# Machine Learning for Business Analytics

Lecture 04

# Recap of lecture 03

- Model selection
  - Forward selection
  - Backward selection

- Regularization (aka shrinkage)
  - Ridge regression $\mathrm{MSE} + \lambda \sum_p \beta_p^2$
  - Lasso regression $\mathrm{MSE} + \lambda \sum_p |\beta_p|$

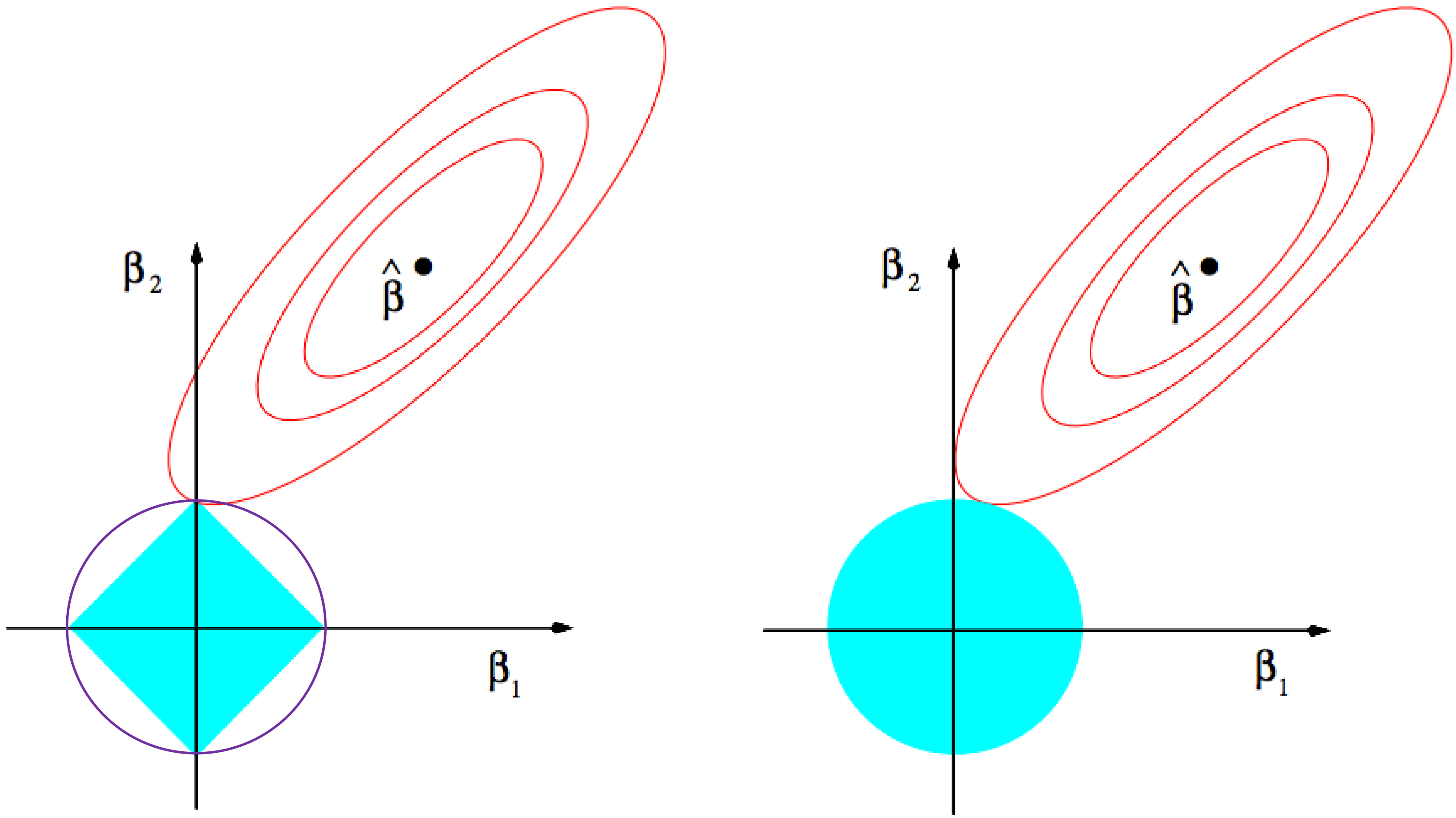



**FIGURE 3.11.** *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \leq t$ *and* $\beta_1^2 + \beta_2^2 \leq t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

(From Elements of Statistical Learning)

# Today, lecture 04

- Cross-validation
  - A way to estimate test (generalization) error, ie, a way to estimate model performance

- So far we accomplished this by setting aside a part of our training set… aka the holdout method

# Evaluating model performance

Three main reasons why we evaluate model performance:

1. Estimate how well we will do in data we have not seen yet (generalization performance)

2. Tweak the learning algorithm to improve its performance (e.g., by tweaking $\lambda$)

3. Compare different algorithms to select the best performing one among them

# Hyperparameters

- Many machine learning models have tunable parameters for which in we need to select a value

- For example, $\lambda$ in ridge and lasso

- Recall that the higher the value $\lambda$ the smaller the $\beta$'s

- Different values of $\lambda$ will yield different predictions and thus different test-set performance, ie, different test MSEs

- How should we select a good $\lambda$?

# Basic attempt at evaluating test MSE (don’t do this!)

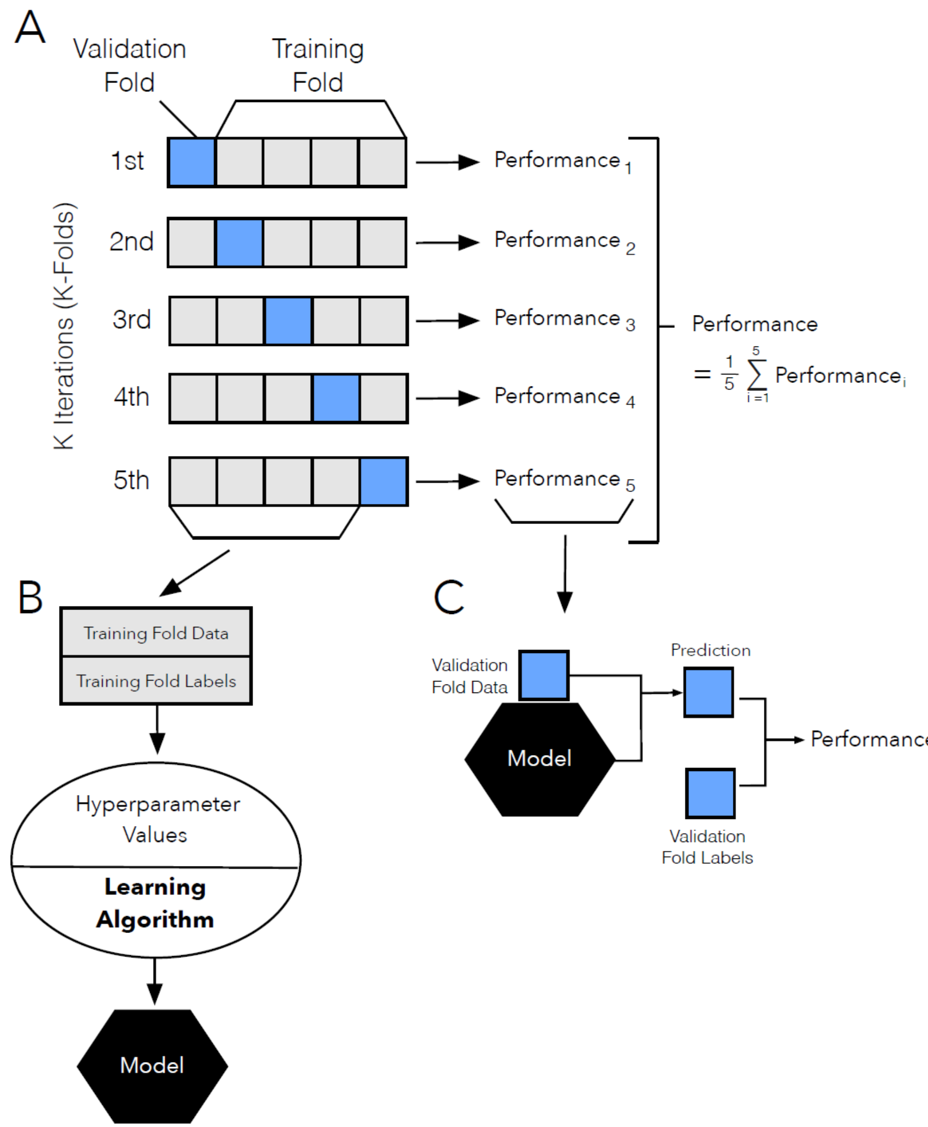
- Set aside hold out set
- Try different values of $\lambda$
- Select the value of $\lambda$ that minimizes MSE on the test set
- What is the problem with this approach?

# Improved attempt

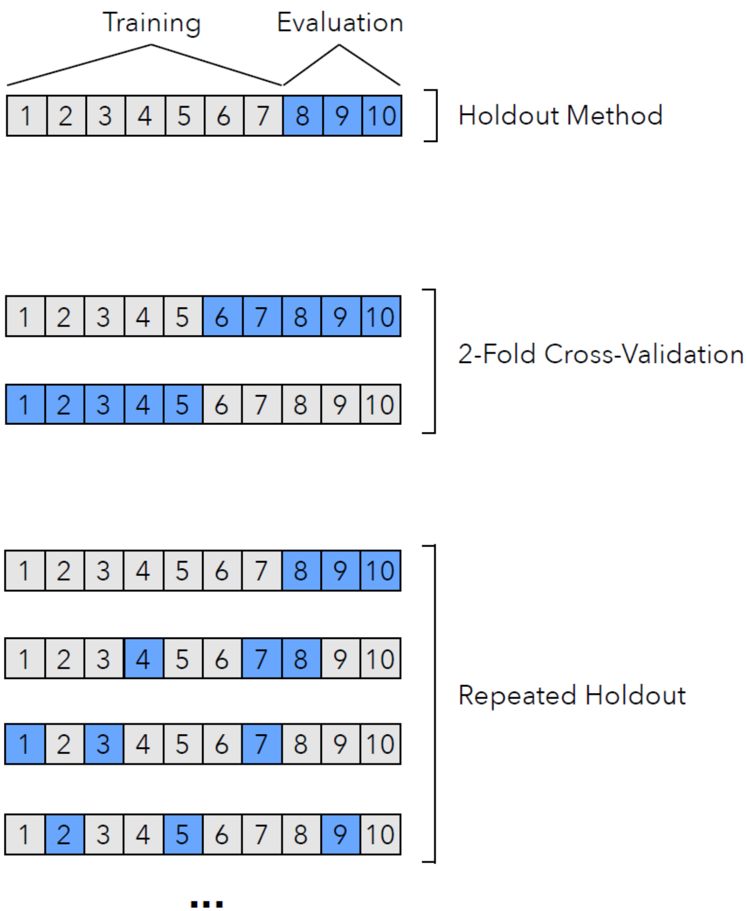
- Split the data into train and test
- Then, split training set in two parts:
  - A training set
  - A validation set
- Fit different models with different values of $\lambda$ using training set, and use validation set to choose the best performing $\lambda$
- Finally evaluate the performance of the model using the best $\lambda$ on the test set
- This is something you can do
- Disadvantage: we have to “throw away” even more data to create both a test and a validation set – higher variance. Can we do better?

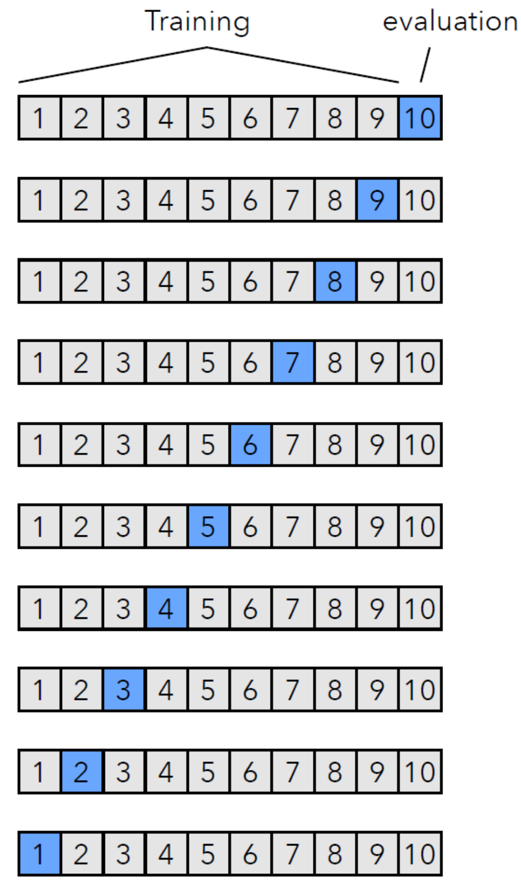
# K-fold cross-validation

- Instead of setting aside one validation set, set aside multiple validation sets
- Now, we use all of our data for training
- Might be pessimistic for small *k* (why?)



# K-fold cross-validation variants



# Bias-variance trade-off for CV

- The holdout method has the high bias. Why?
  - Consider what happens when you split your data in half and train on the first half, estimate MSE of the second half

- LOOCV has very high variance
  - First, it is easy to see that bias is low in this case since we are only dropping one observation for each model
  - However, variance will be high; consider any two leave-one-out models; do they use similar data? Will they yield similar estimates?
  - Recall, the variance of correlated variables increases in the corr. coef.

$$\mathrm{Var}(\overline{X}) = \frac{\sigma^2}{n} + \frac{n-1}{n}\rho\sigma^2.$$

- K-fold CV: empirically provides a good trade-off between bias and variance

# How do we select the right *k*?

- Largely an empirical question – depends on the application
- Having said that…
- Larger *k* (ie, more folds) will provide a more unbiased estimate of the test MSE
- …but there is no free lunch: as *k* increase so does the variance of our estimate of the MSE
- Practical concern: the larger the *k*, the greater the computational cost
- Rule of thumb: 5- or 10-fold CV tend to work well

# END

- End