Welcome to General Assembly

- **WiFi** GA Guest
- **Password** yellowpencil

GENERAL ASSEMBLY

DATA SCIENCE

DAT10SYD

Week 3 - Model Evaluation

Course Plan

UNITS

UNIT 1: FOUNDATIONS OF DATA MODELING

UNIT 2: DATA SCIENCE IN THE REAL WORLD

Paul & James review final project ideas

Git & GitHub – 1 Pager Guide!

(Part B) EVERY CLASS:

At the START of the class, you'll need to sync the latest materials from the COURSE repo:

(1) Make sure you are in the dat11syd directory:
```
cd ~/workspace/dat11syd
```
(2) Make sure to select the "master" branch of your repo:
```
git checkout master
```
(3) Fetch the latest changes from the UPSTREAM repo (i.e the course repo)
```
git fetch upstream
```
(4) Merge the changes from the upstream repo to your master branch:
```
git merge upstream/master
```

DURING the class:

(5) Before editing, either copy files to your "students/" folder, or rename them

At the END of every class:

(6) Make sure you are in the dat11syd directory:
```
cd ~/workspace/dat11syd
```
(7) Add any files that you've updated to your git registry:
```
git add -A
```
(8) Commit the changes with a sensible comment:
```
git commit -m "my updates for lesson 7"
```
(9) Push your changes to your PERSONAL repo:
```
git push origin master
```

DONE!!!!!

1. Recap from last time
2. Evaluating machine learning models
3. Why is this important?
4. Correctly assessing the accuracy of a model
5. Lab
6. Review

DATA SCIENCE PART TIME COURSE

RECAP:

Last Lesson

Accuracy of a Classification Model

		Actual		"Precision"
		Condition positive	Condition negative	
Prediction	Test outcome positive	**True positive** (TP) = 20	**False positive** (FP) = 180	Positive predictive value = TP / (TP + FP) = 20 / (20 + 180) = **10%**
	Test outcome negative	**False negative** (FN) = 10	**True negative** (TN) = 1820	Negative predictive value = TN / (FN + TN) = 1820 / (10 + 1820) ≈ **99.5%**
	"Recall"	**Sensitivity** = TP / (TP + FN) = 20 / (20 + 10) ≈ **67%**	**Specificity** = TN / (FP + TN) = 1820 / (180 + 1820) = **91%**	
		"True positive rate"	"True negative rate"	

True Positive Rate

$$TPR = \frac{TP}{Actual\ P}$$

False Positive Rate

$$FPR = \frac{FP}{Actual\ N}$$

True Negative Rate

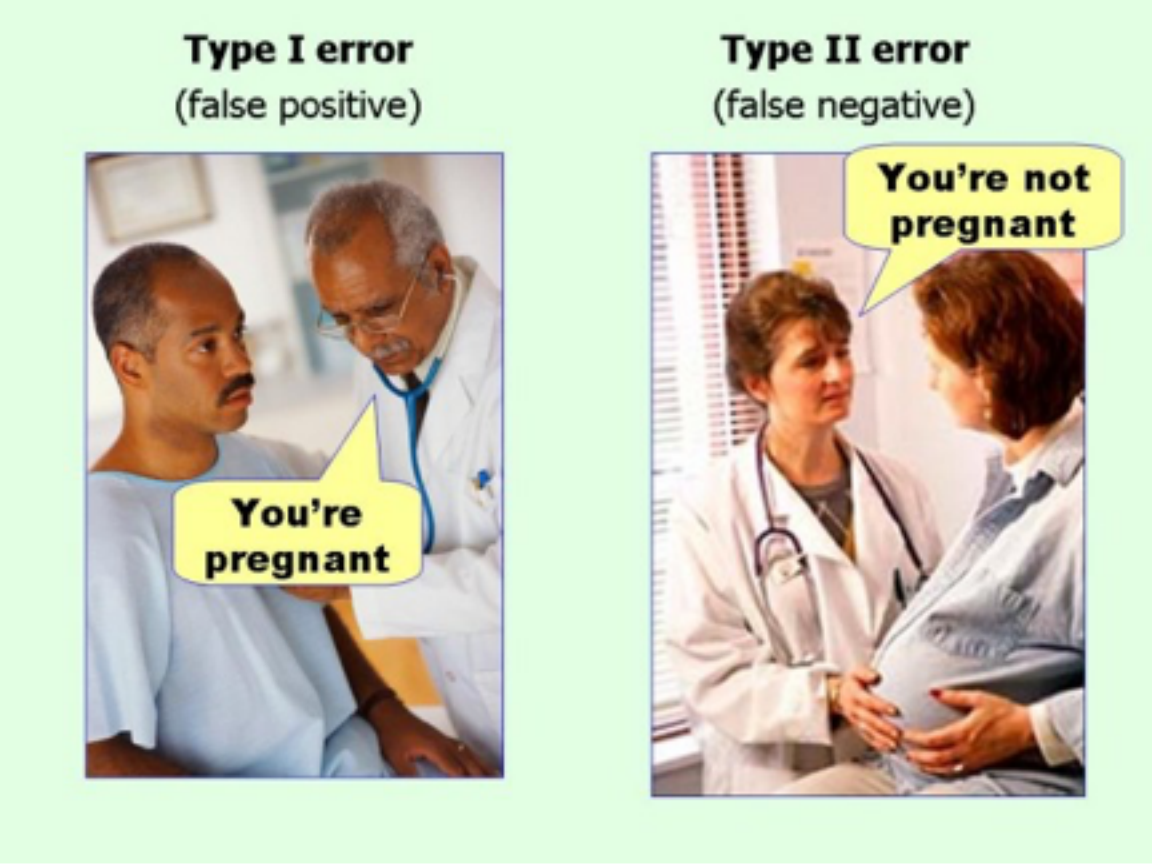
$$TNR = \frac{TN}{Actual\ N}$$

False Negative Rate

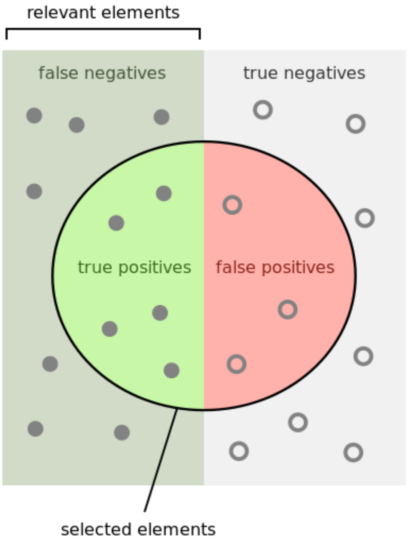
$$FNR = \frac{FN}{Actual\ P}$$

$$\boxed{Accuracy = \frac{TP + TN}{Total\ Population}}$$

$$Accuracy = \frac{20 + 1820}{20 + 1820 + 180 + 10} = \frac{1840}{2030} = 0.91 = 91\%$$


Type I error
(false positive)
You're pregnant
Type II error
(false negative)
You're not pregnant

PRECISION & RECALL

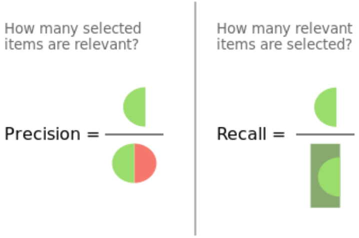


Precision :

of those we guessed were positive, how often were we right?

Recall = **Sensitivity** :

how many of actual positives did we capture?



$$F1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

F1 measure :

balance of Precision and Recall

DATA SCIENCE PART TIME COURSE

THE POINT OF EVALUATING MODELS

Why do we need to evaluate models?

Why might we need to be rigorous in evaluating models?

DATA SCIENCE PART TIME COURSE

ESSENTIALS OF MODEL EVALUATION

Q: What's wrong with training error?

Q: What's wrong with training error?

A: Training error is not a good estimate of accuracy beyond training data.

Q: How low can we push the training error if we can make the model arbitrarily complex. Effectively "memorizing" the entire training set ?

Q: How low can we push the training error if we can make the model arbitrarily complex. Effectively "memorizing" the entire training set ?

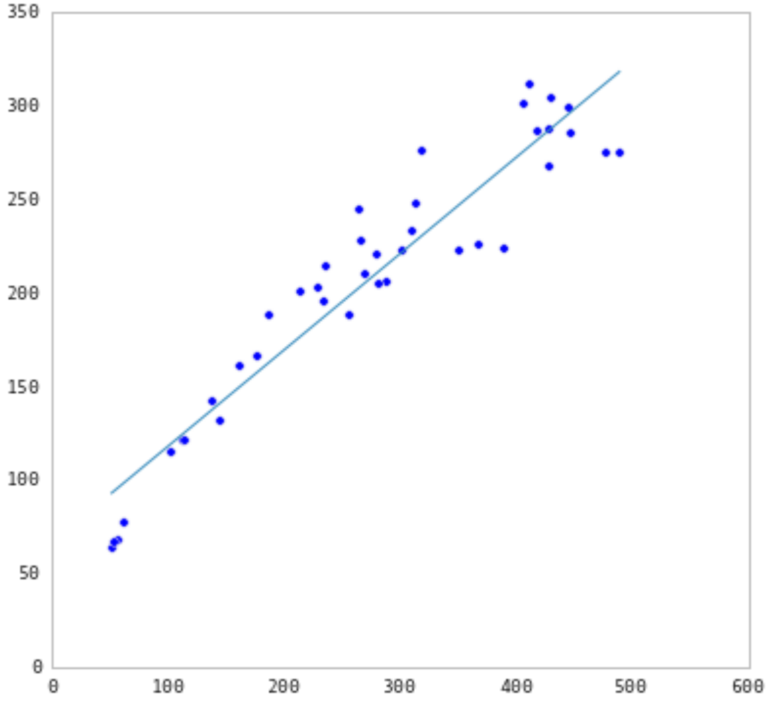
A: Down to zero!

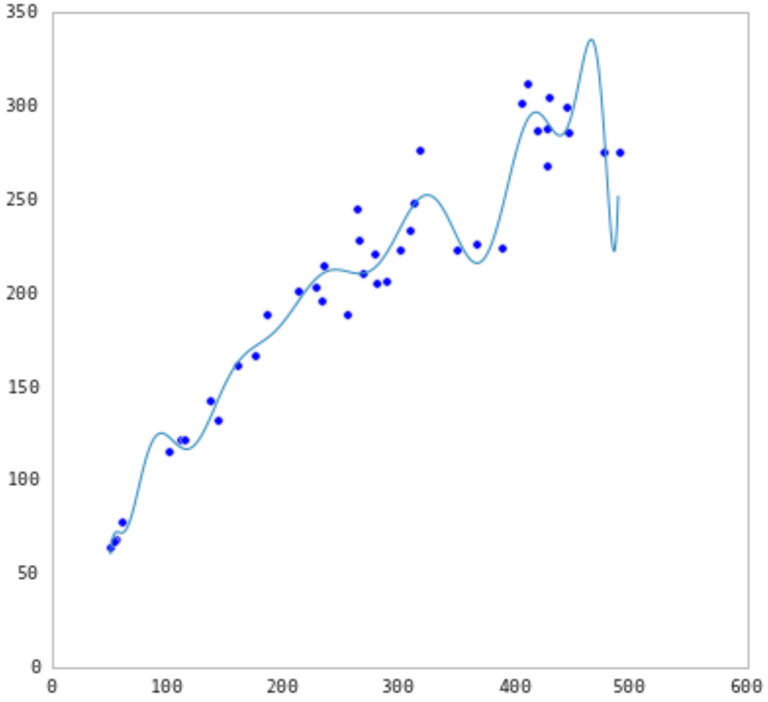
WHY THIS MATTERS

The data that we are given for prediction won't always be the end of the data we are interested in! We may not have access to all the data of interest

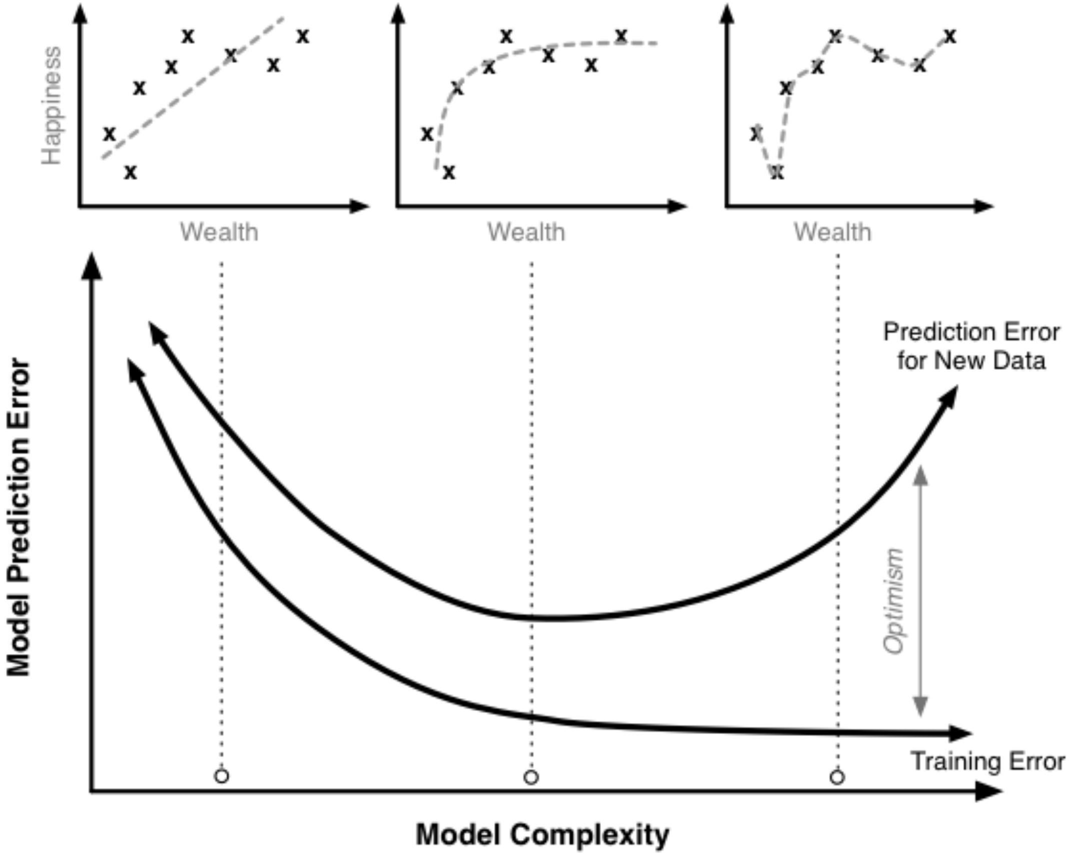
We will gather data and build and iterate over models however a main reason for building the model was to predict unseen test cases.

TRAINING ERROR



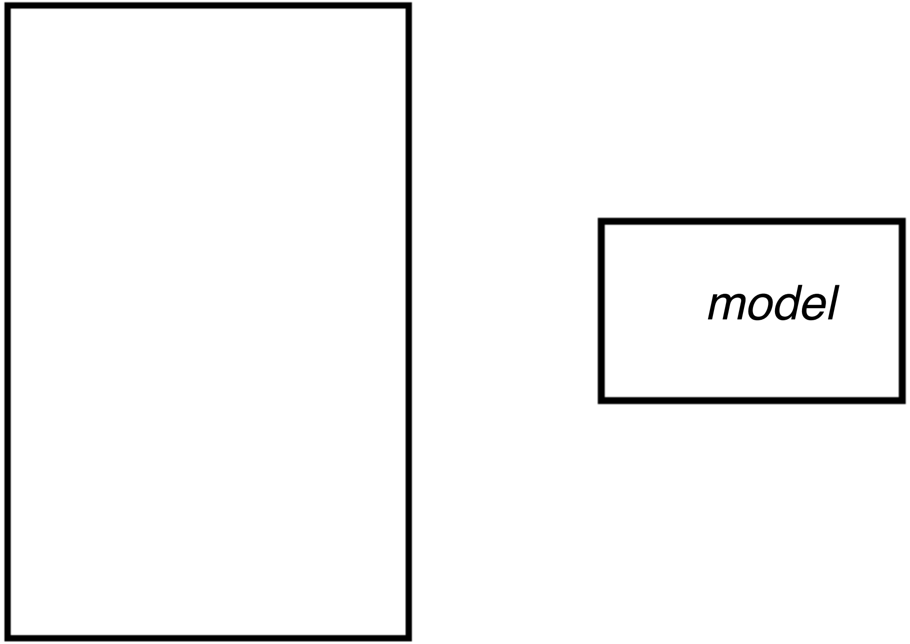


TEST vs TRAIN



TEST SET APPROACH

Q: How can we make a model that generalises well?

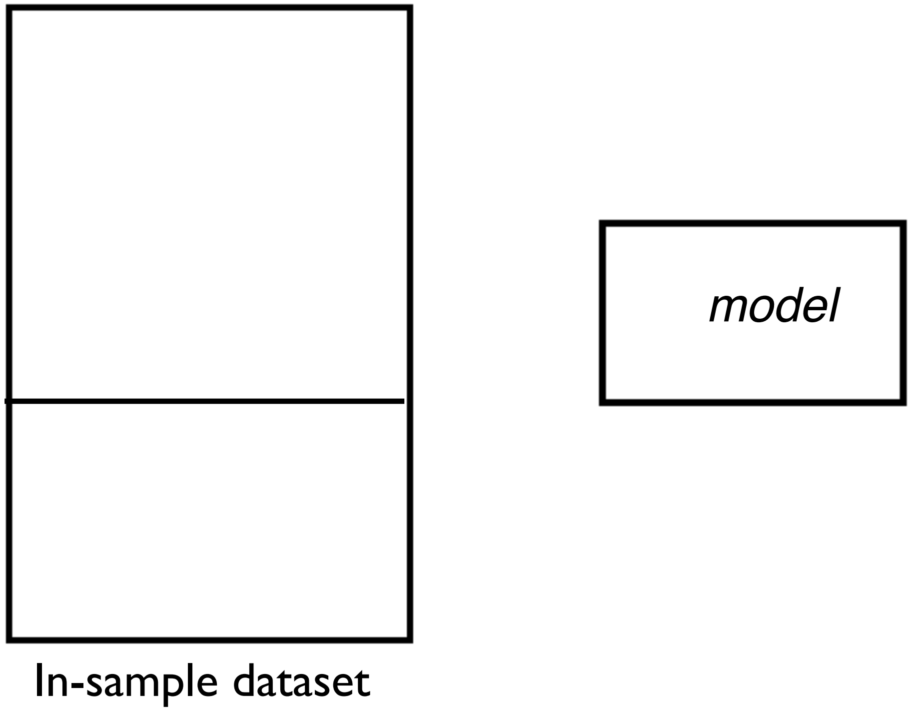


In-sample dataset

TEST SET APPROACH

Q: How can we make a model that generalises well?

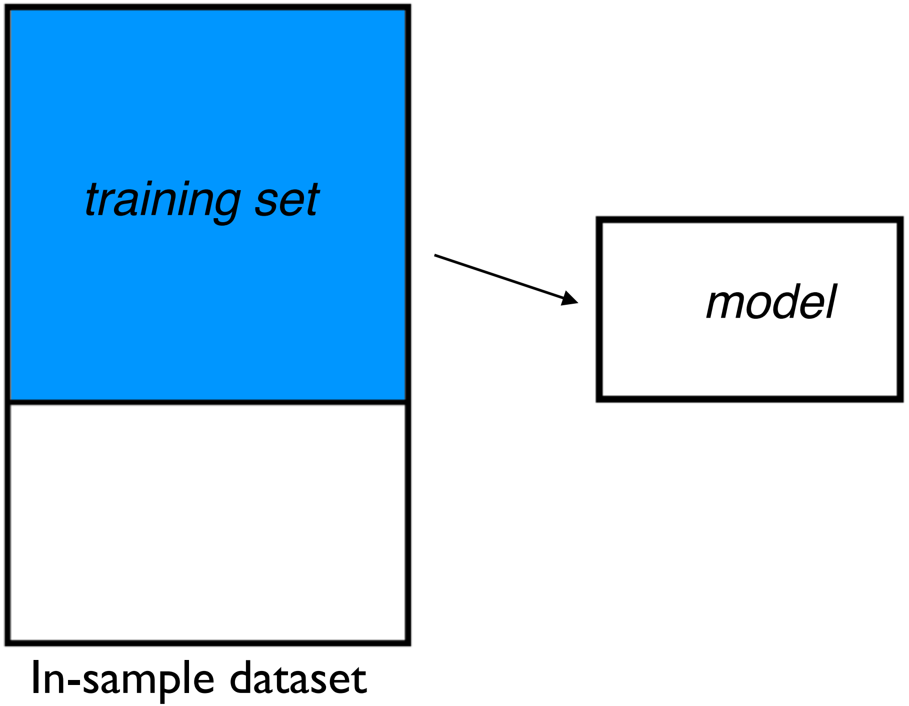
1) *split dataset*



TEST SET APPROACH

Q: How can we make a model that generalizes well?

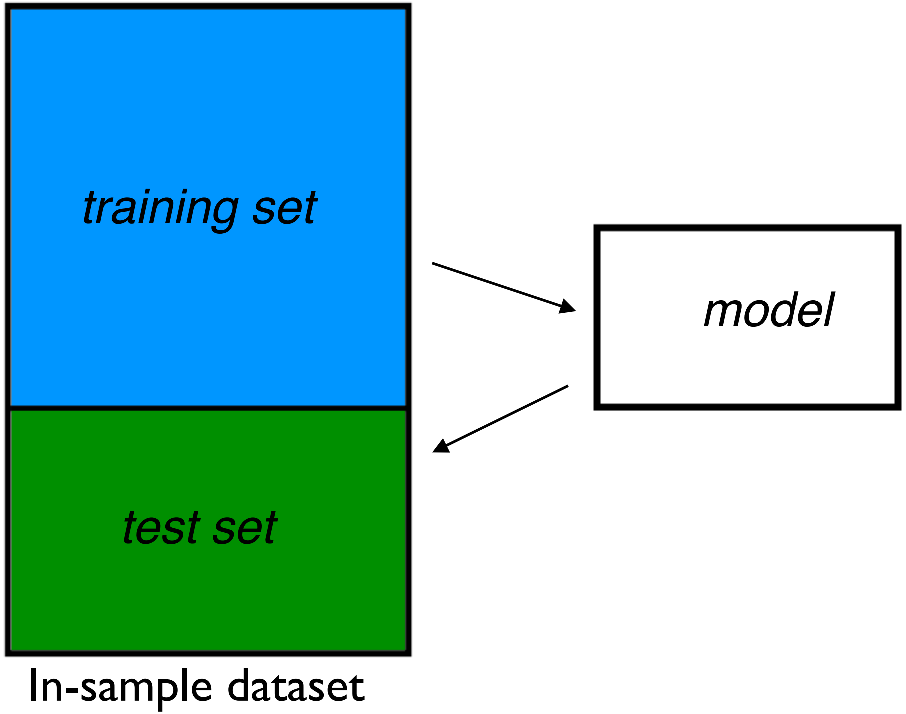
1) *split dataset*
2) *train model*



TEST SET APPROACH

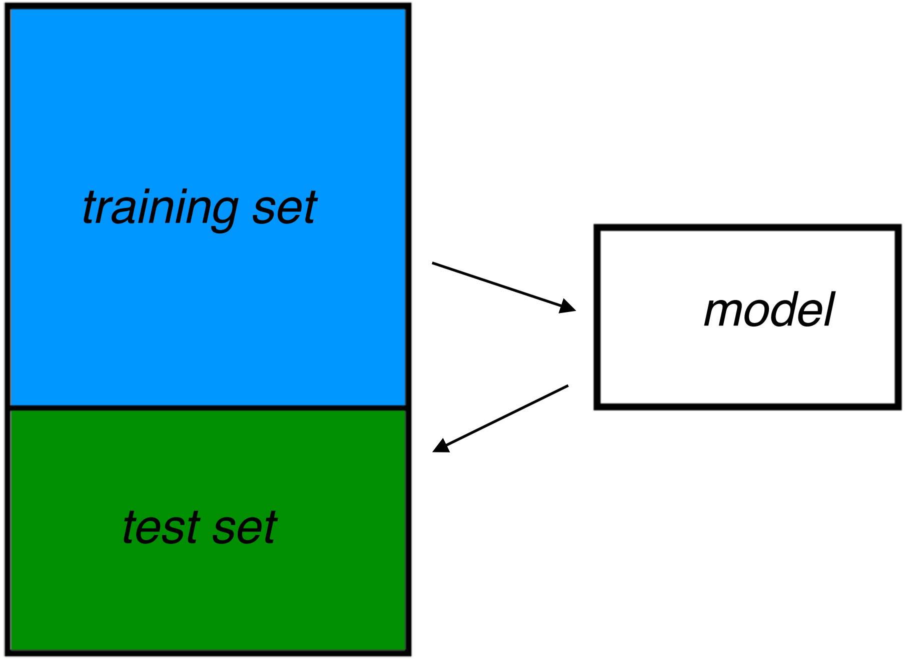
Q: How can we make a model that generalizes well?

1) *split dataset*
2) *train model*
3) *test model*



Q: How can we make a model that generalizes well?

1) *split dataset*
2) *train model*
3) *test model*
4) *parameter tuning*

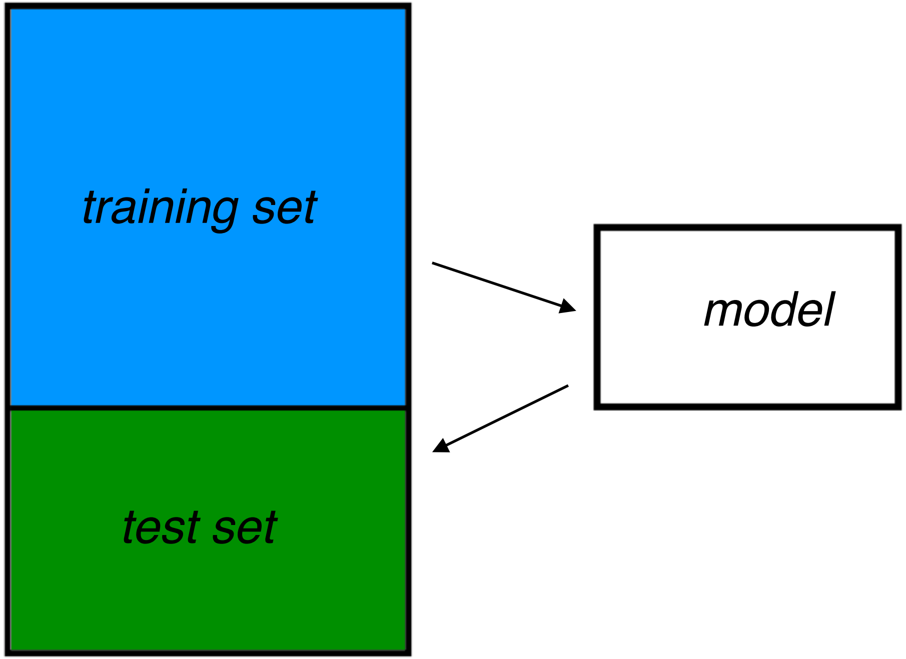


In-sample dataset

TEST SET APPROACH

Q: How can we make a model that generalizes well?

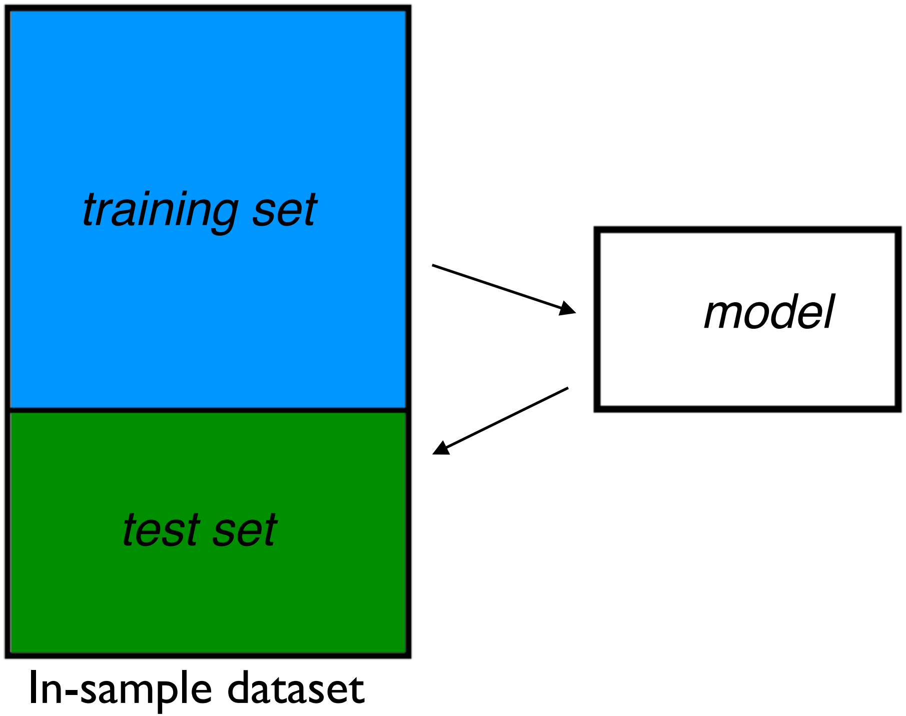
1) *split dataset*
2) *train model*
3) *test model*
4) *parameter tuning*
5) *choose best model*



In-sample dataset

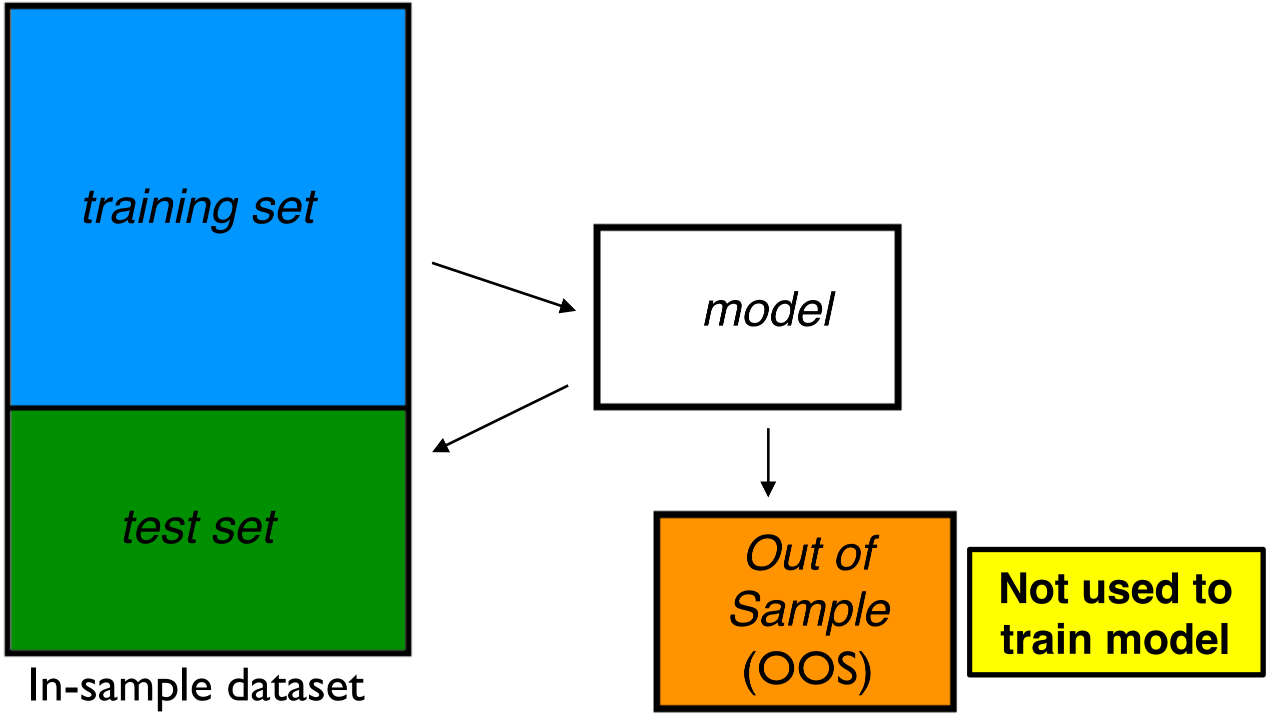
Q: How can we make a model that generalizes well?

1) *split dataset*
2) *train model*
3) *test model*
4) *parameter tuning*
5) *choose best model*
6) *train on **all** data*



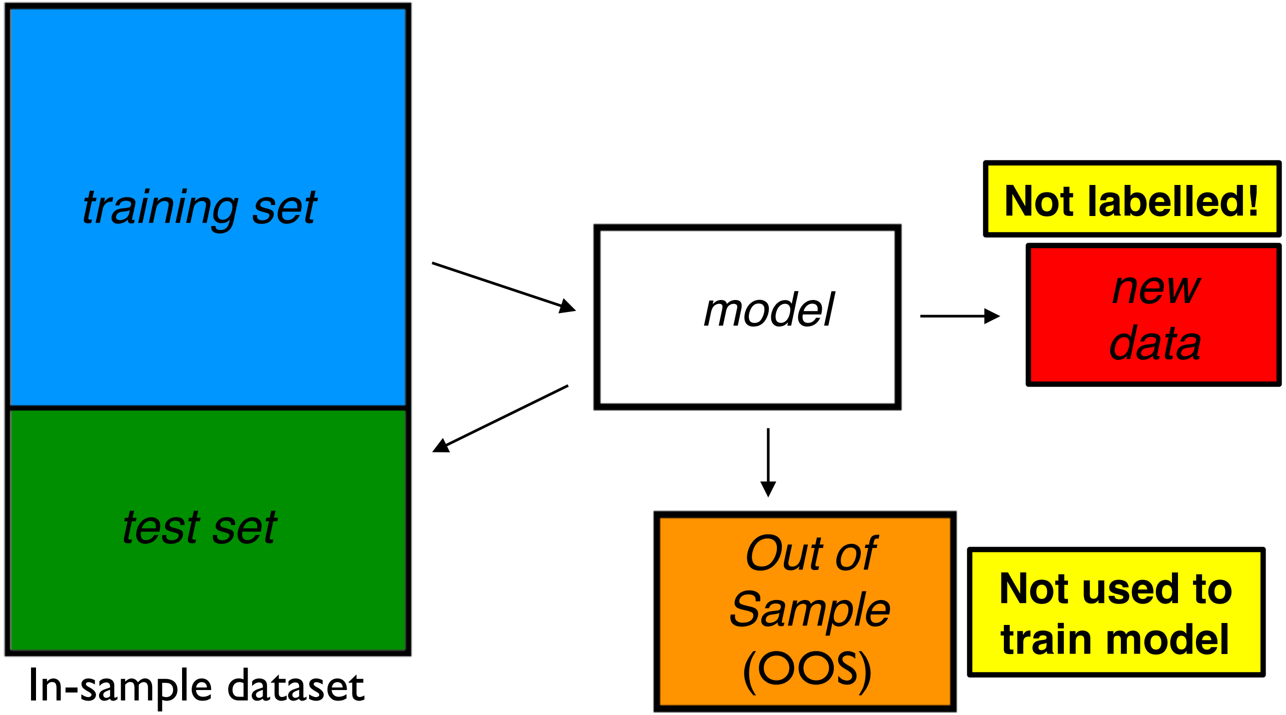
Q: How can we make a model that generalizes well?

1) *split dataset*
2) *train model*
3) *test model*
4) *parameter tuning*
5) *choose best model*
6) *train on __**all**__ data*
7) *<u>test</u> predictions on OOS data*



Q: How can we make a model that generalizes well?

1) *split dataset*
2) *train model*
3) *test model*
4) *parameter tuning*
5) *choose best model*
6) *train on **all** data*
7) *test predictions on OOS data*
8) *Apply model: create labels for new data*



LAB1

Git & GitHub – 1 Pager Guide!

(Part B) EVERY CLASS:

At the START of the class, you'll need to sync the latest materials from the COURSE repo:

(1) Make sure you are in the dat11syd directory:
```
cd ~/workspace/dat11syd
```
(2) Make sure to select the "master" branch of your repo:
```
git checkout master
```
(3) Fetch the latest changes from the UPSTREAM repo (i.e the course repo)
```
git fetch upstream
```
(4) Merge the changes from the upstream repo to your master branch:
```
git merge upstream/master
```

DURING the class:

(5) Before editing, either copy files to your "students/" folder, or rename them

At the END of every class:

(6) Make sure you are in the dat11syd directory:
```
cd ~/workspace/dat11syd
```
(7) Add any files that you've updated to your git registry:
```
git add -A
```
(8) Commit the changes with a sensible comment:
```
git commit -m "my updates for lesson 7"
```
(9) Push your changes to your PERSONAL repo:
```
git push origin master
```

DONE!!!!!

Suppose we do the train/test split.

Q: How well does test set error predict Out of Sample Error?

Suppose we do the train/test split.

Q: How well does test set error predict Out of Sample Error?

A: On its own, not very well.

Suppose we do the train/test split.

Q: How well does test set error predict Out of Sample Error?

A: On its own, not very well.

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the test set error remain the same?

Suppose we do the train/test split.

Q: How well does test set error predict Out of Sample Error?

A: On its own, not very well.

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the test set error remain the same?

A: Of course not!

NOTE

The test set error gives a *high-variance estimate* of OOS accuracy.

Something is still missing!

Thought experiment:

Different train/test splits will give us different test set errors.

Q: What if we did a bunch of these and took the average?

A: Now you're talking!

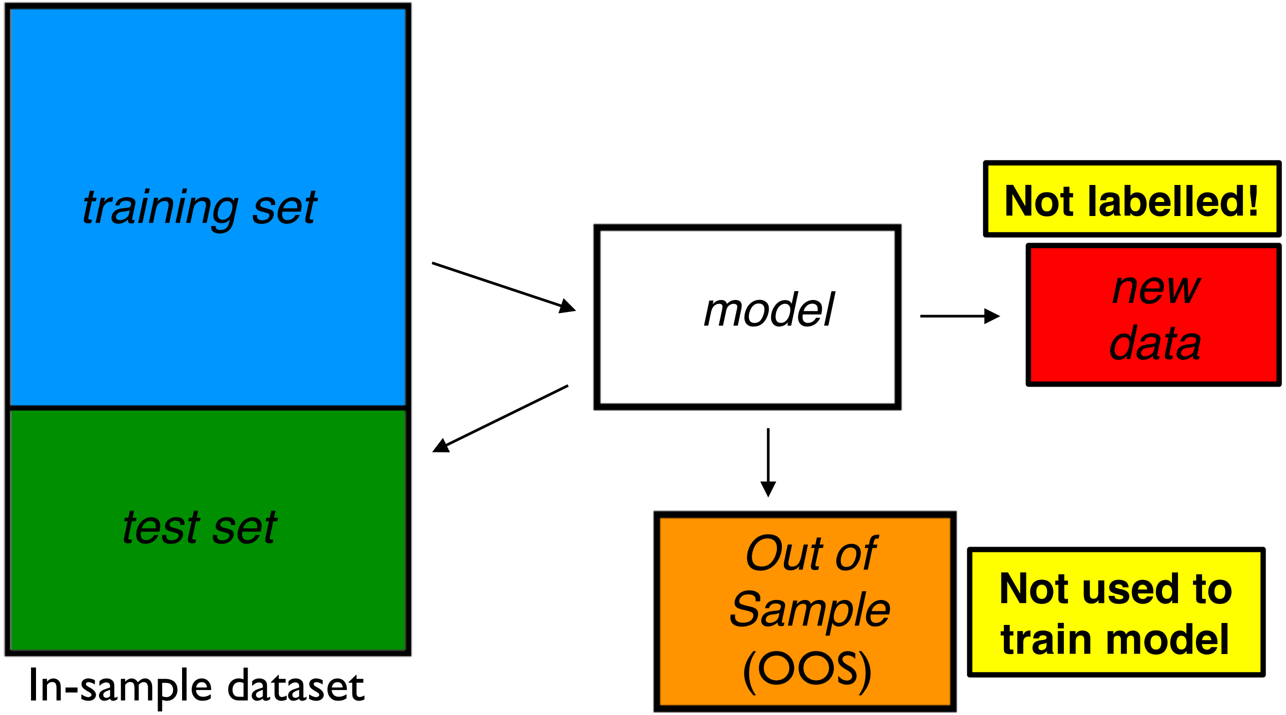
Cross-validation!

DATA SCIENCE PART TIME COURSE

CROSS VALIDATION

Q: How can we make a model that generalizes well?

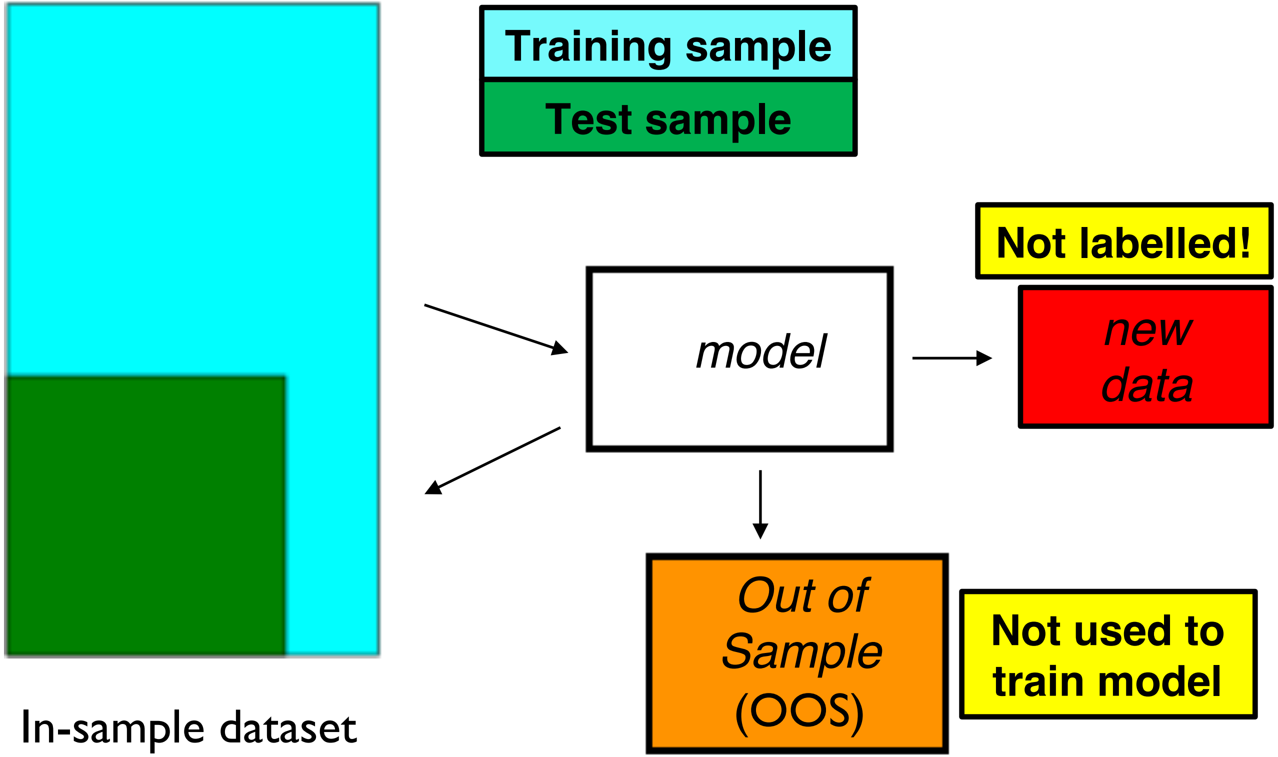
1) *split dataset*
2) *train model*
3) *test model*
4) *parameter tuning*
5) *choose best model*
6) *train on **all** data*
7) *test predictions on OOS data*
8) *Apply model: create labels for new data*



TEST SET APPROACH

Q: How can we make a model that generalizes well?

1) *split dataset*
2) *train model*
3) *test model*
4) *parameter tuning*
5) *choose best model*
6) *train on **__all__** data*
7) *__test__ predictions on OOS data*
8) *Apply model: create labels for new data*



Steps for K-fold cross-validation:

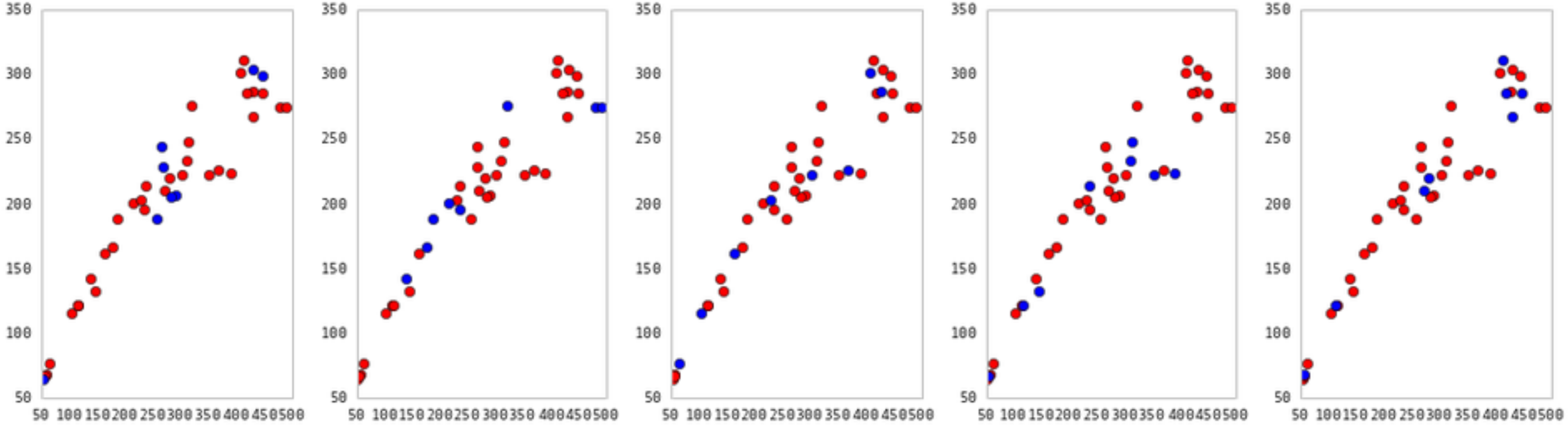
1) *Randomly split the dataset into K equal partitions.*
2) *Use partition 1 as test set & union of other partitions as training set.*
3) *Calculate test set error.*
4) *Repeat steps 2-3 using a different partition as the test set at each iteration.*
5) *Take the average test set error as the estimate of OOS accuracy.*

CROSS VALIDATION

Divide data into K roughly equal-sized parts ($K = 5$ here)

1	2	3	4	5
Validation	Train	Train	Train	Train

CROSS VALIDATION



5-fold cross-validation: red = training folds, blue = test fold

Source: http://nbviewer.ipython.org/github/fonnesbeck/Bios366/blob/master/notebooks/Section6_3-Model-Selection-and-Validation.ipynb

Features of K-fold cross-validation:

- *More accurate estimate of OOS prediction error.*
- *More efficient use of data than single train/test split.*
  - *Each record in our dataset is used for both training and testing.*
- *Presents tradeoff between efficiency an computational expense.*
  - *10-fold CV is 10x more expensive than a single train/test split*
- *Can be used for parameter tuning and model selection.*

BIAS & VARIANCE TRADEOFF

Training many models over many in-sample datasets will give different errors.

BIAS

This is how different the "averaged model" prediction is to the actual data

(High Bias = Large overall difference between best prediction and actuals)

VARIANCE

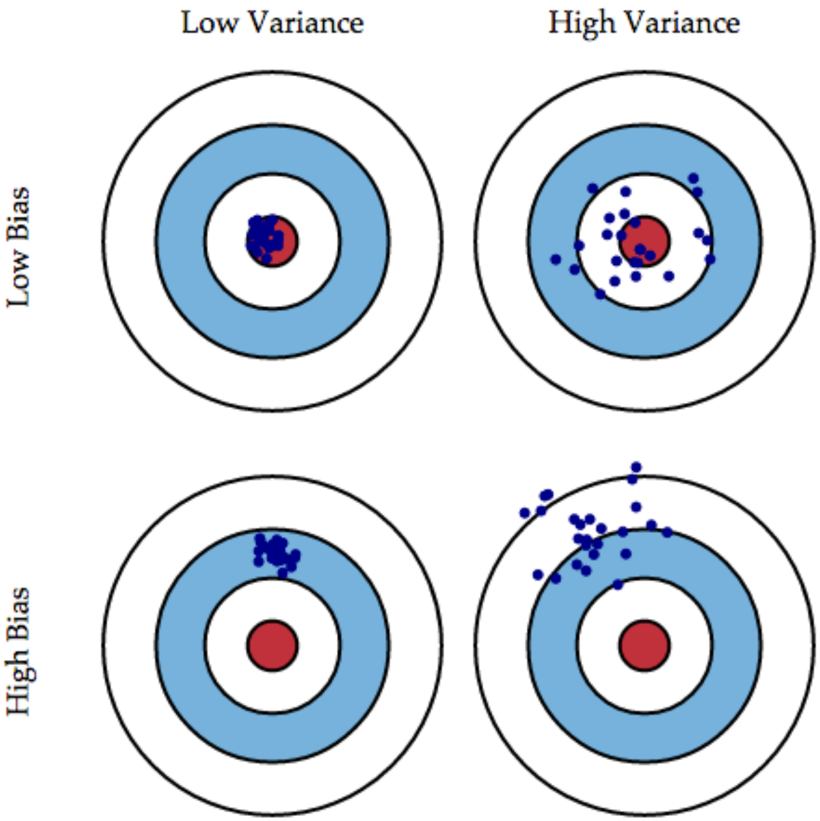
This is how variable different model predictions are for a given data point.

BIAS & VARIANCE

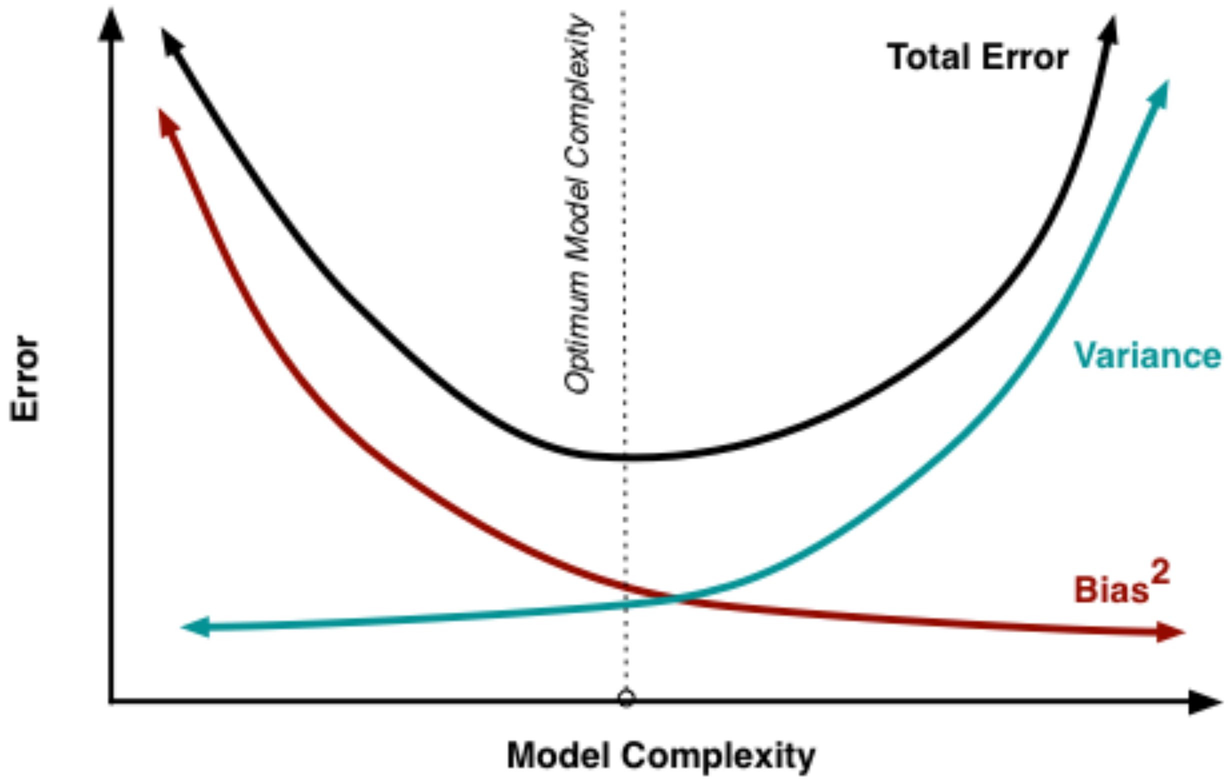
Together, these can tell us whether we are underfitting or overfitting

Imagine 25 different models (of the same type) created using 25 different samples of the data
e.g: Predicting the yield of apples from trees (using the same features in each model, but different data samples)

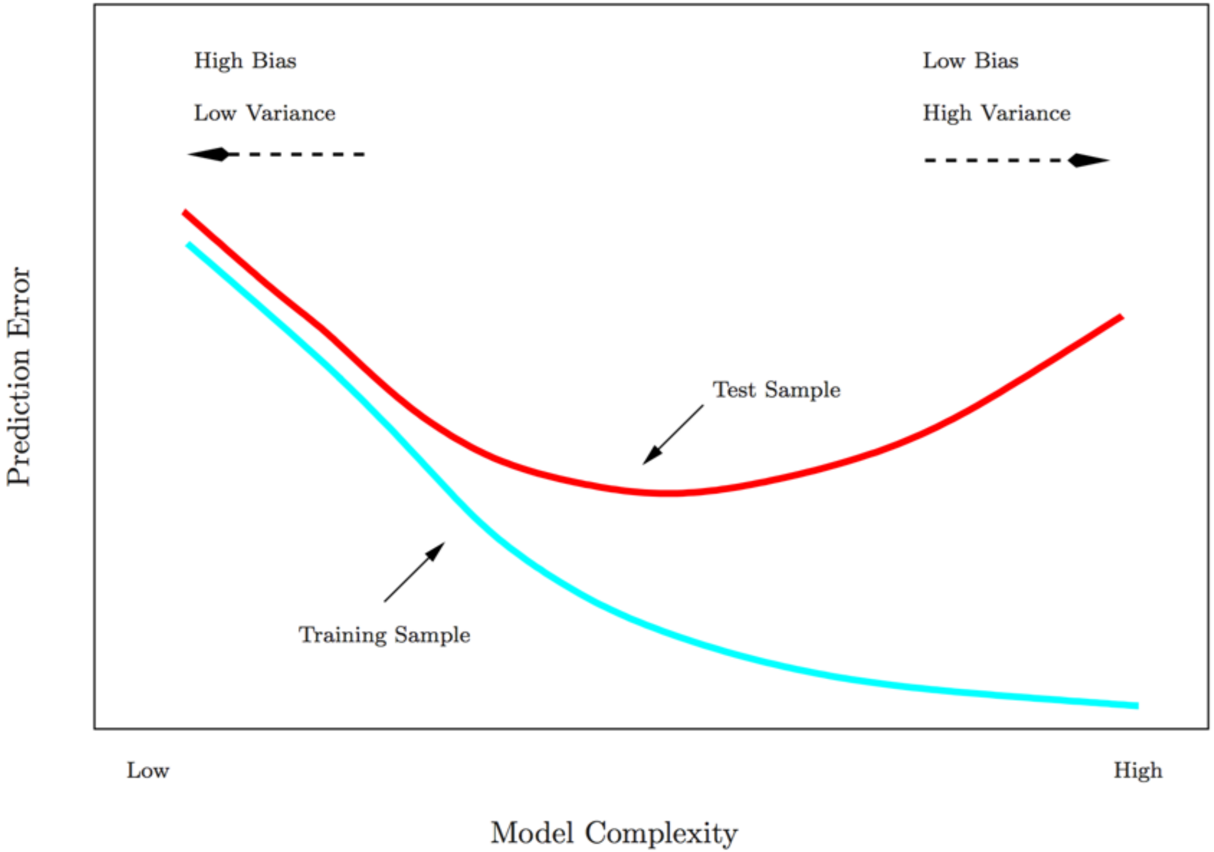
These models are then used to predict 1 value,
e.g: comparing the yield for the same tree (as predicted in the 25 models)



BIAS - VARIANCE TRADEOFF



BIAS - VARIANCE TRADEOFF



LAB - Evaluation Metrics

DISCUSSION TIME

- Questions from previous lesson?
- What are we trying to do when we use Logistic Regression?
- How would you evaluate a regression problem?

QUESTIONS

- What are we trying to do when we use Logistic Regression?
- Why use it instead of Linear Regression for classification?
- Evaluating a logistic Regression model

DATA SCIENCE

HOMEWORK

Pre-reading: An Introduction to Statistical Learning Chapter 6 - Model selection & regularisation

Caltech's Learning From Data course visualising bias and variance (15 mins)
http://work.caltech.edu/library/081.html

Rahul Patwari has a great video on ROC Curves (12 minutes)
https://www.youtube.com/watch?v=21Igj5Pr6u4

Have a look at scikit-learn's documentation on model evaluation
http://scikit-learn.org/stable/modules/model_evaluation.html