GENERAL ASSEMBLY

DATA SCIENCE

MODEL EVALUATION PROCEDURES

TRAINING ERROR VS TEST ERROR

Training error is the error over the training sample (when we train and test on the same dataset)

Test error is the error over an **independent** test sample

We use the test error to estimate the model's use but training error is not a good estimator for test error

Q: What's wrong with training error?

Thought experiment:

Suppose we train our model using the entire dataset.

Q: How low can we push the training error?

- *We can make the model arbitrarily complex (effectively "memorizing" the entire training set).*

A: Down to zero!

Q: What's wrong with training error?

Thought experiment:

Suppose we train our model using the entire dataset.

Q: How low can we push the training error?

- ***We can make the model arbitrarily complex (effectively "memorizing" the entire training set).***

A: Down to zero!

This phenomenon is called *overfitting*.

Q: What's wrong with training error?

Thought experiment:

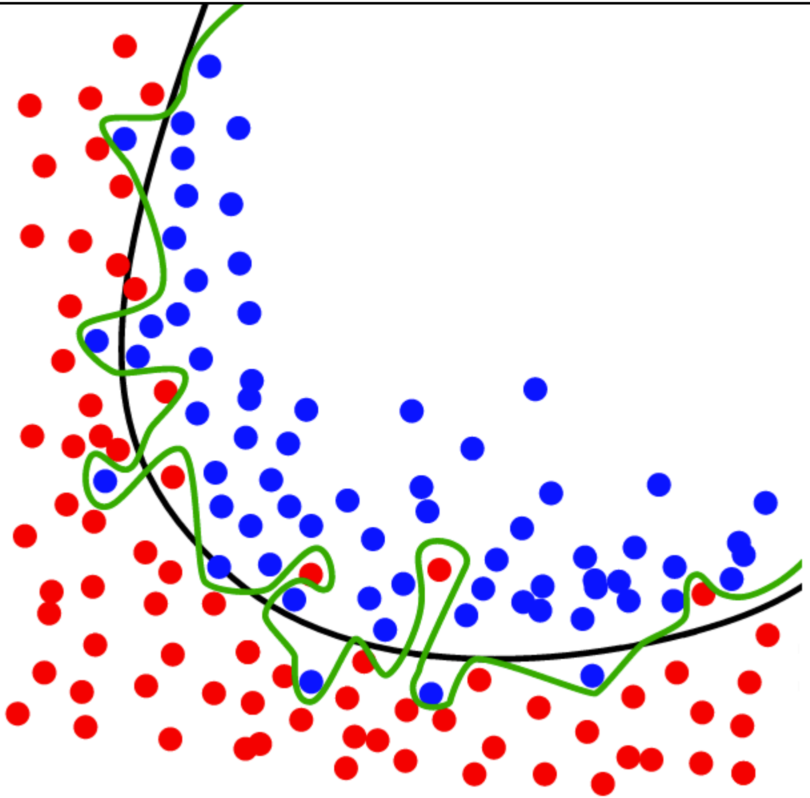
Suppose we train our model using the entire dataset.

Q: How low can we push the training error?

- *We can make the model arbitrarily complex (effectively "memorizing" the entire training set).*

A: Down to zero!

This phenomenon is called *overfitting*.

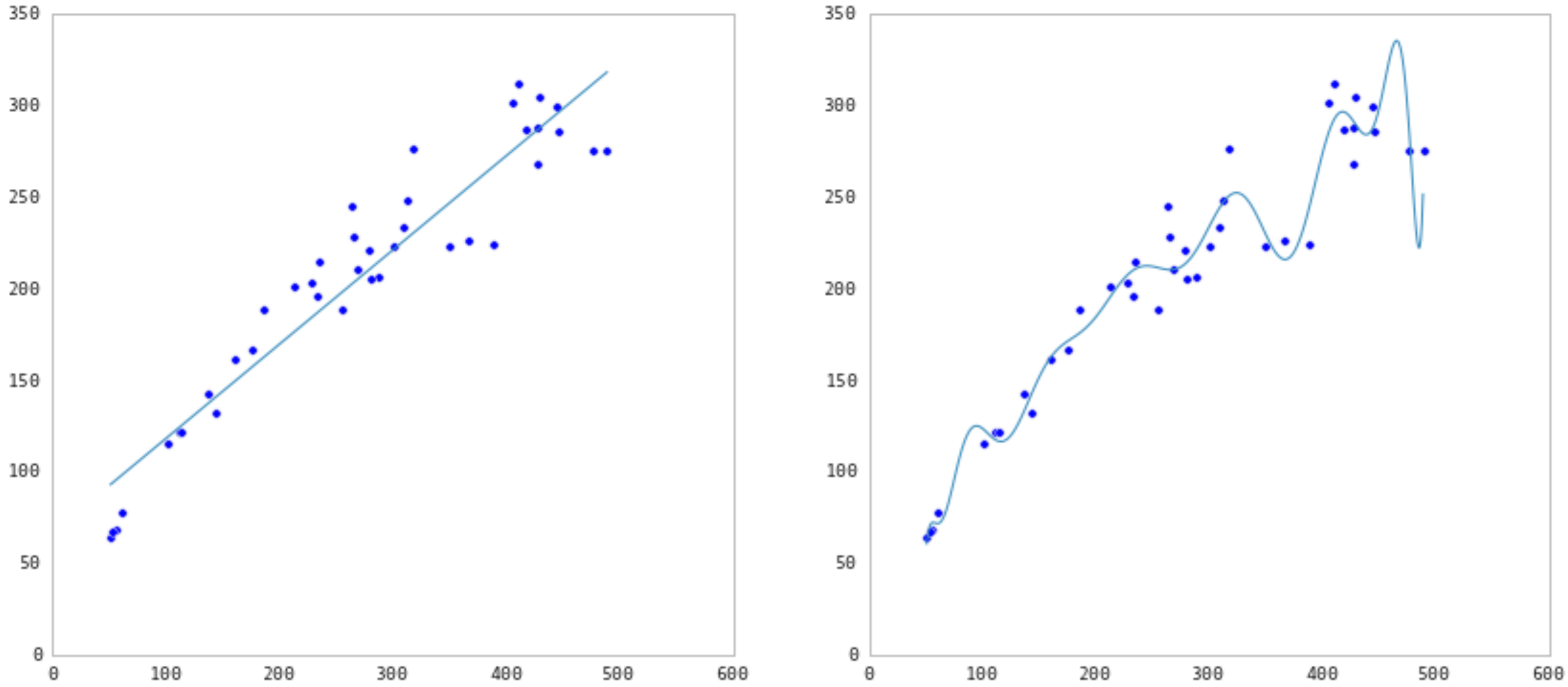


The black line gets a good "sense" of the shape of the data

The green line is overfit, its trying too hard

Source: http://www.dtreg.com

UNDERFITTING AND OVERFITTING



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

Q: What's wrong with training error?

Thought experiment:

Suppose we train our model using the entire dataset.

Q: How low can we push the training error?

- *We can make the model arbitrarily complex (effectively "memorizing" the entire training set).*

A: Down to zero!

Q: What's wrong with training error?

Thought experiment:

Suppose we train our model using the entire dataset.

Q: How low can we push the training error?

- *We can make the model arbitrarily complex (effectively "memorizing" the entire training set).*

A: Down to zero!

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

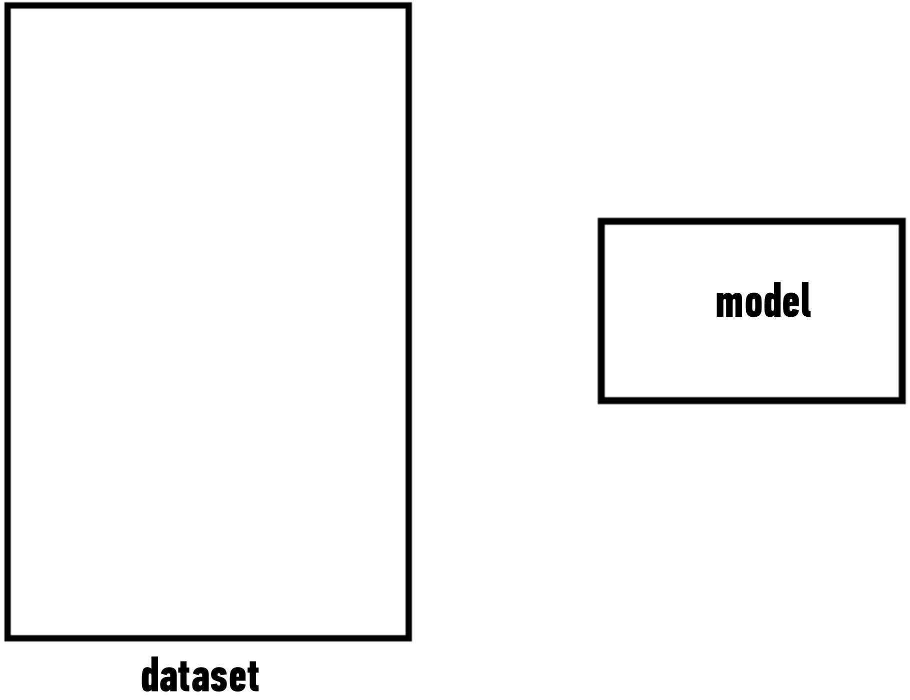
WHY THIS MATTERS

The data that we are given for prediction won’t always be the end of the data stream!

We will gather data and build and iterate over models however the whole ***point*** of building the model was to predict unseen test cases

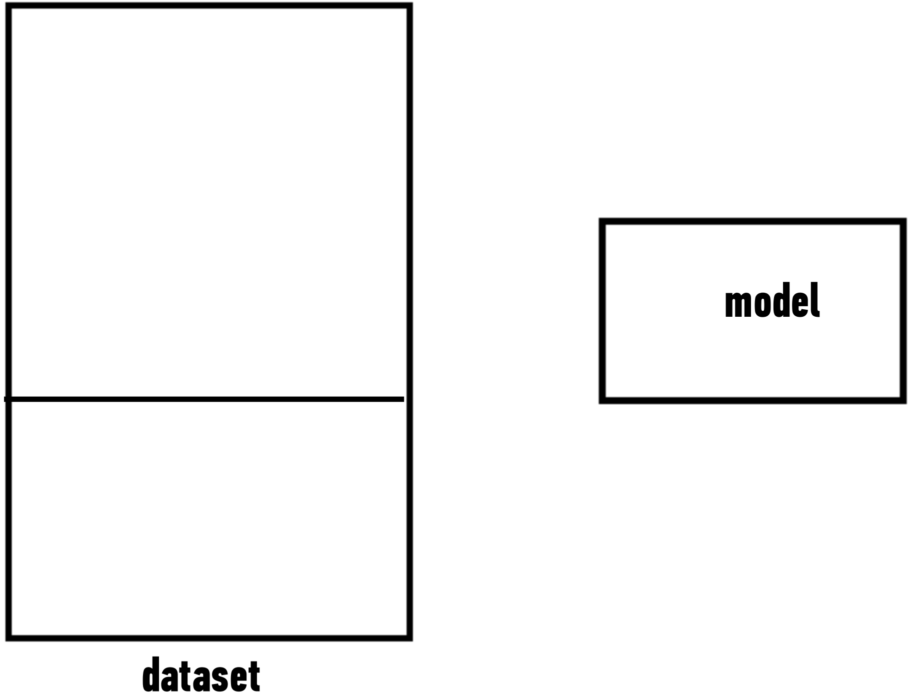
Examples: new UFO sightings will come in, new Iris’ will be found, new children will be born

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



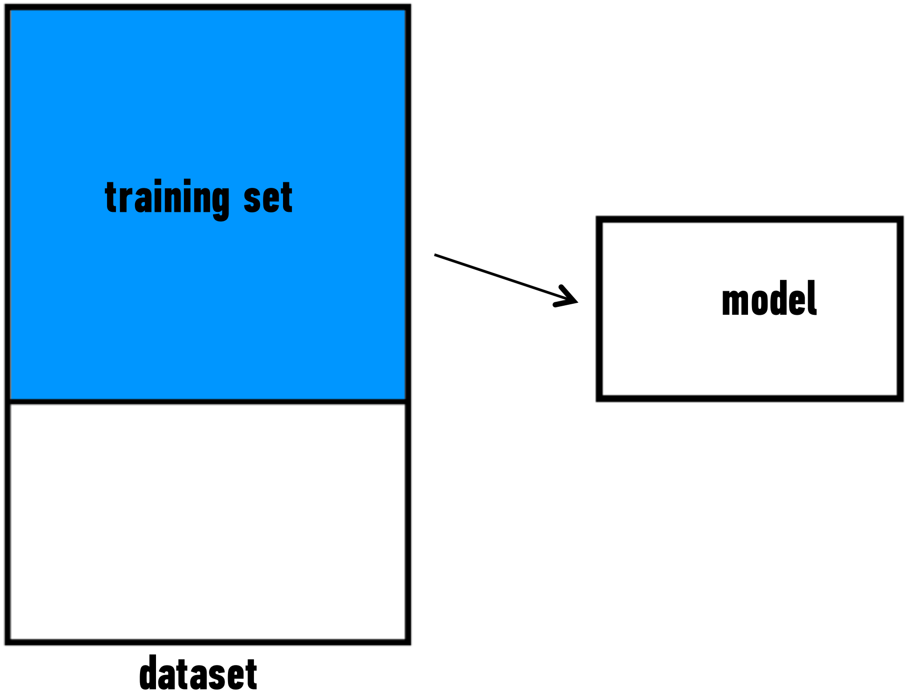
Q: How can we make a model that generalizes well?

1) split dataset



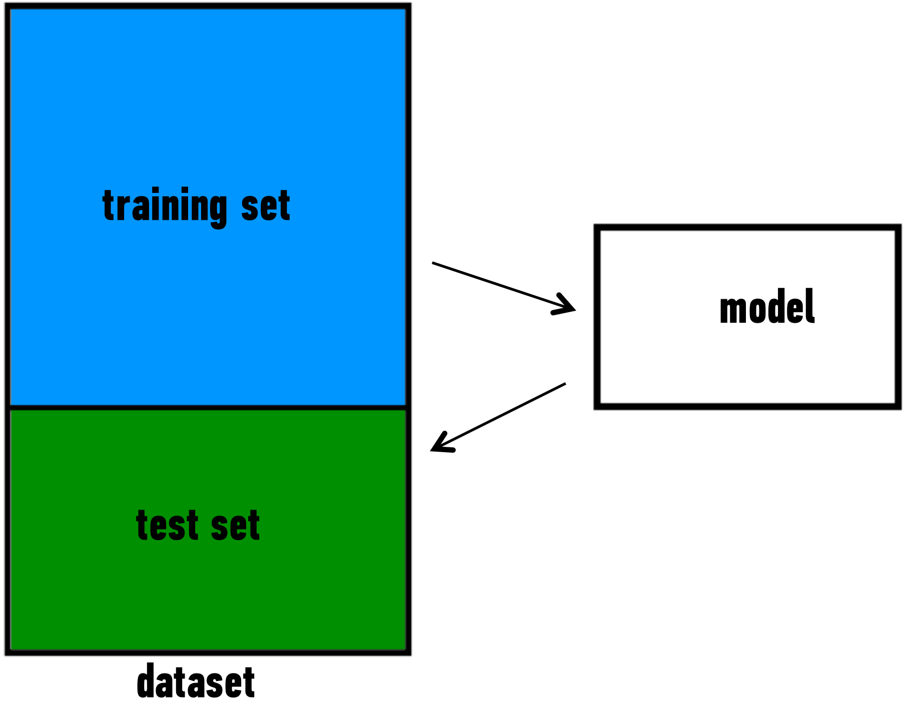
Q: How can we make a model that generalizes well?

1) split dataset
2) train model



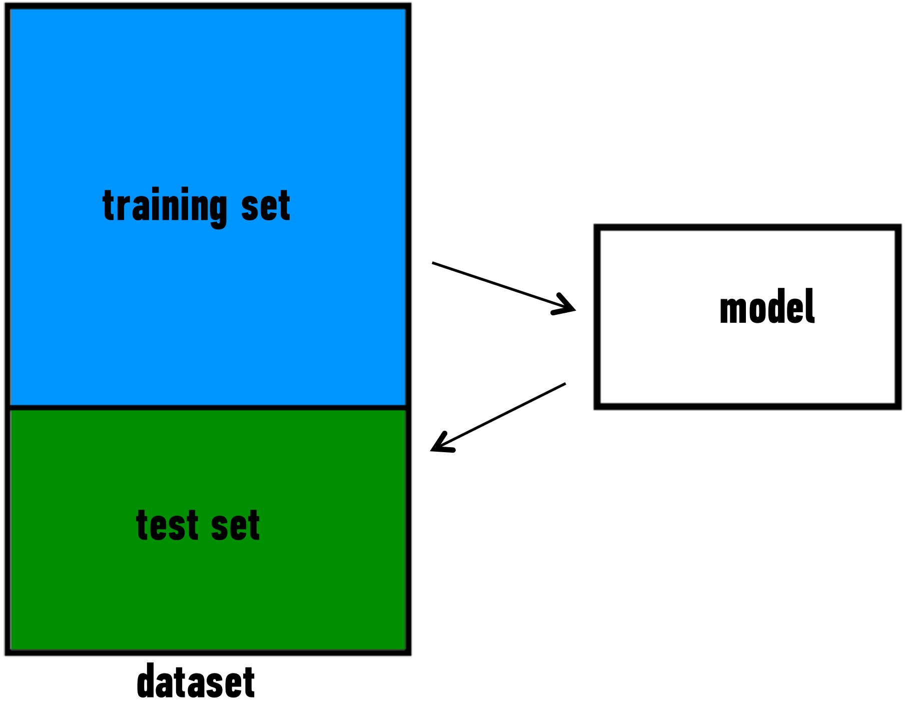
Q: How can we make a model that generalizes well?

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



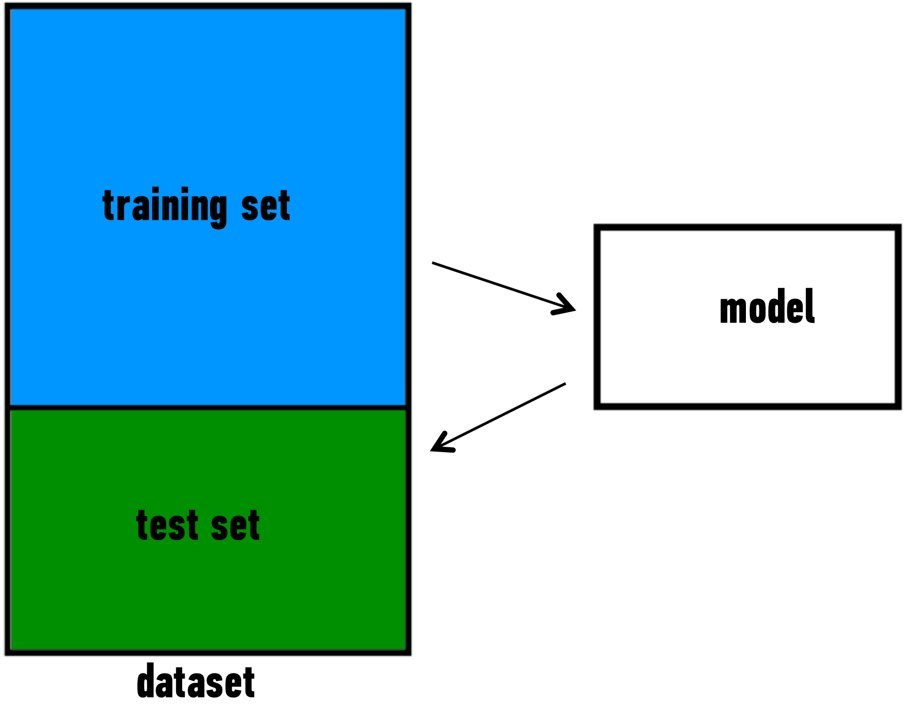
Q: How can we make a model that generalizes well?

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



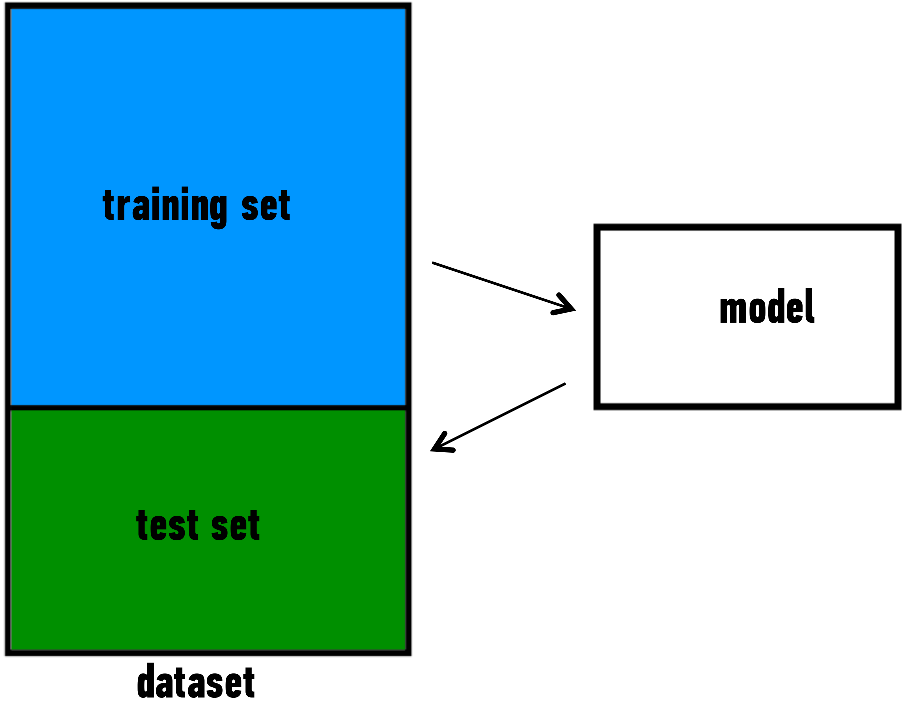
Q: How can we make a model that generalizes well?

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



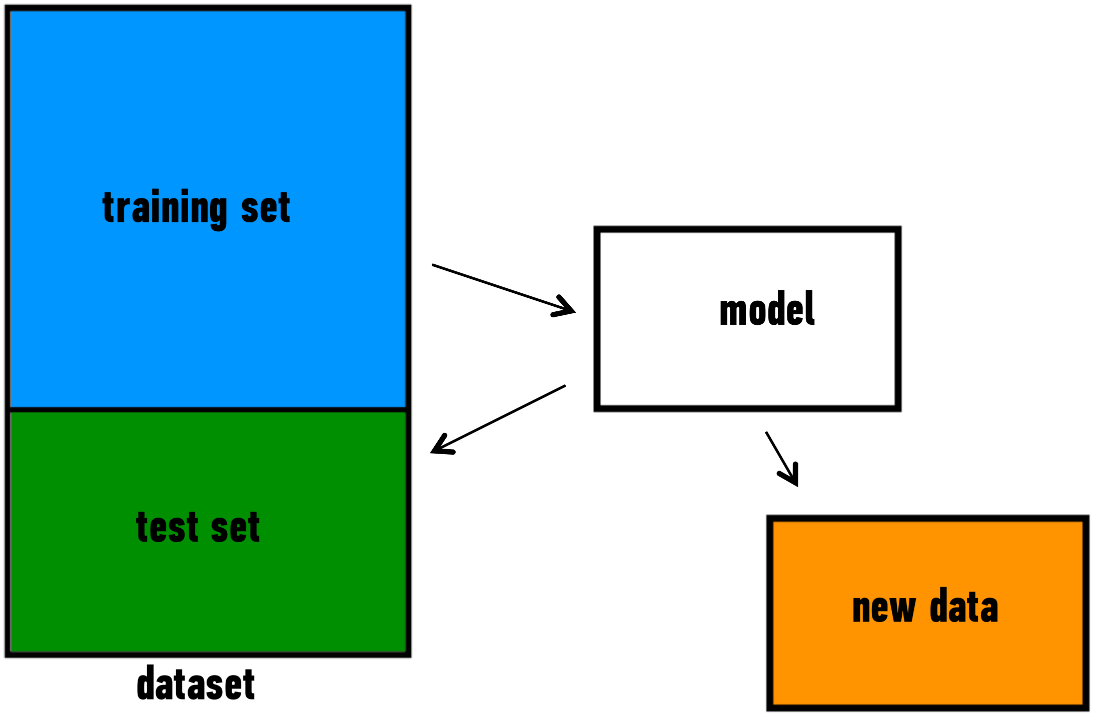
Q: How can we make a model that generalizes well?

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



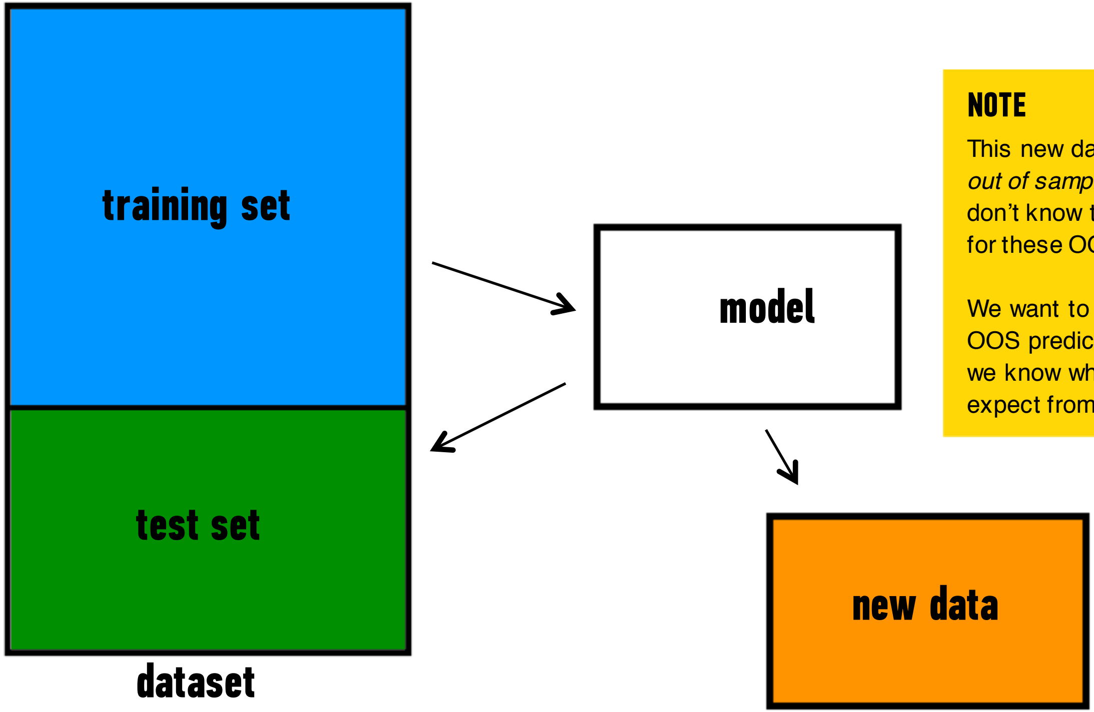
Q: How can we make a model that generalizes well?

1) split dataset
2) train model
3) test model
4) parameter tuning
5) choose best model
6) train on all data
7) make predictions on new 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) make predictions on new data



NOTE

This new data is called *out of sample* data. We don't know the labels for these OOS records!

We want to estimate OOS prediction error so we know what to expect from our model.

Suppose we do the train/test split.

Q: How well does test set error predict OOS?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the test set error remain the same?

A: Of course not!

Suppose we do the train/test split.

Q: How well does test set error predict OOS?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the test set error remain the same?

A: Of course not!

A: On its own, not very well.

Suppose we do the train/test split.
Q: How well does test set error predict OOS?

Thought experiment:
Suppose we had done a different train/test split.
Q: Would the test set error remain the same?
A: Of course not!

A: On its own, not very well.

NOTE

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

Something is still missing!

Q: How can we do better?

Thought experiment:

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

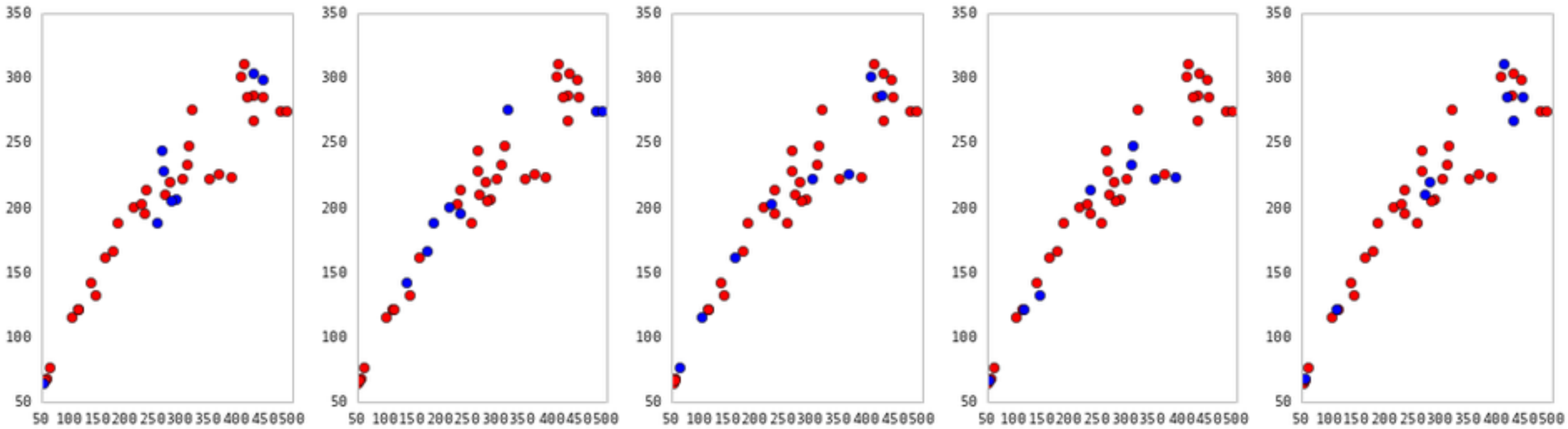
Q: What if we did a bunch of these and took the average?

A: Now you're talking!

A: Cross-validation.

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.



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:

1) More accurate estimate of OOS prediction error.

2) More efficient use of data than single train/test split.
   - Each record in our dataset is used for both training and testing.

3) Presents tradeoff between efficiency an computational expense.
   - 10-fold CV is 10x more expensive than a single train/test split

4) Can be used for parameter tuning and model selection.

DATA SCIENCE