CMPSCI 240: "Reasoning Under Uncertainty"

Lecture 20x

Not-A-Prof. Phil Kirlin
pkirlin@cs.umass.edu

April 5, 2012

Bayesian Reasoning (Recap)

- The maximum likelihood hypothesis is the hypothesis that assigns the highest probability to the observed data:

$$H^{\mathrm{ML}} = \operatorname*{argmax}_i P(D \mid H_i)$$

- The maximum a posteriori (MAP) hypothesis is the hypothesis that that maximizes the posterior probability given D:

$$\begin{aligned} H^{\mathrm{MAP}} &= \operatorname*{argmax}_i P(H_i \mid D) \\ &= \operatorname*{argmax}_i \frac{P(D \mid H_i)\, P(H_i)}{P(D)} \\ &\propto \operatorname*{argmax}_i P(D \mid H_i)\, P(H_i) \end{aligned}$$

- $P(H_i)$ is called the prior probability (or just prior).
- $P(H_i|D)$ is called the posterior probability.

Independent Pieces of Data (Recap)

Definition

If we have 2 pieces of data D_1 and D_2 that are are conditionally independent given H_i, then the probability of $D_1 \cap D_2$ given H_i is

$$\begin{aligned} P(D_1 \cap D_2 \mid H_i) &= P(D_1 \mid H_i) P(D_2 \mid D_1, H_i) \\ &= P(D_1 \mid H_i) P(D_2 \mid H_i) \end{aligned}$$

Independent Pieces of Data (Recap)

Definition

If we have 2 pieces of data D_1 and D_2 that are are conditionally independent given H_i, then the probability of $D_1 \cap D_2$ given H_i is

$$
\begin{aligned}
P(D_1 \cap D_2 \mid H_i) &= P(D_1 \mid H_i) P(D_2 \mid D_1, H_i) \\
&= P(D_1 \mid H_i) P(D_2 \mid H_i)
\end{aligned}
$$

If we have m conditionally independent pieces of data $D_1, \ldots, D_m$, then

Independent Pieces of Data (Recap)

Definition

If we have 2 pieces of data D_1 and D_2 that are are conditionally independent given H_i, then the probability of $D_1 \cap D_2$ given H_i is

$$\begin{aligned} P(D_1 \cap D_2 \,|\, H_i) &= P(D_1 \,|\, H_i) P(D_2 \,|\, D_1, H_i) \\ &= P(D_1 \,|\, H_i) P(D_2 \,|\, H_i) \end{aligned}$$

If we have m conditionally independent pieces of data $D_1, \ldots, D_m$, then

$$P(D_1 \cap \ldots \cap D_m \,|\, H_i) = \prod_{j=1}^{m} P(D_j \,|\, H_i)$$

Combining Evidence (Recap)

Definition

If we have k disjoint, exhaustive hypotheses $H_1, \ldots, H_k$ (e.g., rainy, dry) and m conditionally independent pieces of observed data $D_1, \ldots, D_m$, then the posterior probability $P(H_i \mid D_1 \cap \ldots \cap D_m)$ of hypothesis H_i ($i = 1, \ldots, k$) given the observed data $D_1 \cap \ldots \cap D_m$ is:

Combining Evidence (Recap)

Definition

If we have k disjoint, exhaustive hypotheses $H_1, \ldots, H_k$ (e.g., rainy, dry) and m conditionally independent pieces of observed data $D_1, \ldots, D_m$, then the posterior probability $P(H_i \mid D_1 \cap \ldots \cap D_m)$ of hypothesis H_i $(i = 1, \ldots, k)$ given the observed data $D_1 \cap \ldots \cap D_m$ is:

$$P(H_i \mid D_1 \cap \ldots \cap D_m) = \frac{\left(\prod_{j=1}^{m} P(D_j \mid H_i)\right) P(H_i)}{P(D_1 \cap \ldots \cap D_m)}$$

where

Combining Evidence (Recap)

Definition

If we have k disjoint, exhaustive hypotheses $H_1, \ldots, H_k$ (e.g., rainy, dry) and m conditionally independent pieces of observed data $D_1, \ldots, D_m$, then the posterior probability $P\left(H_i \mid D_1 \cap \ldots \cap D_m\right)$ of hypothesis H_i $(i = 1, \ldots, k)$ given the observed data $D_1 \cap \ldots \cap D_m$ is:

$$P\left(H_i \mid D_1 \cap \ldots \cap D_m\right) = \frac{\left(\prod_{j=1}^{m} P\left(D_j \mid H_i\right)\right) P\left(H_i\right)}{P\left(D_1 \cap \ldots \cap D_m\right)}$$

where

$$P(D_1 \cap \ldots \cap D_m) = \sum_{i=1}^{k} \left(\prod_{j=1}^{m} P(D_j \mid H_i)\right) P(H_i)$$

Classification

- Classification is the problem of identifying which of a set of categories (called classes) a particular item belongs.

Classification

- Classification is the problem of identifying which of a set of categories (called classes) a particular item belongs.
- Lots of real-world problems can be set up as classification tasks:

Classification

- Classification is the problem of identifying which of a set of categories (called classes) a particular item belongs.
- Lots of real-world problems can be set up as classification tasks:
  - Spam filtering (classes: spam, not spam)

Classification

- Classification is the problem of identifying which of a set of categories (called classes) a particular item belongs.
- Lots of real-world problems can be set up as classification tasks:
  - Spam filtering (classes: spam, not spam)
  - Handwriting recognition & OCR (classes: one for each letter, number, or symbol)

Classification

- Classification is the problem of identifying which of a set of categories (called classes) a particular item belongs.
- Lots of real-world problems can be set up as classification tasks:
  - Spam filtering (classes: spam, not spam)
  - Handwriting recognition & OCR (classes: one for each letter, number, or symbol)
  - Text classification, image classification, music classification, etc.

Classification

- Classification is the problem of identifying which of a set of categories (called classes) a particular item belongs.
- Lots of real-world problems can be set up as classification tasks:
  - 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 some sort of label to items can be set up as a classification task.

Classification

- An algorithm that does classification is called a classifier. Classifiers take some sort of item as input and output the class it thinks that item belongs to.

Classification

- An algorithm that does classification is called a classifier. Classifiers take some sort of item as input and output the class it thinks that item belongs to.
- Lots of classifiers are based on Bayesian reasoning:

Classification

- An algorithm that does classification is called a classifier. Classifiers take some sort of item as input and output the class it thinks that item belongs to.
- Lots of classifiers are based on Bayesian reasoning:
  - The classes become the hypotheses that are being tested.

Classification

- An algorithm that does classification is called a classifier. Classifiers take some sort of item as input and output the class it thinks that item belongs to.
- Lots of classifiers are based on Bayesian reasoning:
    - The classes become the hypotheses that are being tested.
    - The item being classified is turned into a collection of data called features. Useful features are attributes of the item that imply a strong connection to certain classes.

Classification

- An algorithm that does classification is called a classifier. Classifiers take some sort of item as input and output the class it thinks that item belongs to.
- Lots of classifiers are based on Bayesian reasoning:
  - The classes become the hypotheses that are being tested.
  - The item being classified is turned into a collection of data called features. Useful features are attributes of the item that imply a strong connection to certain classes.
  - The classification algorithm is typically either maximum likelihood or MAP, depending on what data we have available.

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).
- A useful set of features might be events corresponding to whether or not certain words appear in the email:

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).
- A useful set of features might be events corresponding to whether or not certain words appear in the email:
  - F_1, F_1^c : "Wallach" appears/does not appear in the email

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).
- A useful set of features might be events corresponding to whether or not certain words appear in the email:
  - F_1, F_1^c : "Wallach" appears/does not appear in the email
  - F_2, F_2^c : "viagra" appears/does not appear in the email

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).
- A useful set of features might be events corresponding to whether or not certain words appear in the email:
  - F_1, F_1^c : "Wallach" appears/does not appear in the email
  - F_2, F_2^c : "viagra" appears/does not appear in the email
  - F_3, F_3^c : "cash" appears/does not appear in the email

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).
- A useful set of features might be events corresponding to whether or not certain words appear in the email:
  - F_1, F_1^c : "Wallach" appears/does not appear in the email
  - F_2, F_2^c : "viagra" appears/does not appear in the email
  - F_3, F_3^c : "cash" appears/does not appear in the email
- Let's say this email contains the words "Wallach" and "cash," but not "viagra."

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).
- A useful set of features might be events corresponding to whether or not certain words appear in the email:
  - F_1, F_1^c : "Wallach" appears/does not appear in the email
  - F_2, F_2^c : "viagra" appears/does not appear in the email
  - F_3, F_3^c : "cash" appears/does not appear in the email
- Let's say this email contains the words "Wallach" and "cash," but not "viagra."
- Therefore, the features for this email are F_1, F_2^c, and F_3.

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).
- A useful set of features might be events corresponding to whether or not certain words appear in the email:
  - F_1, F_1^c : "Wallach" appears/does not appear in the email
  - F_2, F_2^c : "viagra" appears/does not appear in the email
  - F_3, F_3^c : "cash" appears/does not appear in the email
- Let's say this email contains the words "Wallach" and "cash," but not "viagra."
- Therefore, the features for this email are F_1, F_2^c, and F_3.
- If we use the MAP rule for classification, we need to compute

$$\begin{aligned} H^{\text{MAP}} &= \underset{i}{\operatorname{argmax}}\, P(D \mid H_i)\, P(H_i) \\ &= \underset{i \in \{spam, notspam\}}{\operatorname{argmax}} P(F_1 \cap F_2^c \cap F_3 \mid H_i)\, P(H_i) \end{aligned}$$

Example: Spam Classification

- When a new email arrives, we want to label it as either spam or not spam (our two classes or hypotheses).
- A useful set of features might be events corresponding to whether or not certain words appear in the email:
  - F_1, F_1^c : "Wallach" appears/does not appear in the email
  - F_2, F_2^c : "viagra" appears/does not appear in the email
  - F_3, F_3^c : "cash" appears/does not appear in the email
- Let's say this email contains the words "Wallach" and "cash," but not "viagra."
- Therefore, the features for this email are F_1, F_2^c, and F_3.
- If we use the MAP rule for classification, we need to compute

$$\begin{aligned} H^{\text{MAP}} &= \operatorname*{argmax}_i P(D \mid H_i)\, P(H_i) \\ &= \operatorname*{argmax}_{i \in \{spam, notspam\}} P(F_1 \cap F_2^c \cap F_3 \mid H_i)\, P(H_i) \end{aligned}$$

- But where do these probabilities come from?

Learning Probabilities From Data

- To use MAP, we need probabilities for $P(H_i)$; that is, $P(spam)$ and $P(not\ spam)$, as well as $P(F_1 \cap F_2^c \cap F_3 \mid H_i)$.

Learning Probabilities From Data

- To use MAP, we need probabilities for $P(H_i)$; that is, $P(spam)$ and $P(not\ spam)$, as well as $P(F_1 \cap F_2^c \cap F_3 \mid H_i)$.
- We can estimate these probabilities if we have access to a lot of email that has already been classified as spam or not spam.

Learning Probabilities From Data

- To use MAP, we need probabilities for $P(H_i)$; that is, $P(spam)$ and $P(not\ spam)$, as well as $P(F_1 \cap F_2^c \cap F_3 \mid H_i)$.
- We can estimate these probabilities if we have access to a lot of email that has already been classified as spam or not spam.
- How can we estimate $P(spam)$?

Learning Probabilities From Data

- To use MAP, we need probabilities for $P(H_i)$; that is, $P(spam)$ and $P(not\ spam)$, as well as $P(F_1 \cap F_2^c \cap F_3 \mid H_i)$.
- We can estimate these probabilities if we have access to a lot of email that has already been classified as spam or not spam.
- How can we estimate $P(spam)$?
- $P(spam) = \dfrac{\#\text{ of emails labeled as spam}}{\#\text{ of total emails}}$

Learning Probabilities From Data

- To use MAP, we need probabilities for $P(H_i)$; that is, $P(spam)$ and $P(not\ spam)$, as well as $P(F_1 \cap F_2^c \cap F_3 \mid H_i)$.
- We can estimate these probabilities if we have access to a lot of email that has already been classified as spam or not spam.
- How can we estimate $P(spam)$?
- $P(spam) = \frac{\#\text{ of emails labeled as spam}}{\#\text{ of total emails}}$
- How can we estimate $P(F_1 \cap F_2^c \cap F_3 \mid spam)$?

Learning Probabilities From Data

- To use MAP, we need probabilities for $P(H_i)$; that is, $P(spam)$ and $P(not\ spam)$, as well as $P(F_1 \cap F_2^c \cap F_3 \mid H_i)$.
- We can estimate these probabilities if we have access to a lot of email that has already been classified as spam or not spam.
- How can we estimate $P(spam)$?
- $P(spam) = \dfrac{\#\text{ of emails labeled as spam}}{\#\text{ of total emails}}$
- How can we estimate $P(F_1 \cap F_2^c \cap F_3 \mid spam)$?
- $P(F_1 \cap F_2^c \cap F_3 \mid spam) = \dfrac{\#\text{ of emails labeled as spam with those exact features}}{\#\text{ of total spam emails}}$

Learning Probabilities From Data

- To use MAP, we need probabilities for $P(H_i)$; that is, $P(spam)$ and $P(not\ spam)$, as well as $P(F_1 \cap F_2^c \cap F_3 \mid H_i)$.
- We can estimate these probabilities if we have access to a lot of email that has already been classified as spam or not spam.
- How can we estimate $P(spam)$?
- $P(spam) = \dfrac{\#\text{ of emails labeled as spam}}{\#\text{ of total emails}}$
- How can we estimate $P(F_1 \cap F_2^c \cap F_3 \mid spam)$?
- $P(F_1 \cap F_2^c \cap F_3 \mid spam) = \dfrac{\#\text{ of emails labeled as spam with those exact features}}{\#\text{ of total spam emails}}$
- Why is that last estimate going to be a problem?

Conditional Independence to the Rescue!

Conditional Independence to the Rescue!

- It is unlikely that we would ever have enough email to get a good estimate of $P(F_1 \cap F_2^c \cap F_3 \mid spam)$ using the previous idea because the number of emails in our collection *with the exact same feature set as our new email* is probably very small, or zero.

Conditional Independence to the Rescue!

- It is unlikely that we would ever have enough email to get a good estimate of $P(F_1 \cap F_2^c \cap F_3 \,|\, spam)$ using the previous idea because the number of emails in our collection *with the exact same feature set as our new email* is probably very small, or zero.
- Therefore, we will assume all our features are conditionally independent of each other, given the hypothesis (spam or not spam).

Conditional Independence to the Rescue!

- It is unlikely that we would ever have enough email to get a good estimate of $P(F_1 \cap F_2^c \cap F_3 \mid spam)$ using the previous idea because the number of emails in our collection *with the exact same feature set as our new email* is probably very small, or zero.
- Therefore, we will assume all our features are conditionally independent of each other, given the hypothesis (spam or not spam).
- Therefore,
  $P(F_1 \cap F_2^c \cap F_3 \mid spam) =$
  $P(F_1 \mid spam) \cdot P(F_2^c \mid spam) \cdot P(F_3 \mid spam)$

Conditional Independence to the Rescue!

- It is unlikely that we would ever have enough email to get a good estimate of $P(F_1 \cap F_2^c \cap F_3 \mid spam)$ using the previous idea because the number of emails in our collection *with the exact same feature set as our new email* is probably very small, or zero.
- Therefore, we will assume all our features are conditionally independent of each other, given the hypothesis (spam or not spam).
- Therefore,
  $P(F_1 \cap F_2^c \cap F_3 \mid spam) =$
  $P(F_1 \mid spam) \cdot P(F_2^c \mid spam) \cdot P(F_3 \mid spam)$
- Those probabilities are easier to get good estimates for!

Conditional Independence to the Rescue!

- It is unlikely that we would ever have enough email to get a good estimate of $P(F_1 \cap F_2^c \cap F_3 \,|\, spam)$ using the previous idea because the number of emails in our collection *with the exact same feature set as our new email* is probably very small, or zero.
- Therefore, we will assume all our features are conditionally independent of each other, given the hypothesis (spam or not spam).
- Therefore,
  $P(F_1 \cap F_2^c \cap F_3 \,|\, spam) =$
  $P(F_1 \,|\, spam) \cdot P(F_2^c \,|\, spam) \cdot P(F_3 \,|\, spam)$
- Those probabilities are easier to get good estimates for!
- A classifier that makes this assumption is called a Naive Bayes classifier.

Learning Probabilities From Data

Learning Probabilities From Data

- How would we estimate $P(F_1 \mid spam)$, or equivalently, the probability an email contains the word "Wallach," given that it's a spam email? (Remember, we have a lot of existing emails already classified as spam or not spam.)

Learning Probabilities From Data

- How would we estimate $P(F_1 \mid spam)$, or equivalently, the probability an email contains the word "Wallach," given that it's a spam email? (Remember, we have a lot of existing emails already classified as spam or not spam.)
- $P(F_1 \mid spam) = \dfrac{\#\text{ of emails labeled as spam containing the word Wallach}}{\#\text{ of total spam emails}}$

Learning Probabilities From Data

- How would we estimate $P(F_1 \mid spam)$, or equivalently, the probability an email contains the word "Wallach," given that it's a spam email? (Remember, we have a lot of existing emails already classified as spam or not spam.)
- $P(F_1 \mid spam) = \frac{\text{\# of emails labeled as spam containing the word Wallach}}{\text{\# of total spam emails}}$
- Spam filters typically operate so every word in an email is its own feature. What happens if we see a word we've never encountered before?

Learning Probabilities From Data

- How would we estimate $P(F_1 \mid spam)$, or equivalently, the probability an email contains the word "Wallach," given that it's a spam email? (Remember, we have a lot of existing emails already classified as spam or not spam.)
- $P(F_1 \mid spam) = \dfrac{\#\text{ of emails labeled as spam containing the word Wallach}}{\#\text{ of total spam emails}}$
- Spam filters typically operate so every word in an email is its own feature. What happens if we see a word we've never encountered before?
- $P(F_1 \mid spam) = \dfrac{\#\text{ of emails labeled as spam containing the word Wallach} + 1}{\#\text{ of total spam emails} + 2}$

Learning Probabilities From Data

- How would we estimate $P(F_1 \mid spam)$, or equivalently, the probability an email contains the word "Wallach," given that it's a spam email? (Remember, we have a lot of existing emails already classified as spam or not spam.)
- $P(F_1 \mid spam) = \dfrac{\text{\# of emails labeled as spam containing the word Wallach}}{\text{\# of total spam emails}}$
- Spam filters typically operate so every word in an email is its own feature. What happens if we see a word we've never encountered before?
- $P(F_1 \mid spam) = \dfrac{\text{\# of emails labeled as spam containing the word Wallach} + 1}{\text{\# of total spam emails} + 2}$
- This is called smoothing, and it removes the chance that a zero probability will wipe out the entire calculation.

Summary of Naive Bayes Classification

- The email can be classified by computing:

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

Summary of Naive Bayes Classification

- The email can be classified by computing:

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

- In other words, compute likelihood $\times$ prior for each hypothesis (spam vs. not spam) and see which has a greater value

Summary

- Estimate the priors using:

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

Summary

- Estimate the priors using:

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

- Estimate the probability of a feature given a class using:

$$P(F_j \mid H_i) = \frac{\#\text{ of emails labeled as } H_i \text{ containing } F_j + 1}{\#\text{ of emails labeled as } H_i + 2}$$