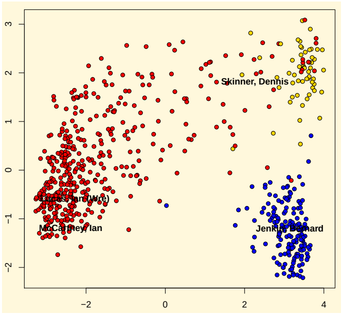
## Where Are We?

- Covered dictionary and related approaches to document classifications
- Continue this idea, but in a more formal modeling way: Naive Bayes
- Look at ways to score texts for ideology
- Discuss opportunities for fast, reliable coding of training set.

## Recall...

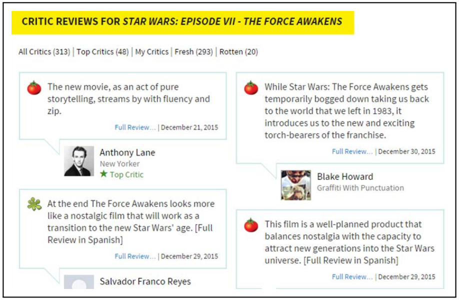
Unsupervised techniques: learning (hidden or latent) structure in unlabeled data.

e.g. PCA of legislators's votes: want to see how they are organized— by party? by ideology? by race?



Supervised techniques: learning relationship between inputs and a labeled set of outputs.

e.g. opinion mining: what makes a critic like or dislike a movie ($y \in \{0, 1\}$)?



**CRITIC REVIEWS FOR *STAR WARS: EPISODE VII - THE FORCE AWAKENS***

All Critics (313) | Top Critics (48) | My Critics | Fresh (293) | Rotten (20)

The new movie, as an act of pure storytelling, streams by with fluency and zip.

Full Review... | December 21, 2015

Anthony Lane
New Yorker
★ Top Critic

At the end The Force Awakens looks more like a nostalgic film that will work as a transition to the new Star Wars' age. [Full Review in Spanish]

Full Review... | December 29, 2015

Salvador Franco Reyes

While Star Wars: The Force Awakens gets temporarily bogged down taking us back to the world that we left in 1983, it introduces us to the new and exciting torch-bearers of the franchise.

Full Review... | December 30, 2015

Blake Howard
Graffiti With Punctuation

This film is a well-planned product that balances nostalgia with the capacity to attract new generations into the Star Wars universe. [Full Review in Spanish]

Full Review... | December 29, 2015

Naive Bayes

## Naive Bayes Classification

- Motivation: emails $d$ arrive and must be classified as belonging to one of two classes $c \in \{\text{spam}, \text{ham}\}$.
- Use the word/feature frequencies the emails contain.
- Naive Bayes, is a family of classifiers which apply Bayes's theorem and make 'naive' assumptions about independence between the features of a document.

$\rightarrow$ Fast, simple, accurate, efficient and therefore popular.

## Many different uses for Naive Bayes Classification

- Face recognition
- Weather prediction
- Medical diagnosis
- Spam detection
- Age/gender identification
- Language identification
- Sentiment analysis
- Authorship identification
- News classification

## Set up

- We're interested in the probability that an email is in a given category, given its features—i.e. frequency of terms.

- The conditional probability of a term $t_k$ occurring in a document, given that document is of class $c$, is $= \Pr(t_k|c)$

- e.g. probability of seeing '`beneficiary`' in a spam email might be 0.9, because a lot of spam emails use that term.

→ We are assuming terms basically occur randomly throughout the document/no position effects

- We can write the probability that a given email $d$ contains all the terms, if it's from a class $c$, as

$$\Pr(d|c) = \prod_{k=1}^{K} \Pr(t_k|c)$$

→ But this is not what we want: we want $\Pr(c|d)$.

## Reminder: Bayes' Theorem

Recall that:

$$\Pr(A|B) = \frac{\Pr(A,B)}{\Pr(B)}$$

- the probability that $A$ occurs given that $B$ occurred = the probability of both $A$ and $B$ occurring, divided by the probability that $B$ occurs.

e.g. you know a die shows an odd number, what is the probability that this odd number is 3? $\Pr(3|\text{odd}) = \frac{\frac{1}{6}}{\frac{1}{2}} = \frac{1}{3}$.

- of course, it is also true that $\Pr(B|A) = \frac{\Pr(B,A)}{\Pr(A)}$.
- but then, since $\Pr(A,B) = \Pr(B,A)$, we must have $\Pr(A|B)\Pr(B) = \Pr(B|A)\Pr(A)$, and thus… Bayes' law

$$\boxed{\Pr(A|B) = \frac{\Pr(A)\Pr(B|A)}{\Pr(B)}.}$$

## And...

- Our interest is in $\Pr(A|B) = \frac{\Pr(A)\Pr(B|A)}{\Pr(B)}$.

- Notice that $\Pr(B)$ itself does not tell us whether a particular value of $A$ is more or less likely to be observed, so drop it and rewrite:

$$\Pr(A|B) \propto \Pr(A)\Pr(B|A)$$

  Here, $\Pr(A)$ is our prior for $A$, while $\Pr(B|A)$ will be the likelihood for the data we saw.

## So. . .

We can express our quantity of interest as:

$$\Pr(c|d) = \frac{\Pr(c)\Pr(d|c)}{\Pr(d)}$$

and

$$\Pr(c|d) \propto \underbrace{\Pr(c)}_{\text{prior}} \underbrace{\prod_{k=1}^{K} \Pr(t_k|c)}_{\text{likelihood}}$$

where $\Pr(c)$ is the prior probability of a document occurring in class $c$; and $\Pr(t_k|c)$ is interpreted as "measure of the how much evidence $t_k$ contributes that $c$ is the correct class"

## Goal

We want to classify new data, based on patterns we observe in our training set (which we will classify by hand).

e.g. We look at 10,000 emails and classify them as $c \in \{\text{spam}, \text{ham}\}$. We use that information, and the terms associated with the two classes, to categorize tomorrow's email.

In particular, we typically want to assign the document to a single best class.

→ The 'best' class is the maximum a posteriori class, $c_{map}$:

$$c_{map} = \arg\max_c \widehat{\Pr(c|d)} = \arg\max_c \widehat{\Pr(c)} \prod_{k=1}^{K} \widehat{\Pr(t_k|c)}$$

## Estimation Notes I

The 'hats' appear because neither $\widehat{\Pr(c)}$ nor $\widehat{\Pr(t_k|c)}$ are known.

$\rightarrow$ they are estimated from the training set.

We can use $\frac{N_c}{N}$ for $\widehat{\Pr(c)}$, where $N_c$ is the number of documents in class $c$ in our training set.

We can use $\frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$ for $\widehat{\Pr(t_k|c)}$.

here $T_{ct}$ is the number of occurrences of $t$ in training documents that come from class $c$, including multiple occurrences.

and denominator is the total number of all terms in the training documents in $c$.

## Example

| | email | words | classification |
|---|---|---|---|
| training | 1 | money inherit prince | spam |
| | 2 | prince inherit amount | spam |
| | 3 | inherit plan money | ham |
| | 4 | cost amount amazon | ham |
| | 5 | prince william news | ham |
| test | 6 | prince prince money | ? |

$$\Pr(\text{prince}|\text{ham}) = \tfrac{1}{9}$$
$$\Pr(\text{prince}|\text{ham}) = \tfrac{1}{9}$$
$$\Pr(\text{money}|\text{ham}) = \tfrac{1}{9}$$
$$\Pr(\text{ham}|\text{d}) \propto \tfrac{3}{5}\tfrac{1}{9}\tfrac{1}{9}\tfrac{1}{9} = 0.00082$$

$$\Pr(\text{prince}|\text{spam}) = \tfrac{2}{6}$$
$$\Pr(\text{prince}|\text{spam}) = \tfrac{2}{6}$$
$$\Pr(\text{money}|\text{spam}) = \tfrac{1}{6}$$
$$\Pr(\text{spam}|\text{d}) \propto \tfrac{2}{5}\tfrac{2}{6}\tfrac{2}{6}\tfrac{1}{6} = 0.0074$$

$$\rightarrow \boxed{c_{map} = \text{spam}}$$

## Estimation Notes II

- $\widehat{\Pr(t_k|c)}$ is the fraction of tokens in documents from class $c$ that are $t$. Can also see it as fraction of positions in documents from class $c$ that contain term $t$. This is a multinomial NB model.
- Could have $\widehat{\Pr(t_k|c)}$ as fraction of documents containing $t$, which implies Bernoulli model (ignores number of occurrences).
- As usual, working with products of probabilities is difficult computationally

$\rightarrow$ take logs:

$$c_{map} = \arg\max_c \left[\log \widehat{\Pr(c)} + \sum \log \widehat{\Pr(t_k|c)}\right]$$

## Estimation Notes III

- Sparsity can be a problem in the training set. Suppose, in our training set of spam emails, we never see the word '`cost`' (but it does occur in the ham set), and it shows up in our actual email tomorrow.

Q What's the probability that email is spam?

→ well, $\widehat{\Pr(t_k|c)} = \widehat{\Pr(\text{`cost'}|}\text{spam}) = 0$. And that will be multiplied into the product. So, $\Pr(\text{spam}|d) = 0$.

- May want to add one to each count: $\frac{T_{ct}+1}{\sum_{t' \in V}(T_{ct'}+1)}$ to avoid wiping out the products (or causing problems for taking logs). Equivalent to adding size of the vocabulary to the counts within the class.

→ Laplace smoothing, equivalent to a uniform prior on term (each term occurs once for each class).

## Classifier is 'Naive'...

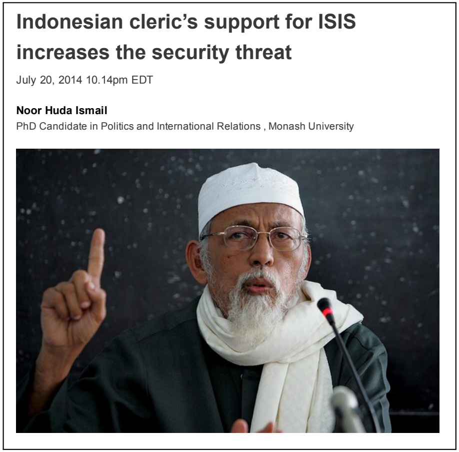
1. We assume conditional independence: probability that a particular feature occurs is independent of any other feature occurring, once we condition on a given category.

e.g. probability of observing '`money`' is independent of probability of observing '`dollars`' given the emails are spam. This implies $\Pr(\text{money}|c) = \Pr(\text{money}|c, \text{dollars})$, enables us to write everything as a simple product, $\prod_{k=1}^{K} \widehat{\Pr(t_k|c)}$.

2. We assume positional independence: probability that a term occurs in a particular place is constant for the entire document. This implies we only need one probability distribution of terms, and that it's valid for every position.

e.g. probability of observing '`dear`' is the same regardless of which word in the document we are considering (1st, 2nd, 3rd etc). Equivalent to bag of words.

## Example: Jihadi Clerics



**Indonesian cleric's support for ISIS increases the security threat**

July 20, 2014 10.14pm EDT

**Noor Huda Ismail**
PhD Candidate in Politics and International Relations , Monash University

Nielsen (2012) investigates why certain scholars of Islam become Jihadi: i.e. why they encourage armed struggle (especially against the west)

Requires that he first classifies scholars as Jihadi and $\neg$ Jihadi: has 27,142 texts from 101 clerics, and difficult to do by hand.

## Jihadi Clerics

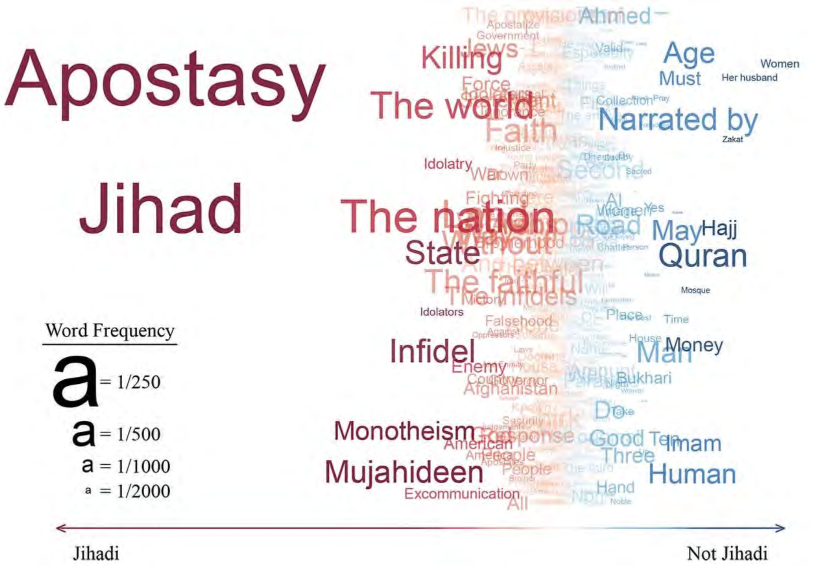
Training set: self-identified Jihadi texts (765), and sample from Islamic website as $\neg$ Jihadi (1951)

Preprocess: drops terms occurring in less than 10%, or more than 40% of documents, and uses 'light' stemmer for Arabic

Can assign a *Jihad Score* to each document: basically the logged likelihood ratio, $\sum_i \log \frac{\Pr(t_k|\text{Jihad})}{\Pr(t_k|\neg\text{ Jihad})}$ (note: doesn't know what 'real world' priors are, so drops them here)

Then for each cleric, concatenate all works into one and give this 'document'/cleric a score.

## Discriminating Words



## Validation: *Exoneration*

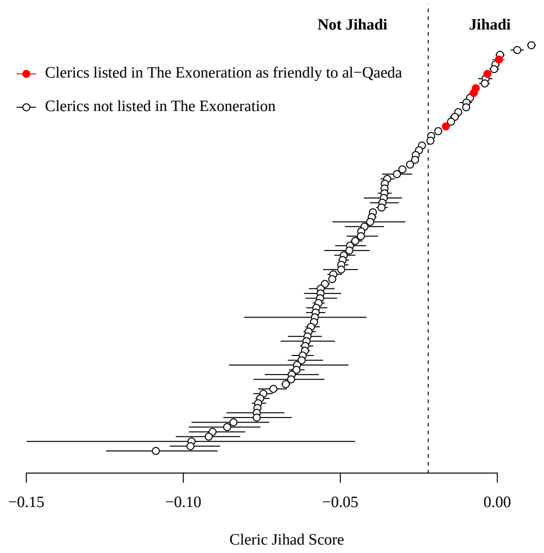


**Figure 4.9:** *Jihad Scores Predict Inclusion in The Exoneration*

## Scoring and Scaling Texts

## Wordscores (Laver, Benoit & Garry, 2003)

- Long standing interest in scaling political texts relative to one another:

e.g. are parties moving together over time, such that manifestos are converging?

e.g. do members of parliament speak in line with their constituency's ideology (roll calls typically uninformative)?

→ LBG suggest a way of scoring documents in a NB style, so that we can answer such questions.

## Basics

1 Begin with a reference set (training set) of texts that have known positions.

e.g. we find a 'left' document and give it score $-1$; and a 'right' document and give it score 1

2 Generate word scores from these reference texts

3 Score the virgin texts (test set) of texts using those word scores.

## Scoring the words

- Suppose we have a given reference document $R$, which is scored as $A_R = 1$. E.g. Neo-Nazi manifesto.
- In document $R$, count the number of times word $i$ occurs, denote as $f_{iR}$. Also record the total number of words in document $R$, and denote as $W_R$ .
- Do the same for Communist party manifesto $L$, which we score as $A_L = -1$. Then calculate $f_{iL}$ and $W_L$.
- Define $P_{iR}$ as the probability of word $i$ given we are in document $R$,

$$P_{iR} = \frac{\frac{f_{iR}}{W_R}}{\frac{f_{iR}}{W_R} + \frac{f_{iL}}{W_L}}$$

- define $P_{iL}$ in similar way.

## Score of a given word $i$...

- ...is then
$$S_i = A_L P_{iL} + A_R P_{iR},$$
- which in our simple case is $S_i = P_{iR} - P_{iL}$.
- and the score of a virgin document is then
$$S_V = \sum_i \frac{f_{iV}}{W_V} \cdot S_i$$

NB $S_V$ is the mean of the scores of the words in $V$ weighted by their term frequency.

NB any new words in the virgin document that were *not* in the reference texts are ignored: the sum is only over the words we've seen in the reference texts.

## Example

- Neo-Nazi manifesto uses '`immigrant`' 25 times in 1000 words, while Communists use it only 5 times.

then $P_{iR} = \frac{0.025}{0.025+0.005} = 0.83.$

and $P_{iL} = \frac{0.005}{0.025+0.005} = 0.16.$

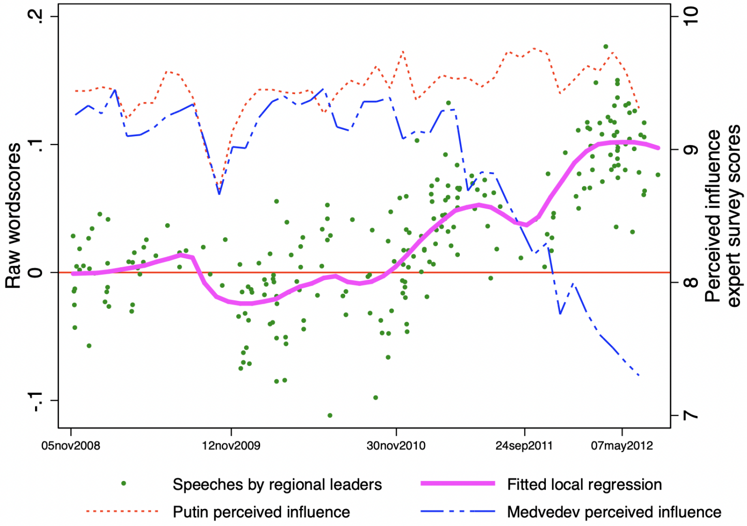
- So $S_i = 0.83 - 0.16 = 0.66$

- We see a virgin manifesto, from the Conservative party, and it mentions `immigrant` 20 times in a thousand words.

the relevant calculation for that word is $0.02 \times 0.66 = 0.0132.$

the virgin manifesto, from Labour party, mentions it 10 times in a thousand words: $0.01 \times 0.66 = 0.006$

## Baturo & Mikhaylov (2013



## Comments

- Extremely influential approach: avoids having to pick features of interest (features that don't distinguish between reference texts have $S_i = 0$)
- Helpful/valid in practice and can have uncertainty estimates to boot.
- Important to obtain extreme and appropriate reference, and score them appropriately. Need to be from domain of virgin texts, and have lots of words.
- Lowe 2008) unhappy : no statistical model, inconsistent scoring assumptions, and difficult to pick up 'centrist language' (is equivalent to any language used commonly by all parties for linguistic reasons).

## Classifier Performance and Comparison

## Performance of Classifiers

How do we evaluate whether our classifier (for documents) is any good?

Think about a two class problem. Suppose we have particular interest in identifying all the Jihadist documents.

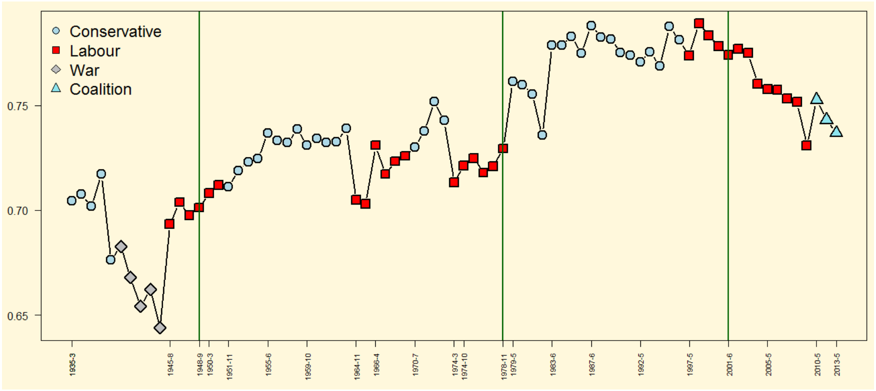
| | |
|---|---|
| TP | the document should be placed in $c$, and method placed it in $c$, we have a true positive. |
| FP | the document should be placed in $\neg c$, and method placed it in $c$, we have a false positive (type I error). |
| FN | the document should be placed in $c$, and method placed it in $\neg c$, we have a false negative (type II error). |
| TN | the document should be placed in $\neg c$, and method placed it in $\neg c$, we have a true negative. |

## Confusion Matrix

| | | Predicted | | |
|---|---|---|---|---|
| | | J | $\neg J$ | Total |
| Actual | J | $a$ TP | $b$ FN | $a+b$ |
| | $\neg J$ | $c$ FP | $d$ TN | $c+d$ |
| | Total | $a+c$ | $b+d$ | $N$ |

- **Accuracy** : $\frac{\text{number correctly classified}}{\text{total number of cases}} = \frac{a+d}{a+b+c+d}$
- **Precision** : $\frac{\text{number of TP}}{\text{number of TP}+\text{number of FP}} = \frac{a}{a+c} \rightarrow$ Fraction of the documents predicted to be $J$, that were in fact $J$.
- **Recall** : $\frac{\text{number of TP}}{\text{number of TP} + \text{number of FN}} = \frac{a}{a+b} \rightarrow$ Fraction of the documents that were in fact $J$, that method predicted were $J$.
- **F** : $2\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}} \rightarrow$ The harmonic mean of precision and recall.

## Aside: Sometimes Classifier Performance is Substantively Meaningful



Use machine to classify left ($-1$) vs right ($+1$) MPs in UK and record classification accuracy. When high, parties are more polarized.

Makes sense in terms of historical record!

## Where Are We?

We've looked at some simple (but high performing) supervised learning ideas for estimating the class of individual documents (Naive Bayes), and for estimating proportions.

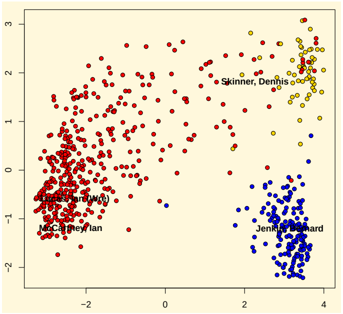
Now want to cover powerful, commonly used techniques from machine learning that appear in social science research: $\boxed{\text{SVM}}$, KNN, etc. . .

These techniques involve some important decisions about the bias-variance tradeoff, and the use of (cross) validation in checking model performance and selecting the best model.

# Remember. . .

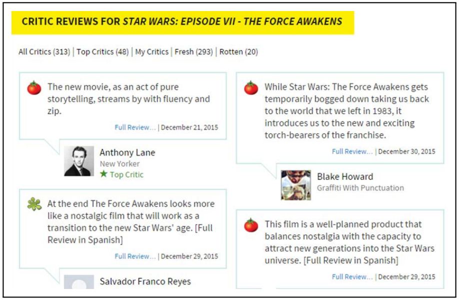
Unsupervised techniques: learning (hidden or latent) structure in unlabeled data.

e.g. PCA of legislators's votes: want to see how they are organized— by party? by ideology? by race?



Supervised techniques: learning relationship between inputs and a labeled set of outputs.

e.g. opinion mining: what makes a critic like or dislike a movie ($y_i \in \{0, 1\}$)?



CRITIC REVIEWS FOR *STAR WARS: EPISODE VII - THE FORCE AWAKENS*

All Critics (313) | Top Critics (48) | My Critics | Fresh (293) | Rotten (20)

The new movie, as an act of pure storytelling, streams by with fluency and zip.

Full Review... | December 21, 2015

Anthony Lane
New Yorker
★ Top Critic

While Star Wars: The Force Awakens gets temporarily bogged down taking us back to the world that we left in 1983, it introduces us to the new and exciting torch-bearers of the franchise.

Full Review... | December 30, 2015

Blake Howard
Graffiti With Punctuation

At the end The Force Awakens looks more like a nostalgic film that will work as a transition to the new Star Wars' age. [Full Review in Spanish]

Full Review... | December 29, 2015

Salvador Franco Reyes

This film is a well-planned product that balances nostalgia with the capacity to attract new generations into the Star Wars universe. [Full Review in Spanish]

Full Review... | December 29, 2015

## Workflow of Supervised Learning: Bias/Variance Tradeoff

## Workflow of Supervised Techniques

1. Obtain/code the training set and decide on relevant features (preprocess).
2. Decide on the algorithm, possibly matched in some way to nature of problem.
3. Adjust algorithm for optimal performance, perhaps using validation set and/or some kind of cross-validation.
4. Report accuracy in test set.

## Notes and Issues

Supervised techniques are about learning relationship between $X$ and labeled data. Often used interchangeably with *machine learning*: idea that computers could ‘learn’ relationships without specific programming.

Often used when $p >> n$: number of parameters (e.g. coefficients on terms) is far larger than number of observations (e.g. speakers or documents).

$\rightarrow$ results in general curse of dimensionality wherein feature matrix is large (e.g. $100k$ columns) and sparse and thus obtaining meaningful estimates is difficult.

So techniques may require careful tuning of *regularization parameters* to obtain good performance.

## Notes and Issues II

Once we have the training set we face a dilemma...

1 we can use our technique to fit a very complicated model to this data (perfectly), including noisy elements. This avoids bias, but it incurs variance (when we move to the test set).

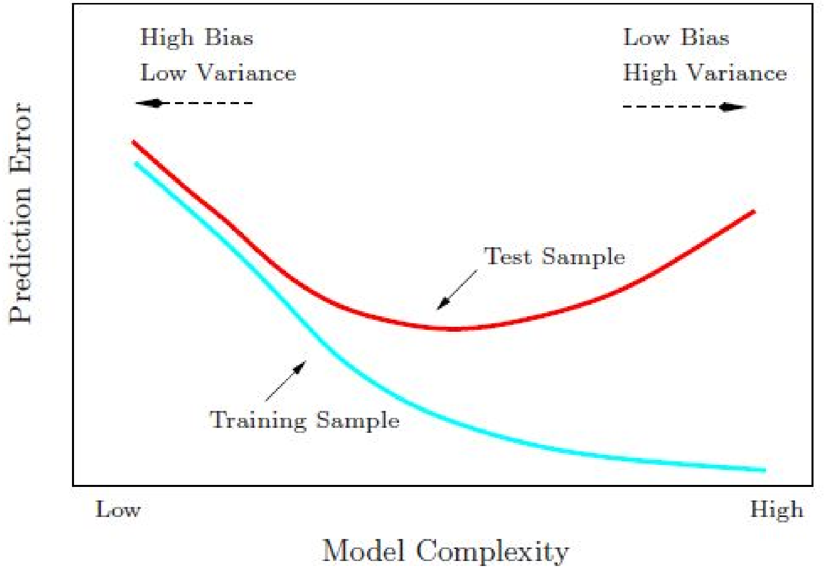
$\rightarrow$ we have overfit to our training set, and may do poorly on our test set.

2 we can use be more relaxed about the fit of our algorithm in the training set, and accept some imperfections in performance. This avoids high variance, but may induce bias in the sense that we miss important relationships in the data.

$\rightarrow$ we have underfit to our training set, and may do poorly on our test set.

So managing the *bias-variance tradeoff* is a key element of supervised learning, and we may need to tune our algorithms with that in mind.

## Bias-Variance Tradeoff (Hastie et al, p38)



## Cross-Validation

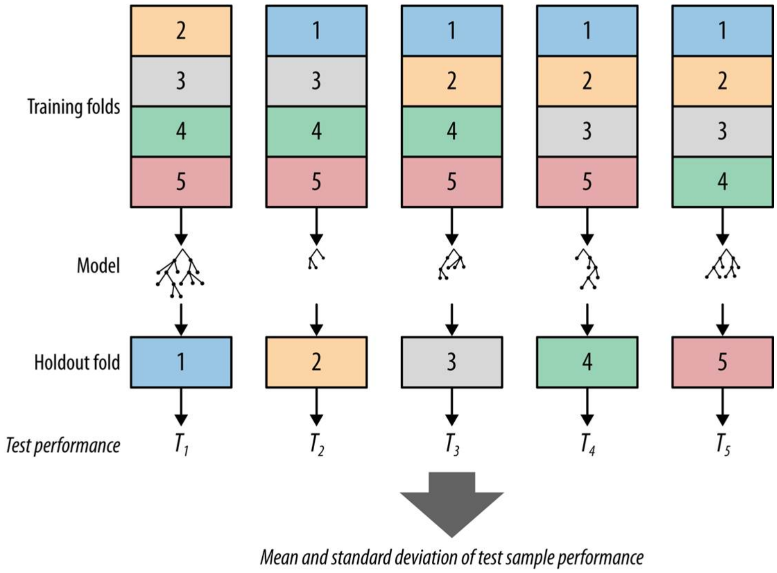
Want to properly estimate model performance (bias and especially variance).

Popular and efficient approach is $k$-fold cross validation, esp when we don't have enough data to form a completely separate validation set.

1. divide all data into $k$ equal size chunks ($k = 10$ is common; $k = n$ is 'leave one out'), and set the parameter(s) of model at particular value,
2. repeat the following $k$ times (folds):
   - 2.1 grab one of the $k$ chunks as a validation set (each only used once)
   - 2.2 grab the other $k - 1$ chunks as a training set
   - 2.3 test on the validation set, record prediction error
3. Average over runs to get prediction error estimate.

$\rightarrow$ Can follow same steps for models of different specifications; variant on this approach can be used for model selection, directly.

## Graphically



## Support Vector Machines

## Background

- Basics of the technique developed by Vladimir Vapnick for his PhD thesis in USSR in the 1960's.
- Never used because no computers powerful enough in USSR to apply it.
- Until. . . Vapnick emigrated to the US and was hired by Bell Labs in the 1990's.
- A very stable classifier. Used for text classification, character recognition, image recognition in general, etc. . . .

## Motivating Example Diermeier et al, 2011

Diermeier et al want to know what terms are most indicative of conservative or liberal positions in legislative debates.

Study the speeches of the 25 most liberal and the 25 most conservative senators (1989–2004), selected based on their NOMINATE scores (from roll call behavior, only)

Code the Republicans as $+1$, Dems as $-1$:
$y_i \in \{-1, +1\}$

## Methodological Problem

Want to know the relationship between using different terms and ideological views.

e.g. is '`Ethanol`' a 'Democrat' term or a 'Republican' term, and to what degree?

Have the (stemmed, stopped, weighted etc) speech term matrix for each Senator as $X$.

Their training set is speech output of most extreme Senators between the 101st and 107th Congress

Their test set is speech output of most extreme Senators in 108th Congress.

What method to use?

## Support Vector Machines

Idea a classifier that builds a model from a binary labeled training set and returns an optimal hyperplane that puts new examples into one of the categories non-probabilistically.

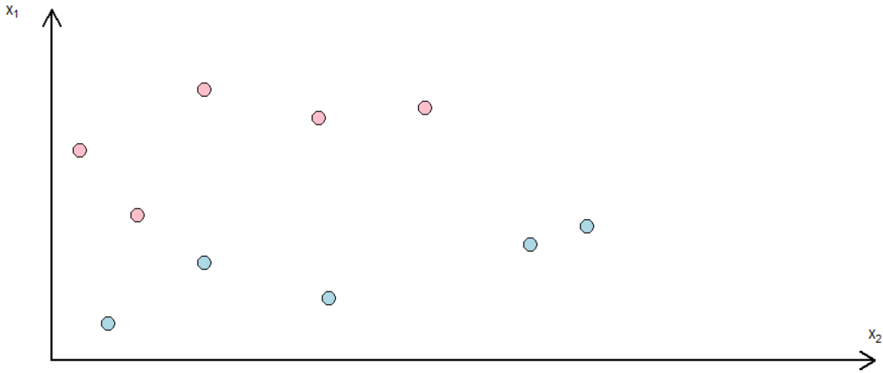
Here We have 10 Senators that belong to one of two classes: five Republicans, five Democrats. $y_i \in \{-1, +1\}$

Each senator has a number of $p$ features, which are the term weights from their speeches. To make things simple, suppose that $p = 2$: there are (only) two features, $x_1$ and $x_2$ per observation.

Assume that the observations are linearly separable: if we plot the Senators in two dimensions $(x_1, x_2)$, we can divide them (perfectly) into the two parties using a straight line.

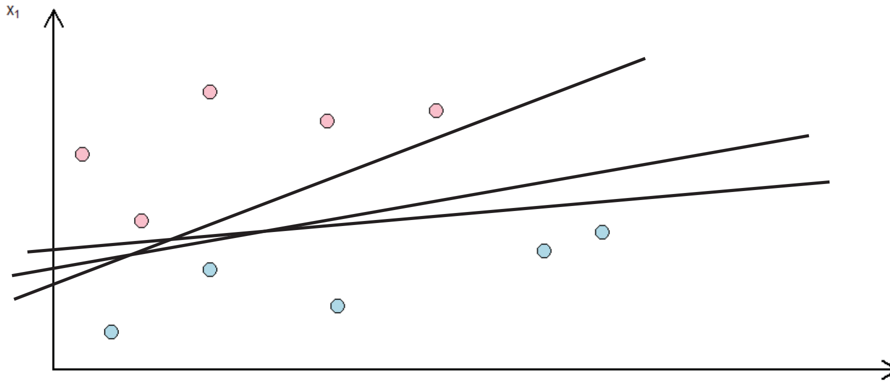
$\rightarrow$ of course, a real problem would have $p$ being large, and the space being very high dimensional, but the logic is the same.

## The 10 Senators



As the parties linearly separably? Where could you draw the line?

## The 10 Senators



Which line should we prefer?

## What is the Optimal Hyperplane?

There are many possible lines.

But we want to avoid ones that pass close to the points: such lines will tend to be sensitive to noise and make classification errors with future examples.

So pick line that gives largest minimum distance from the training cases. That is, the line that's as far as possible from the closest cases on both sides.

$\rightarrow$ That optimal line—the separating hyperplane—is the maximum margin hyperplane. It will maximize the margin of the training data.

## How to get it I: Notation Variants

We can write any line as $y = ax + b$ or $y - ax - b = 0$.

Often rewrite terms as $\mathbf{w} = [-b, -a, 1]$ and $\mathbf{x} = [1, x, y]$

$\rightarrow$ $\mathbf{w}^T\mathbf{x} = -b \times 1 + (-a) \times x + 1 \times y = y - ax - b = 0$.

or use dot product: $\mathbf{w} \cdot \mathbf{x} = 0$

Also popular: can use a two dimensional form—

$\mathbf{w} = [-a, 1]$ and $\mathbf{x} = [x, y]$, so that $\mathbf{w} \cdot \mathbf{x} = y - ax$

thus $\mathbf{w} \cdot \mathbf{x} - b = 0$ (since $0 = y - ax - b$)

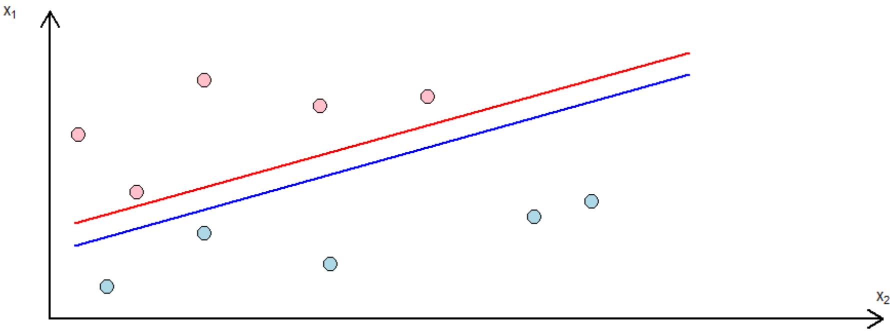
## How to get it II: working with the equation

Our hyperplane (line) will separate the data, and will satisfy $\mathbf{w} \cdot \mathbf{x} - b = 0$

We can think of it as the line that is equidistant, i.e. half-way between, two parallel hyperplanes ($H_R$ and $H_D$ ) that separate the Reps from the Dems.

Those parallel lines should be as far from each other as possible without misclassifying anybody: the distance between them is the margin and we want that to be maximized.

## How Could We Do Better?



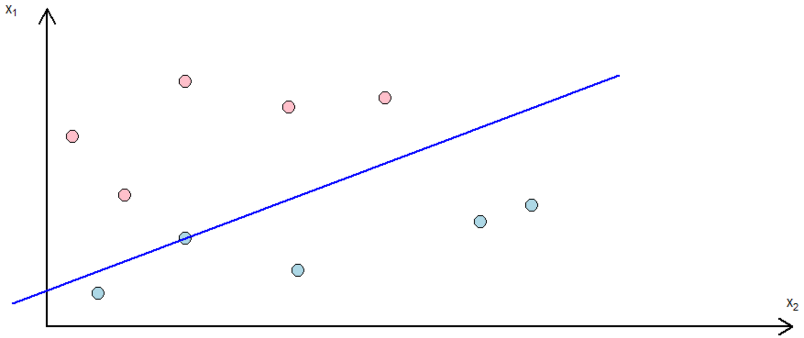
## How to get it II: working with the equation

Our hyperplane (line) will separate the data, and will satisfy $\mathbf{w} \cdot \mathbf{x} - b = 0$

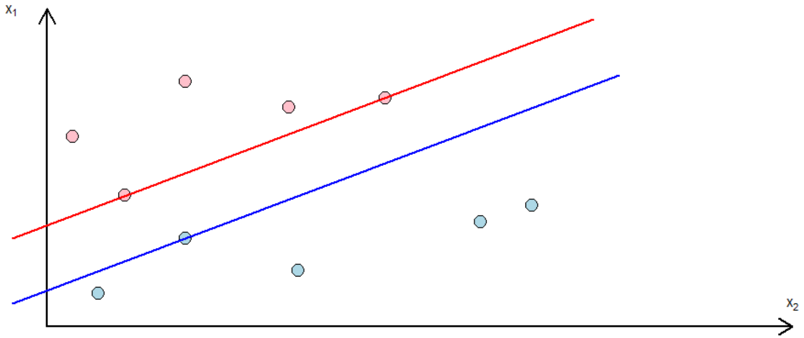
We can think of it as the line that is equidistant, i.e. half-way between, two parallel hyperplanes ($H_R$ and $H_D$) that separate the Reps from the Dems.

Those parallel lines should be as far from each other as possible without misclassifying anybody: the distance between them is the margin and we want that to be maximized.

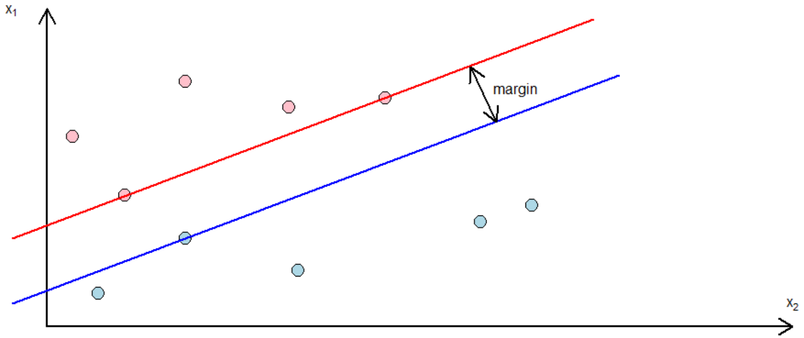
## Graphically. . .



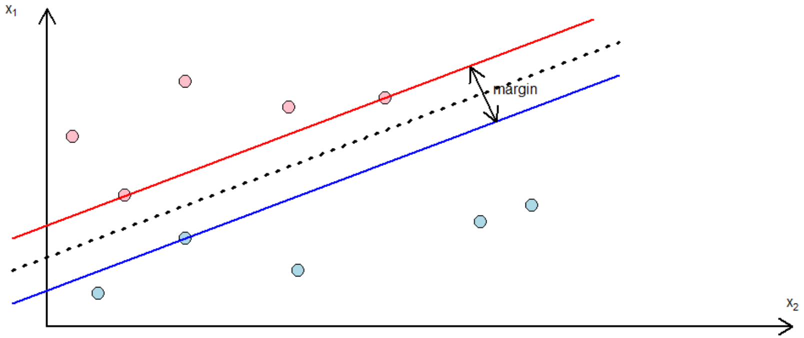
## Graphically. . .



## Graphically. . .



## Graphically. . .



## The parallel hyperplanes

We want all the Republicans ($y_i = 1$)—and *only* the Republicans—to be classified as Republicans (given their $x$s).

This means that 'their' hyperplane should 'capture' them all, and separate them fully from the Democrats (in our 2D space).

→ requires that $\mathbf{w} \cdot \mathbf{x} - b \geq 1$, if $y_i = 1$

Same idea for the Democrats: $\mathbf{w} \cdot \mathbf{x} - b \leq -1$, if $y_i = -1$

The distance between the two hyperplanes ($H_R$ and $H_D$) we construct is the margin, and its width is $\frac{2}{||\mathbf{w}||}$, where $||\mathbf{w}||$ is the norm of $\mathbf{w}$.

→ This is not trivially obvious! If you want more background, here is an incredible lecture by Patrick Winston at MIT. . . .

→ By the way this talk by Winston on "How to Speak" was legendary at MIT. You should all watch it as well.

## Optimization Problem

minimize $||\mathbf{w}||$ subject to $y_i(\mathbf{w} \cdot \mathbf{x} - b) \geq 1 \ \forall i$

This will give us the two hyperplanes $H_R$ and $H_D$, and the line equidistant between them will be our classifier.

Turns out that the minimization of $||\mathbf{w}||$ is amenable to quadratic programming methods (the economists in the room are familiar with at least one of these—constrained optimization using the Lagrangian multiplier).

## Notes

At least one of the Senators, in one of the parties, will be on ‘their’ line. Why?

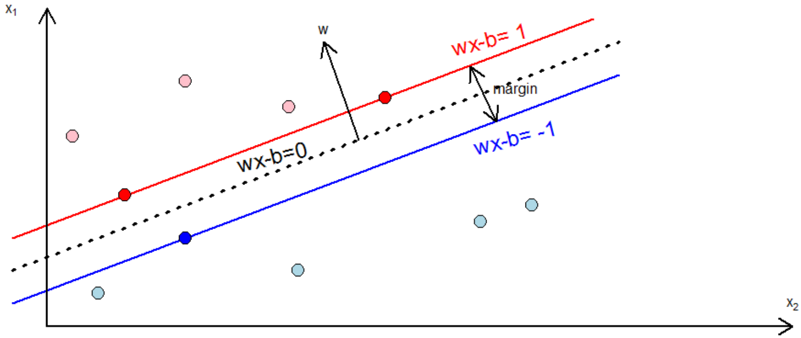
A: Suppose nobody is on either line. Then we must be able to move the lines ‘apart’ a little further and make the margin bigger. . .

But as soon as someone is pushed between the lines, we’ve broken the rule about no misclassifications, and that’s a constraint we have to follow.

In fact, the points closest to the separating hyperplane will be the Senators lying on their (respective) parallel hyperplanes.

We use the term support vectors to describe the training examples closest to the hyperplane. Those support vectors completely determine where our maximum-margin hyperplane will be.

## The Support Vectors



The support vectors lie on...

$\mathbf{w} \cdot \mathbf{x} - b = 1$ or $\mathbf{w} \cdot \mathbf{x} - b = -1$

## SVM weights

So We have a hyperplane that separates the Democrats from the Republicans.

The vector **w** is orthogonal to that, and when we multiply it (dot product) by **x** (and add *b* term) we get the predicted class of a 'new' Senator.

i.e. if the product is positive $\rightarrow$ Republican; if negative $\rightarrow$ Democrat

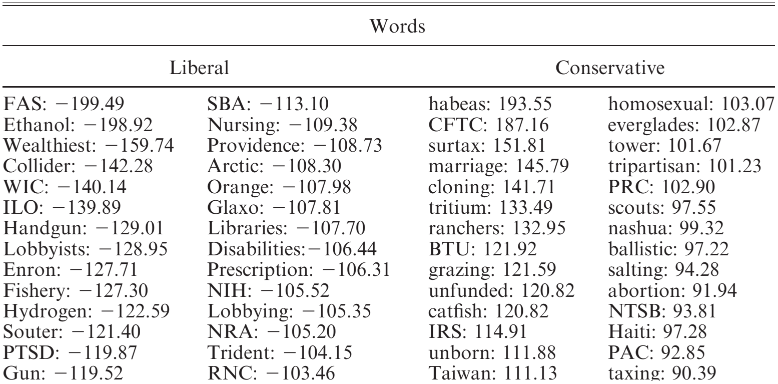
Plus for each feature, $x_1, x_2, \ldots$, the vector **w** gives us a weight. The absolute size of the weights—relative to one another—is a measure of how important that feature is for separating the Senators into Democrat and Republican.

NB SVMs typically have few features with non-zero weights, and those that are non-zero come from the support vectors (the 'important' observations)

## Back to Diermeier et al, 2011

Achieve 92% accuracy (!)

Sort words according to coefficients: very positive weights imply conservative words; very negative weights imply liberal words. Argue that it is ‘values’ rather than economics that separates liberals from conservatives.



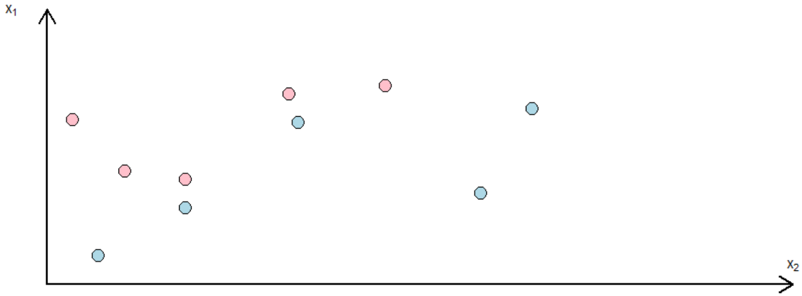
| Words | | | |
|---|---|---|---|
| Liberal | | Conservative | |
| FAS: −199.49 | SBA: −113.10 | habeas: 193.55 | homosexual: 103.07 |
| Ethanol: −198.92 | Nursing: −109.38 | CFTC: 187.16 | everglades: 102.87 |
| Wealthiest: −159.74 | Providence: −108.73 | surtax: 151.81 | tower: 101.67 |
| Collider: −142.28 | Arctic: −108.30 | marriage: 145.79 | tripartisan: 101.23 |
| WIC: −140.14 | Orange: −107.98 | cloning: 141.71 | PRC: 102.90 |
| ILO: −139.89 | Glaxo: −107.81 | tritium: 133.49 | scouts: 97.55 |
| Handgun: −129.01 | Libraries: −107.70 | ranchers: 132.95 | nashua: 99.32 |
| Lobbyists: −128.95 | Disabilities: −106.44 | BTU: 121.92 | ballistic: 97.22 |
| Enron: −127.71 | Prescription: −106.31 | grazing: 121.59 | salting: 94.28 |
| Fishery: −127.30 | NIH: −105.52 | unfunded: 120.82 | abortion: 91.94 |
| Hydrogen: −122.59 | Lobbying: −105.35 | catfish: 120.82 | NTSB: 93.81 |
| Souter: −121.40 | NRA: −105.20 | IRS: 114.91 | Haiti: 97.28 |
| PTSD: −119.87 | Trident: −104.15 | unborn: 111.88 | PAC: 92.85 |
| Gun: −119.52 | RNC: −103.46 | Taiwan: 111.13 | taxing: 90.39 |

## Beyond basic (hard margin) SVM

## What if. . .

The Senators were not linearly separable? (‘soft margin’ SVM problem)

## Graphically. . .



## What if. . .

The Senators were not linearly separable? (‘soft margin’ SVM problem)

Can introduce a hinge loss function into the minimization problem. . .

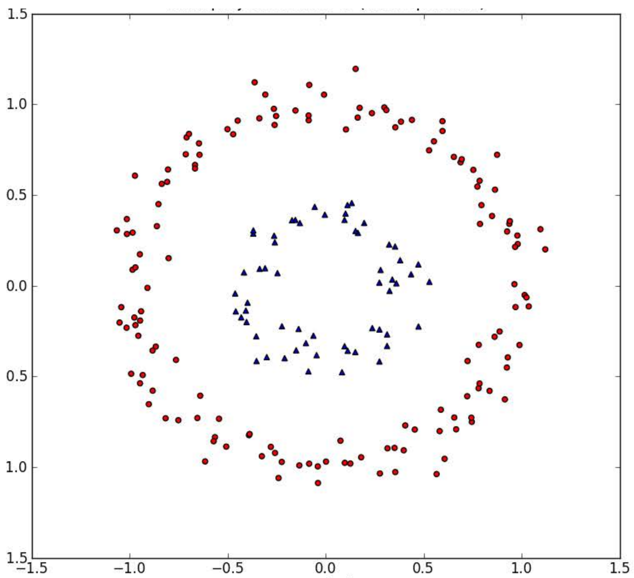
$= 0$ if the $x$s are on the ‘correct’ side of the margin. . .

And proportional to the distance from the margin *if* the point is on the ‘wrong’ side of the margin.

Hyperplane(s) will be drawn in way that is more sensitive to ‘bigger’ mistakes in classification.

## What if. . .

The situation was considerably ‘worse’, such that neither a ‘hard margin’ nor ‘soft margin’ linear approach would work?



from www.eric-kim.net

## What if. . .

The situation was considerably 'worse', such that neither a 'hard margin' nor 'soft margin' linear approach would work?

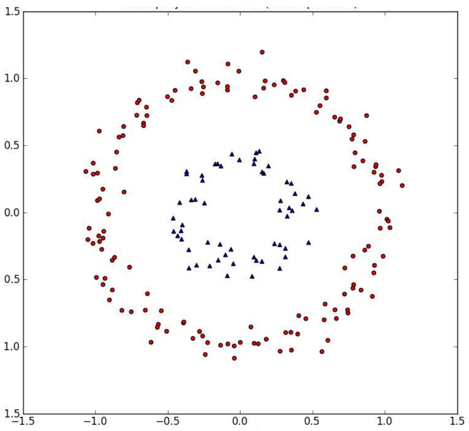
May be a way to transform the data, using a transformation $\phi$, such that it *can* now be separated using a linear classifier.

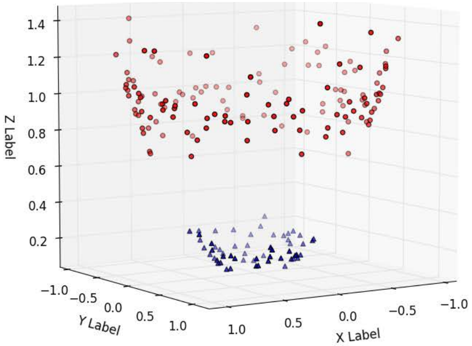
When the data is in a two dimensional feature space, we can lift it into a three dimensional feature space using

$$\phi(x_1, x_2) = (x_1, x_2, x_1^2 + x_2^2).$$

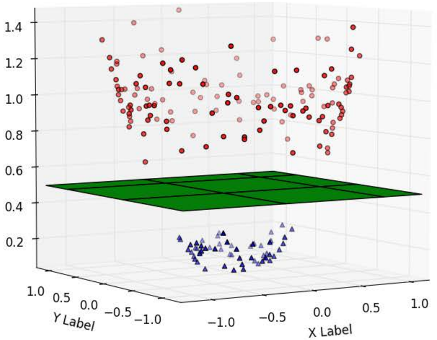
Then use a linear SVM on the transformed data set, and then map back to the original 2D space.

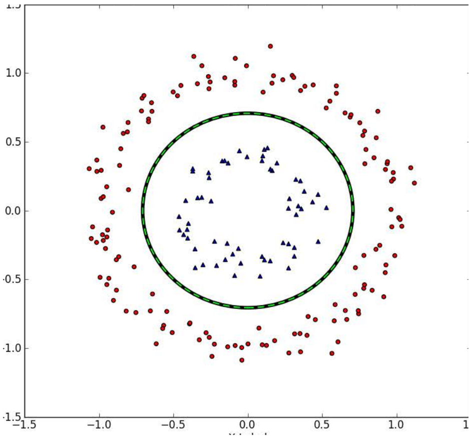
$\rightarrow$ Results in a non-linear hyperplane once back in 2 dimensions.





from `www.eric-kim.net`





from www.eric-kim.net

## Kernel Methods

Explicitly transforming data into a new space can be very expensive in terms of computation.

But What if we could do the transformation, and take the distances between observations implicitly, and use them for our classification?

$\rightarrow$ exactly what kernel methods do: use kernel functions, $K(\mathbf{x}_i, \mathbf{x}_j)$ to compute the $i, j$ pairwise dot products for training data *as if* it had been transformed. Can then feed those inner products to the classifier.

this 'kernel trick' cuts cost considerably, though choosing and tuning the 'correct' kernel may be difficult.

For text analysis, string kernels use a function $K(a, b)$ to implicity calculate the distance between strings of characters via the number of subsequences they have in common.