ISTA 421/521
Introduction to Machine Learning

Lecture 20: Finishing Bayes Classifier, Nearest Neighbors, Assessing Classifiers

Clay Morrison
clayton@sista.arizona.edu
Gould-Simpson 819
Phone 621-6609

4 November 2014

Main parametric modeling frameworks

- Minimizing a Loss function
  - Linear model
  - Linear least mean squares
- Maximum Likelihood
  - Probabilistic model of uncertainty (noise, error)
  - Maximize the likelihood w.r.t. parameters
  - Linear model with additive Gaussian noise
- Bayesian Approach
  - Treat parameters as random variables
  - Use Bayes Theorem to combine likelihood & prior to find posterior distribution
- Estimation Techniques (often used in Bayesian approaches)
  - Gradient methods (Widrow-Hoff (1st), Newton-Rhapson (2nd))
  - Laplace Approximation
  - Monte Carlo estimation of expectation; Metropolis-Hastings

Approaches to Avoiding Over-fitting
i.e., how to achieve ***generalization***
Regularization
Cross Validation (estimating the generalization error)

- Classification (& Regression)
- Clustering
- Projection

predicting output

Main algorithmic families of Machine Learning

Another Naïve Bayes Example: Classifying Text

- 20 newsgroup dataset
- Total of 20,000 documents
- Task: assign a new document to one of the 20 newsgroups
- ***Bag-of-words*** model:
  - Total of *M* unique words for all documents
  - $\mathbf{X}_n$ is vector of counts of each of the *M* word types
    - (so $x_{n,m}$ is the number of times word *m* appears in document *n*)

$$p(\mathbf{x}_n|T_n = c, \ldots) = \prod_{m=1}^{M} p(x_{nm}|T_n = c, \ldots)$$

$\mathbf{X}_n$ is vector of counts of each of the *M* word types
so $x_{n,m}$ is the number of times word *m* appears in document *n*

$$p(\mathbf{x}_n|T_n = c, \ldots) = \prod_{m=1}^{M} p(x_{nm}|T_n = c, \ldots)$$

(e.g., coin, prob of each face)

Bernoulli $\xrightarrow{\text{Sums over repeated Bernoulli draws}}$ **Binomial**

$$P(x|p) = p^x(1-p)^{1-x}$$

$$P(x|p,n) = \binom{n}{x} p^x (1-p)^{n-x}$$

Bernoulli → >2 valued → Categorical

Binomial → >2 valued → Multinomial

$$P(x|p_1, \ldots, p_k) = p_1^{\mathbf{1}_{x=1}} p_2^{\mathbf{1}_{x=2}} \cdots p_k^{\mathbf{1}_{x=k}}$$

Categorical $\xrightarrow{\text{Sums over repeated Categorical draws}}$ **Multinomial**

(e.g., dice, prob of each side)

$$P(x_1, \ldots, x_k|p_1, \ldots, p_k, n) = \frac{n!}{x_1! \cdots x_k!} p_1^{x_1} \cdots p_k^{x_k}$$

Another Naïve Bayes Example: Classifying Text

- Note: the *bag-of-words* model ignores word order.

  1: The quick brown fox jumps over the lazy dog.

  2: Dog quick lazy the jumps fox brown the over.

m = word
n = document
x_{nm} = count of word *m* in document *n*
q_m = probability of word *m*

- **Multinomial** distribution:

$$P(\mathbf{x}_n|\mathbf{q}) = \left(\frac{s_n!}{\prod_{m=1}^{M} x_{nm}!}\right) \prod_{m=1}^{M} q_m^{x_{nm}}$$

Prior:

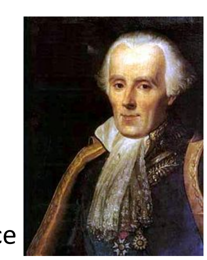
$$P(T_{\text{new}} = c|\mathbf{X},\mathbf{t}) = \frac{1}{C}$$

- Max likelihood estimate of *q*: $q_{cm} = \frac{\sum_{n=1}^{N_c} x_{nm}}{\sum_{m'=1}^{M}\sum_{n=1}^{N_c} x_{nm'}}$
- Prediction:

$$P(T_{\text{new}} = c|\mathbf{x}_{\text{new}},\mathbf{X},\mathbf{t}) = \frac{p(\mathbf{x}_{\text{new}}|T_{\text{new}} = c,\mathbf{X},\mathbf{t})P(T_{\text{new}} = c|\mathbf{X},\mathbf{t})}{\sum_{c'=1}^{C} p(\mathbf{x}_{\text{new}}|T_{\text{new}} = c',\mathbf{X},\mathbf{t})P(T_{\text{new}} = c'|\mathbf{X},\mathbf{t})}$$

The "Zero-Frequency Problem" (1)

- What if an nominal attribute value doesn't occur with every class value?
  (e.g., no outlook "overcast" for class "no play")
  - Probability will be zero!
  - A posteriori probability will also be zero!
    (no matter how likely the other values are!)
- Remedy for **Bernoulli/Categorical likelihood**: add 1 to the count for every attribute value-class combination (*Laplace estimator* or *rule of succession*)
- Result: probabilities will never be zero
  (also stabilizes probability estimates)



Pierre-Simon, marquis de Laplace

The “Zero-Frequency Problem” (2)

(Bernoulli/Categorical likelihood continued…)

- In some cases, adding a constant different from 1 might be more appropriate: μ
- Example: attribute *outlook* for class *yes*

Sunny	Overcast	Rainy
$\frac{2+\frac{\mu}{3}}{9+\mu}$	$\frac{4+\frac{\mu}{3}}{9+\mu}$	$\frac{3+\frac{\mu}{3}}{9+\mu}$

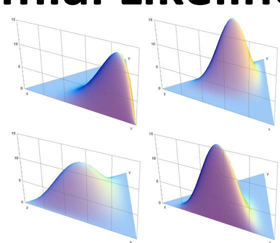
- Weights (p’s) don’t need to be equal

  (but p’s **must sum to 1** !)

$$\frac{2+(\mu\cdot p_1)}{9+\mu} \qquad \frac{4+(\mu\cdot p_2)}{9+\mu} \qquad \frac{3+(\mu\cdot p_3)}{9+\mu}$$

The Bayesian Approach to Zero-Frequency problem for Multinomial Likelihood

- The **Dirichlet** Prior:



Example Dir. as distribution on a 3-simplex

$$p(\mathbf{q}_c|\boldsymbol{\alpha}) = \frac{\Gamma\left(\sum_{m=1}^{M}\alpha_m\right)}{\prod_{m=1}^{M}\Gamma(\alpha_m)}\prod_{m=1}^{M} q_{cm}^{\alpha_m-1}$$

- MAP estimate of Multinomial Likelihood and Dirichlet prior:

$$q_{cm} = \frac{\alpha - 1 + \sum_{n=1}^{N_c} x_{nm}}{M(\alpha-1) + \sum_{m'=1}^{M}\sum_{n=1}^{N_c} x_{nm'}}$$

- Called “smoothing” because as α is increased, q_{cm} gets closer to $1/M$

Beta $p(x|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$

Conjugate prior over Bernoulli parameter

Sums over repeated Bernoulli draws

Bernoulli → Binomial $P(x|p,n) = \binom{n}{x} p^x (1-p)^{n-x}$

$P(x|p) = p^x(1-p)^{1-x}$

$P(x|p_1, ..., p_k) = p_1^{\mathbf{1}_{x=1}} p_2^{\mathbf{1}_{x=2}} \cdots p_k^{\mathbf{1}_{x=k}}$

$P(x_1, ..., x_k|p_1, ..., p_k, n) = \frac{n!}{x_1! \cdots x_k!} p_1^{x_1} \cdots p_k^{x_k}$

Categorical → Multinomial

Sums over repeated Categorical draws

Conjugate prior over category probabilities

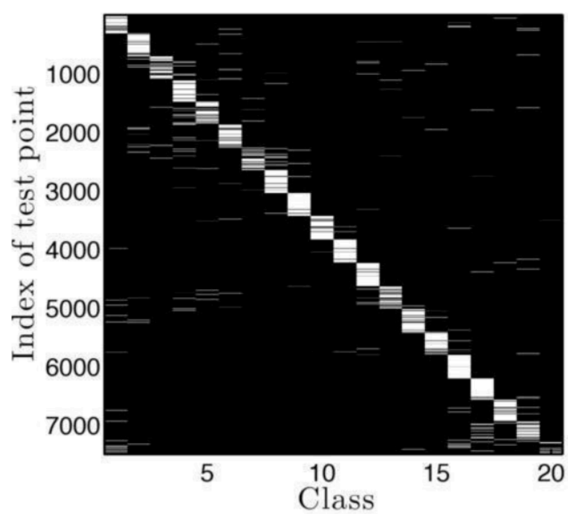
Dirichlet

$p(x_1, ..., x_k|\alpha_i, ..., \alpha_k) = \frac{\Gamma(\sum_{i=1}^K \alpha_i)}{\prod_{i=1}^K \Gamma(\alpha_i)} \prod_{i=1}^K x_i^{\alpha_i - 1}$

Document Classification Results

- Trained on 11,000 documents
- Results of classifying 7,000 held-out documents:

Note: the test points are ordered by true class, so the more the squares on the diagonal tend to pure white, the better. White-to-grey rows off the diagonal indicate confusion between classes. (E.g., there is confusion between classes 19 & 17, and 16 & 20.)



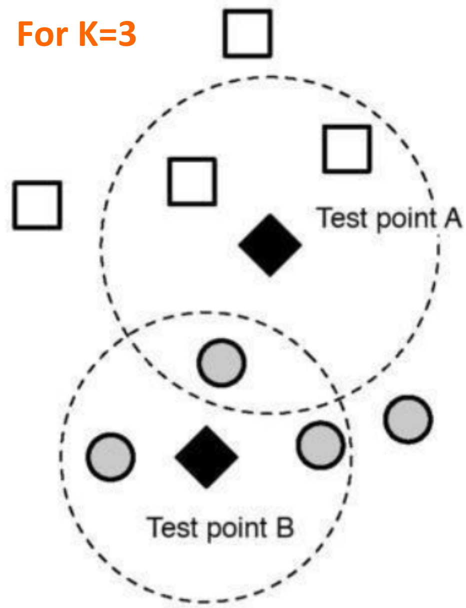
Non-Probabilistic Classifiers

Non-probabilistic Classifier: KNN

- **K-nearest neighbors** (KNN)
- Very popular because very simple *and* excellent empirical performance
- Handles both binary and multi-class data
- Makes no assumptions about the parametric form of the decision boundary
- **Does not have a training phase** – just store the training data and do computation when time to classify

KNN Classification

- Find the K “training points” that are closest to x_{new} .
- Select the majority class amongst these neighbors (or average, for regression)



KNN Classification

- Can use any **distance metric**

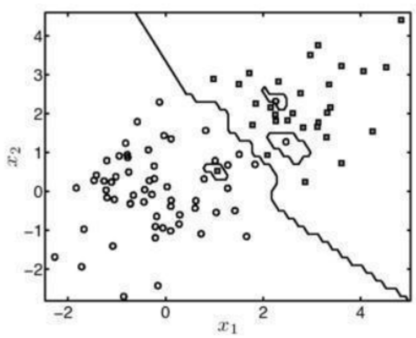
$d : X \times X \to \mathbf{R}$

1. $d(x, y) \geq 0$ (non-negativity, or “separation axiom”)
2. $d(x, y) = 0$ if and only if $x = y$ (identity of indiscernibles)
3. $d(x, y) = d(y, x)$ (symmetry)
4. $d(x, z) \leq d(x, y) + d(y, z)$ (triangle inequality, or “subadditivity”)

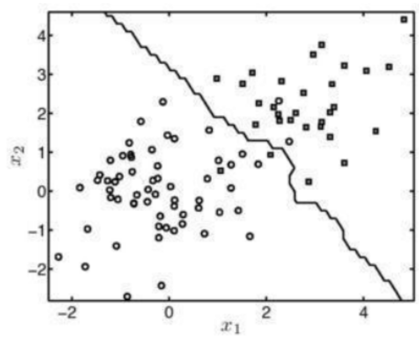
- Therefore, can be used on any data for which we can define a distance between two objects
- KNN has been used successfully for
  - Strings (string edit distance)
  - Graphs (graph edit distance)
  - Images (local feature similarity)

KNN Classification

- Three ingredients: Data, Distance Metric, K
- How to choose K ?
  - If K is too **small**, classification may be heavily influenced by noise



(a) Decision boundary when $K = 1$

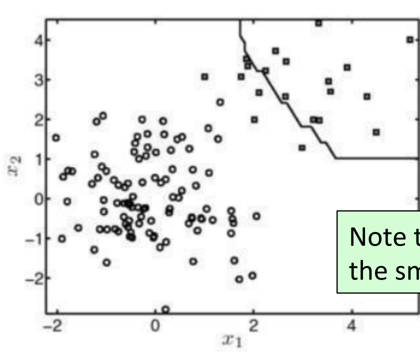


(b) Decision boundary when $K = 5$

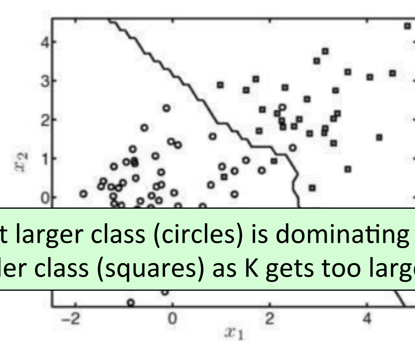
Including more neighbors has effect of "regularizing" the boundary

KNN Classification

- Three ingredients: Data, Distance Metric, K
- How to choose K ?
  - Increasing K reduces over-fitting, but to a point.
  - If K is too **big**, loose structure (extreme case, N_1=10, K ≥ 21)



(b) Decision boundary when $K = 39$



(b) Decision boundary when $K = 5$

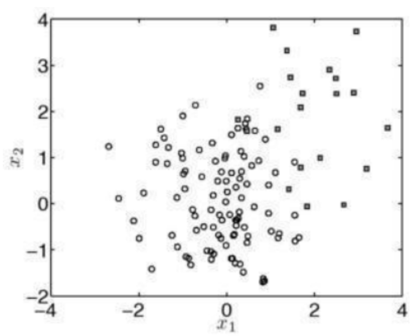
Including more neighbors has effect of "regularizing" the boundary

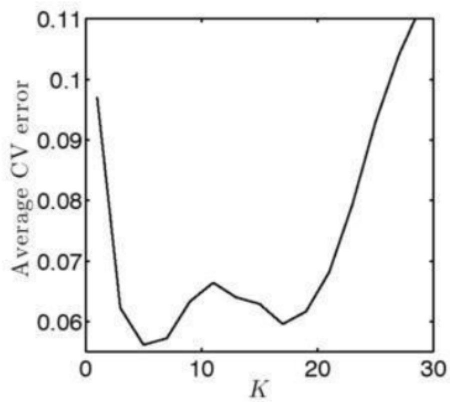
Note that larger class (circles) is dominating the smaller class (squares) as K gets too large

KNN Classification

- Three ingredients: Data, Distance Metric, K
- How to choose K ?
  - Most popular way to choose K: cross-validation!
  - Simple performance measure: proportion of mistakes

10-fold CV, repeated 100 times





Assessing Classifiers

Assessing Classifiers

- **Consider Binary Classification**
- Decisions can be right or wrong
- How many ways can you be right? Wrong?

	Truly "Yes"	Truly "No"
Say "Yes"	True positive	False Positive
Say "No"	False Negative	True Negative

Lots of functions!

	Truly "Yes"	Truly "No"
Say "Yes"	True positive	False Positive
Say "No"	False Negative	True Negative

true positive (TP)
eqv. with hit
true negative (TN)
eqv. with correct rejection
false positive (FP)
eqv. with false alarm, Type I error
false negative (FN)
eqv. with miss, Type II error
sensitivity or true positive rate (TPR)
eqv. with hit rate, recall
$TPR = TP / P = TP / (TP + FN)$
false positive rate (FPR)
eqv. with fall-out
$FPR = FP / N = FP / (FP + TN)$
accuracy (ACC) ← "classification accuracy"
$ACC = (TP + TN) / (P + N)$
specificity (SPC) or True Negative Rate
$SPC = TN / N = TN / (FP + TN) = 1 - FPR$
positive predictive value (PPV)
eqv. with precision
$PPV = TP / (TP + FP)$
negative predictive value (NPV)
$NPV = TN / (TN + FN)$
false discovery rate (FDR)
$FDR = FP / (FP + TP)$
Matthews correlation coefficient (MCC)
$$MCC = (TP * TN - FP * FN)/\sqrt{PNP'N'}$$
F1 score
$F1 = 2TP / (P + P')$

Source: Fawcett (2006).

Lots of functions!

	Truly "Yes"	Truly "No"
Say "Yes"	True positive	False Positive
Say "No"	False Negative	True Negative

True Positive Rate

TP / (TP + FN)

False Positive Rate

FP / (FP + TN)

(Are these rates conditional, joint, or marginal probabilities?)

$$P(A \mid B) = \frac{P(A,B)}{P(B)}$$

If A = Truly "Yes" and B = Say "Yes"

$TP = A$ and B

$TP + FN = B$

Related Ideas:

Sensitivity and Specificity

- Common measures of **medical diagnostic tests**
- ***Sensitivity*** is like true positive rate
- ***Specificity*** is the number of **detected *negatives*** divided by the total number of negatives:

	Truly "Yes"	Truly "No"
Say "Yes"	True positive	False Positive
Say "No"	False Negative	True Negative

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

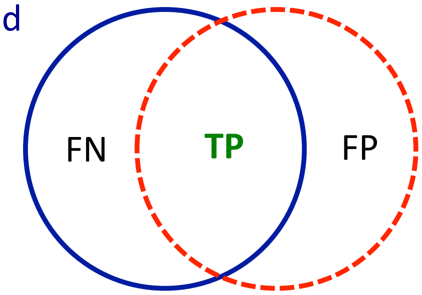
(= True Positive Rate)

$$\text{Specificity} = \frac{TN}{TN + FP}$$

(NOT False Positive Rate! Really: 1 – False Positive Rate)

Related Ideas:
Information Retrieval

The documents you'd like to retrieve



The documents you actually retrieved

	Truly "Yes"	Truly "No"
Say "Yes"	True positive	False Positive
Say "No"	False Negative	True Negative

Recall

TP / **(TP + FN)**

(How many of the original docs did I get?)

Precision

TP/ **(TP + FP)**

(How many of the docs I did get are the ones I wanted?)

F-measure:

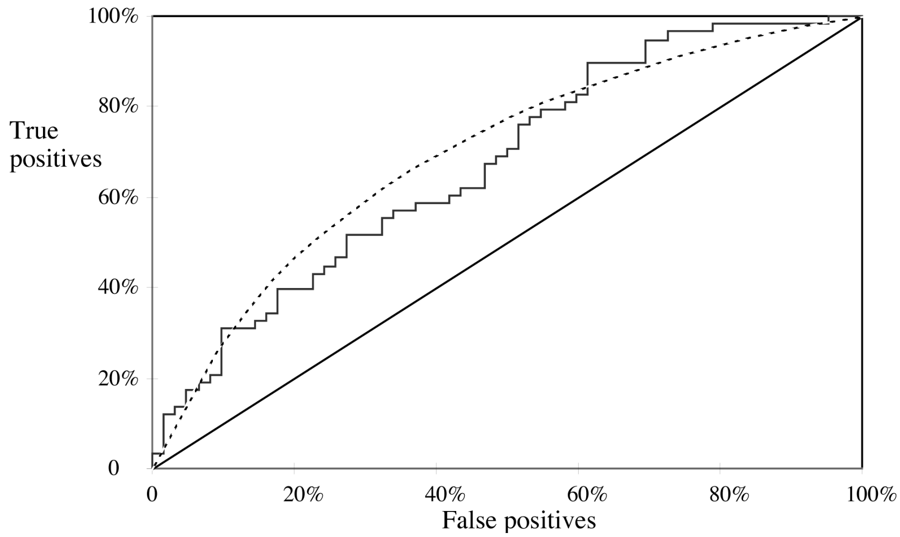
$$F = \frac{(2 \times \text{recall} \times \text{precision})}{(\text{recall} + \text{precision})}$$

… an average (harmonic mean) of precision & recall

Receiver Operating Characteristic (ROC)

- Many classification algorithms have real-valued output that is thresholded.
- The ***ROC curve*** show how performance varies as we change the threshold
- Procedure:
  - While varying the thresholds, run the classifier on test data and collect performance statistics (TP, FP, TN, FN)
  - Plot
    - **True positive rate** (aka sensitivity, hit rate: TP / (TP + FN)) against
    - **False positive rate** (aka false alarm rate or 1 – the specificity (FP / (TN+FP))
  - Cross-Validation is often used to “smooth” the curve (i.e., get better estimates of the performance tradeoff)
  - Possibly calculate associated measure (area under the curve)

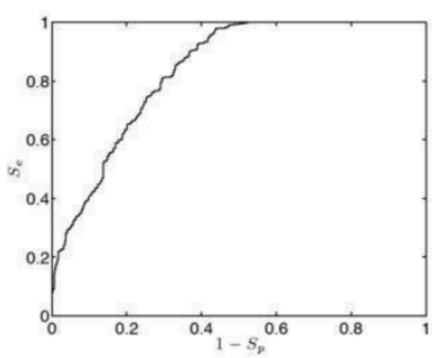
Example ROC curve



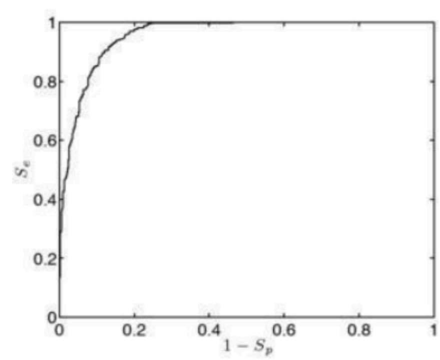
Jagged curve: one set of test data
Smooth curve (dotted): use of cross-validation

ROC curves can also help adjust other parameters

- Example of a classifier that not only has a threshold that can be varied for class decisions (i.e., what the ROC curve is built from), but also has other free parameters



(a) $\gamma = 0.01$



(b) $\gamma = 50$