GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

EVALUATION METRICS AND PROCEDURES

FOR PREDICTION:

I. EVALUATION METRICS
II. EVALUATION PROCEDURES

I. EVALUATION METRICS

- For categorical labels
- For rankings/scorings
- For numeric predictions

Confusion Matrix:

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

Confusion Matrix:

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

- What are the labels?

Confusion Matrix:

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

- What are the labels?
- What is “positive”? (Connect to “true positives” etc.)

Confusion Matrix:

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

- What are the labels?
- What is "positive"? (Connect to "true positives" etc.)
- How good is the result?

Confusion Matrix:

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

- What are the labels?
- What is “positive”? (Connect to “true positives” etc.)
- How good is the result?
- How can we quantify how good it is?

Confusion Matrix:

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

- What are the labels?
- What is “positive”? (Connect to “true positives” etc.)
- How good is the result?
- How can we quantify how good it is?
- How can we extend to more than two labels?

EVALUATION METRICS – CATEGORICAL LABELS

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

EVALUATION METRICS – CATEGORICAL LABELS

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

Accuracy = (blue + green) / total

Precision = blue / (blue + red)

Recall = blue / (blue + yellow)

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

Accuracy = (blue + green) / total

Precision = blue / (blue + red)

Recall = blue / (blue + yellow)

True Positive Rate =

blue / (blue + yellow)

False Positive Rate =

red / (red + green)

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

Accuracy = (blue + green) / total

Precision = blue / (blue + red)

Recall = blue / (blue + yellow)

True Positive Rate =
blue / (blue + yellow)

False Positive Rate =
red / (red + green)

Also F scores, which combine precision and recall

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

Accuracy = (blue + green) / total

Precision = blue / (blue + red)

Recall = blue / (blue + yellow)

True Positive Rate =
blue / (blue + yellow)

False Positive Rate =
red / (red + green)

Also F scores, which combine precision and recall

and kappas

	Predicted Spam	Predicted Ham
Actually Spam	1516	2
Actually Ham	1	855

Accuracy = (blue + green) / total

Precision = blue / (blue + red)

Recall = blue / (blue + yellow)

True Positive Rate =
blue / (blue + yellow)

False Positive Rate =
red / (red + green)

Also F scores, which combine precision and recall

and kappas

NOTE

There are more methods and many more terms that can be used for many of these!

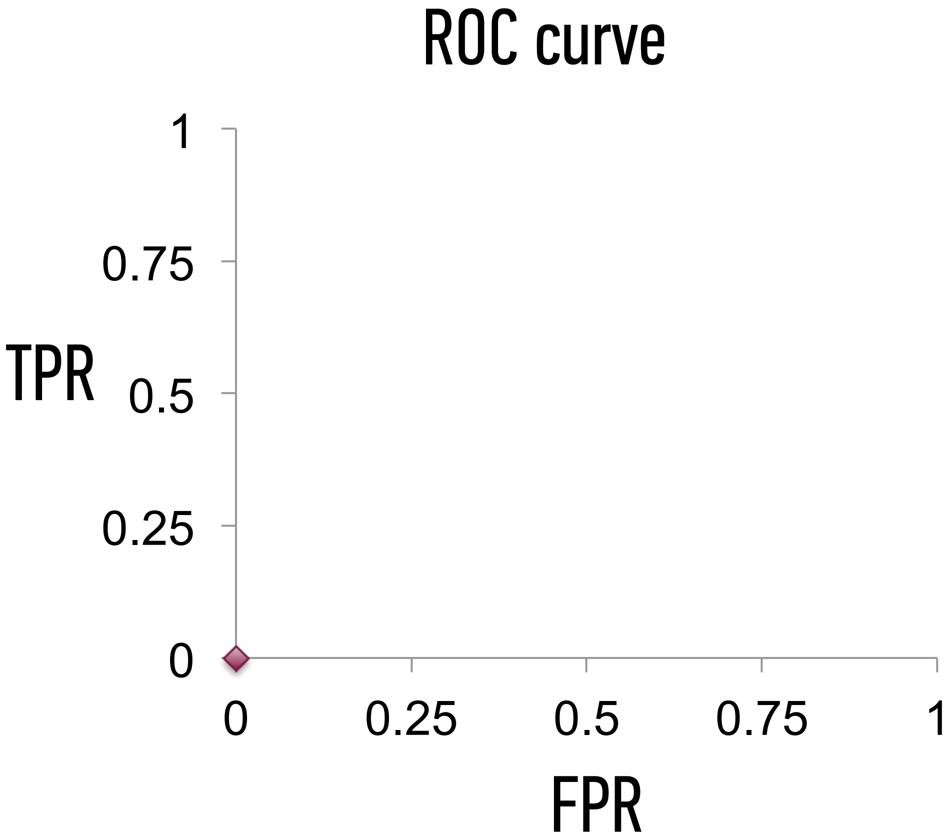
Email Number	Score	True Label
5	0.93	Spam
8	0.91	Spam
2	0.84	Spam
1	0.6	Ham
7	0.54	Spam
3	0.22	Ham
4	0.10	Ham
6	0.02	Ham

Every email gets a spamminess score.

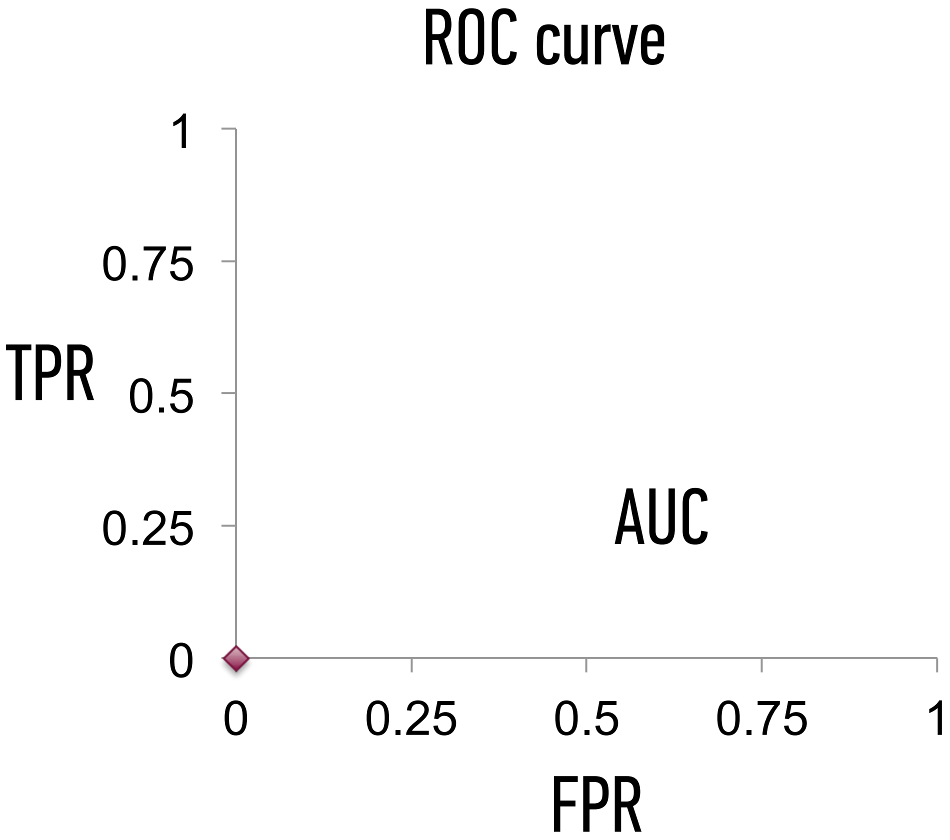
Choosing a cut-off, this becomes a classification.

How do we choose a cut-off?
How do we evaluate the ranking without choosing a cut-off?

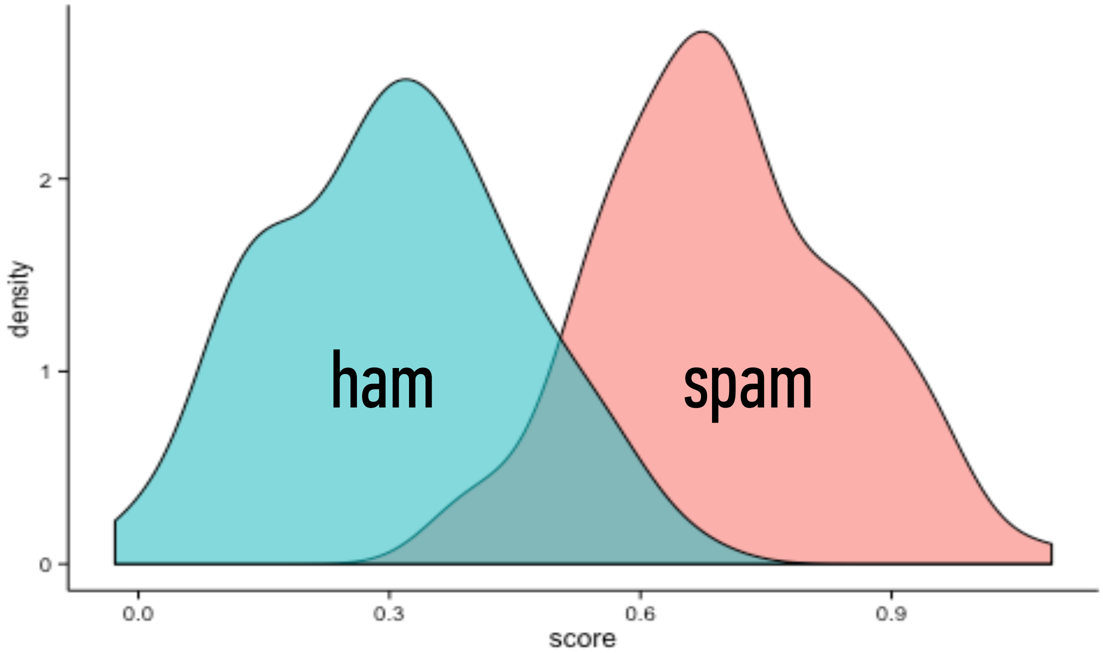
Email Number	Score	True Label
5	0.93	Spam
8	0.91	Spam
2	0.84	Spam
1	0.6	Ham
7	0.54	Spam
3	0.22	Ham
4	0.10	Ham
6	0.02	Ham



Email Number	Score	True Label
5	0.93	Spam
8	0.91	Spam
2	0.84	Spam
1	0.6	Ham
7	0.54	Spam
3	0.22	Ham
4	0.10	Ham
6	0.02	Ham



Another interpretation of AUC (cf. common language effect size)



For ratings/scoring that aren't for classification, there are other evaluation metrics such as Kendall's tau, types of gain, etc.

Briefly:

- Mean Squared Error
- Mean Absolute Error
- others possible

II. EVALUATION PROCEDURES

Q: What's wrong with training error?

Q: What's wrong with training error?

Thought experiment:

Suppose we train our model using the entire dataset.

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?

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).*

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!

NOTE

This phenomenon is called *overfitting*.

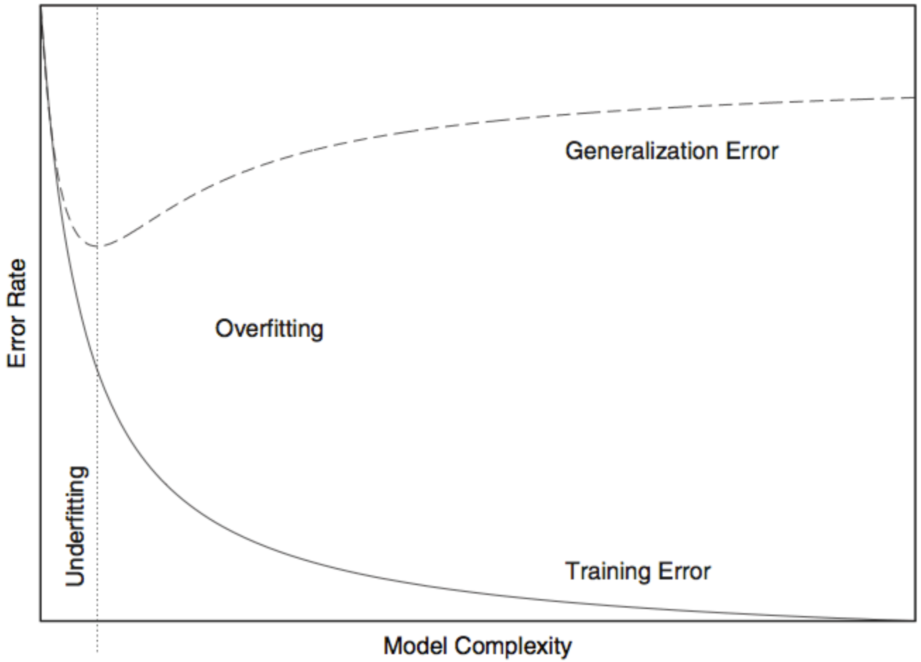
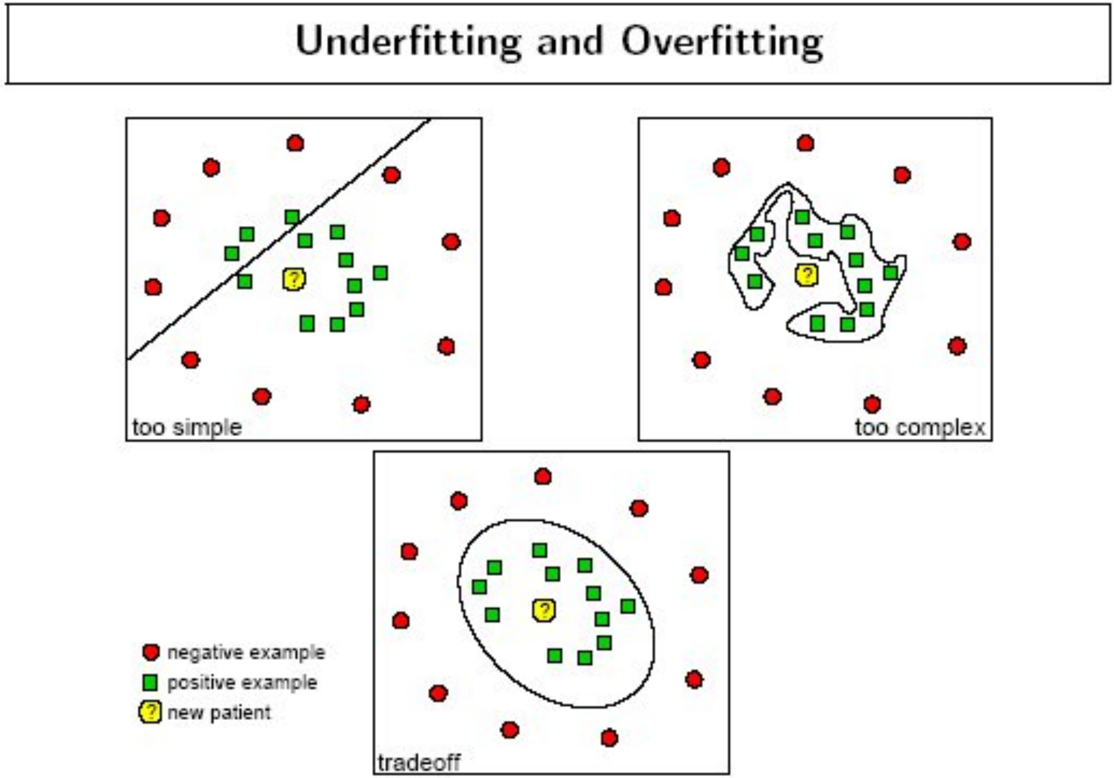


FIGURE 18-1. Overfitting: as a model becomes more complex, it becomes increasingly able to represent the training data. However, such a model is overfitted and will not generalize well to data that was not used during training.

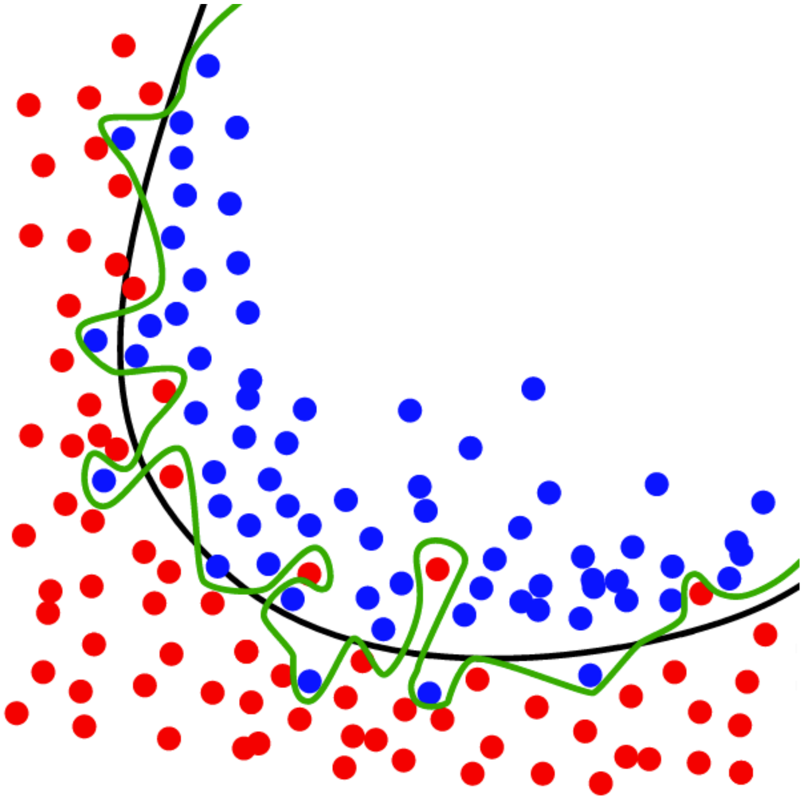
source: Data Analysis with Open Source Tools, by Philipp K. Janert. O'Reilly Media, 2011.

OVERFITTING – EXAMPLE



source: http://www.dtreg.com

OVERFITTING – EXAMPLE



source: http://www.dtreg.com

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).*

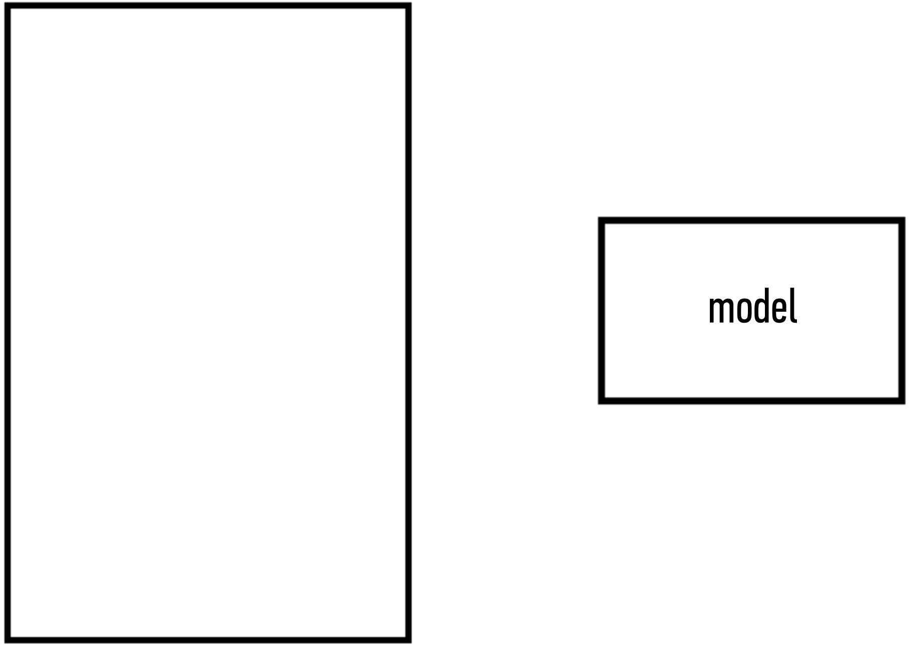
A: Down to zero!

NOTE

This phenomenon is called *overfitting*.

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

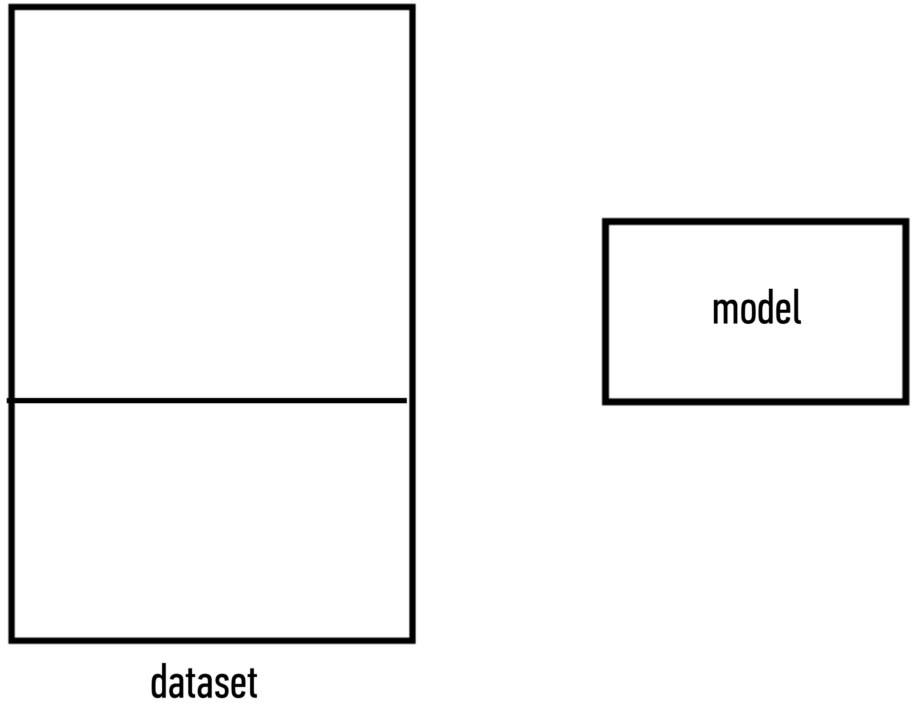
Q: How can we make a model that generalizes well?



dataset

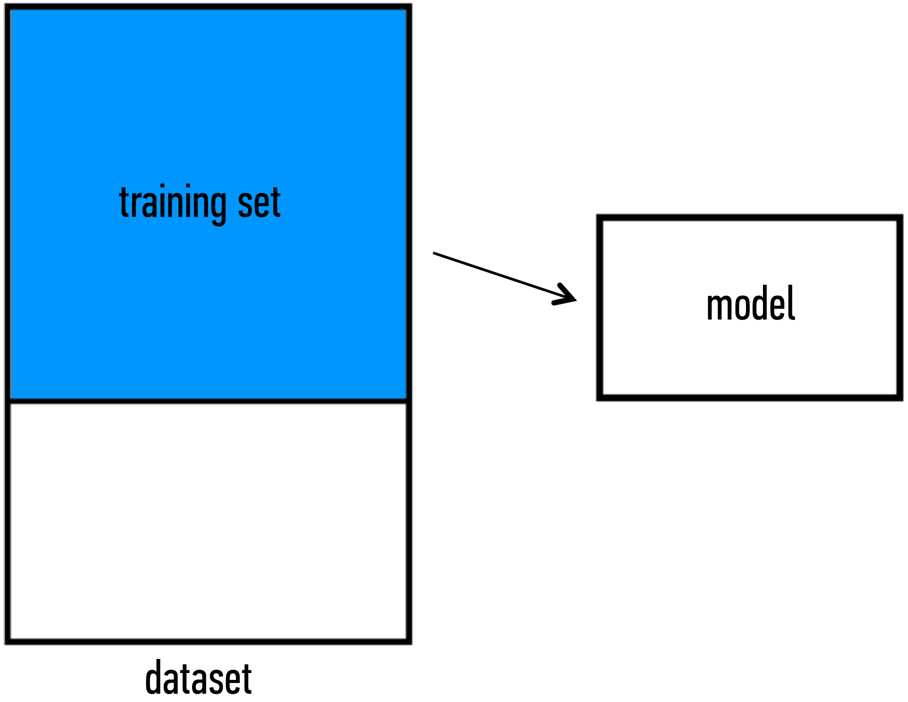
Q: How can we make a model that generalizes well?

1) split dataset



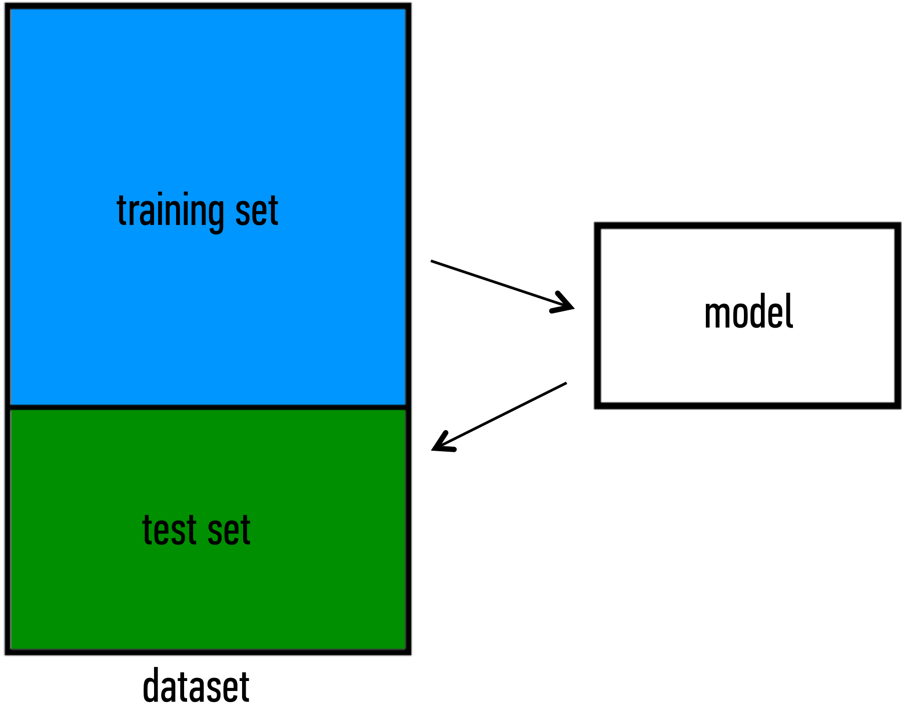
Q: How can we make a model that generalizes well?

1) split dataset
2) train model



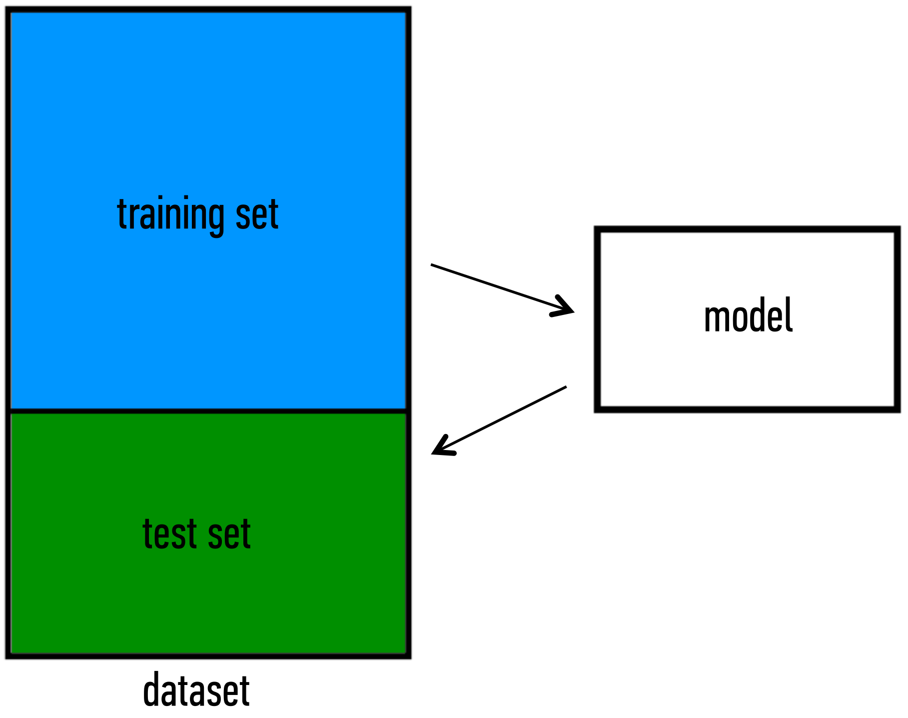
Q: How can we make a model that generalizes well?

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



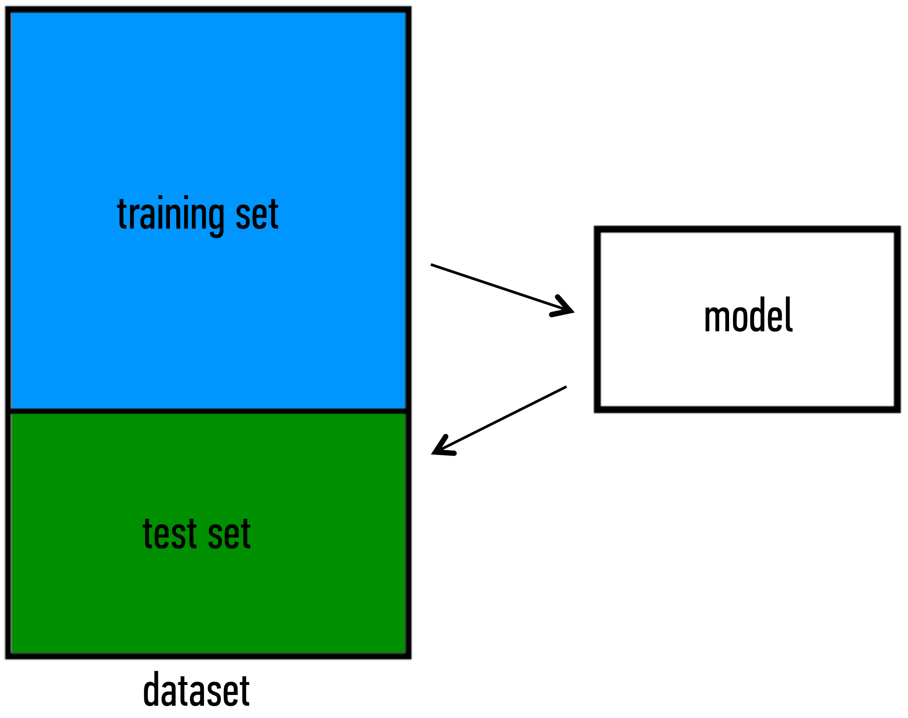
Q: How can we make a model that generalizes well?

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



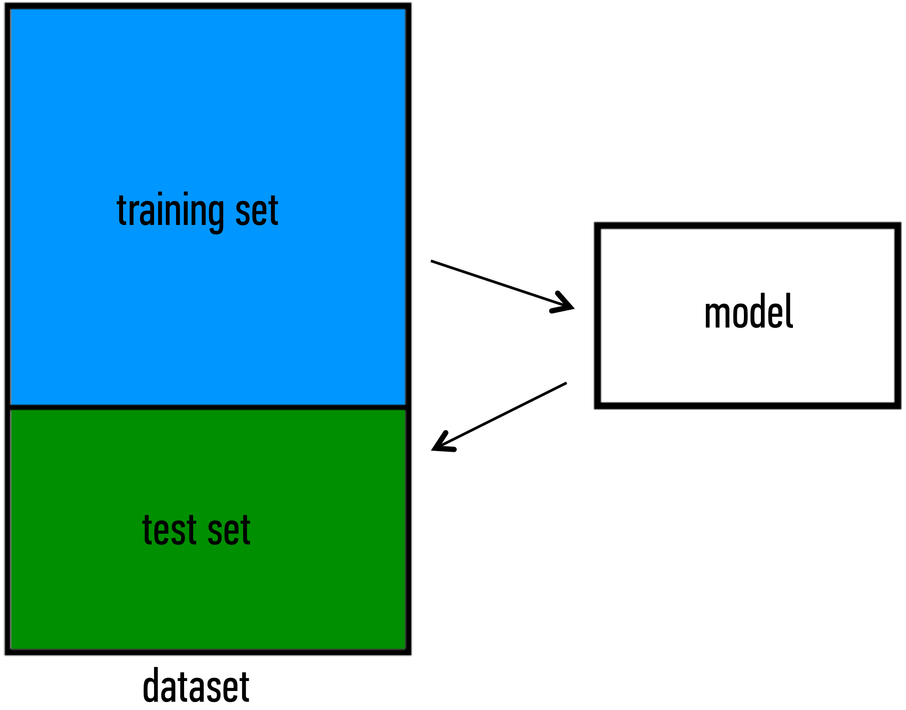
Q: How can we make a model that generalizes well?

1) split dataset
2) train model
3) test model
4) iterate
5) choose final model



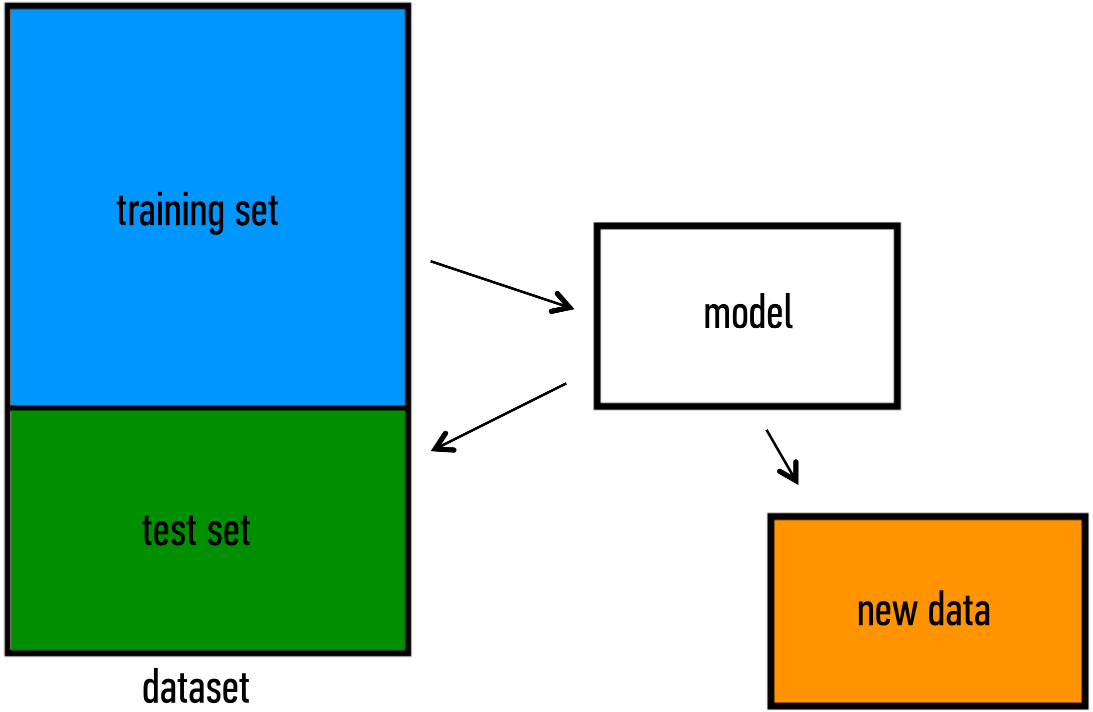
Q: How can we make a model that generalizes well?

1) split dataset
2) train model
3) test model
4) iterate
5) choose final model
6) train on all data



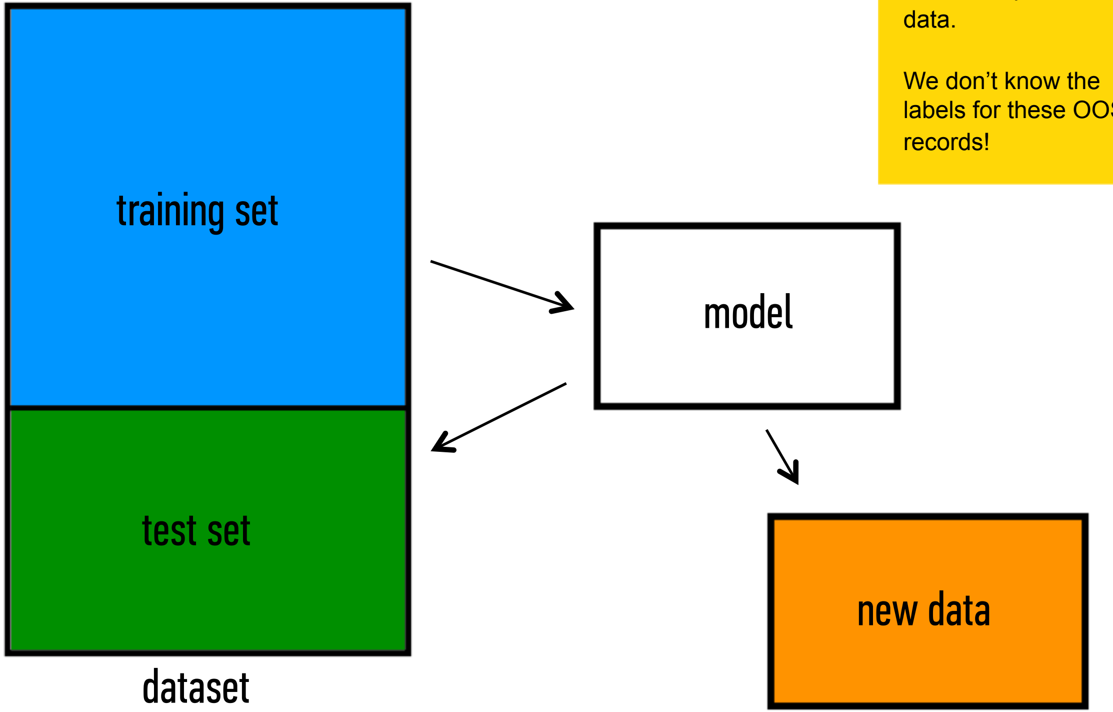
Q: How can we make a model that generalizes well?

1) split dataset
2) train model
3) test model
4) iterate
5) choose final model
6) train on all data
7) make predictions



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

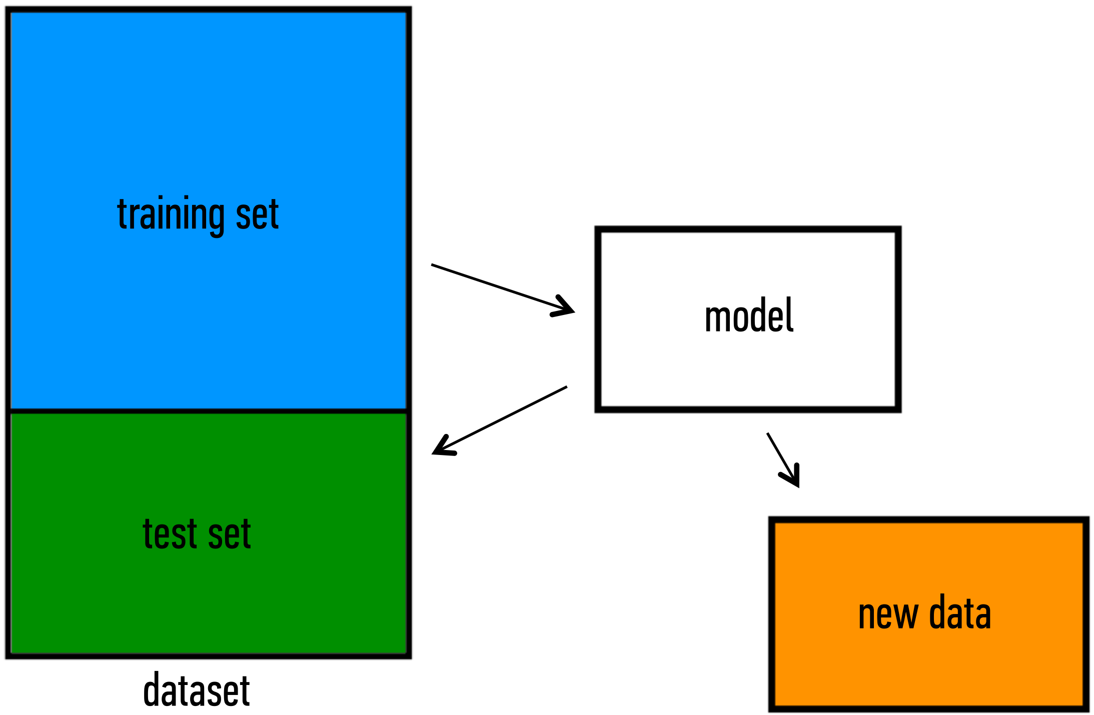
1) split dataset
2) train model
3) test model
4) iterate
5) choose final model
6) train on all data
7) make predictions



NOTE
This new data is called *out of sample* data.

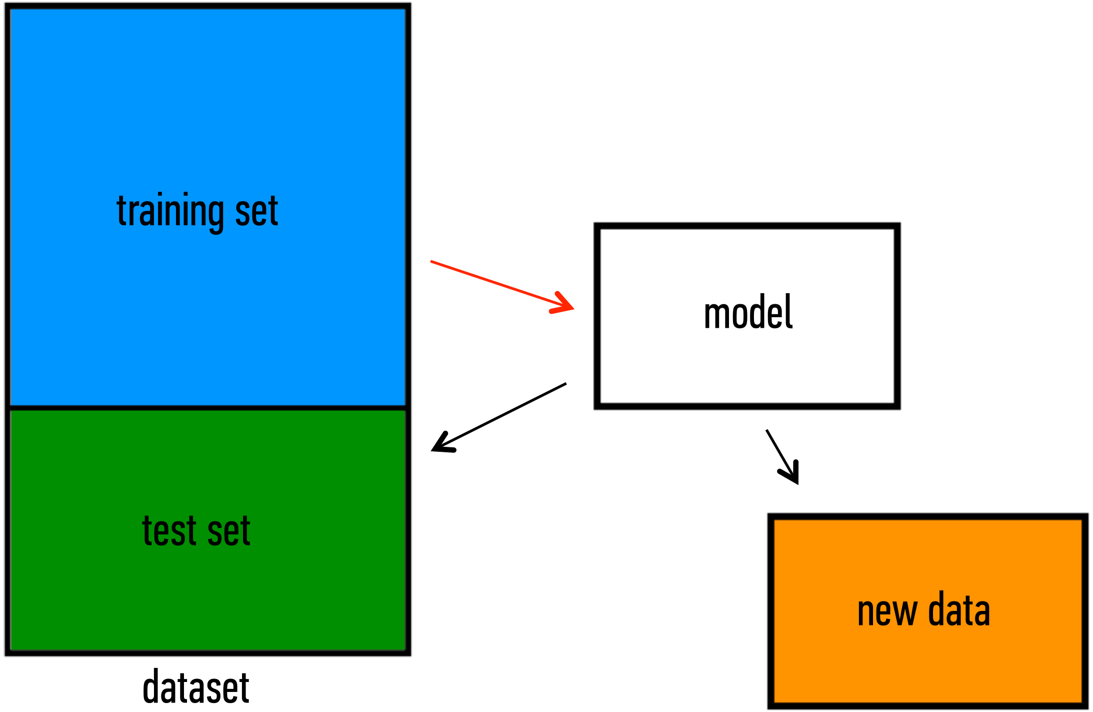
We don't know the labels for these OOS records!

Q: What types of prediction error will we run into?



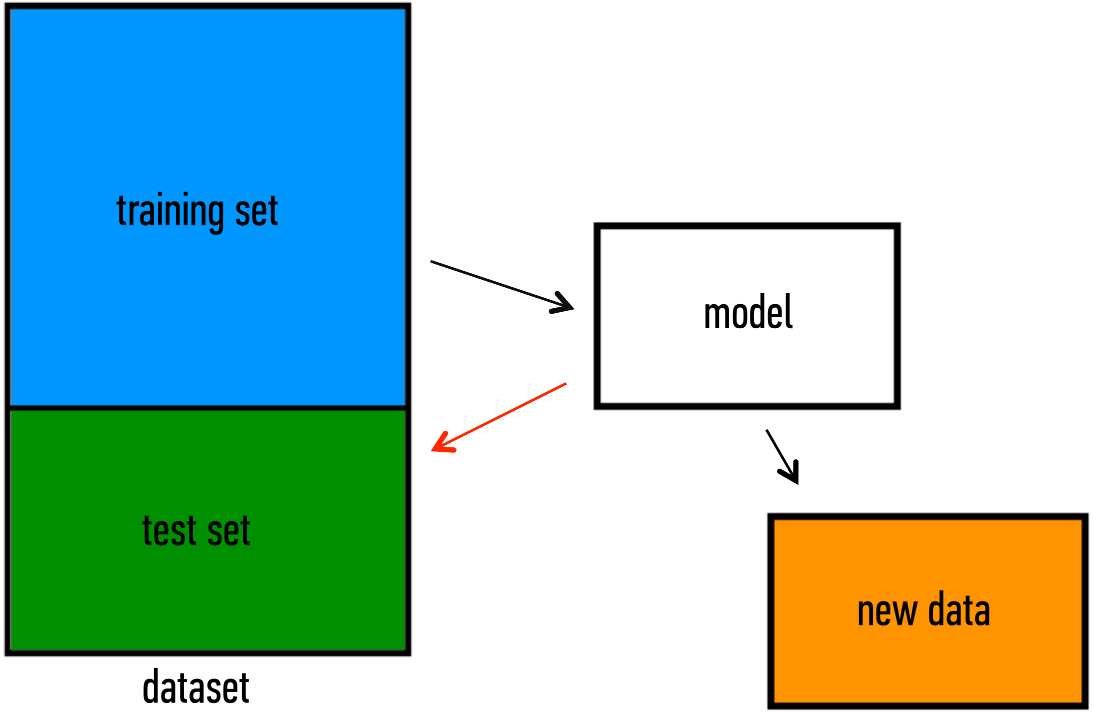
Q: What types of prediction error will we run into?

1) training error



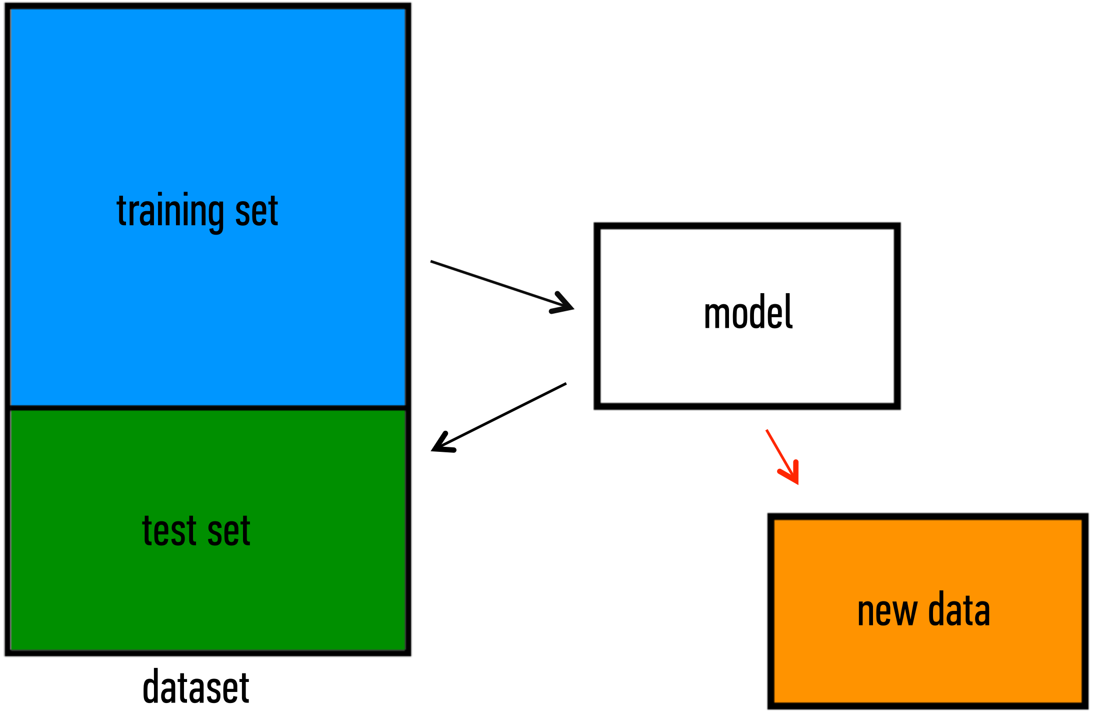
Q: What types of prediction error will we run into?

1) training error
2) generalization error



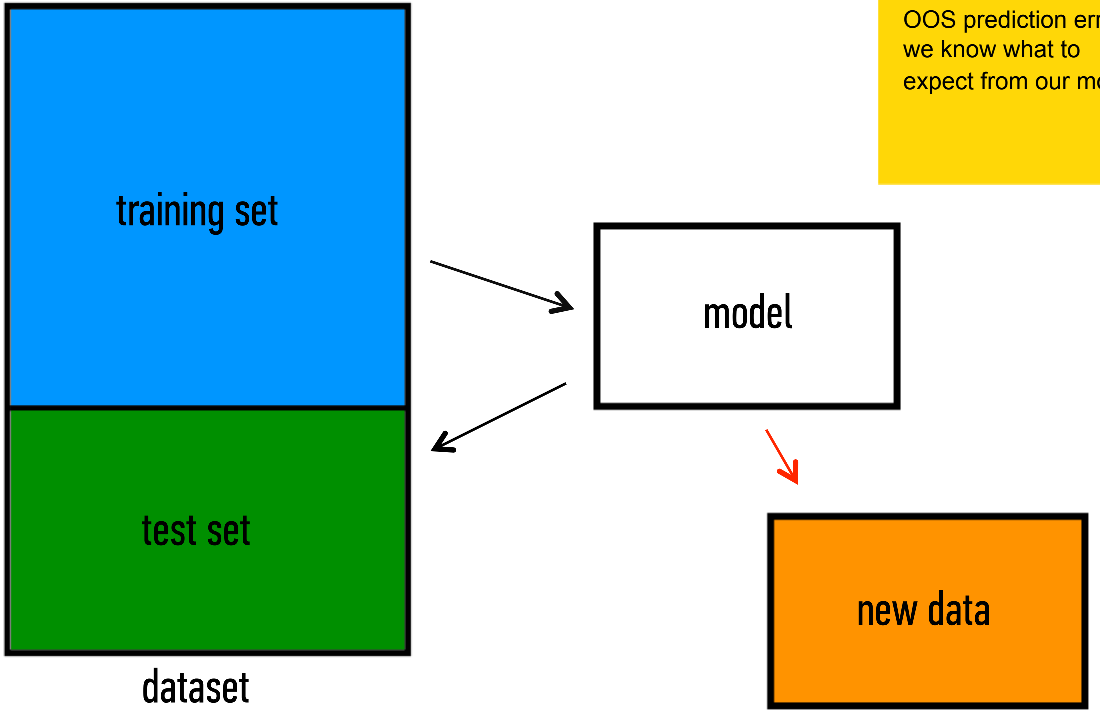
Q: What types of prediction error will we run into?

1) training error
2) generalization error
3) OOS error



Q: What types of prediction error will we run into?

1) training error
2) generalization error
3) OOS error



NOTE

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

Suppose we do the train/test split.

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the generalization error remain the same?

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the generalization error remain the same?

A: Of course not!

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the generalization 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 generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the generalization error remain the same?

A: Of course not!

A: On its own, not very well.

NOTE

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

Something is still missing!

Something is still missing!

Q: How can we do better?

Something is still missing!

Q: How can we do better?

Thought experiment:

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

Something is still missing!

Q: How can we do better?

Thought experiment:

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

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

Something is still missing!

Q: How can we do better?

Thought experiment:

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

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

A: Now you're talking!

Something is still missing!

Q: How can we do better?

Thought experiment:

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

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

A: Now you're talking!

A: Cross-validation.

Steps for n-fold cross-validation:

Steps for n-fold cross-validation:

1) Randomly split the dataset into n equal partitions.

Steps for n-fold cross-validation:

1) Randomly split the dataset into n equal partitions.
2) Use partition 1 as test set & union of other partitions as training set.

Steps for n-fold cross-validation:

1) Randomly split the dataset into n equal partitions.
2) Use partition 1 as test set & union of other partitions as training set.
3) Find generalization error.

Steps for n-fold cross-validation:

1) Randomly split the dataset into n equal partitions.
2) Use partition 1 as test set & union of other partitions as training set.
3) Find generalization error.
4) Repeat steps 2-3 using a different partition as the test set at each iteration.

Steps for n-fold cross-validation:

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

Features of n-fold cross-validation:

Features of n-fold cross-validation:

1) More accurate estimate of OOS prediction error.

Features of n-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.

Features of n-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 and computational expense.
    - 10-fold CV is 10x more expensive than a single train/test split

Features of n-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 and computational expense.
   - 10-fold CV is 10x more expensive than a single train/test split
4) Can be used for model selection.

Features of n-fold cross-validation:

> NOTE
>
> Leave one out cross-validation is a special case of n-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 and computational expense.
   - 10-fold CV is 10x more expensive than a single train/test split
4) Can be used for model selection.