### Bayesian Classification

We represent a data item based on the features:

\[ D = \{ f_1, f_2, \ldots, f_n \} \]

#### Training

\[ a: \quad p(a | D) = p(a | f_1, f_2, \ldots, f_n) \quad \Rightarrow \quad p(Label | f_1, f_2, \ldots, f_n) \]

\[ b: \quad p(b | D) = p(b | f_1, f_2, \ldots, f_n) \quad \Rightarrow \quad p(Label | f_1, f_2, \ldots, f_n) \]

*For each label/class, learn a probability distribution based on the features*

### Bayesian Classification

We represent a data item based on the features:

\[ D = \{ f_1, f_2, \ldots, f_n \} \]

#### Classifying

\[ label = \arg \max_{l \in \text{Labels}} P(l | f_1, f_2, \ldots, f_n) \]

Given a new example, classify it as the label with the largest conditional probability.

---

**Admin**

- **Assignment 4**
  - Start working on part 2 now!
  - I’ll post solutions to part 1 soon
    - Compare with what you submitted and make sure you understand your mistakes (if any)
    - Use part 1 to test your code!
- **Review on Tuesday**
  - E-mail me any topics you want me to revisit
Bayes rule for classification

\[ P(\text{Label} \mid \text{Features}) = \frac{P(F \mid L)P(L)}{P(F)} \]

The Naive Bayes Classifier

**Conditional Independence Assumption:** features are independent of each other given the class:

\[ P(f_1, \ldots, f_n \mid L) = P(f_1 \mid L)P(f_2 \mid L)\cdots P(f_n \mid L) \]

Naïve Bayes classifier

\[ P(f_1, \ldots, f_n \mid L) = P(f_1 \mid L)P(f_2 \mid L)\cdots P(f_n \mid L) \]

Bayesian Classification

**Classifying** Given a new example, classify it as the label with the largest conditional probability

Two Classes

\[
\begin{align*}
P(\text{positive} \mid \text{features}) &= \frac{P(f_1 \mid \text{positive})P(f_2 \mid \text{positive})\cdots P(f_n \mid \text{positive})}{P(F)} \\
P(\text{negative} \mid \text{features}) &= \frac{P(f_1 \mid \text{negative})P(f_2 \mid \text{negative})\cdots P(f_n \mid \text{negative})}{P(F)}
\end{align*}
\]

Compare and pick the largest!
Bayesian Classification

Classifying
Given an new example, classify it as the label with the largest conditional probability

Two Classes

Two Classes

\[
P(\text{positive} \mid \text{features}) = \frac{P(f_1 \mid \text{positive})P(f_2 \mid \text{positive}) \cdots P(f_n \mid \text{positive})P(\text{positive})}{P(F)}
\]

\[
P(\text{negative} \mid \text{features}) = \frac{P(f_1 \mid \text{negative})P(f_2 \mid \text{negative}) \cdots P(f_n \mid \text{negative})P(\text{negative})}{P(F)}
\]

What is \(P(F)\)? Does it matter for classification?

Estimating parameters

\[
\hat{P}(l) = \frac{N(l)}{N}
\]

number of items with label
total number of items

How do we estimate these?

Maximum likelihood estimates

\[
\hat{P}(l) = \frac{N(l)}{N}
\]

number of items with label
number of items with label

\[
\hat{P}(f_i \mid l) = \frac{N(f_i, l)}{N(l)}
\]

number of items with the label with feature
number of items with label

Any problems with this approach?

Hint: What was the \(p(I \text{ thought I hated it but loved it})\)?
Maximum likelihood estimates

\[ \hat{P}(f_i \mid l) = \frac{N(f_i, l)}{N(l)} \]

\[ \hat{P}(\text{flu} \mid \text{muscle _ aches}) = \frac{N(\text{flu}, \text{muscle _ aches})}{N(\text{flu})} \]

\[ \hat{P}(\text{thought} \mid \text{positive}) = \frac{N(\text{thought}, \text{positive})}{N(\text{positive})} \]

What if we have seen no training cases where patient had no flu and muscle aches? Or no positive documents with the word “thought”?

Problem with Max Likelihood

Zero probabilities cannot be conditioned away, no matter the other evidence!

\[ \text{label} = \arg\max_{l \in \text{Labels}} \hat{P}(l) \prod_i \hat{P}(f_i \mid l) \]

If ANY \( p(f \mid l) = 0 \), then the whole probability is 0

Ideas?

Smoothing to Avoid Overfitting

Make every event a little probable...

\[ \hat{P}(f_i \mid l) = \frac{N(f_i, l) + \lambda}{N(l) + k\lambda} \]

Many ways to smooth….
Unseen features

Note that this is different from coming in with a feature we’ve never seen before (in any of the classes)
  – For example, “bloating”

![Diagram of feature relationships]

How?

Classification evaluation

<table>
<thead>
<tr>
<th>Labeled data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

How can we see how well we’re doing?

<table>
<thead>
<tr>
<th>Training data</th>
<th>classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Testing data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

Pretend like we don’t know the labels
Classification evaluation

Data | Label
--- | ---
1 | 0

Classify

1 | 1

Pretend we don't know the labels

Classification evaluation measures?

Accuracy
- num correct / total

Class specific measures
- Precision
  - num correct with class A / num predicted class A
- Recall
  - num correct with class A / num with class A
- F1-measure
  - 2 * (precision * recall) / (precision + recall)

Why have these class specific measures?
NB: The good and the bad

Good
- Easy to understand
- Fast to train
- Reasonable performance

Bad
- We can do better
- Independence assumptions are rarely true
- Smoothing is challenging
- Feature selection is usually required

The mind-reading game

How good are you at guessing random numbers?

Repeat 100 times:
- Computer guesses whether you’ll type 0/1
- You type 0 or 1

http://seed.ucsd.edu/~mindreader/
[written by Y. Freund and R. Schapire]

The mind-reading game

The computer is right much more than half the time…

Another example

Database of 20,000 images of handwritten digits, each labeled by a human

[28 x 28 greyscale; pixel values 0-255; labels 0-9]

Use these to learn a classifier which will label digit-images automatically…
Another example

Database of 20,000 images of handwritten digits, each labeled by a human

Features?

[28 x 28 grayscale; pixel values 0-255; labels 0-9]

The learning problem

features = \{0,1,...,255\}^{784}
labels = \{0,1,...,9\}

28x28 features, values ranging from 0 to 255

train a predictive model

classifier

defines

predicts
0
1
2
3
4
5
6
7
8
9

Points in a feature space

One way to view the data

Test example: what class?
Test example = Class 1

Nearest neighbor

Image to label | Nearest neighbor
---|---
1 → 2
3 → 3
4 → 4

Overall: error rate = 6 (on test set)

What is the error rate for random guessing?

What does it get wrong?

Who knows… but here’s a hypothesis:
Each digit corresponds to some connected region of $\mathbb{R}^784$.
Some of the regions come close to each other; problems occur at these boundaries.

What does it get wrong?

Any ideas for improving this?
What does it get wrong?

A random point in this ball has only a 70% chance of being in \( R_2 \).

How can we approximate this?

k-Nearest Neighbor (k-NN)

To classify an example \( d \):
- Find \( k \) nearest neighbors of \( d \)
- Choose as the class the majority class within the \( k \) nearest neighbors

Can get rough approximations of probability of belonging to a class as fraction of \( k \).

k-Nearest Neighbor (k-NN)

To classify an example \( d \):
- Find \( k \) nearest neighbors of \( d \)
- Choose as the class the majority class within the \( k \) nearest neighbors

Does not explicitly compute boundary or model

Also called:
- Case-based learning
- Memory-based learning
- Lazy learning

Example: k=6 (6-NN)

Which class?
**k Nearest Neighbor**

What value of k should we use?
- Using only the closest example (1NN) to determine the class is subject to errors due to:
  - A single atypical example
  - Noise
- Pick k too large and you end up with looking at neighbors that are not that close
- Value of k is typically odd to avoid ties; 3 and 5 are most common.

**k-NN decision boundaries**

The decision boundaries are places in the features space where the classification of a point changes.

Where are the decision boundaries for k-NN?

**k-NN: The good and the bad**

**Good**
- No training is necessary
- No feature selection necessary
- Scales well with large number of classes
  - Don’t need to train n classifiers for n classes

**Bad**
- Classes can influence each other
  - Small changes to one class can have ripple effect
  - Scores can be hard to convert to probabilities
  - Can be more expensive at test time
- “Model” is all of your training examples which can be large
Feature space

$f_1, f_2, f_3, \ldots, f_m$  m-dimensional space

How big will $m$ be for us?

Bias/variance trade-off

Is this a tree?

Bias/variance trade-off

Is this a tree?
Bias/variance trade-off

Is this a tree?

Bias/Variance

Bias: How well does the model predict the training data?
- high bias – the model doesn’t do a good job of predicting the training data (high training set error)
- The model predictions are biased by the model

Variance: How sensitive to the training data is the learned model?
- high variance – changing the training data can drastically change the learned model

Another way to think about it is model complexity

Simple models
- may not model data well
- high bias

Complicated models
- may overfit to the training data
- high variance

Why do we care about bias/variance?

We want to fit a polynomial to this, which one should we use?
Bias/variance trade-off

Bias: How well does the model predict the training data?
- high bias – the model doesn’t do a good job of predicting the training data (high training set error)
  - The model predictions are biased by the model

Variance: How sensitive to the training data is the learned model?
- high variance – changing the training data can drastically change the learned model

High variance OR high bias?

High bias

High variance
Bias/variance trade-off

What do we want?

Bias: How well does the model predict the training data?
- high bias – the model doesn’t do a good job of predicting the training data (high training set error)
- The model predictions are biased by the model

Variance: How sensitive to the training data is the learned model?
- high variance – changing the training data can drastically change the learned model

Compromise between bias and variance