Classification

CSE 258 – Lecture 3
Web Mining and Recommender Systems
Learning outcomes

This week we want to:
• Explore techniques for **classification**
• Try some simple solutions, and see why they might fail
• Explore more complex solutions, and their advantages and disadvantages
• Understand the relationship between classification and regression
• Examine how we can reliably **evaluate** classifiers under different conditions
Recap
Last week we started looking at supervised learning problems.

\[ f(\text{data}) \rightarrow \text{labels} \]
Last week...

We studied **linear regression**, in order to learn linear relationships between features and parameters to predict **real-valued** outputs.

\[ X \theta = y \]

- **matrix of features** (data)
- **unknowns** (which features are relevant)
- **vector of outputs** (labels)
Last week...

$ratings \Rightarrow \text{star rating}$

$f(\text{user features, movie features}) \rightarrow \text{star rating}$
Four important ideas from last week:

1) Regression can be cast in terms of maximizing a likelihood

\[ y_i = X_i \cdot \theta + \mathcal{N}(0, \sigma^2) \]

\[ \max_\theta \prod_i p_\theta(y_i | x_i) = \prod_i \exp \left( y_i - x_i \cdot \theta \right)^2 \]
Four important ideas from last week:

2) Gradient descent for model optimization

1. Initialize $\theta$ at random
2. While (not converged) do

$$\theta := \theta - \alpha f'(\theta)$$
3) Regularization & Occam’s razor

**Regularization** is the process of penalizing model complexity during training

\[
\arg \min_{\theta} = \frac{1}{N} \| y - X\theta \|^2_2 + \lambda \| \theta \|^2_2
\]

How much should we trade-off accuracy versus complexity?
Four important ideas from last week:

4) Regularization pipeline

1. Training set – select model parameters
2. Validation set – to choose amongst models (i.e., hyperparameters)
3. Test set – just for testing!
A validation set is constructed to “tune” the model’s parameters

- Training set: used to optimize the model’s parameters
- Test set: used to report how well we expect the model to perform on unseen data
- Validation set: used to tune any model parameters that are not directly optimized
A few “theorems” about training, validation, and test sets

• The training error increases as lambda increases
• The validation and test error are at least as large as the training error (assuming infinitely large random partitions)
• The validation/test error will usually have a “sweet spot” between under- and over-fitting
Today...

How can we predict **binary** or **categorical** variables?

\[ f(\text{data}) \xrightarrow{?} \text{labels} \]

\{0,1\}, \{True, False\}

\{1, \ldots, N\}
Today...

Will I purchase this product?
(yes)

Will I click on this ad?
(no)
Today...

What animal appears in this image?

(mandarin duck)
What are the categories of the item being described?

(book, fiction, philosophical fiction)

From Booklist

Houellebecq's deeply philosophical novel is about an alienated young man searching for happiness in the computer age. Bored with the world and too weary to try to adapt to the foibles of friends and coworkers, he retreats into himself, descending into depression while attempting to analyze the passions of the people around him. Houellebecq uses his nameless narrator as a vehicle for extended exploration into the meanings and manifestations of love and desire in human interactions. Ironically, as the narrator attempts to define love in increasingly abstract terms, he becomes less and less capable of experiencing that which he is so desperate to understand. Intelligent and well written, the short novel is a thought-provoking inspection of a generation's confusion about all things sexual. Houellebecq captures precisely the cynical disillusionment of disaffected youth. Bonnie Johnston --This text refers to an out of print or unavailable edition of this title.
Today...

We’ll attempt to build **classifiers** that make decisions according to rules of the form

\[ y_i = \begin{cases} 
1 & \text{if } X_i \cdot \theta > 0 \\
0 & \text{otherwise}
\end{cases} \]
1. Naïve Bayes
Assumes an *independence* relationship between the features and the class label and “learns” a simple model by counting

2. Logistic regression
Adapts the *regression* approaches we saw last week to binary problems

3. Support Vector Machines
Learns to classify items by finding a hyperplane that separates them
This week...

**Ranking** results in order of how likely they are to be relevant
This week...

**Evaluating classifiers**

- False positives are nuisances but false negatives are disastrous (or vice versa)
  - Some classes are very rare
- When we only care about the “most confident” predictions

E.g. which of these bags contains a weapon?
We want to associate a probability with a label and its negation:

\[ p(label|data) \]
\[ p(\neg label|data) \]

(classify according to whichever probability is greater than 0.5)

**Q:** How far can we get just by counting?
Naïve Bayes

e.g. $p(\text{movie is “action” } | \ \text{schrarzenegger in cast})$

Just count!

#fims with Arnold = 45

#action films with Arnold = 32

$p(\text{movie is “action” } | \ \text{schrarzenegger in cast}) = \frac{32}{45}$
What about:

\[ p(\text{movie is “action”} \mid \text{schwarzenneger in cast and release year = 2017 and mpaa rating = PG and budget < $1000000}) \]

#(training) fims with Arnold, released in 2017, rated PG, with a budged below $1M = 0

#(training) action fims with Arnold, released in 2017, rated PG, with a budged below $1M = 0
Q: If we’ve never seen this combination of features before, what can we conclude about their probability?

A: We need some simplifying assumption in order to associate a probability with this feature combination.
Naïve Bayes assumes that features are conditionally independent given the label

$$(\text{feature}_i \perp \text{feature}_j | \text{label})$$
Naïve Bayes

\[(\text{feature}_i \perp \perp \text{feature}_j \mid \text{label})\]

\[a \perp \perp b \Rightarrow p(a, b) = p(a) p(b)\]

\[a \perp \perp b \mid c \Rightarrow p(a, b \mid c) = p(a \mid c) p(b \mid c)\]

\[a = \text{I'm wearing shorts}\]

\[b = \text{you're wearing shorts}\]

\[c = \text{it's summer}\]
Conditional independence?

\[(a \perp b | c)\]

(a is conditionally independent of b, given c)

“if you know c, then knowing a provides no additional information about b”

(I remembered my umbrella \(\perp\) the streets are wet | it’s raining)
Naïve Bayes

\[
\text{(feature}_i \perp \perp \text{feature}_j | \text{label})
\]

\[
p(\text{feature}_i, \text{feature}_j | \text{label}) = p(\text{feature}_i | \text{label})p(\text{feature}_j | \text{label})
\]
Naïve Bayes

\[ p(label|features) = \frac{p(a|b) \cdot p(\text{features}|label)}{p(\text{features})} \]

posterior \quad prior \quad likelihood

evidence

\[ p(label) \prod_{i} p(\text{feature}_{i}|label) \]
Naïve Bayes

\[ p(label|features) = \frac{p(label) \prod_i p(feature_i|label)}{p(features)} \]

\[ p(\neg label|features) = p(\neg label) \frac{\prod_i p(feature_i|label)}{p(features)} \]

The denominator doesn’t matter, because we really just care about

\[ p(label|features) \text{ vs. } p(\neg label|features) \]

both of which have the same denominator
The denominator doesn’t matter, because we really just care about

\[ p(label|features) \text{ vs. } p(\neg label|features) \]

both of which have the same denominator
Example 1

Amazon editorial descriptions:

Amazon.com Review

For most children, summer vacation is something to look forward to. But not for our 13-year-old uncle, and cousin who detest him. The third book in J.K. Rowling's Harry Potter series catapults Dursleys' dreadful visitor Aunt Marge to inflate like a monstrous balloon and drift up to the ceiling (and from officials at Hogwarts School of Witchcraft and Wizardry who strictly forbid students to) out into the darkness with his heavy trunk and his owl Hedwig.

As it turns out, Harry isn't punished at all for his errant wizardry. Instead he is mysteriously released, violently purple bus to spend the remaining weeks of summer in a friendly inn car his third year at Hogwarts explains why the officials let him off easily. It seems that Sirius Black is loose. Not only that, but he's after Harry Potter. But why? And why do the Dementors, the guar are unaffected? Once again, Rowling has created a mystery that will have children and adults clamoring for the next book. Fortunately, there are four more in the works. (Ages 9 and older) --Karin Snelson --This text is

50k descriptions:

http://jmcauley.ucsd.edu/cse258/data/amazon/book_descriptions_50000.json
Example 1

\[ P(\text{book is a children's book} \mid \text{“wizard” is mentioned in the description and “witch” is mentioned in the description}) \]

Code available on:  
http://jmcauley.ucsd.edu/cse258/code/week2.py
Example 1

Conditional independence assumption:

“if you know a book is for children, then knowing that wizards are mentioned provides no additional information about whether witches are mentioned”

obviously ridiculous
Double-counting

Q: What would happen if we trained two regressors, and attempted to “naively” combine their parameters?
Double-counting

\[ \text{height} = 1.2 \text{weight} \]

\[ \text{height} = 15 \text{ shoe size} \]

\[ H = 1.2W + 15\text{s} \]
A: Since both features encode essentially the same information, we’ll end up double-counting their effect.
Logistic Regression also aims to model

\[ p(label|data) \]

By training a classifier of the form

\[ y_i = \begin{cases} 
1 & \text{if } X_i \cdot \theta > 0 \\
0 & \text{otherwise}
\end{cases} \]
Logistic regression

**Last week:** regression

\[ y_i = X_i \cdot \theta \]

**This week:** logistic regression

\[ y_i = \begin{cases} 
1 & \text{if } X_i \cdot \theta > 0 \\
0 & \text{otherwise} 
\end{cases} \]
Q: How to convert a real-valued expression \((X_i \cdot \theta \in \mathbb{R})\) into a probability \((p_{\theta}(y_i|X_i) \in [0, 1])\)
A: sigmoid function: $\sigma(t) = \frac{1}{1+e^{-t}}$
Logistic regression

Training:

\[ X_i \cdot \theta \] should be maximized when \( y_i \) is positive and minimized when \( y_i \) is negative

\[
\arg \max_{\theta} \prod_{y_i=1} \Pr(y_i|X_i) \cdot \prod_{y_i=0} (1 - \Pr(y_i|X_i)) \cdot \prod_{y_i=1} \sigma(X_i; \theta) \cdot \prod_{y_i=0} (1 - \sigma(X_i; \theta))
\]
Logistic regression

How to optimize?

\[ L_\theta(y|X) = \prod_{y_i=1} p_\theta(y_i|X_i) \prod_{y_i=0} (1 - p_\theta(y_i|X_i)) \]

• Take logarithm
• Subtract regularizer
• Compute gradient
• Solve using gradient ascent
Logistic regression

\[ L_\theta(y|X) = \prod_{y_i=1} p_\theta(y_i|X_i) \prod_{y_i=0} (1 - p_\theta(y_i|X_i)) \]

\[ \ell_\theta(y|X) = \begin{cases} 
\sum y_i \log \sigma(X_i; \theta) & \text{if } y_i = 1 \\
\sum (1 - y_i) \log (1 - \sigma(X_i; \theta)) & \text{if } y_i = 0
\end{cases} \]

\[ = \sum_{y_i=1} \log \left( \frac{1}{1 + e^{-X_i; \theta}} \right) + \sum_{y_i=0} \log \left( \frac{e^{-X_i; \theta}}{1 + e^{-X_i; \theta}} \right) \]

\[ = \sum_{y} \log (1 + e^{-X; \theta}) - \sum_{y_i=0} X_i; \theta \]
Logistic regression

\[ l_\theta(y|X) = \sum_i -\log(1 + e^{-X_i \cdot \theta}) + \sum_{y_i=0} -X_i \cdot \theta - \lambda \|\theta\|_2^2 \]

\[ \frac{\partial l}{\partial \theta_k} = -\sum_i x_{ik} e^{X_i \cdot \theta} \frac{e^{-X_i \cdot \theta}}{1 + e^{-X_i \cdot \theta}} + \sum_{y_i=0} (1 - \sigma(X_i \cdot \theta)) + \sum_{y_i=0} -x_{ik} - 2 \lambda \theta_k \]
Logistic regression

Log-likelihood:

\[ l_\theta(y|X) = \sum_i - \log(1 + e^{-X_i \cdot \theta}) + \sum_{y_i=0} -X_i \cdot \theta - \lambda \| \theta \|^2 \]

Derivative:

\[ \frac{\partial l}{\partial \theta_k} = \sum_i X_{ik} (1 - \sigma(X_i \cdot \theta)) + \sum_{y_i=0} -X_{ik} - 2\lambda \theta_k \]
The most common way to generalize \textit{binary} classification (output in \{0,1\}) to \textit{multiclass} classification (output in \{1 \ldots N\}) is simply to train a binary predictor for each class.

e.g. based on the description of this book:
\begin{itemize}
  \item Is it a Children’s book? \{yes, no\}
  \item Is it a Romance? \{yes, no\}
  \item Is it Science Fiction? \{yes, no\}
  \item ...
\end{itemize}

In the event that predictions are inconsistent, choose the one with the highest confidence.
Further reading:

- On Discriminative vs. Generative classifiers: A comparison of logistic regression and naïve Bayes (Ng & Jordan ‘01)
- Boyd-Fletcher-Goldfarb-Shanno algorithm (BFGS)
CSE 258 – Lecture 3
Web Mining and Recommender Systems

Supervised Learning - Support Vector Machines
So far we've looked at **logistic regression**, which is a classification model of the form:

\[
y_i = \begin{cases} 
1 & \text{if } X_i \cdot \theta > 0 \\
0 & \text{otherwise}
\end{cases}
\]

- In order to do so, we made certain **modeling assumptions**, but there are many different models that rely on different assumptions
- In this lecture we’ll look at another such model
Motivation: SVMs vs Logistic regression

Q: Where would a logistic regressor place the decision boundary for these features?
Q: Where would a logistic regressor place the decision boundary for these features?
SVMs vs Logistic regression

• Logistic regressors don’t optimize the number of “mistakes”
• No special attention is paid to the “difficult” instances – every instance influences the model
• But “easy” instances can affect the model (and in a bad way!)
• How can we develop a classifier that optimizes the number of mislabeled examples?
A classifier can be defined by the hyperplane (line) $\theta x - \alpha = 0$
Observation: Not all classifiers are equally good
Support Vector Machines

- An SVM seeks the classifier (in this case a line) that is furthest from the nearest points
- This can be written in terms of a specific optimization problem:

\[
\arg\min_{\theta, \alpha} \frac{1}{2} \|\theta\|_2^2 \\
\text{such that} \quad \forall i \, y_i (\theta \cdot X_i - \alpha) \geq 1
\]
**But**: is finding such a separating hyperplane even possible?
Support Vector Machines

Or: is it actually a good idea?
Support Vector Machines

Want the margin to be as wide as possible

While penalizing points on the wrong side of it
Support Vector Machines

Soft-margin formulation:

$$\arg\min_{\theta, \alpha, \xi > 0} \frac{1}{2} \|\theta\|_2^2 + C \sum_i \xi_i$$

such that

$$\forall i, y_i (\theta \cdot X_i - \alpha) \geq 1 - \xi_i$$
Summary of Support Vector Machines

• SVMs seek to find a hyperplane (in two dimensions, a line) that optimally separates two classes of points.
• The “best” classifier is the one that classifies all points correctly, such that the nearest points are as far as possible from the boundary.
• If not all points can be correctly classified, a penalty is incurred that is proportional to how badly the points are misclassified (i.e., their distance from this hyperplane).
CSE 258 – Lecture 3
Web Mining and Recommender Systems

Supervised Learning - Code example
Judging a book by its cover

Images features are available for each book on
http://jmcauley.ucsd.edu/cse258/data/amazon/book_images_5000.json

4096-dimensional image features
Judging a book by its cover

Example: train a classifier to predict whether a book is a children’s book from its cover art

(code available on)
http://jmcauley.ucsd.edu/code/week2.py
• The number of errors we made was extremely low, yet our classifier doesn’t seem to be very good – why? (stay tuned next lecture!)
The classifiers we’ve seen today all attempt to make decisions by associating weights (theta) with features (x) and classifying according to

$$y_i = \begin{cases} 1 & \text{if } X_i \cdot \theta > 0 \\ 0 & \text{otherwise} \end{cases}$$
Summary

- **Naïve Bayes**
  - Probabilistic model \( p(label|data) \)
  - Makes a conditional independence assumption of the form \( (feature_i \perp \!\!\!\!\perp feature_j | label) \) allowing us to define the model by computing \( p(feature_i | label) \) for each feature
  - Simple to compute just by counting

- **Logistic Regression**
  - Fixes the “double counting” problem present in naïve Bayes

- **SVMs**
  - Non-probabilistic: optimizes the classification error rather than the likelihood
Questions?