Lecture 9: Classification II: Logistic Regression
Modeling Social Data, Spring 2017
Columbia University

Stephanie Huang

March 24, 2017

1 Naive Bayes

1.1 Basic Ideas

Say we want to predict a class c given features x. For example, say we want to classify emails as spam or not spam. Our classes c would be $\{0, 1\}$ where 0 represents not spam and 1 represents spam. Our features x would consist of values $\in \{0, 1\}$ representing whether a word is present or not.

What we want to predict is $p(c|x)$ or the probability of a class c given the features x.

Using Bayes' Rule we can get:

$$p(c|x) = \frac{p(x|c)p(c)}{p(x)} \tag{1}$$

Here, $p(x|c)$ represents the probability of seeing all words given the class c.

We also make the "naive" assumption of independence and assume that each word occurrence $p(x_i|c)$ is a "coin flip" with a Bernoulli distribution. Thus we get:

$$p(c|x) = \frac{\prod p(x_i|c)p(c)}{p(x)} \tag{2}$$

$$p(x_j|c) = \theta_{jc}^{x_j}(1 - \theta_{jc})^{1-x_j} \tag{3}$$

Here, θ is just shorthand for probabilities s.t. θ_{jc} is the probability of the jth word in class c.

If we take logs then we get:

$$\log p(c|x) = \sum_j \log[\theta_{jc}^{x_j}(1 - \theta_{jc})^{1-x_j}] + \log \frac{\theta_c}{p(x)} \tag{4}$$

$$\log p(c|x) = \sum_j x_j \log \frac{\theta_{jc}}{1 - \theta_{jc}} + \sum_j \log(1 - \theta_{jc}) + \log \frac{\theta_c}{p(x)} \tag{5}$$

Where, using the product rule, we can get:

$$p(x) = \sum_c p(x, c) = \sum_c p(x|c)p(c) \tag{6}$$

Looking quickly at the computational cost for this equation, we know that the first term is dependent on the number of words present in the document we're looking at, and the second term is dependent on the size of the vocabulary.

If there are two classes, with the possible values of c being 1 or 0, then we can represent the Naive Bayes classifier in log odds form as follows:

$$\log \frac{p(c = 1|x)}{p(c = 0|x)} \tag{7}$$

If the log odds is greater than 0, then we predict $c = 1$, and if it's less than 0, we predict $c = 0$.

Using our equation (5) we get:

$$\log \frac{p(c=1|x)}{p(c=0|x)} = \sum_j x_j \log \frac{\theta_{j1}(1-\theta_{j0})}{\theta_{j0}(1-\theta_{j1})} + \sum_j \log \frac{1-\theta_{j1}}{1-\theta_{j0}} + \log \theta_1 \theta_0 \tag{8}$$

We can assign the following values to simplify our classifier:

$$w_j = \log \frac{\theta_{j1}(1-\theta_{j0})}{\theta_{j0}(1-\theta_{j1})} \tag{9}$$

$$w_0 = \sum_j \log \frac{1-\theta_{j1}}{1-\theta_{j0}} + \log \theta_1 \theta_0 \tag{10}$$

$$\log \frac{p(c=1|x)}{p(c=0|x)} = \mathbf{w} \cdot \mathbf{x} + w_0 \tag{11}$$

If we precompute w_0 then the computational cost is just the number of nonzero words.

1.2 Finding θ_j

To explain how to find individual θ values for our classifier, we did a derivation of the probability of seeing n heads for N coin flips, since a similar concept is applied to the probabilities of seeing words in documents, or other applications of the classifier.

We have the the probability of seeing n heads given θ is:

$$p(n|\theta) = C\theta^n (1-\theta)^{N-n} \tag{12}$$

We can take the log for likelihood, then take the derivative and set to zero to find the max likelihood.

$$\mathcal{L} = \log p(n|\theta) = \log C + n \log \theta + (N-n) \log(1-\theta) \tag{13}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \theta} = \frac{n}{\theta} + \frac{N-n}{1-\theta} \tag{14}$$

$$\frac{n}{\theta} = \frac{N-n}{1-\theta} \tag{15}$$

$$n - n\theta = N\theta - n\theta \tag{16}$$

$$\theta = \frac{n}{N} \tag{17}$$

Using this, then we can get:

$$\theta_{j1} = \frac{n_{j1}}{n_1} = \frac{\#\text{ of spam docs w/ word j}}{\text{total }\#\text{ of spam docs}} \tag{18}$$

$$\theta_1 = \frac{n_1}{N} = \frac{\#\text{ of spam docs}}{\text{total }\#\text{ docs}} \tag{19}$$

1.3 Time and Space

With N documents, c classes, and k different words, the training time for a Naive Bayes classifier is $O(N \cdot \overline{k})$ where $\overline{k}$ is the average number of words in a document. The required space *seems* like it would be $O(k \cdot c)$ but storing using a sparse data structure would save space.

2 Logistic Regression

2.1 The Logistic Function

Say we start with a predictor that has the same form as the Naive Bayes classifier that we ended with:

$$\log \frac{p}{1-p} = \mathbf{w} \cdot \mathbf{x} \tag{20}$$

(Note: There is an implicit intercept in this equation that we are ignoring.)
Using this predictor, we can get some useful values for $p(x|w)$ and $1 - p(x|w)$:

$$\frac{p}{1-p} = e^{\mathbf{w} \cdot \mathbf{x}} \tag{21}$$

$$p = e^{w \cdot x} - pe^{w \cdot x} \tag{22}$$

$$p(1 + e^{w \cdot x}) = e^{w \cdot x} \tag{23}$$

$$p = \frac{e^{w \cdot x}}{1 + e^{w \cdot x}} \tag{24}$$

$$p(x|w) = \frac{1}{1 + e^{-w \cdot x}} \tag{25}$$

$$1 - p(x|w) = 1 - \frac{1}{1 + e^{-w \cdot x}} = \frac{1}{1 + e^{w \cdot x}} \tag{26}$$

Equation 25 is a sigmoid function that when plotted gives us the S-shaped logistic curve, as seen below:

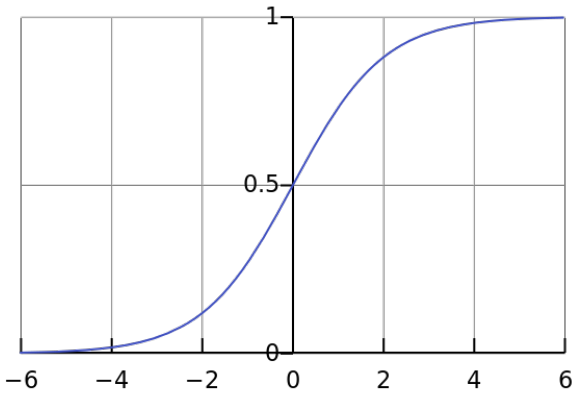

Figure 1: https://en.wikipedia.org/wiki/Sigmoid_function

2.2 Determining the Model

Say we have a set of labels $y_i \in 0, 1$ where 1 represents spam and 0 represents ham, and a set of documents x_i.
Then we get likelihood as follows, where $p_i = p(x_i|w)$:

$$p(D|w) = \prod_i p_i^{y_i} (1 - p_i)^{1 - y_i} \tag{27}$$

Taking the log likelihood, we get:

$$\mathcal{L} = -\log p(D|w) \tag{28}$$

$$\mathcal{L} = -\sum_i \{y_i \log p_i + (1 - y_i) \log(1 - p_i)\} \tag{29}$$

$$\mathcal{L} = -\sum_i \{y_i \log \frac{p_i}{1-p_i} + \log(1-p_i)\} \tag{30}$$

From 21 and 26 above, we can get:

$$\mathcal{L} = -\sum_i \{y_i w \cdot x - \log(1+e^{w \cdot x})\} \tag{31}$$

Then finding maximum likelihood:

$$0 = \frac{\partial \mathcal{L}}{\partial w_k} = -\sum_i \{y_i x_{ik} - \frac{1}{1+e^{w \cdot x_i}} e^{w \cdot x_i} x_{ik}\} \tag{32}$$

Using equation 24 we get:

$$0 = -\sum_i \{y_i - p_i\} x_{ik} \tag{33}$$

$$0 = -\sum_i \{y_i - \frac{1}{1+e^{-w \cdot x}}\} x_{ik} \tag{34}$$

There is no closed form solution to this equation, so we use gradient descent:

$$w \leftarrow w - \eta \frac{\partial \mathcal{L}}{\partial w} \tag{35}$$

In matrix form we get:

$$w \leftarrow w + \eta X^T (y - p) \tag{36}$$

Which can also be represented in component form as:

$$w_k \leftarrow w_k + \eta \sum_i (y_i - p_i) x_{ik} \tag{37}$$

2.3 Side Note: Regularized Logistic Regression

If we don't want the weights in our logistic regression model to get too big, we can use regularization. Then our likelihood equation becomes:

$$\mathcal{L} = -\sum_i \{y_i \log p_i + (1-y_i) \log(1-p_i)\} + \frac{1}{2}\lambda ||w||^2 \tag{38}$$

$$\frac{\partial \mathcal{L}}{\partial w} = -\sum_i (y_i - p_i) x_{ik} + \lambda w_k \tag{39}$$

The last term in (38) is our regularization term.
Then our update rule becomes:

$$w_k \leftarrow w_k - \eta \frac{\partial \mathcal{L}}{w_k} = w_k + \eta \sum_i (y_i - p_i) x_{ik} - \lambda w_k \tag{40}$$

$$w_k \leftarrow (1 - \eta\lambda) w_k + \eta \sum_i (y_i - p_i) x_{ik} \tag{41}$$

Essentially what we're doing in the update is shrinking the weights first before moving against the gradient the updating our weights.

3 Evaluating Classifiers

There are a couple of different metrics and terms that are often used to evaluate classifiers:

- **accuracy**: The fraction of times you predict the correct label.
- **calibration**: Measures how often an event with predicted probability p occurs.
- **confusion matrix**: a way of representing predicted and actual values.

	predicted	
actual	1	0
1	true positive (TP)	false negative (FN)
0	false positive (FP)	true negative (TN)

- **precision**: The fraction of positive predictions that are true. Can also be represented as $\frac{\text{TP}}{\text{TP+FP}}$
- **recall** (aka true positive rate): fraction of true examples that we predict to be positive. $\frac{\text{TP}}{\text{TP+FN}}$
- **false positive rate**: fraction of false examples that are predicted positive. $\frac{\text{TP}}{\text{TP+FN}}$
- **receiver operator characteristic curve (ROC)**: we can plot the TPR (probability of true detection) vs. FPR (probability of a false alarm) and by changing our probability threshold, we can get a curve, as shown below:

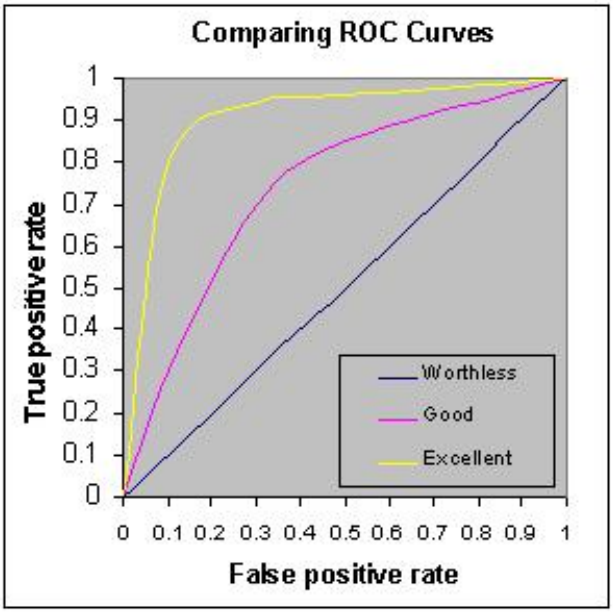

Figure 2: http://gim.unmc.edu/dxtests/roc3.htm

- **area under curve (AUC)**: the area under the ROC curve is equivalent to the accuracy for a balanced classification

4 "Guest Lecture": Computational Social Science: Exciting Progress and Future Challenges

Speaker: Duncan Watts

4.1 Introduction

A problem of particular interest in social science is the **"micro-macro" or emergence problem**. This is the phenomena of collective social behavior that spawns from an individual level. Unfortunately, it's hard to study this problem empirically, especially on a "macro" scale. However, the advent of the Internet means that there is a dramatic increase in the scale, scope, and granularity of data available. Web platforms also allow for an increase in the speed and scale of experiments.

In this talk, we looked at computational social science from two main examples: "big data" social contagion and "small data" virtual lab experiments.

4.2 Social Contagion using Big Data

In this section of the talk, we look at how things go viral. When we talk about "viral" or "social contagion," our concept of this term comes from biological/epidemiological ideas of a multigenerational branching spread. But historically, S-shaped curves have always been used to represent social data and infer "diffusion" events and virality.

There are two main problems with this historical approach:

1. An S-shaped curve can result from many different processes (e.g. marketing efforts, population heterogeneity)
2. We usually only see and collect data for successful diffusion events, so we have few examples of unsuccessful ones.

Exploring social contagion requires the use of individual-level data and data from unsuccessful attempts. The web helps solve some of these problems. The following are some examples of projects that involve the web, big data, and social contagion.

4.2.1 Online Diffusion Project (2012: Goel, Watts, Goldstein)

Associated Paper: S. Goel, D. J. Watts, and D. G. Goldstein, The structure of online diffusion networks. In Proceedings of the 13th ACM Conference on Electronic Commerce (EC '12). 2012.

This project involved 6 unrelated projects that were all designed in an attempt to go viral. There were consistent diffusion patterns across all the projects. They found that 99% of adoptions are ≤ 1 "hop" from the seed, or origin/source, suggesting that there isn't much virality in spreading.

4.2.2 "Structural Virality" Project (2015: Goel, Anderson, Hofman, Watts)

Associated Paper: S. Goel, A. Anderson, J. Hofman, and D. J. Watts, The Structural Virality of Online Diffusion. Management Science 62(1):180-196. 2015.

This project looked at every video, news story, image, and petition on Twitter over a timespan of 12 months. This was about 1.4 billion different "events". They restricted this dataset to only "popular" things by taking only events that had over 100 retweets, resulting in about 350,000 items to look at.

They looked at the average shortest path length for nodes in the viral structure to classify the structure. Structures with an average shortest path length of 2 were "broadcast" and those with an average shortest path length of log(n) were "viral".

What they found was that there are all sorts of patterns and structures in the "virality" of a viral event's spread that is unrelated to time scale. They found that popularity $\neq$ virality. Popularity is usually driven by celebrities, or the individuals that are the largest source of broadcasting.

4.2.3 How Predictable are Cascades? (2016: Martin, Hofman, Sharma, Anderson, Watts)

Associated Paper: T. Martin, J. Hofman, A. Sharma, A. Anderson, and D. J. Watts, Exploring Limits to Prediction in Complex Social Systems. Proceedings of the 25th ACM International World Wide Web Conference (WWW). 2016.

This paper looked at whether popularity/cascade size is predictable. What they found was that using only past success as a predictor could get the same result as a more complex model with more features. This suggests that there's an asymptotical limit to how well and how much popularity/cascade size can be predicted, and that perhaps sociological theories are not enough to predict behavior in these cases because there is too much randomness.

4.3 Virtual Labs

Traditional physical behavioral lab experiments have a lot of limitations (validity in external environments, expensive, slow.) Virtual labs allow for more realism, scale, and duration in the experimental design space.

We went over 3 examples of how virtual labs allow for expansion in scale, realism, and time.

4.3.1 Scale: Music Lab (2006: Salganik, Dodds, Watts)

Associated Paper: M. J. Salganik, P. S. Dodds, and D. J. Watts, "Experimental study of inequality and unpredictability in an artificial cultural market." Science, 311:854-856, 2006.

The Music Lab was an experiment that looked at social influence and market dynamics, and involved getting thousands of volunteers recruited from social media to participate in a web app that was designed expressly for the purposes of the experiment. This experiment is an example of how virtual labs allow for massive increases in the scale of an experiment.

4.3.2 Realism: Crisis Mapping in the Lab (2016: Mao, Mason, Suri, Watts)

Associated Paper: A. Mao, W. Mason, S. Suri, and D. J. Watts, An Experimental Study of Team Size and Performance on a Complex Task. PLoS ONE 11(4). 2016.

4.3.3 Time: Month-long Prisoner's Dilemma Experiment

Associated Paper: A. Mao, L. Dworkin, S. Suri, D. J. Watts, Resilient cooperators stabilize long-run cooperation in the finitely repeated Prisoners Dilemma. 2017.