1 Linear Model Selection and Regularization

Linear models are often simple and easy to interpret at the cost of having high bias if the relationship in the data is not linear. Some considerations about linear models

- If $n >> p$, least square estimates often have less variance. If n is larger than p, then least square estimates can have some variance. While if $n < p$, we are looking at non unique solutions which can cause lot of variation in the test predictions.
- It is often the case that many of the predictors do not have a relationship with the response. Hence, it is a good idea to remove those and make the model more interpretable at the cost of some bias. Least square estimates almost never give zero coefficients.

There are major ways in which the number of variables in the model can be reduced

- Selecting a **subset of variables** that go well with the response. This itself can be done by forward selection, backward elimination etc.
- **Shrinking** some of the **coefficients** to zero. This is a great help in reducing the variance of the predictions.
- **Dimension Reduction** helps in projecting the p predictors onto a M dimensional space where $M < p$. This utilizes linear combinations to create a set of new features.

1.1 Subset Selection

1.1.1 Best Subset Selection

This is a naive approach that essentially tries to find the best model among 2^p models that are trained on all possible subsets of the p variables. As we increase the subset of variables, the training error will monotonically decrease whereas the same cannot be said for the test error. A number of criteria like test MSE, R^2, AIC etc can be used to pick the models.

In case of classification models, similar argument holds and a more general error metric *deviance* can be used. *Deviance* is defined as $-2 * \log \textit{likelihood}$ of the data. The smaller the *deviance*, the better the model fit.

The huge search space presented by this approach easily overfits as the search space presents more opportunities to find better fits. However, this causes a higher variance in the predictions on future data and can possibly also have higher test error.

1.1.2 Forward Stepwise Selection

This is a greedy approach that significantly shrinks the search space being checked (in comparison to the best subset selection approach).
Forward Stepwise Selection Algorithm

1. Let M_0 denote the null model, i.e., the model with no predictors
2. For $k = 0, 1, \dots, p-1$
   (a) Consider all $p-k$ models formed by adding a single predictor to the model M_k
   (b) Select the best model M_{k+1} among the $p-k$ models on the basis of the error metric
3. From the models $M_0, M_1, \dots, M_p$, select the one with the lowest cross validation error on the evaluation choosing the appropriate error metric

This approach effectively has reduced the search space from 2^p to $1 + p(p+1)/2$. Although, now it is not guaranteed that the model selected will be the best one among 2^p.

1.1.3 Backward Stepwise Selection

This approach is the opposite of forward stepwise selection. We recursively reduce the number of variables in our model.

1. Let M_p denote the complete model, i.e., the model with all p predictors
2. For $k = p, p-1, \ldots, 1$
   (a) Consider all $k-1$ models that keep all but one predictors in the current model M_k
   (b) Among these, select the best model M_{k-1} with the lowest error
3. From the models $M_0, M_1, \ldots, M_p$, select the one with the lowest cross validation error on the evaluation choosing the appropriate error metric

The number of models explored is same as the forward stepwise method.
A hybrid approach is usually selected where we start with the usual forward selection method, but while adding variables, we do not add a variable if it does not give significant improvement. Another approach can be to remove redundant variables using p-test at every step of forward selection.

1.2 Metrics for evaluating Subset Models

In linear models, as we add more variables, the training error usually monotonically decreases. Test error may not behave in the same way. When training a model, the coefficients obtained are specific for minimizing the training error and hence will have less bias in comparison to the test error.
Hence, subset evaluation using training error will usually favour models with more number of variables. To overcome this

- Correct the training error estimate to correctly calculate test error
- Use a validation test or k-fold validation for better estimate of test error

1.2.1 C_p Estimate

For a least square fitted model,

$$C_p = \frac{1}{n}(RSS + 2p\hat{\sigma}^2)$$

p is the number of predictors and $\hat{\sigma}^2$ is the estimate of the error associated with each observation. This is typically evaluated using the model built on all p predictors.
Clearly, as p increases, we are penalizing the model more to compensate for the decrease in the training RSS. When $\hat{\sigma}$ is an unbiased estimate of σ, we can show that this is infact an unbiased estimate of the test MSE.

1.2.2 Akaike Information Criterion (AIC)

AIC is defined for a large class of models fit by the maximum likelihood estimate.
For least squares fit in linear models, the errors are assumed to be gaussian and thus, AIC and least squares mean the same thing. For this case

$$AIC = \frac{1}{n\hat{\sigma}^2}(RSS + 2p\hat{\sigma}^2)$$

where we have omitted an additive constant for the sake of simplicity.
For least squares models, C_p and AIC are proportional to each other.

1.2.3 Bayesian Information Criterion (BIC)

BIC is derived from a Bayesian point of view, but ends up looking similar to the above defined errors.
For least squares error without constants, the BIC is

$$BIC = \frac{1}{n\hat{\sigma}^2}(RSS + \log(n)p\hat{\sigma}^2)$$

Note that the $\log(n)$ term will put a heavier weight on the error term for large p. Hence, BIC will tend to select models with lower number of variables in comparison to say C_p.

1.2.4 Adjusted R^2

Recall that R^2 is defined as $1 - RSS/TSS$. Adjusted R^2 is

$$Adjusted\ R^2 = 1 - \frac{\frac{RSS}{n-p-1}}{\frac{TSS}{n-1}}$$

This adjusted R^2 might increase or decrease when adding variables due to the terms corresponding to p. The intuition is that, after the correct number of variables have been identified, the decrease in RSS is less in comparison to the decrease in $n-p-1$ which will slightly increase the Adjusted R^2.

1.3 Shrinkage Methods

Instead of using a subset of predictors, we can also use all of the predictors and shrink the coefficients towards zero. This approach significantly reduces the variance in the model estimates. The famous ones here are *Ridge Regression* and *Lasso Regression*.

1.3.1 Ridge Regression

Ridge Regression is very similar to the least square estimate for linear regression except that we add a term corresponding to the squared sum of the regression coefficients in the error.

$$\begin{aligned} error &= RSS + \lambda \sum_{j=1} p\beta_j^2 \\ &= (Y - X\beta)^T(Y - X\beta) + \lambda\beta^T\beta \end{aligned}$$

λ is a tuning parameter that needs to be chosen separately. It acts as a weight between the error in the data and how large are the regression coefficients. It is also known as the shrinkage penalty.
Note that we will not include the intercept term in shrinkage because it is simply the mean estimate of the model when all the predictors are zero and may not necessarily zero.

Using least squares estimate,

$$\begin{aligned} error &= (Y - X\beta)^T(Y - X\beta) + \lambda\beta^T\beta \\ \frac{\partial error}{\partial\beta} &= 0 \\ \implies 0 &= -Y^TX - Y^TX + \beta^TX^TX + \beta^TX^TX + \lambda\beta^T + \lambda\beta^T \\ \beta^T(X^TX + \lambda I) &= Y^TX \\ (X^TX + \lambda I)^T\beta &= X^TY \\ (X^TX + \lambda I)\beta &= X^TY \\ &\boxed{\beta = (X^TX + \lambda I)^{-1}X^TY} \end{aligned}$$

$\lambda = 0$ will result in the simple least squares regression while $\lambda \rightarrow \inf$ will force the coefficients to go towards zero.

As is clear from the formula, the error term is sensitive to the actual scale of the coefficients which is ultimately dependent on the predictors themselves. In a simple least squares regression, the coefficients will scale up and down depending on how the data is scaled. The same is not true for Ridge Regression.
Hence when using **Ridge Regression, it is always advisable to *standardize* the predictors** before training the model using

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1} n(x_{ij} - \bar{x}_j)^2}}$$

The success of Ridge Regression is based in the **bias variance tradeoff**. If the data is linear, simple linear regression will have a very low bias but high variance, making it sensitive to the training data. As λ is introduced, it forces the model to have less flexibility by reducing the coefficiets value and subsequently their power on the prediction. This causes a reduction in the variance at expense of slight increase in bias. However, this trend is not monotonic with increasing λ and the appropriate value must be chosen based on the errors observed.

1.3.2 Lasso Regression

Notice that Ridge Regression will try to reduce the value of some of the coefficients, but it will never set them to exactly zero. Hence, we will end up using all the p predictors in the model which may not be interpretable if the value of p is large.

Lasso Regression comes over this disadvantage by defining the error as

$$error = RSS + \lambda \sum_{j=1}^{p} \mid \lambda_j \mid$$

When λ **is sufficiently large, Lasso Regression forces some of the variables to be exactly zero**. This is very useful in reducing the subset of variables that we use in the model, thereby increasing model interpretability.

1.3.3 Alternative Formulation to Ridge and Lasso Regression

These regressions can also be considered as solving a constrained optimisation problem

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 \right\} \quad \text{subject to} \qquad \sum_{j=1}^{p} \beta_j^2 \leq s$$

$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 \right\} \quad \text{subject to} \qquad \sum_{j=1}^{p} \mid \beta_j \mid \leq s$$

for Ridge and Lasso regression respectively. This holds true because these regressions are effectively trying to limit the size of the coefficients themselves. For $\lambda = 0$, we have no bound on the size and s in equations above is close to inf. As λ increases, s will start to decrease and be 0 for $\lambda \rightarrow \inf$.

The equations above can be interpreted as finding the minimum RSS among the points that lie inside the geometric shapes defined by the constraints. For $p = 2$, Ridge defines a circle $\beta_1^2 + \beta_2^2 \leq s$ and Lasso defines a diamond $\mid \beta_1 \mid + \mid \beta_2 \mid \leq s$.

1.3.4 Variable Selection Property of Lasso Regression

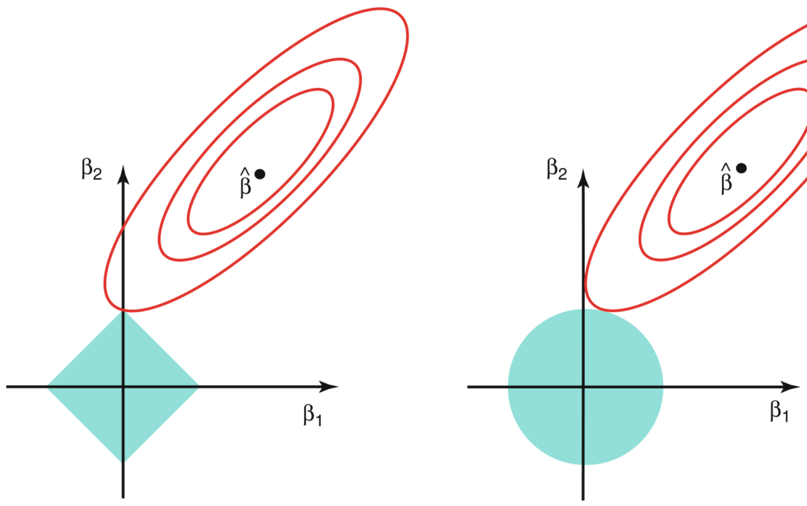

Figure 1: For $p = 2$, the left and right images correspond to Lasso and Ridge Regression.

In the figure 1, $\hat{\beta}$ corresponds to the least squares estimate of β and the contours in red colour show the same value of RSS. The blue coloured regions correspond to the constraints defined above (diamond for lasso and circle for ridge).
Clearly, circle being a smooth shape, the lowest RSS contour will not usually touch it at one of the axis points. However, for the sharp diamond shape, the least RSS value is likely to be encountered along the axis. The shapes of the constraint regions can be controlled through λ and for lasso, more constrained models (small s or higher λ) will cause more of the coefficients to be zero. The argmuents are very well valid in higher dimensions as well.

1.3.5 Bayesian Interpretation

Lasso and Ridge Regression are also natural solutions when the the coefficients are assumed to have certain specific priors.

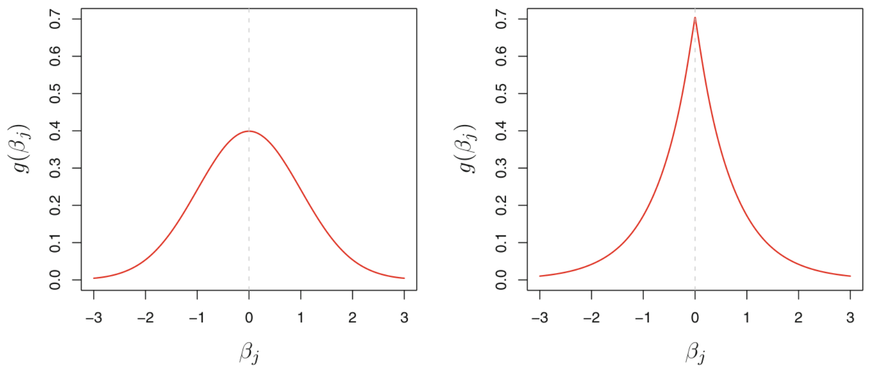

Figure 2: Gaussian prior on left and double exponential on the right

$$p(\beta|X,Y) \propto f(Y|X,\beta)p(\beta|X) = f(Y|X,\beta)p(\beta) \qquad \text{assuming X is constant}$$
$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$
$$\text{Assume, } p(\beta) = \prod_{j=1}^{p} \beta_p \qquad \text{for some density function } g$$

The following are observed for different priors on β

- When the density function of β_j is assumed to be a standard normal, the posterior is same as solving the ridge regression error function
- When the density function is assumed to be a double exponential, the posterior is the same as solving lasso error function

From the visuals of the priors in figure 2, double exponential is steeply peaked at zero, which clearly implies that the prior itself assumes some of the coefficients are likely zero. On the other hand, the gaussian priod is flatter and does not necessarily require the coefficients to be zero.

1.4 Dimension Reduction Methods

The methods seen above achieve reduction in variance by either reducing the number of variables being used for estimation, or reduce the value of coefficients of those variables.
Dimensionality reduction uses a **linear map** to convert the original p variables to M variables where $M < p$.

$$Z_m = \sum_{j=1}^{p} \phi_{jm} X_j \qquad \text{where } \phi_{jm} \text{ are constants}$$
$$RSS = \sum_{i=1}^{n} (y_i - (\theta_0 + \sum_{m=1}^{M} \theta_m Z_m))^2 \qquad \text{where } \theta_m \text{ are linear regression coefficients for } Z_m$$

This new formulation restructures the original least squares formula. The whole process now is

- Obtain the linear map using techniques like PCA
- Use least squares on the transformed predictors to obtain regression estimates

1.4.1 Principal Components Analysis (PCA)

PCA is one of the most common methods used for dimensionality reduction. It is based on the principle of finding those directions that **maximize the variance of data**. The intuition behind finding this direction is that this separates the data best given the high variance. Hence, the performance of a classifier/regressor would be better if the training was done using the transformed predictors.

Assume that we want to find M components for PCA, where $M <= p$. Then,

1. Find the direction/projection of the data that gives the maximum variance under the constraint $\sum_{j=1}^{p} \phi_{jm}^2 = 1$ (normalized vector) for obtaining the m^{th} transformed predictor
2. If the total components found is $< M$, repeat step 1 with the added constraint that the next component has zero correlation with the previous component

This constraint of finding components with zero correlation essentially means that in the multidimensional space, we are finding a set of **orthogonal vectors**. The order of choosing the new predictors is in decreasing order of the information they contain. Thus, the first predictor will now containt the most information.

Derivation of PCA It is a good idea to **standardize the variables before running PCA**. Not only is it easier to derive the components, but they are also unaffected by the scale of the individual variables (since we know that scaling a variable multiplies the variance by square of the scaler and PCA is all about finding maximum variances !).

After setting the the mean of all predictors to zero and standard deviation to one, the components can be derived by considering the following optimization problem

$$\underset{\phi_{11},\dots,\phi_{p1}}{\text{maximize}}\{\frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p}\phi_{j1}x_{ij})^2\} \text{ subject to } \sum_{j=1}^{p}\phi_{j1}^2 = 1$$

$$\text{Or, } \underset{\Phi_1}{\text{maximize}}\,\frac{1}{n}Z_1^TZ_1 \text{ subject to } \sum_{j=1}^{p}\phi_{j1}^2 = 1$$

$$\text{Where } Z_1 = X\Phi_1$$

$$\text{Define the Lagrangian } L(\Phi_1,\lambda) = \frac{1}{n}(\Phi_1^TX^TX\Phi_1) - \lambda(\Phi^T\Phi - 1)$$

$$\frac{\partial L(\Phi_1,\lambda)}{\partial\lambda} = 0, \text{ and } \frac{\partial L(\Phi_1,\lambda)}{\partial\Phi_1} = 0$$

$$\text{Giving, } 2\frac{1}{n}(\Phi_1^TX^TX) - 2\lambda\Phi_1^T = 0 \text{ and } \Phi_1^T\Phi_1 - 1 = 0$$

$$\frac{X^TX}{n}\Phi_1 = \lambda\Phi_1$$

$$\text{Since all } X_i \text{ are centered, } \frac{X^TX}{n} \text{ is the covariance matrix } \Sigma_X$$

$$\Sigma_X\Phi_1 = \lambda\Phi_1 \text{ with } \Phi_1^T\Phi_1 = 1$$

Which is nothing but the **Eigenvectors of the Covariance Matrix of X**. Notice that the maximization is achieved through the eigenvector with the highest eigenvalue. Since Σ_X is positive semi-definite, all the eigenvalues will be ≥ 0.
Furthermore, we know that all eigenvectors are orthogonal to each other, and hence they will also satisfy the condition that all the linear mapping vectors are not correlated to each other.
If we try to find the coefficients for Z_2, we are looking at the exact same optimization, but with the additional constraint that Φ_2 is not correlated to Φ_1. This will yield the second eigenvector of the covariance matrix.
Hence, **the linear maps for obtaining the PCA transformations are nothing but the eigenvectors of the covariance matrix Σ_X of the original data X**. We have a total of p eigenvectors and thus, the maximum components obtainable is also p.

Explained Variance The variance explained by the m^{th} component is nothing but the ratio of variance of Z_m and total variance of the data. Mathematically this is

$$\begin{aligned}\text{Explained variance of component } m &= \frac{\frac{1}{n}\sum_{i=1}^{n}(\sum_{j=1}^{p}\phi_{jm}x_{ij})}{\frac{1}{n}\sum_{i=1}n\sum_{j=1}px_{ij}^2} \\ &= \frac{\frac{1}{n}Z_m^TZ_m}{tr(\frac{X^TX}{n})} \\ &= \frac{\Phi_m^T\frac{X^TX}{n}\Phi_m}{tr(\Sigma_X)}\end{aligned}$$

However, note that in the derivation, Φ_m is the eigenvector and thus, $(X^TX)\Phi_m = \lambda_m\Phi_m$ and $\Phi_m^T\Phi_m = 1$. Substituiting in the above equation,

$$\begin{aligned}\text{Explained variance } &= \frac{\lambda_m}{tr(\Sigma_X)} \\ \text{or, } \boxed{\text{Explained variance } = \frac{\lambda_m}{\lambda_1 + \cdots + \lambda_p}}\end{aligned}$$

Where the last equation comes from the fact that the trace of a matrix is simply the sum of it's eigenvalues.
A **Scree Plot** is a plot between the explained variance and the index of the prinicple components. It's cumulative version can be used to determine the number of components to keep on the basis of how much of the total variance we want to explain. The elbow point of the Scree Plot can also help determine the components at which the explained variance drops significantly.

Principal Components Regression (PCR) is simply using some m out of M components to perform linear regression. This approach will typically work best when a few components of the PCA sufficiently explain the whole data. This way we reduce the variance at the cost of slight change in bias. PCR can also be viewed as a continuous version of Ridge Regression.
Partial Least Squares This method is closely related to PCA/PCR. In the previous methods, an unsupervised approach was used to project the matrix M onto a lower dimensional space. This transformation did not take Y, the response, into consideration. PLS will incorporate Y as well for finding the transformations.
The algorithm is as follows

1. Calculate the coefficients for the first component as the coefficient obtained by regressing Y on X_j
2. Using this component, regress X_j's on Z_1 and get the residuals. These are the unexplained components of the variables.
3. Calculate Z_2 using Y and these residuals
4. Repeate the process above till M components are obtained

Though this procedure looks slightly more involved, it performs not better than lasso regression and PCR in practice. Although this method reduces bias, it also leads to quite an increase in variance as well, making the method not very useful.

1.5 Curse of Dimensionality

Most of the methods discuss here **work well when** $n >> p$. There can be many reasons why the model may not perform well in higher dimensions, but the major one will be the fact that as more and more dimensions are added to the model, chances of overfitting increase and so do the chances that the additional variables are simply noise.
We refer to the problem of training a model a high dimensional problem when $p > n$, or we have more data than number of predictors. Note that it is easy to obtain $R^2 = 1$ in such a setting which consequently leads to $\hat{\sigma}^2 \approx 0$. Metrics like C_p, AIC, BIC become useless.
Hence, in higher dimensional settings, it is important to obtain model performance on unseen data as there is a good chance of obtaining perfect results on the training set. Metrics associated with the training set can thus prove to be misleading.

Intuition for the curse of dimensionality can be easily obtained in the context of KNN.
Suppose the data is uniformly distributed along any dimension considered. Let the model be built in such a way that when making predictions, it uses 10% of the data along all dimensions. In the case of a single dimension, we need 10% of the data. In the case of 10 dimensions, we will need to check $10\%^{10} = 10^{-8}\%$ of data. Clearly this number grows as we consider more and more dimensions.
Thus, as the number of dimensions grow, we require substantially more data as the data observable in the neighborhood of the point is extremely small. This illustrates the fact that in higher dimensions, having less data will lead to a poor model.