Announcements – Assignments

- **Readings 04:**
  - link posted to course site
  - due Sunday

- **HW 02:**
  - Due Wednesday night (last night)

- **HW 03:**
  - Released today
  - Due next Wednesday night
Final Project – Deliverables

- **Project ideation – Friday May 28\textsuperscript{st}**
  - [https://www.overleaf.com/read/yzpgxcgsgvdvp](https://www.overleaf.com/read/yzpgxcgsgvdvp)

- roughly 250 word overview of what you are interested in
Final Project – Deliverables

- Project ideation – Friday May 28th
  • 5 points

- Project proposal – Friday June 4th – Sunday June 6th
  • 9 points

- Project presentations – Monday June 14th
  • 6 points

- Project submissions – Friday June 18th
  • 15 points

http://coms2710.barnard.edu/final_project
When computing the same thing across a row or column, what should we do?

1. Define a function
2. apply the function

Looping through a dataframe is not ideal
“A computer program does what you tell it to do, not what you want it to do.”

Be careful when looping and adding to lists
A mathematical model calculated based on sample data ("training data") makes predictions or decisions without being explicitly programmed to perform the task.
Different Types of Machine Learning

- **Supervised Learning**
  - Learn rule from data and answers

- **Unsupervised Learning**
  - Learn a rule for patterns from data

- **Reinforcement Learning**
  - try your rule on a piece of data, and get feedback on how good your rule was
Prediction
Based on incomplete information

One way of making predictions:

- To predict an outcome for an individual,
- find others who are like that individual
- and whose outcomes you know.
- Use those outcomes as the basis of your prediction.
Two types of predictions: Classification & Regression

Classification = Categorical
Regression = Numeric

Predicting sentiment:
- Classification
  👍  👎
- Regression: [-1, ..., 1]
Prediction Example: Hot dog or not Hot dog?
<table>
<thead>
<tr>
<th>Email/Username</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>David, Adam</td>
<td>Tennis this week? - in playing tennis on Tuesday. It &gt;&gt;&gt;&gt; will b...</td>
</tr>
<tr>
<td>Citi Alerts</td>
<td>Your Citibank account statement is available online - com to y...</td>
</tr>
<tr>
<td>Humane Rescue Allia.</td>
<td>Your HRA E-Newsletter - Read news and events updates from ...</td>
</tr>
<tr>
<td>SLEEP NUMBER</td>
<td>Check out these limited-time Weekend Specials - PLUS get fre...</td>
</tr>
<tr>
<td>aishagaddafi11119</td>
<td>Inquiry for Investment. - Inquiry for Investment. Assalamu Alai...</td>
</tr>
</tbody>
</table>
What is this medical article about?

MEDLINE Article

MeSH Subject Category Hierarchy

- Antagonists and Inhibitors
- Blood Supply
- Chemistry
- Drug Therapy
- Embryology
- Epidemiology
- ...
...zany characters and richly applied satire, and some great plot twists

It was pathetic. The worst part about it was the boxing scenes...

...awesome caramel sauce and sweet toasty almonds. I love this place!

...awful pizza and ridiculously overpriced...
Broad applications of sentiment analysis

- **Movie**: is this review positive or negative?
- **Products**: what do people think about the new iPhone?
- **Public sentiment**: how is consumer confidence?
- **Politics**: what do people think about this candidate or issue?
- **Prediction**: predict election outcomes or market trends from sentiment
Text Classification

Input:
- a document $d$
- a fix set of classes $C = \{c_1, c_2, \ldots, c_n\}$
- A training set of $n$ labeled documents $(d_1, c_1), (d_2, c_2), \ldots, (d_n, c_n)$

Output:
- A learned classifier $f$
  - $f$ is a mapping from $d \rightarrow c$
Attributes (features) of an example

Classifier

Predicted label of the example
Setup for training and evaluating a classifier

Data → Sample → Labels

Model association between attributes and labels

Training Set → Test Set

Estimate classifier’s performance
scikit-learn uses a standard set of functions for all models

The two main ones for our purposes

- `model.fit(X, y)` — train the model with the given data set
- `model.predict(X_test)` — get predictions for the given test set
Different types of classifiers

- Neural Networks
- K-Nearest Neighbors
- Logistic Regression
- Naive Bayes
- ....
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!
Classify document based on BoW

What is the probability of the class given the BoW

\[ f(\text{seen}) = 2 \]
\[ f(\text{sweet}) = 1 \]
\[ f(\text{whimsical}) = 1 \]
\[ f(\text{recommend}) = 1 \]
\[ f(\text{happy}) = 1 \]
\[ \ldots \]

\[ \text{ Class } = C \]
\[ P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)} \]
Bayes Rule for documents and classes

Given document $d$, what is the probability of category $c$

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}$$
Choose category $c$ that has the highest probability given document $d$

$$
c_{MAP} = \arg\max_{c \in C} P(c \mid d)$$

$$= \arg\max_{c \in C} \frac{P(d \mid c)P(c)}{P(d)}$$

$$= \arg\max_{c \in C} P(d \mid c)P(c)$$

MAP is “maximum a posteriori” = most likely class

Bayes Rule

Dropping the denominator

Slide from Dan Jurafsky
Choose category $c$ that has the highest probability given document $d$

$$c_{MAP} = \arg\max_{c \in C} P(d \mid c) P(c)$$

How do we represent document $d$?

Answer: Bag of Words

$$= \arg\max_{c \in C} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$
Naive Bayes Independent Assumption

\[ P(x_1, x_2, \ldots, x_n \mid c) \]

- **Bag of Words assumption**: Assume position doesn’t matter
- **Conditional Independence**: Assume the probabilities \( P(x_i \mid c_j) \) are independent given the class \( c \).

\[ P(x_1, \ldots, x_n \mid c) = P(x_1 \mid c) \cdot P(x_2 \mid c) \cdot P(x_3 \mid c) \cdot \ldots \cdot P(x_n \mid c) \]

Plugging this into our prediction equation:

\[ c_{MAP} = \arg\max_{c \in C} P(x_1, x_2, \ldots, x_n \mid c)P(c) \]

\[ c_{NB} = \arg\max_{c \in C} P(c) \prod_{x \in X} P(x \mid c) \]
\[ c_{NB} = \arg \max_{c \in C} P(c_j) \prod_{x \in X} P(x | c) \]

Count Frequencies in training data
\[ c_{NB} = \arg \max_{c \in C} P(c_j) \prod_{x \in X} P(x \mid c) \]

Count Frequencies in training data

\[ \hat{P}(c_j) = \]

\[ \hat{P}(x_i \mid c_j) = \]
Computing probabilities

\[ c_{NB} = \arg\max_{c \in C} P(c_j) \prod_{x \in X} P(x \mid c) \]

Count Frequencies in training data

\[ \hat{P}(c_j) = \frac{N_{c_j}}{N_{total}} \]

\[ \hat{P}(x_i \mid c_j) = \]
Computing probabilities

\[ c_{NB} = \arg\max_{c \in C} P(c_j) \prod_{x \in X} P(x \mid c) \]

Count Frequencies in training data

\[ \hat{P}(c_j) = \frac{N_{c_j}}{N_{total}} \]

\[ \hat{P}(x_i \mid c_j) = \frac{\text{count}(x_i, c_i)}{\sum_{x \in V} \text{count}(x, c_i)} \]

fraction of times word \( x_i \) appears among all words in documents of topic \( c_i \)
\[ c_{NB} = \arg\max_{c \in C} P(c_j) \prod_{x \in X} P(x \mid c) \]

Count Frequencies in training data

\[ \hat{P}(c_j) = \frac{N_{c_j}}{N_{total}} \]

\[ \hat{P}(x_i \mid c_j) = \frac{\text{count}(x_i, c_i)}{\sum_{x \in V} \text{count}(x, c_i)} \]

fraction of times word \( x_i \) appears among all words in documents of topic \( c_i \)

Maximum Likelihood Estimation

Slide from Dan Jurafsky
Question

What if we have seen no training positive documents with the word fantastic?

\[
\hat{P}("\text{fantastic}\mid\text{positive}) = \frac{\text{count("fantastic", positive)}}{\sum_{w \in V} \text{count}(w, \text{positive})} = 0
\]

Probability of class will be 0, regardless of other words

\[
c_{NB} = \arg\max_{c \in C} P(c_{j}) \prod_{x \in X} P(x \mid c)
\]
Smoothing

\[ P(x_i|c_j) = \frac{\text{count}(x_i, c_i) + 1}{\sum_{x \in V}(\text{count}(x, c_i) + 1)} \]

\[ = \frac{\text{count}(x_i, c_i) + 1}{(\sum_{x \in V} \text{count}(x, c_i)) + |V|} \]

Laplacian smoothing (add 1)
Learning a Naive Bayes Classifier

- From training corpus, extract Vocabulary

- Calculate $P(c_j)$ terms
  - For each $c_j$ in $C$ do
    
    $$docs_j \leftarrow \text{all docs with class } c_j$$
    
    $$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total # documents}|}$$

- Calculate $P(w_k \mid c_j)$ terms
  
  - $Text_j \leftarrow \text{single doc containing all } docs_j$
  
  - For each word $w_k$ in Vocabulary
    
    $$n_k \leftarrow \# \text{ of occurrences of } w_k \text{ in } Text_j$$
    
    $$P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha \mid \text{Vocabulary} \mid}$$

Slide from Dan Jurafsky
Predict with Naive Bayes Classifier

Give a document of composed of words $X$ choose the class $c$ that maximizes the Naive Bayes equation

$$c_{NB} = \arg\max_{c \in C} P(c_j) \prod_{x \in X} P(x \mid c)$$
scikit-learn uses a standard set of functions for all models

The two main ones for our purposes

- `model.fit(X, y)` — train the model with the given data set
- `model.predict(X_test)` — get predictions for the given test set
Consideration in Naive Bayes

Unknown Words
- words that are not in our training data but are in our test data
- Ignore them
  - Pretend they are not in our test

Stop Words
- For NB, removing them doesn’t usually help
Naive Bayes Example
Let's walk through an example of training and testing naive Bayes with add-one smoothing. We'll use a sentiment analysis domain with the two classes positive (+) and negative (-), and take the following miniature training and test documents simplified from actual movie reviews.

<table>
<thead>
<tr>
<th>Cat</th>
<th>Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>just plain boring</td>
</tr>
<tr>
<td></td>
<td>entirely predictable and lacks energy</td>
</tr>
<tr>
<td></td>
<td>no surprises and very few laughs</td>
</tr>
<tr>
<td></td>
<td>very powerful</td>
</tr>
<tr>
<td></td>
<td>the most fun film of the summer</td>
</tr>
<tr>
<td>+</td>
<td>very powerful</td>
</tr>
<tr>
<td>+</td>
<td>the most fun film of the summer</td>
</tr>
<tr>
<td>Test</td>
<td>? predictable with no fun</td>
</tr>
</tbody>
</table>

The prior $P(c)$ for the two classes is computed via Eq. 4.11 as:

- $P(+)$ = $\frac{3}{5}$
- $P(-)$ = $\frac{2}{5}$

The word 'with' doesn't occur in the training set, so we drop it completely (as mentioned above, we don't use unknown word models for naive Bayes). The likelihoods from the training set for the remaining three words 'predictable', 'no', and 'fun', are as follows, from Eq. 4.14 (computing the probabilities for the remainder of the words in the training set is left as an exercise for the reader):

- $P(\text{predictable} | +) = \frac{0 + 1}{20 + 20} = \frac{1}{40}$
- $P(\text{predictable} | -) = \frac{1 + 1}{20 + 20} = \frac{1}{40}$
- $P(\text{no} | +) = \frac{0 + 1}{20 + 20} = \frac{1}{40}$
- $P(\text{no} | -) = \frac{1 + 1}{20 + 20} = \frac{1}{40}$
- $P(\text{fun} | +) = \frac{1 + 1}{20 + 20} = \frac{1}{40}$
- $P(\text{fun} | -) = \frac{0 + 1}{20 + 20} = \frac{1}{40}$

For the test sentence $S = \text{predictable with no fun}$, after removing the word 'with', the chosen class, via Eq. 4.9, is therefore computed as follows:

- $P(+ | S) = \frac{3 \times 2 \times 1}{40 \times 40} = \frac{6}{1000}$
- $P(- | S) = \frac{2 \times 1 \times 2}{40 \times 40} = \frac{4}{1000}$

The model thus predicts the class negative for the test sentence.
4.3 Worked example

Let's walk through an example of training and testing naive Bayes with add-one smoothing. We'll use a sentiment analysis domain with the two classes positive (+) and negative (-), and take the following miniature training and test documents simplified from actual movie reviews.

<table>
<thead>
<tr>
<th>Cat</th>
<th>Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>- just plain boring</td>
</tr>
<tr>
<td></td>
<td>- entirely predictable and lacks energy</td>
</tr>
<tr>
<td></td>
<td>- no surprises and very few laughs</td>
</tr>
<tr>
<td></td>
<td>+ very powerful</td>
</tr>
<tr>
<td></td>
<td>+ the most fun film of the summer</td>
</tr>
<tr>
<td>Test</td>
<td>? predictable with no fun</td>
</tr>
</tbody>
</table>

1. Prior from training:

\[
\hat{P}(c_j) = \frac{N_{c_j}}{N_{total}}
\]

2. Drop "with"

3. Likelihoods from training:

\[
p(w_i|c) = \frac{\text{count}(w_i, c) + 1}{\left(\sum_{w \in V} \text{count}(w, c)\right) + |V|}
\]

4. Scoring the test set:

Hint: for this example, do we care about words not in test?