CSC 480/580 Principles of Machine Learning

05 Practical Considerations

Jason Pacheco

Department of Computer Science

Outline

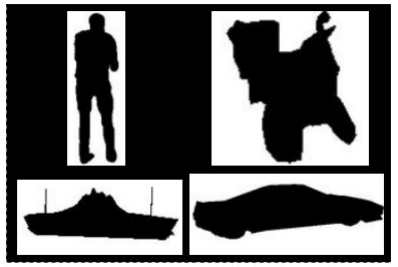
- The role of features in supervised learning
- Classification metrics beyond error rate
- Model Evaluation & Comparison
- Debugging Learning Algorithms
- Bias / Variance Tradeoff

The role of features in supervised learning

The importance of good feature representation

- Pixel representation:
  - represent an image as a w*h*3 dimensional vector
  - treat all coordinates as the same role
  - throw away all locality information in the image
  - hard to classify (even for human)!

- Shape representation:
  - represent a colored image with a w*h black-white image

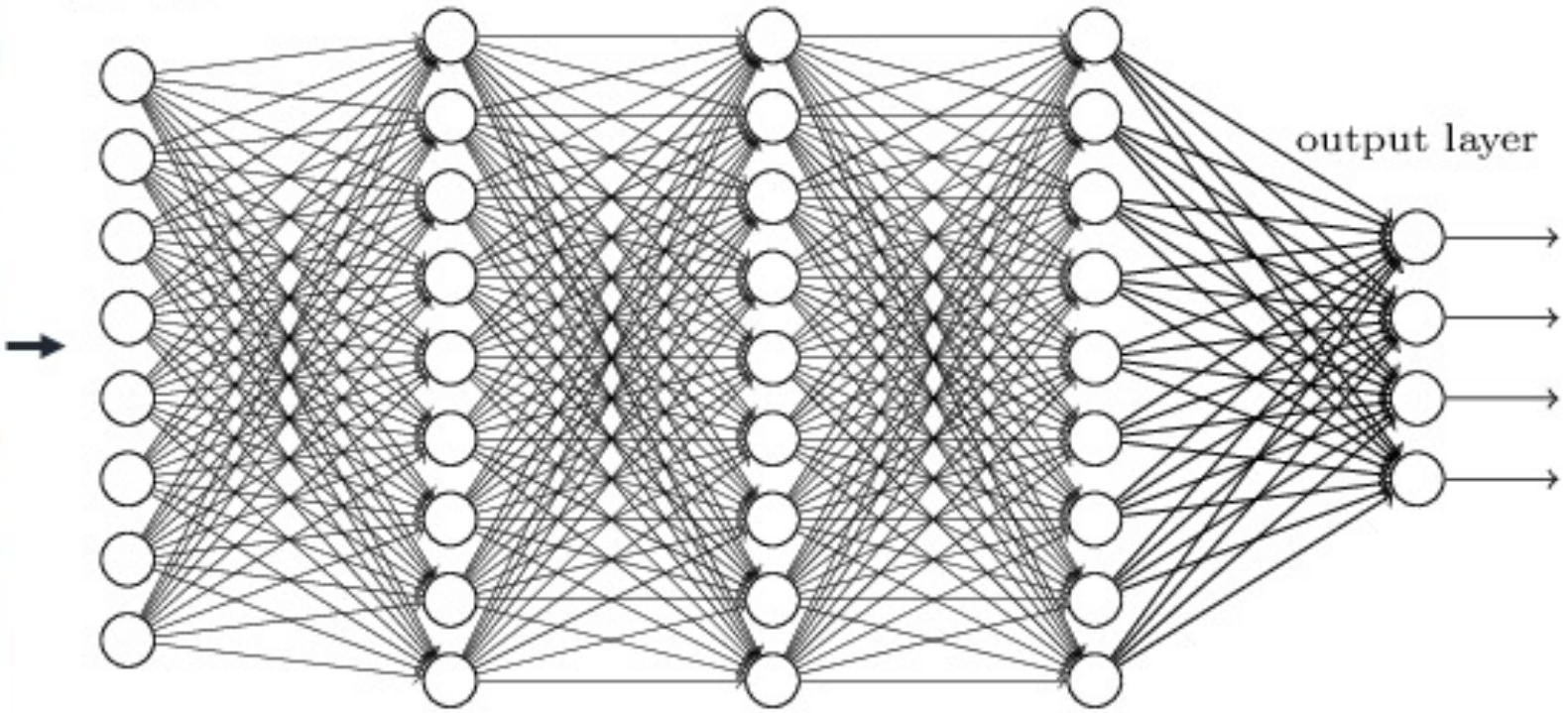


- Bag-of-words representation:

	free	offer	lecture	cs	Spam?
Email 1	2	1	0	0	+1
Email 2	0	1	3	1	-1

Deep neural networks learn hierarchical feature representations

hidden layer 1 hidden layer 2 hidden layer 3



https://www.rsipvision.com/exploring-deep-learning/

Irrelevant and redundant features

- Irrelevant features
  - y is independent of f
  - y = Road walkability, f = duck swimming in the pond

  - If #features is large and #examples is small ⇒ spurious correlation between some feature & label

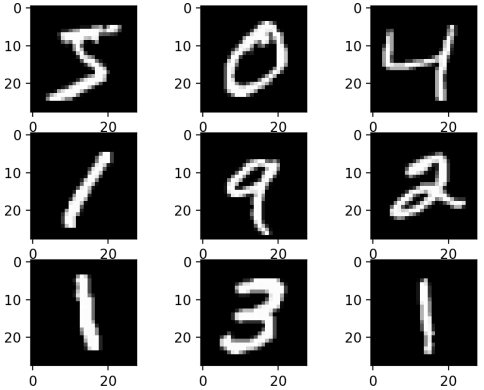
- Redundant features
  - Given f_1, y is (nearly) independent of f_2
  - E.g. detecting fire hydrants -- $f_1 = \text{pixel}_{(20,93)}, f_2 = \text{pixel}_{(21,93)}$

- Learning decision trees implicitly handles these two issues
- How about nearest neighbors / Perceptron?

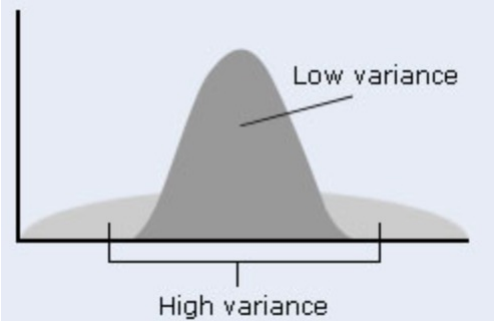
https://www.microsoft.com/en-us/research/blog/ppe-a-fast-and-provably-efficient-rl-algorithm-for-exogenous-noise/

Feature pruning

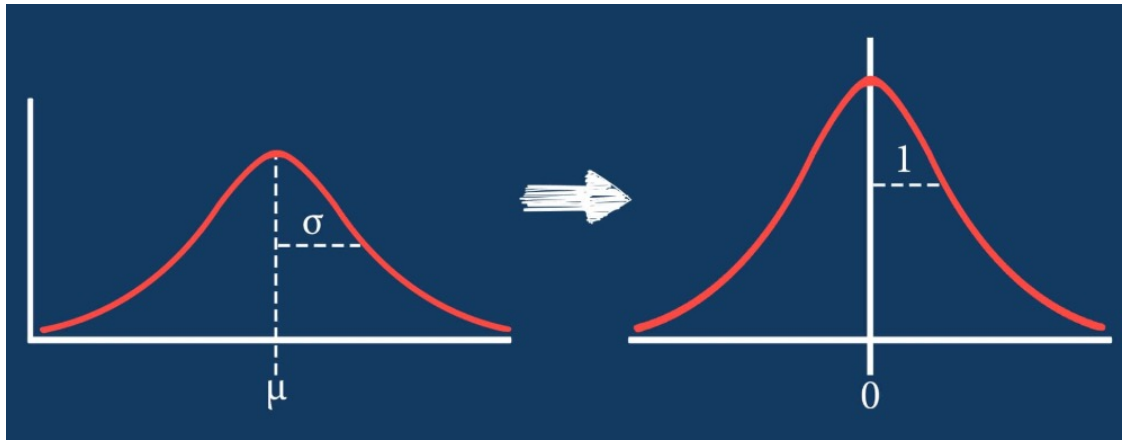
- Removing features that are not very useful for prediction
  - E.g. text classification with bag-of-word representation, remove words that appear in <= K docs
  - E.g. digit classification, remove pixels with low variance



$$\mu_f = \frac{1}{N}\sum_{i=1}^{N} x_{i,f} \qquad \sigma_f^2 = \frac{1}{N}\sum_{i=1}^{N}(x_{i,f} - \mu_f)^2$$



Feature normalization



- Centering:
  - $x'_{i,f} = x_{i,f} - \mu_f \Rightarrow \mu'_f = 0$
- Variance scaling:
  - $x'_{i,f} = x_{i,f}/\sigma_f \Rightarrow (\sigma'_f)^2 = 1$
- Absolute scaling
  - $x'_{i,f} = x_{i,f}/r_f$, where $r_f = \max_i |x_{i,f}|$ $\Rightarrow$ range of $x'_{i,f}$ in [-1,+1]

- Same transformation applied to both training set and test data
- Example normalization: $x'_i = \frac{x_i}{\|x_i\|}$ sometimes also can be applied

Classification metrics beyond error rate

Confusion matrix

- E.g. activity recognition

- $P(\hat{y} = \text{skip} \mid y = \text{jump}) = 11\%$

task: activity recognition from video

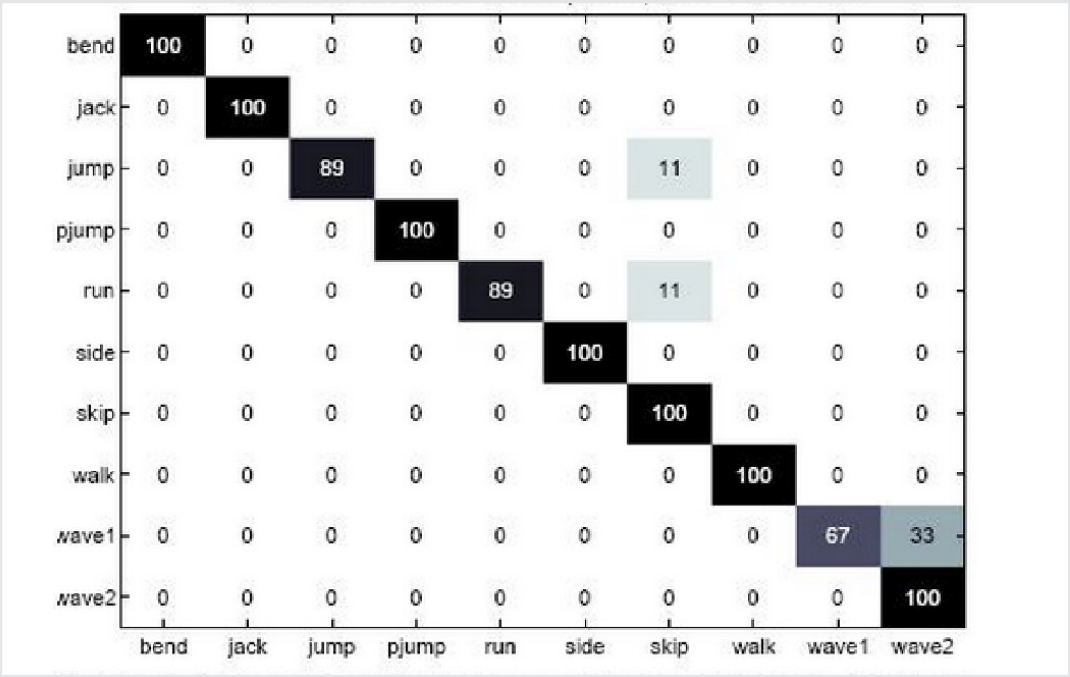


figure from vision.jhu.edu

Class imbalance problem

- E.g., 5% positive, 95% negative
- Implicit assumption:

  misclassifying positive example is more costly than misclassifying negative examples

- Standard ML algorithm aims to find h that minimizes unweighted training error $\sum_{i=1}^{n} I(h(x_i) \neq y_i)$ **I: indicator function; =1 if the predicate is true, =0 otherwise**
  - Tend to produce classifiers that always predict negative

- Mitigation: Duplicate the examples in the minority class to make dataset balanced

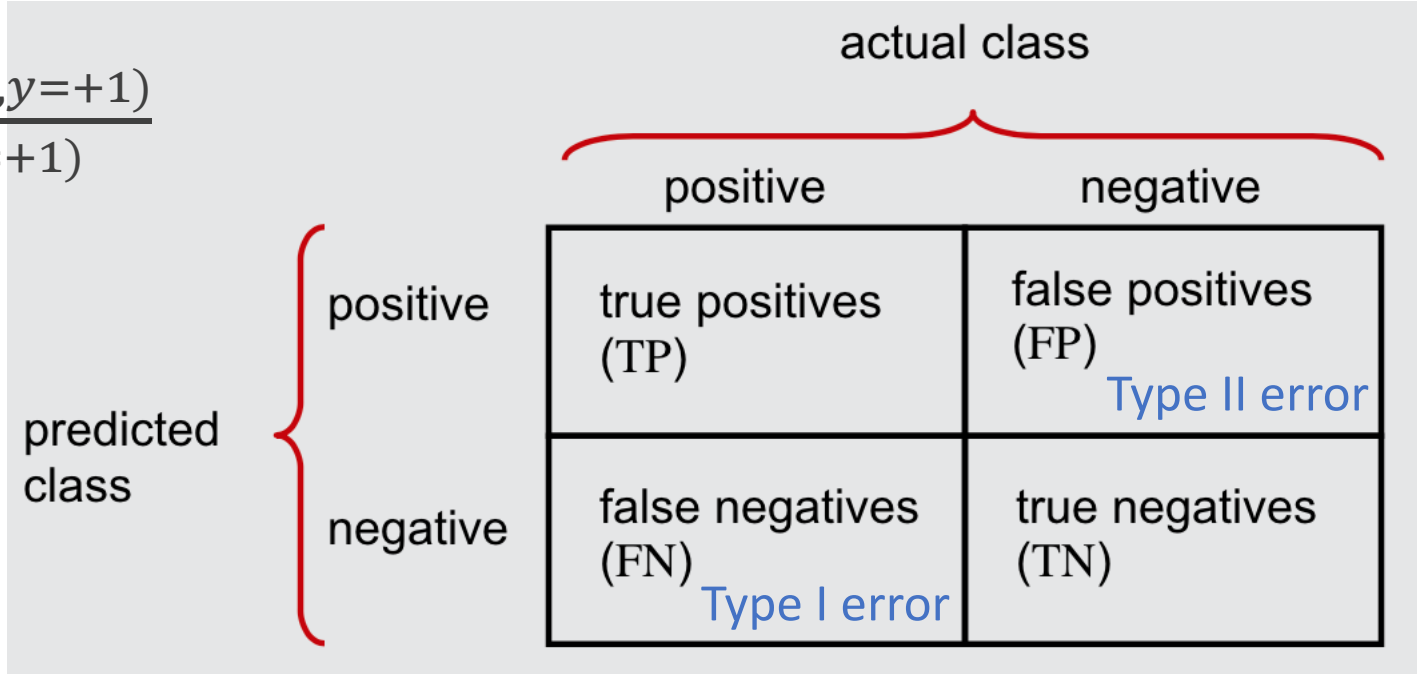
  repeat every positive example w times, where $w = P(y = -1)/P(y = +1)$

New measures of classification performance

- True positive rate (TPR) = $\frac{\mathrm{TP}}{\mathrm{P}} = \frac{P(\hat{y}=+1, y=+1)}{P(y=+1)}$

  (aka recall, sensitivity)
- True negative rate (TNR) = $\frac{\mathrm{TN}}{\mathrm{N}}$

  (specificity)
- False positive rate (FPR) = $\frac{\mathrm{FP}}{\mathrm{N}}$
- False negative rate (FNR) = $\frac{\mathrm{FN}}{\mathrm{P}}$



P = TP + FN N = FP + TN

Applications:

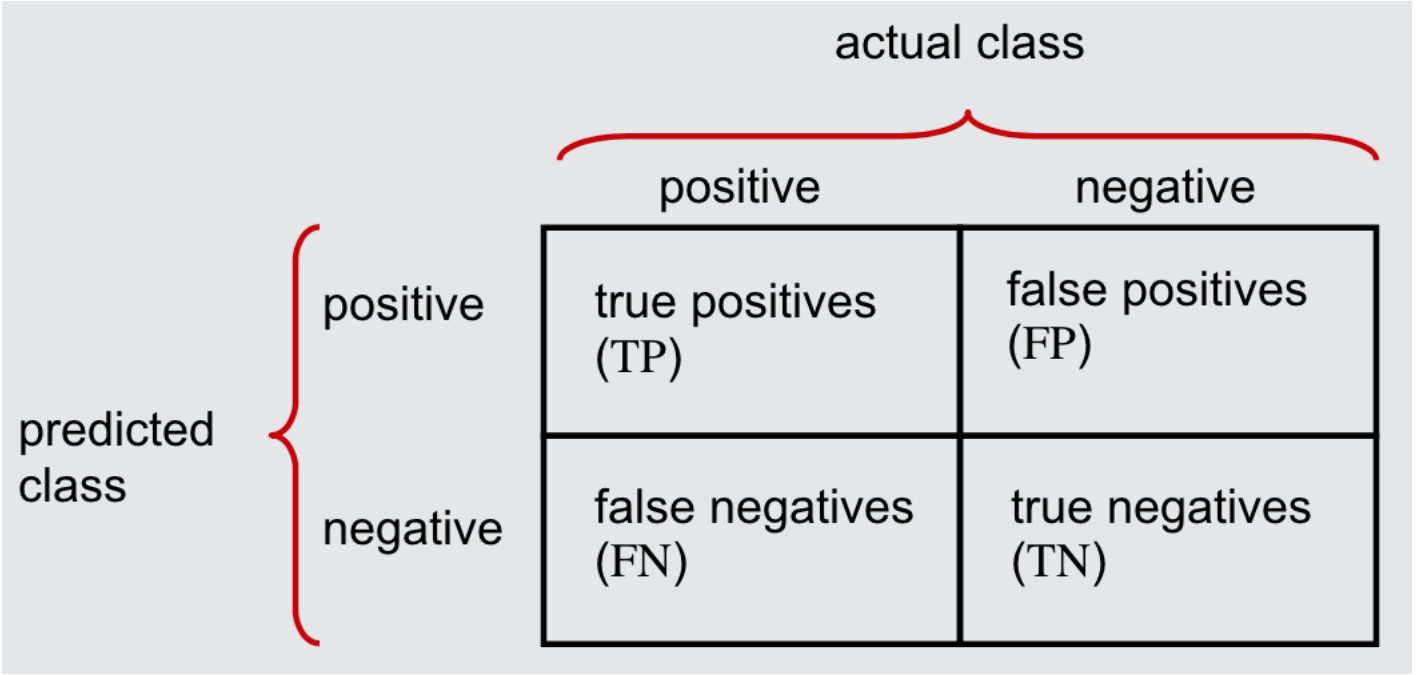
- Search engine: precision & recall
- Cancer classification: FNR vs. FPR

- Precision = $\frac{\mathrm{TP}}{\mathrm{P}-called} = \frac{P(\hat{y}=+1, y=+1)}{P(\hat{y}=+1)}$, $\mathrm{P} - called = \mathrm{TP} + \mathrm{FP}$

Adjusting TP, FP, TN, FN via thresholding

- Classification scores on test examples

$c(x_i)$	y_i
.99	+
.98	+
.72	-
.51	-
.24	+



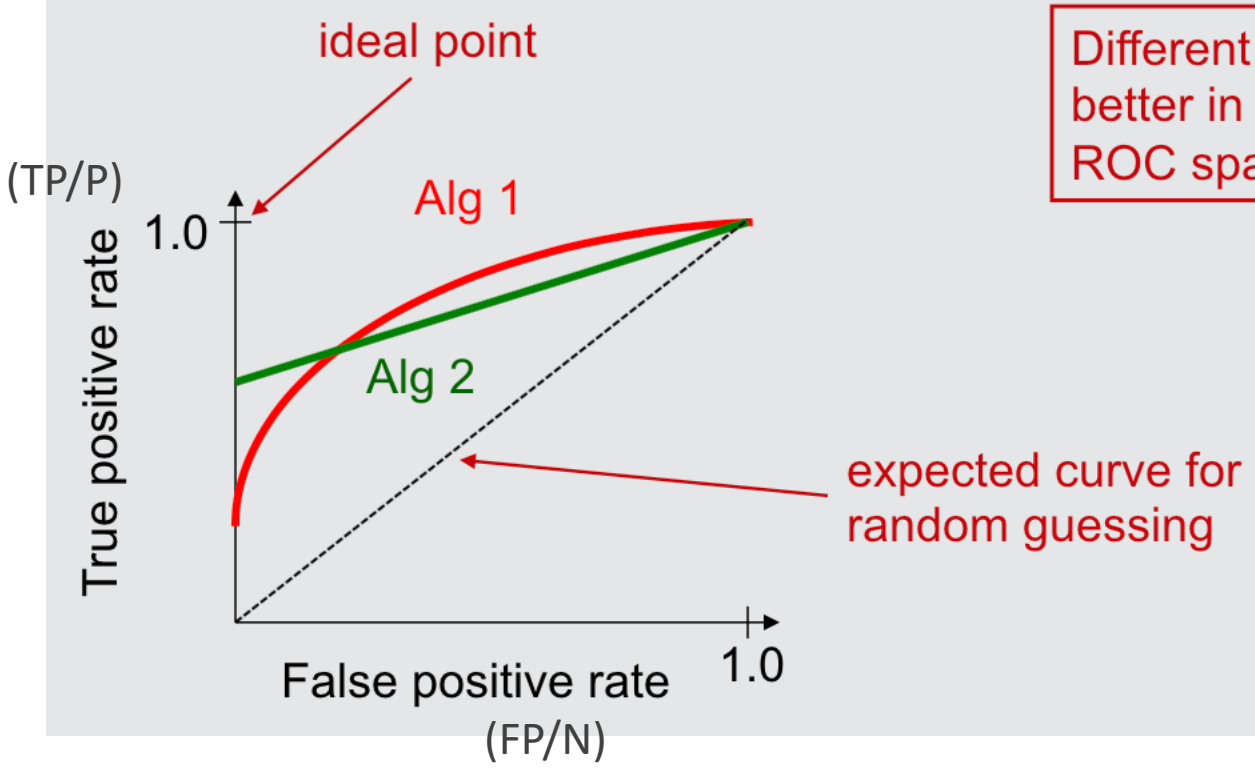
P = TP + FN N = FP + FN

- Induce classifier $h_t(x) = \begin{cases} +1, & c(x) > t \\ -1, & c(x) \leq t \end{cases}$
- Choice of threshold t:
  - $t = +\infty$: $h_t \equiv -1 \Rightarrow \mathrm{TPR} = 0,\ \mathrm{FPR} = 0$
  - $t = 0$: $h_t \equiv +1 \Rightarrow \mathrm{TPR} = 1,\ \mathrm{FPR} = 1$

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}}, \quad \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}}$$

ROC curve

A *Receiver Operating Characteristic* (*ROC*) curve plots the TP-rate vs. the FP-rate as a threshold on the confidence of an instance being positive is varied



Different methods can work better in different parts of ROC space.

	$c(x_i)$	y_i
	.99	+
	.98	+
	.72	+
	.51	-
	.24	-

ROC curve

$$TPR = \frac{TP}{P}, \quad FPR = \frac{FP}{N}$$

Conceptually, consider every possible threshold, put a dot for each, and connect them.

Walking down a consecutive + segment => FPR stays, TPR increases (vertical)

Walking down a consecutive – segment => TPR stays, FPR increases (horizontal)

- results in staircase shape
- Popular alternative: just plot when going from + to -.

instance	confidence positive		correct class
Ex 9	.99		+
Ex 7	.98	TPR= 2/5, FPR= 0/5	+
Ex 1	.72		-
Ex 2	.70		+
Ex 6	.65	TPR= 4/5, FPR= 1/5	+
Ex 10	.51		-
Ex 3	.39		-
Ex 5	.24	TPR= 5/5, FPR= 3/5	+
Ex 4	.11		-
Ex 8	.01	TPR= 5/5, FPR= 5/5	-

Calculating ROC curve

instance	confidence positive		correct class
Ex 9	.99		+
Ex 7	.98	TPR= 2/5, FPR= 0/5	+
Ex 1	.72		-
Ex 2	.70		+
Ex 6	.65	TPR= 4/5, FPR= 1/5	+
Ex 10	.51		-
Ex 3	.39		-

in descending order in c

let $\left(\left(y^{(1)}, c^{(1)}\right) \ldots \left(y^{(m)}, c^{(m)}\right)\right)$ be the test-set instances sorted according to predicted confidence $c^{(i)}$ that each instance is positive

let num_neg, num_pos be the number of negative/positive instances in the test set

$TP = 0,\ FP = 0$

$last\_TP = 0$

for $i = 1$ to m

// find thresholds where there is a pos instance on high side, neg instance on low side

if $(i > 1)$ and $(c^{(i)} \neq c^{(i-1)})$ and $(y^{(i)} ==$ neg $)$ and $(TP > last\_TP)$

$FPR = FP / num\_neg,\quad TPR = TP / num\_pos$

output (FPR, TPR) coordinate

$last\_TP = TP$

if $y^{(i)} ==$ pos

$++TP$

else

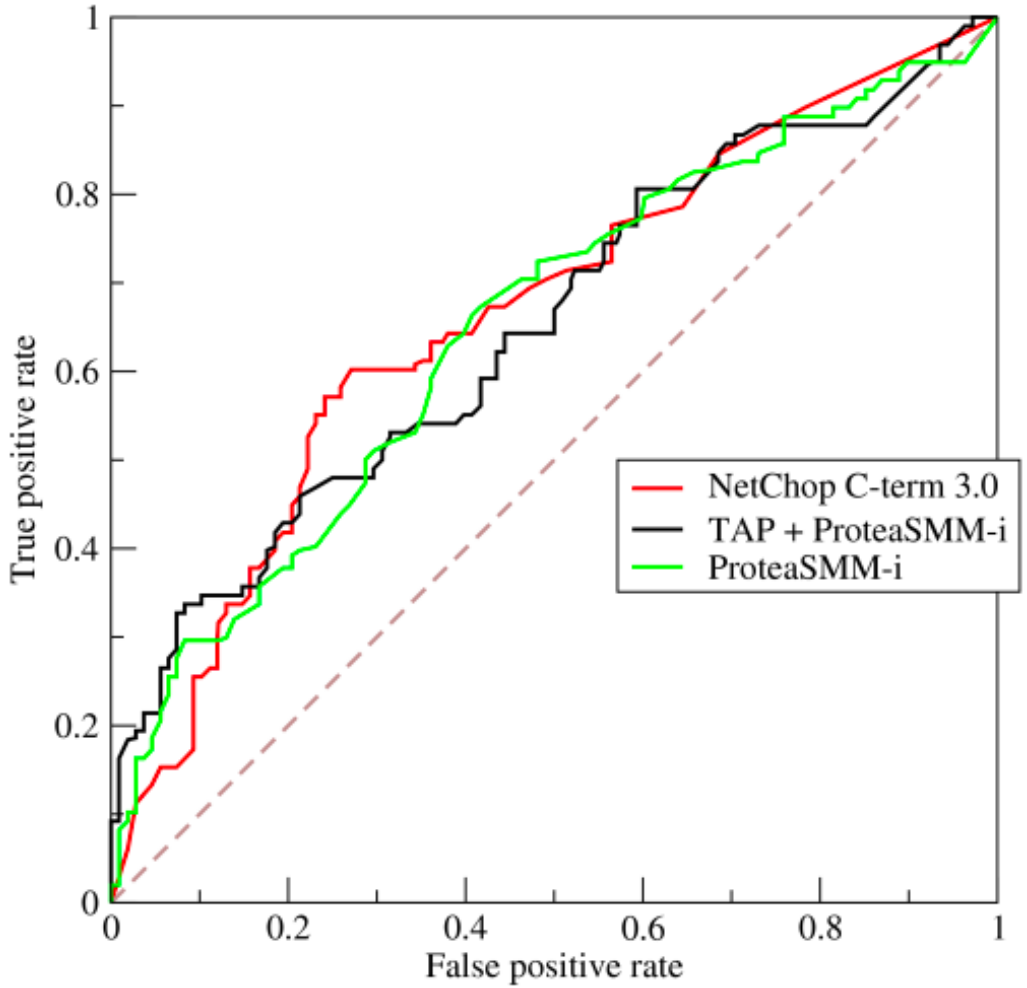
$++FP$

bookkeeping TP and FP

$FPR = FP / num\_neg,\ TPR = TP / num\_pos$

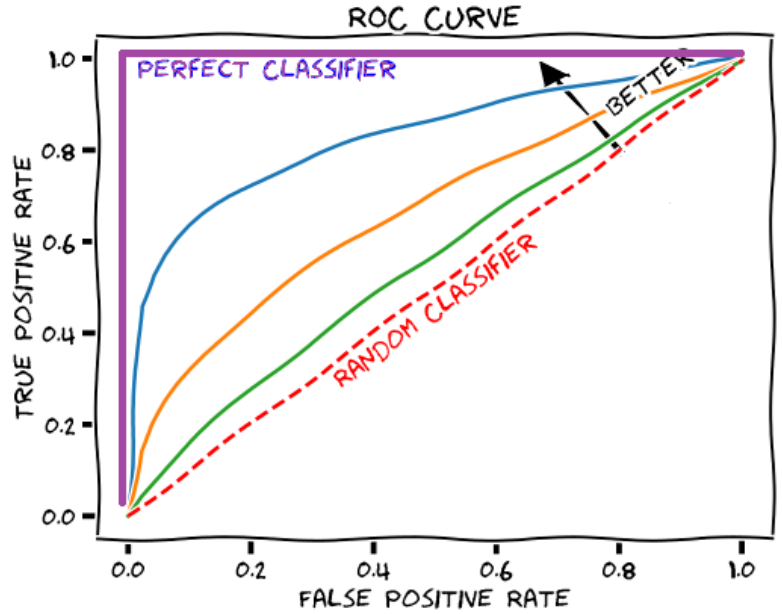
output (FPR, TPR) coordinate

ROC curve examples



from Wikipedia

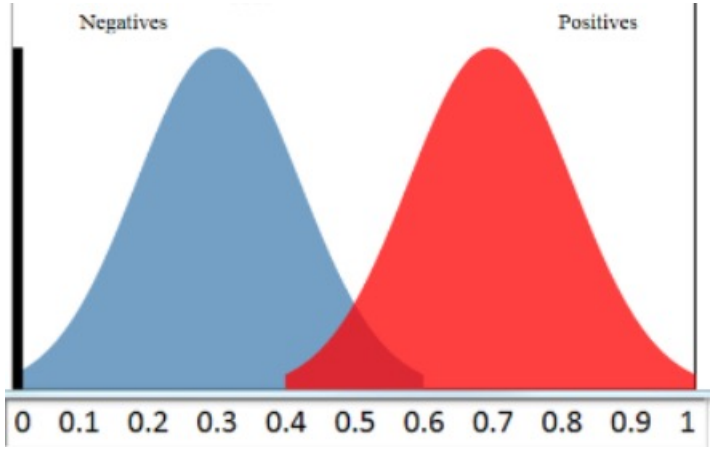
Area under ROC curve



- The boss says “could you just give me one number?”
- **AUC**: Area Under the ROC curve
  - Perfect scorer: AUC = 1
  - Random scorer: AUC = 0.5

- In-class question: Why does the perfect scoring c have a ROC that is ‘Γ’ shaped ?
  - What would a ``worst-case’’ ROC look like?

- Interpretation: “how well does c distinguish between + and -?”



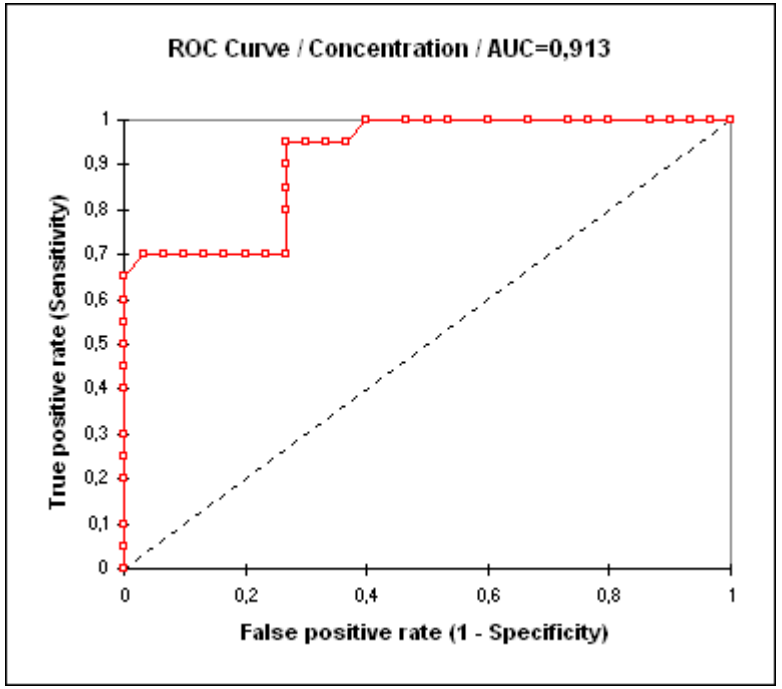
https://glassboxmedicine.com/2019/02/23/measuring-performance-auc-auroc/

Area under ROC curve

- **Fact:** AUC of scorer c has the following formula:

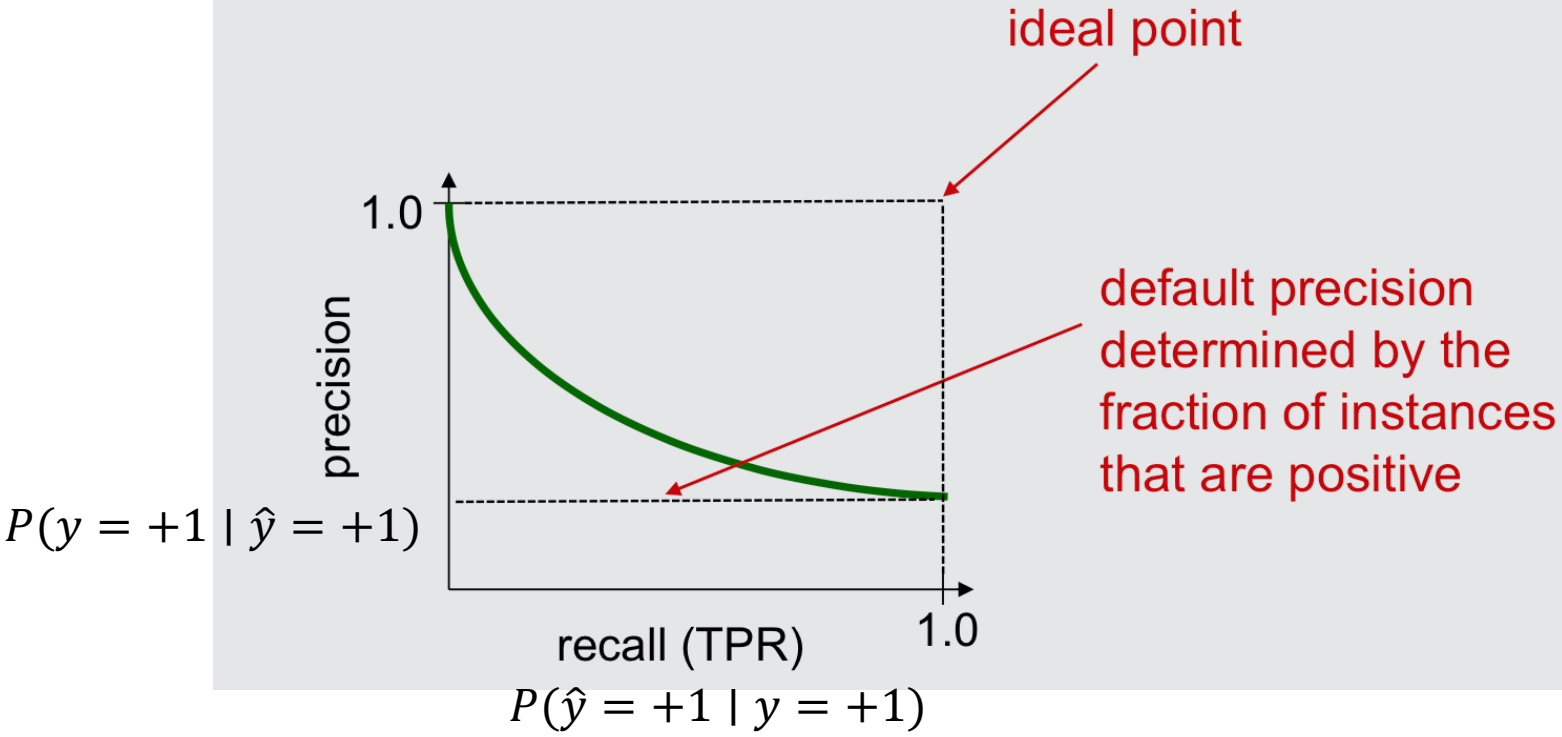
$$\mathrm{AUC}(c) = \frac{\sum_{(x_-,-1)\in S_-} \sum_{(x_+,+1)\in S_+} I(c(x_+) > c(x_-))}{N_- \cdot N_+}$$

Fraction of test example pairs whose scores' ordering contradict with their labels



Precision-Recall (PR) curve

A *precision/recall curve* plots the precision vs. recall (TP-rate) as a threshold on the confidence of an instance being positive is varied



$c(x_i)$	y_i
.99	+
.98	+
.72	+
.51	-
.24	-

- This is *usually* a trade-off curve (not always): $t \downarrow \Rightarrow$ recall $\uparrow$, precision usually $\downarrow$

PR-curve example

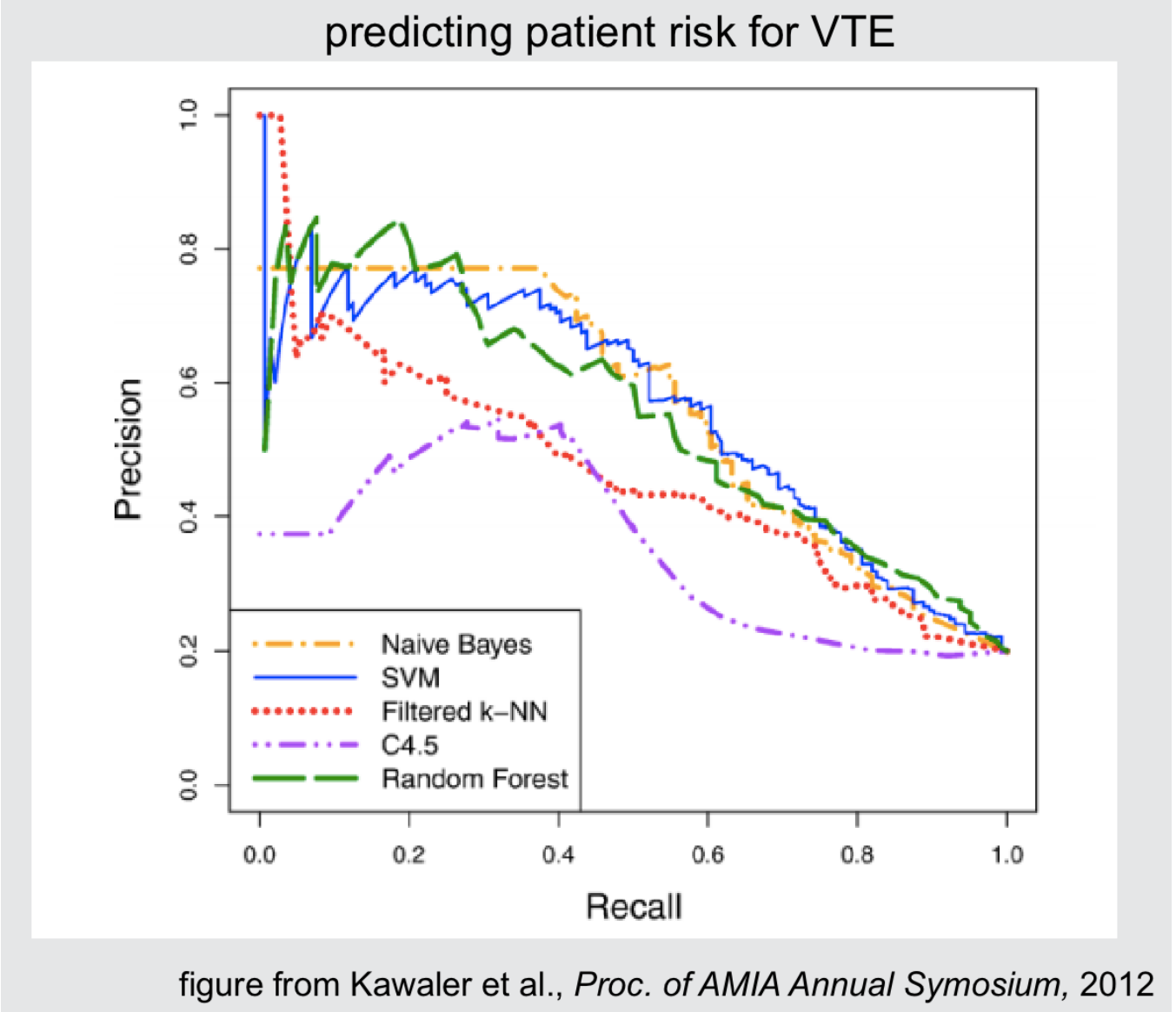


figure from Kawaler et al., *Proc. of AMIA Annual Symosium,* 2012

Summary of precision-recall

- Reporting one number
- Take the harmonic mean: **F1 score**

- Fact: minimum of two numbers <= harmonic mean <= geometric mean <= arithmetic mean

$$F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}}$$

- Emphasizes the smaller measure
  - E.g. $\text{recall} = 0.1, \text{precision} = 0.9 \Rightarrow F_1 = 0.18$

- Area under PR-curve is also a popular metric

	0.0	0.2	0.4	0.6	0.8	1.0
0.0	0.00	0.00	0.00	0.00	0.00	0.00
0.2	0.00	0.20	0.26	0.30	0.32	0.33
0.4	0.00	0.26	0.40	0.48	0.53	0.57
0.6	0.00	0.30	0.48	0.60	0.68	0.74
0.8	0.00	0.32	0.53	0.68	0.80	0.88
1.0	0.00	0.33	0.57	0.74	0.88	1.00

Table 5.2: Table of f-measures when varying precision and recall values.

Outline

- The role of features in supervised learning
- Classification metrics beyond error rate
- Model Evaluation & Comparison
- Debugging Learning Algorithms
- Bias / Variance Tradeoff

Model Evaluation & Comparison

Motivation: evaluating & comparing ML models

Example

- Your ML model f has test set error = 6.9%
- Your colleague Gabe's ML model g has test set error = 6.8%
- How confident are we to conclude g is better than f?

better = has smaller generalization error

- Intuition: We should be more (less) confident, if the test set is larger (smaller)
- Our uncertainty can be quantified with a *confidence interval*
- Determining the best model can be done rigorously with *hypothesis testing*

Disclaimer: we only focus on reviewing the key ideas (standard stats courses spend >= 5 lectures)

Hypothesis

- Statements about parameter / property θ of a distribution / population

Examples

- Average GPA < 2.8
- Probability of head of a coin > 0.6
- People eat more on weekends than weekdays

Hypothesis testing

- How to test?
- Design experiment, collect data, check:

If data shows strong evidence against H_0:

Reject H_0 (in favor of H_1)

Else

Do not reject H_0

Note: does not necessarily mean "accept H_0 "

- Analogy with the legal principle:
  - Presume innocent (H_0) until proven guilty (H_1) with strong evidence against innocence

Application: A/B Testing

An Internet company tries out an alternative of user interface (UI) on **randomly chosen subset of users** to collect their feedback (e.g. rating)

- E.g, choosing b/w list view vs grid view

How do we know if the new UI is better than older one, with statistical significance?

Application: A/B Testing

Evaluator	1	2	3	4	5	6
Old UI	5	2	2	5	4	2
New UI	4	4	1	3	3	5

Compute the score differences:

Evaluator	1	2	3	4	5	6
Score difference δ	-1	+2	-1	-2	-1	+3

Can view δ's as drawn from some distribution with unknown mean μ

"Does new UI improve over old UI?" is now a hypothesis testing problem:

$$H_0\colon \mu \leq 0, \qquad H_1\colon \mu > 0$$

we can perform a *t-test* based on data

A baby hypothesis test

Example Understanding the side-effect of a new drug in raising blood pressure

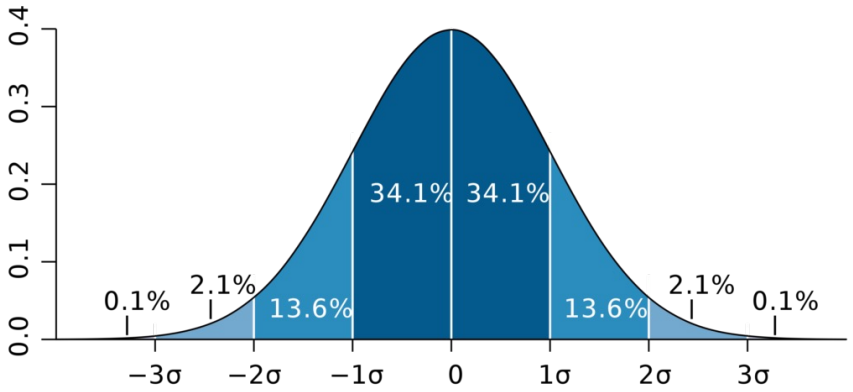
Null hypothesis H_0: $\mu \leq 0$,

μ = mean blood pressure change after taking new drug

Suppose I have *one* data point X drawn from $N(\mu, \sigma^2)$, when should I reject H_0?

E.g. Patient A has a significant increase in blood pressure

Test: reject if $X > c$



How to choose c?

Choose c such that we most likely don't reject when H_0 happens

$$P_{H_0}(X > c) \leq \alpha = 0.025 \text{ (say)}$$

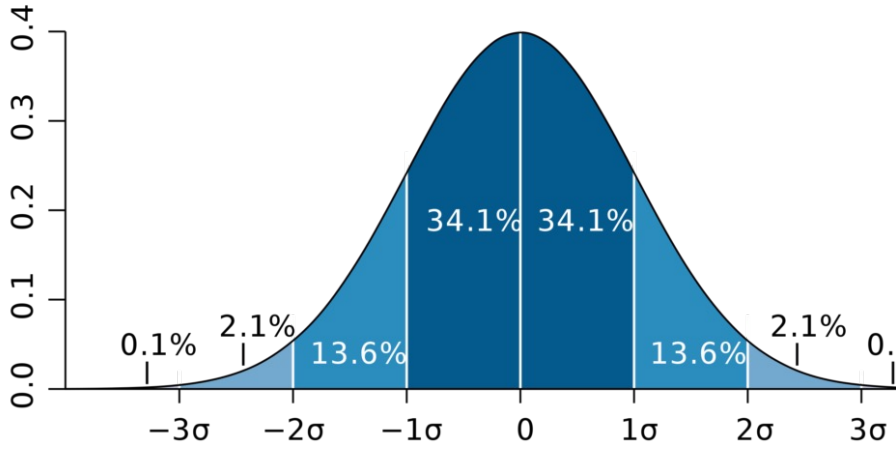
α: significance level

A smaller $\alpha \Rightarrow$ More inclined to keep H_0

A baby hypothesis test

Fact If $X \sim N(0, \sigma^2)$, then

$$P(X > 1.96\sigma) = 0.025$$



This continues to hold if *X*'s mean is < 0

Thus, choosing $c = 1.96\sigma$ ensures that

$$P_{H_0}(X > c) \leq \alpha = 0.025$$

Our final test:

Reject H_0 if $X > 1.96\sigma$, otherwise accept

Example $\sigma = 6$ (mmHg), and $X = 15$ (mmHg)

$$15 > 1.96 \times 6 \Rightarrow \text{Reject}$$

How to extend this to mulitple observations? To unknown σ?

Paired t-test

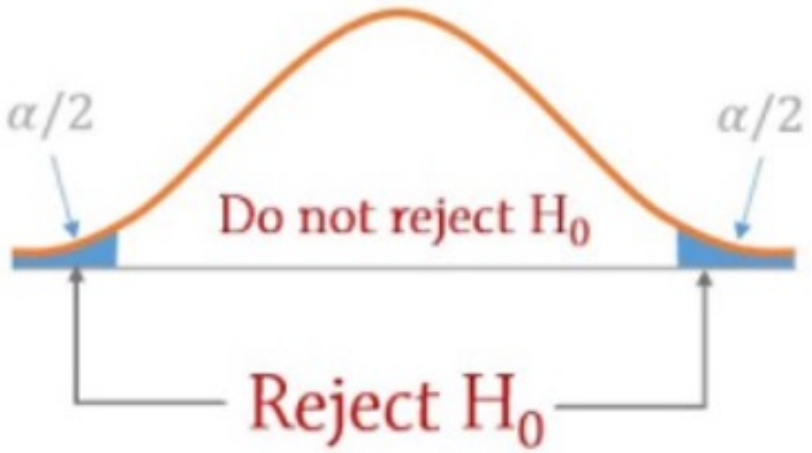
with mean μ_X with mean μ_Y

- Given: $S_X = (X_1, \dots, X_n)$ and $S_Y = (Y_1, \dots, Y_n)$
  - $H_0: \mu_X = \mu_Y$
  - $H_1: \mu_X \neq \mu_Y$

Evaluator	1	2	..
A	5	2	..
B	4	1	..

A reasonable proposal:

- Let $\delta_i := X_i - Y_i$, for all $i = 1, \dots, n$
- Let $\bar{\delta}_n := \frac{1}{n}\sum_{i=1}^{n} \delta_i$
- Reject H_0 if $|\bar{\delta}_n| > c$



- How to choose c?

The student-t distribution

Fact $X_1, \ldots, X_n$ is an iid sample with unknown μ & σ^2. Let $\hat{\sigma}_n = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X}_n)^2}$.

Then, approximately:

$$\sqrt{n}\frac{\bar{X}_n - \mu}{\hat{\sigma}_n} \sim \text{student-t}(n-1)$$

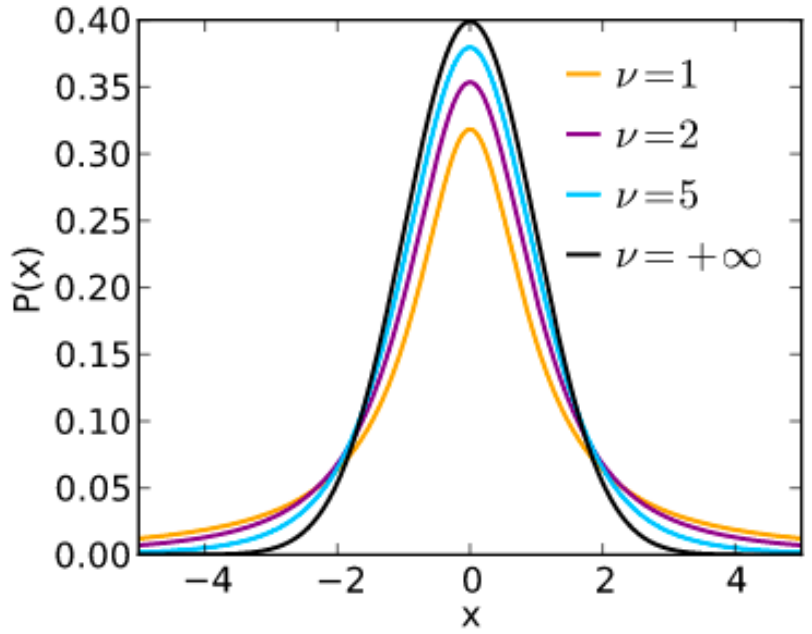
Abbreviated as $t(n-1)$

degree of freedom

student-t(ν) is a family of distributions

William S Gosset
(aka *Student*)

While working at
Guinness



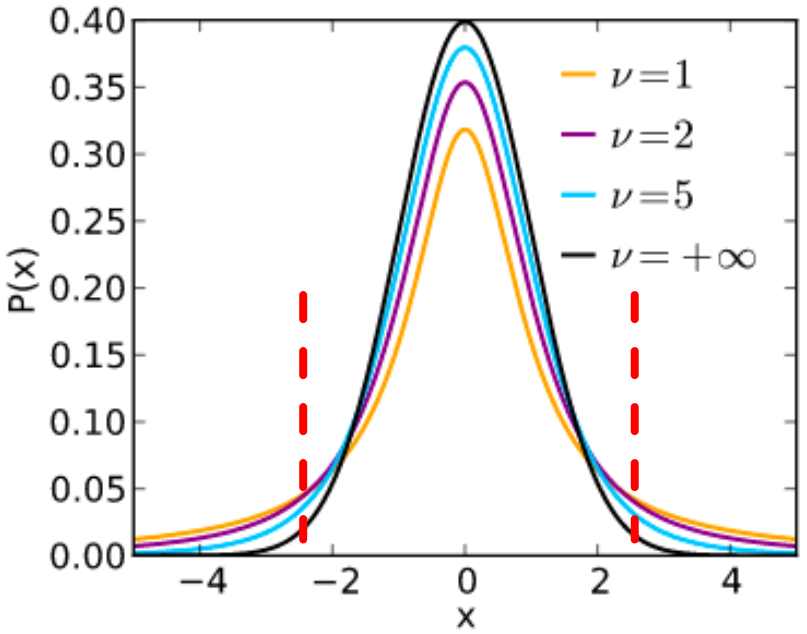
Paired t-test

- Given: $S_X = (X_1, \dots, X_n)$ and $S_Y = (Y_1, \dots, Y_n)$
  - $H_0: \mu_X = \mu_Y$
  - $H_1: \mu_X \neq \mu_Y$

Evaluator	1	2	..
A	5	2	..
B	4	1	..

Revised proposal:

- Let $\delta_i := X_i - Y_i$, for all $i = 1, \dots, n$
- Let $\bar{\delta}_n := \frac{1}{n}\sum_{i=1}^{n} \delta_i$ and $\hat{\sigma}_n = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\delta_i - \bar{\delta}_n)^2}$
- Reject H_0 if $\left|\sqrt{n}\frac{\bar{\delta}_n}{\hat{\sigma}_n}\right| > c$



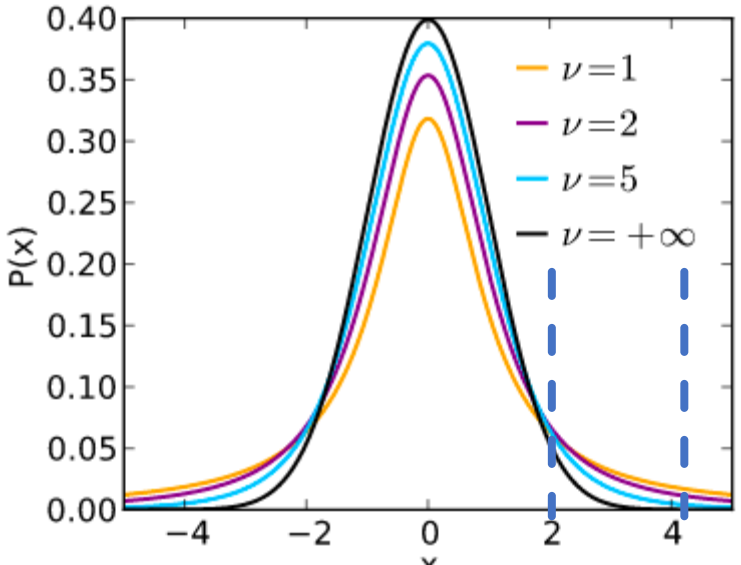
When H_0 happens, Follows $t(n-1)$ distribution

- How to choose c? **Choose c such that $\mathrm{P}_{Z\sim t(n-1)}(|Z| > c) = \alpha(\text{say}, 0.05)$**

Paired t-test

- Choose c such that $P_{Z\sim t(n-1)}(|Z| > c) = \alpha(\text{say}, 0.05)$

- $c = \left(1 - \frac{\alpha}{2}\right)$-quantile of the $t(n-1)$ distribution
  - abbrev. $t_\alpha(n-1)$



$t_\alpha(30) = 2.0 \quad t_\alpha(2) = 4.3$

- In summary:

Reject H_0 if $\left|\sqrt{n}\frac{\bar{\delta}_n}{\hat{\sigma}_n}\right| > t_\alpha(n-1)$

```
import scipy.stats as st
alpha = 0.05
st.t.ppf(1-alpha/2,df=2)
=> 4.302652729911275

st.t.ppf(1-alpha/2,df=5)
=> 2.5705818366147395

st.t.ppf(1-alpha/2,df=10)
=> 2.2281388519649385

st.t.ppf(1-alpha/2,df=30)
=> 2.0422724563012373

st.t.ppf(1-alpha/2,df=100)
=> 1.9839715184496334
```

Example

- An online store is deciding between: List view (current design) vs Grid view (new design).
- Over 18 days, suppose we collect purchase rate data and have computed that

$$\bar{\delta}_n = 0.24 \text{ and } \hat{\sigma}_n = 0.30$$

(in percent)

- Can we conclude with $\alpha = 0.05$ significance level that one view is better than the other?

Smaller $\alpha \Rightarrow$ Conclusion is stronger

- t-test:

$$\text{Reject } H_0 \text{ if } \left|\sqrt{n}\frac{\bar{\delta}_n}{\hat{\sigma}_n}\right| \geq t_\alpha(17)$$

($\sqrt{18}$, 0.24, 0.30)

```
1 st.t.ppf(1-0.05/2, 17)
```

```
np.float64(2.1098155778331806)
```

3.39 > 2.10 => Yes, one view is better

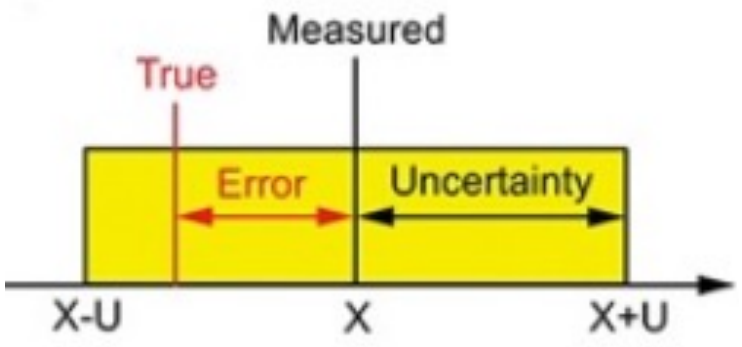
Specifically, $\bar{\delta}_n > 0$ => we conclude that grid view is better

Confidence interval

- In many applications, we'd like to make statements with *uncertainty quantifications*
  - "Given the test data, I estimate the generalization error of this classifier to be 0.10 ± 0.05"
  - "Given the data, I estimate the mean height of UA students to be 172 ± 2cm"
  - Also ubiquitous when making *measurements*

"Make multiple measurements and take the average"
-- Science teacher

- This is called *interval estimation*



A baby confidence interval

- Suppose I have *one* datapoint X from $N(\mu, \sigma^2)$, where would I guess μ to be at?

E.g. μ = mean blood pressure change after taking new drug

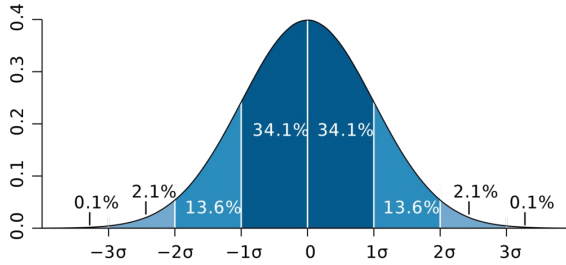
- **Example** $X = 8\text{mmHg}$, and $\sigma = 3\text{mmHg}$

Fact If $X \sim N(\mu, \sigma^2)$, then

$$P(-1.96\sigma \leq X - \mu \leq 1.96\sigma) = 0.95$$

with 95% confidence, X falls within 1.96 standard deviation of μ

i.e, with 95% confidence, μ falls within 1.96 standard deviation of X



$[X - 1.96\sigma, X + 1.96\sigma]$ is a 95%-confidence interval for μ

95% -- confidence level – the higher, the interval is more likely to contain μ

t-confidence interval

- What if we have multiple data points? What if σ is unknown?
- Idea:
  - Take average over multiple data points => call it $\bar{X}_n$
  - Use the fact that $\sqrt{n}\frac{\bar{X}_n-\mu}{\hat{\sigma}_n} \sim$ student-t$(n-1)$

Sample standard deviation

William S Gosset (aka *Student*)

While working at Guinness

- **Fact:** $\left[\bar{X}_n \pm \frac{t_\alpha(n-1)\hat{\sigma}_n}{\sqrt{n}}\right]$ is a $(1-\alpha)$-confidence interval for μ

t-confidence interval in action

- Suppose we have seen a classifier's mistake indicator on n=18 test examples, with

$$\bar{X}_n = 0.10 \text{ and } \hat{\sigma}_n = 0.3$$

Evaluator	1	2	..
Mistake?	0	1	..

- Find a 95% confidence interval on μ, the classifier's generalization error

- The interval is

$$[\bar{X}_n \pm \frac{t_\alpha (n-1)\hat{\sigma}_n}{\sqrt{n}}]$$

0.10 2.1 0.3 $\sqrt{18}$

```
1 st.t.ppf(1-0.05/2, 17)
```

```
np.float64(2.1098155778331806)
```

- which is [0.10 – 0.148, 0.10 + 0.148]

Confidence interval & Hypothesis testing

- Confidence intervals may be used for hypothesis testing

Model \ Test example	1	2	..
A	0	1	..
B	1	1	..

- E.g. Is model A better than model B?
- Model A's generalization error has confidence interval

$$[0.10 - 0.148, 0.10 + 0.148]$$

- Model B's has confidence interval

$$[0.32 - 0.1, 0.32 + 0.1]$$

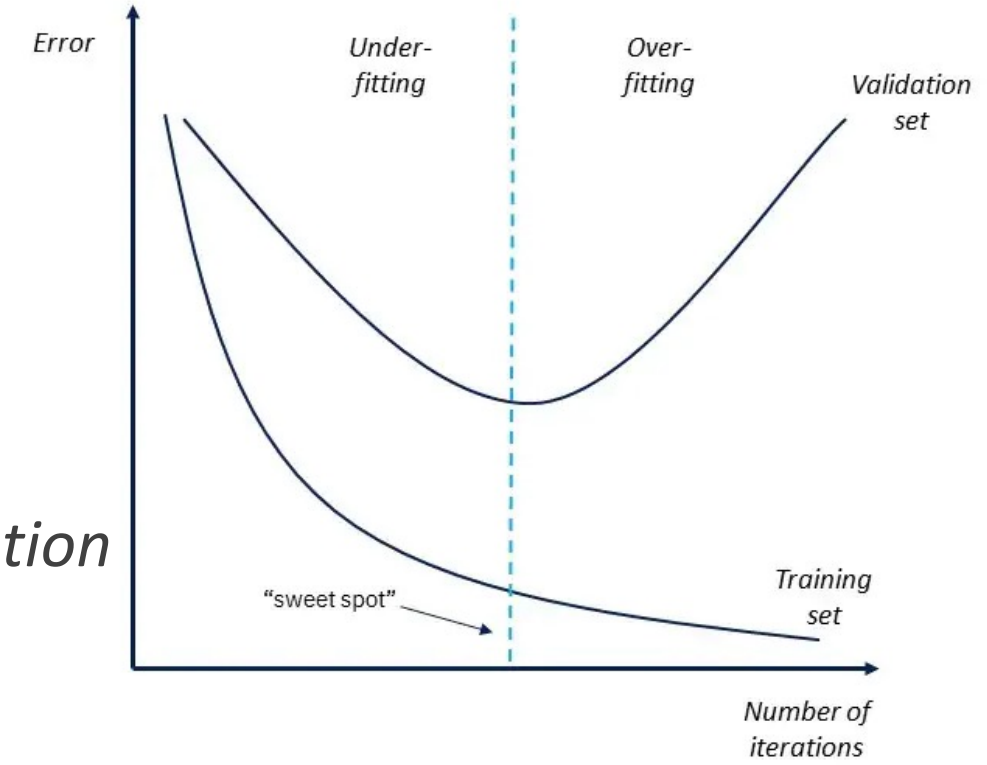
- In this example, we cannot confidently conclude that model A is better than model B
- But we can if model B's confidence interval is $[0.42 - 0.1, 0.42 + 0.1]$

Debugging Learning Algorithms

Debugging Learning Algorithms

Is the problem with generalization to test data?

- Is it doing well on training?
  - Unrealistic to do *better* on test than on training
  - If not, problem may be *representation*: need better features or better data
- If it does well on training then problem is *generalization*
  - Model may be too complex (overfitting)
  - Too many features, not enough training data



Debugging Learning Algorithms

Do you have adequate representation?

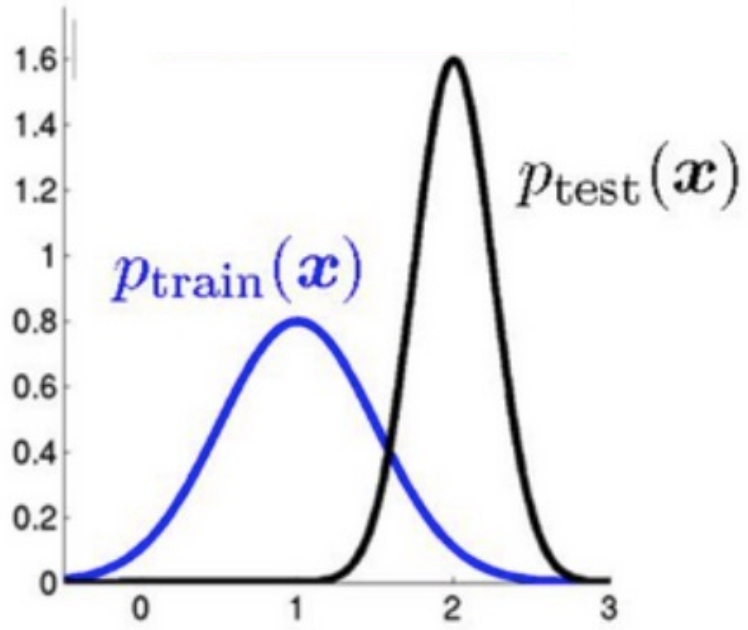
- Your feature set could be inadequate
- For binary classification try this…
  - Add a feature (maybe call it `CheatingIsFun`)
  - Set value to +1 for positive instances and -1 for negative instances
  - This feature is a *perfect indicator*-problem is now trivially solvable
  - Does your algorithm solve it?
- If your algorithm doesn't get near 0% error then you *may* have a bug! (or more data / less features)
- If it does then you need better features or a different model (e.g. decision tree vs. linear model)

Debugging Learning Algorithms

Is there a mismatch between training and test?

- Training data may be inadequate

- If enough training & test data:
  - Do results change with different train / test split?
  - If so then test distribution is probably strange



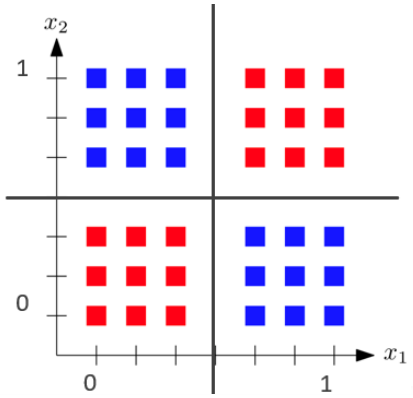
Debugging Learning Algorithms

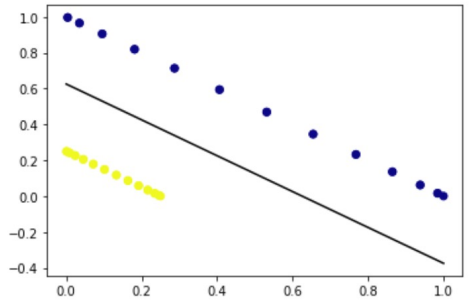
Is the learning algorithm implemented correctly?

- Is it optimizing the loss function that you intended?

$\sum_{i=1}^{n} \ell(f(x_i), y_i)$ ℓ**: loss function**

- Try measuring / visualizing your loss function during training -- is it going down?
- Do the data meet your algorithmic assumptions?
- Hand-craft datasets where you know the desired behavior
  - KNN on XOR dataset
  - Perceptron on data that is trivially linearly-separable
  - Decision tree on axis-aligned data
  - Generally, create dataset that meets assumptions of your algorithm

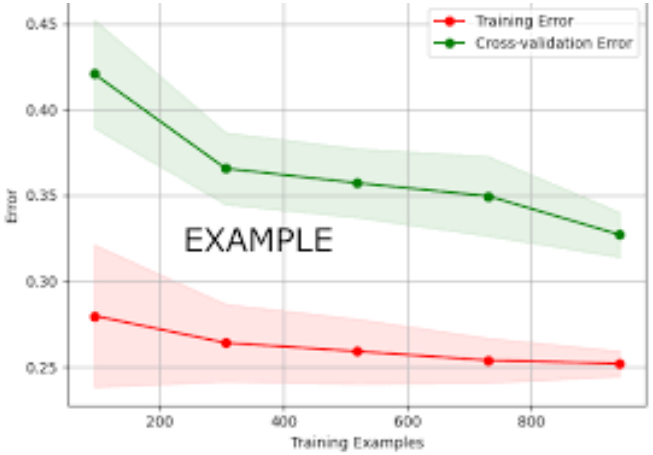




Debugging Learning Algorithms

Do you have enough data?

- Conventional wisdom: Always have *at least* as many training data as you have learnable model parameters
  - Caveat: may not be required in modern overparameterized setting
- Try training on 80% of your training data
  - Does performance suffer?
  - How much? A lot?
  - If so, then getting more data is likely helpful
  - If not, then you may be *data saturated*-look elsewhere

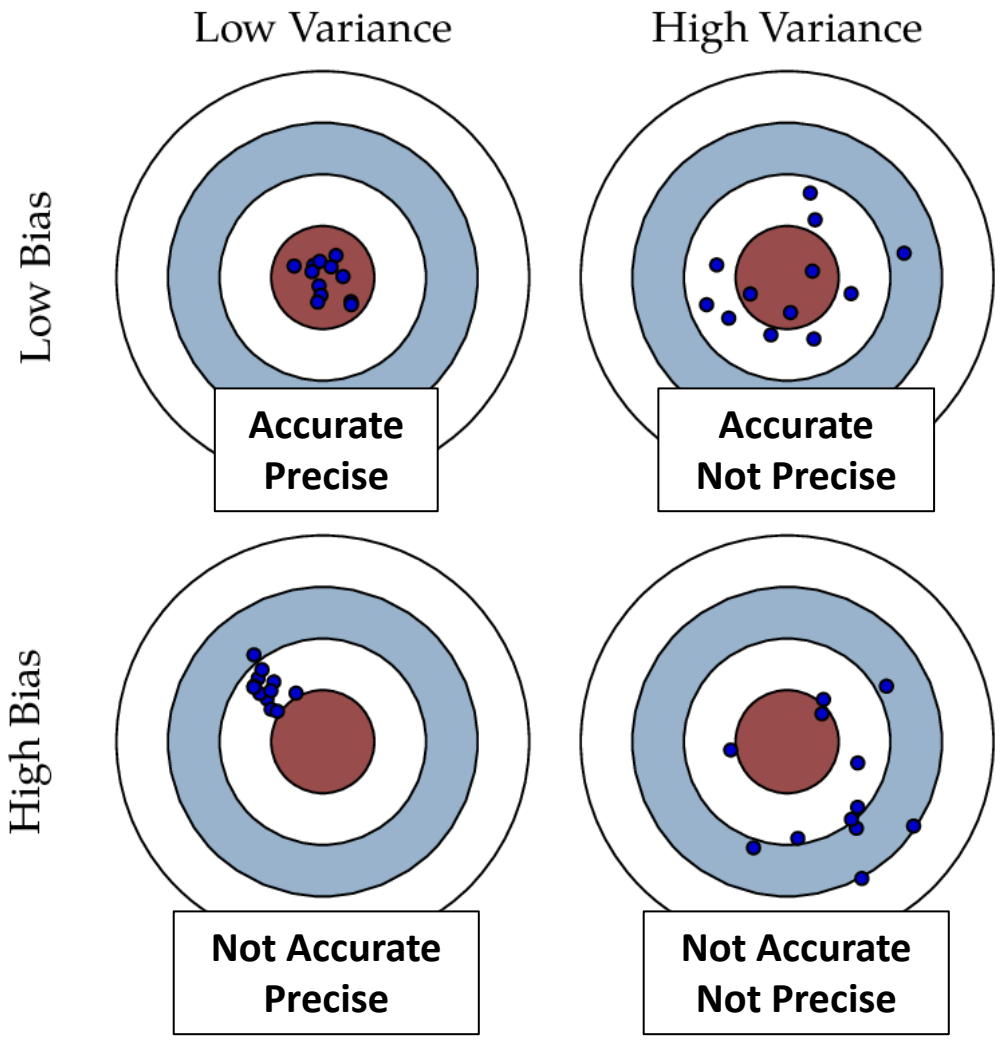


- More training data should never lead to *worse* performance (just slower training)

Bias / Variance Tradeoff

Bias-Variance Tradeoff

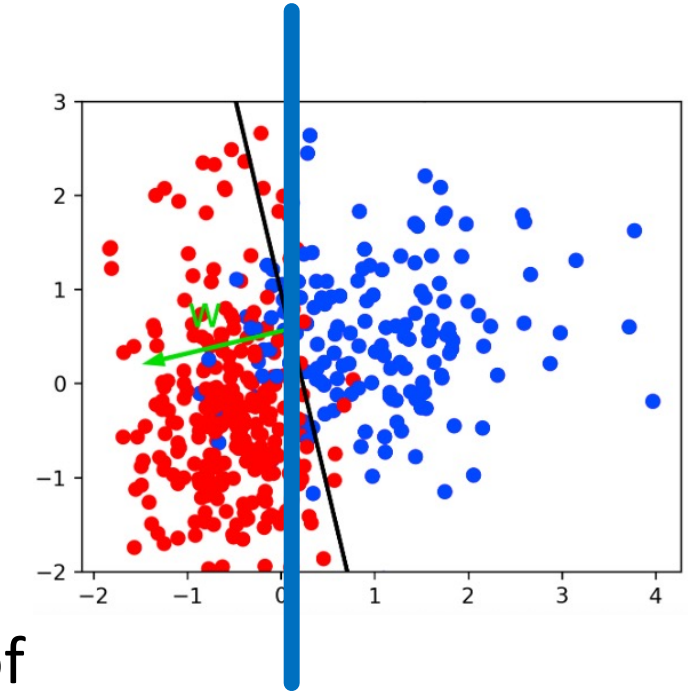
Suppose an archer takes multiple shots at a target…



Same for learning a prediction model

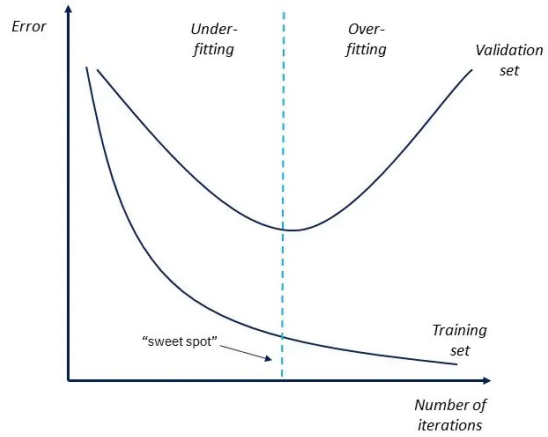
Target: ground truth model

Shot: learned model based on realization of a training dataset



Bias-Variance Tradeoff

Recall:



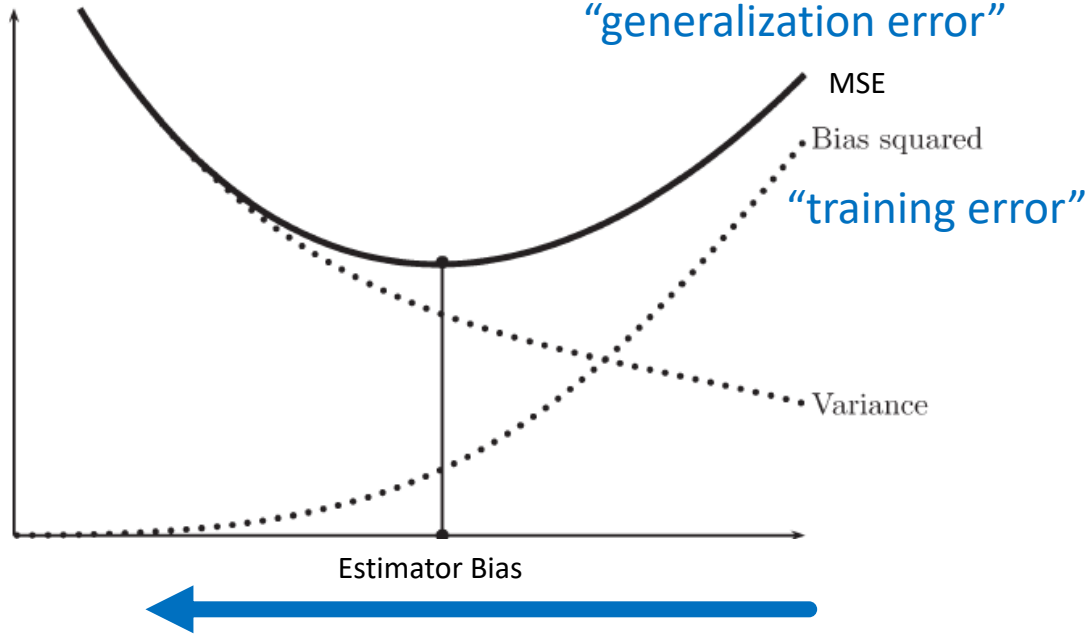
"generalization error"

Evaluate the quality of estimate $\hat{\theta}$ using **mean squared error**,

$$\mathrm{MSE}(\hat{\theta}) = \mathbf{E}\left[(\hat{\theta} - \theta)^2\right] = \mathrm{bias}^2(\hat{\theta}) + \mathbf{Var}(\hat{\theta})$$

$$\mathrm{bias}(\hat{\theta}) = \mathrm{E}[\hat{\theta}] - \theta$$

- Bias-variance is a fundamental tradeoff in statistical estimation
- MSE increases as **square** of bias
- Estimators with nonzero bias may have lower MSE than unbiased estimators



"training error"

Model more complex

Bias-Variance Decomposition

$$
\begin{aligned}
\mathrm{MSE}(\hat{\theta}) &= \mathbf{E}\left[(\hat{\theta}(X) - \theta)^2\right] \\
&= \mathbf{E}\left[\left(\hat{\theta} - \mathbf{E}[\hat{\theta}] + \mathbf{E}[\hat{\theta}] - \theta\right)^2\right] \\
&= \mathbf{E}[(\hat{\theta} - \mathbf{E}[\hat{\theta}])^2] + 2(\mathbf{E}[\hat{\theta}] - \theta)\mathbf{E}[\hat{\theta} - \mathbf{E}[\hat{\theta}]] + \mathbf{E}[(\mathbf{E}[\hat{\theta}] - \theta)^2] \\
&= \mathbf{E}[(\hat{\theta} - \mathbf{E}[\hat{\theta}])^2] + \left(\mathbf{E}[\hat{\theta}] - \theta\right)^2 \\
&= \mathrm{Var}(\hat{\theta}) + \mathrm{bias}^2(\hat{\theta})
\end{aligned}
$$

Additional reading

- Bootstrap confidence intervals: https://math.mit.edu/~dav/05.dir/class24-prep-a.pdf
- Permutation test: https://www.jwilber.me/permutationtest/
- STAT 566 lecture slides (at UA): https://www.math.arizona.edu/~jwatkins/stat566s20s.html

Next topic

- Linear models revisited: classification, regression, loss minimization formulations

- Assigned reading: CIML Chapter 7

- *Note: We are skipping Chapter 6 for now!*