Machine Learning: Classification

CS311
David Kauchak
Spring 2013

Some material borrowed from:
Sara Owsley Sood and others

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

Bayesian Classification

We represent a data item based on the features:

$$D = \langle f_1, f_2, \ldots, f_n \rangle$$

Training

a: $p(a \mid D) = p(a \mid f_1, f_2, \ldots, f_n)$

b: $p(b \mid D) = p(b \mid f_1, f_2, \ldots, f_n)$

$$p(Label \mid f_1, f_2, \ldots, f_n)$$

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

Bayesian Classification

We represent a data item based on the features:

$$D = \langle f_1, f_2, \ldots, f_n \rangle$$

Classifying

$$label = \underset{l \in Labels}{\operatorname{argmax}} P(l \mid f_1, f_2, \ldots, f_n)$$

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

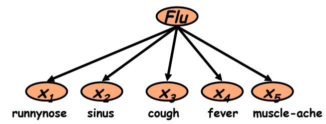
Bayes rule for classification

conditional (posterior) probability

prior probability

$$P(Label \mid Features) = \frac{P(F \mid L)P(L)}{P(F)}$$

The Naive Bayes Classifier



Conditional Independence Assumption: features are independent of eachother 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

$$P(f_1, \ldots, f_n \mid l) = P(f_1 \mid l)P(f_2 \mid l) \cdots P(f_n \mid l)$$

$$P(Label \mid Features) = \frac{P(F \mid L)P(L)}{P(F)}$$

$$= \frac{P(f_1 \mid L)P(f_2 \mid L) \cdots P(f_n \mid L)P(L)}{P(F)}$$

Bayesian Classification

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

Two Classes

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

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

Compare and pick the largest!

Bayesian Classification

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

Two Classes

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

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

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

Bayesian Classification

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

Two Classes

$$\hat{P}(positive \mid features) = P(f_1 \mid positive)P(f_2 \mid positive)\cdots P(f_n \mid positive)P(positive)$$

$$\hat{P}(negative \mid features) = P(f_1 \mid negative)P(f_2 \mid negative)\cdots P(f_n \mid negative)P(negative)$$

Compare these two and see which is larger

$$label = \operatorname*{argmax}_{l \in Labels} P(f_1 \mid l)P(f_2 \mid l)\ldots p(f_n \mid l)P(l)$$

Estimating parameters

$$label = \operatorname*{argmax}_{l \in Labels} P(f_1 \mid l)P(f_2 \mid l)\ldots p(f_n \mid l)P(l)$$

How do we estimate these?

Maximum likelihood estimates

$$\hat{P}(l) = \frac{N(l)}{N} \qquad \frac{\text{number of items with label}}{\text{total number of items}}$$

$$\hat{P}(f_i \mid l) = \frac{N(f_i, l)}{N(l)} \qquad \frac{\text{number of items with the label with feature}}{\text{number of items with label}}$$

Any problems with this approach?

Hint: What was the p(I thought I hated it but loved it)?

Maximum likelihood estimates

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

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

$$\hat{P}(flu \mid muscle\_aches) = \frac{N(flu, muscle\_aches)}{N(flu)}$$

$$\hat{P}(thought \mid positive) = \frac{N(thought, positive)}{N(positive)}$$

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

Maximum likelihood estimates

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

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

$$\hat{P}(flu \mid muscle\_aches) = \frac{N(flu, muscle\_aches)}{N(flu)}$$

$$= 0$$

What are the implications of this?

Problem with Max Likelihood

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

$$label = \operatorname{argmax}_{l \in Labels} \hat{P}(l) \prod_i \hat{P}(f_i \mid l)$$

If **ANY** p(f | l) = 0, then the whole probability is 0

Ideas?

Smoothing to Avoid Overfitting

Make every event a little probable…

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

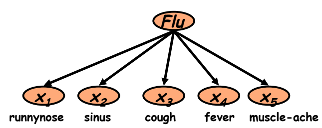
# of features

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)

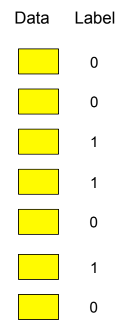
– For example, "bloating"



How?

Classification evaluation

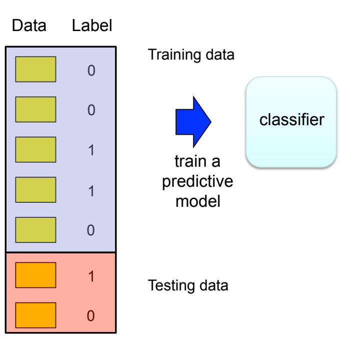
Labeled data



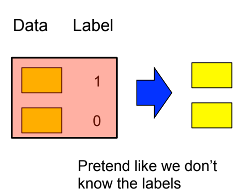
How can we see how well we're doing?

Classification evaluation

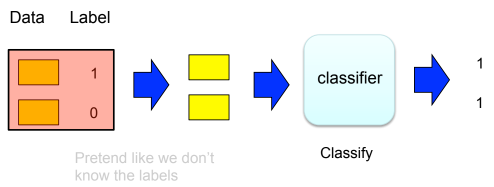
Labeled data



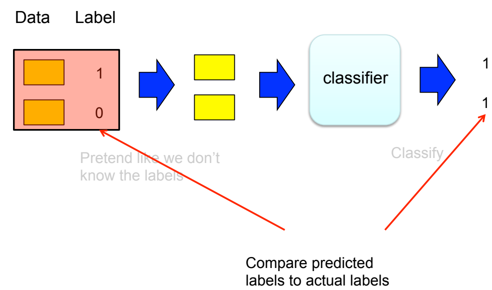
Classification evaluation



Classification evaluation



Classification evaluation



Classification evaluation measures?

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

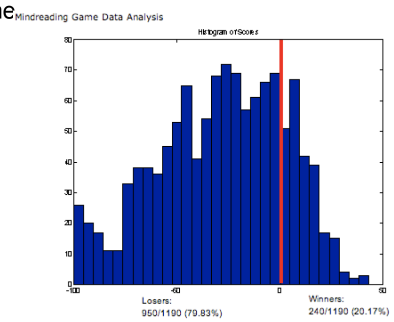
How good are you at guessing random numbers?

```
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]

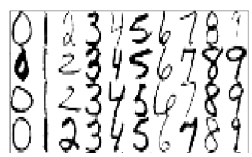
The mind-reading game

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

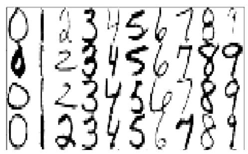


[28 x 28 greyscale; pixel values 0-255; labels 0-9]

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



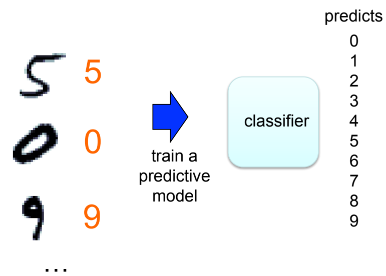
[28 x 28 greyscale; pixel values 0-255; labels 0-9]

Features?

The learning problem

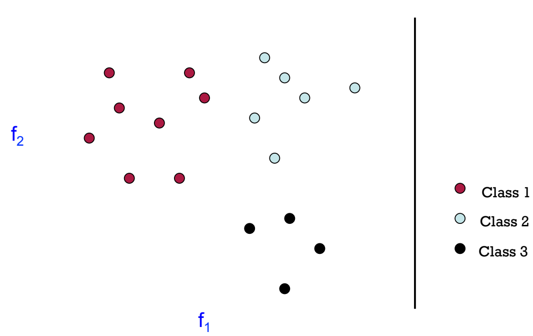
features = $\{0,1,\ldots,255\}^{784}$ 28x28 features, values ranging from 0 to 255

labels = $\{0,1,\ldots,9\}$

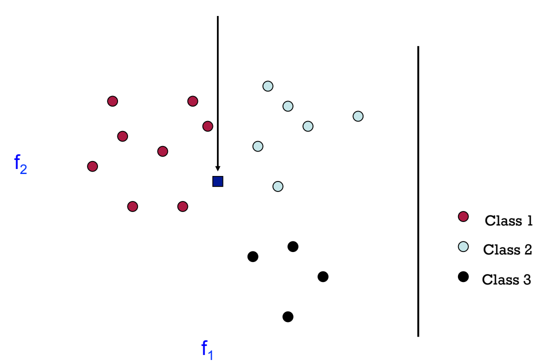


Points in a feature space

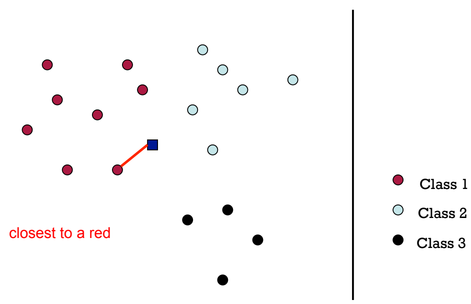
One way to view the data



Test example: what class?

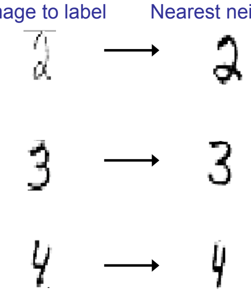


Test example = Class 1



Nearest neighbor

Image to label → Nearest neighbor



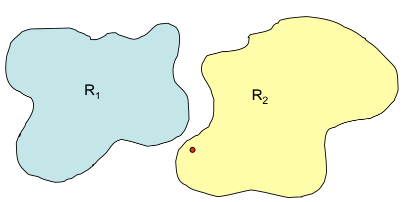
Overall:
error rate = 6
(on test set)

What is the error rate for random guessing?

What does it get wrong?

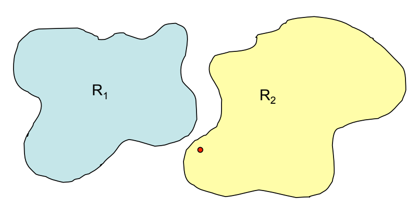
Who knows… but here's a hypothesis:

Each digit corresponds to some connected region of R^{784}. Some of the regions come close to each other; problems occur at these boundaries.



What does it get wrong?

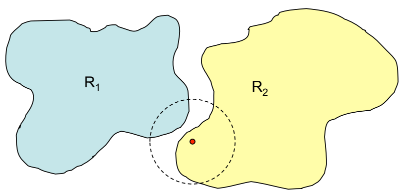
Any ideas for improving this?



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 neightbors

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 neightbors

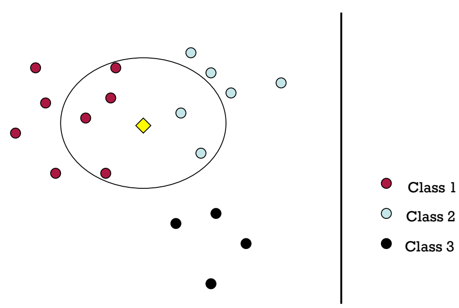
Does not explicitly compute boundary or model

Also called:

- Case-based learning
- Memory-based learning
- Lazy learning

Example: k=6 (6-NN)

Which class?



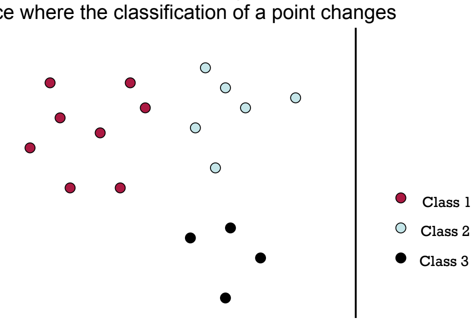
k Nearest Neighbor

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.

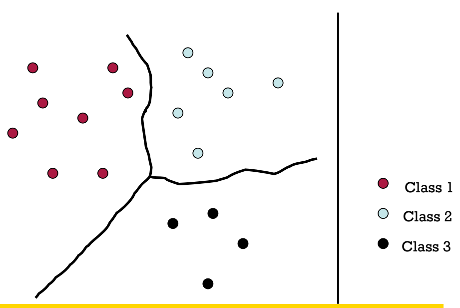
k-NN decision boundaries

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 decision boundaries



k-NN gives locally defined decision boundaries between classes

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

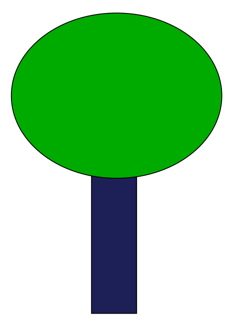
Is this a tree?

Bias/variance trade-off

Is this a tree?

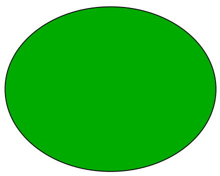
Bias/variance trade-off

Is this a tree?



Bias/variance 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

Bias/Variance

Another way to think about it is model complexity

Simple models

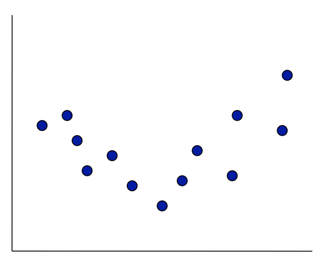
- may not model data well
- high bias

Complicated models

- may overfit to the training data
- high variance

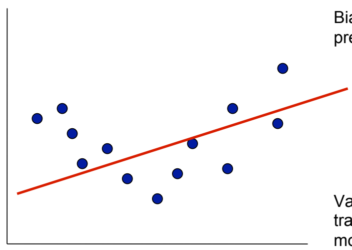
Why do we care about bias/variance?

Bias/variance trade-off



We want to fit a polynomial to this, which one should we use?

Bias/variance trade-off



High variance OR high bias?

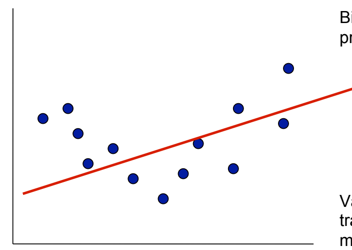
Bias: How well does the model predict the training data?

- 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

Bias/variance trade-off



High bias

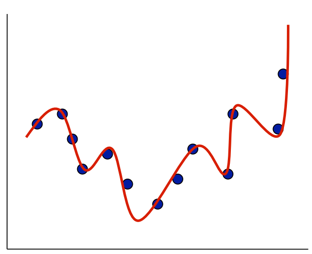
Bias: How well does the model predict the training data?

- 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

Bias/variance trade-off



High variance OR high bias?

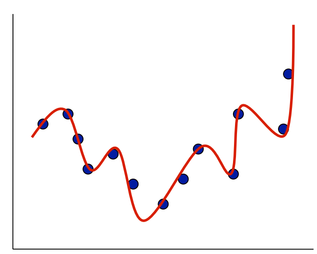
Bias: How well does the model predict the training data?

- 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

Bias/variance trade-off



High variance

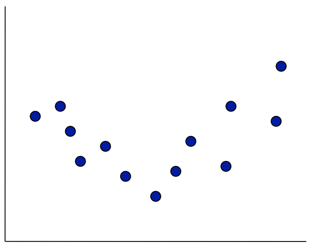
Bias: How well does the model predict the training data?

- 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

Bias/variance trade-off



What do we want?

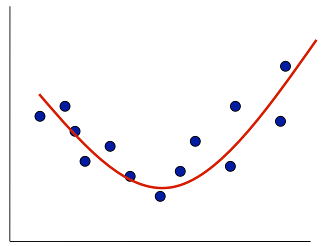
Bias: How well does the model predict the training data?

- 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

Bias/variance trade-off



Compromise between bias and 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