Classification Methods
Classification Methods: Supervised Machine Learning

Lots of kinds!
- Naïve Bayes
- Logistic regression
- Neural networks
- k-Nearest Neighbors
- random forests
- ...

Classifi+ication Methods: Supervised Machine Learning
Classification Methods: Supervised Machine Learning

**Input:**
- a document $d$
- a fixed set of classes $C = \{c_1, c_2, \ldots, c_T\}$
- A training set of $m$ hand-labeled documents $(d_1, c_1), \ldots, (d_m, c_m)$

**Output:**
- a learned classifier $\gamma: d \rightarrow c$
Generative and Discriminative Classifiers
Logistic Regression

• Important analytic tool in natural and social sciences

• Baseline supervised machine learning tool for classification

• Is also the foundation of neural networks
Suppose we're distinguishing cat from dog images
Generative Classifier:

- Build a model of what's in a cat image
  - Knows about whiskers, ears, eyes
  - Assigns a probability to any image:
    - how cat-y is this image?

Also build a model for dog images

Now given a new image:
Run both models and see which one fits better
Discriminative Classifier

Just try to distinguish dogs from cats
Components of a probabilistic machine learning classifier

Given $m$ input/output pairs $(x^{(i)}, y^{(i)})$:

1. A feature representation of the input
   For each input $x^{(i)}$, a vector of features $[x_1, x_2, \ldots, x_N]$

2. A classification function that computes $\hat{y}$, the estimated class via $p(y|\hat{x})$

3. An objective function for learning, like cross-entropy loss

4. An algorithm for optimizing the objective function: stochastic gradient descent.
Logistic Regression Classifiers
Is this spam?

Subject: Important notice!
From: Stanford University <newsforum@stanford.edu>
Date: October 28, 2011 12:34:16 PM PDT
To: undisclosed-recipients:;

Greats News!

You can now access the latest news by using the link below to login to Stanford University News Forum.


Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information about the new services.

© Stanford University. All Rights Reserved.
Who wrote which Federalist papers?

• Anonymous essays try to convince New York to ratify U.S Constitution written by Jay, Madison, Hamilton.
• Authorship of 12 of the letters in dispute
• Solved by Mosteller and Wallace (1963) using Bayesian methods

Image: Tyler Feder (https://www.allfreepapercrafts.com/Free-Printables/Hamilton-Paper-Dolls)
What is the subject of this research article?

MEDLINE

Antagonists and Inhibitors
Blood Supply
Chemistry
Drug Therapy
Embryology
Epidemiology
...

?
Positive or negative movie review?

+ ...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...
Input:
  - a document
  - a fixed set of classes

Output: a predicted class
Binary Classification in Logistic Regression

Given a series of input/output pairs: \((x^i, y^i)\)

For each observation \(x^{(i)}\)

- We represent \(x^{(i)}\) by a feature vector \([x_1, x_2, ..., x_n]\)
- We compute an output \(\hat{y}^i \in [0, 1]\): a predicted class
Features in logistic regression

"this movie is awesome!"

For feature $x_i$, weight $w_i$ tells is how important is $x_i$

- $x_i =$"review contains ‘awesome’":
- $x_j =$"review contains ‘abysmal’":
- $x_k =$“review contains ‘mediocre’":

\[
\begin{bmatrix}
\text{awesome} & \text{abysmal} & \text{mediocre} \\
1 & 0 & 0
\end{bmatrix}
\]
Features in logistic regression

"this movie is awesome!"

For feature $x_i$, weight $w_i$ tells is how important is $x_i$

- $x_i =$"review contains ‘awesome’": $w_i: +10$
- $x_j =$"review contains ‘abysmal’": $w_j: -10$
- $x_k =$“review contains ‘mediocre’": $w_k: -2$

\[
\begin{bmatrix}
\text{awesome} & \text{abysmal} & \text{mediocre} \\
1 & 0 & 0
\end{bmatrix}
\]
Logistic Regression for one observation $x$

Input observation: $\textbf{Vector } \textbf{x} = [x_1, x_2, \ldots, x_n]$

Weights (one per feature): $\textbf{W} = [w_1, w_2, \ldots, w_n]$

Output: a predicted class $
\hat{y} \in \{0, 1\}$
How to do classification

For each feature $x_i$, weight $w_i$ tells us importance of $x_i$

- Plus, bias $b$

We'll sum up all the weighted features and the bias:

$$z \left( \sum_{i=1}^{n} w_i x_i \right) + b$$

$$z = W \cdot X + b$$
How to do classification

For each feature $x_i$, weight $w_i$ tells us importance of $x_i$

- Plus, bias $b$

We'll sum up all the weighted features and the bias:

$$z = \sum_{i=1}^{n} w_i x_i + b$$

$$z = W \cdot X + b$$

"This movie is awesome!"

$X = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$

awesomemmediocre

$W = \begin{bmatrix} 10 & -10 & -2 \end{bmatrix}$

$$W \cdot X = \sum_{i=1}^{n} w_i x_i = w_1 x_1 + w_2 x_2 + w_3 x_3$$

$$= 10 \cdot 1 + (-10) \cdot 0 + (-2) \cdot 0 = 10$$
But we want a probabilistic classifier

We need to formalize “sum is high”.

We’d like a principled classifier that gives us a probability, just like Naive Bayes did.

We want a model that can tell us:

\[ p(y = 1 | x, w) \]
\[ p(y = 0 | x, w) \]
The problem: $z$ isn't a probability, it's just a number!

$$Z + W \cdot X = b$$

Solution: apply a function to $z$ that ranges from 0 to 1

Sigmoid

$$\sigma(x) = \frac{1}{1 + e^{-x}} = \frac{1}{1 + e^{-p(-z)}}$$
The very useful sigmoid or logistic function

\[ \sigma(z) = \frac{1}{1 + e^{-z}} \]
Idea of logistic regression

Compute $w \cdot x + b$

Then pass it through the sigmoid function: $\sigma(w \cdot x + b)$

Treat it as a probability
Making probabilities with sigmoids

\[ P(y=1 \mid x, w) = \sigma(wx + b) \]

\[ = \frac{1}{1 + \exp(- (wx + b))} \]

\[ P(y=0 \mid x, w) = 1 - \sigma(wx + b) \]
Turning a probability into a classifier

\[
\text{decision } (x) = \begin{cases} 
1 & \text{if } p(y=1|w,x) > 0.5 \\
0 & \text{otherwise}
\end{cases}
\]
The probabilistic classifier

\[ P(y=1) = \sigma(w \cdot x + b) = \frac{1}{1 + \exp(-(w \cdot x + b))} \]
Turning a probability into a classifier

\[
\text{decision}(x) = \begin{cases} 
1 & \text{if } w \cdot x + b > 0 \\
0 & \text{otherwise}
\end{cases}
\]
The two phases of logistic regression

Training: We learn weights $W$ and $b$ using stochastic gradient descent.

Test: Given an example $x$, we compute $p(y|x)$ using the learned weights $W$ and $b$, and return whichever labeled ($y=1$ or $y=0$) has higher probability.
Logistic Regression Example:
Text Classification
It's hokey. There are virtually no surprises, and the writing is second-rate. So why was it so enjoyable?

For one thing, the cast is great.

Another nice touch is the music. I was overcome with the urge to get off the couch and start dancing. It sucked me in, and it'll do the same to you.
Features

\[ f_1 : \text{count (positive words)} \in \text{doc} \]
\[ f_2 : \text{count (negative words)} \in \text{doc} \]
\[ f_3 : 30 \text{ if "no" \in \text{doc}} \]
\[ \text{otherwise} \]
\[ f_4 : \text{count (1st & 2nd person pronouns)} \in \text{doc} \]
\[ f_5 : 30 \text{ if "!" \in \text{doc}} \]
\[ f_6 : \log(\text{word count of doc}) \]
Features

\( f_1 : \text{count (positive words)} \in \text{doc} \)

\( f_2 : \text{count (negative words)} \in \text{doc} \)

\( f_3 : \begin{cases} 1 & \text{if "no" } \in \text{doc} \\ 0 & \text{otherwise} \end{cases} \)

\( f_4 : \text{count (1st \\& 2nd person pronouns)} \in \text{doc} \)

\( f_5 : \begin{cases} 0 & \text{if "!" } \in \text{doc} \\ 1 & \text{otherwise} \end{cases} \)

\( f_6 : \log(\text{word count of doc}) \)

\( x = [3, 2, 1, 3, 0, 4.19] \)

\( \text{log(665)} = 4.19 \)
Weights

\[ f_1: \text{count (positive words) in doc} \quad 2.5 \]
\[ f_2: \text{count (negative words) in doc} \quad -5 \]
\[ f_3: \begin{cases} 0 & \text{if "no" in doc} \\ \text{otherwise} & \end{cases} \quad -1.2 \]
\[ f_4: \text{count (1st & 2nd person pronouns) in doc} \quad 0.5 \]
\[ f_5: \begin{cases} 0 & \text{if "!" in doc} \\ \text{otherwise} & \end{cases} \quad 2 \]
\[ f_6: \log(\text{word count of doc}) \quad 0.7 \]

\[ W = [2.5, -5, -1.2, 0.5, 2, 0.7] \]

\[ b = 0.1 \]
Classifying sentiment for input $x$

$x = [3, 2, 1, 8, 0, 4.19]$

$W = [2.5, -5, -1.2, 0.5, 2, 0.7]$

$b = 0.1$

$p(+1x) = \sigma(W \cdot x + b)$

$= \sigma(2.5 \cdot 3 + -5 \cdot 2 + -1.2 \cdot 1 + 0.5 \cdot 8 + 2 \cdot 0 + 0.7 \cdot 4.19 + 0.1)$

$= \sigma(0.8333)$

$= 0.7$

$p(-1x) = 1 - 0.7 = 0.3$
Classification in (binary) logistic regression: summary

Given:

- a set of classes: (+ sentiment, - sentiment)
- a vector \( \mathbf{x} \) of features \([x_1, x_2, \ldots, x_n]\)
  - \(x_1 = \text{count( "awesome" )}\)
  - \(x_2 = \log(\text{number of words in review})\)
- A vector \( \mathbf{w} \) of weights \([w_1, w_2, \ldots, w_n]\)
- \(w_i\) for each feature \(f_i\)

\[
P(y = 1) = \sigma(w \cdot x + b) = \frac{1}{1 + \exp(-(w \cdot x + b))}
\]
Evaluating Classifiers
Evaluation

Consider a binary text classification task:
Is this passage from a book a "smell experience" or not?

Towards Olfactory Information Extraction from Text:
A Case Study on Detecting Smell Experiences in Novels

Ryan Brate and Paul Groth
University of Amsterdam
Amsterdam, the Netherlands
r.brate@gmail.com
p.t.groth@uva.nl

Marieke van Erp
KNAW Humanities Cluster
Digital Humanities Lab
Amsterdam, the Netherlands
marieke.van.erp@dh.huc.knaw.nl

Abstract

Environmental factors determine the smells we perceive, but societal factors shape the importance, sentiment and biases we give to them. Descriptions of smells in text, or as we call them ‘smell experiences’, offer a window into these factors, but they must first be identified. To the best of our knowledge, no tool exists to extract references to smell experiences from text. In
Evaluation

Consider a binary text classification task:
Is this passage from a book a "smell experience" or not?

You build a "smell" detector
- Positive class: paragraph that involves a smell experience
- Negative class: all other paragraphs
The 2-by-2 confusion matrix

### Truth

<table>
<thead>
<tr>
<th>Smell</th>
<th>No Smell</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smell</td>
<td>True Positives</td>
</tr>
<tr>
<td>No Smell</td>
<td>False Negatives</td>
</tr>
</tbody>
</table>

### Prediction

- **Accuracy:** \[
\frac{TP + TN}{TP + FP + TN + FN}
\]
- **Precision:** \[
\frac{TP}{TP + FP}
\]
- **Recall:** \[
\frac{TP}{TP + FN}
\]
Evaluation: Accuracy

Why don't we use **accuracy** as our metric?

Imagine we saw 1 million paragraphs

- 100 of them mention smells
- 999,900 talk about something else

We could build a classifier that labels every paragraph "not about smell"
Evaluation: Precision

% of items the system detected (i.e., items the system labeled as positive) that are in fact positive (according to the human gold labels)

\[
\text{PRECISION} = \frac{TP}{TP + FP} \quad \frac{O}{O + O}
\]
Evaluation: Recall

% of items actually present in the input that were correctly identified by the system.

\[
\text{RECALL} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad \frac{0}{0 + 10} = 0
\]
Why Precision and recall

Our no-smells classifier
  ◦ Labels nothing as "about smell"

Accuracy =

Recall =

Precision =
A combined measure: F

F measure:
a single number that combines P and R

\[ F_\beta = \frac{(\beta^2 + 1) \cdot PR}{\beta^2 P + R} \]

Balanced F when \( \beta = 1 \)

\[ \frac{2PR}{P + R} \]