Ch. 6. Linear Model Selection and Regularization

The linear model has distinct advantages in terms of inference and, on real-world problems, is often surprisingly competitive in relation to non-linear methods. Here we discuss some ways in which the simple linear model can be improved, by replacing plain least squares fitting with some alternative fitting procedures.

Alternative fitting procedures can yield better *prediction accuracy* and *model interpretability*.

- *Prediction Accuracy*: Provided that the true relationship between the response and the predictors is approximately linear, the least squares estimates will have low bias. If n >> p -- that is, if n, the number of obs, is much larger than p, the number of variables --- then the least squares estimates tend to also have low variance, and hence will perform well on test obs. However, if **n is not much larger than p**, then there can be **a lot of variability in the least squares fit, resulting in overfitting** and consequently poor predictions on future obs not used in model training. And if **p > n**, then there is no longer a unique least squares coeff to estimate: **the variance is infinite so the method cannot be used at all**. By **constraining or shrinking the estimated coefficients**, we can often substantially r**educe the variance** at the cost of a **negligible increase in bias**. This can lead to substantial improvements in the accuracy with which we can predict the response for obs not used in model training.
- *Model interpretability*: It is often the case that **some or many of the variables** used in a multiple regression model **are in fact not associated with the response**. Including such irrelevant variables leads to unnecessary complexity in the resulting model. By removing these variables -- that is, by setting the corresponding coefficient estimates to zero -- we can obtain a model that is more easily interpreted. Now least squares is extremely unlikely to yield any coeff estimates that are exactly zero. Here we see some approaches for automatically performing feature selection or variable selection -- that is, for excluding irrelevant variables from a multiple regression model.

Here we will discuss three important classes of methods alternative to least squares:

- *Subset selection*. This approach involves identifying a subset of the p predictors that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.
- *Shrinkage*. This approach involves fitting a model involving all p predictors. However, the estimated coefficients are shrunken towards zero relative to the least squares estimates. This shrinkage (aka **regularization**) has the effect of reducing variance. Depending on what type of shrinkage is performed, some of the coefficients may be estimated to be exactly zero. Hence, shrinkage methods can also perform variable selection.
- *Dimension Reduction*. This approach involves **projecting** the p predictors into a **M-dimensional subspace**, where **M < p**. This is achieved by computing M different

linear combinations, or projections, of the variables. Then these M projections are used as predictors to fit a linear regression model by least squares.

These concepts apply not only to the linear model for regression but also to other methods.

Best subset selection

To perform *best subset selection*, we fit a separate least squares regression for each possible combination of the p predictors. We then look at all of the resulting models, with the goal of identifying the one that is best.

best subset selection Algorithm:

1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each obs.
2. For k = 1, …, p:
    a. Fit all $\binom{p}{k}$ models that contain exactly k predictors.
    b. pick the best among these models $\binom{p}{k}$, and call it M_k. Here best is defined as having the smallest RSS, or equivalently largest R^2.
3. Select a single best model from among M_0, …, M_p using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Now we must simply choose one among these p + 1 best models. This task must be performed with care, because the RSS of these p + 1 models decreases monotonically, and the R^2 increases monotonically, as the number of features included in the models increases. The problem is that a low or high indicates a model with a low training error, whereas we are interested in a model with a low test error. Therefore, in step 3, we use cross-validated prediction error C_p, BIC, or adjusted R^2.

While the best subset selection is a simple and conceptually appealing approach, it suffers from computational limitations. The number of possible models that must be considered grows rapidly as p increases. In general there are 2^p models that involve subsets of p predictors. So if p = 10, then there are approximately 1000 possible models to be considered, and if p = 20, then there are over one million possibilities. Consequently best subset selection becomes computationally infeasible for values of p greater than ~40, even with extremely fast computers.

Forward Stepwise Selection

Forward Stepwise Selection considers a much smaller set of models. It begins with a model containing no predictors, and then adds predictors to the model, one-at-a-time, until all of the predictors are in the model. In particular, at each step the variable that gives the greatest additional improvement to the fit is added to the model.

Forward stepwise selection Algorithm:

1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each obs.
2. For k = 0, …, p - 1:
    a. Consider all p - k models that augment the predictors in M_k with one additional predictor.
    b. Choose the *best* among these p - k models, and call it M_{k+1}. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among M_0, …, M_p using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Unlike best subset selection, which involved fitting 2^p models, forward stepwise selection involves fitting one null model, along with 1 + p(p + 1)/2 models. For example, with p = 20: best sub. selection → 1,048,576 models have to be fitted, forward stepwise selection → 211 models have to be fitted.

Though forward stepwise selection tends to do well in practice, it is not guaranteed to find the best possible model out of all 2^p models containing subsets of the p predictors.

Backwards Stepwise Selection

Unlike forward stepwise selection, it begins with the full least squares model containing all p predictors, and then iteratively removes the least useful predictor, one-at-a-time.

Forward stepwise selection Algorithm:

1. Let M_p denote the full model, which contains all p predictors.
2. For k = p, p -1,…, 1:
    a. Consider all k models that contain all but one of the predictors in M_k, for a total of k - 1 predictors.
    b. Choose the *best* among these k models, and call it M_{k-1}. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among M_0, …, M_p using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Like forward stepwise selection, the backward selection approach searches through only 1 + p(p + 1)/2 models. Also like forward stepwise selection, backward stepwise selection is not guaranteed to yield the best model containing a subset of the p predictors.

Backward selection can only be applied if the number of samples n is larger than the number of variables p (so that the full model can be fit). In contrast, forward stepwise can be used even when n < p, and so is the only viable subset method when p is very large.

Hybrid Approaches

The best subset, forward stepwise, and backward stepwise selection approaches generally give similar but not identical models. As another alternative, hybrid versions of forward and backward stepwise selection are available, in which variables are added to the model sequentially, in analogy to forward selection.
However, after adding each new variable, the method may also remove any variables that no longer provide an improvement in the model fit. Such an approach attempts to more closely mimic best subset selection while retaining the computational advantages of forward and backward stepwise selection.

Choosing the Optimal Method

In order to implement these methods, we need a way to determine which of these models is *best*. As mentioned, the model containing all of the predictors will always have the smallest RSS and the largest R^2.
In order to select the best model with respect to test error, we need to estimate this test error. Two common approaches:

1. We can indirectly estimate test error by making an adjustment to the training error to account for the bias due to overfitting.
2. We can directly estimate the test error using either a validation set approach or a cross-validation approach.

A number of techniques for **adjusting** the training error for the model size are available. These approaches can be used to select among a set of models with different numbers of variables:

For a fitted least squares model containing d predictors, the **C_p** estimate of test MSE is computed using the equation

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

Where $\hat{\sigma}^2$ is an estimate of the variance of the error ϵ associated with each response measurement. Typically it is estimated using the full model containing all predictors. Essentially the C_p statistic adds a penalty of $2d\hat{\sigma}^2$ to the training RSS in order to adjust for the fact that the training error tends to underestimate the test error. Clearly the penalty increases as the number of predictors in the model increases. C_p tends to be an unbiased estimate of test MSE. As a consequence, the C_p statistic tends to take on a small value for models with a low test error.

The **AIC** criterion is defined for a large class of models fit by maximum likelihood. In the case of the linear model with Gaussian errors, maximum likelihood and least squares are the same thing. In this case AIC is given by

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

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

BIC is derived from a Bayesian point of view, but ends up looking similar to C_p and AIC as well. For the least squares model with d predictors, the BIC is, up to irrelevant constants, given by

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

Like C_p the BIC will tend to take on a small value for a model with a low test error, but it generally (if n > 7) places an heavier penalty on models with many variables, and hence results in the selection of smaller models than C_p.

Adjusted R^2 is defined as

$$Adjusted\ R^2 = 1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$$

Unlike the other statistics presented here, a large value of adjusted R^2 indicates a model with a small test error.
The intuition behind the adjusted R^2 is that, once all of the correct variables have been included in the model, adding additional noise variables will lead to only a small decrease in RSS. Since adding noise variables leads to an increase in d, such variables will lead to an increase in RSS/(n - d -1)and consequently a decrease in the adjusted R^2. Therefore, in theory, the model with the largest adjusted R^2 will have only correct variables and no noise variables.

Validation and Cross-Validation

As an alternative to the approaches just discussed, we can directly estimate the test error using the validation set and cross-validation methods discussed in Chapter 5. We can compute the validation set error or the cross-validation error for each model under consideration and then select the model for which the resulting estimated test error is smallest. This procedure has an advantage relative to AIC, BIC, C_p and adjusted R^2, in that it provides a direct estimate of the test error, and makes fewer assumptions about the true underlying model. It can also be used in a wider range of model selection tasks, even in cases where it is hard to pinpoint the model degrees of freedom (e.g. the # of predictors in the model) or hard to estimate the error variance $\boldsymbol{\sigma}^2$.

In general, it can happen that several models present similar AIC, or cross-validation errors (I picked two metrics, it's valid for all the others). In this case we can select a model using the *one-standard-error rule*. We first calculate the standard error of the estimated test MSE for each model size, and then select the smallest model for which the estimated test error is within one standard error of the lowest point on the curve. The rationale here is that if a set of models appear to be more or less equally good, then we might as well choose the simplest model -- that is, the model with the smallest number of predictors.

Ridge Regression

Ridge regression is very similar to least squares, except that the coefficients are estimated by minimizing a slightly different quantity. In particular, the ridge regression coefficient estimates $\widehat{\beta}^R$ are the values that minimize

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \lambda\sum_{j=1}^{p}\beta_j^2$$

where $\lambda \geq 0$ is a *tuning parameter*, to be determined separately.

As with least squares, ridge regression seeks coefficient estimates that fit the data well, by making the RSS small. However, the second term, $\lambda\sum_{j=1}^{p}\beta_j^2$, called a *shrinkage penalty*, is small when are close to zero, and so it has the effect of shrinking the estimates of towards zero.
In statistical parlance, ridge regression uses an ℓ_2 penalty.

The λ tuning parameter serves to control the relative impact of these two terms on the regression coefficient estimates. When $\lambda = 0$, the penalty term has no effect, and ridge regression will produce the least squares estimates. However, as $\lambda \to \infty$, the impact of the shrinkage penalty grows, and the ridge regression coefficient estimates will *approach* zero.

Ridge regression will produce a different set of coefficient estimates, $\widehat{\beta}_\lambda^R$, for each value of λ.
Selecting a good value of λ is critical.

Note that the shrinkage penalty is not applied to the intercept $\boldsymbol{\beta}_0$ as we want to shrink the estimated association of each variable with the response, not the intercept which is simply a measure of the mean value of the response when the $x_{i1}, \ldots, x_{ip} = 0$.
The standard least squares coefficient estimates are *scale equivariant*. In contrast, the ridge regression coefficient estimates can change substantially when multiplying a given predictor by a constant.

$X_j\widehat{\beta}_{j,\lambda}^R$ will depend not only on λ, but also on the scaling of the jth predictor. In fact, the value of $X_j\widehat{\beta}_{j,\lambda}^R$ may even depend on the scaling of the *other* predictors. Therefore, it is best to apply ridge regression after *standardizing the predictors* by dividing each observation x_{ij} by the estimated standard deviation of the jth predictor, so that they are all on the same scale and all of the standardized predictors will have a standard deviation of one.

Ridge regression's advantage over least squares is rooted in the *bias-variance trade-off*. As λ increases, the flexibility of the ridge regression fit decreases, leading to decreased variance but increased bias.

In general, in situations where the relationship between the response and the predictors is close to linear, the least squares estimates will have low bias but may have high variance. This means that a small change in the training data can cause a large change in the least squares coefficient estimates.
High variance situations are for example when p is close to n.
If p > n, LSE cannot even be used. In these cases ridge regression works better than LSE.
Ridge regression has a computational advantage over best subset selection and other subset based methods: for any fixed value of λ, ridge regression only fits a single model, and the model fitting procedure can be performed quite quickly.
In fact, the computations required to solve ridge regression *simultaneously for all values of* λ, are almost identical to those for fitting a model using least squares.

The Lasso

Ridge regression does have one obvious disadvantage. Unlike best subset, forward stepwise, and backward stepwise selection, which will generally select models that involve just a subset of the variables, ridge regression will include all p predictors in the final model. The penalty $\lambda \sum_{j=1}^{p} \beta_j^2$ will shrink all of the coefficients towards zero, but it will not set any of them exactly at zero (unless λ = ∞).

This may not be a problem for prediction accuracy, but it can create a challenge in model interpretation in settings in which the number of variables p is quite large.

The *lasso* is a relatively recent alternative to ridge regression that overcomes this disadvantage. The lasso coefficients $\widehat{\beta}_{\lambda}^{L}$, minimize the quantity:

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

Comparing it to the ridge regression, we see that the only difference is that the term in the ridge regression penalty has been replaced by in the lasso penalty

In statistical parlance, the lasso uses an ℓ_1 penalty.

As with ridge regression, the lasso shrinks the coefficient estimates towards zero. However, in the case of the lasso, the ℓ_1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large.

Hence, much like best subset selection, the lasso performs variable selection. As a result, models generated from lasso are generally much easier to interpret than those produced by ridge regression. We say that the lasso yields sparse models -- that is, models that involve only a subset of the variables.

However, which method leads to better prediction accuracy? It **depends**, neither ridge regression or lasso will universally dominate the other.
In general, one might expect the **lasso** to perform better in a setting where a relatively small number of predictors have substantial coefficients, and the remaining predictors have coefficients that are very small or that are equal to zero.
Ridge regression will perform better when the response is a function of many predictors. However, the number of predictors that is related to the response is never known a priori for real data sets. A technique such as cross-validation can be used in order to determine which approach is better on a particular data set.

Ridge regression more or less shrinks every dimension of the data by the same proportion (so it can never go to zero), whereas the lasso more or less shrinks all coefficients towards zero by a similar amount, and sufficiently small coefficients are shrunken all the way to zero.

Selecting the Tuning Parameter

Implementing ridge regression and the lasso requires a method for selecting a value for the tuning parameter λ. Cross-validation provides a simple way to tackle this problem. We choose a grid of λ values, and compute the cross-validation error for each value of λ. We then select the tuning parameter value for which the cross-validation error is smallest. Finally, the model is re-fit using all of the available observations and the selected value of the tuning parameter.

Dimension Reduction Methods

We now explore a class of approaches that *transform* the predictors and then fit a least squares model using the transformed variables.

Let Z_1, Z_2, … Z_m represent M < p *linear combinations* of our original p predictors. That is: $Z_m = \sum_{j=1}^{p} \phi_{jm} X_j$ for some constraints $\Phi_{1m}, \Phi_{2m}, \dots, \Phi_{pm}$, m = 1,...,M. We can then fit the linear regression model $y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \varepsilon_i$ using least squares.

If the constraints $\Phi_{1mk}, \Phi_{2m}, \dots, \Phi_{pm}$ are chosen wisely, then such dimension reduction approaches can often outperform least squares regression.

The term *dimension reduction* comes from the fact that this approach reduces the problem of estimating the p + 1 coefficients $\square_0$, $\square_1$, ..., $\square_p$ to the simpler problem of estimating the M + 1 coefficients θ_0, θ_1, …, θ_M where M < p.
In other words, the dimension of the problem has been reduced from p + 1 to M + 1.

Dimension reduction can be thought of a special case of the original linear regression model, with a constraint to the estimated coefficients $\square_j$ that now must take the form

$$\beta_j = \sum_{m=1}^{M} \theta_m \phi_{jm}$$

This constraint has the potential to bias the coefficient estimates. However, in situations where p is large relative to n, selecting a value of M << p can significantly reduce the variance of the fitted coefficients.

All dimension reduction methods work in two steps. First, the transformed predictors are obtained. Second, the model is fit using these M predictors. However, the choice of Z_1, Z_2, … Z_m, or equivalently, the selection of the ϕ_{jm}'s can be achieved in different ways. In this chapter, we will consider two approaches for this task: *principal components* and *partial least squares*.

Principal Components Regression

PCA is a technique for reducing the dimension of a n x p data matrix X.
The first principal component direction of the data is the one along which the observations vary the most.
If we *projected* the obs along this line(direction), we would have the largest possible variance.
Projecting a point onto a line simply involves finding the location on the line which is closest to the point.

How can it be defined mathematically? Example: 2 predictors X_1 and X_2.
The first principal component is $Z_1 = \phi_{11} \times (X_1 - \overline{X}_1) + \phi_{21} \times (X_2 - \overline{X}_2)$ with ϕ_{11} and ϕ_{21} such that they maximize the variance of Z_1 and subjected to $\phi^2_{11} + \phi^2_{21} = 1$ since otherwise we could increase ϕ_{11} and ϕ_{21} arbitrarily in order to blow up the variance.

There is another interpretation for PCA: the first principal component vector defines the line that is as close as possible to the data.
The first principal component appears to capture most of the information contained in the predictors.

In geral, one can construct up to p distinct principal components. The second principal component Z_2 is a linear combination of the variables that is **uncorrelated** with Z_1, and has the largest variance subject to this constraint. It turns out that the zero correlation condition of Z_1 with Z_2 is equivalent to the condition that the direction must be *perpendicular* or *orthogonal*, to the first principal component direction.
And so on for successive principal components.

The Principal Components Regression Approach

The *principal components regression* (PCR) approach involves constructing the first M principal components, Z_1, …, Z_M, and then using these components as the predictors in a linear regression model that is fit using least squares.

The key idea is that often a small number of principal components suffice to explain most of the variability in the data, as well as the relationship with the response.

In other words, we assume that the directions in which $X_1,...,X_p$ show the most variation are the directions that are associated with Y.

While this assumption is not guaranteed to be true, it turns out to be a reasonable enough approximation to give good results.

If the assumption underlying PCR holds, then fitting a least squares model to Z_1, …, Z_M will lead to better results than fitting a least squares model to $X_1,...,X_p$, since most or all of the info in the data that relates to the response is contained in $Z_1,...,Z_M$, and by estimating only $M << p$ coefficients we can mitigate overfitting.

PCR will not do well if many principal components are required in order to adequately model the response.
In contrast, PCR will tend to do well in cases when the first few principal components are sufficient to capture most of the variation in the predictors as well as the relationship with the response.

It has to be noted that, even though PCR provides a simple way to perform regression using $M < p$ predictors, it is **not a feature selection method**. This is because each of the M principal components used in the regression is a linear combination of **all p** of the original features. Therefore, while PCR often performs quite well in many practical settings, it does not result in the development of a model that relies upon a small set of the original features. In this sense, PCR is more closely related to ridge regression that to the lasso.

In PCR, the number of principal components, M, is typically chosen by cross-validation.
It is *generally* recommended to standardize each predictor prior to generating the principal components. This standardization ensures that all variables are on the same scale.
In the absence of standardization, the high-variance variables will tend to play a larger role in the principal components obtained, and the scale on which the variables are measured will ultimately have an effect on the final PCR model.
However, if the variables are all measured in the same units, then one might choose not to standardize them.

Partial Least Squares

The PCR approach that we just described involves identifying linear combinations, or directions, that best represent the predictors $X_1,...,X_p$. These directions are identified in an *unsupervised* way, since the response Y is not used to help determine the principal component directions. That is, the response does not *supervise* the identification of the principal components.
Consequently, PCR suffers from a drawback: there is no guarantee that the directions that best explain the predictors will also be the best directions to use for predicting the response.

We now present *partial least squares* (PLS), a *supervised* alternative to PCR.
Unlike PCR, PLS identifies the new features (linear combinations of the original features) in a supervised way -- that is, it makes use of the response Y in order to identify new features that not only approximate the old features well, but also that are related to the response.

Roughly speaking: PLS attempts to find directions that help explain both the response and the predictors.

First PLS direction is computed like this:
After standardizing the p predictors, PLS computes the Z_1 by setting each ϕ_{jm} in $Z_m = \sum_{j=1}^{p} \phi_{jm} X_j$ equal to the coefficient from the simple linear regression of Y onto X_j.
One can show that this coefficient is proportional to the correlation between Y and X_j,
Hence, in computing $Z_1 = \sum_{j=1}^{p} \phi_{j1} X_j$, PLS places the highest weight on the variables that are most strongly related to the response.

To identify the second direction we first *adjust* each of the variables for Z_1, by regressing each variable on Z_1 and taking *residuals*. These residuals can be interpreted as the remaining information that has not been explained by the first PLS direction. We then compute Z_2 using this *orthogonalized* data in exactly the same fashion as Z_1 was computed based on the original data.

This iterative approach can be repeated M times to identify multiple PLS components $Z_1,...,Z_M$. Finally, at the end of this procedure, we use least squares to fit a linear model to predict Y using $Z_1,...Z_M$ in exactly the same fashion as for PCR.

As with PCR the number M of partial least squares directions used in PLS is a tuning parameter that is typically chosen by cross-validation.
We generally standardize the predictors and response before performing PLS.

In practice, it often performs no better than ridge regression or PCR. While the supervised dimension reduction of PLS can reduce bias, it also has the potential to increase variance, so that the overall benefit of PLS relative to PCR is a wash.

Considerations in High Dimensions

Most traditional statistical techniques for regression and classification are intended for the low-dimensional setting in which n, the number of observations, is much greater than p, the number of features.
Nowadays, it is commonplace to collect an almost unlimited number of features measurements (p very large). While p can be extremely large, the number of observations n is often limited due to cost, sample availability, or other considerations.

Data sets containing more features than observations are often referred to as high-dimensional. Classical approaches such as least squares linear regression are not appropriate in this setting. Many of the issues that arise in the analysis of high-dimensional data were discussed earlier, since they apply also when n > p: these include the role of the bias-variance trade-off and the danger of overfitting. Though these issues are always relevant, they can become particularly important when the number of features is very large relative to the number of obs.
The considerations that we will now discuss also apply if p is slightly smaller than n, and are best always kept in mind when performing supervised learning.

What Goes Wrong in High Dimensions?

Imagine using linear least squares regression. Example: p = 2, n = 2. The regression line will fit the data exactly. This is problematic because this perfect fit will almost certainly lead to overfitting of the data. The problem is simple: when p > n or p ~ n, a simple least squares regression line is too flexible and hence overfits the data.

With a fixed number of observations n, if we increase the number of features p, the model R^2 increases and correspondingly the training set MSE decreases to 0 as the number of features increases, *even though the features are completely unrelated to the response*.
On the other hand, the MSE on an *independent test set* becomes extremely large as the number of features included in the model increases, because including the additional predictor leads to a vast increase in the variance of the coefficient estimates.

C_p, AIC, and BIC and are not appropriate in the high-dimensional setting, because estimating $\hat{\sigma}^2$ is problematic. Similarly, problems arise in the application of adjusted R^2 in the high-dimensional setting, since one can easily obtain a model with an adjusted R^2 value of 1. Clearly, alternative approaches that are better suited to the high-dimensional settings are required.

Regression in High Dimensions

It turns out that many of the methods seen in this chapter for fitting less flexible least squares models, such as forward stepwise selection, ridge regression, the lasso, and principal components regression, are particularly useful for performing regression in the high-dimensional setting. Essentially, these approaches avoid overfitting by using a less flexible fitting approach than least squares.

Degrees of freedom of lasso fit → simply the number of non-zero coefficient estimates in the lasso solution, and is a measure of the flexibility of the lasso fit.

1. regularization or shrinkage plays a key role in high-dimensional problems
2. appropriate tuning parameter selection is crucial for good predictive performance,
3. the test error tends to increase as the dimensionality of the problem increases, unless the additional features are truly associated with the response

The third point above is in fact a key principle in the analysis of high-dimensional data, which is known as the curse of dimensionality.

In general, adding *signal features* that are truly associated with the response will improve the fitted model, in the sense of leading to a reduction in test set error.

However, adding noise features that are not truly associated with the response will lead to a deterioration in the fitted model, and consequently an increased test set error. This is because noise features increase the dimensionality of the problem, exacerbating the risk of overfitting(since noise features may be assigned nonzero coefficients due to chance associations with the response on the training set) without any potential upside in terms of improved test set error.

Interpreting Results in High Dimensions

When we perform the lasso, ridge regression, or other regression procedures in the high-dimensional setting, we must be quite cautious in the way that we report the results obtained.
In the high-dimensional setting, the multicollinearity problem is extreme: any variable in the model can be written as a linear combination of all the other variables in the model. Essentially this means that we can never know exactly which variables (if any) truly are predictive of the outcome, and we can never identify the *best* coefficients for use in the regression. At most, we can hope to assign large regression coefficients to variables that are correlated with the variables that truly are predictive of the outcome.

This does not detract from the value of the model obtained but we must be careful not to overstate the results obtained, and to make it clear that what we have identified is simply one of many possible models and that it must be further validated on independent data sets.

It is also important to be particularly careful in reporting errors and measures of model fit in the high-dimensional setting. We have seen that when p > n, it is easy to obtain a useless model that has zero residuals.

Therefore, one should *never* use sum of squared errors, p-values, R^2 statistics, or other traditional measures of model fit on the training data as evidence of a good model fit in the high-dimensional setting.

It is important to instead report results on an independent test set, or cross-validation errors.

For instance, MSE or R^2 on an independent test set is a valid measure of model fit, but the MSE on the training set certainly is not.