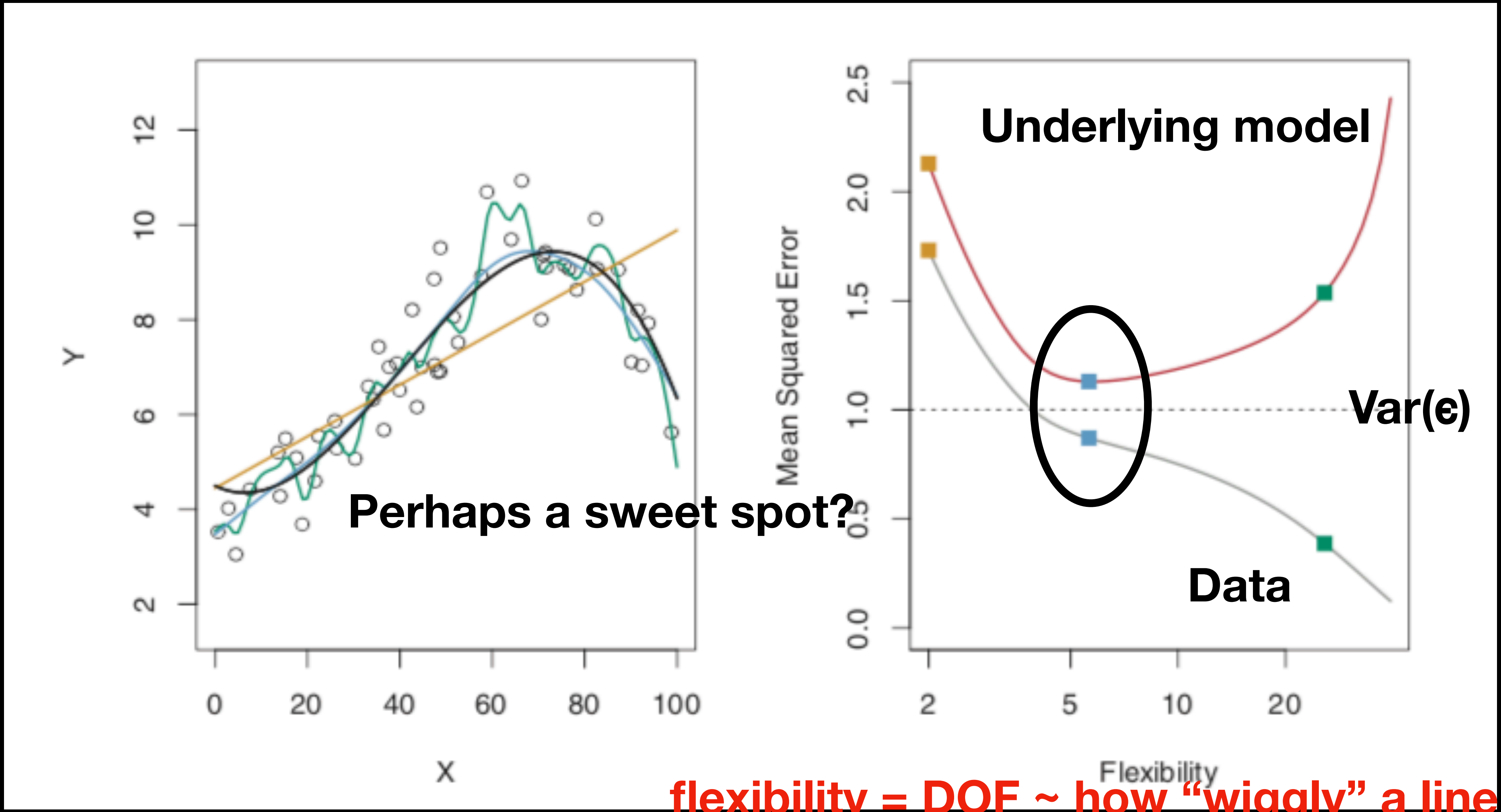
Welcome to Week 16!

You made it!!

Bias-Variance Trade-Off (First Glance)

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon).$$



flexibility = DOF ~ how "wiggly" a line is

—	Actual underlying function - y
o	Simulated data with added error (є)
🟧	Linear fit
🟦	Low "flexibility" smooth spline
🟩	High "flexibility" smooth spline

fits data well, but underlying model badly

So far…

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots + \beta_n x_n$$

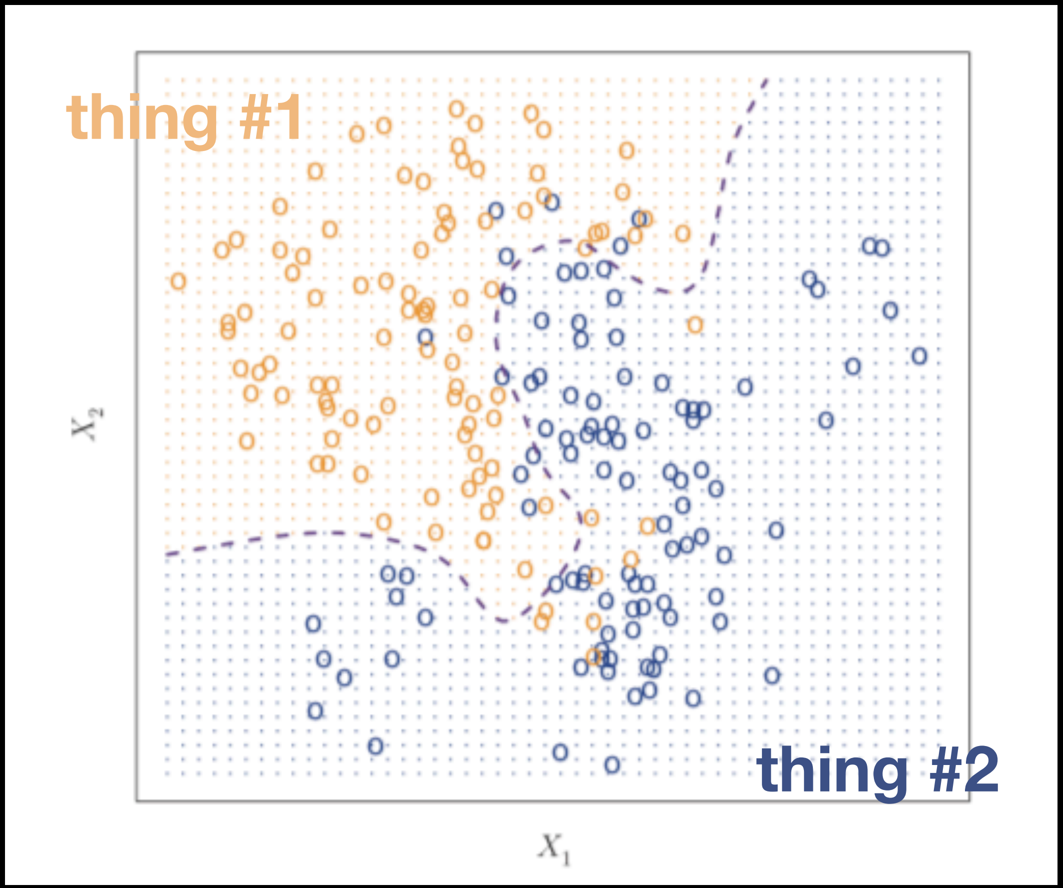
predicted y

intercept

So far we've been saying:
"I have a basic idea of what the functional form of y looks like (a line or a logit) - computer go find the parameters of that functional form.

This is nice because we have some hope of gaining intuition from our models.

Now we classify…

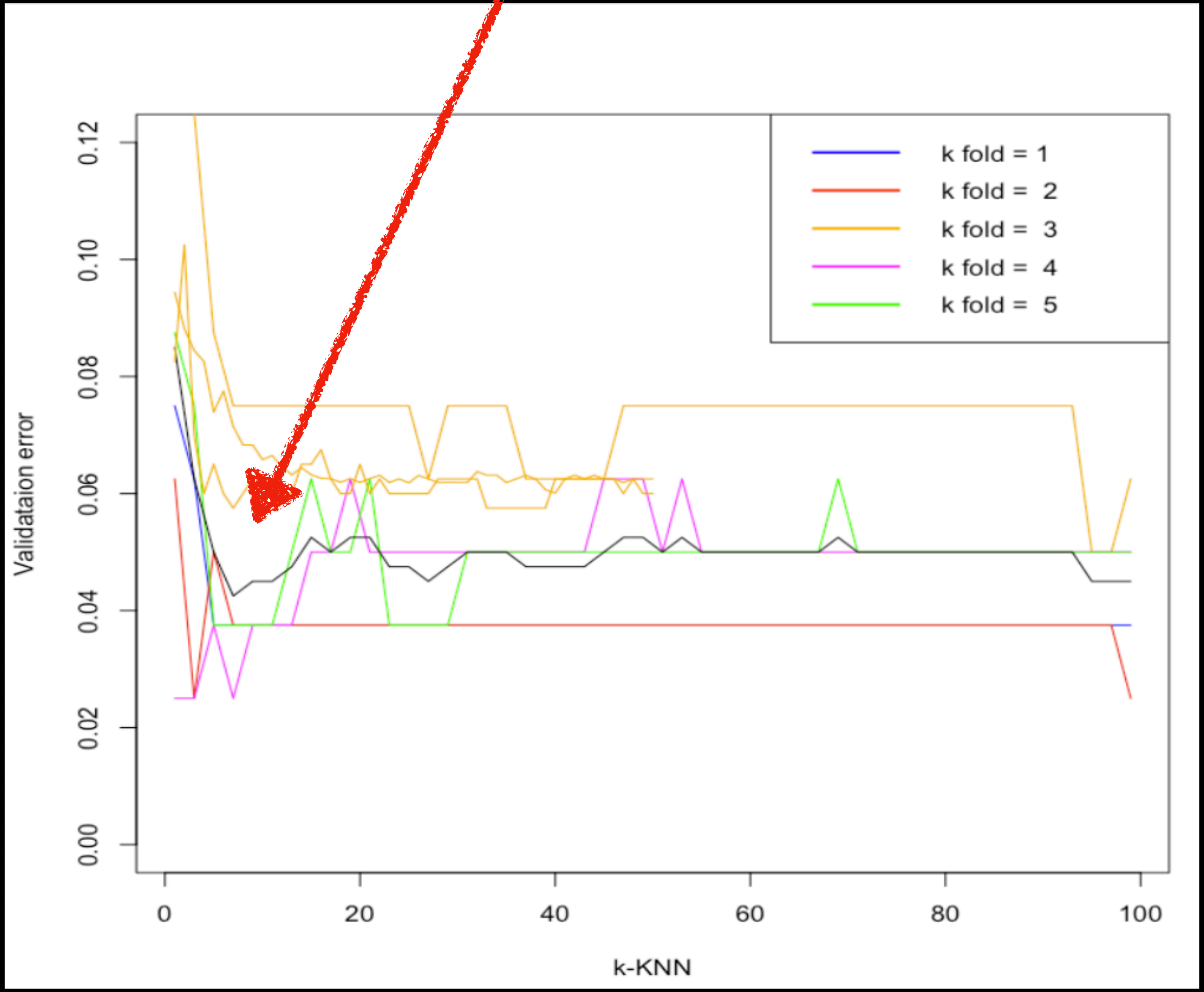


"I want to know where thing #1 and thing #2 live in some 2D space - computer, go figure out the boundary between these two things and let me know"

This is nice because we don't have to assume some model beforehand.

Cross-Validation Methods

Best KNN k is about here



But we are only able to calculate the test error because we know the background distribution.

Last Week

Cross-Validation (Ch. 5)

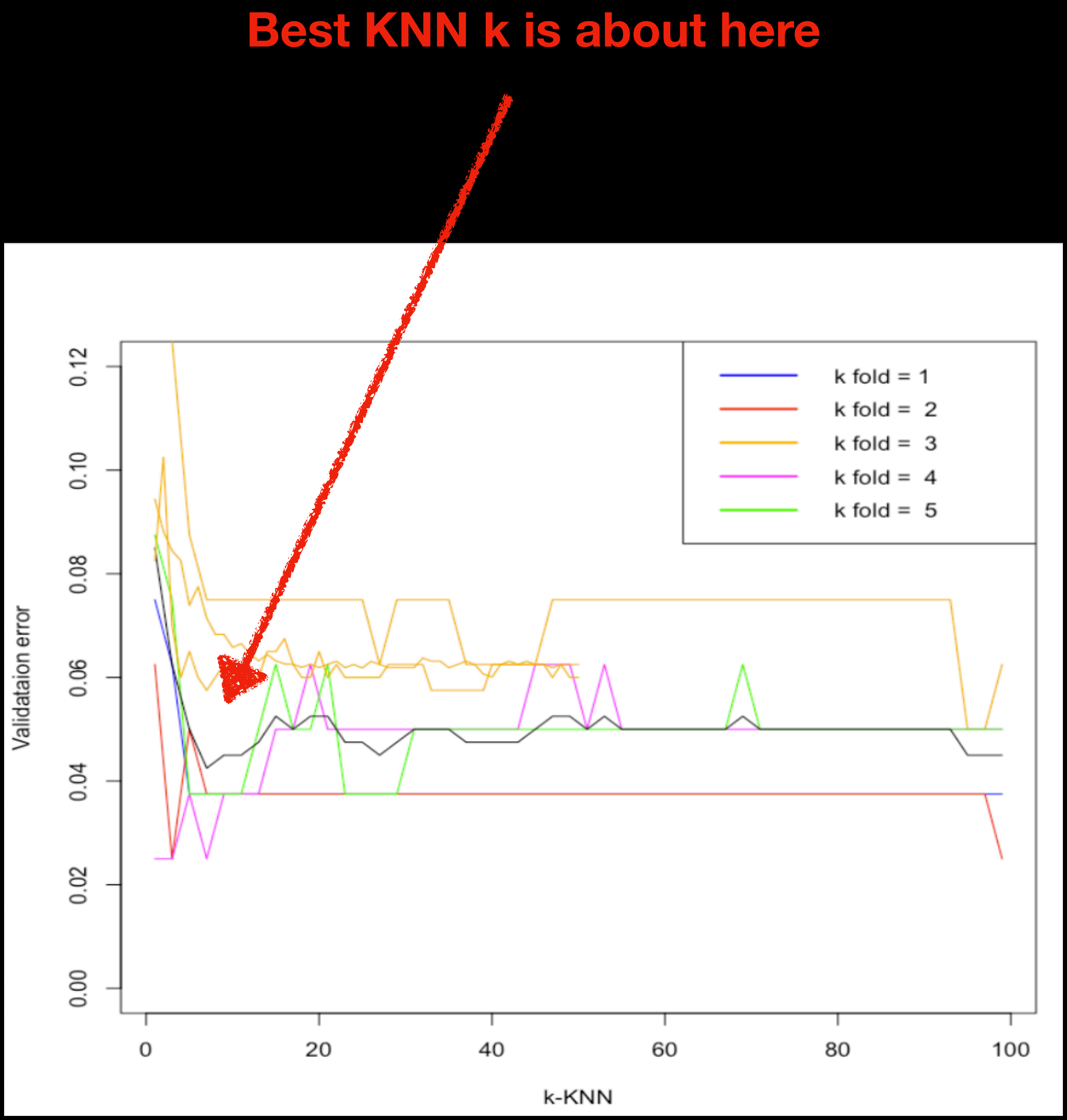
Test fits with subsets of data

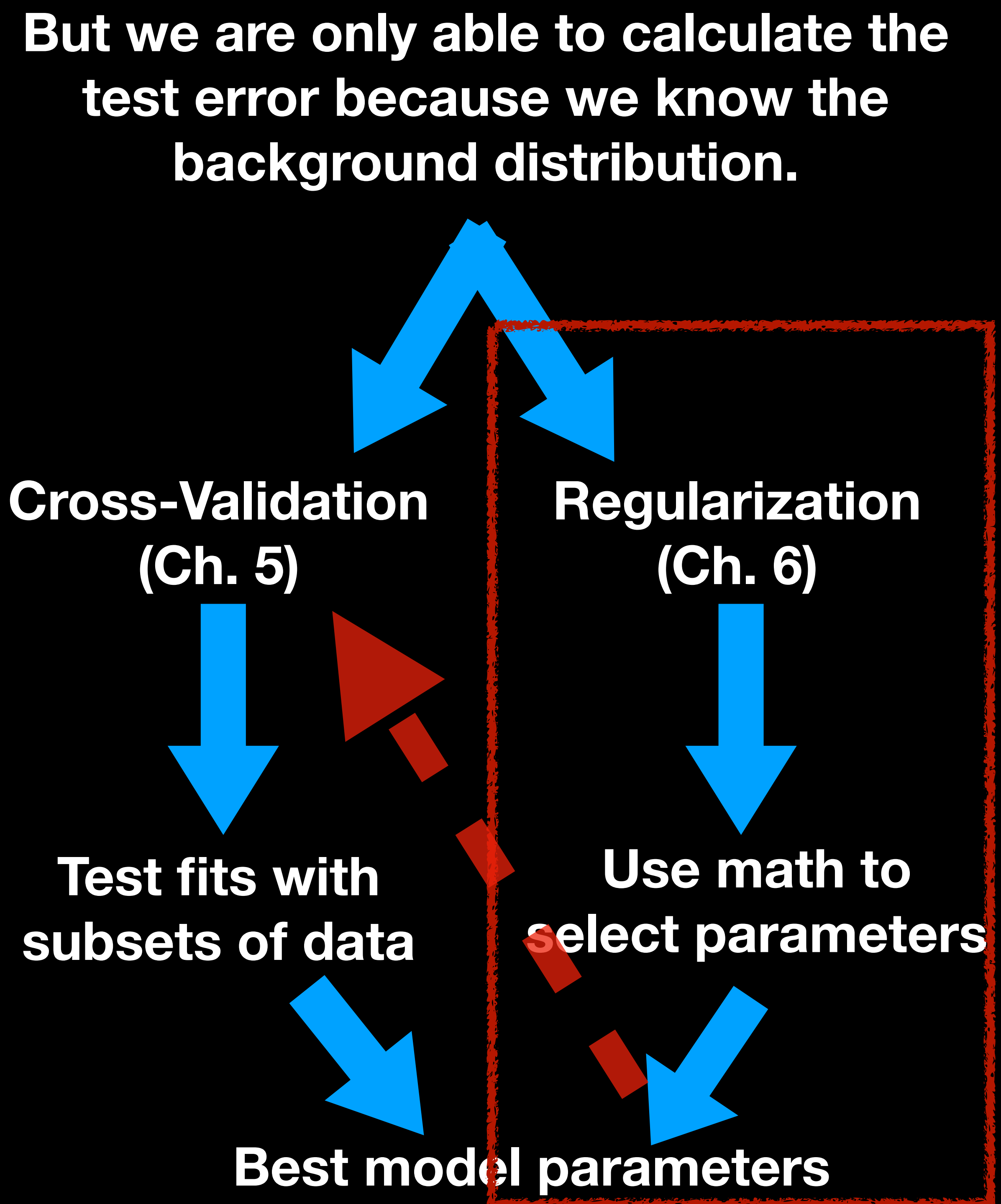
Regularization (Ch. 6)

Use math to select parameters

Best model parameters

Cross-Validation Methods





But we are only able to calculate the test error because we know the background distribution.
Cross-Validation (Ch. 5)
Regularization (Ch. 6)
Test fits with subsets of data
Use math to select parameters
Best model parameters

Improvements on the Linear Model

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon$$

Linear models are great! (Interpretability & predictive performance)

But we can enhance them by modifying the process of choosing parameters - alternatives to least squares fitting

Prediction Accuracy: especially when $p > n$, to control the variance.

Model Interpretability: By removing irrelevant features — that is, by setting the corresponding coefficient estimates to zero — we can obtain a model that is more easily interpreted. We will present some approaches for automatically performing feature selection.

Improvements on the Linear Model

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon$$

Linear models are great! (Interpretability & predictive performance)

But we can enhance them by modifying the process of choosing parameters - alternatives to least squares fitting

Subset selection

… we actually essentially already covered this!

A few more details…

Subset selection

We identify a subset of the p predictors (of k possible) that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.

Brute Force Way:

1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.

2. For k = 1,2,...p:

   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.

   (b) Pick the best among these p models, and call it M_k. Here k best is defined as having the smallest RSS, or equivalently largest R^2.

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

Subset selection

We identify a subset of the p predictors (of k possible) that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.

Brute Force Way:

1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.

2. For k = 1,2,...p:

   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.

   (b) Pick the best among these p models, and call it M_k. Here k best is defined as having the smallest RSS, or equivalently largest R^2.

3. Select a single best model from among $M_0, \ldots, M_p$ using cross-validated prediction error, $\boxed{C_p \text{ (AIC), BIC, or adjusted } R^2.}$

Different measures of goodness of fit - accounting how many parameters are fit

Subset selection

We identify a subset of the p predictors (of k possible) that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.

Brute Force Way:

1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.

2. For k = 1,2,...p:

   (a) Fit all $\binom{p}{k}$ models that conta

   (b) Pick the best among these M_k. Here k best is defined as having the smallest RSS, or equivalently largest R^2.

this number can get big ~ 2^p
p=2: 4 (including null)
p=3: 8
...
p=10: 1024

3. Select a single best model from among $M_0, \ldots, M_p$ using cross-validated prediction error, $\boxed{C_p \text{ (AIC), BIC, or adjusted } R^2.}$

Different measures of goodness of fit - accounting how many parameters are fit

Subset selection

- For computational reasons, best subset selection cannot be applied with very large p.
- Best subset selection may also suffer from statistical problems when p is large: *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.*
- Thus an enormous search space can lead to overfitting and high variance of the coefficient estimates.
- For both of these reasons, stepwise methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection.

Great news: we already covered this!

Subset selection: Forward & Backward (Review)

- Forward stepwise selection 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.
- Like forward stepwise selection, backward stepwise selection provides an efficient alternative to best subset selection.
- However, 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.
- *Neither are guaranteed to find the best possible model out of all 2^p models containing subsets of the p predictors.*
- Backward selection requires that 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.

Subset selection: Forward & Backward (Review)

Find models that minimize:

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

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

d predictors, n data points

estimate from $SE^2/n = \sigma^2$ or RSE

$$\mathrm{AIC} = \frac{1}{n\hat{\sigma}^2}\left(\mathrm{RSS} + 2d\hat{\sigma}^2\right)$$

Idea here is to get a measurement of the mean square error that accounts for # of parameters in a model

$$\mathrm{BIC} = \frac{1}{n\hat{\sigma}^2}\left(\mathrm{RSS} + \log(n)d\hat{\sigma}^2\right)$$

With four parameters I can fit an elephant, and with five I can make him wiggle his trunk.
- von Neumann

Or: large values as R_{adj}^2

Subset selection: Forward & Backward (Review)

Find models that minimize:

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

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

d predictors, n data points

estimate from $SE^2/n = \sigma^2$ or RSE

$$\mathrm{AIC} = \frac{1}{n\hat{\sigma}^2}\left(\mathrm{RSS} + 2d\hat{\sigma}^2\right)$$

Idea here is to get a measurement of the mean square error that accounts for # of parameters in a model

$$\mathrm{BIC} = \frac{1}{n\hat{\sigma}^2}\left(\mathrm{RSS} + \log(n)d\hat{\sigma}^2\right)$$

With four parameters I can fit an elephant, and with five I can make him wiggle his trunk.
- von Neumann

Or: large values as R_{adj}^2

Optional R notes.

Shrinkage Methods

Shrinkage Methods

Lets recall how we find a linear model:

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

Least squares minimization (i.e. minimize RSS)

In contrast, the Ridge or Lasso regression coefficient estimates β^R and β^L are the values that minimize

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

Ridge

Don't panic, its just a bit more math (and we'll get R to do it for us)

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

Lasso

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

Shrinkage Methods

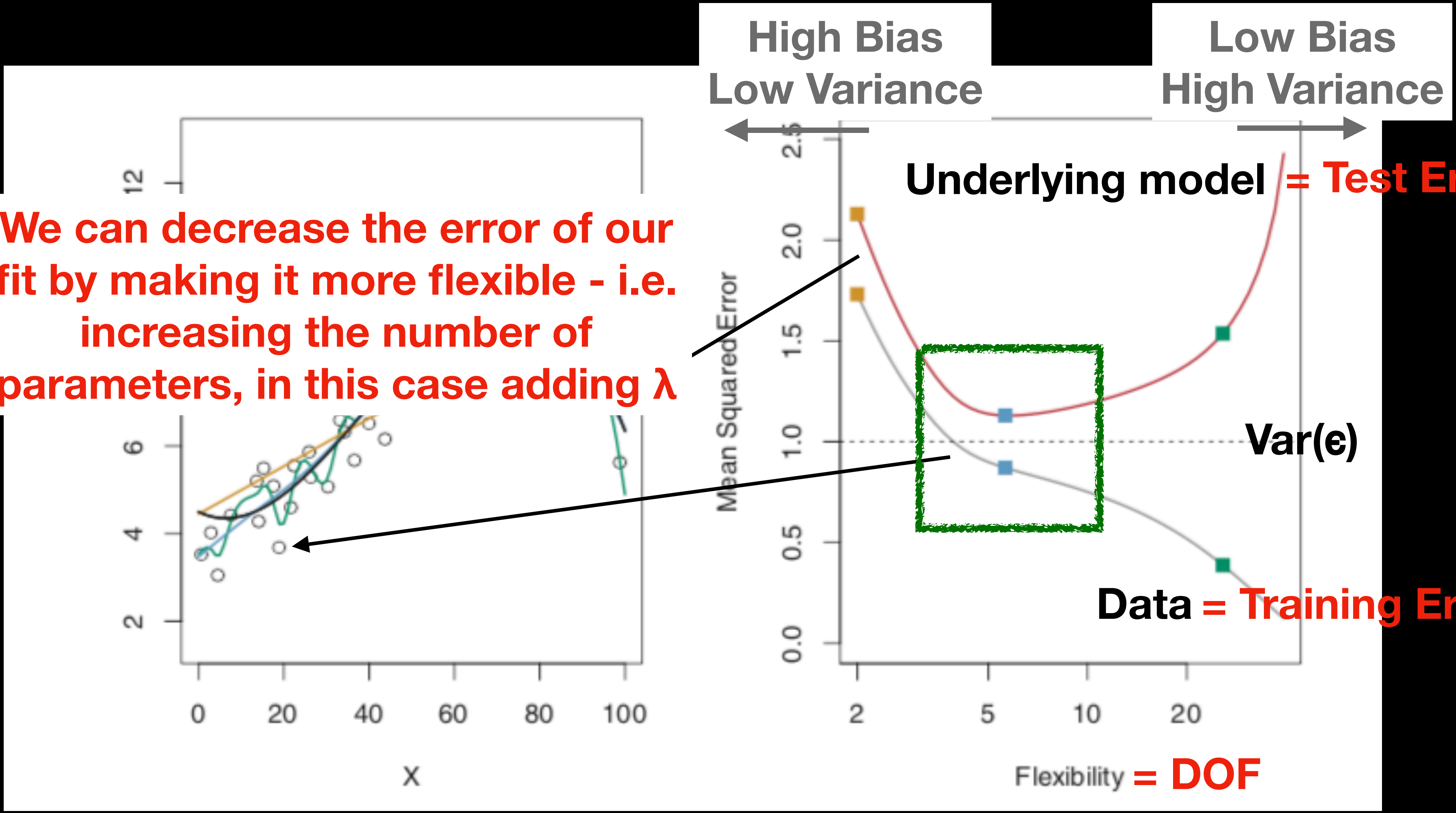
$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \text{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2$$

Ridge

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

Lasso

Why would we make our lives more complicated like this?



We can decrease the error of our fit by making it more flexible - i.e. increasing the number of parameters, in this case adding λ

Shrinkage Methods

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

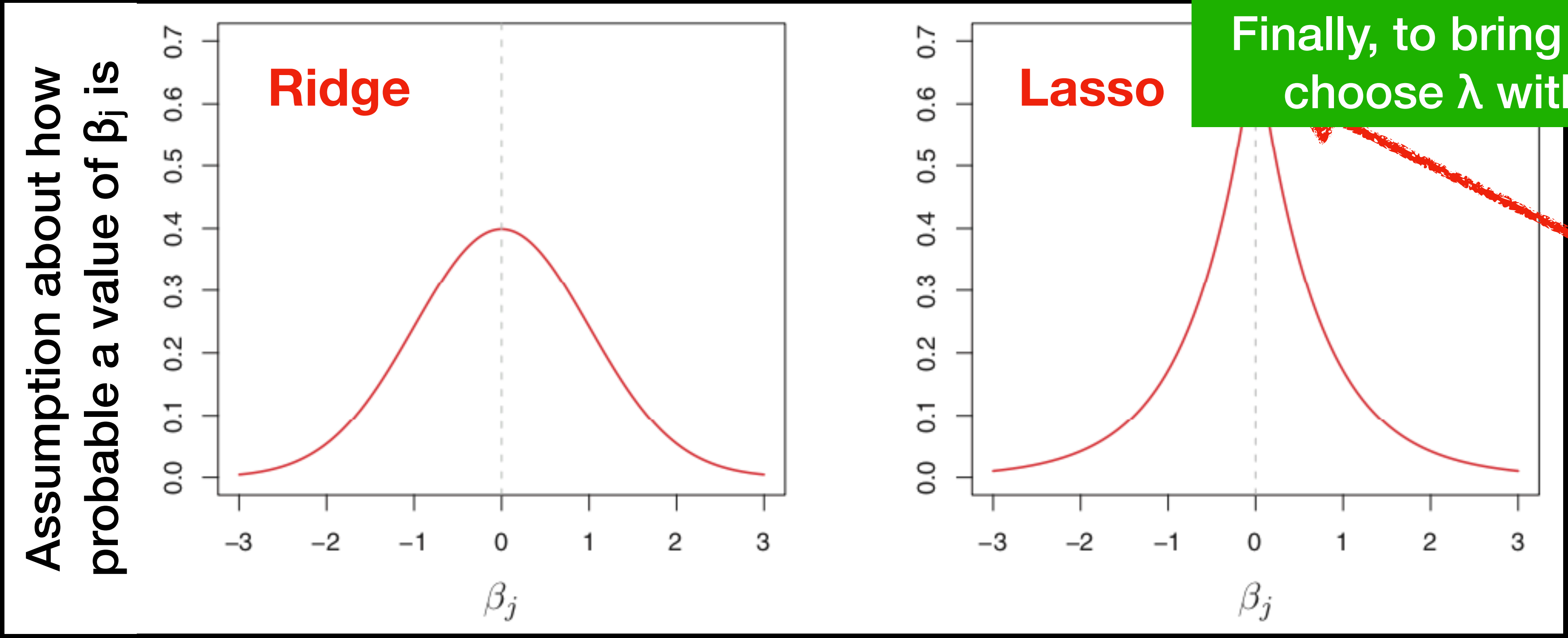
Ridge

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

Lasso

Why would we make our lives more complicated like this?

(1) We can decrease the error of our fit by making it more flexible - i.e. increasing the number of parameters, in this case adding $\boxed{\lambda}$
(2) Natural way to perform variable selection $\rightarrow$ "Shrinkage" of less important parameters to zero



Finally, to bring it full circle - we can choose λ with Cross-Validation

Parameters are more likely to be zero

Shrinkage Methods

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

Ridge

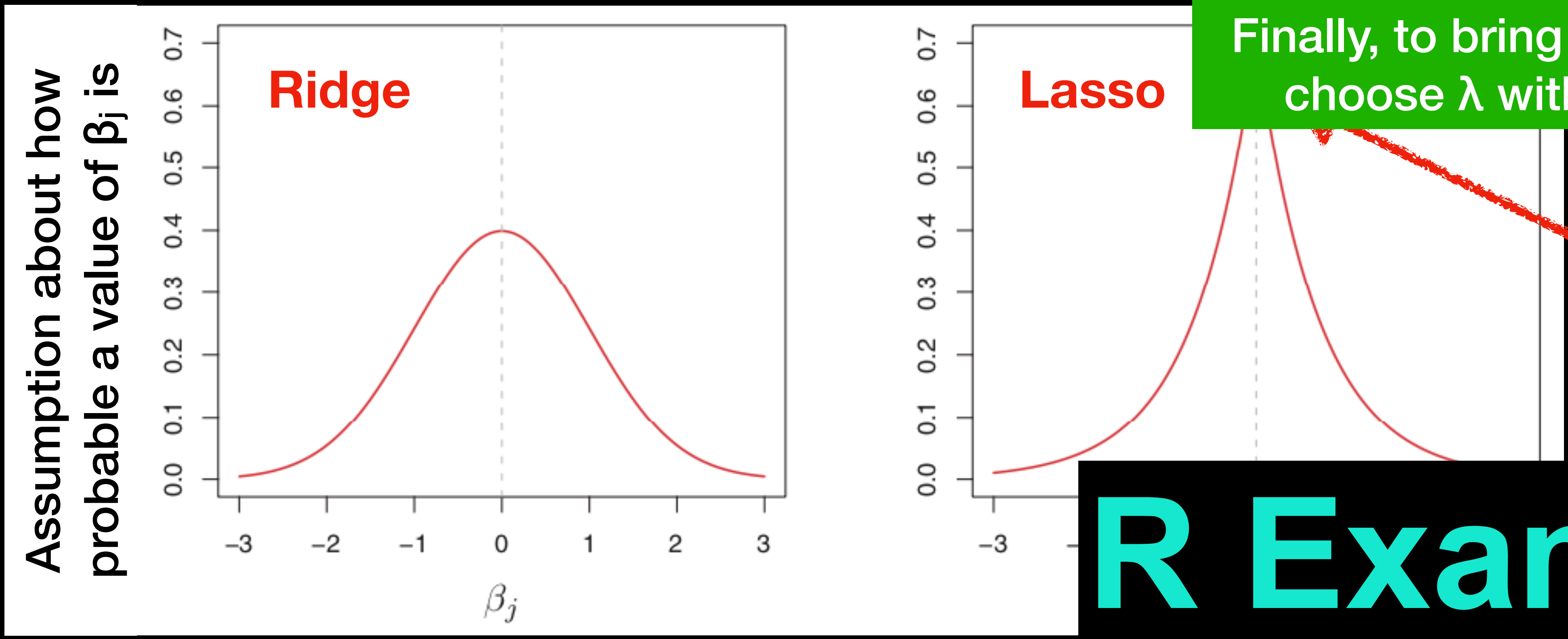
$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \text{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|$$

Lasso

Why would we make our lives more complicated like this?

(1) We can decrease the error of our fit by making it more flexible - i.e. increasing the number of parameters, in this case adding $\boxed{\lambda}$

(2) Natural way to perform variable selection $\rightarrow$ **"Shrinkage" of less important parameters to zero**



Finally, to bring it full circle - we can choose λ with Cross-Validation

Parameters are more likely to be zero

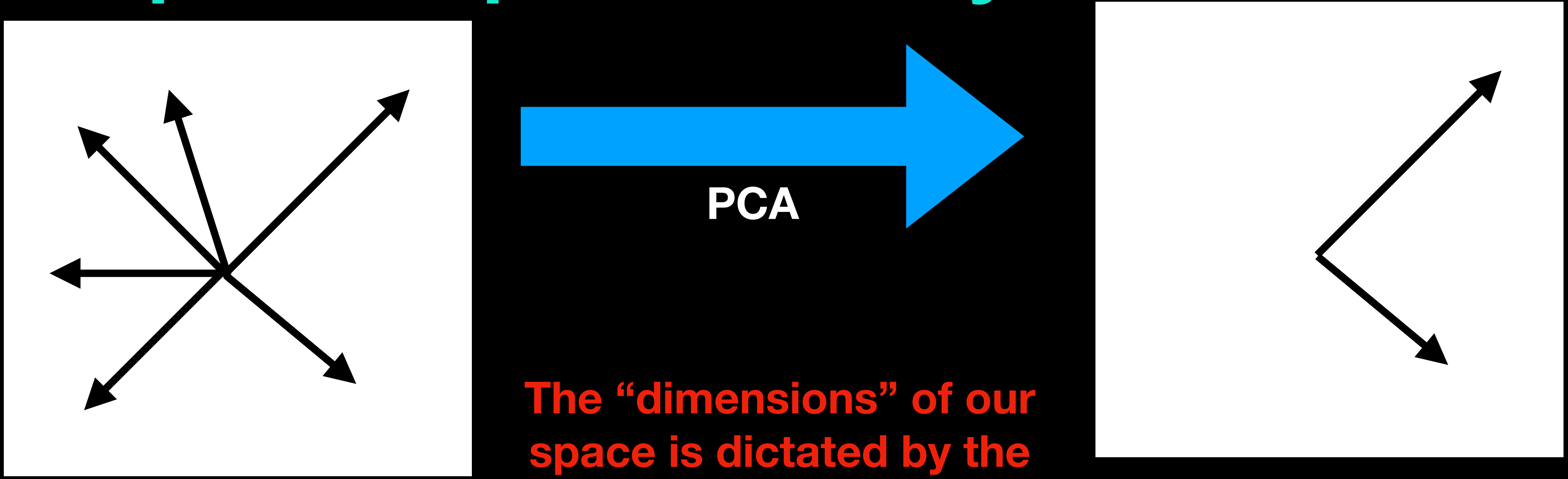
R Examples!

Unsupervised Learning:
An intro to Principle Component Analysis

Unsupervised Learning:
An intro to Principle Component Analysis

… what do we do when we don’t know anything

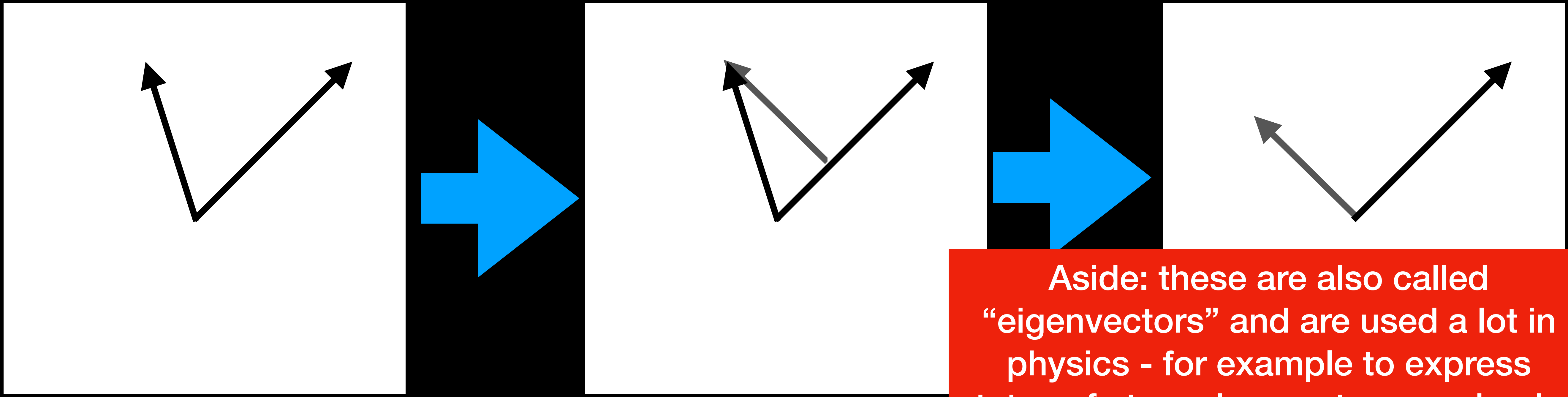
Principle Component Analysis: In Pictures



How many vectors do I need to define a 2D space?

The “dimensions” of our space is dictated by the number of parameters we have

Minimum number of vectors to define a space in a certain number of dimensions



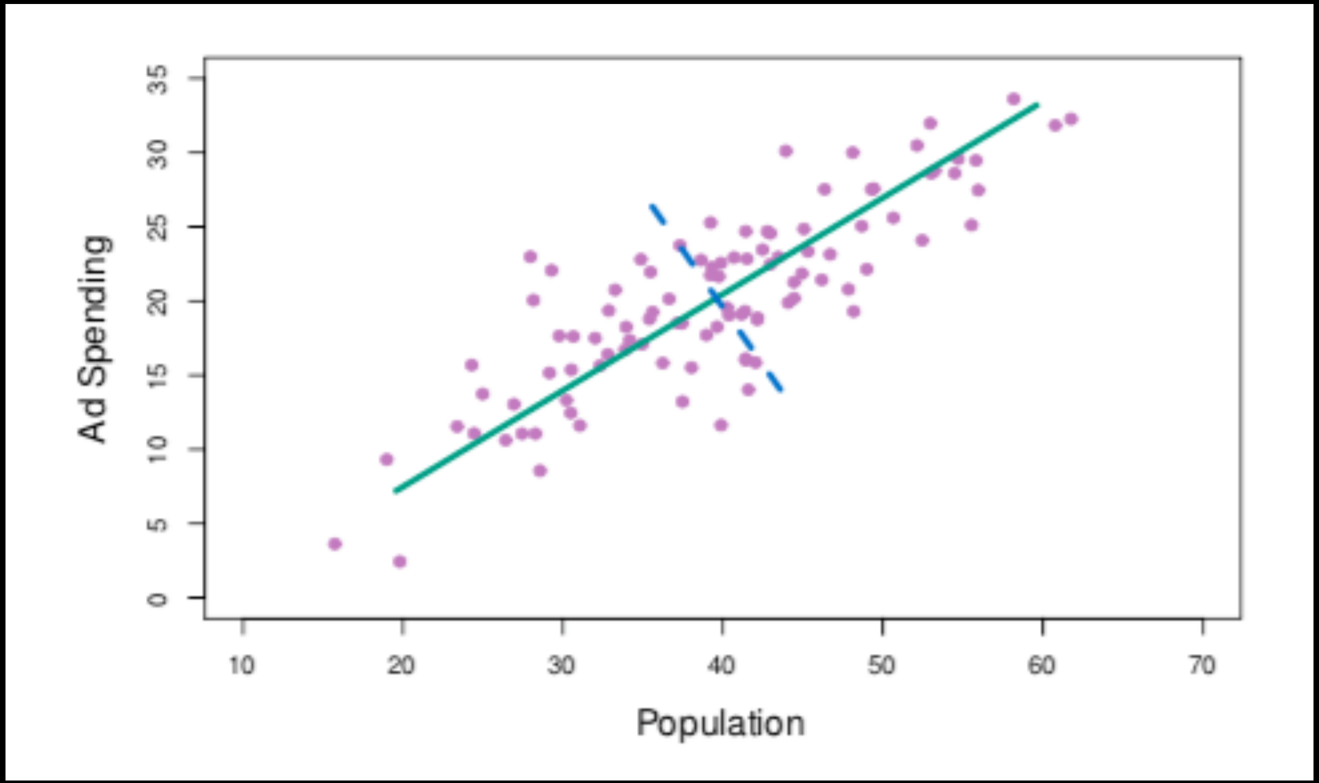
Aside: these are also called “eigenvectors” and are used a lot in physics - for example to express states of atoms in quantum mechanics

Constructing orthogonal vectors

Principle Component Analysis

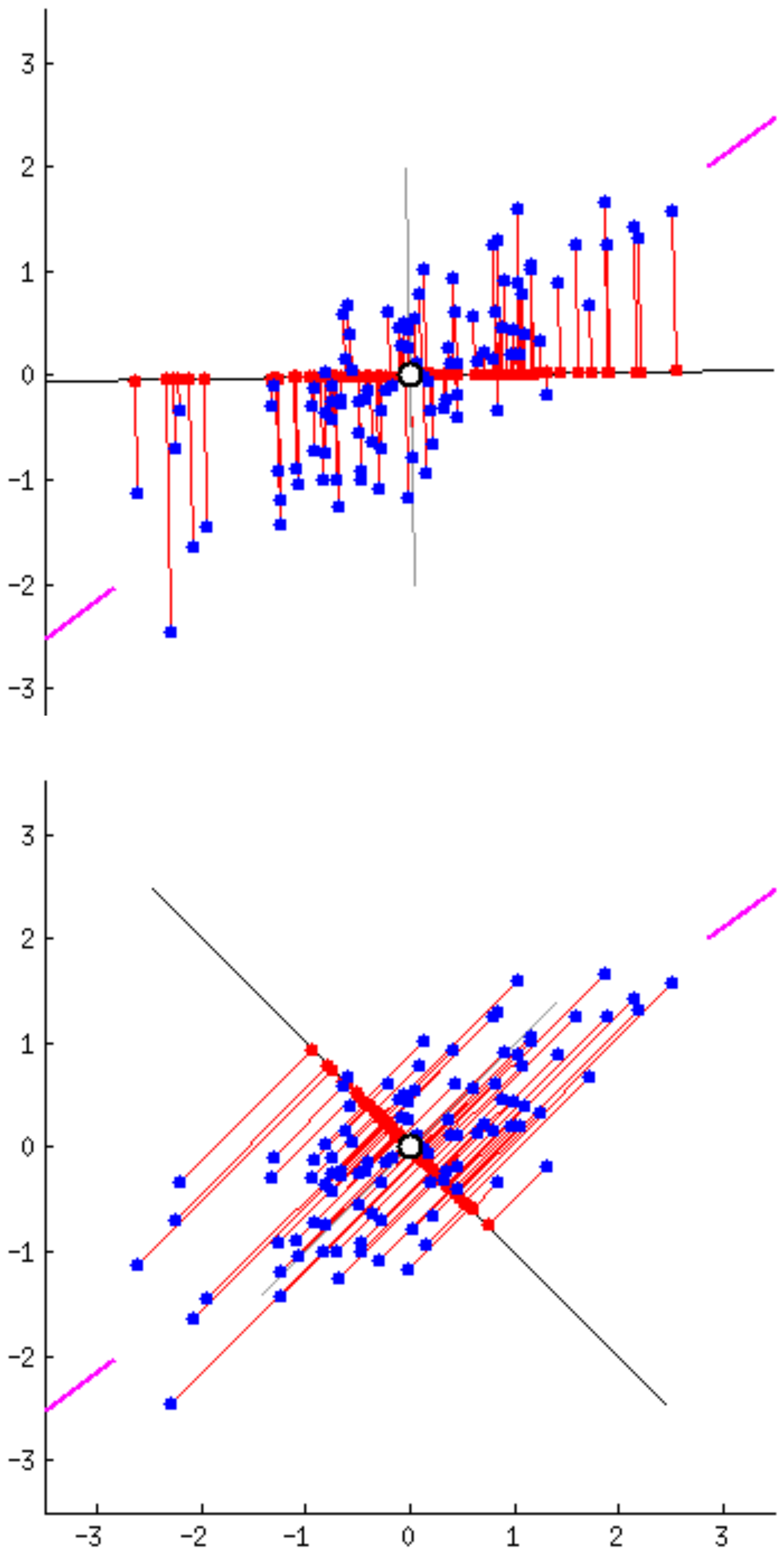
Can also think about PCA in terms of variance:

- The first principal component is that (normalized) linear combination of the variables with the largest variance.
- The second principal component has largest variance, subject to being uncorrelated with the first.
- And so on.
- Hence with many correlated original variables, we replace them with a small set of principal components that capture their joint variation.



We've essentially already fitted the first PC by fitting lines!

Quick R example!



3
2
1
0
-1
-2
-3
3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3