Cross Validation

INFO-4604, Applied Machine Learning
University of Colorado Boulder

Oct 26, 2020

Prof. Abe Handler

HW3 grades will be back shortly

HW4 is out

Start early! Parts of this are harder

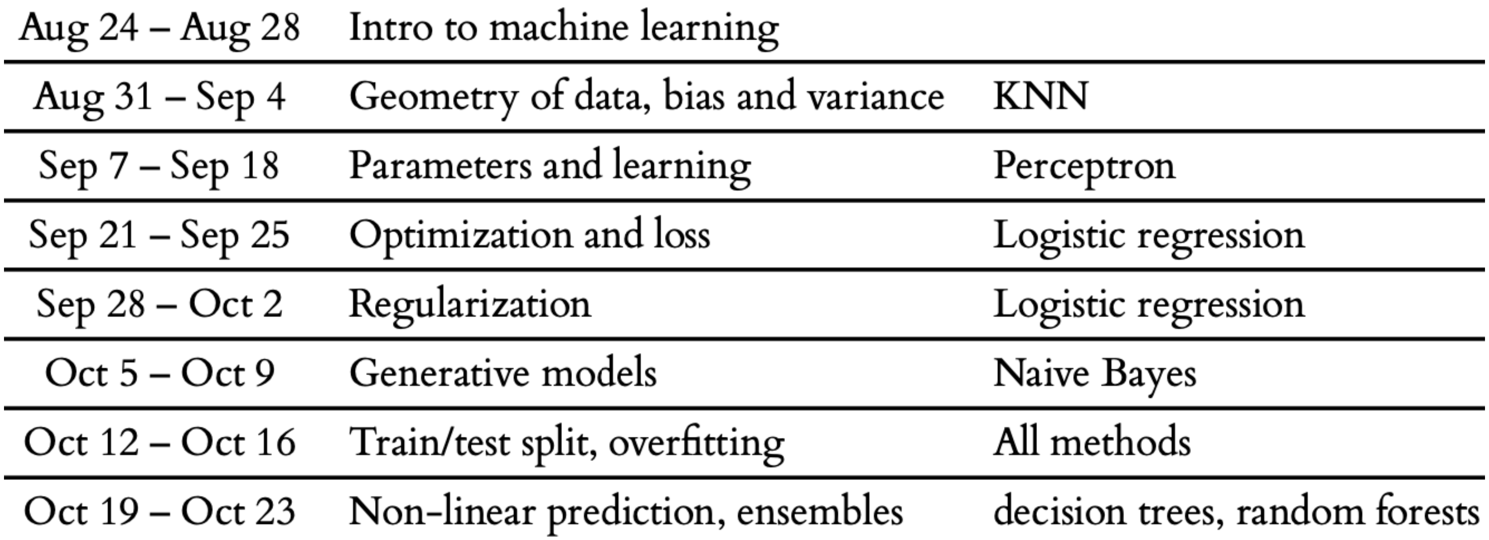
A note about grades

If you are concerned about your grade in this class, please email me or come to office hours. **My only concern is that you learn the material as well as possible this semester.** Grades are a check on your learning; not the point of the class!

So if you are worried about your grade and willing to put in work to show that you understand the material, we can figure out a way for you to show what you know. **Don’t wait until the end of the semester.**

Where are we?

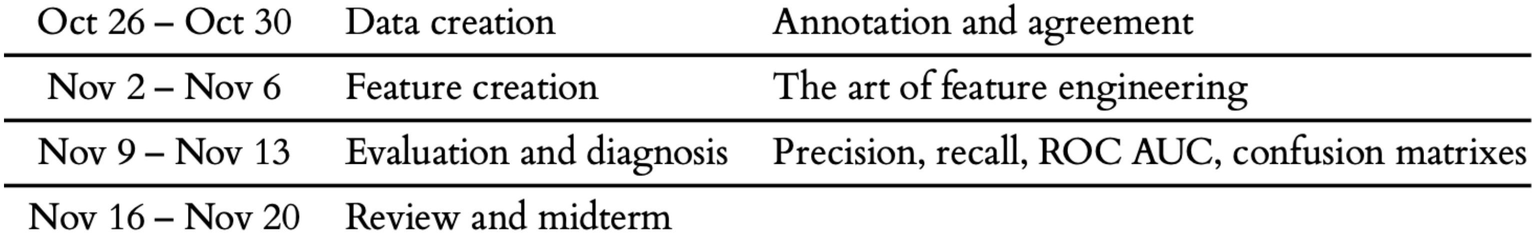
Unit 1: Intro to ML



Aug 24 – Aug 28	Intro to machine learning	
Aug 31 – Sep 4	Geometry of data, bias and variance	KNN
Sep 7 – Sep 18	Parameters and learning	Perceptron
Sep 21 – Sep 25	Optimization and loss	Logistic regression
Sep 28 – Oct 2	Regularization	Logistic regression
Oct 5 – Oct 9	Generative models	Naive Bayes
Oct 12 – Oct 16	Train/test split, overfitting	All methods
Oct 19 – Oct 23	Non-linear prediction, ensembles	decision trees, random forests

Where are we?

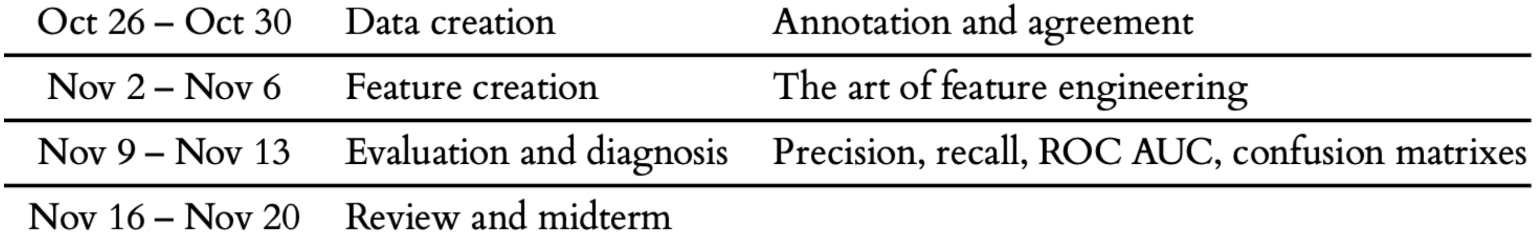
Unit 2: Practical ML



Oct 26 – Oct 30	Data creation	Annotation and agreement
Nov 2 – Nov 6	Feature creation	The art of feature engineering
Nov 9 – Nov 13	Evaluation and diagnosis	Precision, recall, ROC AUC, confusion matrixes
Nov 16 – Nov 20	Review and midterm	

Where are we?

Unit 2: Practical ML



Oct 26 – Oct 30	Data creation	Annotation and agreement
Nov 2 – Nov 6	Feature creation	The art of feature engineering
Nov 9 – Nov 13	Evaluation and diagnosis	Precision, recall, ROC AUC, confusion matrixes
Nov 16 – Nov 20	Review and midterm	

Midterm will be a small, week-long project

Where are we?

- So a little bit of shift this week
- Less focus on ML fundamentals like optimization or generative models
- More focus on applying ML in practice
- But still some very fundamental concepts coming up!

For example, cross validation!

Evaluation

In homework, you’ve seen that:

- training data is usually separate from **test data**
- training accuracy is often much higher than test accuracy
  - Training accuracy is what your classifier is optimizing for (plus regularization), but not a good indicator of how it will perform

Evaluation

Distinction between:

- **in-sample** data
  - The data that is available when building your model
  - “Training” data in machine learning terminology
- **out-of-sample** data
  - Data that was not seen during training
  - Also called **held-out data** or a **holdout set**
  - Useful to see what your classifier will do on data it hasn’t seen before
  - Usually assumed to be from the same distribution as in-sample data

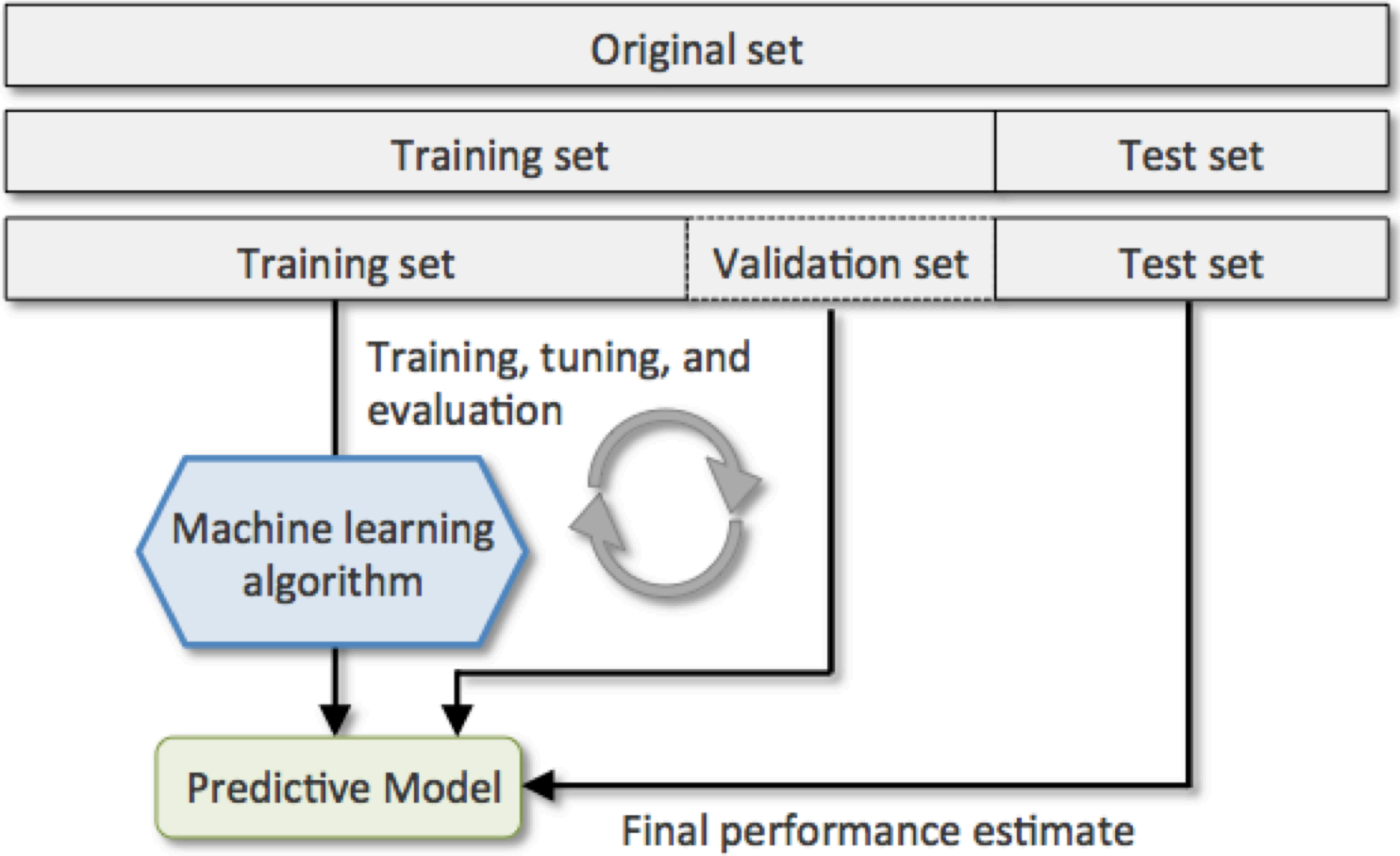
Evaluation

Ideally, you should be “blind” to the test data until you are ready to evaluate your final model

Often you need to evaluate a model repeatedly (e.g., you’re trying to pick the best regularization strength, and you want to see how different values affect the performance)

- If you keep using the same test data, you risk overfitting to the test set
- Should use a different set, still held-out from training data, but different from test set
- We’ll revisit this later

Evaluation


Original set
Training set
Test set
Training set
Validation set
Test set
Training, tuning, and evaluation
Machine learning algorithm
Predictive Model
Final performance estimate

Held-Out Data

Typically you set aside a random sample of your labeled data to use for testing

- A lot of ML datasets you download will already be split into training vs test, so that people use the same splits in different experiments

How much data to set aside for testing? Tradeoff:

- What are the tradeoffs?

Held-Out Data

Typically you set aside a random sample of your labeled data to use for testing

- A lot of ML datasets you download will already be split into training vs test, so that people use the same splits in different experiments

How much data to set aside for testing? Tradeoff:

- Smaller test set: less reliable performance estimate
- Smaller training set: less data for training, probably worse classifier (might underestimate performance)

Held-Out Data

How to spot leaks?

- Ask yourself, am I *really* generalizing to unseen data
- Think through the assumptions in your process
- Experience, alas. Just keep going with ML …

Held-Out Data

A common approach to getting held-out estimates is **k-fold cross validation**

General idea:

- split your data into *k* partitions (“folds”)
- use all but one for training, use the last fold for testing
- Repeat *k* times, so each fold gets used for testing once

This will give you *k* different held-out performance estimates

- Can then average them to get final result

Held-Out Data

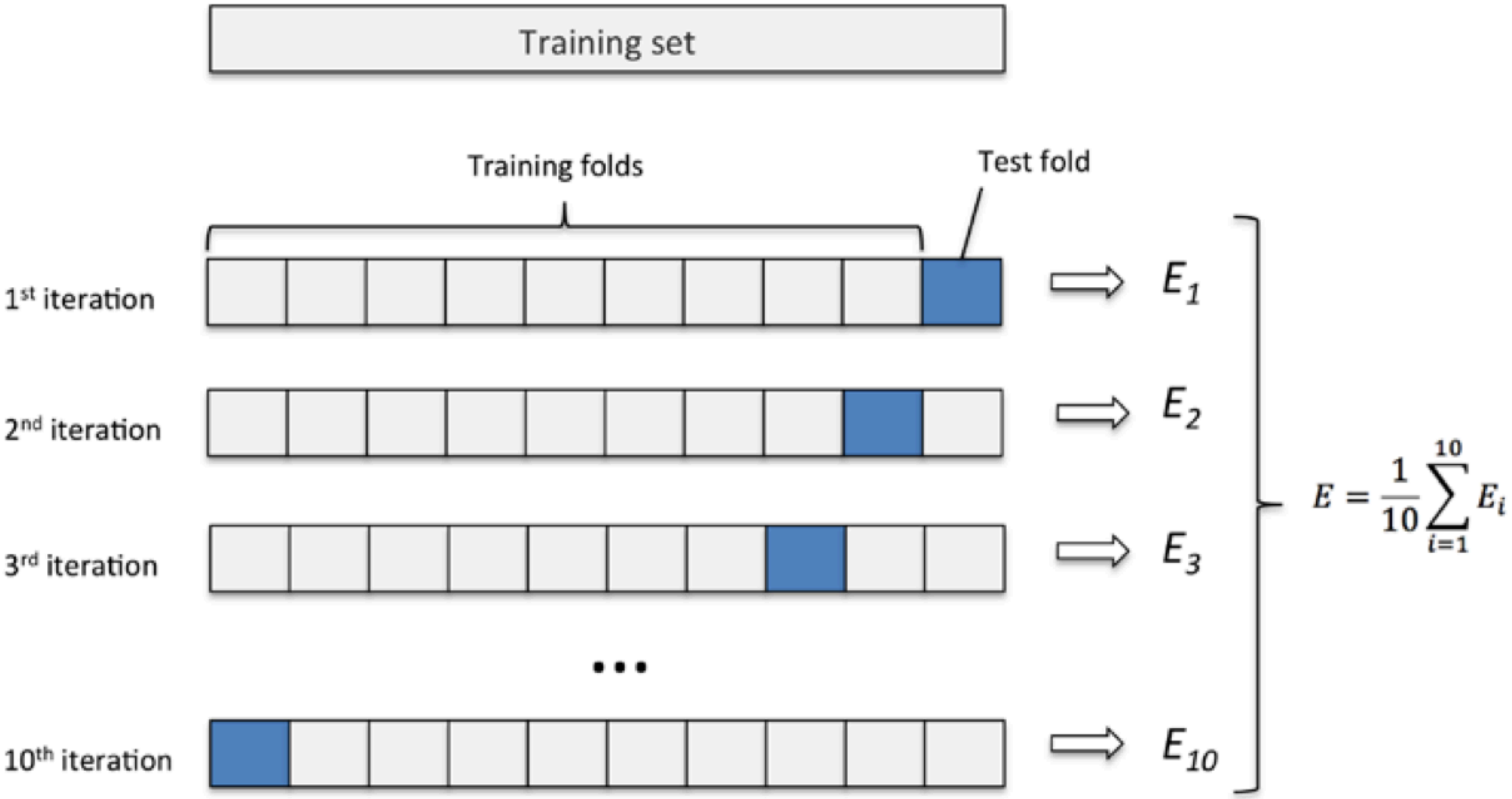


Illustration of 10-fold cross-validation

Held-Out Data

How to choose *k*?

- Generally, larger is better, but limited by efficiency
- Most common values: 5 or 10
- Smaller *k* means less training data used, so your estimate may be an underestimate

When *k* is the number of instances, this is called

leave-one-out cross-validation

- Useful with small datasets, when you want to use as much training data as possible

Held-Out Data

Benefits of obtaining multiple held-out estimates:

- More robust final estimate; less sensitive to the particular test split that you choose
- Multiple estimates also gives you the variance of the estimates; can be used to construct confidence intervals (but not doing this in this class)

Other Considerations

When splitting into train vs test partitions, keep in mind the unit of analysis

Some examples:

- If you are making predictions about people (e.g., guessing someone’s age based on their search queries), probably shouldn’t have data from the same person in both train and test
    - Split on people rather than individual instances (queries)
- If time is a factor in your data, probably want test sets to be from later time periods than training sets
    - Don’t use the future to predict the past

More on leaks

Watch out for information “leaks” between the train and test set

- Say you have data on what customers buy what in what store locations
- You want a general model of what certain kinds of customers tend to buy in which location
- You train a model using the training set, which has data on customer A and store B
- If customer A and store B also appear in the test set, you are applying your knowledge of particular customers and stores in the test set
- You don’t have a general model! You have a model of particular customers

More on leaks

Basically you need to ask: how do I split the dataset fairly?

Other Considerations

If there are errors in your annotations, then there will be errors in your estimates of performance

- Example: your classifier predicts “positive” sentiment but it was labeled “neutral”
- If the label actually should have been (or at least could have been) positive, then your classifier will be falsely penalized

This is another reason why it’s important to understand the quality of the annotations in order to correctly understand the quality of a model

Other Considerations

If your test performance seems “suspiciously” good, trust your suspicions

- Make sure you aren’t accidentally including any training information in the test set. **This is common.**

General takeaway:

- Make sure the test conditions are as similar as possible to the actual prediction environment
- **Don’t trick yourself into thinking your model works better than it does**