CPSC 340 Lecture 6 Outline

Kaitain Xie

1 Laplace Smoothing

$$\frac{\#\text{ spam messages with } x_i + 1}{\#\text{ spam messages } + 2}$$

2 Decision Theory

- 4 scenarios: true positive, false positive , true negative, false negative

Predict/True	True 'spam'	True 'spam'
Predict 'spam'	True Positive	False Positive
Predict 'not spam'	False Negative	True Negative

- Each scenario has a cost
- Expected cost of prediction:

$$\begin{aligned} E[C(\hat{y} = \text{spam})] = p(y_i = \text{spam}|x_i)C(\hat{y}_i = \text{spam}, y_i = \text{spam}) \\ + p(y_i = \text{not spam}|x_i)C(\hat{y}_i = \text{spam}, y_i = \text{not spam}) \end{aligned}$$

- if $E[C(\hat{y}_i = \text{spam})] > E[C(\hat{y}_i = \text{not spam})]$ then "not spam"

3 Decision Trees vs Naive Bayes

Decision Trees

1. Sequence of rules based on 1 feature
2. Training: 1 pass over data per depth
3. Greedy splitting as approximation
4. Testing: look at features in rules
5. New data: might need to change tree
6. Accuracy: good if simple rules based on individual features work ("symptoms")

Naive Bayes

1. Simultaneously combine all features
2. Training: 1p ass over data to count
3. Conditional independence assumption
4. Testing: look at all features
5. New data: update counts
6. Accuracy: good if features almost independent given label (text)

4 Parametric vs Non-Parametric

Parametric Models

- **fixed** # parameters: *size of model is* $O(1)$ *in terms of 'n'*
  - E.g., decision tree just stores rules
  - E.g., naive bayes just store counts
- more data $\rightarrow$ more accurate
- eventually more data doesn't help since model is too simple

Non-parametric Models

- *# parameters grow with 'n'* $\rightarrow$ size of model depends on 'n'
- more data $\rightarrow$ more complicated

5 K-Nearest Neighbours (KNN)

- non-parametric
- find 'k' training examples x_i that are most "similar" to $\hat{x}$
- classify using *most common label*
- assumption: objects with similar features likely have similar labels
- most common distance: *Euclidean distance*
- 'n' $\uparrow \rightarrow$ more complicated
- dimensionality
  - volume of space grows **exponentially** with dimension
  - **exponentially** more points needed
  - if 'd' is big, 'n' needs to be huge
- application: Optical Character Recognition

5.1 KNN Implementation

- no training phase ("lazy" learning)
- predictions are **expensive** ($O(nd)$)

6 Norms

- L_2 or "Euclidean" norm

$$\|r\|_2 = \sqrt{r_1^2 + r_2^2}$$

- L_1 or "Manhattan" norm

$$\|r\|_1 = |r_1| + |r_2|$$

- L_∞ or "max" norm

$$\|r\|_\infty = max\{|r_1|, |r_2|\}$$

6.1 Norms as Measures of Distance

- Euclidean

$$\|x-y\|_2 = \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2}$$

- Manhattan

$$\|x-y\|_1 = |x_1-y_1| + |x_2-y_2|$$

- max

$$\|x-y\|_\infty = max\{|x_1-y_1|, |x_2-y_2|\}$$

6.2 Norsm in d-Dimensions

- Euclidean

$$\|r\|_2 = \sqrt{\sum_{j=1}^{d} r_j^2}$$

- Manhattan

$$\|r\|_1 = \sum_{j=1}^{d} |r_j|$$

- max

$$\|r\|_\infty = max\{|r_j|\}$$