COMPSCI361: Introduction to Machine Learning

Fundamentals of Learning

Jörg Simon Wicker

The University of Auckland

SCIENCE
SCHOOL OF COMPUTER SCIENCE
MACHINE LEARNING

Today we will cover...

- Motivation
- Error
- Golden Rule of Machine Learning
- IID
- Fundamental Trade-Off
- Validation Error
- Hyperparameters
- Optimization Bias
- Cross-Validation
- Classifier Evaluation Metrics
- Comparing Classifiers
- The Best Machine Learning Model
  - Examples
  - Examples

Partially based on Slides from University of British Columbia

Supervised Learning Applications

- We motivated supervised learning by the "food allergy" example
- But we can use supervised learning for any input:output mapping
  - E-mail spam filtering
  - Optical character recognition on scanners
  - Recognizing faces in pictures
  - Recognizing tumours in medical images
  - Speech recognition on phones
  - Your problem in industry/research?

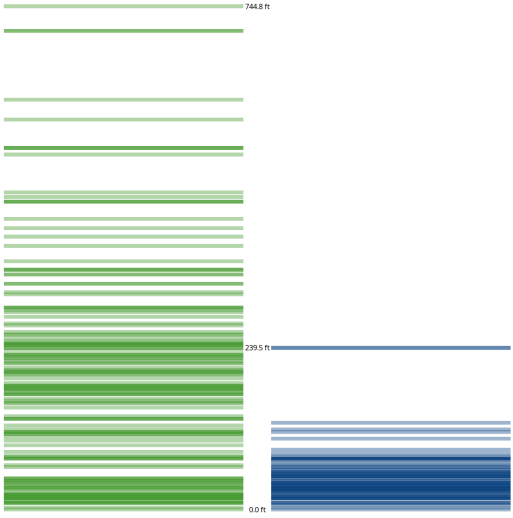
Motivation

Motivation: Determine Home City

- We are given data from 248 homes
- For each home/example, we have these features
  - Elevation
  - Year
  - Bathrooms–Bedrooms
  - Price
  - Square feet
- Goal is to build a program that predicts *SF* or *NY*

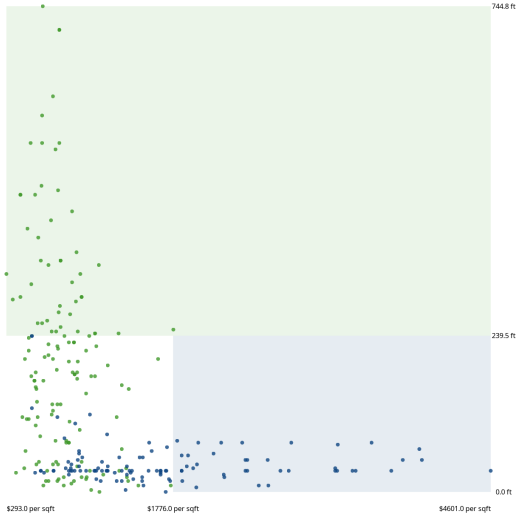
This example and images of it come from: `http://www.r2d3.us/visual-intro-to-machine-learning-part-1`

Elevation

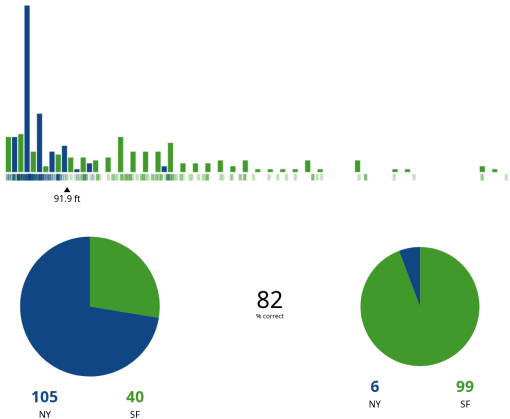


Elevation vs. Price/SqFt

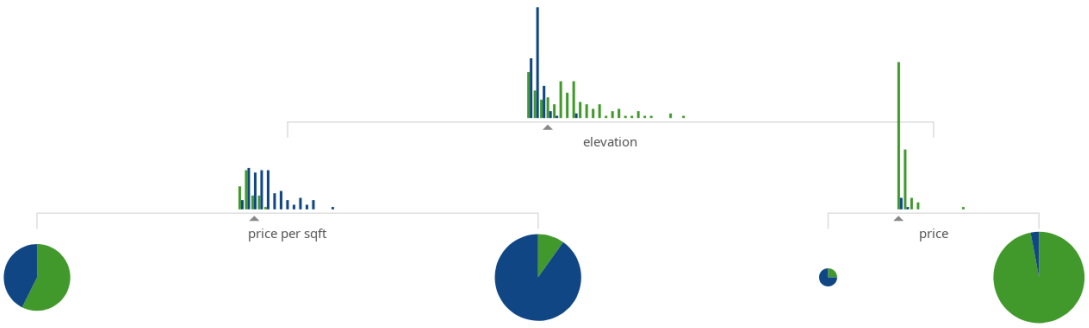
Simple Decision Tree



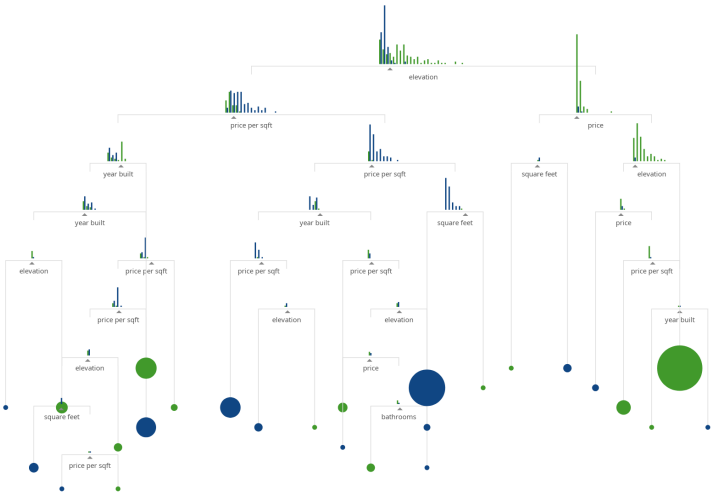
Depth vs. Accuracy



Depth vs. Accuracy



Depth vs. Accuracy



Error

Error

- Eventually, we achieved a perfect classification on the data
  - This is great, right?
- With this decision tree, 'training accuracy' is 1
  - It perfectly labels the data we used to make the tree
- We are now given features for 217 new homes
- What is the 'testing accuracy' on the new data?
  - How does it do on data not used to make the tree?

- Overfitting: lower accuracy on new data
  - Our rules got too specific to our exact training dataset
  - Some of the "deep" splits only use a few examples (bad "coupon collecting")

Supervised Learning

- We are given training data where we know the labels

$$X = \begin{array}{|ccccccc|} \hline \textit{Egg} & \textit{Milk} & \textit{Fish} & \textit{Wheat} & \textit{Shellfish} & \textit{Peanuts} & \dots \\ \hline 0.0 & 0.7 & 0.0 & 0.3 & 0.0 & 0.00 & \dots \\ 0.3 & 0.7 & 0.0 & 0.6 & 0.0 & 0.01 & \dots \\ 0.0 & 0.0 & 0.0 & 0.8 & 0.0 & 0.00 & \dots \\ 0.3 & 0.7 & 1.2 & 0.0 & 0.1 & 0.01 & \dots \\ 0.3 & 0.0 & 1.2 & 0.3 & 0.1 & 0.01 & \dots \\ \hline \end{array} \quad y = \begin{array}{|c|} \hline \textit{Sick?} \\ \hline 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ \hline \end{array}$$

- But there is also testing data we want to label

$$\tilde{X} = \begin{array}{|ccccccc|} \hline \textit{Egg} & \textit{Milk} & \textit{Fish} & \textit{Wheat} & \textit{Shellfish} & \textit{Peanuts} & \dots \\ \hline 0.5 & 0.0 & 1.0 & 0.6 & 2.0 & 1.0 & \dots \\ 0.0 & 0.7 & 0.0 & 1.0 & 0.0 & 0.0 & \dots \\ 3.0 & 1.0 & 0.0 & 0.5 & 0.0 & 0.0 & \dots \\ \hline \end{array} \quad \tilde{y} = \begin{array}{|c|} \hline \textit{Sick?} \\ \hline ? \\ ? \\ ? \\ \hline \end{array}$$

Supervised Learning

- Typical supervised learning steps
  1. Build model based on training data X and y (training phase)
  2. Model makes predictions $\hat{y}$ on test data $\tilde{X}$ (testing phase)
- Instead of training error, consider test error:
  - Are predictions $\hat{y}$ similar to true unseen labels $\tilde{y}$?
- Why we do this?

Goal of Machine Learning

- In machine learning:
  - What we care about is the test error
- Course analogy:
  - The training error is practice on the exam
  - The test error is the actual exam
  - Goal: do well on actual exam, not the practice one
- Memorization vs learning:
  - Can do well on training data by memorizing it
  - You have only learned if you can do well in new situations

Golden Rule of Machine Learning

Golden Rule of Machine Learning

- Even though what we care about is test error:
  - **The test data cannot influence the training phase in any way**
- We're measuring test error to see how well we do on new data:
  - If used during training, does not measure this
  - You can start to overfit if you use it during training
  - Course analogy: you are cheating on the exam

Golden Rule of Machine Learning

- You also shouldn't change the test set to get the result you want
- Note the golden rule applies to hypothesis testing in scientific studies
  - Data that you collect can't influence the hypotheses that you test
- **Extremely common and a major problem**, coming in many forms
  - Collect more data until you coincidentally get significance level you want
  - Try different ways to measure performance, choose the one that looks best
  - Choose a different type of model / hypothesis after looking at the test data
- If you want to modify your hypotheses, you need to test on new data
- Or at least be aware and honest about this issue when reporting results

Golden Rule of Machine Learning

- It is really extremely common:
  - Replication crisis in Science – https://en.wikipedia.org/wiki/Replication_crisis
  - Why Most Published Research Findings are False – https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1182327/
  - False-Positive Psychology: Undisclosed Flexibility in Data Collection and Analysis Allows Presenting Anything as Significant – https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1850704
  - HARKing: Hypothesizing After the Results are Known – http://journals.sagepub.com/doi/abs/10.1207/s15327957pspr0203_4
  - Hack Your Way To Scientific Glory – https://fivethirtyeight.com/features/science-isnt-broken/
  - Psychology's Replication Crisis Has Made The Field Better (some solutions) – https://fivethirtyeight.com/features/psychologys-replication-crisis-has-made-the-field-better/

Is Learning Possible?

- Does training error say anything about test error?
  - In general, NO: Test data might have nothing to do with training data
  - E.g., "adversary" takes training data and flips all labels.
- In order to learn, we need assumptions:
  - The training and test data need to be related in some way
  - Most common assumption: independent and identically distributed (IID)

IID

IID Assumption

- Training/test data is independent and identically distributed (IID) if:
  - All examples come from the same distribution (identically distributed)
  - The example are sampled independently (order does not matter)

Age	Job?	City	Rating	Income
23	Yes	Ham	A	22,000.00
23	Yes	Wel	BBB	21,000.00
22	No	Ham	CC	0.00
25	Yes	Akl	AAA	57,000.00

- Examples in terms of cards – which is IID?
  - Pick a card, put it back in the deck, re-shuffle, repeat
  - Pick a card, put it back in the deck, repeat
  - Pick a card, don't put it back, re-shuffle, repeat.

IID in the Food Allergy Example

- Is the food allergy data IID?
  - Do all the examples come from the same distribution?
  - Does the order of the examples matter?
- No!
  - Being sick might depend on what you ate yesterday (not independent)
  - Your eating habits might have changed over time (not identically distributed)
- What can we do about this?
  - Just ignore that data isn't IID and hope for the best?
  - For each day, maybe add the features from the previous day?
  - Maybe add time as an extra feature?

Learning Theory

- Why does the IID assumption make learning possible?
  - Patterns in training examples are likely to be the same in test examples
- The IID assumption is rarely true:
  - But it is often a good approximation
  - There are other possible assumptions
- Also, we're assuming IID across examples but not across features
- Learning theory explores how training error is related to test error
- We'll look at a simple example, using this notation:
  - E_{train} is the error on training data
  - E_{test} is the error on testing data

Fundamental Trade-Off

Fundamental Trade-Off

- Start with $E_{test} = E_{test}$, then add and subtract E_{train} on the right:

$$\underbrace{E_{test}}_{\text{test error}} = \underbrace{(E_{test} - E_{train})}_{\text{approximation error}} + \underbrace{E_{train}}_{\text{training error}}$$

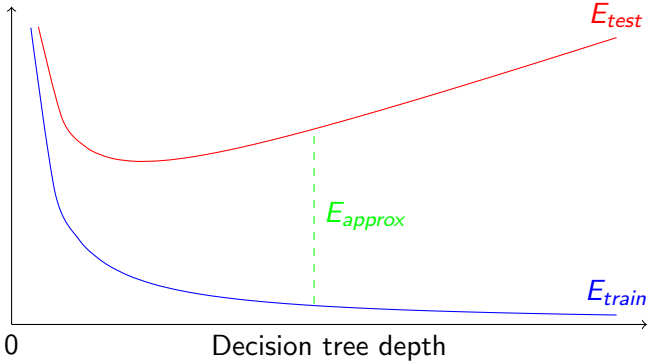
- How does this help?
  - If $E_{approx} = E_{test} - E_{train}$ is small, then E_{train} is a good approximation to E_{test}
- What does E_{approx} ("amount of overfitting") depend on?
  - It tends to get smaller as 'n' gets larger
  - It tends to grow as model get more "complicated"

Fundamental Trade-Off

- This leads to a fundamental trade-off:
  1. E_{train}: how small you can make the training error vs.
  2. E_{approx}: how well training error approximates the test error
- Simple models (like decision stumps):
  - E_{approx} is low (not very sensitive to training set)
  - But E_{train} might be high
- Complex models (like deep decision trees):
  - E_{train} can be low
  - But E_{approx} might be high (very sensitive to training set).

Fundamental Trade-Off

- Training error vs. test error for choosing depth:
  - Training error is high for low depth (underfitting)
  - Training error gets better with depth
  - Test error initially goes down, but eventually increases (overfitting)



Refined Fundamental Trade-Off

- Let E_{best} be the irreducible error (lowest possible error for any model)
  - For example, irreducible error for predicting coin flips is 0.5
- Some learning theory results use E_{best} to futher decompose E_{test}

$$E_{test} = \underbrace{(E_{test} - E_{train})}_{variance} + \underbrace{(E_{train} - E_{best})}_{bias} + \underbrace{E_{best}}_{noise}$$

- This is similar to the bias-variance decomposition
  - Term 1: measure of variance (how sensitive we are to training data)
  - Term 2: measure of bias (how low can we make the training error)
  - Term 3: measure of noise (how low can any model make test error)

Refined Fundamental Trade-Off

- Decision tree with high depth
  - Very likely to fit data well, so bias is low
  - But model changes a lot if you change the data, so variance is high
- Decision tree with low depth
  - Less likely to fit data well, so bias is high
  - But model does not change much you change data, so variance is low
- And degree does not affect irreducible error
  - Irreducible error comes from the best possible model

Bias-Variance Decomposition

- You may have seen bias-variance decomposition
  - Assumes $\tilde{y}_i = \bar{y}_i + \epsilon$, where ϵ has mean 0 and variance σ^2
  - Assumes we have a learner that can take n training examples and use these to make predictions $\hat{y}_i$
- Expected squared test error in this setting is

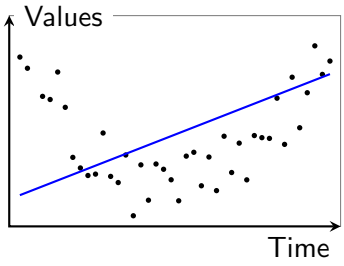
$$\underbrace{\mathbb{E}[(\tilde{y}_i - \hat{y}_i)^2]}_{\text{test squared error}} = \underbrace{\mathbb{E}[(\hat{y}_i - \bar{y}_i)]^2}_{\text{bias}} + \underbrace{(\mathbb{E}[\hat{y}_i^2] - \mathbb{E}[\hat{y}_i]^2)}_{\text{variance}} + \underbrace{\sigma}_{\textit{noise}}$$

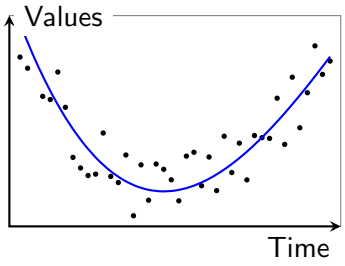
  - Where expectations are taken over possible training sets of n examples
  - Bias is expected error due to having wrong model
  - Variance is expected error due to sensitivity to the training set
  - Noise (irreducible error) is the best we can hope for given the noise (E_{best})

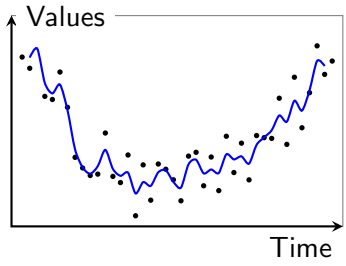
Bias-Variance vs. Fundamental Trade-Off

- Both decompositions serve the same purpose
  - Trying to evaluate how different factors affect test error
- They both lead to the same 3 conclusions
  1. Simple models can have high E_{train} / bias, low E_{approx} / variance
  2. Complex models can have low E_{train} / bias, high E_{approx} / variance
  3. As you increase n, E_{approx} / variance goes down (for fixed complexity)

Another Perspective







Validation Error

Validation Error

- How do we decide decision tree depth?
- We care about test error
- But we can't look at test data
- So what do we do?
- One answer: Use part of the training data to approximate test error
- Split training examples into training set and validation set:
  - Train model based on the training data
  - Test model based on the validation data.

Validation Error

$$X = \left[\begin{array}{ccccccc} 0.0 & 0.7 & 0.0 & 0.3 & 0.0 & 0.00 & \dots \\ 0.3 & 0.7 & 0.0 & 0.6 & 0.0 & 0.01 & \dots \\ 0.0 & 0.0 & 0.0 & 0.8 & 0.0 & 0.00 & \dots \\ \hline 0.3 & 0.7 & 1.2 & 0.0 & 0.1 & 0.01 & \dots \\ 0.3 & 0.0 & 1.2 & 0.3 & 0.1 & 0.01 & \dots \end{array}\right] \quad y = \left[\begin{array}{c} 1 \\ 1 \\ 0 \\ \hline 1 \\ 1 \end{array}\right] \quad \begin{array}{l} \Big\} \text{training} \\ \Big\} \text{validation} \end{array}$$

1. Train: $model = train(X_{train}, Y_{train})$
2. Predict: $\hat{y} = predict(model, X_{validate})$
3. Validate: $error = \sum(\hat{y} \neq y_{validate})$

- Note: if examples are ordered, split should be random

Validation Error

- IID data: validation error is unbiased approximation of test error

$$\mathbb{E}[E_{valid}] = \mathbb{E}[E_{test}]$$

- Course analogy
  - You have 2 practice exams
  - You hide one exam, and spend a lot of time working through the other
  - You then do the other practice exam, to see how well you'll do on the test
- We typically use validation error to choose "hyper-parameters"

Hyperparameters

Notation – Parameters and Hyper-Parameters

- The decision tree rule values are called parameters
  - Parameters control how well we fit a dataset
  - We train a model by trying to find the best parameters on training data
- The decision tree depth is a called a hyper-parameter
  - Hyper-parameters control how complex our model is
  - We can't train a hyper-parameter
    - You can always fit training data better by making the model more complicated
  - We validate a hyper-parameter using a validation score
- (Hyper-parameter is sometimes used for parameters not fit with data)

Choosing Hyper-Parameters with Validation Sets

- So to choose a good value of depth (hyper-parameter), we could?
  - Try a depth-1 decision tree, compute validation error
  - Try a depth-2 decision tree, compute validation error
  - Try a depth-3 decision tree, compute validation error
  - ...
  - Try a depth-20 decision tree, compute validation error
  - Return the depth with the lowest validation error
- After you choose the hyper-parameter, we usually re-train on the full training set with the chosen hyper-parameter

Optimization Bias

Optimization Bias

- Another name for overfitting is optimization bias
  - How biased is an error that we optimized over many possibilities?
- Optimization bias of parameter learning
  - During learning, we could search over tons of different decision trees
  - So we can get lucky and find one with low training error by chance
    - Overfitting of the training error
- Optimization bias of hyper-parameter tuning
  - Here, we might optimize the validation error over 20 values of depth
  - One of the 20 trees might have low validation error by chance
    - Overfitting of the validation error

Example of Optimization Bias

- Consider a multiple-choice (a, b, c, d) test with 10 questions
  - If you choose answers randomly, expected grade is 25% (no bias)
  - If you fill out two tests randomly and pick the best, expected grade is 33%
    - Optimization bias of $\sim$8%
  - If you take the best among 10 random tests, expected grade is $\sim$47%
  - If you take the best among 100, expected grade is $\sim$62%
  - If you take the best among 1000, expected grade is $\sim$73%
  - If you take the best among 10000, expected grade is $\sim$82%
    - You have so many chances that you expect to do well
- But on new questions the random choice accuracy is still 25%

Factors Affecting Optimization Bias

- If we instead used a 100-question test then
  - Expected grade from best over 1 randomly-filled test is 25%
  - Expected grade from best over 2 randomly-filled test is $\sim$27%
  - Expected grade from best over 10 randomly-filled test is $\sim$32%
  - Expected grade from best over 100 randomly-filled test is $\sim$36%
  - Expected grade from best over 1000 randomly-filled test is $\sim$40%
  - Expected grade from best over 10000 randomly-filled test is $\sim$47%
- The optimization bias grows with the number of things we try
  - Complexity of the set of models we search over
- But, optimization bias shrinks quickly with the number of examples
  - But it's still non-zero and growing if you over-use your validation set

Overfitting to the Validation Set?

- Validation error usually has lower optimization bias than training error
  - Might optimize over 20 values of depth, instead of millions+ of possible trees
- But we can still overfit to the validation error (common in practice)
  - Validation error is only an unbiased approximation if you use it once
  - Once you start optimizing it, you start to overfit to the validation set
- This is most important when the validation set is small
  - The optimization bias decreases as the number of validation examples increases
- Remember, our goal is still to do well on the test set (new data), not the validation set (where we already know the labels)

Validation Error and Optimization Bias

- Optimization bias is small if you only compare a few models
  - Best decision tree on the training set among depths 1, 2, 3,..., 10
  - Risk of overfitting to validation set is low if we try 10 things
- Optimization bias is large if you compare a lot of models
  - All possible decision trees of depth 10 or less
  - Here we are using the validation set to pick between a billion+ models
    - Risk of overfitting to validation set is high: could have low validation error by chance
  - If you did this, you might want a second validation set to detect overfitting
- And optimization bias shrinks as you grow size of validation set.

Side note: Optimization Bias leads to Publication Bias

- Suppose that 20 researchers perform the exact same experiment
- They each test whether their effect is significant ($p < 0.05$)
  - 19/20 find that it is not significant
  - But the 1 group finding it is significant publishes a paper about the effect
- This is again optimization bias, contributing to publication bias
  - A contributing factor to many reported effects being wrong

Cross-Validation

Cross-Validation (CV)

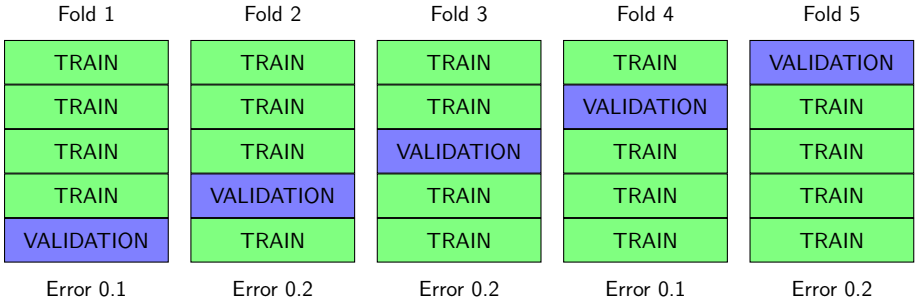
- Isn't it wasteful to only use part of your data?
- 5-fold cross-validation
  - Train on 80% of the data, validate on the other 20%
  - Repeat this 5 more times with different splits, and average the score

$$X = \begin{bmatrix} \dots \\ \hline \dots \\ \hline \dots \\ \hline \dots \\ \hline \dots \end{bmatrix} \quad y = \begin{bmatrix} \dots \\ \hline \dots \\ \hline \dots \\ \hline \dots \\ \hline \dots \end{bmatrix} \begin{matrix} \} \text{ fold 1} \\ \} \text{ fold 2} \\ \} \text{ fold 3} \\ \} \text{ fold 4} \\ \} \text{ fold 5} \end{matrix}$$

1. Train on folds $1, 2, 3, 4$, compute error on fold 5
2. Train on folds $1, 2, 3, 5$, compute error on fold 4
3. Train on folds $1, 2, 4, 5$, compute error on fold 3

   ...
6. Take average of the 5 errors as approximation of test error

Cross-Validation (CV)



- CV error estimate for this hyper-parameter $mean(errors) = 0.16$

Cross-Validation (CV)

- You can take this idea further (k-fold cross-validation)
  - 10-fold cross-validation: train on 90% of data and validate on 10%
    - Repeat 10 times and average (test on fold 1, then fold 2,..., then fold 10)
  - Leave-one-out cross-validation: train on all but one training example
    - Repeat n times and average
- Gets more accurate but more expensive with more folds
  - To choose depth we compute the cross-validation score for each depth
- As before, if data is ordered then folds should be random splits
  - Randomize first, then split into fixed folds
- Usually used in classification: stratified cross-validation
  - This enforces that the class distribution in all folds is approximately the same as in the full data set

Classifier Evaluation Metrics

Confusion Matrix

- Given m classes, an entry, $CM_{i,j}$ in a confusion matrix indicates number of tuples in class i that were labeled by the classifier as class j

		Predicted Class	
		C_1	$\neg C_1$
Actual Class	C_1	True positives (TP)	False negatives (FN)
	$\neg C_1$	False positives (FP)	True negative (TN)

- FP is often called type I error – reject the true null hypothesis
- FN is called type II error – reject false null hypothesis

Classifier Evaluation Metrics: Accuracy, Precision and Recall

- Classifier accuracy, or recognition rate
  - Percentage of test set tuples that are correctly classified

$$Accuracy = (TP + TN)/All$$

- Precision: Exactness: what % of tuples that the classifier labeled as positive are actually positive?

$$P = Precision = \frac{TP}{TP + FP}$$

- Recall: Completeness: what % of positive tuples did the classifier label as positive?

$$R = Recall = \frac{TP}{TP + FN}$$

- There is an inverse relationship between precision & recall

Classifier Evaluation Metrics: F-measure

- $F - measure$ (or $F - score$): harmonic mean of precision and recall

$$F_b = (1 + \beta^2) \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

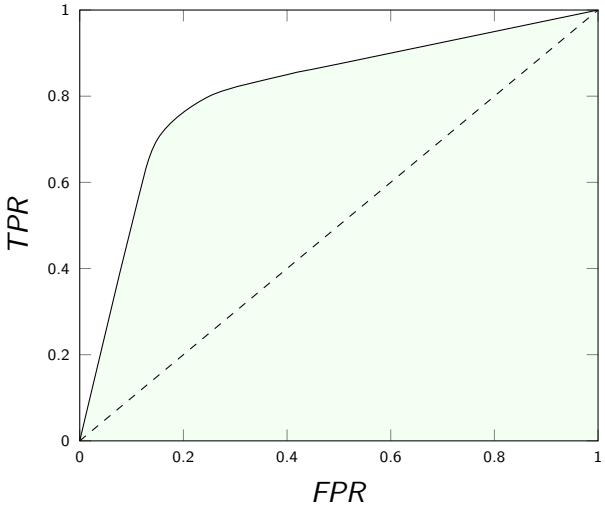
  - In general, it is the weighted measure of precision & recall
  - β is a weight – assign β times as much weight to recall as to precision
  - Most commonly used: $F_1 - measure$ with $\beta = 1$

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

ROC Curves

- ROC (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between true positive rate $TPR = \frac{TP}{TP+FN}$ and false positive rate $FPR = \frac{FP}{FP+TN}$
- The area under the ROC curve (AUC: Area Under Curve, also AUROC) is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- The closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model

ROC curves



Comparing Classifiers

Comparing Classifiers – Which one is better?

- Suppose we have 2 classifiers, M_1 and M_2, which one is better?
- Use 10-fold cross-validation to obtain error (accuracy, AUROC, ...)
- These mean error rates are just estimates of error on the true population of future data cases
- What if the difference between the 2 error rates is just attributed to chance?
  - Use a test of statistical significance

Estimating Confidence Intervals: Null Hypothesis

- Perform 10-fold cross-validation
- Assume samples follow a t distribution with k–1 degrees of freedom (here, k=10)
- Use t-test (or Student's t-test)
- Null Hypothesis: M_1 & M_2 are the same
- If we can reject null hypothesis, then
  - we conclude that the difference between M_1 & M_2 is statistically significant
  - Chose model with lower error rate

Estimating Confidence Intervals: t-test

- If only 1 test set available: pairwise comparison
  - For ith round of 10-fold cross-validation, the same cross partitioning is used to obtain $err(M_1)_i$ and $err(M_2)_i$
  - Average over (at least!) 10 rounds to get $\overline{err}(M_1)$ and $\overline{err}(M_2)$
  - t-test computes t-statistic with k-1 degrees of freedom:

$$t = \frac{\overline{err}(M_1) - \overline{err}(M_2)}{\sqrt{var(M_1 - M_2)/k}},$$

where

$$var(M_1 - M_2) = \frac{1}{k} \sum_{i=1}^{k} [err(M_1)_i - err(M_2)_i - (\overline{err}(M_1) - \overline{err}(M_2))]^2$$

Estimating Confidence Intervals: Statistical Significance

- Are M_1 & M_2 significantly different?
- Compute t. Select significance level (e.g. sig = 5%)
- Consult table for t-distribution: Find t value corresponding to k-1 degrees of freedom (here, 9)
- t-distribution is symmetric: typically upper % points of distribution shown → look up value for confidence limit $z = sig/2$ (here, 0.025)
- If $t > z$ or $t < -z$, then t value lies in rejection region:
  - Reject null hypothesis that mean error rates of M_1 & M_2 are same
  - Conclude: statistically significant difference between M_1 & M_2
- Otherwise, conclude that any difference is chance

Multiple Comparisons

- Beyond paired comparison: Nemenyi test, Friedman test, Critical Distance plots
- Janez Demšar: “Statistical Comparisons of Classifiers over Multiple Data Sets” – see Assignment 1

The Best Machine Learning Model

The Best Machine Learning Model

- Decision trees are not always most accurate on test error
- What is the best machine learning model?
- An alternative measure of performance is the generalization error
  - Average error over all x_i vectors that are not seen in the training set
  - How well we expect to do for a completely unseen feature vector
- No free lunch theorem
  - There is no best model achieving the best generalization error for every problem
  - If model A generalizes better to new data than model B on one dataset, there is another dataset where model B works better
- This question is like asking which is best among rock, paper, and scissors

The Best Machine Learning Model

- Implications of the lack of a best model
  - We need to learn about and try out multiple models
- So which ones to study?
  - We'll usually motivate each method by a specific application
  - But we're focusing on models that have been effective in many applications
- Caveat of no free lunch (NFL) theorem
  - The world is very structured
  - Some datasets are more likely than others
  - Model A really could be better than model B on every real dataset in practice
- Machine learning research
  - Large focus on models that are useful across many applications.

Examples

Should you trust them?

- Scenario 1
  - I built a model based on the data you gave me
  - It classified your data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Should you trust them?
- Probably not:
  - They are reporting training error
  - This might have nothing to do with test error
  - E.g., they could have fit a very deep decision tree.
- Why probably?
  - If they only tried a few very simple models, the 98% might be reliable
  - E.g., they only considered decision stumps with simple 1-variable rules

Should you trust them?

- Scenario 2
  - I built a model based on half of the data you gave me
  - It classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the validation error once
  - This is an unbiased approximation of the test error
  - Trust them if you believe they did not violate the golden rule

Should you trust them?

- Scenario 3
  - I built 10 models based on half of the data you gave me
  - One of them classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the validation error a small number of times
  - Maximizing over these errors is a biased approximation of test error
  - But they only maximized it over 10 models, so bias is probably small
  - They probably know about the golden rule.

Should you trust them?

- Scenario 4
  - I built 1 billion models based on half of the data you gave me
  - One of them classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably not
  - They computed the validation error a huge number of times
  - They tried so many models, one of them is likely to work by chance
- Why probably?
- If the 1 billion models were all extremely-simple, 98% might be reliable.

Should you trust them?

- Scenario 5
  - I built 1 billion models based on the first third of the data you gave me
  - One of them classified the second third of the data with 98% accuracy
  - It also classified the last third of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the first validation error a huge number of times
  - But they had a second validation set that they only looked at once
  - The second validation set gives unbiased test error approximation
  - This is ideal, as long as they didn't violate golden rule on the last third
  - And assuming you are using IID data in the first place.

Examples

Should you trust them?

- Scenario 1
  - I built a model based on the data you gave me
  - It classified your data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Should you trust them?
- Probably not:
  - They are reporting training error
  - This might have nothing to do with test error
  - E.g., they could have fit a very deep decision tree.
- Why probably?
  - If they only tried a few very simple models, the 98% might be reliable
  - E.g., they only considered decision stumps with simple 1-variable rules

Should you trust them?

- Scenario 2
  - I built a model based on half of the data you gave me
  - It classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the validation error once
  - This is an unbiased approximation of the test error
  - Trust them if you believe they did not violate the golden rule

Should you trust them?

- Scenario 3
  - I built 10 models based on half of the data you gave me
  - One of them classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the validation error a small number of times
  - Maximizing over these errors is a biased approximation of test error
  - But they only maximized it over 10 models, so bias is probably small
  - They probably know about the golden rule.

Should you trust them?

- Scenario 4
  - I built 1 billion models based on half of the data you gave me
  - One of them classified the other half of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably not
  - They computed the validation error a huge number of times
  - They tried so many models, one of them is likely to work by chance
- Why probably?
- If the 1 billion models were all extremely-simple, 98% might be reliable.

Should you trust them?

- Scenario 5
  - I built 1 billion models based on the first third of the data you gave me
  - One of them classified the second third of the data with 98% accuracy
  - It also classified the last third of the data with 98% accuracy
  - It should get 98% accuracy on the rest of your data
- Probably
  - They computed the first validation error a huge number of times
  - But they had a second validation set that they only looked at once
  - The second validation set gives unbiased test error approximation
  - This is ideal, as long as they didn't violate golden rule on the last third
  - And assuming you are using IID data in the first place.

Summary

- Training error vs. testing error
  - What we care about in machine learning is the testing error
- Golden rule of machine learning
  - The test data cannot influence training the model in any way
- Independent and identically distributed (IID)
  - One assumption that makes learning possible
- Fundamental trade-off:
  - Trade-off between getting low training error and having training error approximate test error
- Validation set:
  - We can save part of our training data to approximate test error
- Hyper-parameters
  - Parameters that control model complexity, typically set with a validation set
- Cross-validation: allows better use of data to estimate test error
- No free lunch theorem: there is no best ML model
- (In CS760, we talk about what happens if IID is too much violated and how this is an issue – Adversarial Learning)

Literature

- Machine Learning – Tom Mitchell
- Pattern Recognition and Machine Learning – Christopher Bishop
- Data Mining – Jiawei Han, Micheline Kamber, Jian Pei
- Data Mining – Ian Witten, Eibe Frank, Mark Hall, Christopher Pal

SCIENCE
SCHOOL OF COMPUTER SCIENCE
MACHINE LEARNING

Thank you for your attention!

https://ml.auckland.ac.nz