2/2/2023

A bit more naive bayes (Multinomial NB)

Input $x^{(i)}$ that is a list of words with length d_i

y	x
+1	great acting and score
-1	terrible directing
+1	great execution
-1	terrible
+1	amazing

total unique words = 8 = |V|

NB assumption:

$$P(x^{(i)} \mid y^{(i)}) = \prod_{j=1}^{x_i} P(x_j^{(i)} \mid y^{(i)})$$

e.g. $P(\text{"great directing"} \mid +1)$

$= P(\text{word 1} = \text{"great"} \mid +1) \cdot P(\text{word 2} = \text{"directing"} \mid +1)$

Additional Assumption: word position doesn't matter

$= P(\text{"great"} \mid +1) \cdot P(\text{"directing"} \mid +1)$

Step 1: Parameter Estimation (AKA training)

$P(y)$

$P(y=+1) = 3/5$

$P(y=-1) = 2/5$

$P(x \mid y)$

$y=+1$

$P(\text{"great"} \mid +1) = \frac{2+\lambda}{7+8\lambda}$

$P(\text{"acting"} \mid +1) = \frac{1+\lambda}{7+8\lambda}$

$\vdots$

$P(\text{"directing"} \mid +1) = \frac{0+\lambda}{7+8\lambda}$

$y=-1$

$P(\text{"terrible"} \mid -1) = \frac{2+\lambda}{3+8\lambda}$

$P(\text{"directing"} \mid -1) = \frac{1+\lambda}{3+8\lambda}$

$\vdots$

$P(\text{"great"} \mid -1) = \frac{0+\lambda}{3+8\lambda}$

Step 2: Inference (use $\lambda=1$)

Given input x = "great directing": compute $p(y \mid x = \text{"great directing"})$

$$= \frac{P(y) \cdot P(x=\text{"great directing"} \mid y)}{P(x=\text{"great directing"})}$$

Normalizing constant

$Y=+1$: $\frac{3}{5} \cdot \frac{3}{15} \cdot \frac{1}{15} = 0.008$

$P(y=+1)$ · $p(\text{"great"} \mid +1)$ · $P(\text{"directing"} \mid +1)$

$y=-1: \frac{2}{5} \cdot \frac{1}{11} \cdot \frac{2}{11} = 0.0066$

$P(y=+1 \mid \text{"great directing"}) = \frac{0.008}{0.008 + 0.0066} = 0.55$

If documents are long-ish, you have to multiply many small probabilities together

Numerical underflow i.e. on computer, everything just becomes zero

Solution: work in log space:

$y=+1$: compute $\log\left(P(y=+1) \cdot P(\text{"great directing"} \mid +1)\right)$

$= \log(3/5) + \log(3/15) + \log(1/15) \approx -2.1$

$y=-1: \log(2/5) + \log(1/11) + \log(2/11) \approx -2.2$

---

to get probabilities:

compute max log score = -2.1

subtract that from everything

$y=+1 = 0$

$y=-1 = -0.1$

then exponentiate

$y=+1 : 1$

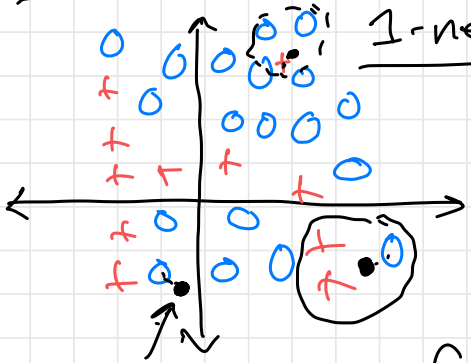
$y=-1 : e^{-0.1} \approx .9$

Finally normalize: $P(y=+1) = \frac{1}{1+.9}$

Announcements

- HW1 due Feb 7: Submit twice on Gradescope
- Project proposal due Feb 16
- Section tomorrow
  - Cross validation
  - Evaluation metrics

	model $p(y \mid x)$ Discriminative	model $p(y)p(x \mid y)$ Generative
Parametric You learn some parameters, afterwards you can throw away the training data	logistic regression Softmax regression w $w^{(1)}, \ldots, w^{(C)}$	Naive Bayes $P(y)$ AKA π $P(x \mid y)$ AKA μ, Σ's
Non-parametric You use training set when making predictions	K-Nearest Neighbors (or 1-Nearest Neighbors)	



test input

closest neighbor is O

⇒ predict O

1-nearest neighbor

↳ Predict same class as closest training example

e.g. Euclidean distance

Generalization:

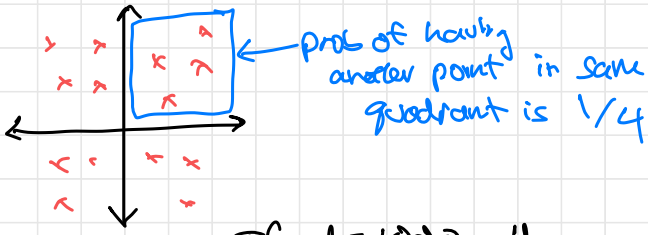
KNN

1) Find K nearest neighbors of test input x

2) Predict label that's most common among the neighbors

Curse of Dimensionality

In high dimensions, you very rarely have nearby neighbors ☹



If $d = 1000$, then chances of this are $\left(\frac{1}{2}\right)^{1000}$

No close neighbors to test data ⇒
using nearest neighbors to predict may not be very good