

# Naïve Bayes Classifiers

# Review

- Let event  $D$  = data we have observed.
- Let events  $H_1, \dots, H_k$  be events representing hypotheses we want to choose between.
- Use  $D$  to pick the "best"  $H$ .
- There are two "standard" ways to do this, depending on what information we have available.

# Maximum likelihood hypothesis

- The maximum likelihood hypothesis ( $H^{\text{ML}}$ ) is the hypothesis that maximizes the probability of the data given that hypothesis.

$$H^{\text{ML}} = \underset{i}{\operatorname{argmax}} P(D \mid H_i)$$

- How to use it: compute  $P(D \mid H_i)$  for each hypothesis and select the one with the greatest value.

# Maximum a posteriori (MAP) hypothesis

- The MAP hypothesis is the hypothesis that maximizes the posterior probability:

$$\begin{aligned} H^{\text{MAP}} &= \operatorname{argmax}_i P(H_i | D) \\ &= \operatorname{argmax}_i \frac{P(D | H_i)P(H_i)}{P(D)} \\ &\propto \operatorname{argmax}_i P(D | H_i)P(H_i) \end{aligned}$$

- The  $P(D | H_i)$  terms are now *weighted* by the hypothesis prior probabilities.

# Posterior probability

- If you need the actual posterior probability for some hypothesis  $H_0$ :

$$\begin{aligned} P(H_0 | D) &= \frac{P(D | H_0)P(H_0)}{P(D)} \\ &= \frac{P(D | H_0)P(H_0)}{\sum_i P(D, H_i)} \\ &= \frac{P(D | H_0)P(H_0)}{\sum_i P(D | H_i)P(H_i)} \end{aligned}$$

# Combining evidence

- If we have multiple pieces of data/evidence (say 2), then we need to compute or estimate

$$P(D_1, D_2 \mid H_0)$$

which is often hard.

- Instead, we assume all pieces of evidence are conditionally independent given a hypothesis:

$$P(D_1, D_2 \mid H_0) = P(D_1 \mid H_0)P(D_2 \mid H_0)$$

# Combining evidence

$$\begin{aligned} P(H_0 \mid D_1, \dots, D_m) &= \frac{P(D_1, \dots, D_m \mid H_0)P(H_0)}{P(D_1, \dots, D_m)} \\ &= \frac{\left[ P(D_1 \mid H_0) \cdots P(D_m \mid H_0) \right] P(H_0)}{P(D_1, \dots, D_m)} \\ &= \frac{\left[ \prod_{j=1}^m P(D_j \mid H_0) \right] P(H_0)}{P(D_1, \dots, D_m)} \end{aligned}$$

where

$$P(D_1, \dots, D_m) = \sum_{i=1}^k \left( \left[ \prod_{j=1}^m P(D_j \mid H_i) \right] P(H_i) \right)$$

# Classification

- Classification is the problem of identifying which of a set categories (called classes) a particular item belongs in.
- Lots of real-world problems are classification problems:
  - spam filtering (classes: spam/not-spam)
  - handwriting recognition & OCR (classes: one for each letter, number, or symbol)
  - text classification, image classification, music classification, etc.
- Almost any problem where you are assigning a label to items can be set up as a classification task.



# Classification

- An algorithm that does classification is called a classifier. Classifiers take an item as input and output the class it thinks that item belongs to. That is, the classifier *predicts* a class for each item.
- Lots of classifiers are based on probabilities and statistical inference:
  - The classes become the hypotheses being tested.
  - The item being classified is turned into a collection of data called features. Useful features are attributes of the item that are strongly correlated with certain classes.
  - The classification algorithm is usually ML or MAP, depending on what data we have available.

# Example: Spam classification

- New email arrives: is it spam or not spam?
- A useful set of features might be the presence or absence of various words in the email:
  - F1,  $\sim$ F1: "Kirlin" appears/does not appear
  - F2,  $\sim$ F2: "viagra" appears/does not appear
  - F3,  $\sim$ F3: "cash" appears/does not appear
- Let's say our new email contains "Kirlin" and "cash," but not "viagra."
- The features for this email are F1,  $\sim$ F2, and F3.
- Let's use MAP for classification.

# Example: Spam classification

- Features:  $F_1, \sim F_2, F_3$ .

$$H^{\text{MAP}} = \underset{i}{\operatorname{argmax}} P(D | H_i) P(H_i)$$

$$H^{\text{MAP}} = \underset{i \in \{\text{spam}, \text{not-spam}\}}{\operatorname{argmax}} P(F_1, \neg F_2, F_3 | H_i) P(H_i)$$

- But where do these probabilities come from?

# Learning probabilities from data

- To use MAP, we need to calculate or estimate  $P(H_i)$  and  $P(F_1, \sim F_2, F_3 \mid H_i)$  for each  $i$ .
- In other words, we need to know:
  - $P(\text{spam})$
  - $P(\text{not-spam})$
  - $P(F_1, \sim F_2, F_3 \mid \text{spam})$
  - $P(F_1, \sim F_2, F_3 \mid \text{not-spam})$

# Learning probabilities from data

- Let's assume we have access to a large number of old emails that are correctly labeled as spam/not-spam.
- How can we estimate  $P(\text{spam})$ ?

$$P(\text{spam}) = \frac{\# \text{ of emails labeled as spam}}{\text{total } \# \text{ of emails}}$$

# Learning probabilities from data

- Let's assume we have access to a large number of old emails that are correctly labeled as spam/not-spam.
- How can we estimate  $P(F_1, \sim F_2, F_3 \mid \text{spam})$ ?

$$P(F_1, \neg F_2, F_3 \mid \text{spam}) = \frac{\# \text{ of spam emails with those exact features}}{\text{total } \# \text{ of spam emails}}$$

- Why is this probably going to be a very rough estimate?

## Conditional independence to the rescue!

- It is unlikely that our set of old emails contains many messages with that exact set of features.
- Let's make an assumption that all of our features are conditionally independent of each other, given the hypothesis (spam/not-spam).

$$P(F_1, \neg F_2, F_3 \mid \text{spam}) =$$

$$P(F_1 \mid \text{spam}) \cdot P(\neg F_2 \mid \text{spam}) \cdot P(F_3 \mid \text{spam})$$

- These probabilities are easier to get good estimates for!
- A classifier that makes this assumption is called a Naïve Bayes classifier.

# Learning probabilities from data

- So now we need to estimate  $P(F_1 \mid \text{spam})$  instead of  $P(F_1, \sim F_2, F_3 \mid \text{spam})$ .
- Equivalently, how can we estimate the probability of seeing "Kirlin" in an email given that the email is spam?

$$P(F_1 \mid \text{spam}) = \frac{\# \text{ of spam emails with the word Kirlin}}{\text{total } \# \text{ of spam emails}}$$



# Another problem to handle...

- What if we see a word we've never encountered before? What happens to its probability estimate? (and why is this bad?)

$$P(F_j \mid \text{spam}) = \frac{\# \text{ of spam emails with word } F_j}{\text{total } \# \text{ of spam emails}}$$

$$P(\text{spam} \mid F_1, \dots, F_m) = \frac{\left[ \prod_{j=1}^m P(F_j \mid \text{spam}) \right] P(\text{spam})}{P(F_1, \dots, F_m)}$$

- Probability of zero destroys the entire calculation!

# Another problem to handle...

- Fix the estimates:

$$P(F_j | \text{spam}) = \frac{\# \text{ of spam emails with word } F_j + 1}{\text{total } \# \text{ of spam emails} + 2}$$

- This is called ***smoothing***. Removes the possibility of a zero probability wiping out the entire calculation.
- "Simulates" two additional spam emails, one with every word, and one with no words.

# Summary of Naïve Bayes

- Naïve Bayes classifies using MAP:

$$\begin{aligned} H^{\text{MAP}} &= \underset{i}{\operatorname{argmax}} P(D \mid H_i)P(H_i) \\ &= \underset{i \in \{\text{spam}, \text{not-spam}\}}{\operatorname{argmax}} P(F_1, \dots, F_m \mid H_i)P(H_i) \\ &= \underset{i \in \{\text{spam}, \text{not-spam}\}}{\operatorname{argmax}} \left[ P(F_1 \mid H_i) \cdots P(F_m \mid H_i) \right] P(H_i) \\ &= \underset{i \in \{\text{spam}, \text{not-spam}\}}{\operatorname{argmax}} \left[ \prod_{j=1}^m P(F_j \mid H_i) \right] P(H_i) \end{aligned}$$

- Compute this for spam and for not-spam; see which is bigger.

# Summary of Naïve Bayes

- Estimating the **prior** for each hypothesis:

$$P(H_i) = \frac{\# \text{ of emails labeled as } H_i}{\text{total } \# \text{ of emails}}$$

- Estimating the probability of a feature given a class (aka **likelihood**):

$$P(F_j | H_i) = \frac{\# \text{ of } H_i \text{ emails with word } F_j + 1}{\text{total } \# \text{ of } H_i \text{ emails} + 2}$$