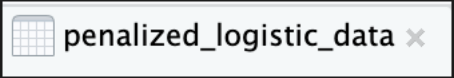
To implement a Lasso, Ridge or Elastic Net penalized regression we will use the package:



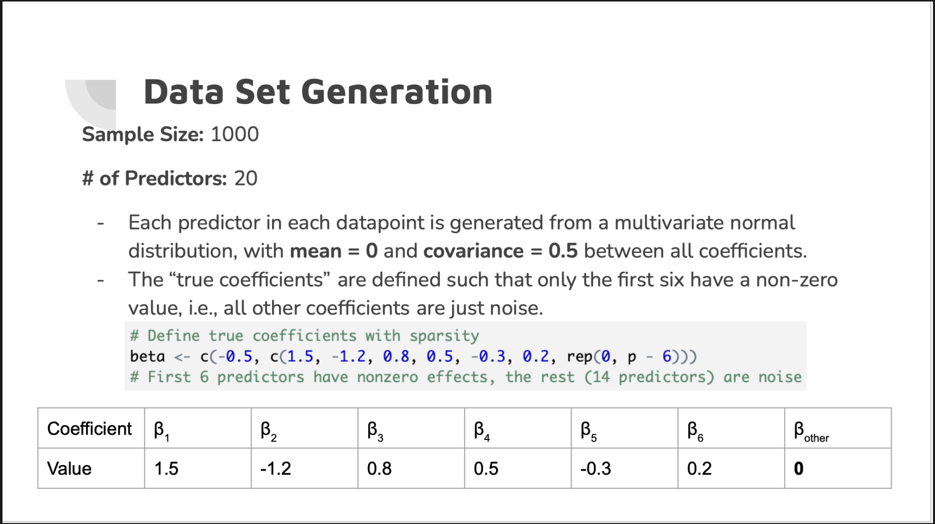
✓	glmnet	Lasso and Elastic-Net Regularized Generalized Linear Models	4.1-8

How to implement a Lasso Regression

Let us create a dataset named


penalized_logistic_data

Creation of the dataset



Data Set Generation

Sample Size: 1000

# of Predictors: 20

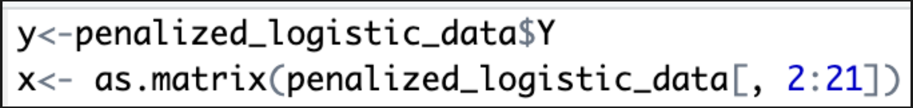
- Each predictor in each datapoint is generated from a multivariate normal distribution, with **mean = 0** and **covariance = 0.5** between all coefficients.
- The "true coefficients" are defined such that only the first six have a non-zero value, i.e., all other coefficients are just noise.

```
# Define true coefficients with sparsity
beta <- c(-0.5, c(1.5, -1.2, 0.8, 0.5, -0.3, 0.2, rep(0, p - 6)))
# First 6 predictors have nonzero effects, the rest (14 predictors) are noise
```

Coefficient	β_1	β_2	β_3	β_4	β_5	β_6	β_{other}
Value	1.5	-1.2	0.8	0.5	-0.3	0.2	**0**

How to implement a Lasso Regression

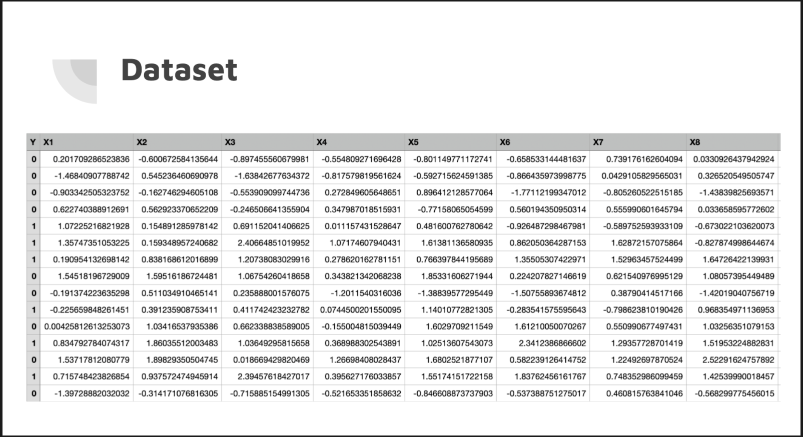
To, implement a Lasso Regression we first extract all the dependent variable values and all the independent variable values into two matrices using the following code:



```
y<-penalized_logistic_data$Y
x<- as.matrix(penalized_logistic_data[, 2:21])
```

Dataset

The dataset looks like:

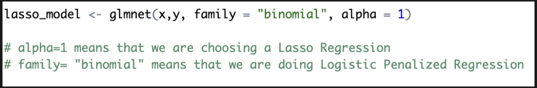


Dataset

Y	X1	X2	X3	X4	X5	X6	X7	X8
0	0.201709286523836	-0.600672584135644	-0.897455560679981	-0.554809271696428	-0.801149771172741	-0.658533144481637	0.739176162604094	0.0330926437942924
0	-1.46840907788742	0.545236460690978	-1.63842677634372	-0.817579819561624	-0.592715624591385	-0.866435973998775	0.0429105829565031	0.326520549505747
0	-0.903342505323752	-0.162746294605108	-0.553909099744736	0.272849605648651	0.896412128577064	-1.77112199347012	-0.805260522515185	-1.43839825693571
0	0.622740388912691	0.562923370652209	-0.246506641355904	0.347987018515931	-0.77158065054599	0.560194350950314	0.555990601645794	0.033658595772602
1	1.07225216821928	0.154891285978142	0.691152041406625	0.011157431528647	0.481600762780642	-0.926487298467981	-0.589752593933109	-0.673022103620073
1	1.35747351053225	0.159348957240682	2.40664851019952	1.07174607940431	1.61381136580935	0.862050364287153	1.62872157075864	-0.827874998644674
1	0.190954132698142	0.838168612016899	1.20738083029916	0.278620162781151	0.766397844195689	1.35505307422971	1.52963457524499	1.64726422139931
0	1.54518196729009	1.59516186724481	1.06754260418658	0.343821342068238	1.85331606271944	0.224207827146619	0.621540976995129	1.08057395449489
0	-0.191374223635298	0.511034910465141	0.235888001576075	-1.2011540316036	-1.38839577295449	-1.50755893674812	0.38790414517166	-1.42019040756719
1	-0.225659848261451	0.391235908753411	0.411742423232782	0.0744500201550095	1.14010772821305	-0.283541575595643	-0.798623810190426	0.968354971136953
0	0.00425812613253073	1.03416537935386	0.662338838589005	-0.155004815039449	1.6029709211549	1.61210050070267	0.550990677497431	1.03256351079153
1	0.834792784074317	1.86035512003483	1.03649295815658	0.368988302543891	1.02513607543073	2.3412386866602	1.29357728701419	1.51953224882831
0	1.53717812080779	1.89829350504745	0.018669429820469	1.26698408028437	1.6802521877107	0.582239126414752	1.22492697870524	2.52291624757892
1	0.715748423826854	0.937572474945914	2.39457618427017	0.395627176033857	1.55174151722158	1.83762456161767	0.748352986099459	1.42539990018457
0	-1.39728882032032	-0.314171076816305	-0.715885154991305	-0.521653351858632	-0.846608873737903	-0.537388751275017	0.460815763841046	-0.568299775456015

How to implement a Lasso Regression

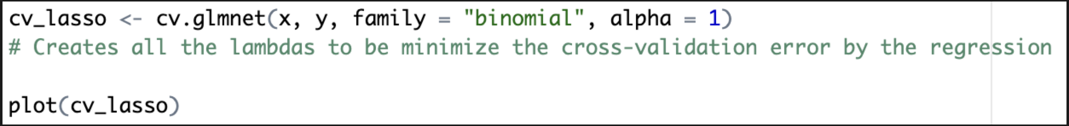
Now, we can use the following function to implement a Lasso regression to the data set



```r
lasso_model <- glmnet(x,y, family = "binomial", alpha = 1)

# alpha=1 means that we are choosing a Lasso Regression
# family= "binomial" means that we are doing Logistic Penalized Regression
```

Minimizing process

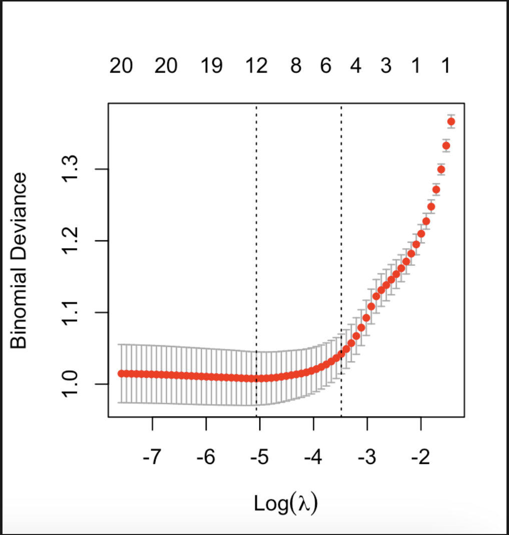
Now, to find the best lambda for our Negative Penalized Log Likelihood Function we implement the code:



```r
cv_lasso <- cv.glmnet(x, y, family = "binomial", alpha = 1)
# Creates all the lambdas to be minimize the cross-validation error by the regression

plot(cv_lasso)
```

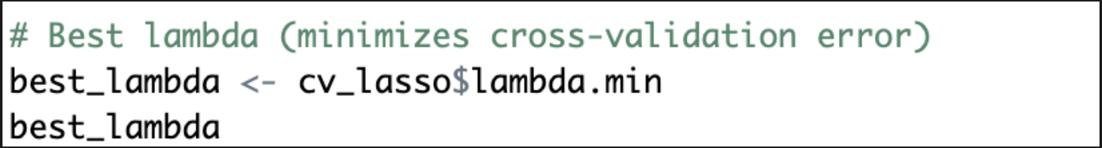
Minimizing process

Which creates the plot



Picking the Optimum Lambda

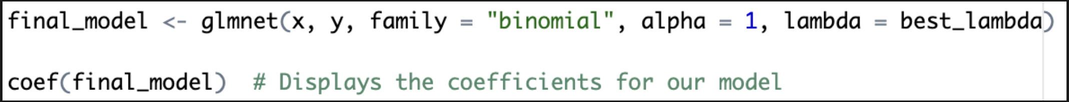
From this plot we can use the following code to pick the optimum Lambda.



```
# Best lambda (minimizes cross-validation error)
best_lambda <- cv_lasso$lambda.min
best_lambda
```

Obtaining the coefficients of the optimal model

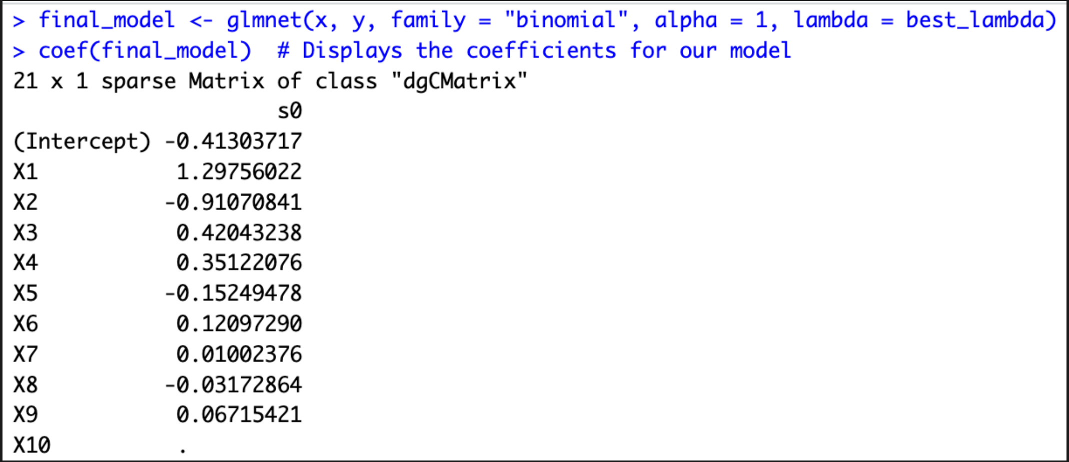
So, now that we found the optimal Lambda. We get the coefficients of the optimal model by the following code:



```r
final_model <- glmnet(x, y, family = "binomial", alpha = 1, lambda = best_lambda)

coef(final_model)  # Displays the coefficients for our model
```

Coefficients

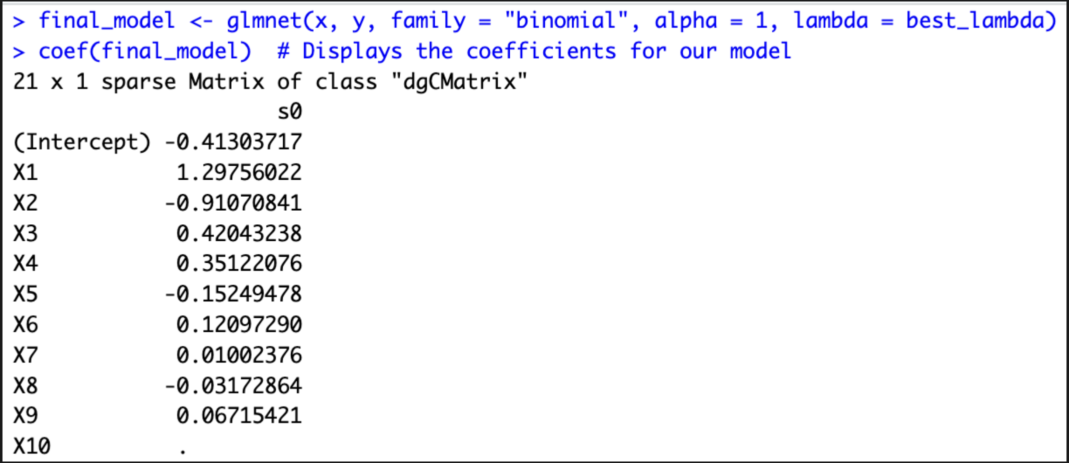
Which outputs the coefficients in the following table



```
> final_model <- glmnet(x, y, family = "binomial", alpha = 1, lambda = best_lambda)
> coef(final_model)  # Displays the coefficients for our model
21 x 1 sparse Matrix of class "dgCMatrix"
                     s0
(Intercept) -0.41303717
X1           1.29756022
X2          -0.91070841
X3           0.42043238
X4           0.35122076
X5          -0.15249478
X6           0.12097290
X7           0.01002376
X8          -0.03172864
X9           0.06715421
X10          .
```

Coefficients

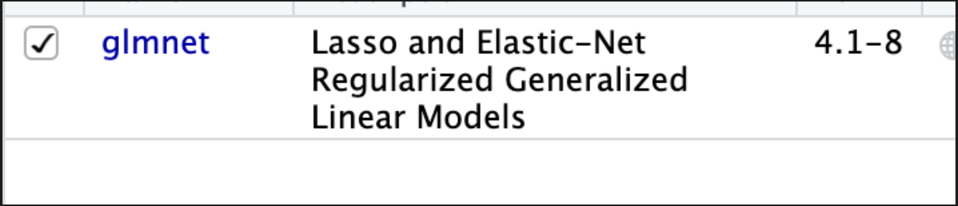
Note that some of the coefficients are zero.



```
> final_model <- glmnet(x, y, family = "binomial", alpha = 1, lambda = best_lambda)
> coef(final_model)  # Displays the coefficients for our model
21 x 1 sparse Matrix of class "dgCMatrix"
                     s0
(Intercept) -0.41303717
X1           1.29756022
X2          -0.91070841
X3           0.42043238
X4           0.35122076
X5          -0.15249478
X6           0.12097290
X7           0.01002376
X8          -0.03172864
X9           0.06715421
X10          .
```

How to implement a Ridge Regression

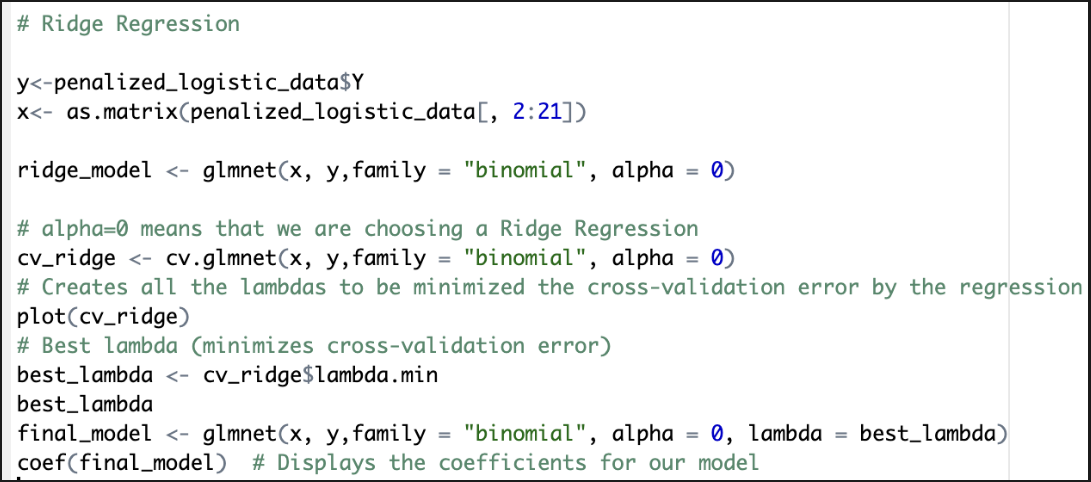
To implement a ridge regression we will use the same package



✓	glmnet	Lasso and Elastic-Net Regularized Generalized Linear Models	4.1-8

How to implement a Ridge Regression

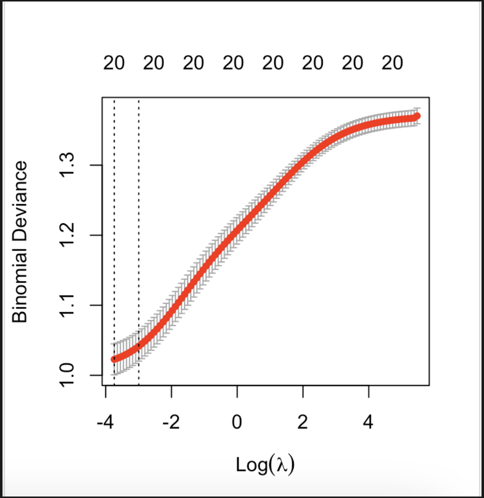
The only difference in the implementation is changing alpha to zero:



```r
# Ridge Regression

y<-penalized_logistic_data$Y
x<- as.matrix(penalized_logistic_data[, 2:21])

ridge_model <- glmnet(x, y,family = "binomial", alpha = 0)

# alpha=0 means that we are choosing a Ridge Regression
cv_ridge <- cv.glmnet(x, y,family = "binomial", alpha = 0)
# Creates all the lambdas to be minimized the cross-validation error by the regression
plot(cv_ridge)
# Best lambda (minimizes cross-validation error)
best_lambda <- cv_ridge$lambda.min
best_lambda
final_model <- glmnet(x, y,family = "binomial", alpha = 0, lambda = best_lambda)
coef(final_model)  # Displays the coefficients for our model
```

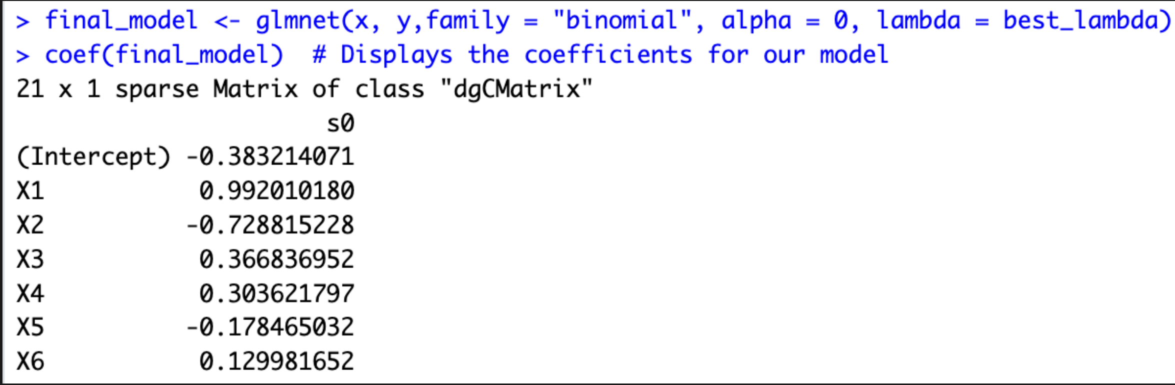
Results of the implementation of the Ridge Regression

From the code we obtain the plot



Results of the implementation of the Ridge Regression

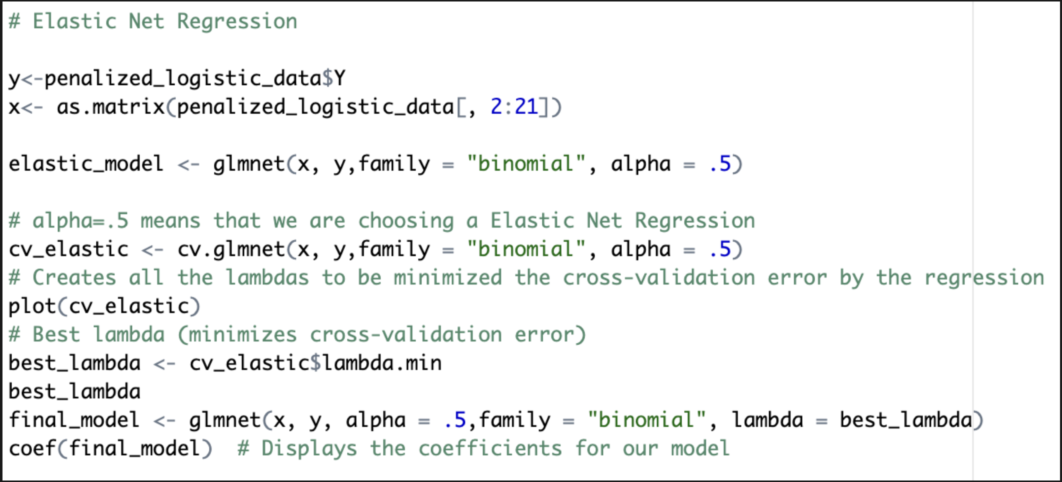
From the code we obtain the coefficients below. Note that none of these coefficients are zero.



```
> final_model <- glmnet(x, y,family = "binomial", alpha = 0, lambda = best_lambda)
> coef(final_model)  # Displays the coefficients for our model
21 x 1 sparse Matrix of class "dgCMatrix"
                        s0
(Intercept) -0.383214071
X1           0.992010180
X2          -0.728815228
X3           0.366836952
X4           0.303621797
X5          -0.178465032
X6           0.129981652
```

Implementation of the Elastic Net Regression

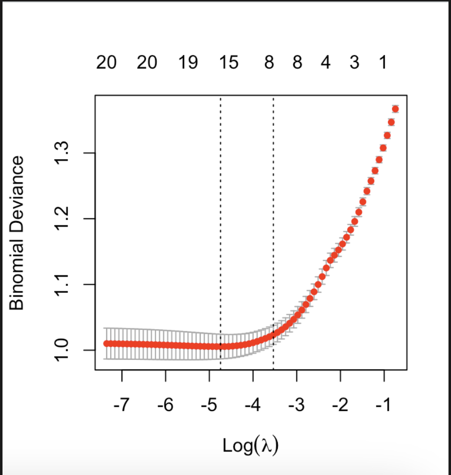
The only difference for the implementation for a Elastic Net Regression is changing the alpha to .5



```r
# Elastic Net Regression

y<-penalized_logistic_data$Y
x<- as.matrix(penalized_logistic_data[, 2:21])

elastic_model <- glmnet(x, y,family = "binomial", alpha = .5)

# alpha=.5 means that we are choosing a Elastic Net Regression
cv_elastic <- cv.glmnet(x, y,family = "binomial", alpha = .5)
# Creates all the lambdas to be minimized the cross-validation error by the regression
plot(cv_elastic)
# Best lambda (minimizes cross-validation error)
best_lambda <- cv_elastic$lambda.min
best_lambda
final_model <- glmnet(x, y, alpha = .5,family = "binomial", lambda = best_lambda)
coef(final_model)  # Displays the coefficients for our model
```

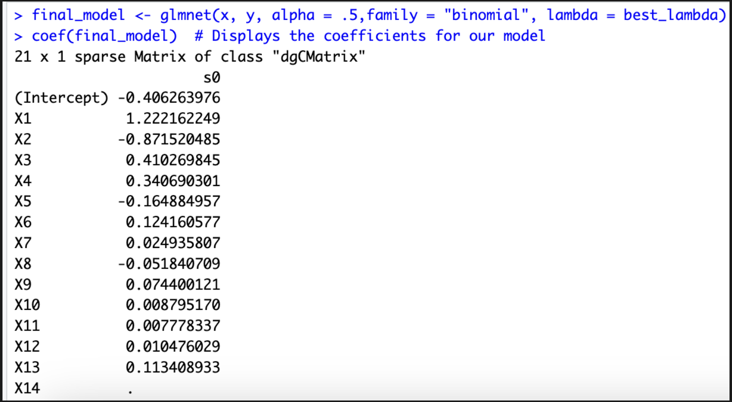
Results of the implementation of the Elastic Net Regression

This code outputs the plot:



Results of the implementation of the Elastic Net Regression

This code outputs the coefficients of our model in this form. Note that the coefficients may or may not be zero.



```
> final_model <- glmnet(x, y, alpha = .5,family = "binomial", lambda = best_lambda)
> coef(final_model)  # Displays the coefficients for our model
21 x 1 sparse Matrix of class "dgCMatrix"
                     s0
(Intercept) -0.406263976
X1           1.222162249
X2          -0.871520485
X3           0.410269845
X4           0.340690301
X5          -0.164884957
X6           0.124160577
X7           0.024935807
X8          -0.051840709
X9           0.074400121
X10          0.008795170
X11          0.007778337
X12          0.010476029
X13          0.113408933
X14          .
```

Summary

	Original	Lasso	Ridge	Elastic Net
Row 1	1.5	1.2	.99	1.22
Row 2	-1.2	-.8	-.72	-.87
Row 3	.8	.3	.36	.4
Row 4	.5	.3	.3	.34
Row 5	-.3	-.07	-.17	-.16
Row 6	.2	.09	.12	.12

Table: Comparison of Original, Lasso, Ridge, and Elastic Net Models

All other values in the regressions have an absolute value smaller than 0.11.

References

1. Hastie, T., Qian, J., Tay, K. (n.d.). Glmnet vignette. Retrieved from https://glmnet.stanford.edu/articles/glmnet.html

2. Hastie, T., Tibshirani, R., Wainwright, M. (2015). Statistical learning with sparsity: The lasso and generalizations (p. 10). Chapman Hall/CRC.

3. IBM. (n.d.). Ridge regression. Retrieved from https://www.ibm.com/think/topics/ridge-regression Nagpal, A. (2024, October 3). L1 and L2 regularization methods, explained. Built In. Retrieved from https://builtin.com/data-science/l2-regularization

4. Kassambara, A. (2018, November 3). Penalized regression essentials: Ridge, lasso elastic net. STHDA. Retrieved from https://www.sthda.com/english/articles/37-model-selection-essentials-in-r/153-penalized-regression-essentials-ridge-lasso-elastic-net/

Refrences

5. DataCamp. (2024, December 4). Loss functions in machine learning explained. Retrieved from https://www.datacamp.com/tutorial/loss-function-in-machine-learning

6. Kassambara, A. (2018, November 3). Penalized logistic regression essentials in R: Ridge, lasso and elastic net. STHDA. Retrieved from https://www.sthda.com/english/articles/36-classification-methods-essentials/149-penalized-logistic-regression-essentials-in-r-ridge-lasso-and-elastic-net/