KE5205 TEXT MINING 2018

TEXT CATEGORIZATION

Institute of Systems Science

National University of Singapore

Objectives of this module

At the end of this module, you can:

- **Describe what is text categorization**
- **Understand how supervised text categorization works**
- **Evaluate a text categorization system with respect to a business scenario**

WHAT IS TEXT CATEGORIZATION?

Text categorization (also known as “classification”)

- Classification is the problem of identifying whether a new observation belongs to a set of categories.(determine if an object is a member of a set or not)

- Text categorization (a.k.a. text classification) is the task of assigning predefined categories to the documents.

Some Examples of Classification

- **Email Spam**. The goal is to predict whether an email is a *spam* and should be delivered to the Junk folder.

- **Credit Card Fraud Detection**: Given credit card transactions for a customer in a month, identify those transactions that were made by the customer and those that were not

- **Product Recommendation**: Given a purchase history for a customer and a large inventory of products, identify those products in which that customer will be interested and likely to purchase.

- **Sentiment analysis**: Given a hotel review, identify whether is a positive comment or a negative comment.

- **Image recognition:** Digits recognition

BUILDING A CLASSIFIER

Creating classifiers

- **Hand-coded classifiers (the “good old days!”)**
  - If <conditions> then <category> else NOT<category>

Example: If (basketball or football or tennis or Golf) then Sports News

From: F. Aiolli, *Text Categorization*, http://www.math.unipd.it/~aiolli/corsi/SI-0607/Lez09.251006.pdf

Naïve Bayes Model

You have a set of reviews (documents) and the class of the reviews

Doc	Text	Class
1	I loved the movie	+
2	I hated the movie	-
3	a great movie, good movie	+
4	Poor acting	-
5	great acting. a good movie	+
6	I hated the poor acting	?

Naïve Bayes Model

D_{new}= "I hated the poor acting"

Compare p(+|D_{new}) with p(-|D_{new})

For a document *d* and a class *c*

$$P(c \mid d) = \frac{P(d \mid c) P(c)}{P(d)}$$

Naïve Bayes Model – Bayes' Theorem

What is the probability that a hearts King got selected from a standard deck of cards (52)?

P(hearts and King)= ?

*P(hearts and King)=P(hearts)*P(King|hearts)*

*P(hearts and King)=P(King)*P(hearts|King)*

$$P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}$$

clubs (♣), diamonds (♦), hearts (♥) and spades (♠)

Naïve Bayes Model

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c) P(c)}{P(d)}$$

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

Naïve Bayes Model

Conditional Independence: Assume the feature probabilities $P(x_i|c_j)$ are independent given the class c

$$P(x_1, \ldots, x_n | c) = P(x_1 | c) \bullet P(x_2 | c) \bullet P(x_3 | c) \bullet \ldots \bullet P(x_n | c)$$

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i | c_j)$$

Naïve Bayes Model

D_{new}= “I hated the poor acting”

*p(+|D_{new})=p(+)*p(I|+)*p(hated|+)*p(the|+)*p(poor|+)*p(I|+)*p(acting|+)=?*

*p(-|D_{new})=p(-)*p(I|-)*p(hated|-)*p(the|-)*p(poor|-)*p(I|-)*p(acting|-)=?*

Naïve Bayes Model

- First attempt: maximum likelihood estimates
  - simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

Naïve Bayes Model

Laplace (add-1) smoothing for Naïve Bayes

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} \left(count(w, c) + 1 \right)}$$

$$= \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c) \right) + |V|}$$

Naïve Bayes Model

Laplace (add-1) smoothing: unknown words

$$\hat{P}(w_u \mid c) = \frac{count(w_u, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + \left|V + 1\right|}$$

$$= \frac{1}{\left(\sum_{w \in V} count(w, c)\right) + \left|V + 1\right|}$$

Naïve Bayes Model

DTM

Doc	Text	Class
1	I loved the movie	+
2	I hated the movie	-
3	a great movie, good movie	+
4	Poor acting	-
5	great acting. a good movie	+

Doc	I	loved	the	movie	hated	a	great	poor	acting	good	Class
1	1	1	1	1							+
2	1		1	1	1						-
3				2		1	1			1	+
4								1	1		-
5				1		1	1		1	1	+

Naïve Bayes Model

*p(+|D_{new})=p(+)*p(I|+)*p(hated|+)*p(the|+)*p(poor|+)*p(I|+)*p(acting|+)=?*

P(+)=0.6

Let n be the number of words in the (+) case:14.

$\mathrm{n_k}$ the number of times word k occurs in these cases

Let $p(W_k|+) = \frac{\mathrm{n_k}+1}{n+|Vocabulary|}$)

Compute: *p(I|+); p(loved|+); p(the|+); p(movie|+); p(a|+); p(great|+); p(acting|+); p(good|+);*

Naïve Bayes Model

D_{new}= "I hated the poor acting"

p(+|D_{new})=p(+)p(I|+)p(hated|+)p(the|+)p(poor|+)p(I|+)p(acting|+)=6.03$*10^{-7}$

p(-|D_{new})=p(-)p(I|-)p(hated|-)p(the|-)p(poor|-)p(I|-) p(acting|-)=1.22$*10^{-5}$

Naïve Bayes Model

- Very simple, easy to implement and fast.
- Need less training data.
- Can be used for both binary and mult-iclass classification problems.
- Can make probabilistic predictions.
- Handles continuous and discrete data.

Decision Tree

- Banks receive many loan applications that has to be processed for approval
- Each application consists of many inputs such as Age, Job status, Housing, Credit history, etc.
- Some applications are approved, others are not; some debtors default, others don’t
- Banks do not like defaulters – they want to approve only applicants who are not likely to default
- Their task is to predict if a new applicant will default or not.
- This a classification problem of **Approving** (non defaulter) or **Rejecting** (defaulter) an applicant.

Decision Tree

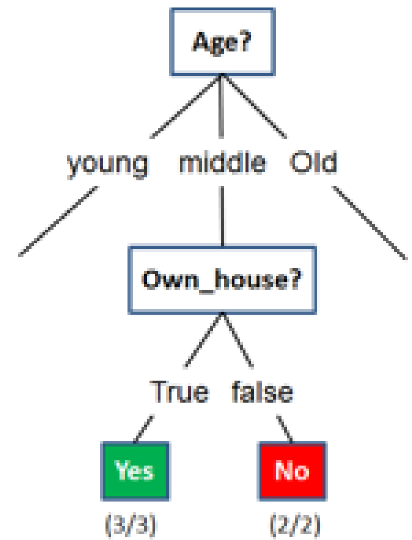
A Bank Loan Example:

ID	Age	Has_job	Own_house	Credit_rating	Outcome
1	young	False	False	fair	No
2	young	False	False	good	No
3	young	True	False	good	Yes
4	young	True	True	fair	Yes
5	young	False	False	fair	No
6	middle	False	False	fair	No
7	middle	False	False	good	No
8	middle	True	True	good	Yes
9	middle	False	True	excellent	Yes
10	middle	False	True	excellent	Yes
11	old	False	True	excellent	Yes
12	old	False	True	good	Yes
13	old	True	False	good	Yes
14	old	True	False	excellent	Yes
15	old	False	False	fair	No

Decision Tree

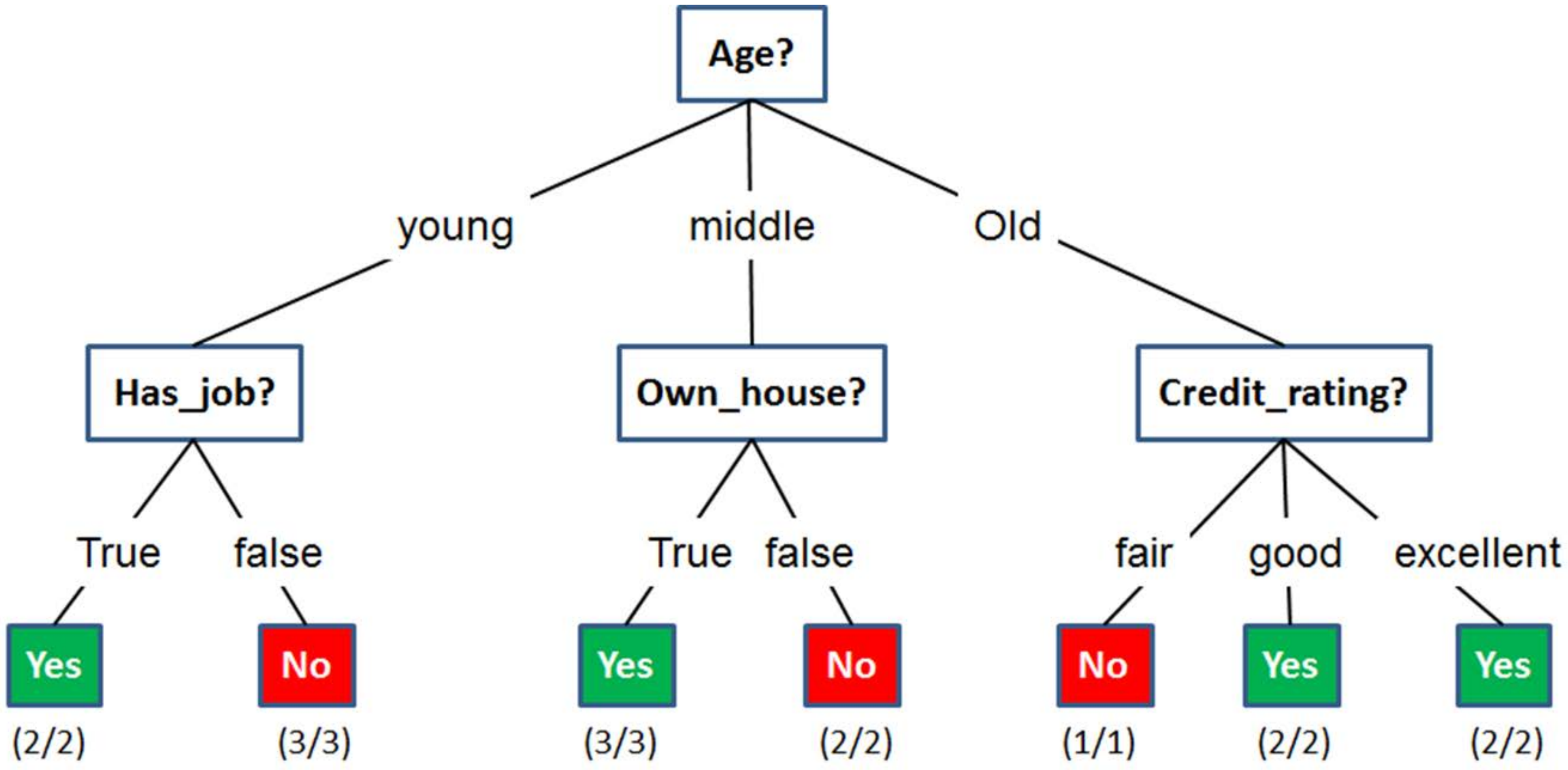
A Bank Loan Example:

ID	Age	Has_job	Own_house	Credit_rating	Outcome
1	young	False	False	fair	No
2	young	False	False	good	No
3	young	True	False	good	Yes
4	young	True	True	fair	Yes
5	young	False	False	fair	No
6	middle	False	False	fair	No
7	middle	False	False	good	No
8	middle	True	True	good	Yes
9	middle	False	True	excellent	Yes
10	middle	False	True	excellent	Yes
11	old	False	True	excellent	Yes
12	old	False	True	good	Yes
13	old	True	False	good	Yes
14	old	True	False	excellent	Yes
15	old	False	False	fair	No



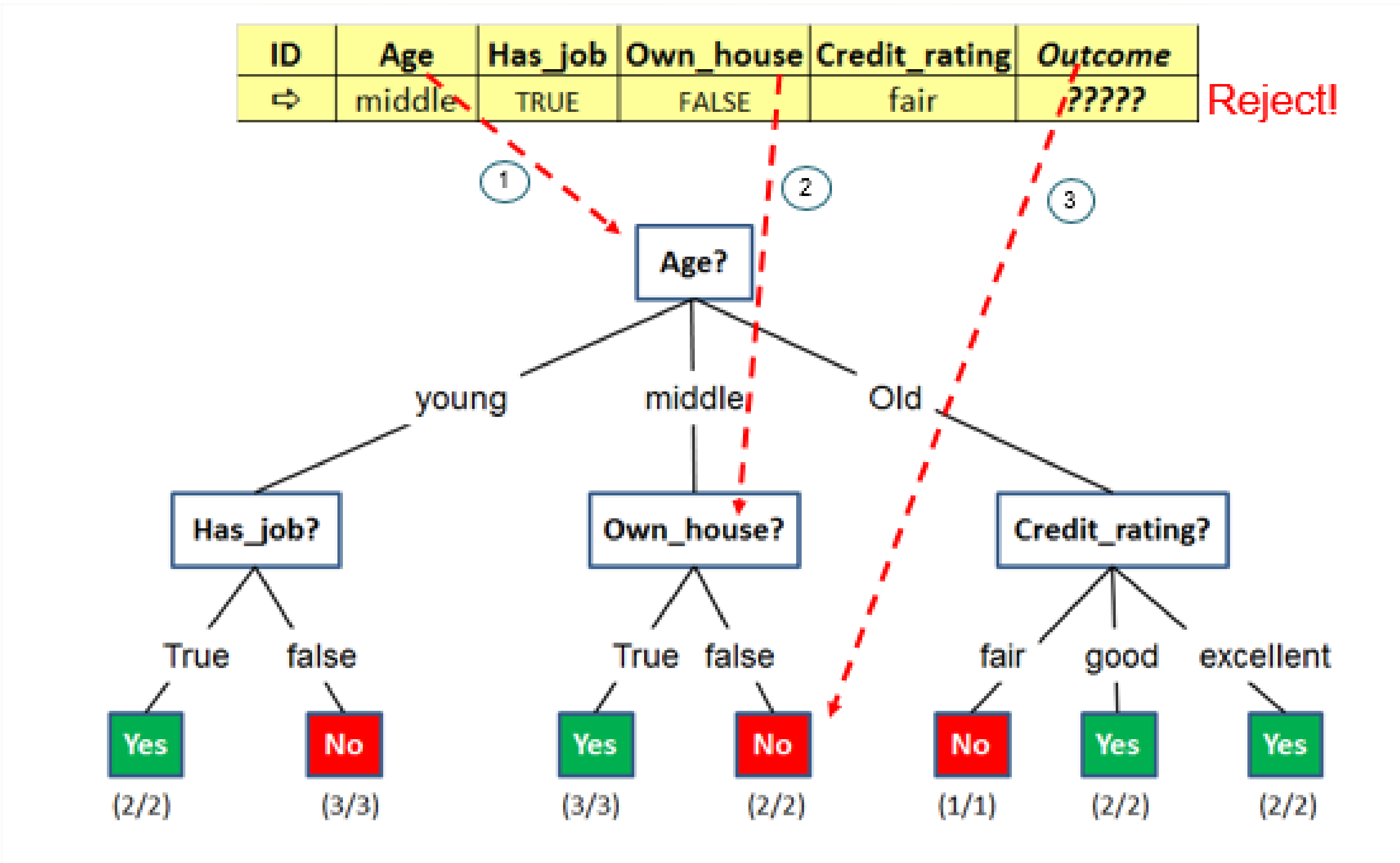
Decision Tree

One possible Decision Tree



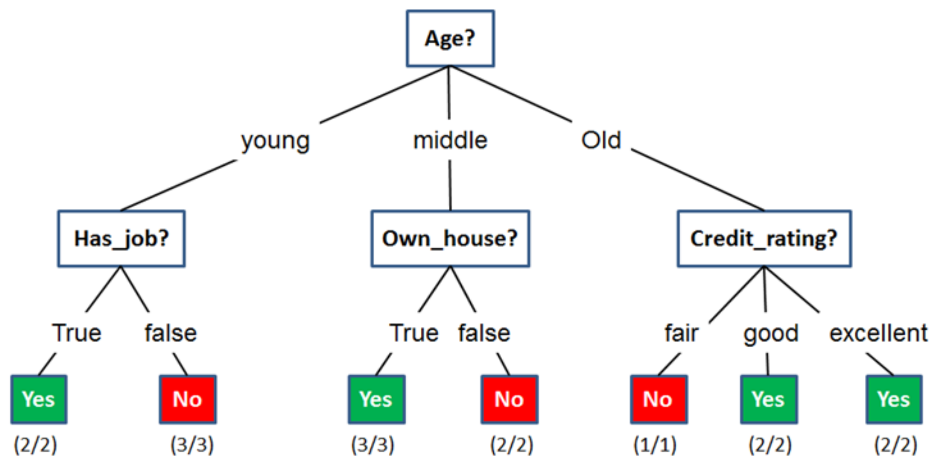
Decision Tree

Decision Trees are used to predict outcome



Decision Tree

- A decision tree consists of:
  - An **Decision node** which performs a test on an attribute
  - A **branch** that represents the outcome of the test
  - A **leaf** node that represents a class label



Decision Tree

How the Algorithm works:

Let $\mathbf{D_t}$ be the set of training records at a node $\boldsymbol{t}$

1. If $\mathbf{D_t}$ contains records that belong to the same class $\mathbf{y_t}$, then $\boldsymbol{t}$ is the leaf node labeled as $\mathbf{y_t}$
2. If $\mathbf{D_t}$ contains records that belong to more than one class, use an **attribute test** to split the data into smaller subsets. A child node is created for each outcome of the test condition and the records in $\mathbf{D_t}$ are distributed to the children based on the outcomes.
   Recursively apply the procedure to each subset.

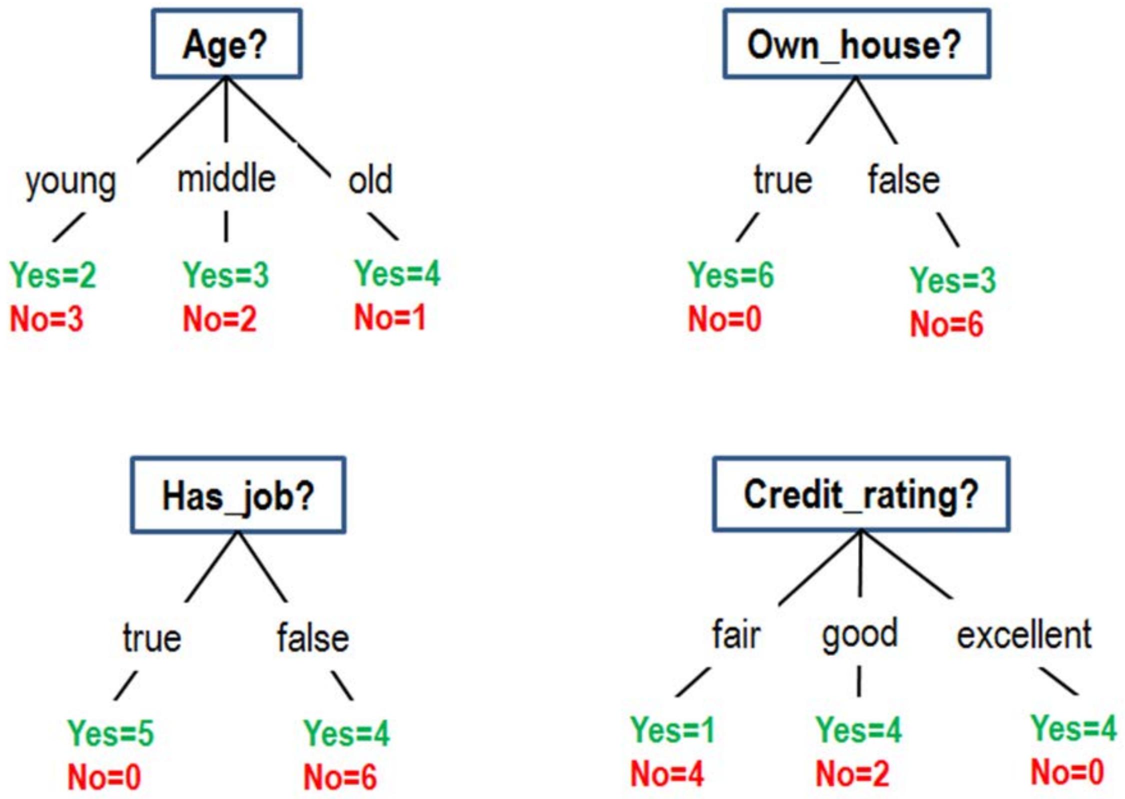
Decision Tree

How to determining the best split:

- The aim is to build and "optimal" decision tree.
- **Which attribute** (age? own_house? has_job?) should be split
- The best attribute is one that best separates the data into groups, where a **single class predominates** (homogeneity= purity) within each group

Decision Tree

Which Attribute do we choose?



Choose an attribute that gives the **best class purity**

Decision Tree

- **Information gain** measures the expected reduction in entropy (impurity)

- ***gain*(*D*, A_i)** = expected reduction in entropy of dataset ***D*** when attribute A_i is selected to branch (i.e. split) the data:

$$gain(D, A_i) = entropy(D) - entropy_{A_i}(D)$$

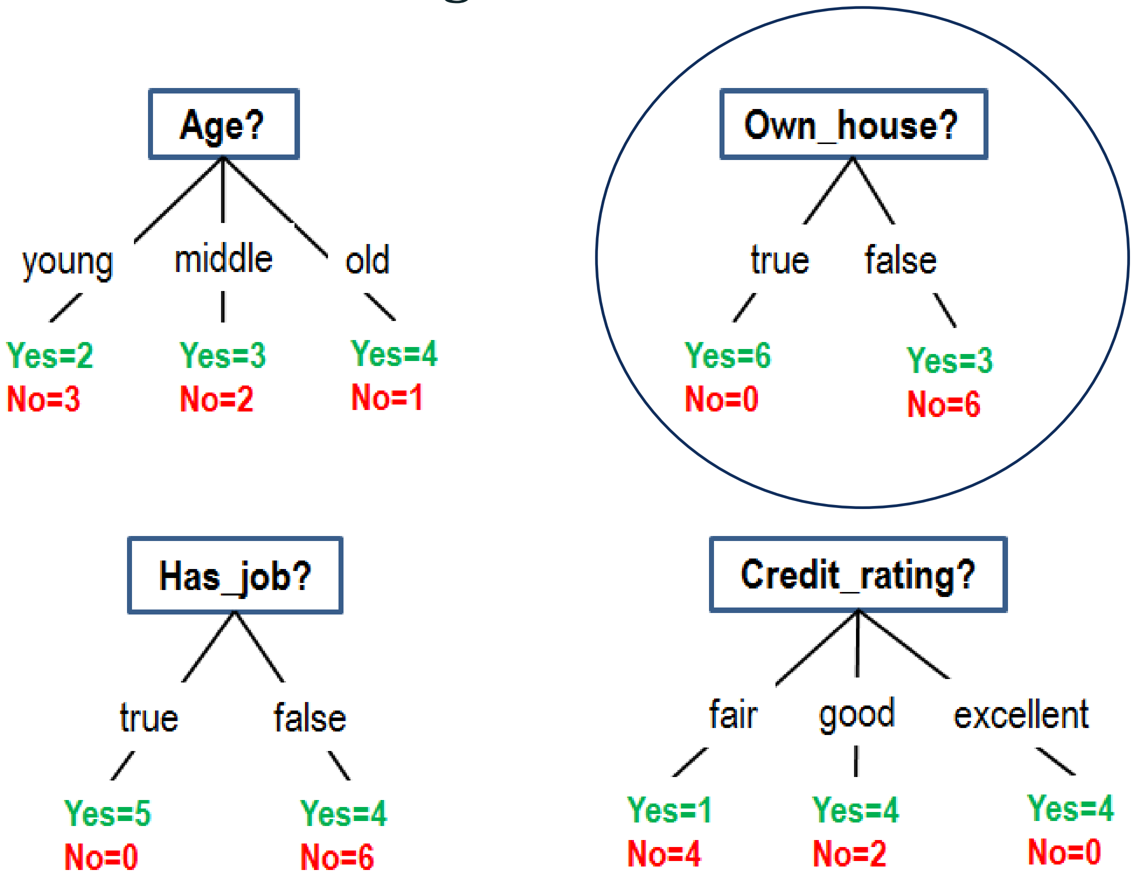
$$\boxed{entropy(D) = -\sum_{i=1}^{C} \Pr(C_i) \cdot \log_2 \Pr(C_i)}$$

Where C = number of classes

- We choose the attribute A_i with the ***highest gain*** to branch/split the current tree

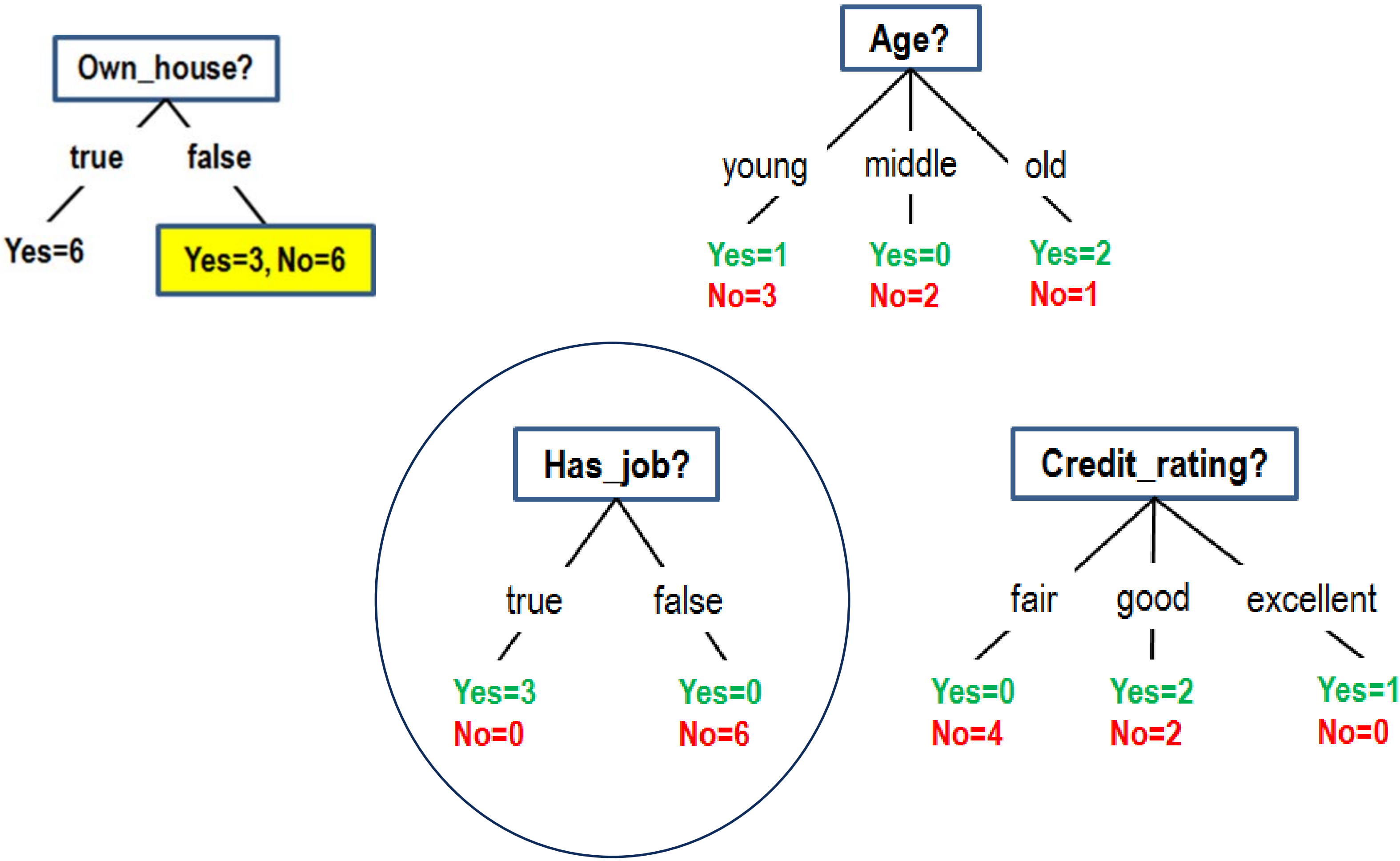
Decision Tree

Example of Tree building



Decision Tree

Re-calculate the purity:



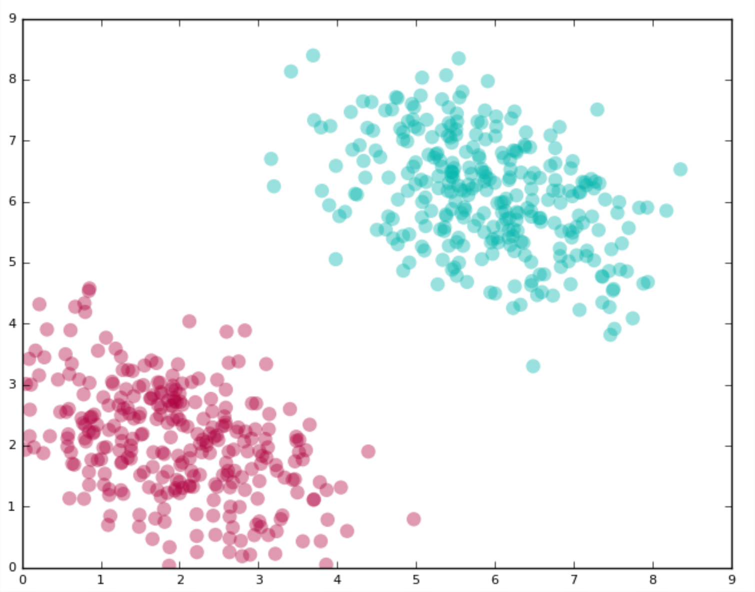
Decision Tree

- Requires simple data preparation
- Is a multi-variable analysis technique
- Able to handle both numerical and categorical data
- Trees can be visualized - simple to understand
- “white box” model - Situation that are observed in the model can be easily explained by boolean logic
- **Can easily overfit**

Support Vector Machine

We want to determine the relationship between Math and Programming scores and the performance in the TM course

The colour of the point represents how he did on the TM



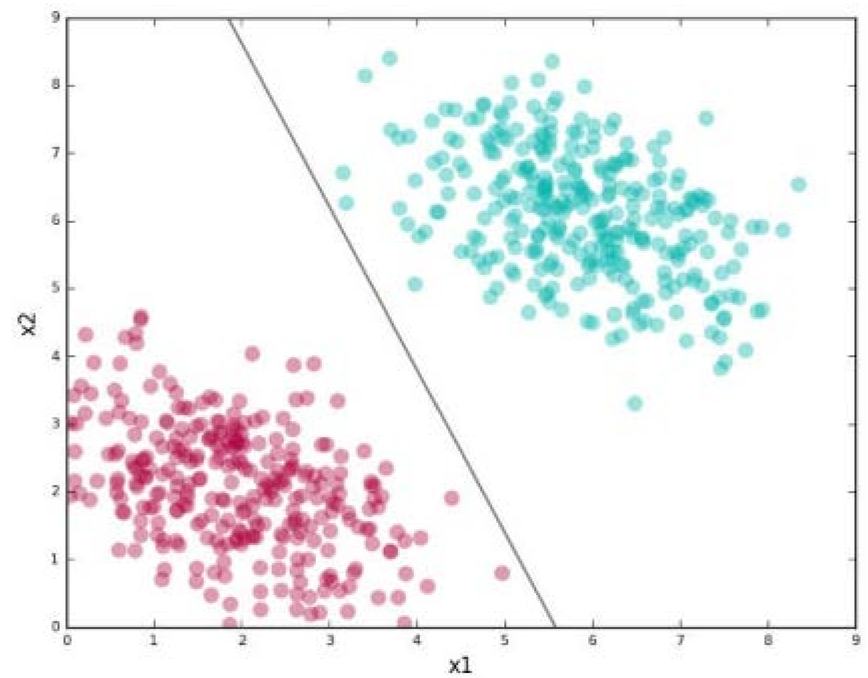
X1: Math Score
X2: Programming Score

Support Vector Machine

Finding a line that passes between the red and green clusters, then split the data into two part

We want each side of the data belongs to one class

The line here is our separating boundary (because it separates out the labels) or classifier (we use it classify points)..

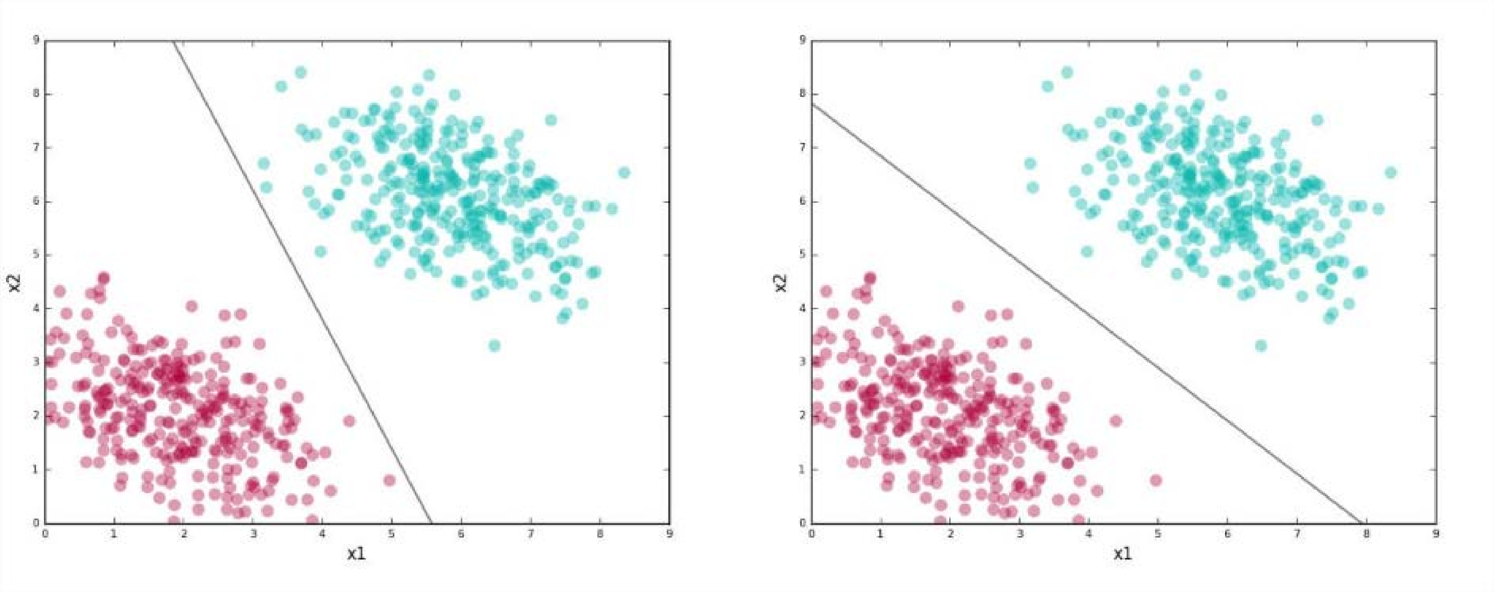


Support Vector Machine

We want to select the best classifier:

1. Find lines that correctly classify the training data

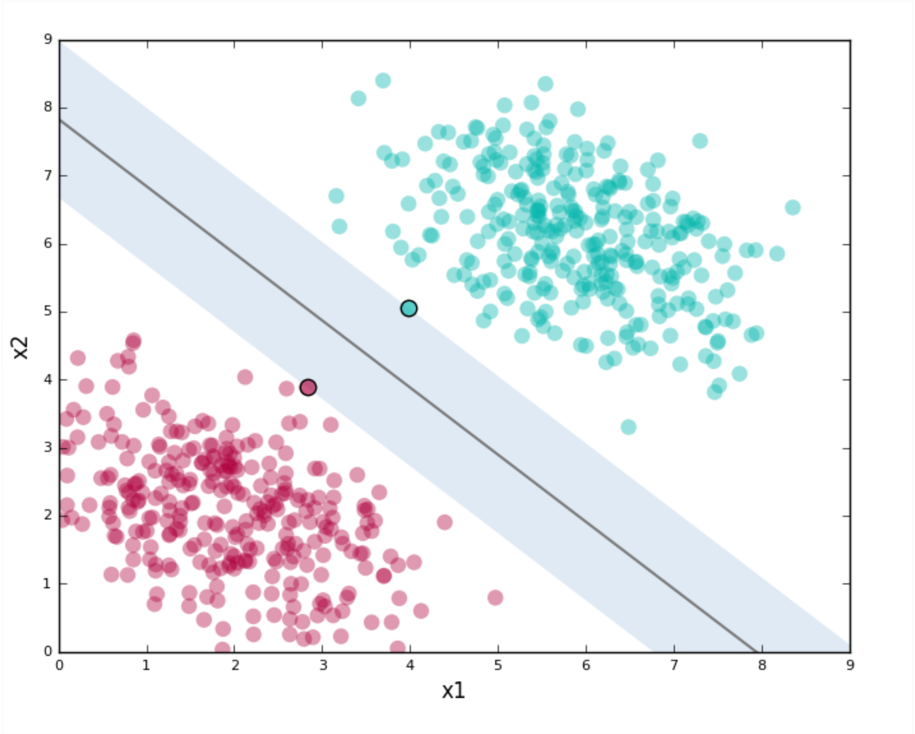
2. Among all such lines, pick the one that has the greatest distance to the points closest to it.



Support Vector Machine

Support vectors: points with black edges (there are two of them)
Margin (the shaded region)

Support Vector Machines give you a way to pick between many possible classifiers in a way that guarantees a higher chance of correctly labeling your test data.

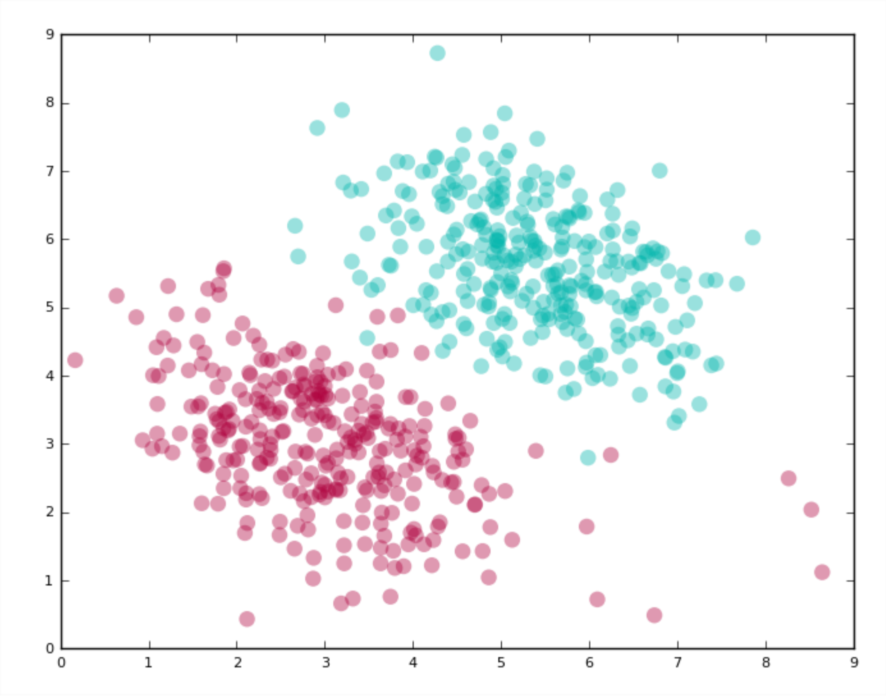


Support Vector Machine

Allowing for Errors:

Easy case that the data is perfectly linearly separable

Real-world data is, however, typically messy. You will almost always have a few instances that a linear classifier can't get right

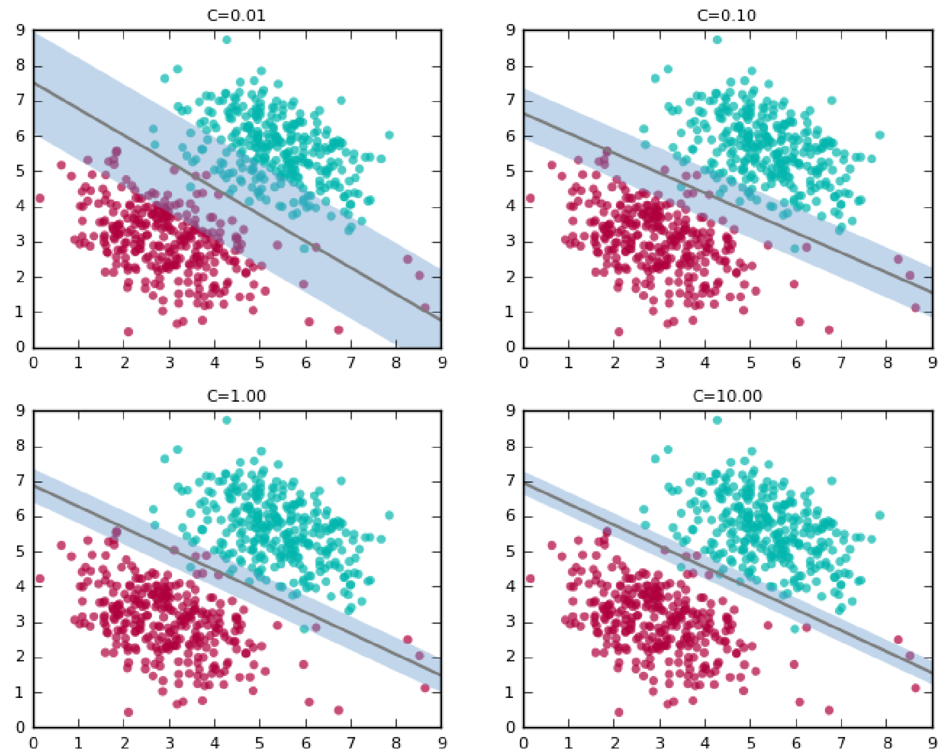


Support Vector Machine

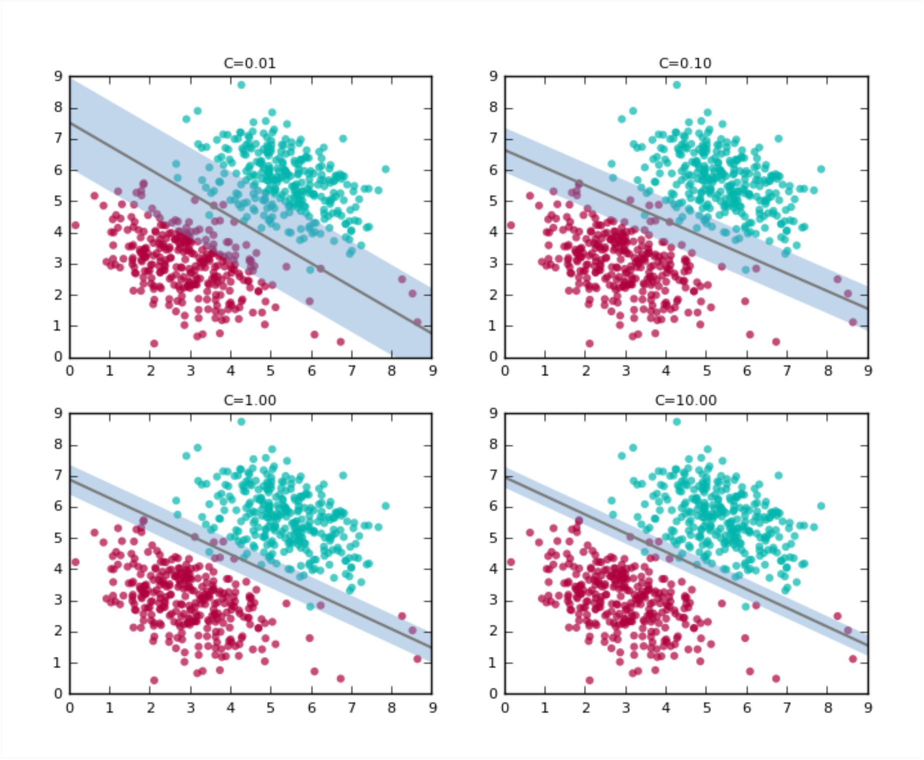
How do SVMs deal with this? They allow you to specify how many errors you are willing to accept.

You can provide a parameter called “C” to your SVM; this allows you to dictate the tradeoff between:

1. Having a wide margin.
2. Correctly classifying training data. A higher value of C implies you want lesser errors on the training data.



Support Vector Machine



The first plot with C=0.01 seems to capture the general trend better, although it suffers from a lower accuracy on the training data compared to higher values for C.

And since this is a trade-off, note how the width of the margin shrinks as we increase the value of C.

Support Vector Machine

Non-linearly Separable Data

How does it handle the cases where the data is absolutely not linearly separable?

Support Vector Machine

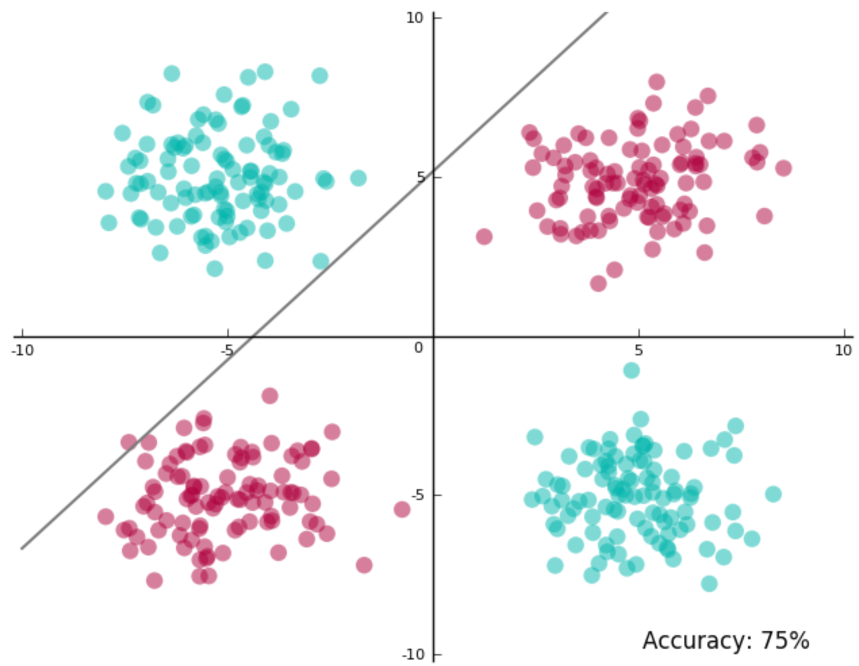
Non-linearly Separable Data

We start with the dataset in the above figure, and project it into a three-dimensional space where the new coordinates are:

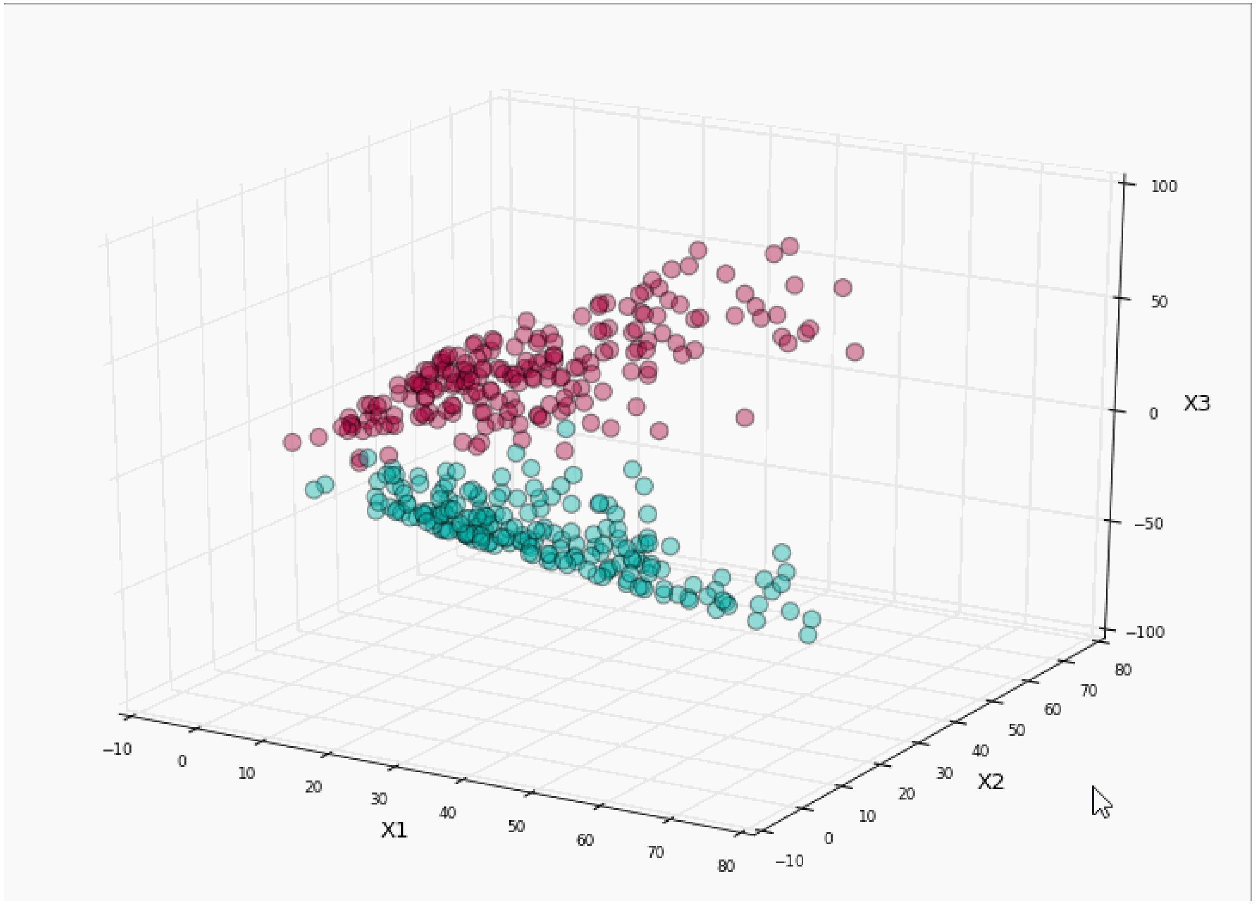
$$X_1 = x_1^2$$
$$X_2 = x_2^2$$
$$X_3 = \sqrt{2}x_1x_2$$

Our corresponding projected point was:

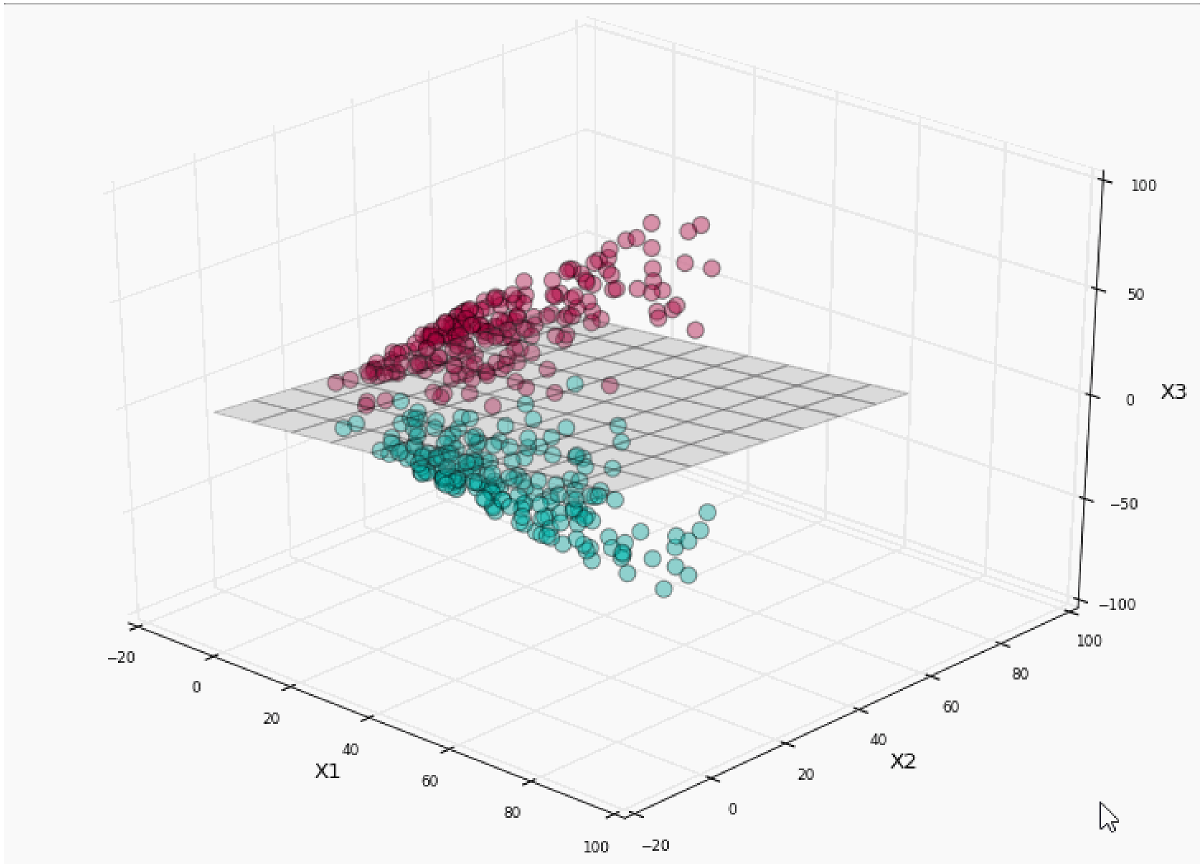
$$\vec{X_i} = (x_{i1}^2, x_{i2}^2, \sqrt{2}x_{i1}x_{i2})$$



Support Vector Machine



Support Vector Machine



Support Vector Machine

We have perfect label separation! Lets project the plane back to the original two-dimensional space and see what the separation boundary looks like:

Support Vector Machine

Kernel:

kernel functions enable us to operate in a high-dimensional, implicit feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between all pairs of data in low feature space.

$$
\begin{aligned}
K(\vec{x_i}, \vec{x_j}) &= (\vec{x_i} \cdot \vec{x_j})^2 \\
&= (x_{i1}x_{j1} + x_{i2}x_{j2})^2 \\
&= x_{i1}^2 x_{j1}^2 + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{i2}x_{j1}x_{j2} \\
&= (x_{i1}^2, x_{i2}^2, \sqrt{2}x_{i1}x_{i2}) \cdot (x_{j1}^2, x_{j2}^2, \sqrt{2}x_{j1}x_{j2})
\end{aligned}
$$

Polynomial kernel:

$$K(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j} + c)^d$$

Support Vector Machine

the same logic can be extended to other infinite-dimensional Kernels

A Gaussian Kernel is defined as,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

For simplicity, suppose $\sigma = 1$

$$\exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2}\right) = C\left\{1 - \underbrace{\frac{\langle \mathbf{x}_i, \mathbf{x}_j\rangle}{1!}}_{1st-order} + \underbrace{\frac{\langle \mathbf{x}_i, \mathbf{x}_j\rangle^2}{2!}}_{2nd-order} - \underbrace{\frac{\langle \mathbf{x}_i, \mathbf{x}_j\rangle^3}{3!}}_{3rd-order} + \ldots\right\}$$

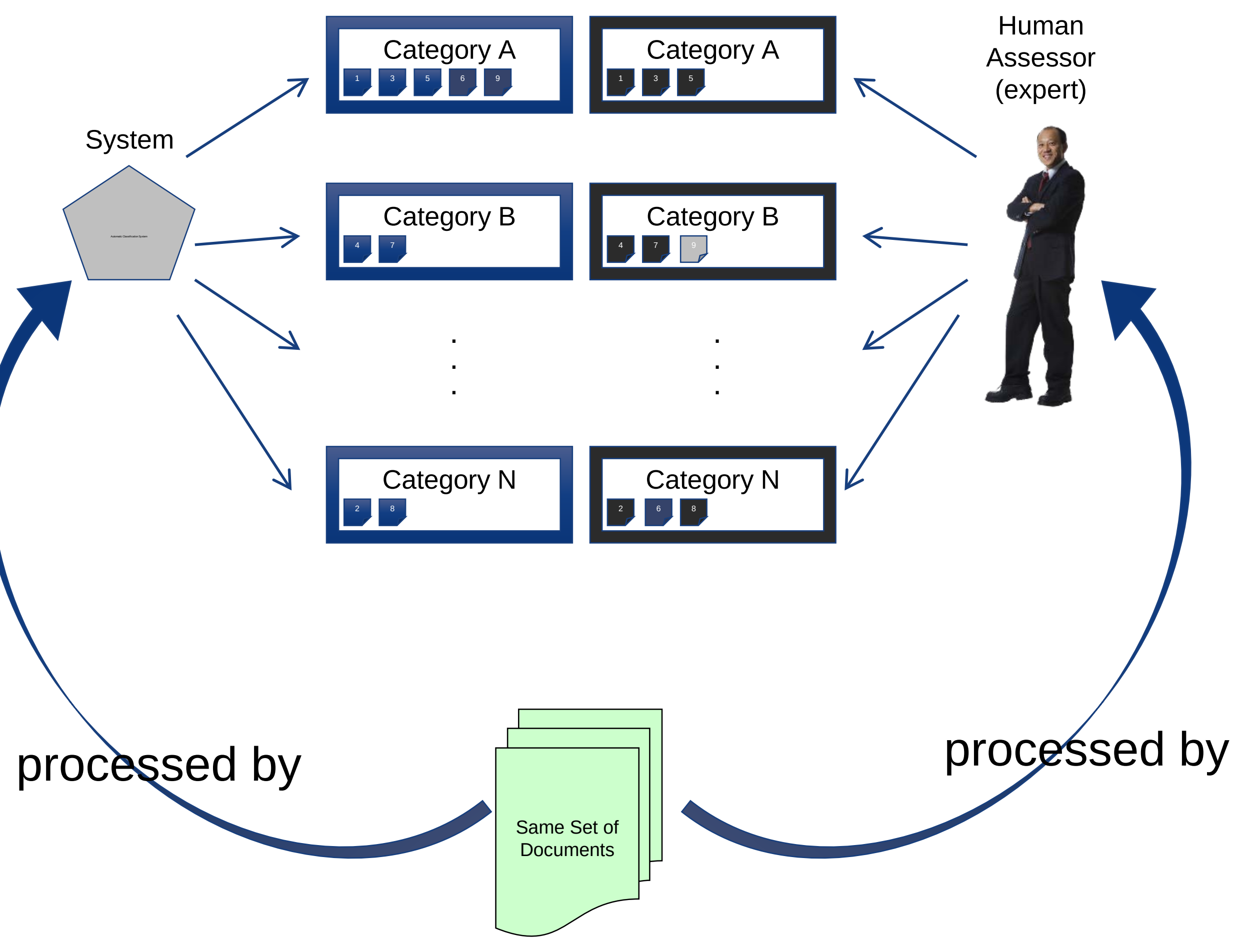
where, $C = \exp\left(-\frac{1}{2}\|\mathbf{x}_i\|^2\right)\exp\left(-\frac{1}{2}\|\mathbf{x}_j\|^2\right)$

EVALUATION

COMPARING DIFFERENT CLASSIFIERS FOR THE SAME CATEGORIZATION TASK

Remember this?

Predicted Categories Actual Categories



Confusion Matrix

		Predicted Categories					
		A	B	C		. . .	N
Actual Categories	A						
	B						
	C						
	⋮						
	N						

Example (using %)

Automatic Classification System

		Predicted Categories						
Actual Categories		A	B	C		. . .	N	
	A	87%	2%	5%		. . .	1%	= 100%
	B	6%	90%	0%		. . .	2%	= 100%
	C	12%	2%	77%		. . .	4%	= 100%
						. . .		
	:					. . .		
	N	21%	0%	4%		. . .	65%	= 100%

Example (using #)

Automatic Classification System

		Predicted Categories						
		A	B	C		. . .	N	
Actual Categories	A	143	34	17		. . .	2	= Tot(A) docs
	B	67	1289	44		. . .	239	= Tot(B) docs
	C	980	234	3454		. . .	88	= Tot(C) docs
						. . .		
	:					. . .		
	N	87	24	63		. . .	650	= Tot(N) docs

Consider the simple 2x2 matrix (2000 documents were classified)

Desired positive prediction

False negative

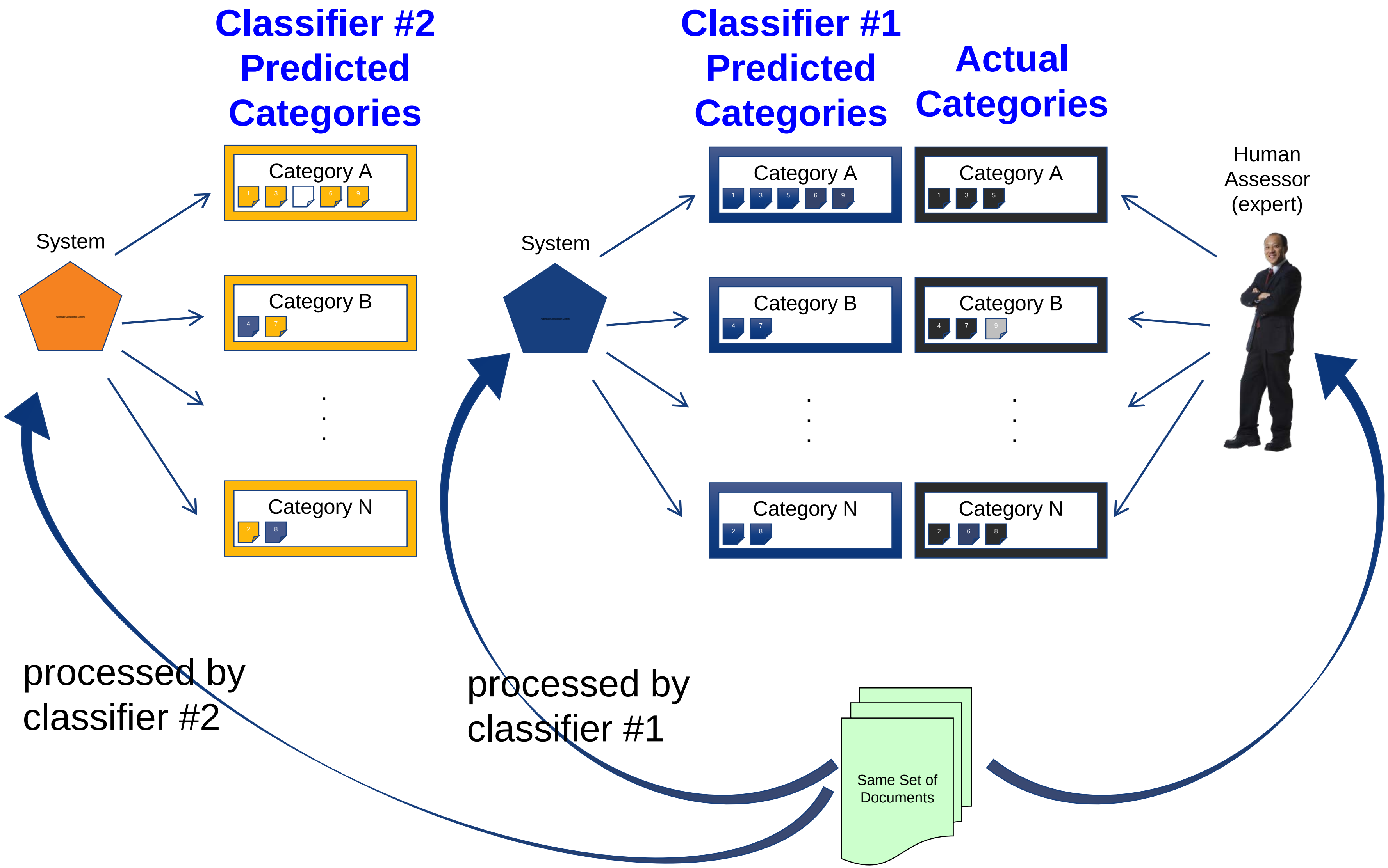
		Predicted	
		Yes	No
Actual	Yes	1350 90%	150 10%
	No	100 20%	400 80%

False positive

Desired negative prediction

EVALUATING MULTIPLE CLASSIFIERS

What happens with 2 classifiers?



Adding a cost function – fraud investigation

Classifier #1

Insurance Claim Statements

Classifier #2

Classifier #1

Actual \ Predicted	Genuine	Fraud
Genuine	900	100
Fraud	40	410

Company loses $80k in fraud
Company pays $255k in costs

Which classifier is better?

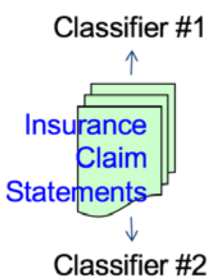
Classifier #2

Actual \ Predicted	Genuine	Fraud
Genuine	700	300
Fraud	2	448

Company loses $4k in fraud
Company pays $374k in costs

The average fraud costs the company $2000
It costs the company $500 to investigate each suspected fraud

Adding a cost function – fraud investigation



Classifier #1

		Predicted	
		Genuine	Fraud
Actual	Genuine	900	100
	Fraud	40	410

Classifier #2

		Predicted	
		Genuine	Fraud
Actual	Genuine	700	300
	Fraud	2	448

The average fraud costs the company $2000
It costs the company $500 to investigate each su

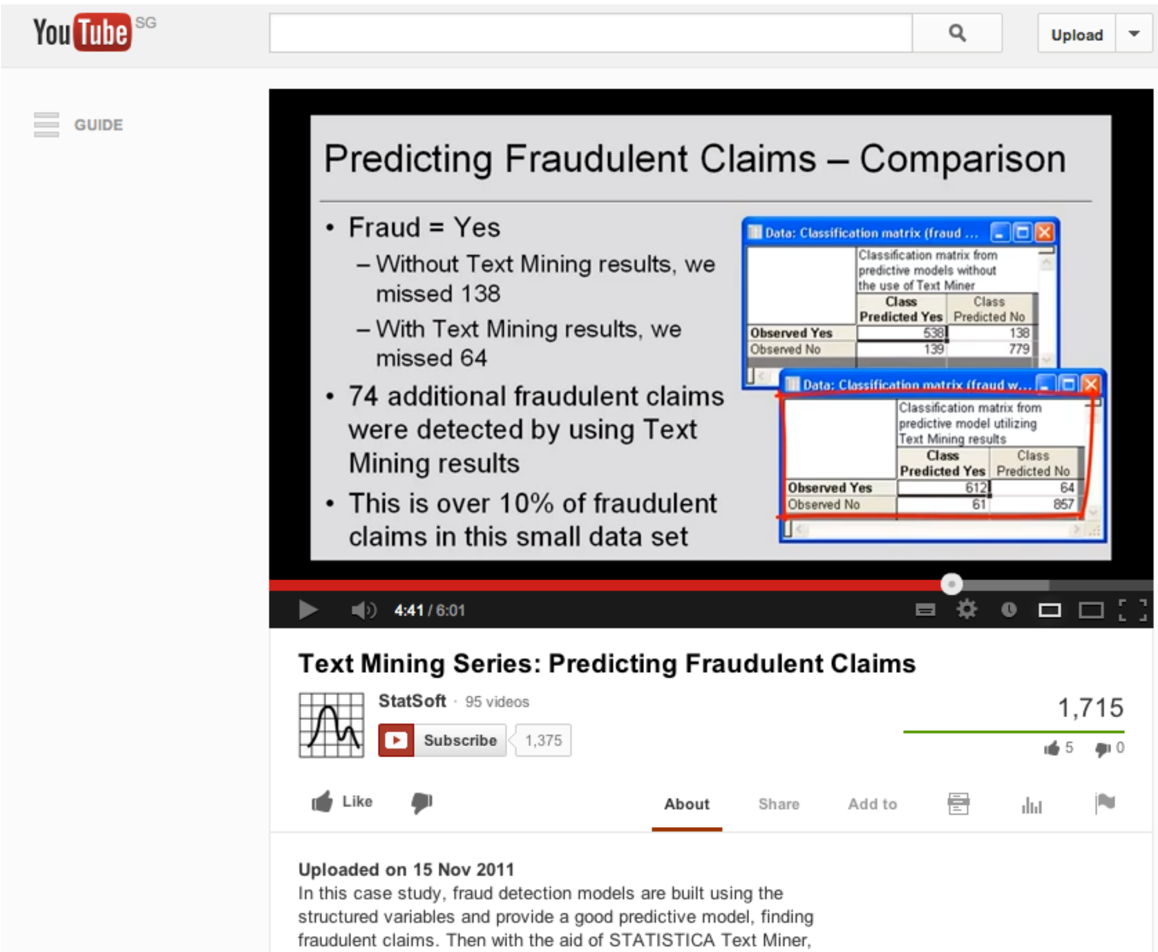
- ***Consider Doing nothing (don't act to identify fraud):***
  - *Predicted fraud = 0 cases @$500 per case costs $0k for investigation.*
  - *Undetected fraud is 450 cases @$2k/fraud loses $900k.*
  - *Overall -$0k -$900k = -$900k*
- ***Analysis for classifier #1:***
  - *Predicted fraud = 510 cases @$500 per case costs $255k for investigation.*
  - *Undetected fraud is 40 cases @$2k/fraud loses $80k.*
  - *Overall -$255k -$80k = -$335k*
- ***Analysis for classifier #2:***
  - *Predicted fraud = 748 cases @$500 per case costs $374k for investigation.*
  - *Undetected fraud is 2 cases @$2k/fraud loses $4k.*
  - *Overall -$374k -$4k = -$378k*

Classifier evaluation

- **Evaluation of classifiers is done with respect to a *business context***
- **Evaluation of classifiers is normally done *empirically***
- **Experimental evaluation focuses on *effectiveness*, i.e., the ability of the classifier to make the right classification decision**
- **Precision & Recall concepts as applied to (multi-class) categorization**
  - Precision is the probability that if a random document d_i is categorized under category c_i, that decision is correct
  - Recall wrt c_i is the probability that if a random document d_i should be categorized under c_i , then the decision is taken

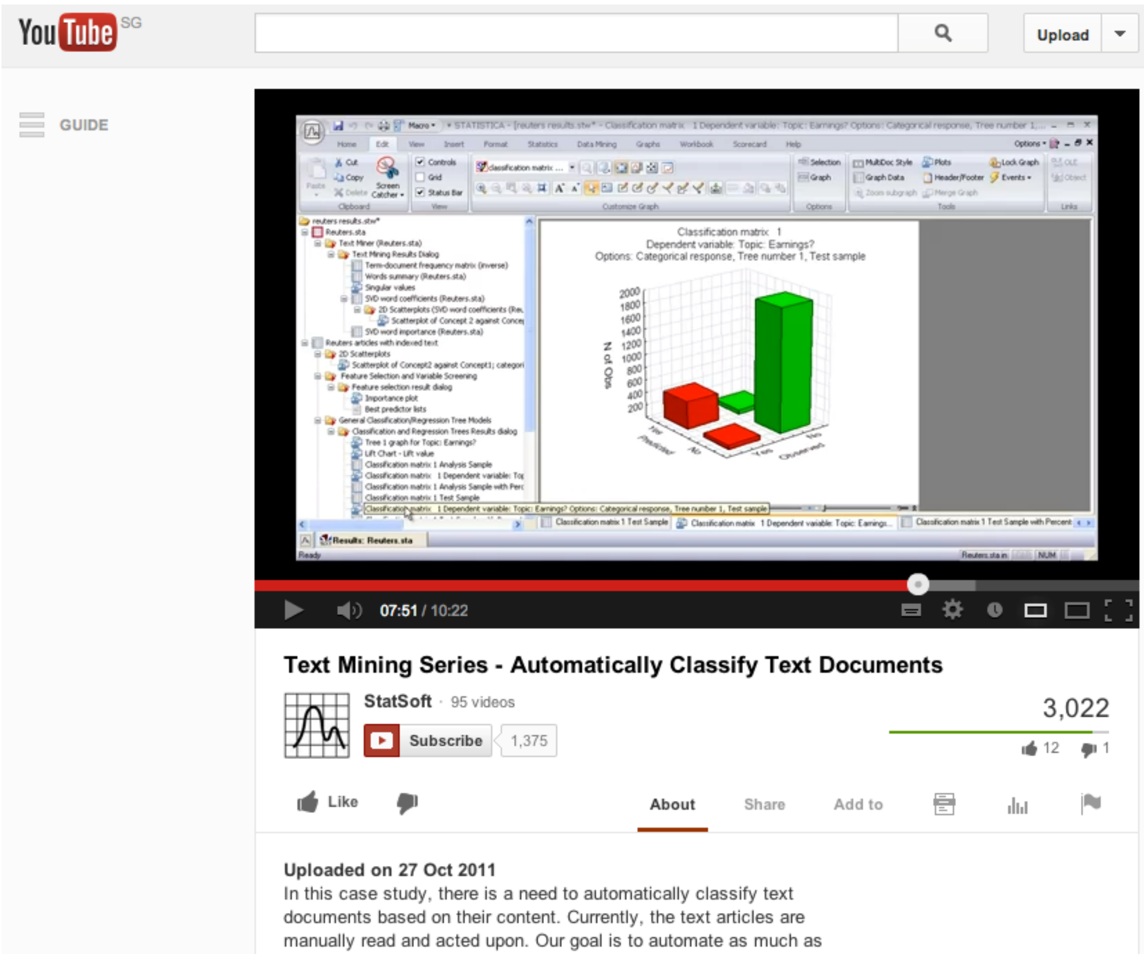
TEXT CATEGORIZATION APPLICATION EXAMPLES

Boosting Identification of Fraudulent Claims



From: http://www.youtube.com/watch?v=OlQpm8qTog4

Automatic Categorization of Documents



From: http://www.youtube.com/watch?v=Q5K3gyQJkC0

Reference & Resources

- Fabrizio Sebastiani, *A Tutorial on Automated Text Categorization*, web.iiit.ac.in/~jawahar/PRA-03/textCat.pdf
- F. Aiolli, *Text Categorization*, downloaded from http://www.math.unipd.it/~aiolli/corsi/SI-0607/Lez09.251006.pdf
- John Elder, Gary Miner, Bob Nisbet. *Practical Text Mining and Statistical Analysis for non-Structured Text Data Applications*, Academic Press, 2012
- Chris Manning & Hinrich Schutze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999
- Scott Weingart, *Topic Modeling for Humanists: A Guided Tour*, downloaded from http://www.scottbot.net/HIAL/index.html@p=19113.html
- Ted Underwood, *Topic Modeling made just simple enough*, downloaded from https://tedunderwood.com/2012/04/07/topic-modeling-made-just-simple-enough/
- NLP resources: http://nlp.stanford.edu/links/statnlp.html