Classification

Vikram Pudi
vikram@iiit.ac.in
IIIT Hyderabad

Talk Outline

- Introduction
  - Classification Problem
  - Applications
  - Metrics
  - Combining classifiers
- Classification Techniques

The Classification Problem

Outlook	Temp (°F)	Humidity (%)	Windy?	Class
sunny	75	70	true	*play*
sunny	80	90	true	*don't play*
sunny	85	85	false	*don't play*
sunny	72	95	false	*don't play*
sunny	69	70	false	*play*
overcast	72	90	true	*play*
overcast	83	78	false	*play*
overcast	64	65	true	*play*
overcast	81	75	false	*play*
rain	71	80	true	*don't play*
rain	65	70	true	*don't play*
rain	75	80	false	*play*
rain	68	80	false	*play*
rain	70	96	false	*play*
sunny	77	69	true	?
rain	73	76	false	?

Play Outside?

Model relationship between class labels and attributes

e.g. outlook = overcast ⇒ class = *play*

⇒ Assign class labels to new data with *unknown* labels

Applications

- Text classification
  - Classify emails into spam / non-spam
  - Classify web-pages into yahoo-type hierarchy
  - NLP Problems
    - Tagging: Classify words into verbs, nouns, etc.
- Risk management, Fraud detection, Computer intrusion detection
  - Given the properties of a transaction (items purchased, amount, location, customer profile, etc.)
  - Determine if it is a fraud
- Machine learning / pattern recognition applications
  - Vision
  - Speech recognition
  - etc.
- All of science & knowledge is about predicting future in terms of past
  - So classification is a very fundamental problem with ultra-wide scope of applications

Metrics

1. accuracy
2. classification time per new record
3. training time
4. main memory usage (during classification)
5. model size

Accuracy Measure

- Prediction is just like tossing a coin (random variable X)
  - "Head" is "success" in classification; X = 1
  - "tail" is "error"; X = 0
  - X is actually a mapping: {"success": 1, "error" : 0}
- In statistics, a succession of independent events like this is called a *bernoulli process*
  - Accuracy = P(X = 1) = p
  - mean value = $\mu = E[X] = p \times 1 + (1-p) \times 0 = p$
  - variance = $\sigma^2 = E[(X-\mu)^2] = p\,(1-p)$
- Confidence intervals: Instead of saying accuracy = 85%, we want to say: accuracy ∈ [83, 87] with a confidence of 95%

Binomial Distribution

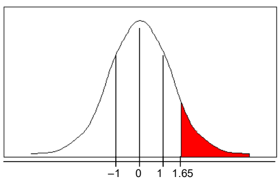
- Treat each classified record as a bernoulli trial
- If there are n records, there are n independent and identically distributed (iid) bernoulli trials, X_i, $i = 1,\ldots,n$
- Then, the random variable $X = \sum_{i=1,\ldots,n} X_i$ is said to follow a *binomial distribution*
  - $P(X = k) = {}^{n}C_k\, p^k (1-p)^{n-k}$
- Problem: Difficult to compute for large n

Normal Distribution

- Continuous distribution with parameters μ (mean), σ^2(variance)
- Probability density:
  $f(x) = (1/\sqrt{(2\pi\sigma^2)}) \exp(-(x-\mu)^2 / (2\sigma^2))$
- Central limit theorem:
  - Under certain conditions, the distribution of the sum of a *large number* of iid random variables is approximately normal
  - A *binomial distribution* with parameters n and p is approximately normal for large n and p not too close to 1 or 0
  - The approximating normal distribution has mean $\mu = np$ and standard deviation $\sigma^2 = (n\,p\,(1 - p))$

Confidence Intervals

Normal distribution with mean = 0 and variance = 1



Pr[$X \geq z$]	z
0.1%	3.09
0.5%	2.58
1%	2.33
5%	1.65
10%	1.28
20%	0.84
40%	0.25

- E.g. $P[-1.65 \leq X \leq 1.65] = 1 - 2 \times P[X \geq 1.65] = 90\%$
- To use this we have to transform our random variable to have mean = 0 and variance = 1
- Subtract mean from X and divide by standard deviation

Estimating Accuracy

- Holdout method
  - Randomly partition data: training set + test set
  - accuracy = |correctly classified points| / |test data points|
- Stratification
  - Ensure each class has approximately equal proportions in both partitions
- Random subsampling
  - Repeat holdout k times. Output average accuracy.
- k-fold cross-validation
  - Randomly partition data: $S_1,S_2,\ldots,S_k$
  - First, keep S_1 as test set, remaining as training set
  - Next, keep S_2 as test set, remaining as training set, etc.
  - accuracy = |*total* correctly classified points| / |total data points|
- Recommendation:
  - Stratified 10-fold cross-validation. If possible, repeat 10 times and average results. (reduces variance)

Is Accuracy Enough?

- If only 1% population has cancer, then a test for cancer that classifies *all* people as *non-cancer* will have 99% accuracy.
- Instead output a confusion matrix:

Actual/ Estimate	Class 1	Class 2	Class 3
Class 1	90%	5%	5%
Class 2	2%	91%	7%
Class 3	8%	3%	89%

Combining Classifiers

- Get k random samples with replacement as training sets (like in random subsampling).
- ⇨ We get k classifiers
- Bagging: Take a *majority vote* for the best class for each new record
- Boosting: Each classifier's vote has a *weight* proportional to its accuracy on training data
- ⇨ Like a patient taking multiple opinions from several doctors

Talk Outline

- Introduction
- Classification Techniques
  1. Nearest Neighbour Methods
  2. Decision Trees
     - ID3, CART, C4.5, C5.0, SLIQ, SPRINT
  3. Bayesian Methods
     - Naïve Bayes, Bayesian Belief Networks
     - Maximum Entropy Based Approaches
  4. Association Rule Based Approaches
  5. Soft-computing Methods:
     - Genetic Algorithms, Rough Sets, Fuzzy Sets, Neural Networks
  6. Support Vector Machines
  7. Convolutional Neural Networks, Deep Learning

Nearest Neighbour Methods

k-NN, Reverse Nearest Neighbours

k-Nearest Neighbours

- Model = Training data
- Classify record R using the *k* nearest neighbours of R in the training data.
- Most frequent class among *k* NNs
- Distance function could be euclidean
- Use an index structure (e.g. R* tree) to find the *k* NNs efficiently

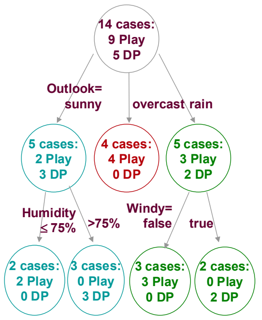
Reverse Nearest Neighbours

- Records which consider *R* as a *k*-NN
- Output most frequent class among RNNs.
- More resilient to outliers.

Decision Trees

Decision Trees

Outlook	Temp (°F)	Humidity (%)	Windy?	Class
sunny	75	70	true	*play*
sunny	80	90	true	*don't play*
sunny	85	85	false	*don't play*
sunny	72	95	false	*don't play*
sunny	69	70	false	*play*
overcast	72	90	true	*play*
overcast	83	78	false	*play*
overcast	64	65	true	*play*
overcast	81	75	false	*play*
rain	71	80	true	*don't play*
rain	65	70	true	*don't play*
rain	75	80	false	*play*
rain	68	80	false	*play*
rain	70	96	false	*play*



Basic Tree Building Algorithm

MakeTree (Training Data D):
    Partition(D)

Partition (Data D):
    if all points in D are in same class: return
    Evaluate splits for each attribute A
    Use best split found to partition D into $D_1, D_2, \ldots, D_n$
    for each D_i:
        Partition (D_i)

ID3, CART

ID3
- Use *information gain* to determine best split
- *gain* = $H(D) - \sum_{i=1\ldots n} P(D_i)\, H(D_i)$
- $H(p_1, p_2, \ldots, p_m) = -\sum_{i=1\ldots m} p_i \log p_i$
- like 20-question game
  - Which attribute is better to look for first: "Is it a living thing?" or "Is it a duster?"

CART
- Only create two children for each node
- Goodness of a split (Φ)
  $\Phi = 2\, P(D_1)\, P(D_2) \sum_{i=1\ldots m} | P(C_j / D_1) - P(C_j / D_2) |$

Shannon's Entropy

- An expt has several possible outcomes
- In N expts, suppose each outcome occurs M times
- This means there are N/M possible outcomes
- To represent each outcome, we need log N/M bits.
  - This generalizes even when all outcomes are not equally frequent.
  - Reason: For an outcome j that occurs M times, there are N/M equi-probable events among which only one cp to j
- Since $p_i = M / N$, information content of an outcome is $-\log p_i$
- So, expected info content: $H = - \Sigma\, p_i \log p_i$

Bayesian Methods

Naïve Bayes

- New data point to classify: $X=(x_1, x_2, \ldots x_m)$
- Strategy:
  - Calculate $P(C_i/X)$ for each class C_i.
  - Select C_i for which $P(C_i/X)$ is maximum

$$
\begin{aligned}
P(C_i/X) &= P(X/C_i)\, P(C_i) / P(X) \\
&\propto P(X/C_i)\, P(C_i) \\
&\propto P(x_1/C_i)\, P(x_2/C_i) \ldots P(x_m/C_i)\, P(C_i)
\end{aligned}
$$

- Naïvely assumes that each x_i is independent
- We represent $P(X/C_i)$ by $P(X)$, etc. when unambiguous