Support Vector Machines

Some slides borrowed from Andrew Moore’s slides on SVMs. His repository is here: http://www.cs.cmu.edu/~awm/tutorials.
Support Vector Machines

• Very popular ML technique
  – Became popular in the late 90s (Vapnik 1995; 1998)
  – Invented in the late 70s (Vapnik, 1979)

• Controls complexity and overfitting, so works well on a wide range of practical problems

• Can handle high dimensional vector spaces, which makes feature selection less critical

• Fast and memory efficient implementations, e.g., `svm_light`

• Not always best solution, especially for problems with small vector spaces
Linear Classifiers

- denotes +1
- denotes -1

\[ f(x, w, b) = \text{sign}(w \cdot x - b) \]

How would you classify this data?
Linear Classifiers

\[ f(x, w, b) = \text{sign}(w \cdot x - b) \]

How would you classify this data?

- \( \bullet \) denotes +1
- \( \circ \) denotes -1
Linear Classifiers

\[ f(x, w, b) = \text{sign}(w \cdot x - b) \]

How would you classify this data?

- \( \bullet \) denotes +1
- \( \circ \) denotes -1
Linear Classifiers

\[ f(x, w, b) = \text{sign}(w \cdot x - b) \]

How would you classify this data?

- denotes +1
- denotes -1
Linear Classifiers

\[ f(x, w, b) = \text{sign}(w \cdot x - b) \]

- denotes +1
- denotes -1

Any of these would be fine..
..but which is best?
Classifier Margin

A linear classifier’s \textbf{margin} is width that boundary could be increased by before hitting a datapoint.

\[ f(x, w, b) = \text{sign}(w \cdot x - b) \]
Maximum Margin

\[ f(x, w, b) = \text{sign}(w \cdot x - b) \]

- denoted +1
- denoted -1

Maximum margin linear classifier is the linear classifier with the largest margin

The simplest kind of SVM, called an LSVM
Maximum Margin

Maximum margin linear classifier is the linear classifier with the largest margin.

The simplest kind of SVM, called an LSVM.

\[ f(x, w, b) = \text{sign}(w \cdot x - b) \]

Support Vectors are the datapoints that margin pushes up against.

- \( \bullet \) denotes +1
- \( \circ \) denotes -1

Copyright © 2001, 2003, Andrew W. Moore
Why Maximum Margin?

- denotes +1
- denotes -1

Support Vectors are those datapoints that the margin pushes up against.

LOOCV = leave one out cross validation

1. Intuitively this feels safest
2. Small errors in boundary location unlikely to cause misclassification
3. LOOCV is easy since model is immune to removal of non-support-vector datapoints
4. Empirically it works very very well

This is the simplest kind of SVM (Called an LSVM)
Specifying a line and margin

- How do we represent this mathematically?
- ...in $m$ input dimensions?
Specifying a line and margin

- **Plus-plane** = \{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = +1 \}
- **Minus-plane** = \{ \mathbf{x} : \mathbf{w} \cdot \mathbf{x} + b = -1 \}

Classify as...

- **+1** if \( \mathbf{w} \cdot \mathbf{x} + b \geq 1 \)
- **-1** if \( \mathbf{w} \cdot \mathbf{x} + b \leq -1 \)
- Universe explodes if \( -1 < \mathbf{w} \cdot \mathbf{x} + b < 1 \)
Learning the Maximum Margin Classifier

• Given a guess of $w$ and $b$ we can
  • Compute whether all data points in the correct half-planes
  • Compute the width of the margin
• Write a program to search the space of $w$s and $b$s to find widest margin matching all the datapoints.
Learning SVMs

• Trick #1: Find points that would be closest to optimal separating plane ("support vectors") and work directly from those instances

• Trick #2: Represent as a **quadratic optimization problem**, and use quadratic programming techniques

• Trick #3 ("kernel trick"):
  – Instead of using raw features, represent data in a high-dimensional feature space constructed from a set of *basis functions* (e.g., polynomial combinations of the base features)
  – Find separating plane/SVM in that space
  – Voila: A nonlinear classifier!
Soft margin classification

• What if data from two classes not linearly separable?
• Allow a fat decision margin to make a few mistakes
• Some points, outliers or noisy examples, are inside or on wrong side of the margin
• Each outlier incurs a cost based on distance to hyperplane
Kernel trick

• What if data from two classes not linearly separable?
• Project data onto a higher dimensional space where it becomes linearly separable
• Many SVMs can take an argument, a kernel, that does the transformation of the data
• Deciding what kernel function to use is done through experimentation
Kernel Trick example

Can’t separate the blue & red points with a line

Figure from S. Bhattacharyya
Use a different kernel

- Applying a kernel can transform data to make it more nearly linearly separable
- E.g., use polar coordinates or map to three dimensions
SVM Performance

- SVMs can handle very large features spaces (e.g., 100K features)
- Relatively fast
- Anecdotally they work very well indeed
- Example: They are among the best-known classifier on a well-studied hand-written character recognition benchmark
100k features?

• SVM used for simple text classification, e.g. classify a text document as
  – expressing sentiment (positive, negative) or political learning (republican, democrat)

• Typically use each word in fixed vocabulary of 25-100k words as a feature/dimension
  – value is TF/IDF number -- frequency of the word in the document relative to all documents

• A product review might have ~200 unique words, social media posts only < 20 unique words

• SVMs typically use sparse matrices for efficiency
Binary vs. multi classification

• SVMs only do **binary** classification 😞
  – E.g.: can’t classify an iris into one of three species
• Note: common for many ML classifiers
• Two approaches to multi classification: OVA and OVO
• Consider Zoo dataset, which classifies animals into one of 7 classes based on 17 attributes
  – **Classes**: mammal, bird, reptile, fish, amphibian, insect, invertebrate
  – **Attributes**: hair, feathers, eggs, milk, aquatic, toothed, fins, ...
OVA or one-vs-all classification

• OVA or one-vs-all: turn n-way classification into n binary classification tasks.

• For zoo problem, train and run 7 binary classifiers:
  – mammal vs. not-mammal
  – fish vs. not-fish
  – bird vs. not-bird, ...

• Pick the one that gives the highest score
  – For an SVM this could be measured the one with the widest margin
OVO or one vs one classification

• OVO or one vs one: turn n-way classes into $N*(N-1)/2$ one-vs-one classifiers
  – mammal vs. bird, mammal vs. reptile ...
  – bird vs. reptile, bird vs. fish, ...
  – fish vs. amphibian, fish vs. insect, ...

• Use resulting scores to choose the classification that wins the most 1x1 pairings
SVMs in scikit-learn

• Scikit-learn has three SVM classifiers: SVC, NuSVC, and LinearSVC
• Data can be either in dense numpy arrays or sparse scipy arrays
• All directly support multi-way classification, SVC and NuSVC using OVO and LinearSVC using OVA
SVM Summary

• SVM is a good classification technique for problems with a large feature space
• Relatively fast to train and apply the model
• The kernel trick can help make some problems more-nearly linearly separable
• Their binary nature makes them a poorer fit for multi-way classification