Introduction to Data Science

CS 5665
Utah State University
Department of Computer Science
Instructor: Prof. Kyumin Lee

In the news…

Online news has made finding current events easier than ever — but what if it could be used to predict future events as well? Eric Horvitz of Microsoft Research and Kira Radinsky of the Technion-Israel Institute of Technology have built a system that mines over 20 years of *New York Times* articles for events that could point to other, later developments. In their test model, Horvitz and Radinsky created a system to draw connections between events, then set it loose on the *Times* database.

The system was able to "learn" correlations between events by looking at sequences of stories in particular places: if an article was published about a drought in one place, for example, there was an 18 percent chance of a drought being reported there later. And both droughts and storms can lead to cholera outbreaks. Not every event (like, say, "cholera outbreak in Rwanda") had enough data to be useful, so it also needed to be able to find and connect patterns from different kinds of events that shared characteristics.

With enough data and a good model, the system could be used to give early warning signs for disasters by mining individual reports and finding larger patterns in them. The general idea of finding ways to predict diseases isn't new, nor is the concept of data mining for prediction, but the wide scope of this project makes it potentially very useful — as long as the system is able to successfully draw correlations between events, and to generalize enough to make them useful, it could be applied to any number of situations.

VIA GIGAOM TECHNOLOGY REVIEW
SOURCE MICROSOFT RESEARCH
RELATED ITEMS PREDICTION DATA MINING DISEASE NEW YORK TIMES MICROSOFT

Mining the Web to Predict Future Events

Kira Radinsky
Technion–Israel Institute of Technology
Haifa, Israel
kirar@cs.technion.ac.il

Eric Horvitz
Microsoft Research
Redmond, WA, USA
horvitz@microsoft.com

ABSTRACT

We describe and evaluate methods for learning to forecast forthcoming events of interest from a corpus containing 22 years of news stories. We consider the examples of identifying significant increases in the likelihood of disease outbreaks, deaths, and riots in advance of the occurrence of these events in the world. We provide details of methods and studies, including the automated extraction and generalization of sequences of events from news corpora and multiple web resources. We evaluate the predictive power of the approach on real-world events withheld from the system.

the LinkedData platform [6]. The goal is to build predictive models that generalize from specific sets of sequences of events to provide likelihoods of future outcomes, based on patterns of evidence observed in near-term newsfeeds. We propose the methods as a means of generating actionable forecasts in advance of the occurrence of target events in the world.

The methods we describe operate on newsfeeds and can provide large numbers of predictions. We demonstrate the predictive power of mining thousands of news stories to create classifiers for a range of prediction problems. We show as examples forecasts on three prediction challenges: proac-

http://research.microsoft.com/en-us/um/people/horvitz/future_news_wsdm.pdf

Cause	Effect	Probability
Drought	Flood	18%
Flood	cholera	1%
Flood in Rwanda	cholera in Rwanda	67%
Flood in Lima	cholera in Lima	33%
Flood in Country with water coverage $> 5\%$	cholera in Country	14%
Flood in Country with water coverage $> 5\%$, population density > 100	cholera in Country	16%

Table 6: Probability transitions for several examples.

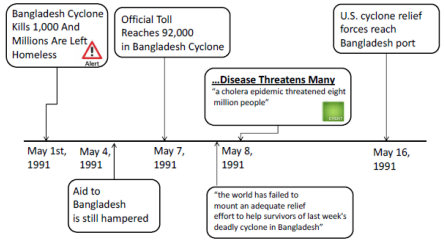


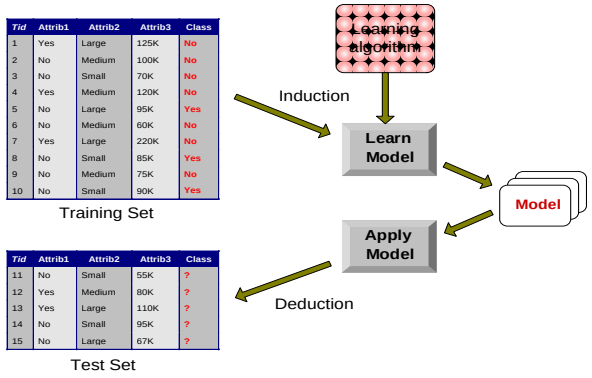
Figure 6: Example of cholera alert following storms in Bangladesh. Triangular alert icons represent inferences of significant upswings in likelihood of forthcoming cholera outbreak.

Practicum

- This Thursday
  - Mallet (Vaibhav)
  - Weka (Akhil & Kamna)
  - LingPipe (Bhagyashree & Sreevidya)

Mining and Analytics: Classification + Decision Trees

Illustrating Classification Task



General Structure of Hunt's Algorithm

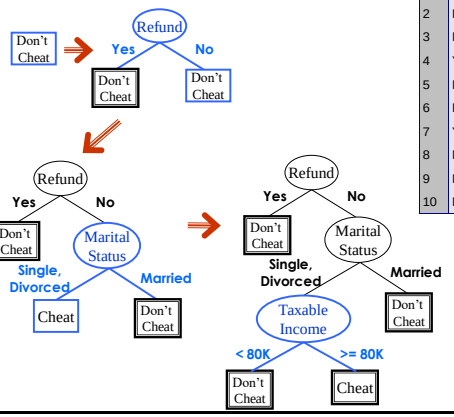
1. Examine the record data and find the best attribute for the first node.
2. Split the record data based on this attribute
3. Recurse on each corresponding child node choosing other attributes

- ***[Recursively apply]*** Let D_t be the set of training records that are associated with node t and $y = \{y_1, y_2, \ldots y_c\}$ be the set of class labels
  - If D_t contains records that belong the same class y_t, then its decision tree consists of a leaf node labeled as y_t
  - If D_t is an empty set, then its decision tree is a leaf node whose class label is determined from other information such as the majority class of the records
  - If D_t contains records that belong to several classes, then a **test condition** based on one of the attributes of D_t is applied to split the data into more homogenous subsets

Example

- Attributes:
  - Refund (Yes, No)
  - Martial Status (Single, Divorced, Married)
  - Taxable Income (quantitative)
- Class:
  - Cheat, Don't Cheat

Hunt's Algorithm



Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Tree Induction

- Determine how to split the records
  - Use greedy heuristics to make a series of locally optimum decision about which attribute to use for partitioning the data
  - At each step of the greedy algorithm, a test condition is applied to split the data in to subsets with a more homogenous class distribution
    - How to specify test condition for each attribute
    - How to determine the best split
- Determine when to stop splitting
  - A stopping condition is needed to terminate tree growing process. Stop expanding a node
    - if all the instances belong to the same class
    - if all the instances have similar attribute values

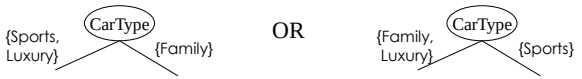
Splitting?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

For Nominal Attributes

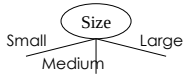
- Multi-way split: Use as many partitions as distinct values.

- Binary split: Divides values into two subsets. Need to find optimal partitioning.

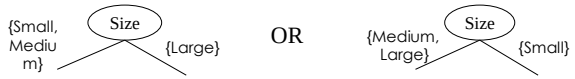


For Ordinal Attributes

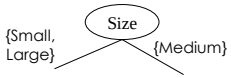
- Multi-way split: Use as many partitions as distinct values.



- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.

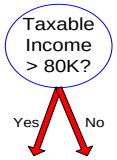


- What about this split?

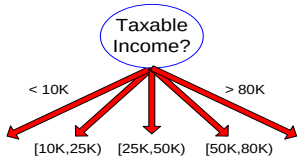


For Quantitative

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Binary Decision: (A < v) or (A ≥ v)
    - consider all possible splits and finds the best cut
    - can be more compute intensive



(i) Binary split

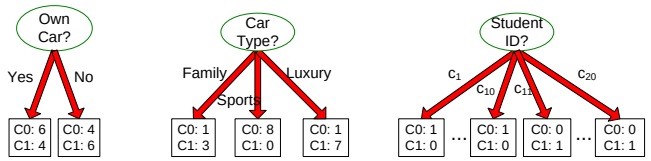


(ii) Multi-way split

But How do we split?

- Splitting Criterion
  - Given the data associated with a particular node in the tree, what **test condition** do we apply?

Before Splitting: 10 records of class 0, 10 records of class 1



Which test condition is the best?

Splitting Criterion

- Ideas?
- Intuition: Prefer nodes with *homogeneous* class distribution

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, **High degree of impurity**	**Homogeneous,** **Low degree of impurity**

- Typical methods (i.e., measuring impurity)
  - Gini Index
  - Entropy / Information Gain
  - Classification error

Splitting Criterion: GINI

- Gini Index for a given node t :

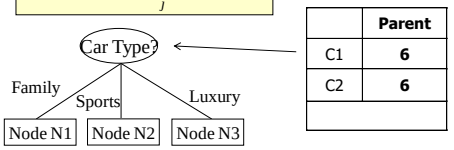
$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t).

- Measure the impurity of a node
  - Maximum (1 - $1/n_c$) when records are equally distributed among all classes, implying least interesting information
  - Minimum (0.0) when all records belong to one class, implying most interesting information

GINI Example

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$



	Parent
C1	6
C2	6

Node N1

C1	0
C2	6

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Gini = 1 – P(C1)² – P(C2)² = 1 – 0 – 1 = 0

Node N2

C1	1
C2	5

P(C1) = 1/6 P(C2) = 5/6

Gini = 1 – (1/6)² – (5/6)² = 0.278

Node N3

C1	2
C2	4

P(C1) = 2/6 P(C2) = 4/6

Gini = 1 – (2/6)² – (4/6)² = 0.444

Splitting Based on GINI

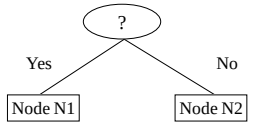
- Used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as,

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, n_i = number of records at child i,
n = number of records at node p.

GINI for Binary Attributes

- Splits into two partitions
- Effect of Weighing partitions:
  - Larger and Purer Partitions are sought for.



	Parent
C1	6
C2	6
Gini = 0.500	

Gini(N1)
= 1 – $(5/7)^2$ – $(2/7)^2$
= 0.408

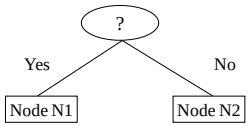
Gini(N2)
= 1 – $(1/5)^2$ – $(4/5)^2$
= 0.320

	N1	N2
C1	5	1
C2	2	4
Gini=?		

Gini(Children)
= 7/12 * 0.408 +
5/12 * 0.320
= 0.371

GINI for Binary Attributes

- Splits into two partitions
- Effect of Weighing partitions:
  - Larger and Purer Partitions are sought for.



	Parent
C1	6
C2	6
Gini = 0.500	

Gini(N1)
= 1 – $(5/7)^2$ – $(2/7)^2$
= 0.408

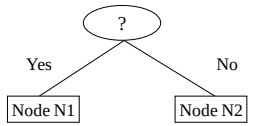
Gini(N2)
= 1 – $(1/5)^2$ – $(4/5)^2$
= 0.320

	N1	N2
C1	5	1
C2	2	4
Gini=0.371		

Gini(Children)
= 7/12 * 0.408 +
5/12 * 0.320
= 0.371

GINI for Binary Attributes

- Splits into two partitions
- Effect of Weighing partitions:
  - Larger and Purer Partitions are sought for.



	Parent
C1	6
C2	6
Gini = 0.500	

Attribute A	N1	N2
C1	0	6
C2	6	0
Gini=0.000		

Attribute B	N1	N2
C1	5	1
C2	1	5
Gini=0.278		

Attribute C	N1	N2
C1	4	2
C2	3	3
Gini=0.486		

Attribute D	N1	N2
C1	3	3
C2	3	3
Gini=0.500		

GINI for Nominal Attributes

- For each distinct value, gather counts for each class in the dataset
- Use the count matrix to make decisions

Multi-way split

	CarType		
	Family	Sports	Luxury
C1	1	2	1
C2	4	1	1
Gini	0.393		

Two-way split
(find best partition of values)

	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	CarType	
	{Sports}	{Family, Luxury}
C1	2	2
C2	1	5
Gini	0.419	

GINI for Quantitative Attributes

- Use Binary Decisions based on one value
- Several Choices for the splitting value
  - Number of possible splitting values = Number of distinct values
- Each splitting value has a count matrix associated with it
  - Class counts in each of the partitions, A < v and A ≥ v
- Simple method to choose best v
  - For each v, scan the database to gather count matrix and compute its Gini index
  - Computationally Inefficient! Repetition of work. $O(n)^2$

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

GINI for Quantitative Attributes

- For efficient computation: for each attribute,
  - Sort the attribute on values
  - Linearly scan these values, each time updating the count matrix and computing gini index
  - Choose the split position that has the least gini index
  - O(nlogn)

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No
Taxable Income										
Sorted Values →	60	70	75	85	90	95	100	120	125	220
Split Positions →	55	65	72	80	87	92	97	110	122	172	230
	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >
Yes	0 3	0 3	0 3	0 3	1 2	2 1	3 0	3 0	3 0	3 0	3 0
No	0 7	1 6	2 5	3 4	3 4	3 4	3 4	4 3	5 2	6 1	7 0
Gini	0.420	0.400	0.375	0.343	0.417	0.400	*0.300*	0.343	0.375	0.400	0.420

Other splitting criteria

- Information Gain
- Classification Error

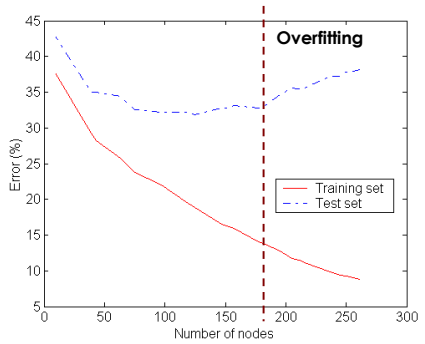
(See readings)

When do we stop splitting?

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values
- Early termination

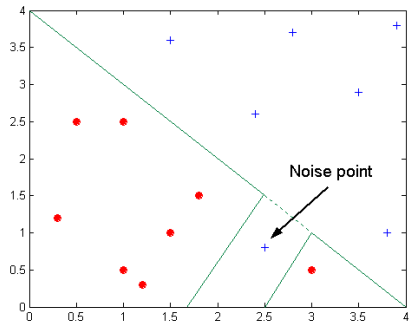
Issues with Classification

Underfitting and Overfitting



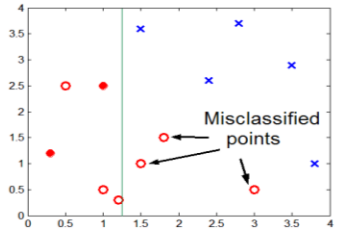
Underfitting: when model is too simple, both training and test errors are large

Overfitting due to Noise



Decision boundary is distorted by noise point

Overfitting due to Insufficient Examples



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records
- Need new ways for estimating errors

Evaluating a Classifier

Accuracy

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$