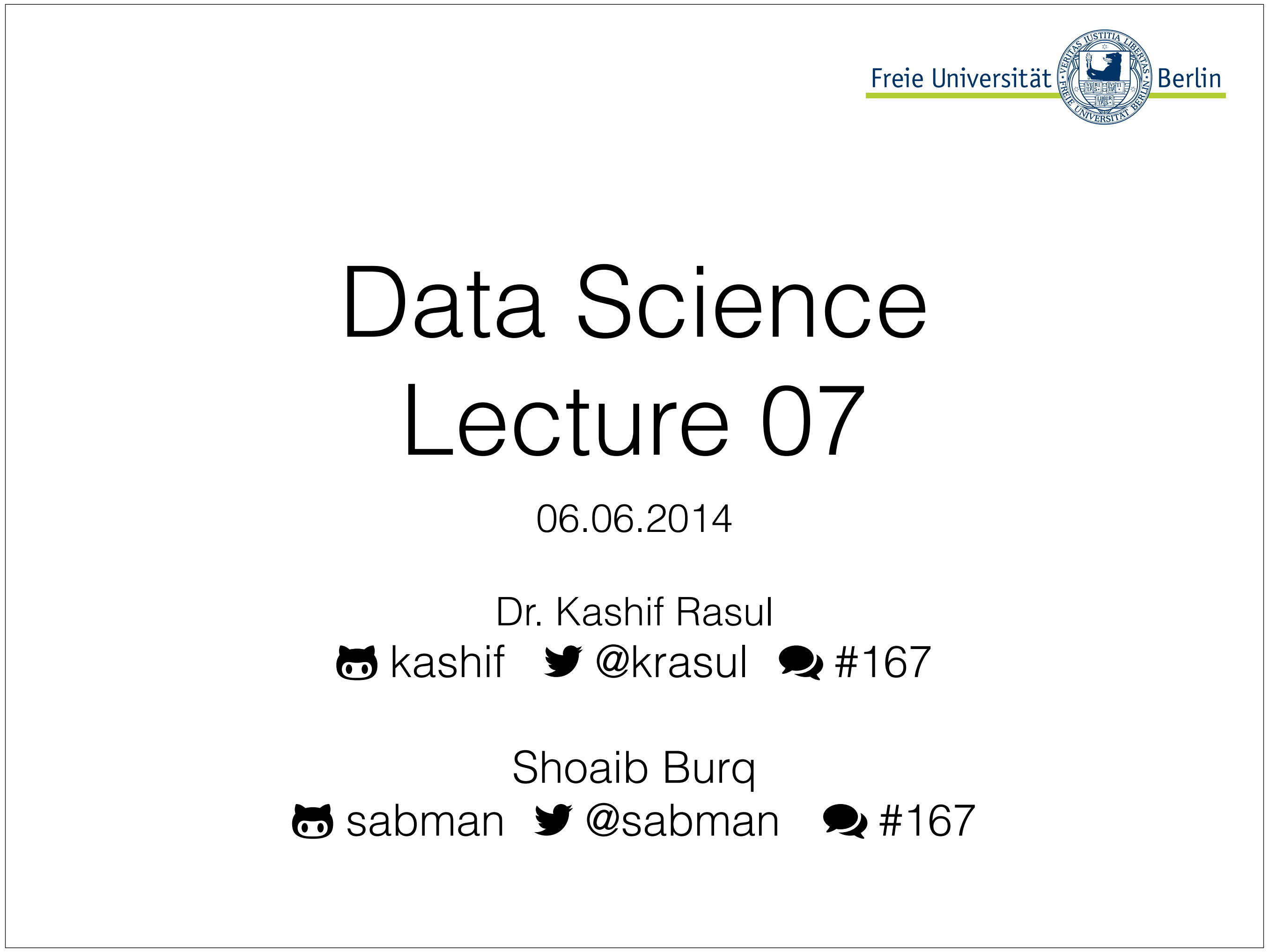


Freie Universität Berlin

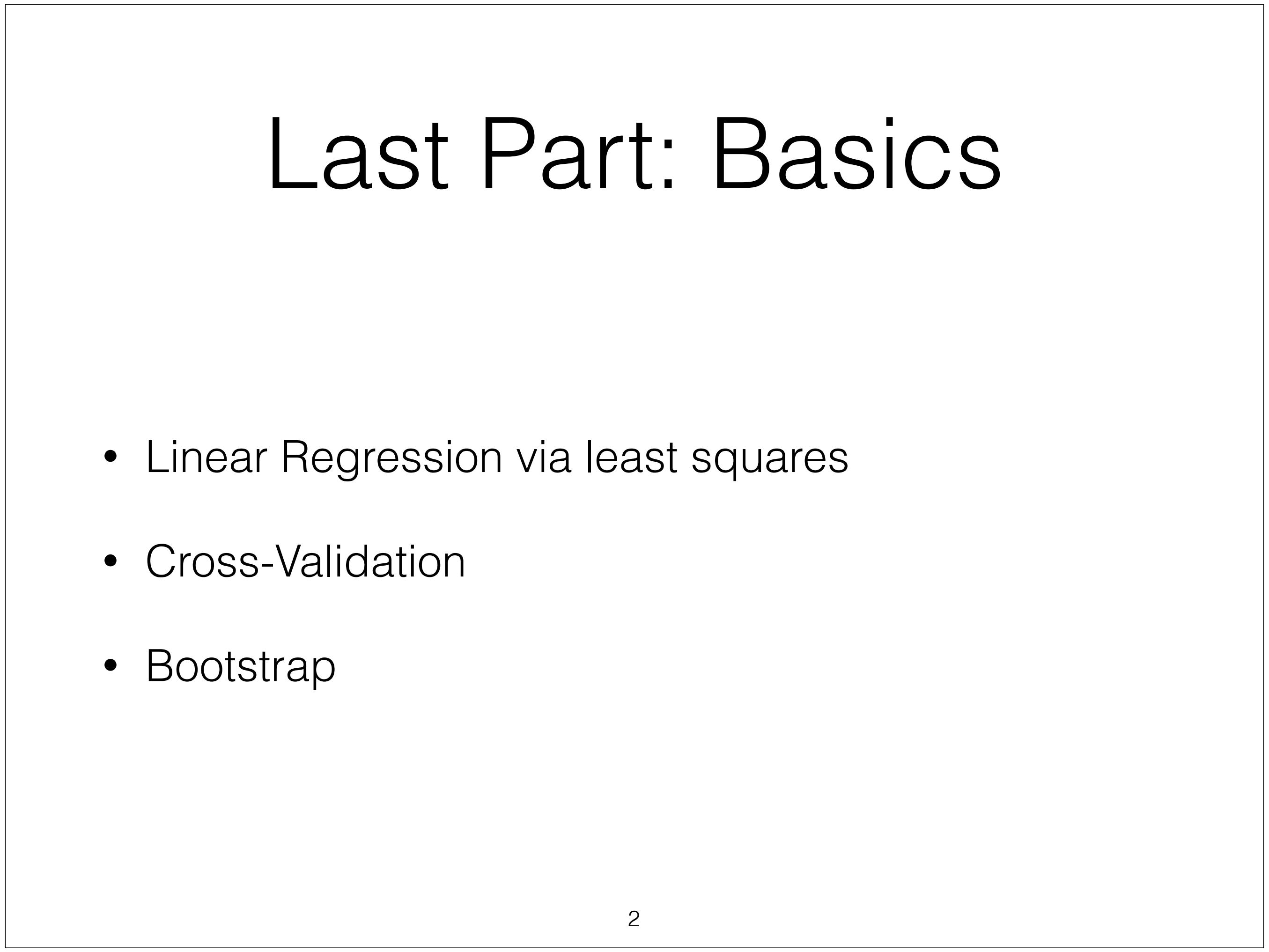
Data Science Lecture 07

06.06.2014

Dr. Kashif Rasul
kashif @krasul #167

Shoaib Burq
sabman @sabman #167

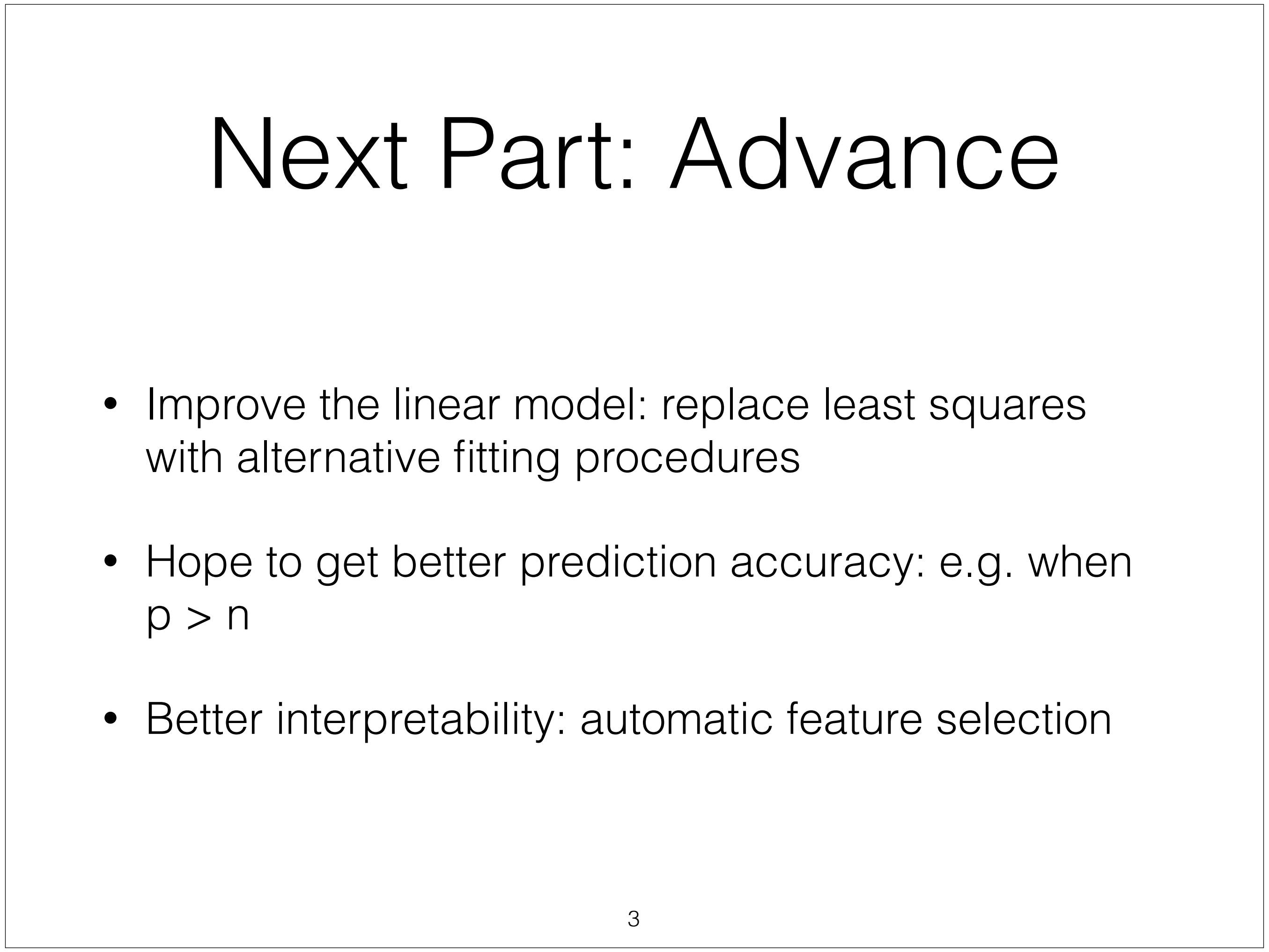
Welcome to Data Science Lecture 7.



Last Part: Basics

- Linear Regression via least squares
- Cross-Validation
- Bootstrap

We have now seen how to fit a standard linear model via least squares, as well as the basics of computing model's performance and selecting it's flexibility by estimating the test error via CV or measuring the standard deviation of any statistic of interest via the Bootstrap. This concluded the initial basics of Statistical learning and it now time to move onto more advance non-linear methods.

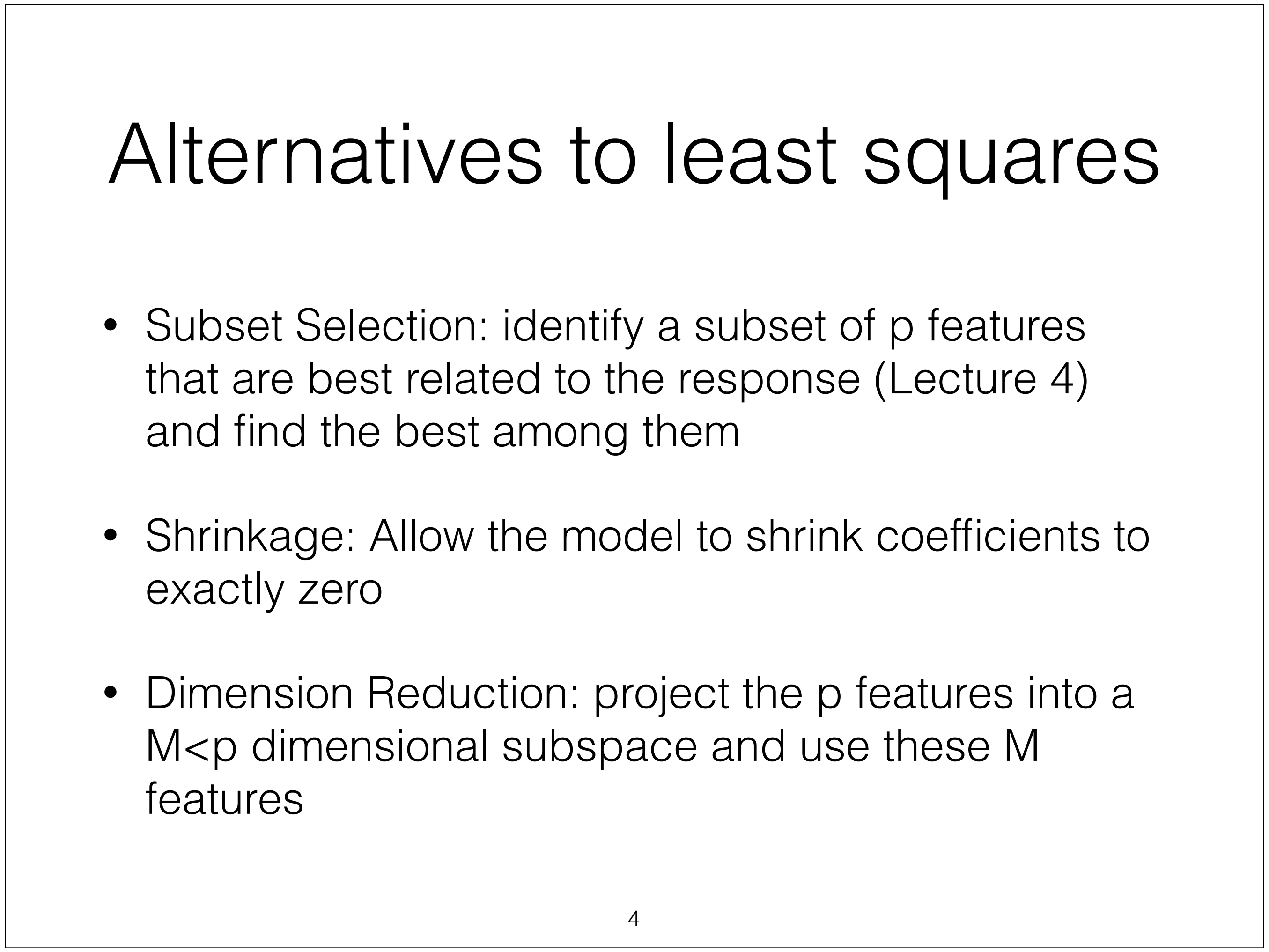


Next Part: Advance

- Improve the linear model: replace least squares with alternative fitting procedures
- Hope to get better prediction accuracy: e.g. when p > n
- Better interpretability: automatic feature selection

But before we start looking into non-linear but still additive models or even more general non-linear models we must emphasise that the linear model has distinct advantages: inference and the fact that on real-world problems it is a surprisingly competitive method compared to other complicated ones. So before we go forward we discuss ways in which the simple linear model can be improved. So what do we want to achieve?

We hope to gain better prediction accuracy especially for situations where $p>n$ where least squares is no longer unique. And we hope for better model interpretability by automatically removing irrelevant variables resulting in a less complex model. Least squares will never automatically set coefficients to zero so we will need a new approach.

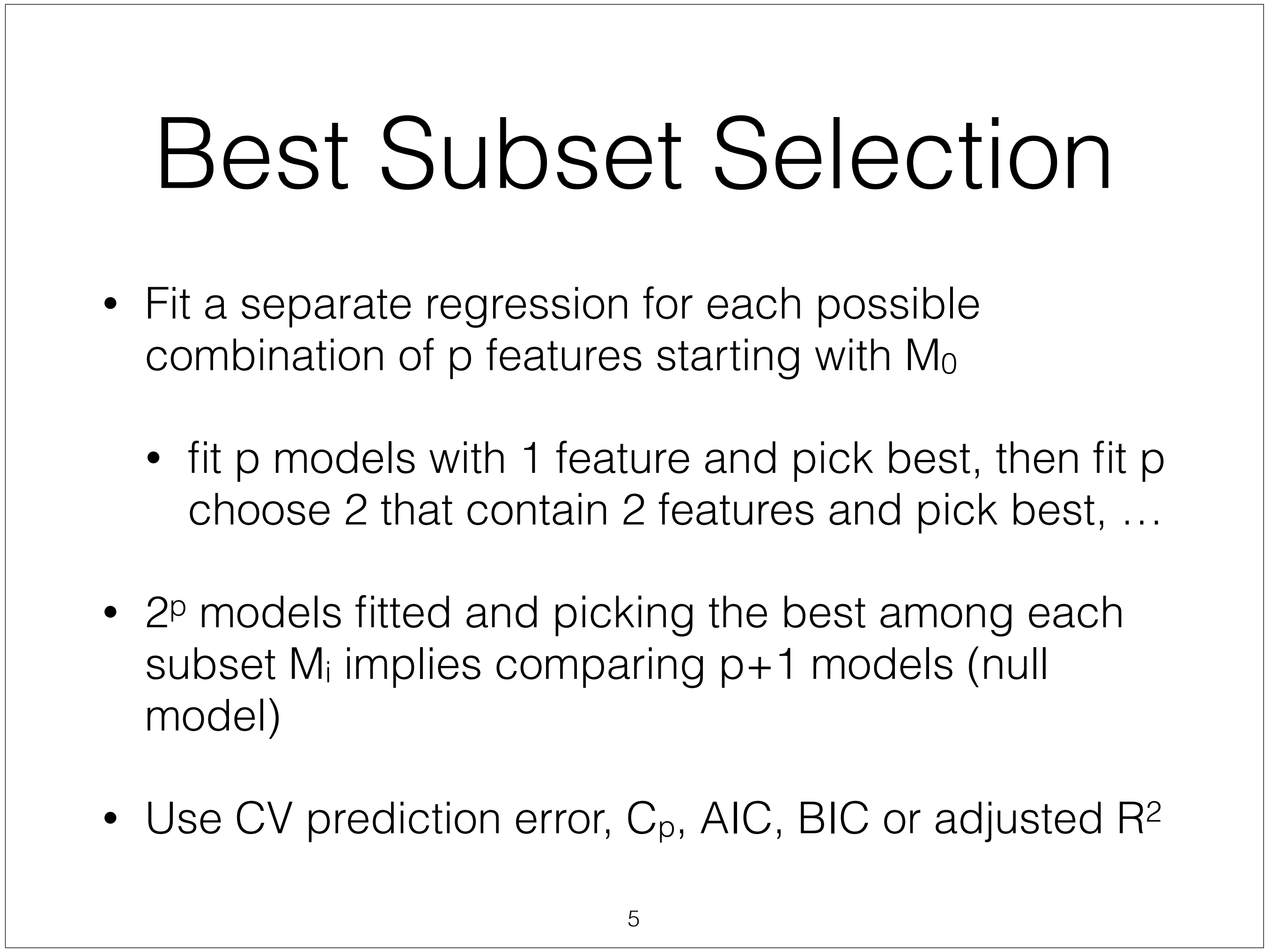


Alternatives to least squares

- Subset Selection: identify a subset of p features that are best related to the response (Lecture 4) and find the best among them
- Shrinkage: Allow the model to shrink coefficients to exactly zero
- Dimension Reduction: project the p features into a M<p dimensional subspace and use these M features

The three most important alternatives that we will cover today are above:

Recall we talked about Subset selection in Lecture 4 when addressing some important questions about Linear regression. In particular do all the predictors help explain the response? We showed how to get the most important 1, 2, 3, etc. predictors but did not discuss how to choose the best among them. But let us recap first…

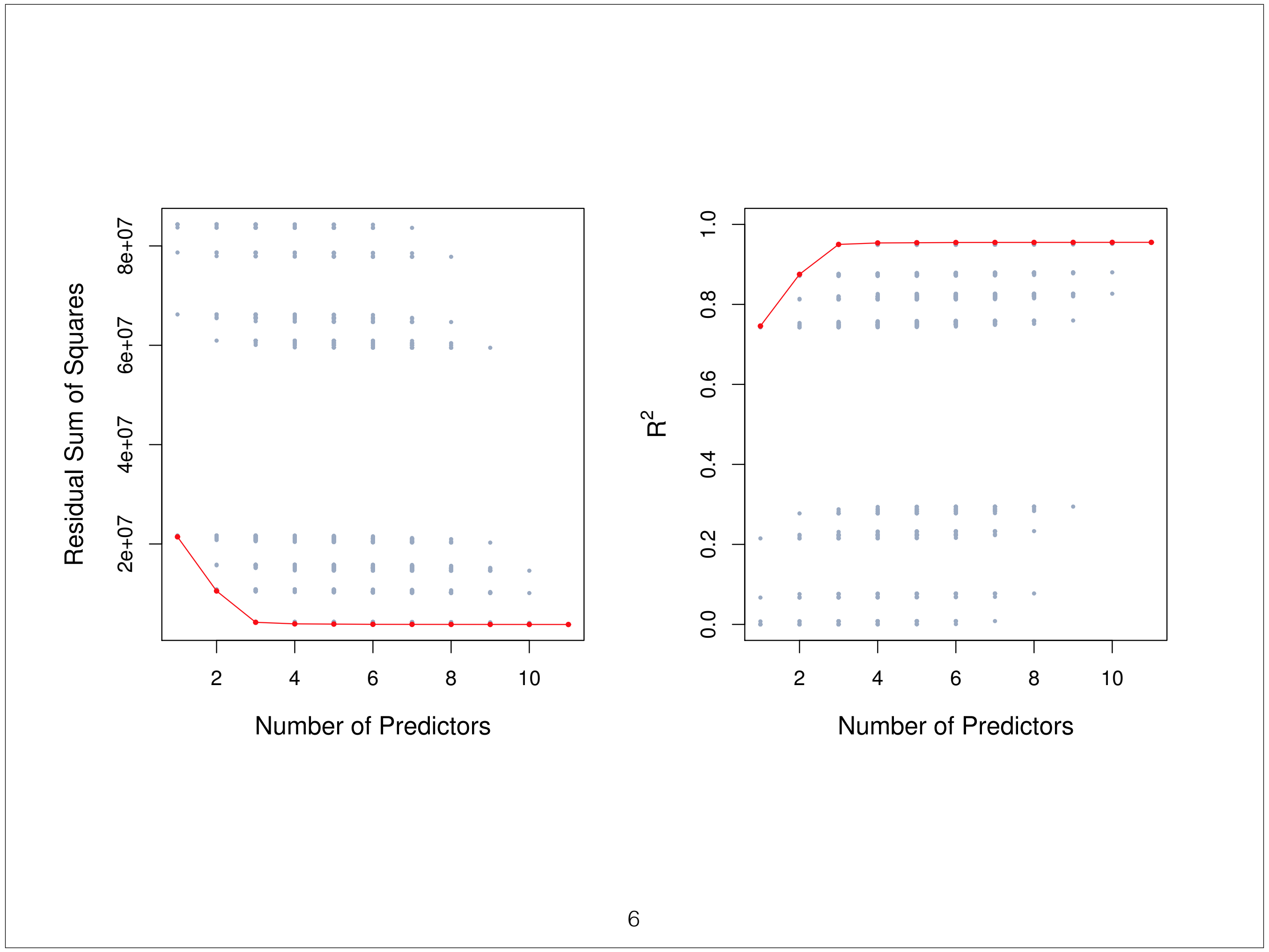


Best Subset Selection

- Fit a separate regression for each possible combination of p features starting with M_0
  - fit p models with 1 feature and pick best, then fit p choose 2 that contain 2 features and pick best, …
- 2^p models fitted and picking the best among each subset M_i implies comparing p+1 models (null model)
- Use CV prediction error, C_p, AIC, BIC or adjusted R^2

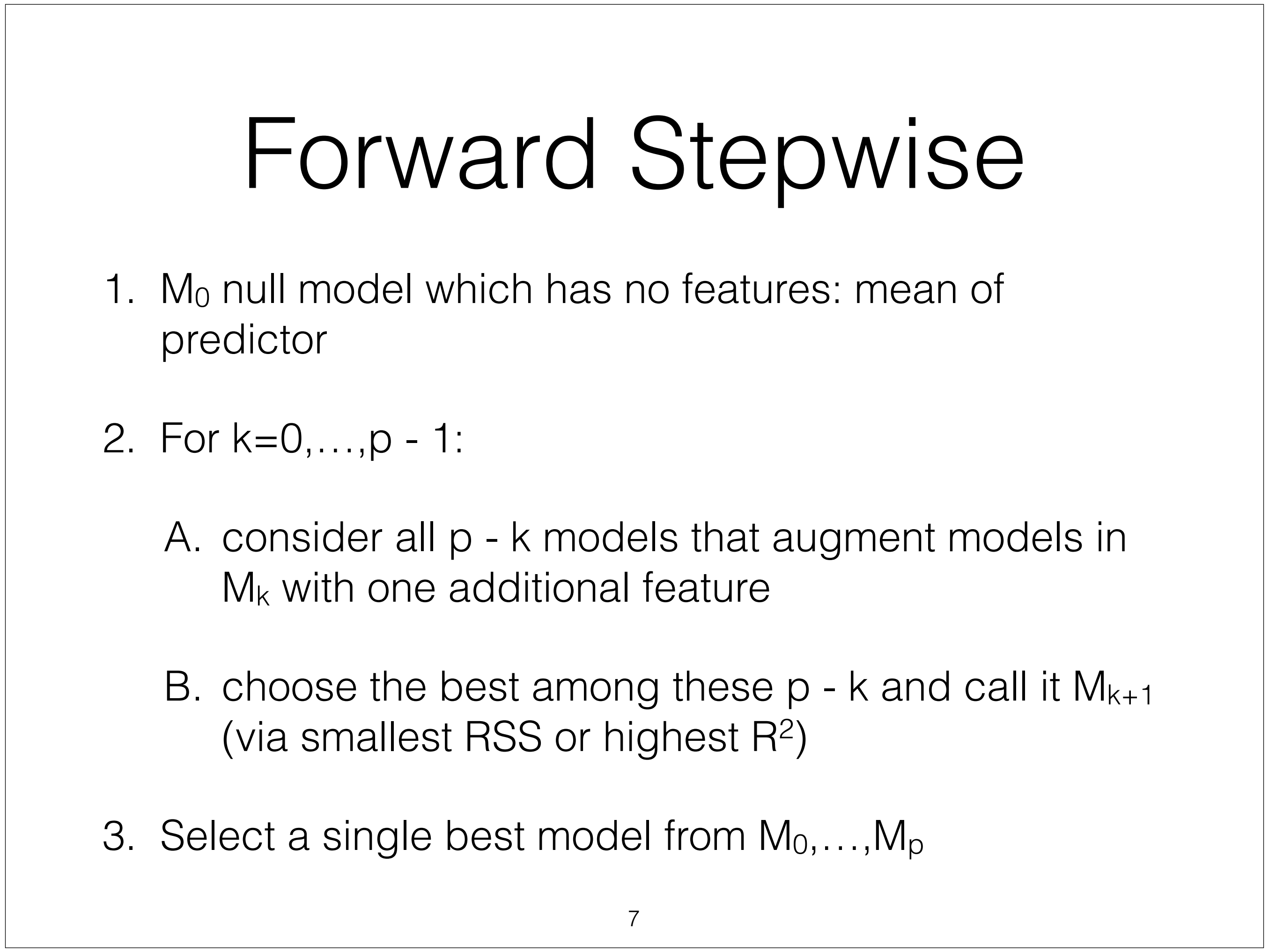
Selecting the best among these p+1 possible models must be done with care since the RSS of these p+1 decreases monotonically and the R^2 increases as the number of features in the model. The problem is that a low RSS or high R^2 indicates a model with a low training error, whereas we wish to choose a model that has low test error. For classification problems the deviance: negative two times the maximised log-likelihood plays the role of the RSS. The smaller the deviance the better the fit. And we will discuss soon how to choose the best model among the various subsets.

The big disadvantage here is the 2^p, for p=20 there are 1 million possibilities and for p > 40 it is not really efficient…



So doing this for the Credit dataset starting with 1 predictor all the way to subsets with 11 predictors, the RSS and R^2 are plotted for each subset and the minimum RSS and the maximum R^2 are tracked. Note the data has 10 predictors but one variable is categorical taking 3 values and is represent by 2 dummy variables.

Again note that it is not surprising that the RSS will decrease monotonically nor that the R^2 increases as the number of predictors increases. The hard bit is trying to figure out which of these 11 models is the best?



Forward Stepwise

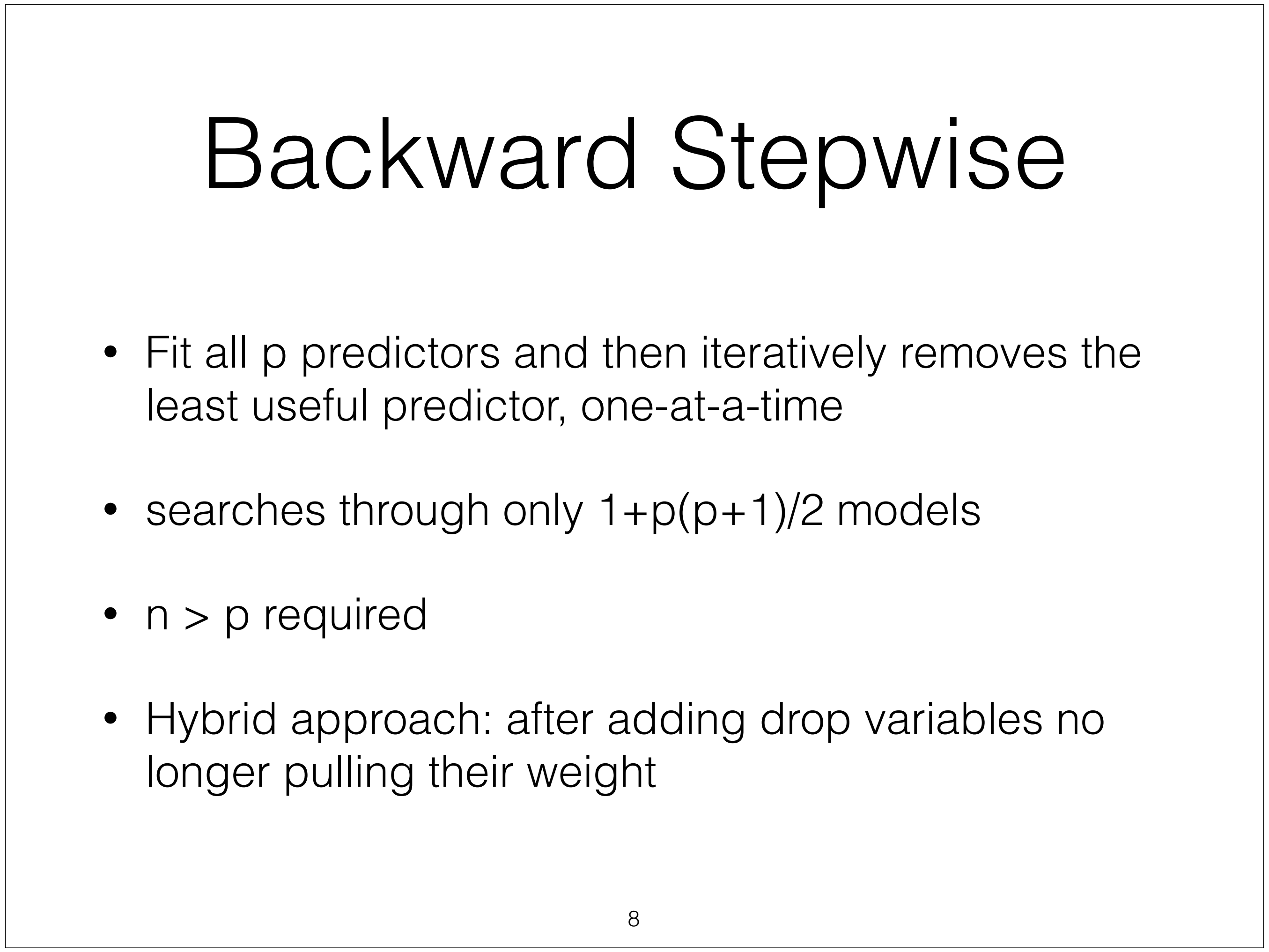
1. M_0 null model which has no features: mean of predictor
2. For k=0,…,p - 1:
    A. consider all p - k models that augment models in M_k with one additional feature
    B. choose the best among these p - k and call it M_{k+1} (via smallest RSS or highest R^2)
3. Select a single best model from M_0,…,M_p

Apart from the higher computational cost, another issue with best subset selection with large p is: the larger the search space, the higher the chance of finding models that look good on the training data, even though they might not have any predictive power on future data. An enormous search space leads to overfitting and high variance of the coefficient estimates. So Forward Stepwise Selection is more attractive since it searches a far more restricted set of models.

Forward stepwise selection begins with a model containing no predictors just the mean of each feature, and then it 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. And unlike best subset selection, which involved fitting 2^p models, here we fit 1+p(p+1)/2 models. E.g. for p = 20 instead of a million it is 211 models.

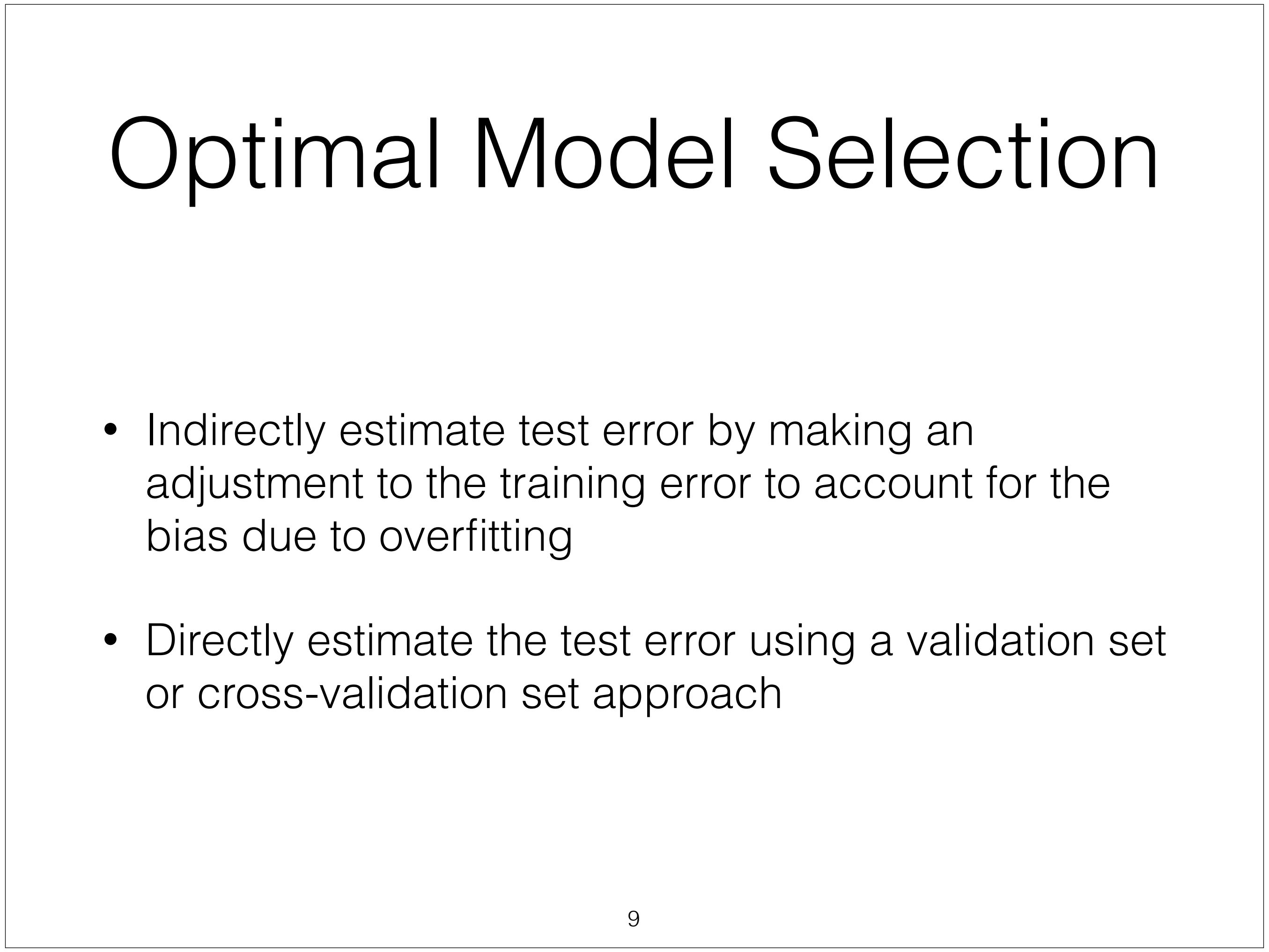
Recall from Lecture 4, one disadvantage here is that forward selection will retain feature as it progresses and so if a feature in the later steps turns out to be redundant, forward selection will still retain it. On the other hand this can be applied when n<p but it is possible to construct sub models M_0,…M_{n-1} only since each sub model is fit using least squares and will not be unique for p>=n.



Backward Stepwise

- Fit all p predictors and then iteratively removes the least useful predictor, one-at-a-time
- searches through only 1+p(p+1)/2 models
- n > p required
- Hybrid approach: after adding drop variables no longer pulling their weight

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.

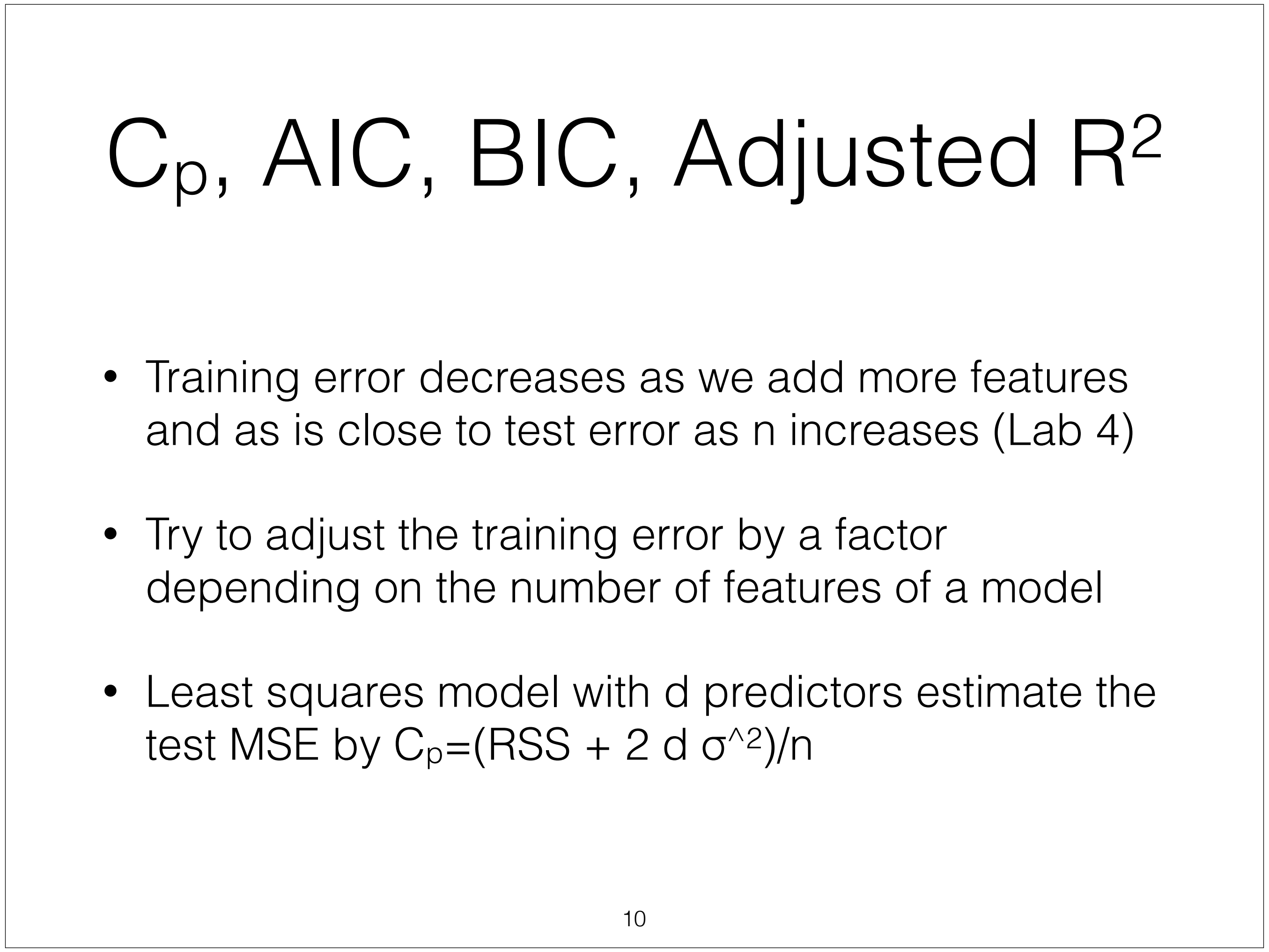


Optimal Model Selection

- Indirectly estimate test error by making an adjustment to the training error to account for the bias due to overfitting
- Directly estimate the test error using a validation set or cross-validation set approach

So RSS and R^2 are not suitable for selecting the best model among a collection of models with different numbers of predictors, since a model with more predictors will have a lower RSS or higher R^2 since it relates to the training error. Instead we want to choose a model with a low test error. Training error can be a poor judge of test error. So in order to select the best model with respect to test error, we need to estimate this test error. There are two common approaches here:

We will look at them now.



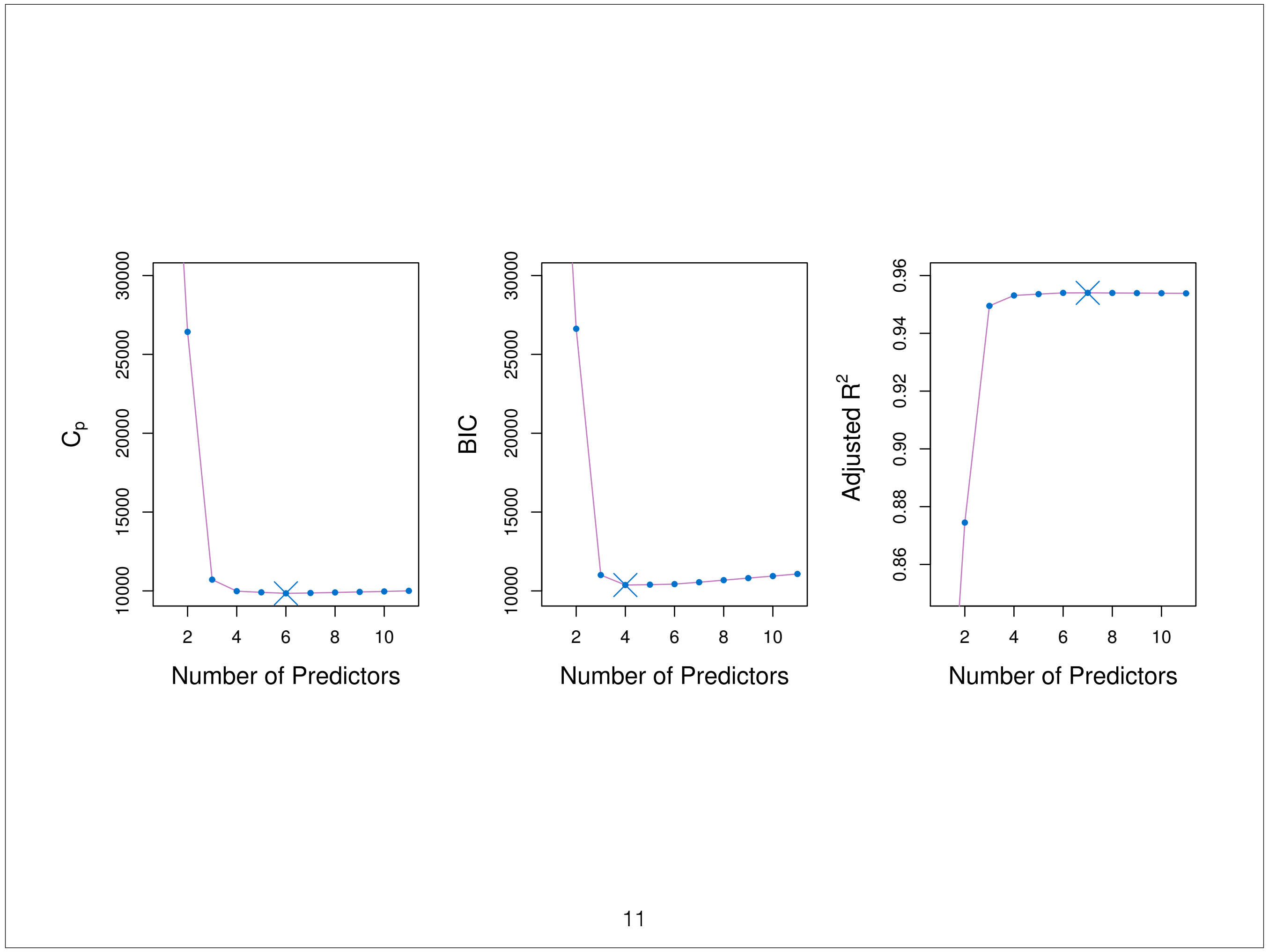
C_p, AIC, BIC, Adjusted R^2

- Training error decreases as we add more features and as is close to test error as n increases (Lab 4)
- Try to adjust the training error by a factor depending on the number of features of a model
- Least squares model with d predictors estimate the test MSE by C_p=(RSS + 2 d $\sigma^{\wedge 2}$)/n

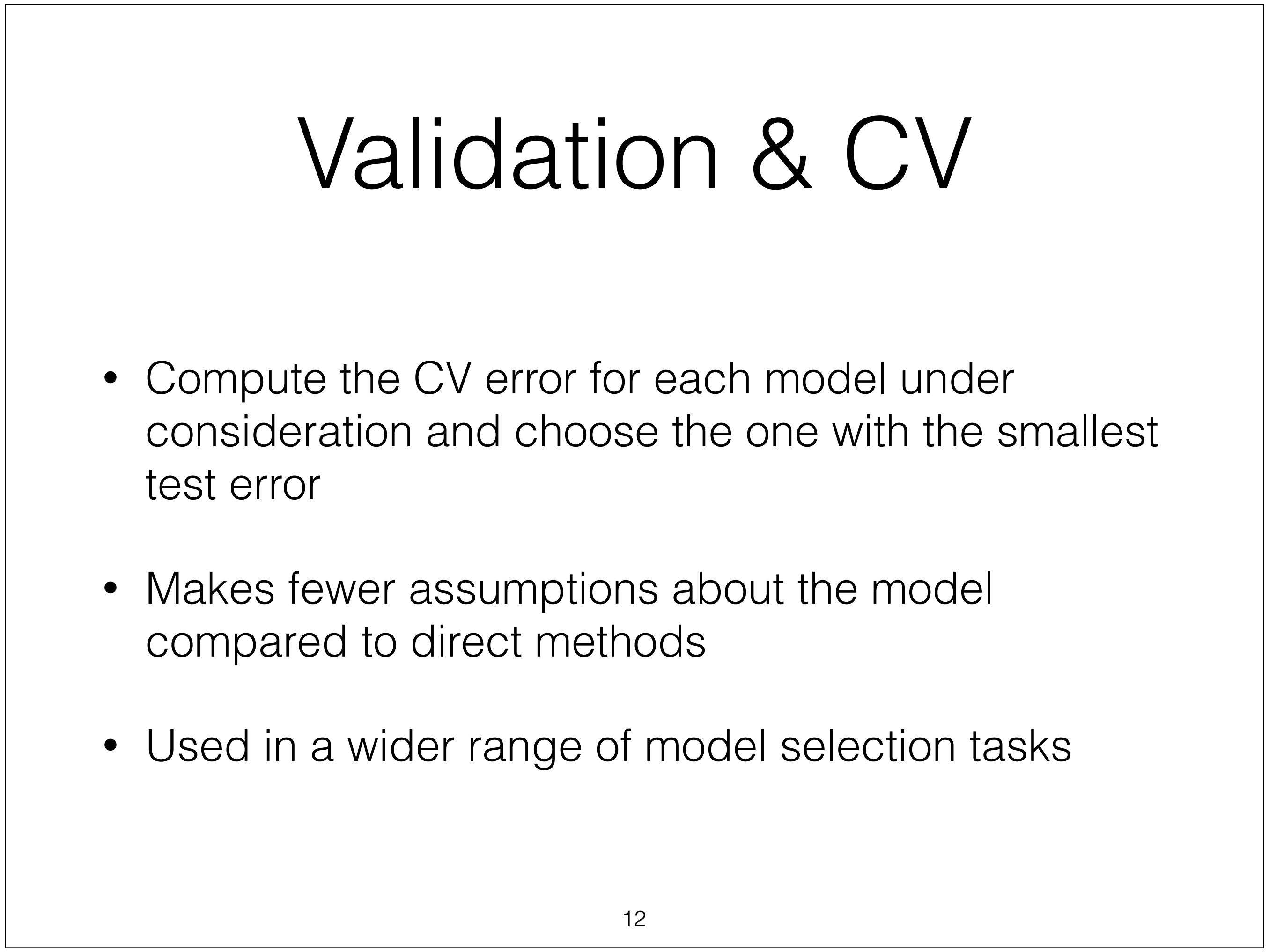
The indirect methods are motivated by the fact that the training error will decrease as we increase the number of predictors but the test error will not. So if we could adjust the training error given the number of predictors it has, we could get an estimate for the test error and from that choose the best model.

Here $\sigma^{\wedge 2}$ is an is an estimate of the variance of the error ε associated with each response measurement in our model. Essential the C_p statistic adds a penalty of 2d $\sigma^{\wedge 2}$ to the training RSS. The penalty goes up as we have more predictors in our model since with more predictors our training RSS goes down. And so C_p estimates the MSE and we choose the model with the lowest C_p value.

AIC and BIC are essentially the same for least squares. And the adjusted R^2 is defined similarly by adjusting the R^2 for a model with d variables. Here we choose the largest adjusted R2 to select the most optimal model. The intuition behind the adjusted R^2: once all the correct variables have been included adding additional noise variables will lead to only a very small decrease in RSS. So increasing d will lead to an in crease in RSS/(n-d-1) and so a decrease in the adjusted R^2. So in theory a model with the largest adjusted R^2 will have only correct variables and no noise variables. Essentially we punishing R^2 for including unnecessary variables.



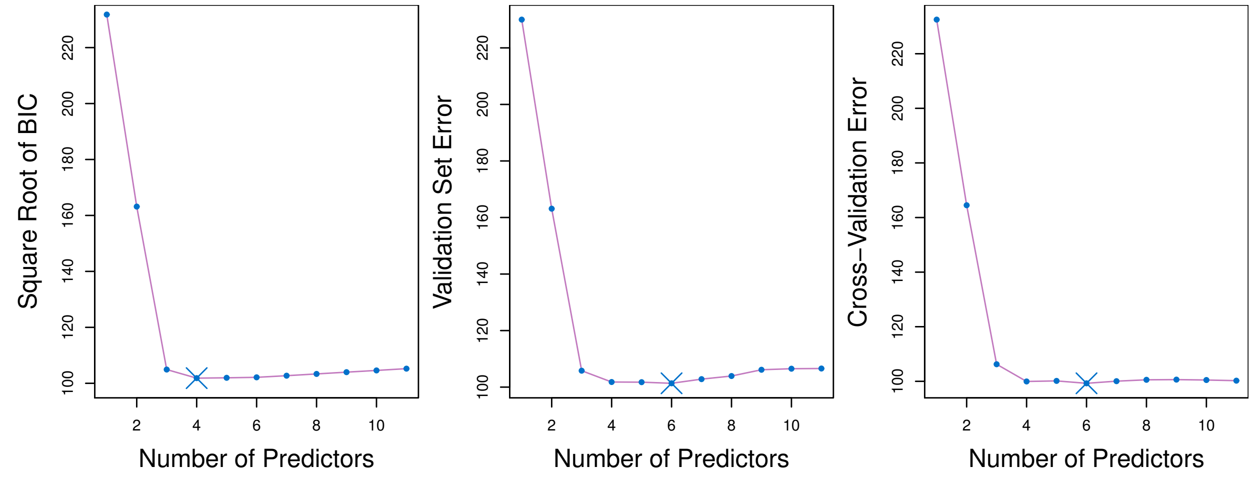
So for the Credit data here is a plot of the 3 statistics and the optimal model selected.



Validation & CV

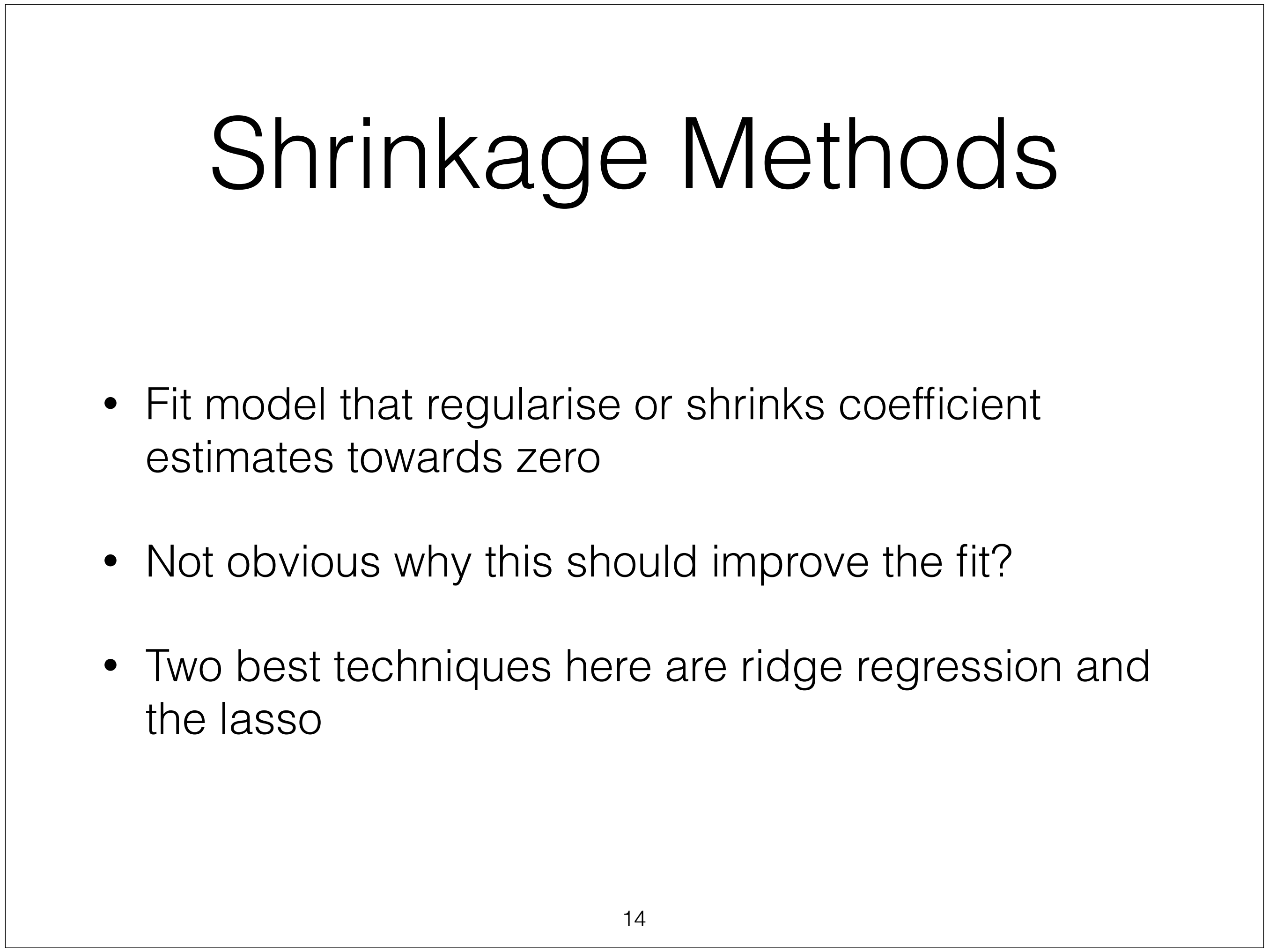
- Compute the CV error for each model under consideration and choose the one with the smallest test error
- Makes fewer assumptions about the model compared to direct methods
- Used in a wider range of model selection tasks

The alternative is to estimate the test error directly via CV. In the past, performing cross-validation was computationally prohibitive for many problems with large p and/or large n, so the indirect approach was more attractive. Nowadays with fast computers this is no longer an issue. So CV is very popular approach.



Here we see the BIC, validation set and 10-fold CV error on the Credit data for the best d-variable models. The validation errors were calculated by randomly selecting 3/4th of the observations as the training set, and the remainder as the validation set. The validation and 10-fold CV results in a 6 variable model, but we see that essentially a 4 or 5 is also the same.

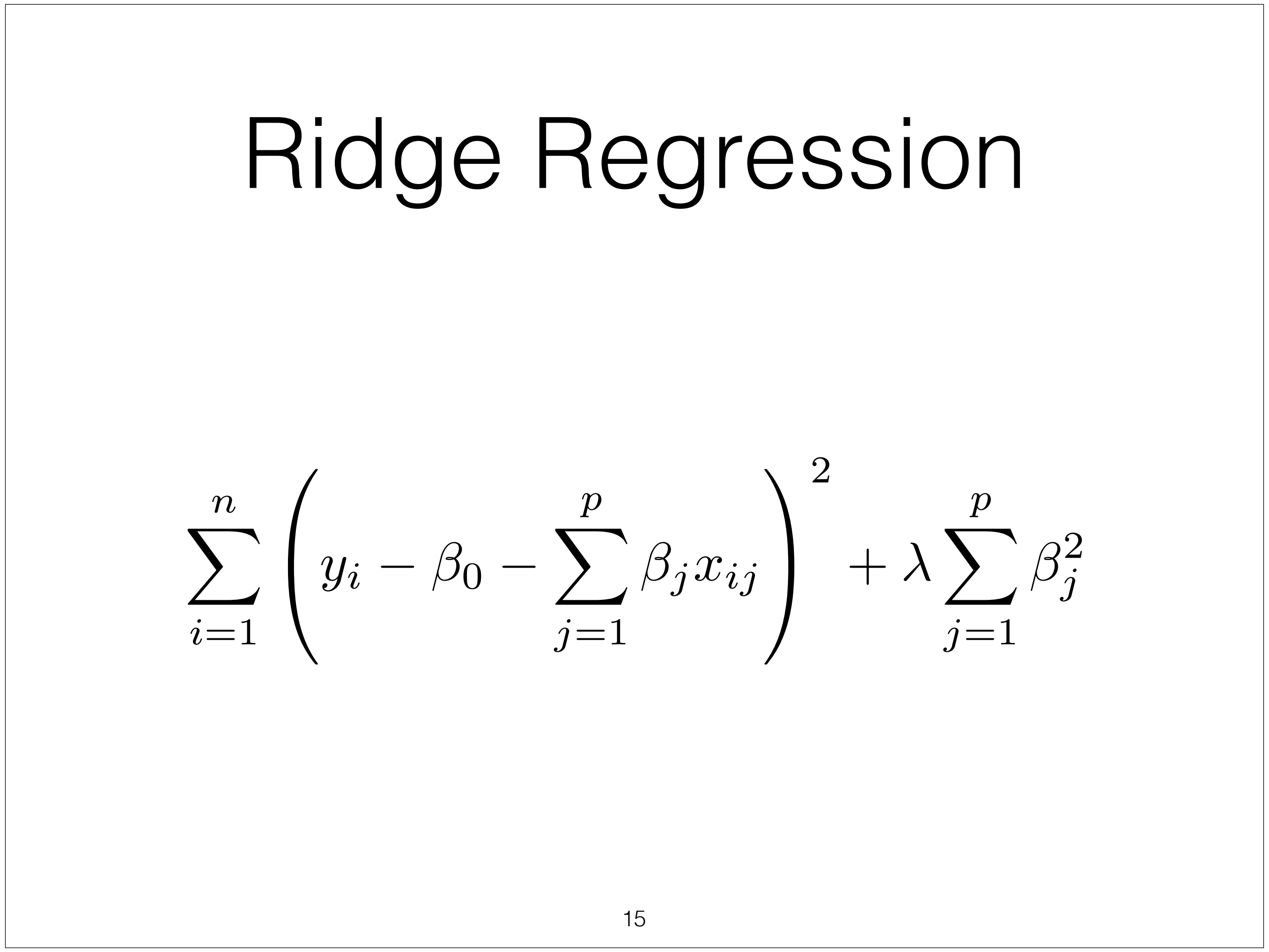
In fact if we repeated the validation set approach using a different split of the data into a training set and a validation set, or if we repeated cross-validation using a different set of cross-validation folds, then the precise model with the lowest estimated test error would surely change. In this setting, we can select a model using the one-standard-error rule: 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. So if a model appears to be more or less good, we might as well choose the simplest one.



Shrinkage Methods

- Fit model that regularise or shrinks coefficient estimates towards zero
- Not obvious why this should improve the fit?
- Two best techniques here are ridge regression and the lasso

An alternative method to least squares is to use a technique that constraints or regularises the coefficients or shrinks them towards zero. We will look at two of the best techniques and give reasons on why this should improve the fit of a model.

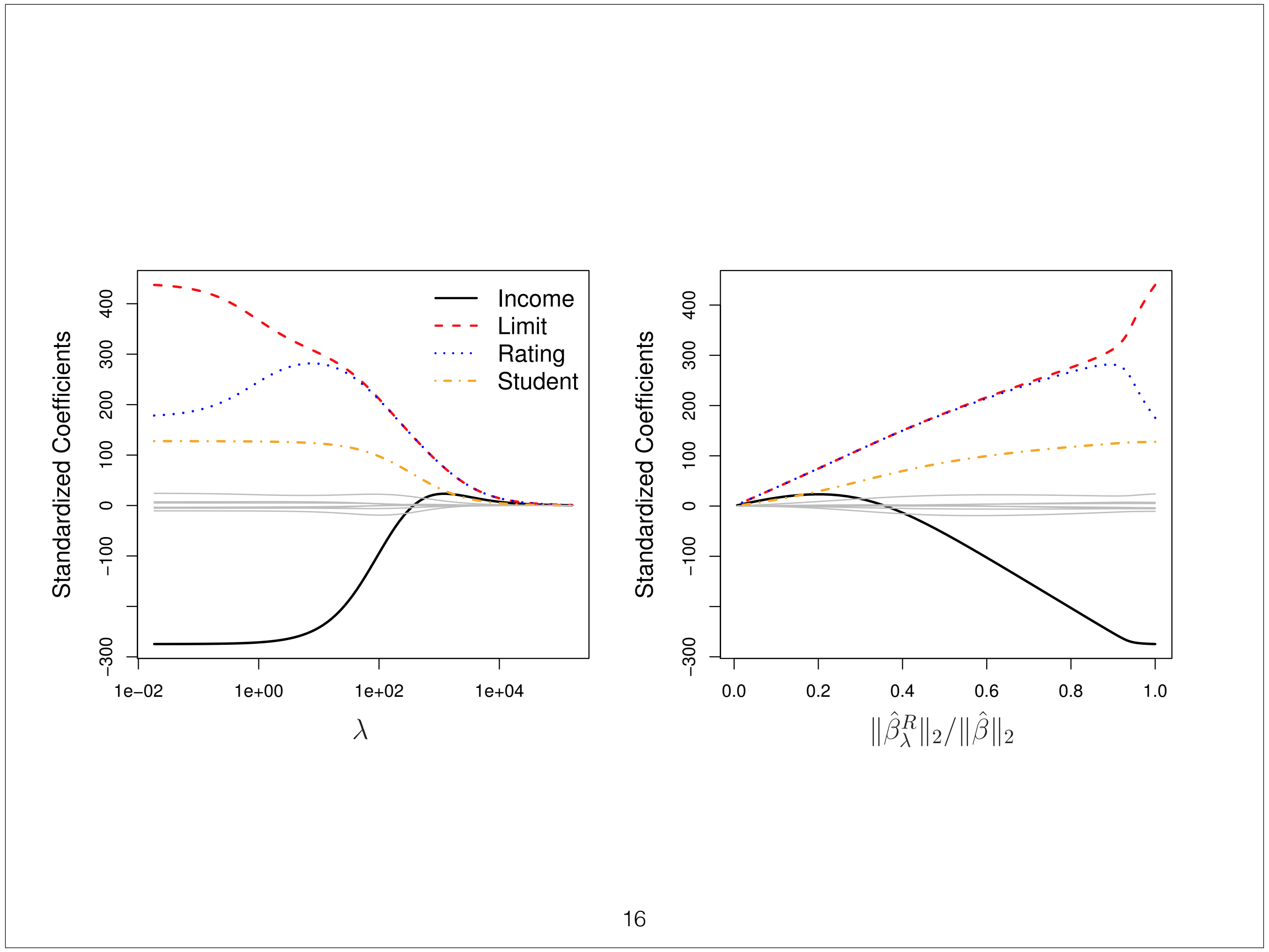


Ridge Regression

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

Recall that least square involved fitting a model that minimised the RSS. In Ridge regression we add a “shrinkage penalty” to the model via a tuning parameter λ>=0. So now λ servers to control the shrinkage penalty since there is only a finite amount that we have to divide up between the coefficients and hopefully the coefficients will tend to zero. Selecting a good λ is very important here, we will discuss that later, where CV will be used. Also note we do not apply the penalty to the intercept, since the intercept is the measure of the mean value of the response when the data is 0.

In particular for ridge regression we minimise the above equation which is the RSS + the second term: the shrinkage penalty and note it is small when the coefficients are close to zero and it has the effect of shrinking the coefficients towards zero and λ tunes the impact of these two terms. The bigger the λ the bigger the more impact the shrinkage penalty has on the model.

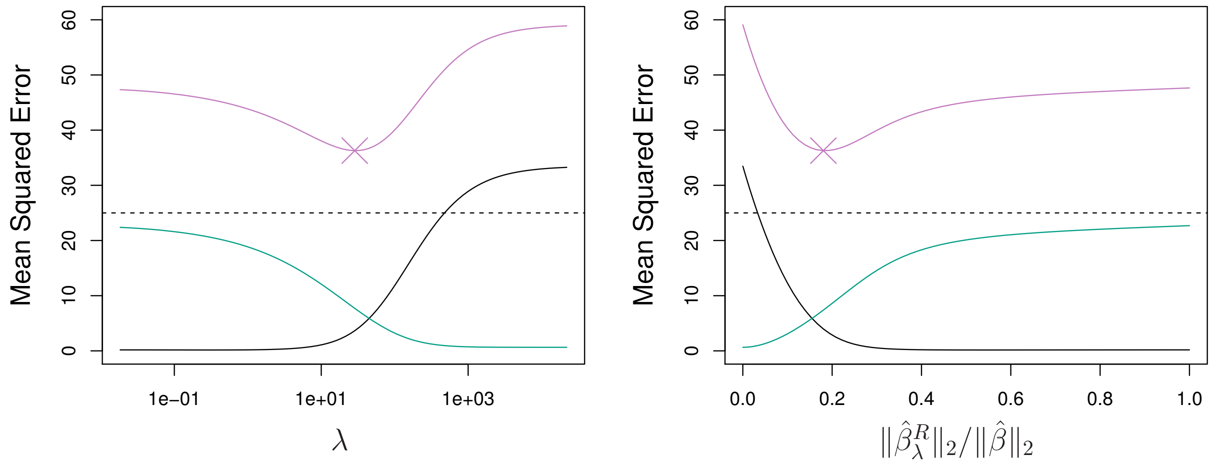


Here on the left are the ridge regression coefficients from the Credit data for different values of λ. On the left the λ is close to zero but as λ increases, the ridge coefficients estimates shrinks towards zero. The plot shows the 4 largest coefficients in different style. While together the estimates might decrease, some coefficients might increase occasionally.

On the left we show on the x-axis what we can think of as the amount the regression coefficient estimates have been shrunken towards zero; a small value indicates that they have been shrunken very close to zero. When λ=0, this ratio is 1. As λ increases the L2 norm of the ridge regression coefficients always goes down. So at λ=∞ it is zero and so is the ratio.

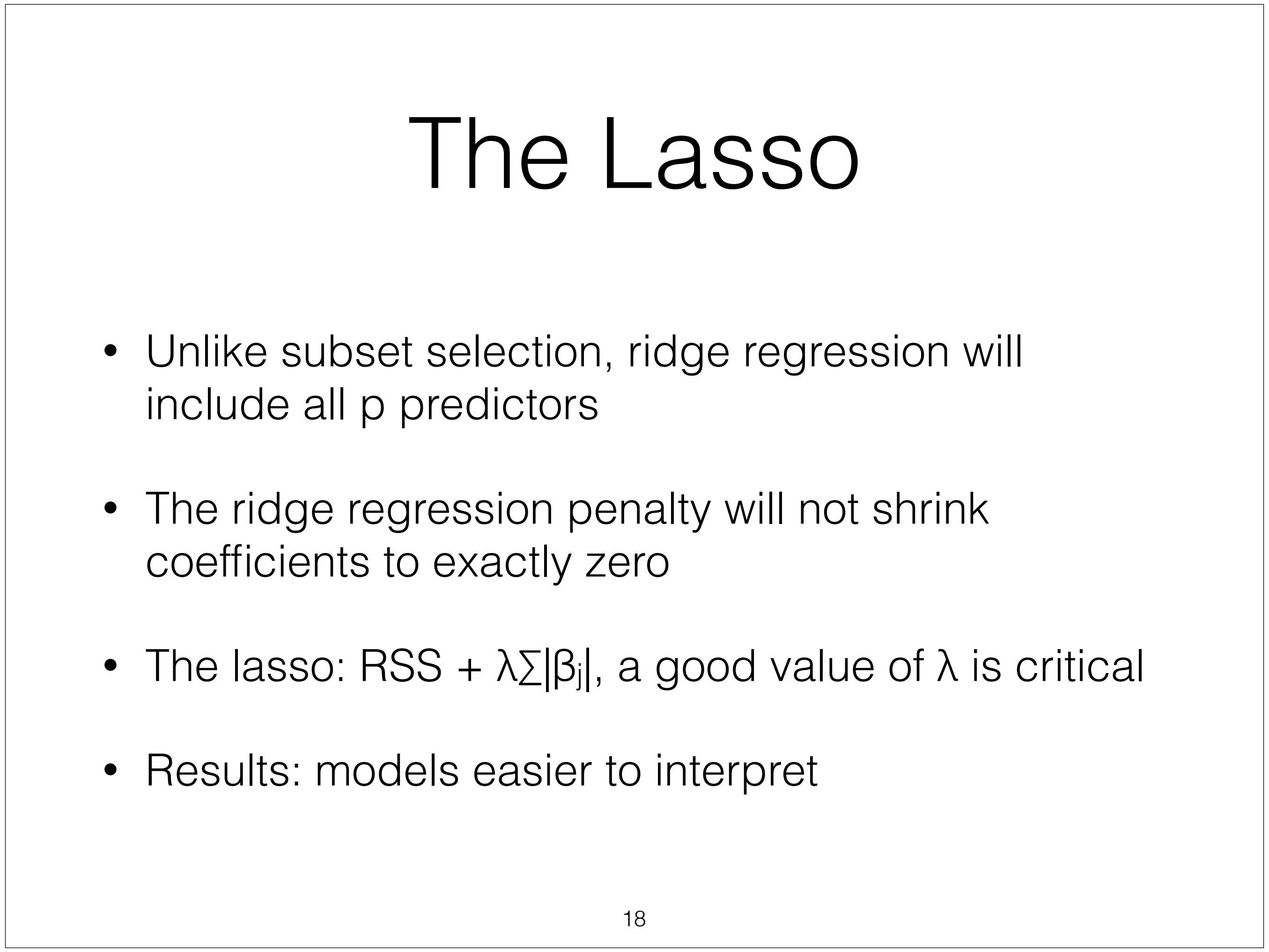
Note that least squares is scale invariant: scaling predictors X by c leads to 1/c factor of coefficients => X_j βj is unchanged. But ridge regression coefficients can change substantially. In fact the value my even depend on the scaling of other predictors! So it is best to apply ridge regression after standardising the predictors so that they are all at the same scale and in fact in this image is with standardised predictors.

Bias-Variance Trade-off



Again the Bias-Variance trade-off is at the core of why ridge regression is more advantageous than least-squares. In this image we have p=45 predictors and n=50 observations. In the left we have a plot in green of the variance of the ridge regression predictions as a function of λ. At λ=0 (least-squares) variance is high and bias is zero. As λ increases, the variance goes down at the expense of an increase in bias. The test MSE in purple is the sum of variance plus squared bias. Beyond a certain point the decrease in variance and the shrinking of the coefficients causes them to be significantly underestimated resulting in a large increase in bias. The minimum MSE is at λ=30. The left hand shows the same for the L2 norm ratios.

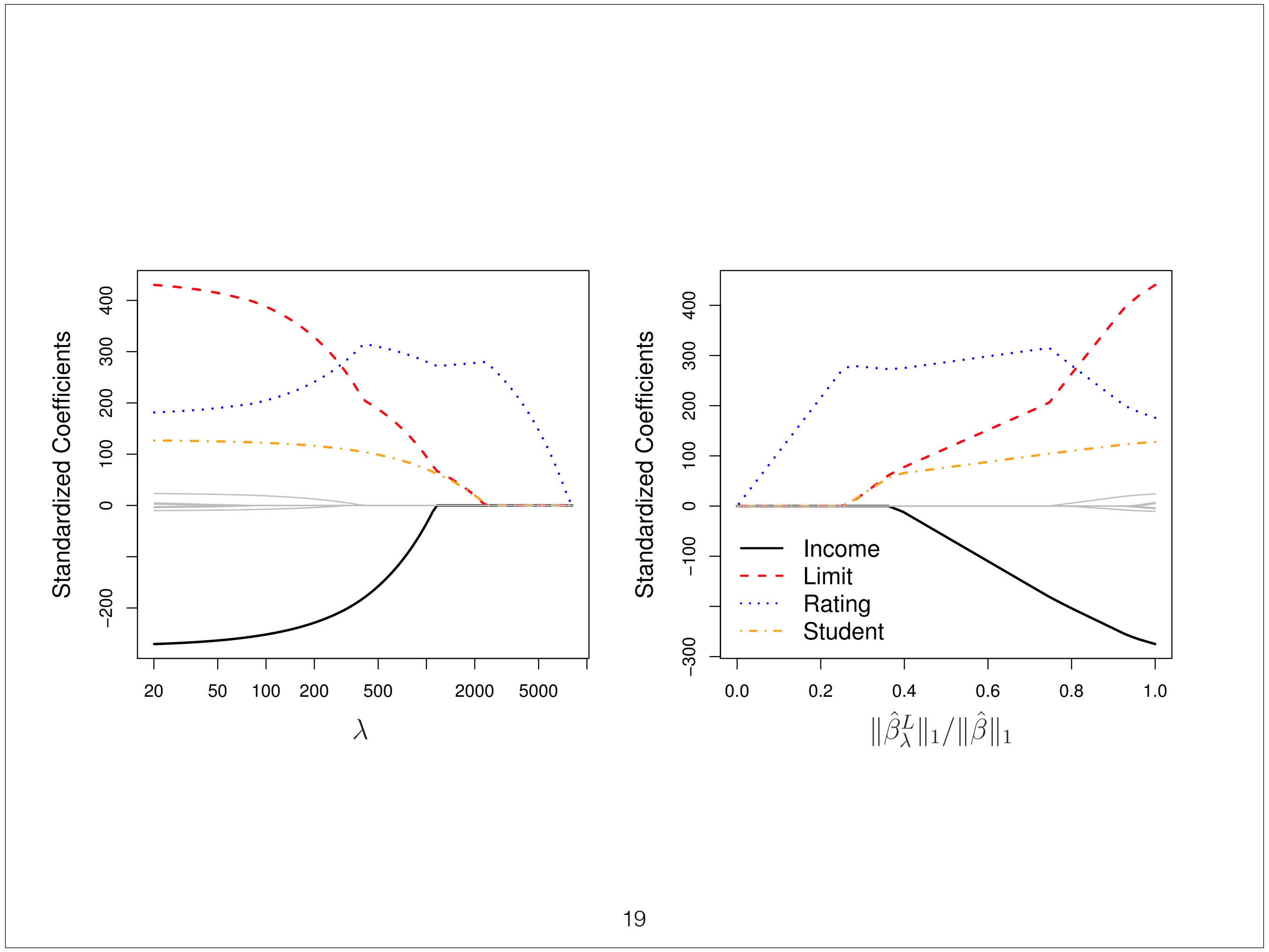
In general when the model is close to linear, the least squares will have low bias but high variance. A small change in the training data can cause a large change in the coefficients estimates. When p is almost as large as n, least squares will be very variable. If p>n, least squares has no unique solution, but ridge regression can still perform well by trading off a small increase in bias for a large decrease in variance. So ridge regression works best in situations where the least squares estimates have high variance.



The Lasso

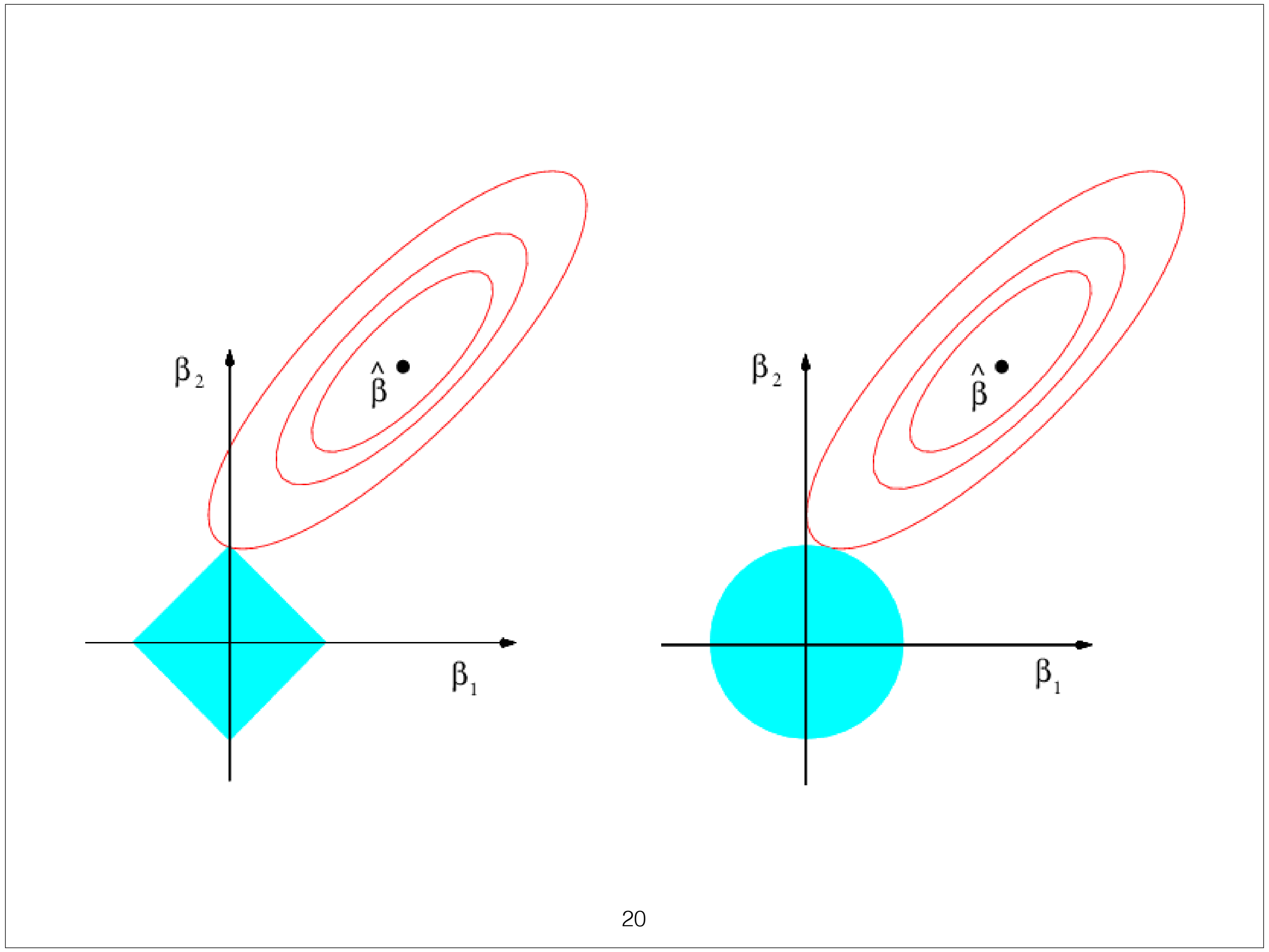
- Unlike subset selection, ridge regression will include all p predictors
- The ridge regression penalty will not shrink coefficients to exactly zero
- The lasso: RSS + $\lambda\sum|\beta_j|$, a good value of λ is critical
- Results: models easier to interpret

Ridge regression will not shrink all of the coefficients towards zero which 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 replaces the L2 norm by a L1 norm which has the effect of forcing the coefficient estimates to be exactly equal to zero. Again selecting a good value of λ for the lasso is critical; where we will use CV for this.



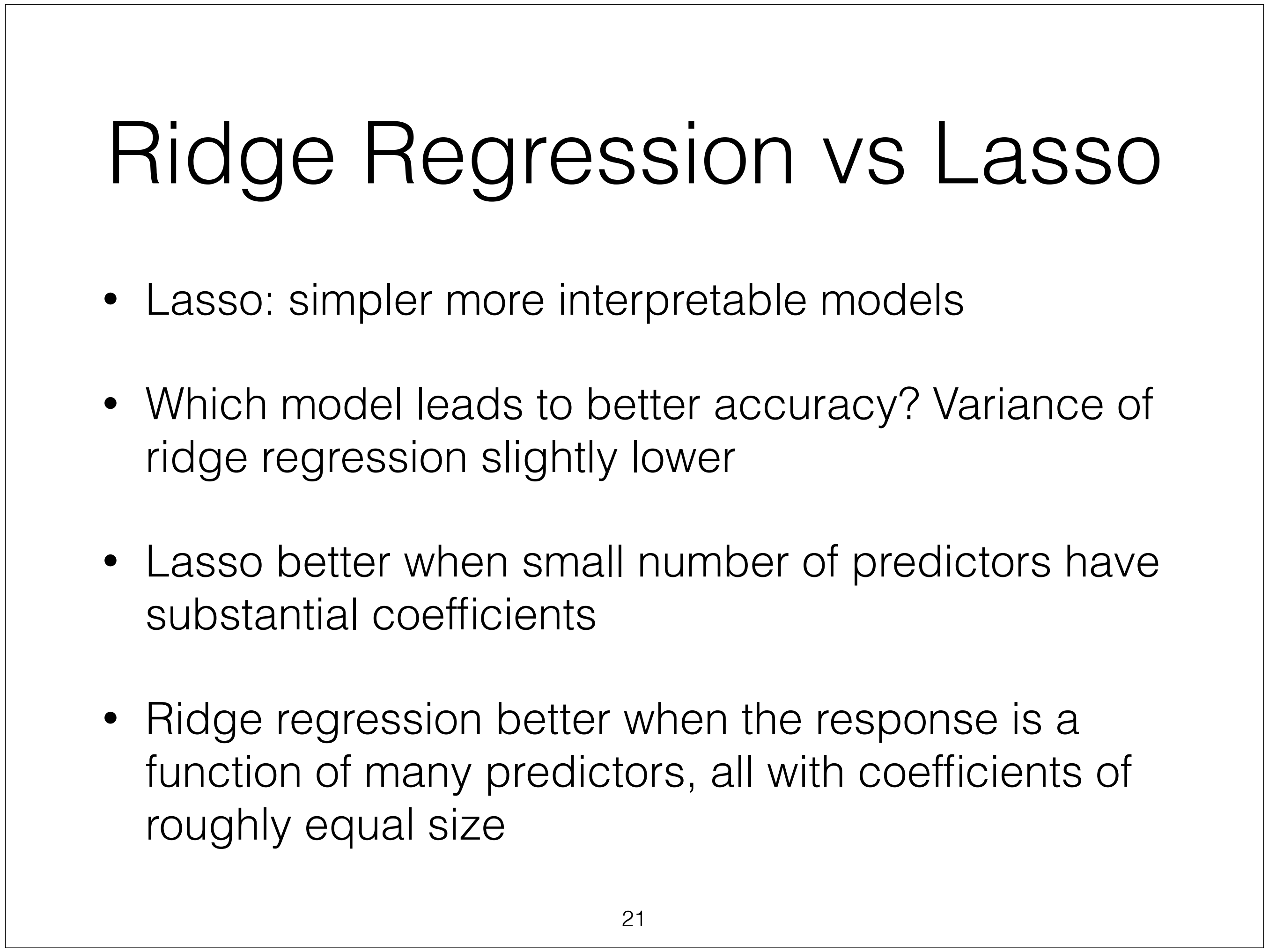
Again for the credit data with Lasso we see the size of the standardised coefficients as we vary λ. We see that this is different to the ridge regression. In the left image as we move from a large λ we get a model with only 1 predictor (rating), then student and limit enter the model almost simultaneously, shortly followed by income. Eventually, the remaining variables enter the model. Hence, depending on the value of λ, the lasso can produce a model involving any number of variables.

We can think of these methods as trying to find the set of coefficients estimates that lead to the smallest RSS subject to the constraint that there is budget for how large the norm of the coefficients can be. When this budget is large, the model is not very restrictive and so the coefficients can be large. When this is small the norm must also be small. And another way to think of these methods is as: computationally feasible alternatives to best subset selection.



Why is it that the Lasso, unlike ridge regression, results in coefficient estimates that are exactly equal to zero? Well it has to do with the shape of the budget constraint shown in blue here for p=2 example. When the budget is large the blue area will contain $\hat{\beta}$ the least square solution and these methods will have the same result as least squares. But given a smaller budget so that $\hat{\beta}$ lies outside it we get an estimate different from least squares since both the constraint and RSS must be minimised and so the solution will lie on the edge of the ellipses and budget shapes.

The L2 shape is smoother like the circle and the L1 has edges or corners and the ellipse will often intersect the budget constraint on an axis where one of the coefficient is zero. In higher dimensions the lasso becomes a polyhedron and ridge regression is a hypersphere, but still this principle holds.



Ridge Regression vs Lasso

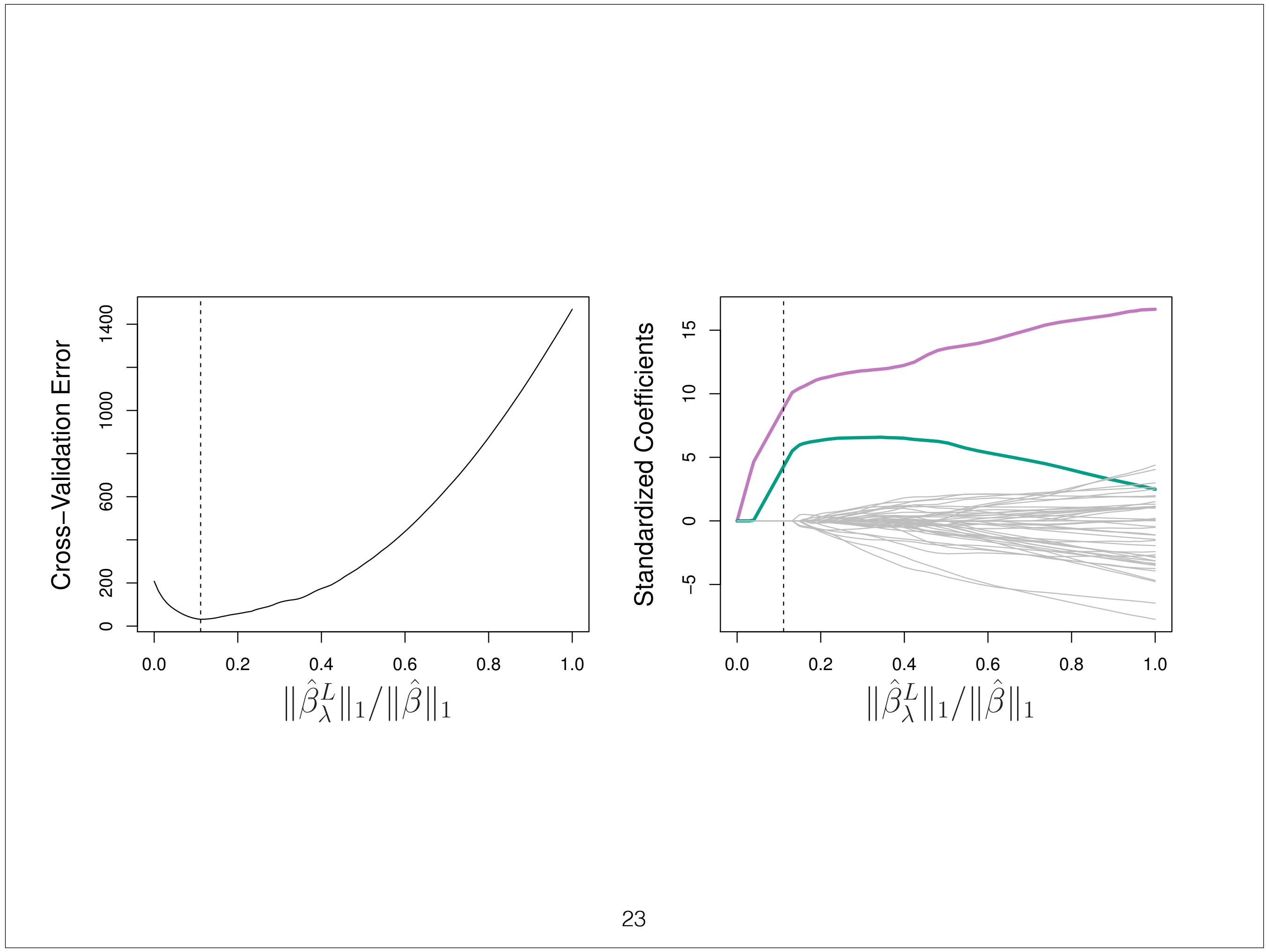
- Lasso: simpler more interpretable models
- Which model leads to better accuracy? Variance of ridge regression slightly lower
- Lasso better when small number of predictors have substantial coefficients
- Ridge regression better when the response is a function of many predictors, all with coefficients of roughly equal size

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.

As with ridge regression, when the least squares estimates have excessively high variance, the lasso solution can yield a reduction in variance at the expense of a small increase in bias, and consequently can generate more accurate predictions. Unlike ridge regression, the lasso performs variable selection, and hence results in models that are easier to interpret.

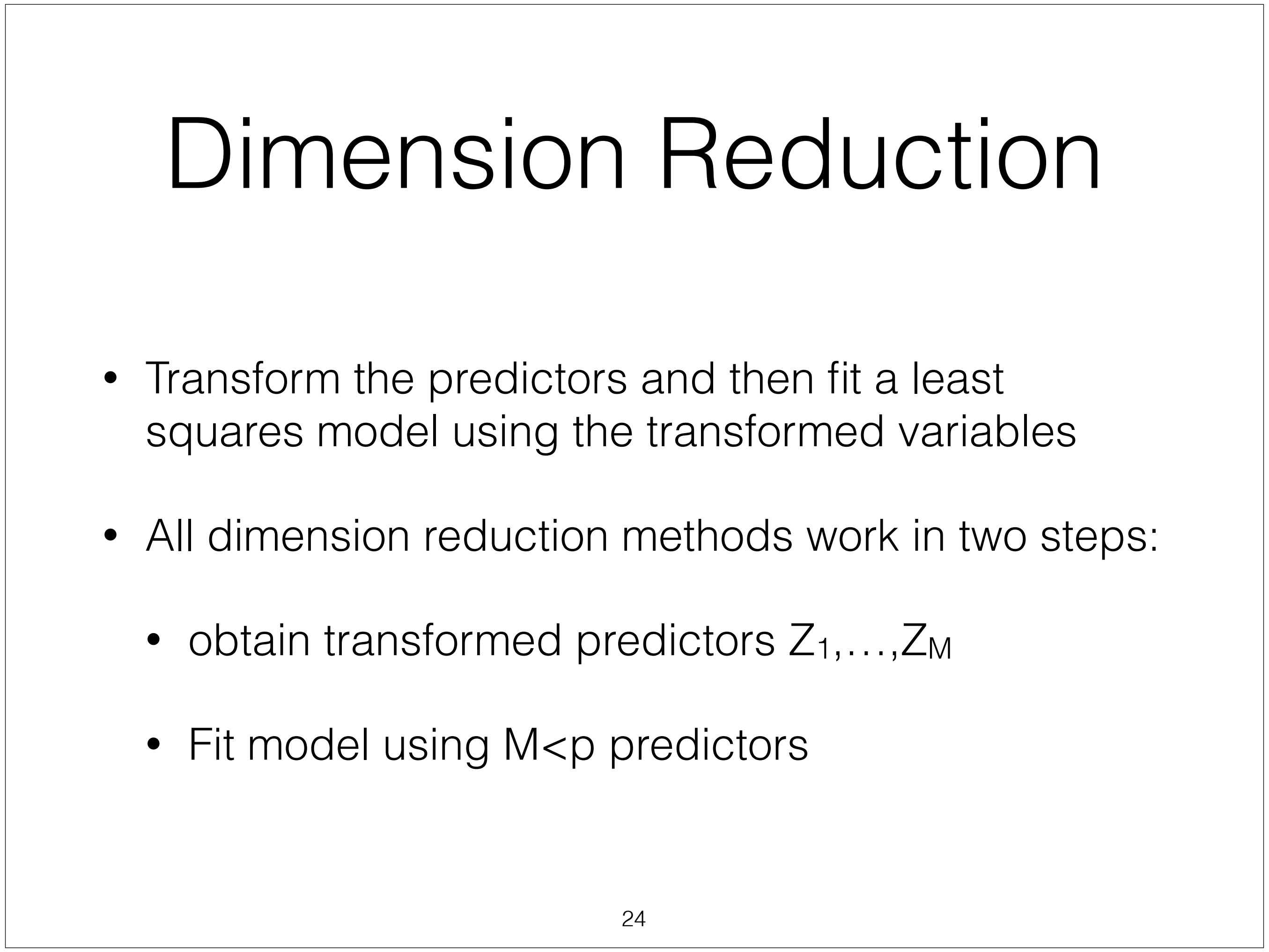
Selecting λ

- Cross-validation provides a simple way to tackle this:
  - Choose a grid of λ values, and compute the cross-validation error for each value of λ
  - Select the tuning parameter value for which the cross-validation error is smallest
  - Model is re-fit using all of the available observations and the selected value of the tuning parameter



So for example we do a 10-fold CV on a Lasso fit for some generated data of 2 real predictors and 45 junk predictors. The vertical dashed lines indicate the point at which the cross-validation error is smallest. The two coloured lines in the right-hand panel represent the two predictors that are related to the response, while the grey lines represent the unrelated predictors; these are often referred to as signal and noise variables, respectively.

Not only has the lasso correctly given much larger coefficient estimates to the two signal predictors, but also the minimum cross-validation error corresponds to a set of coefficient estimates for which only the signal variables are non-zero. Hence cross-validation together with the lasso has correctly identified the two signal variables in the model, even though this is a challenging setting, with p = 45 variables and only n = 50. In contrast least-squares given by the right edge of the right panel assigns a large value to only one of the good predictors.

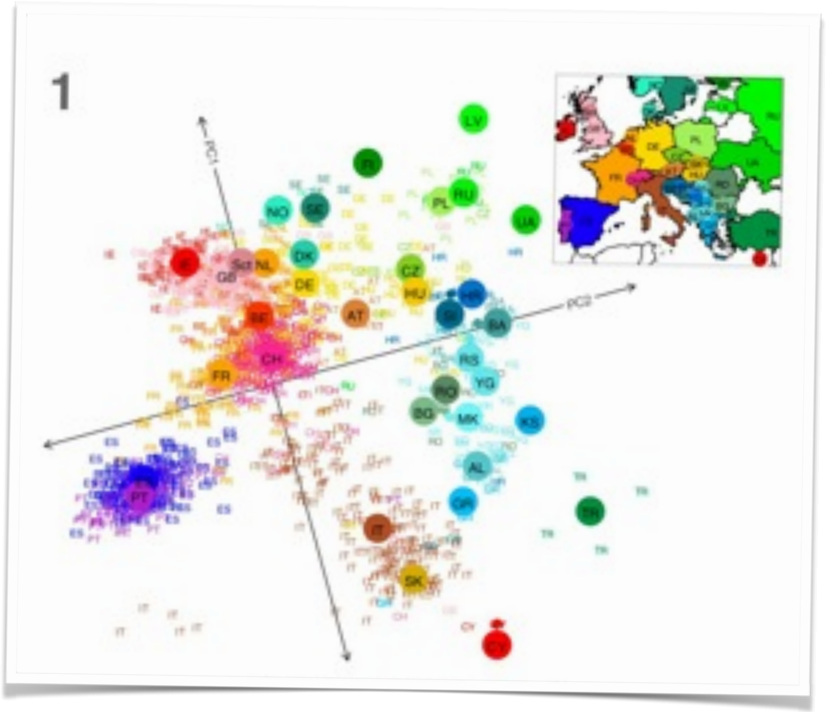


Dimension Reduction

- Transform the predictors and then fit a least squares model using the transformed variables
- All dimension reduction methods work in two steps:
  - obtain transformed predictors $Z_1,\dots,Z_M$
  - Fit model using $M<p$ predictors

The choice of the $Z_1,\dots,Z_M$ can be achieved in a number of different ways. We will look at PCA.

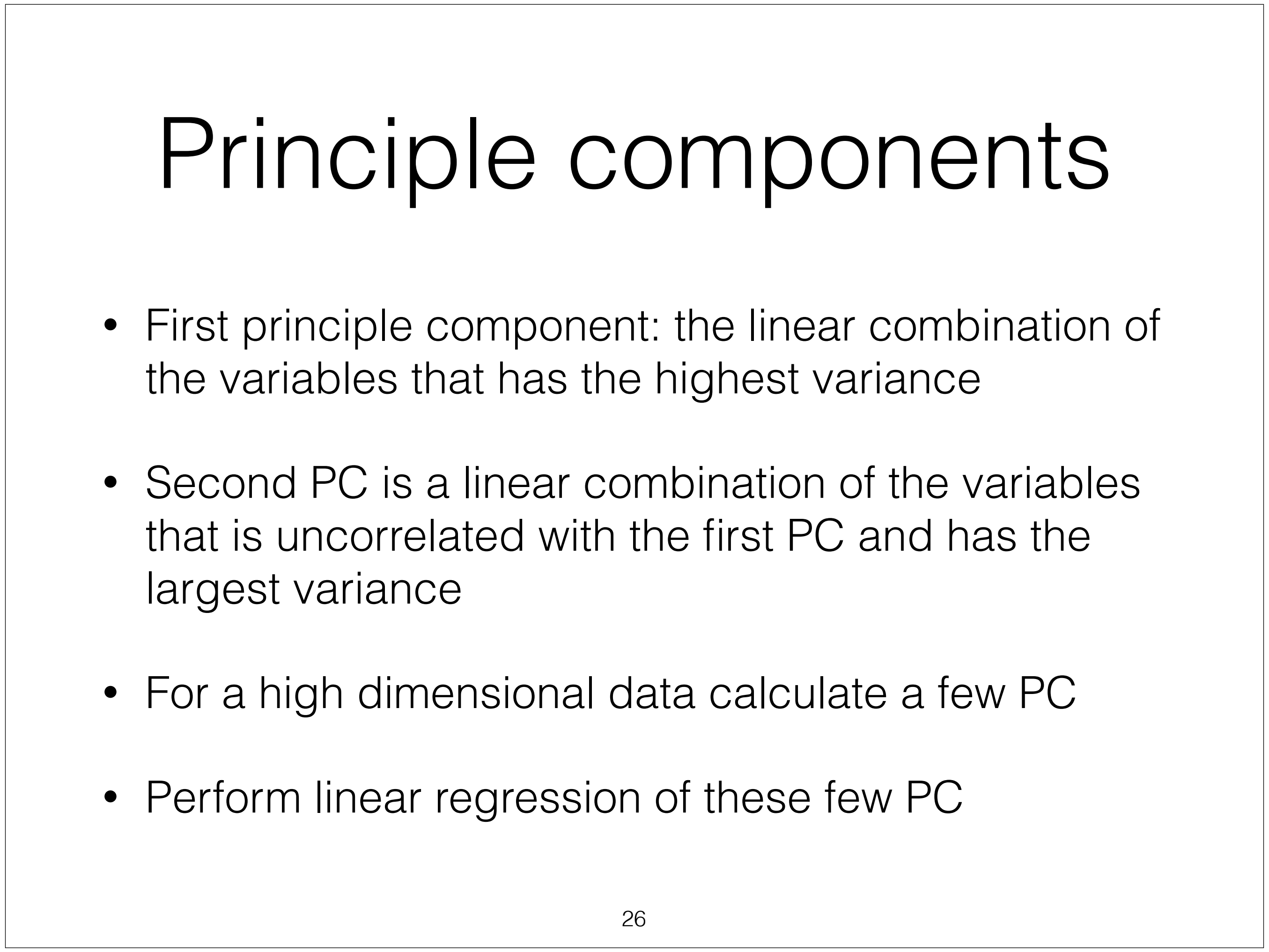
PCA



The general idea of PCA a very versatile technique used in many branches of science is summarised in this picture of genetic data of 3,192 people drawn from throughout Europe. When the researchers looked at the DNA of any two individuals, they found that the number of genetic differences between them was proportional to the geographic distance that separates their respective home countries. Even within countries the researchers saw that groups with similar cultural histories shared similar genetics.

Essentially PCA is technique for reducing the dimension of a n × p data matrix X. The first principal component direction of the data is that along which the observations vary the most. In this gene example it was the latitude feature. So if we projected the data onto this component then the resulting projected observations would have the largest possible variance; projecting the observations onto any other line would yield projected observations with lower variance.

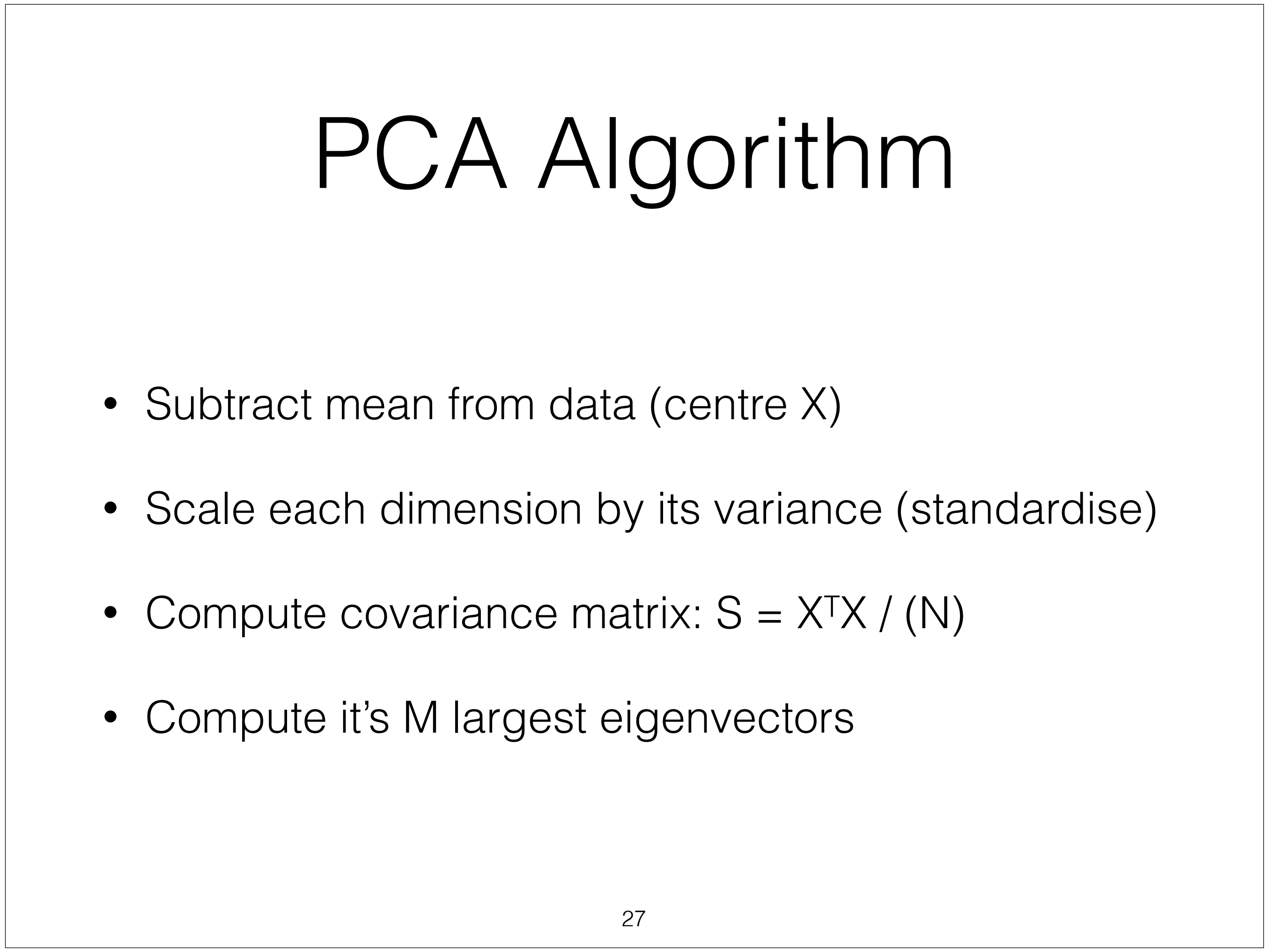
In general, one can construct up to p distinct principal components. The second principal component Z_2 is a linear combination of the variables that is uncorrelated with Z_1, and has largest variance subject to this constraint. The zero correlation condition of Z_1 with Z_2 is equivalent to the condition that the direction must be perpendicular. And the second correlation on the gene data gives us an approximate map of Europe!



Principle components

- First principle component: the linear combination of the variables that has the highest variance
- Second PC is a linear combination of the variables that is uncorrelated with the first PC and has the largest variance
- For a high dimensional data calculate a few PC
- Perform linear regression of these few PC

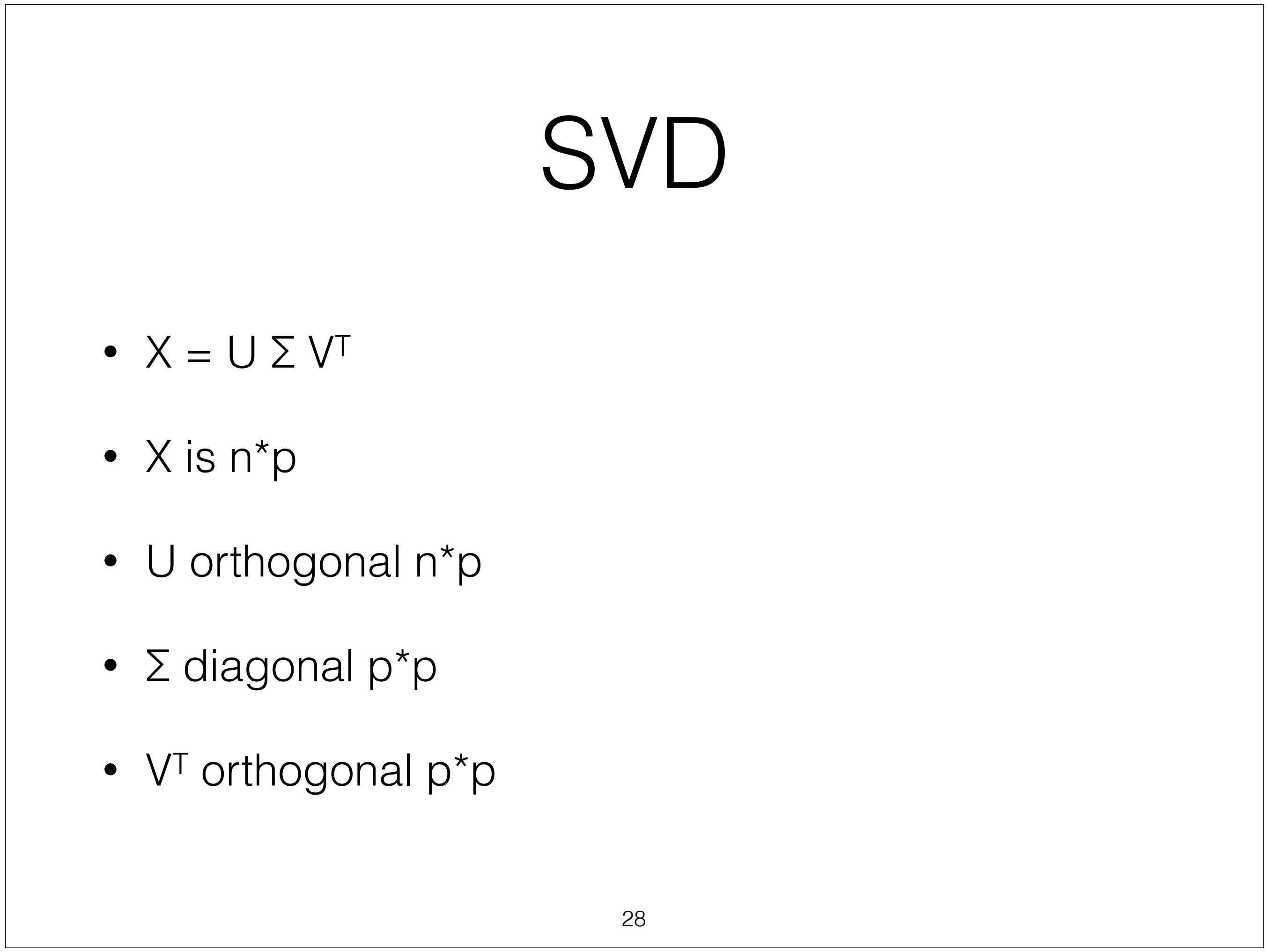
There is also another interpretation for PCA: the first principal component vector defines the line that is as close as possible to the data.



PCA Algorithm

- Subtract mean from data (centre X)
- Scale each dimension by its variance (standardise)
- Compute covariance matrix: S = X^TX / (N)
- Compute it’s M largest eigenvectors

The issue with this algorithm is that X might be huge or multiplying two small floats might lead to precision issues numerically. This is not the optimal way of doing PCA. A better approach is SVD.



SVD

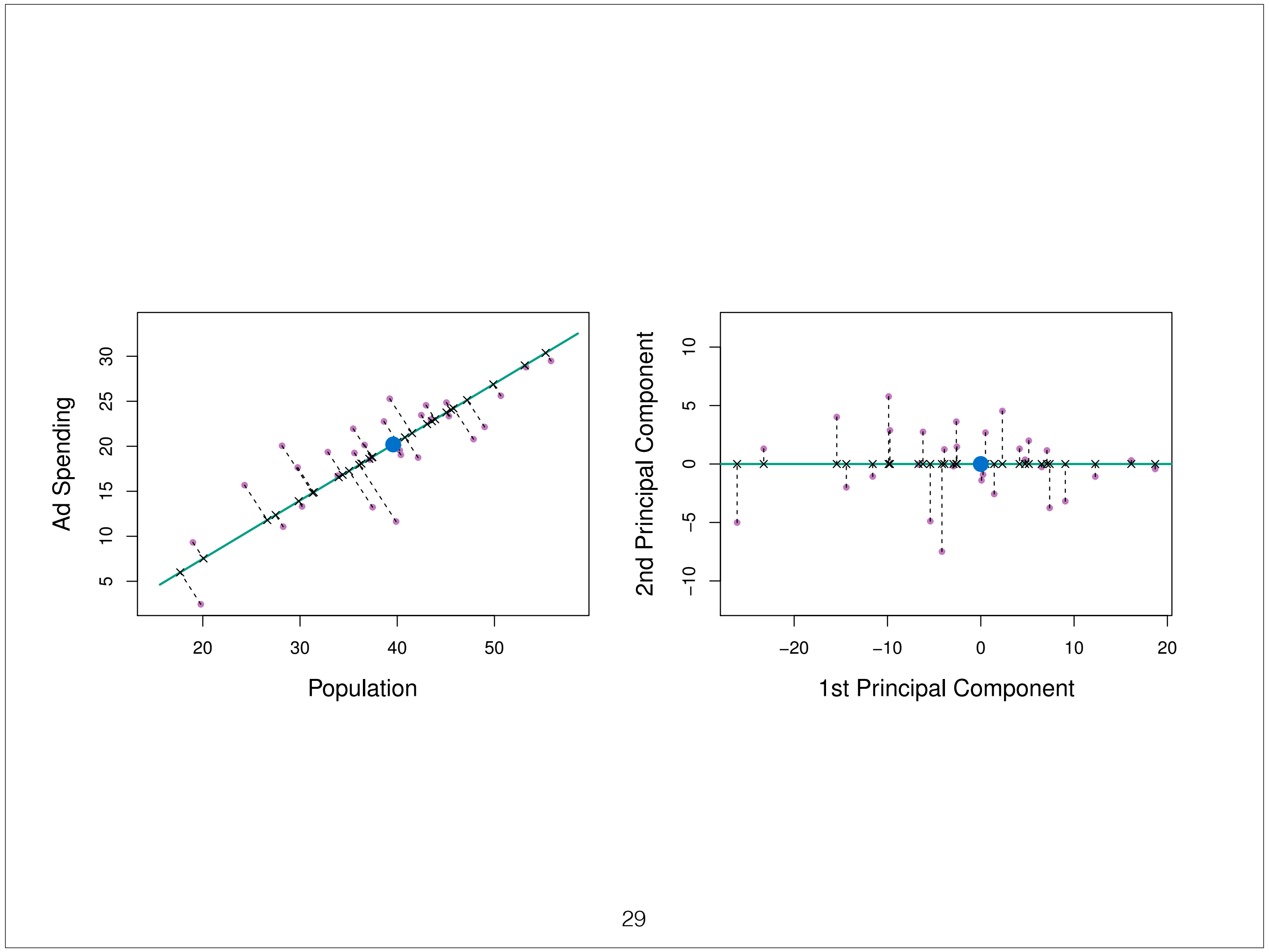
- $X = U \Sigma V^T$
- X is n*p
- U orthogonal n*p
- Σ diagonal p*p
- V^T orthogonal p*p

SVD or singular value decomposition is a very useful linear transformation and comes up in many applications and circumstances. Turns out we can take any n x p matrix X whatever it is, and we can decompose it into a product of 3 matrices. The U has the same dimension as the data matrix and the other two is square of size k*k.

Turns out that if we plug this into the formula for the covariance matrix S, the vectors in V are the eigenvectors of X (or the principle components) and the diagonals of the Σ matrix are the square root of the eigenvalues of S. And this is a very efficient way of getting the principle components in a robust way.

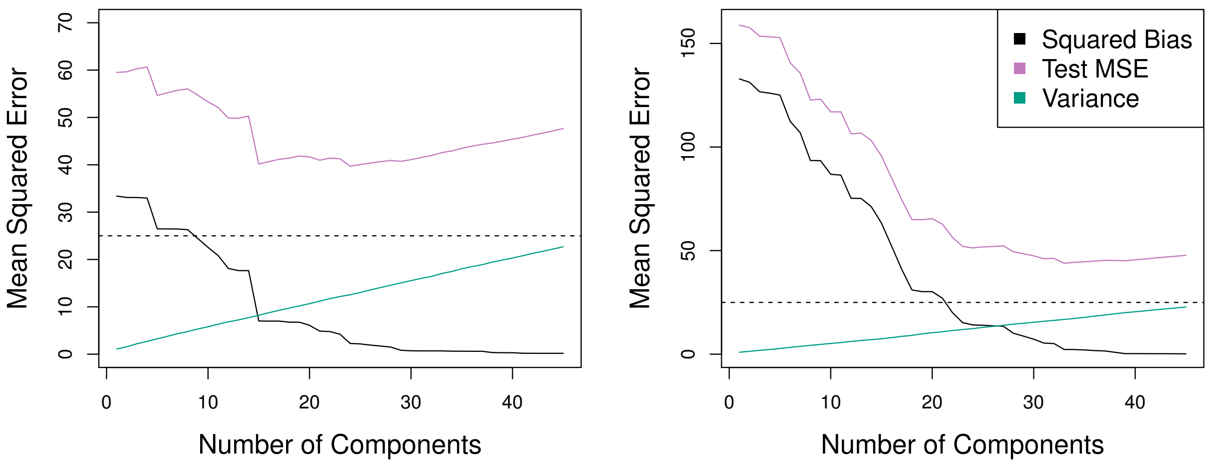
In fact we can even use the SVD to compute the pseudo-inverse of a matrix, which plays the role of the inverse of a matrix which is singular. So SVD is a very useful.

And for dimensionality reduction we can do an approximate SVD where the size of U, Σ, and V^T is n*M, M*M, and M*M respectively for some M<p.



So for the advertising data we have a plot here of population and Ad Spending. The first principal component direction is shown in green. It is the dimension along which the data vary the most, and it also defines the line that is closest to all n of the observations. The blue dot is the mean of both ad spending and population. On the right we rotate it so we see the axis of the 2 principle components.

The principal components regression (PCR) approach then uses 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 $X_1,\ldots,X_p$ shows the most variation are the directions that are associated with Y. While this is not guaranteed to be true, it is a reasonable assumption and gives good results.

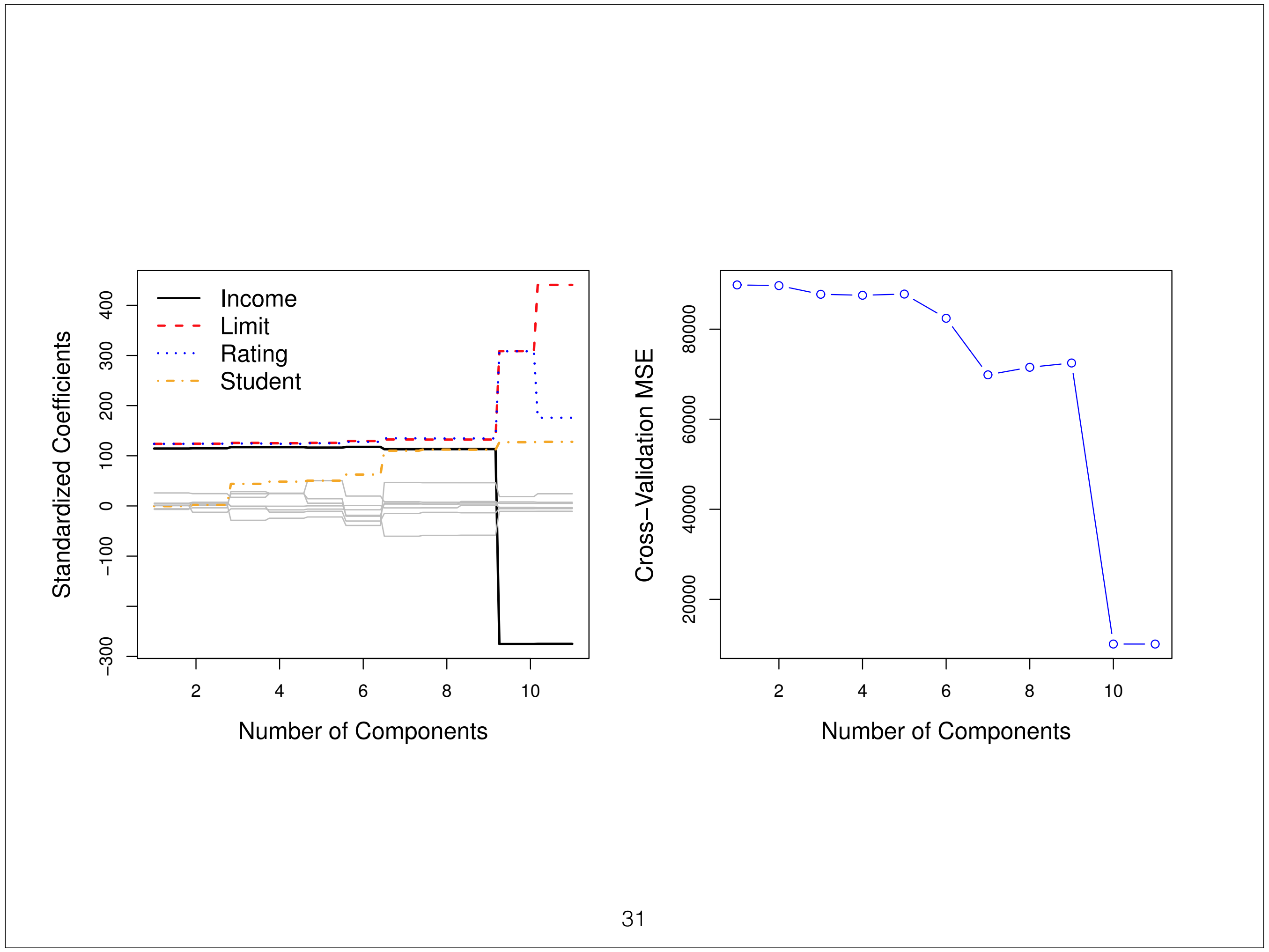


So let us look at our simulated data where n=50 and p=45 and the first data (left) is a function of all the predictors and the second one (right) is a function of only 2 with the rest being noise. And we plot these curves as a function of the number of components M.

As more PC are used in the regression model, the bias decreases, but the variance increases and we get a typical U-shaped curve for the MSE. When M=45 we are back to least-squares. And we see that performing PCA with a good M leads to an improvement over least square. It is however not as good as the shrinkage methods.

The worse performance of PCR is a consequence of the fact that the data were generated in such a way that 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.

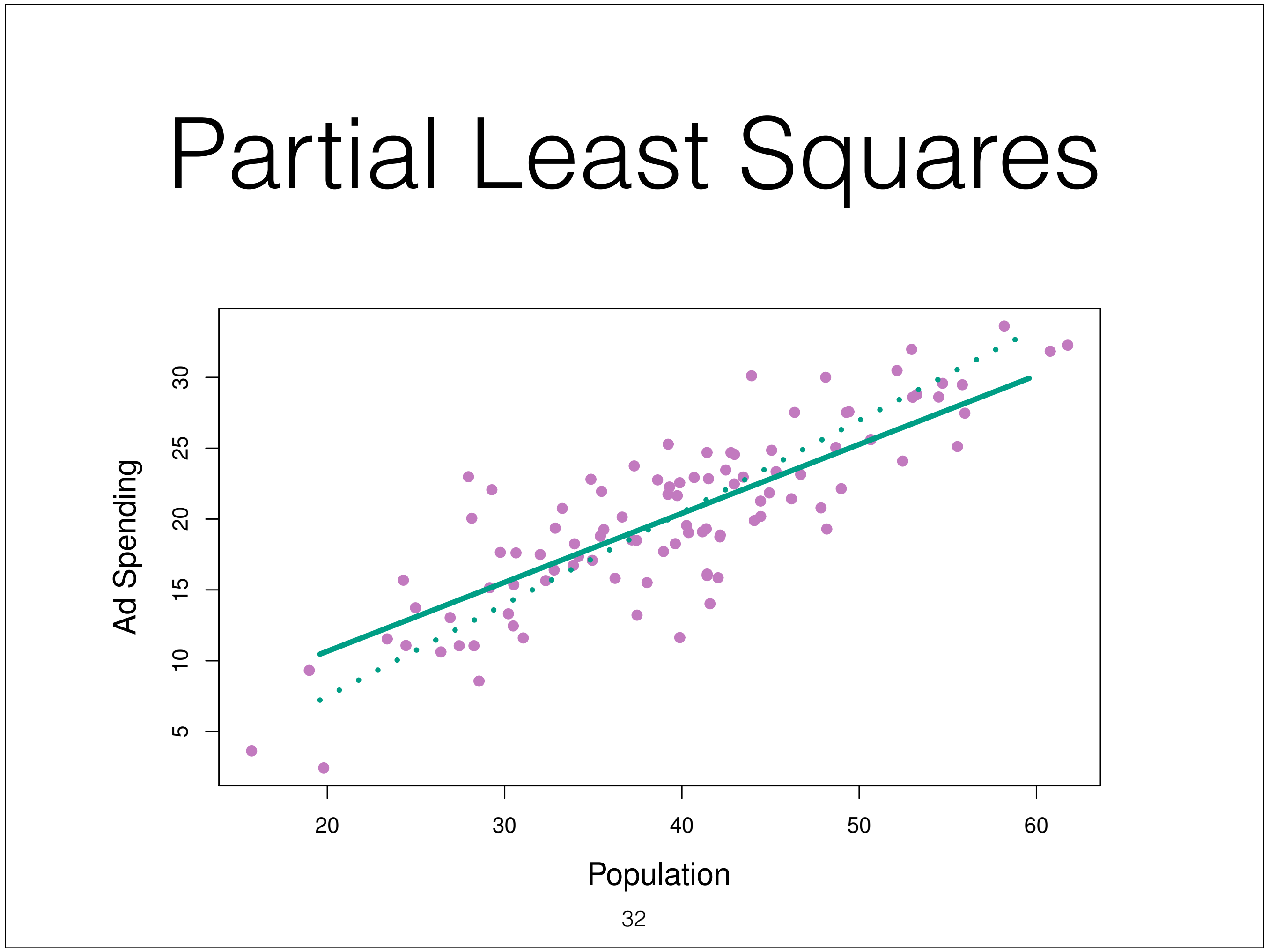
Also note that PCA for M<p is not a feature selection method: each of the M principal components used in the regression is a linear combination of all p of the original features.



In PCR, the number of principal components, M, is typically chosen by cross-validation. The results of applying PCR to the Credit data set are shown above. The lowest cross- validation error occurs when there are M = 10 components; this corresponds to almost no dimension reduction at all, since PCR with M = 11 is equivalent to simply performing least squares.

When performing PCR, we generally recommend standardising each predictor.

We can also ask how much of the information in a given data set is lost by projecting the observations onto the first few principal components? That is, how much of the variance in the data is not contained in the first few principal components? More generally, we are interested in knowing the proportion of variance explained (PVE) by each principal component. And we will see later how to compute and plot this as a function of the number of components via a scree plot. We will discuss more when talking about unsupervised methods.



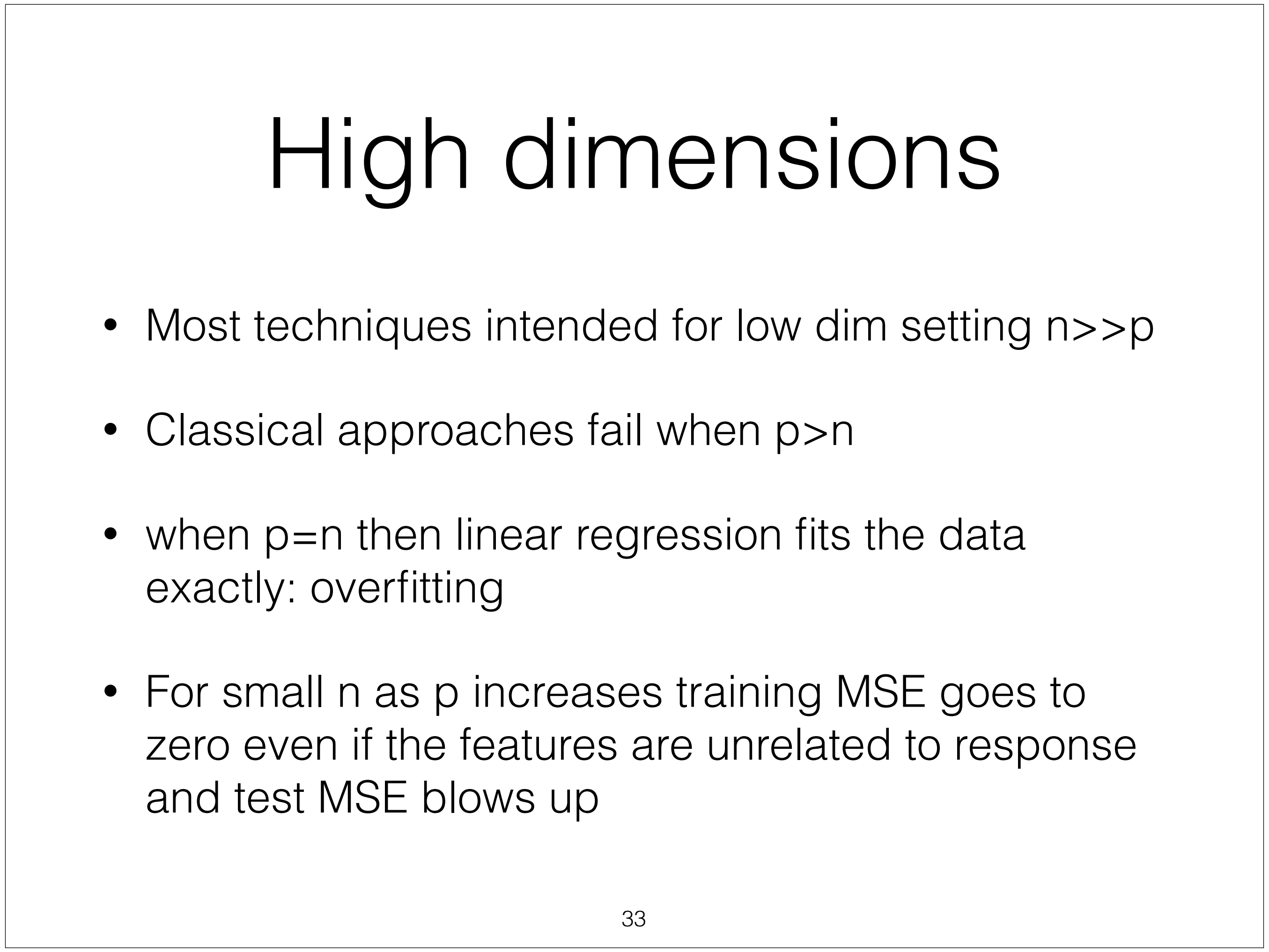
Partial Least Squares

The PCR approach that we just described involves identifying linear combinations, or directions, that best represent the predictors $X_1, \ldots, X_p$. These directions are identified in an unsupervised way, since the response Y is not used to help determine the principal component directions. So 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.

Partial least squares (PLS), is a supervised alternative to PCR. Unlike PCR, PLS identifies these new 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, the PLS approach attempts to find directions that help explain both the response and the predictors.

We see an example of PLS on the advertising data. The solid green line indicates the first PLS direction, while the dotted line shows the first principal component direction. The PLS direction does not fit the predictors as closely as does PCA, but it does a better job explaining the response.

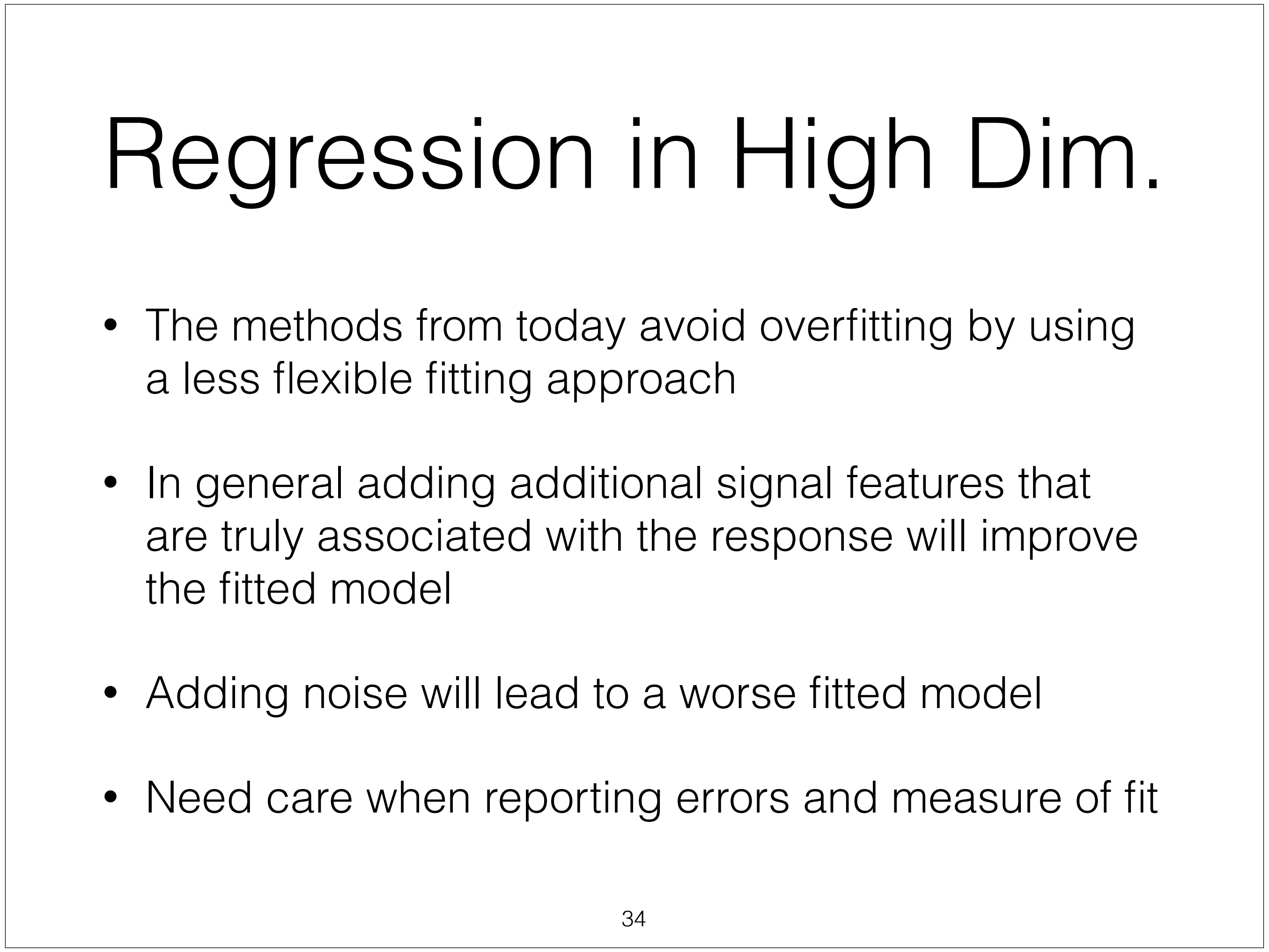
In practice it often performs no better than ridge regression or PCR, since it can reduce bias but it also has the potential to increase variance. PCR is popular in chemometrics where many variables arise from digitised spectrometry signals.



High dimensions

- Most techniques intended for low dim setting n>>p
- Classical approaches fail when p>n
- when p=n then linear regression fits the data exactly: overfitting
- For small n as p increases training MSE goes to zero even if the features are unrelated to response and test MSE blows up

These problems indicate the importance of applying extra care when analysing data sets with a large number of variables, and of always evaluating model performance on an independent test set.



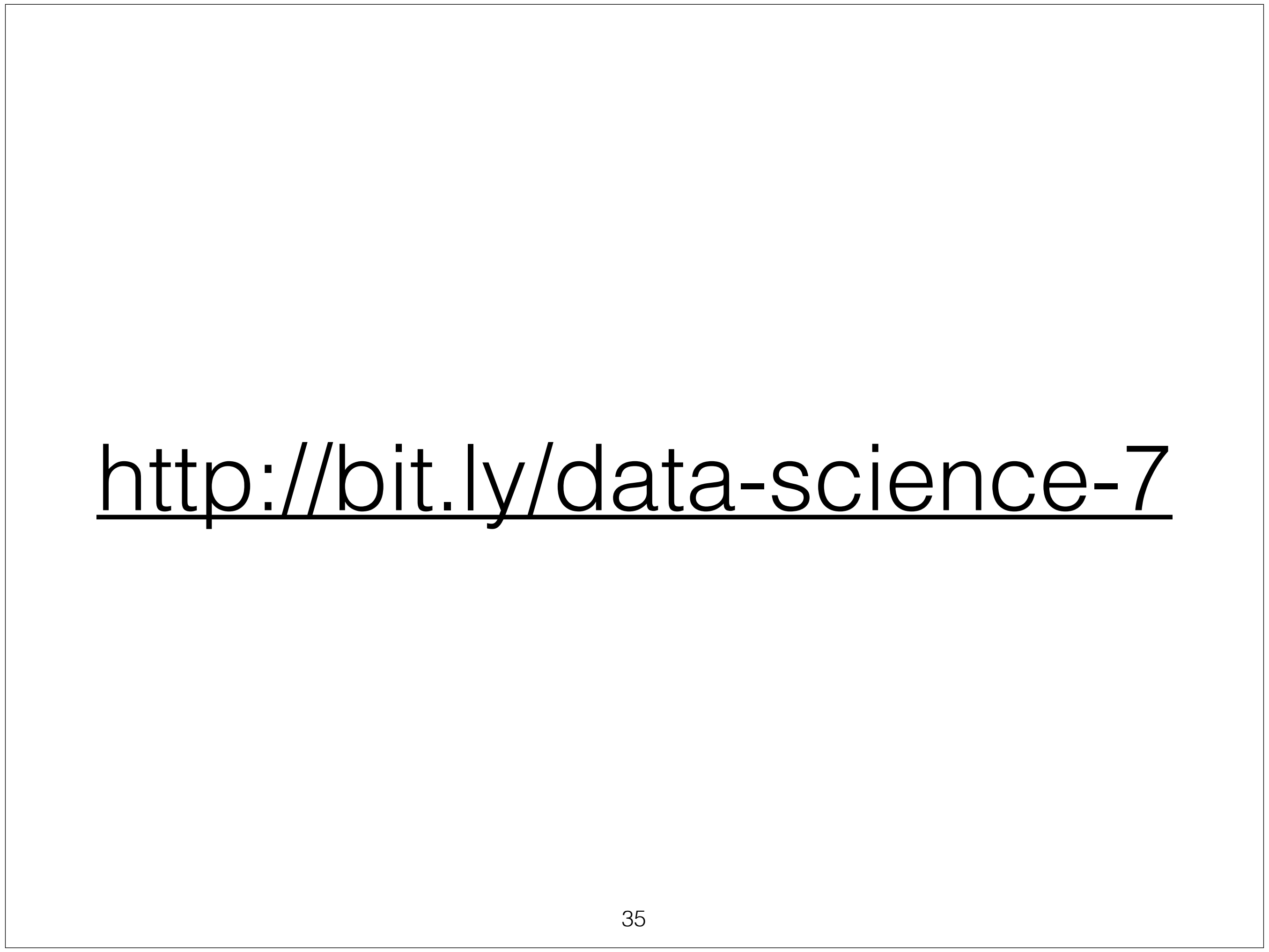
Regression in High Dim.

- The methods from today avoid overfitting by using a less flexible fitting approach
- In general adding additional signal features that are truly associated with the response will improve the fitted model
- Adding noise will lead to a worse fitted model
- Need care when reporting errors and measure of fit

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 without any potential upside in terms of improved test set error.

Thus, we see that new technologies that allow for the collection of measurements for thousands or millions of features are a double-edged sword: they can lead to improved predictive models if these features are in fact relevant to the problem at hand, but will lead to worse results if the features are not relevant.

It is also important to be particularly careful in reporting errors and measures of model fit in the high-dimensional setting. When p > n, it is easy to obtain a useless model that has zero residuals. Therefore, one should never use the statistics from the past lectures and instead report results on an independent test set or CV errors.



http://bit.ly/data-science-7

Thank you that’s it for this lecture. Your feedback is greatly appreciated as it will help us to improve and drive the course in the direction you would like it to go.