DATA MINING LECTURE 11

Classification

Naïve Bayes

Supervised Learning

Graphs And Centrality

NAÏVE BAYES CLASSIFIER

Bayes Classifier

- A probabilistic framework for solving classification problems
- **A, C** random variables
- Joint probability: **Pr(A=a,C=c)**
- Conditional probability: **Pr(C=c | A=a)**
- Relationship between joint and conditional probability distributions

$$\Pr(C,A) = \Pr(C \mid A) \times \Pr(A) = \Pr(A \mid C) \times \Pr(C)$$

- **Bayes Theorem**:

$$P(C \mid A) = \frac{P(A \mid C)P(C)}{P(A)}$$

Bayesian Classifiers

- How to classify the new record X = ('Yes', 'Single', 80K)

Tid	**Refund**	**Marital Status**	**Taxable Income**	**Evade**
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**

Find the class with the highest probability given the vector values.

Maximum Aposteriori Probability estimate:

- Find the value c for class C that maximizes P(C=c| X)

How do we estimate P(C|X) for the different values of C?

- We want to estimate P(C=Yes| X)
- and P(C=No| X)

Bayesian Classifiers

- In order for probabilities to be well defined:
  - Consider each attribute and the class label as random variables
  - Probabilities are determined from the data

Tid	**Refund**	**Marital Status**	**Taxable Income**	**Evade**
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**

Evade C
Event space: {Yes, No}
$P(C) = (0.3, 0.7)$

Refund A_1
Event space: {Yes, No}
$P(A_1) = (0.3,0.7)$

Martial Status A_2
Event space: {Single, Married, Divorced}
$P(A_2) = (0.4,0.4,0.2)$

Taxable Income A_3
Event space: R
$P(A_3) \sim Normal(\mu,\sigma^2)$
$\mu = 104$:sample mean, $\sigma^2=1874$:sample var

Bayesian Classifiers

- Approach:
  - compute the posterior probability $P(C \mid A_1, A_2, \ldots, A_n)$ using the Bayes theorem

$$\boxed{P(C \mid A_1 A_2 \ldots A_n) = \frac{P(A_1 A_2 \ldots A_n \mid C) P(C)}{P(A_1 A_2 \ldots A_n)}}$$

  - Maximizing

    $P(C \mid A_1, A_2, \ldots, A_n)$

    is equivalent to maximizing

    $P(A_1, A_2, \ldots, A_n \mid C)\, P(C)$
    - The value $P(A_1, \ldots, A_n)$ is the same for all values of C.

- How to estimate $P(A_1, A_2, \ldots, A_n \mid C)$?

Naïve Bayes Classifier

- Assume conditional independence among attributes A_i when class C is given:
  - $P(A_1, A_2, \ldots, A_n | C) = P(A_1|C)\, P(A_2|C) \cdots P(A_n|C)$
  - We can estimate $P(A_i | C)$ from the data.
  - New point $X = (A_1 = \alpha_1, \ldots A_n = \alpha_n)$ is classified to class c if

$$P(C = c|X) = P(C = c) \prod_i P(A_i = \alpha_i|c)$$

is maximum over all possible values of C.

Example

- Record

  X = (Refund = Yes, Status = Single, Income =80K)
- For the class C = 'Evade', we want to compute:

  P(C = Yes|X) and P(C = No| X)
- We compute:
  - P(C = Yes|X) = P(C = Yes)*P(Refund = Yes |C = Yes)

    *P(Status = Single |C = Yes)

    *P(Income =80K |C= Yes)
  - P(C = No|X) = P(C = No)*P(Refund = Yes |C = No)

    *P(Status = Single |C = No)

    *P(Income =80K |C= No)

How to Estimate Probabilities from Data?

Tid	**Refund**	**Marital Status**	**Taxable Income**	**Evade**
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**

Class Prior Probability:

$$P(C = c) = \frac{N_c}{N}$$

N_c: Number of records with class c

N = Number of records

P(C = No) = 7/10

P(C = Yes) = 3/10

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

Discrete attributes:

$$P(A_i = a | C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

$N_{a,c}$: number of instances of class c

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

Discrete attributes:

$$P(A_i = a|C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

P(Refund = Yes|No) = 3/7

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

Discrete attributes:

$$P(A_i = a|C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

P(Refund = Yes|Yes) = 0

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

Discrete attributes:

$$P(A_i = a|C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

P(Status=Single|No) = 2/7

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

Discrete attributes:

$$P(A_i = a | C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

P(Status=Single|Yes) = 2/3

How to Estimate Probabilities from Data?

Tid	**Refund**	**Marital Status**	**Taxable Income**	**Evade**
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**

- Normal distribution:

$$P(A_i = a \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(a-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

  - One for each (a_i, ci) pair
- For Class=No
  - sample mean $\mu = 110$
  - sample variance $\sigma^2 = 2975$
- For Income = 80

$$P(Income = 80 \mid No) = \frac{1}{\sqrt{2\pi}(54.54)} e^{-\frac{(80-110)^2}{2(2975)}} = 0.0062$$

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

- Normal distribution:

$$P(A_i = a \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(a-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

  - One for each (a_i, ci) pair
- For Class=Yes
  - sample mean $\mu = 90$
  - sample variance $\sigma^2 = 2975$
- For Income = 80

$$P(Income = 80 \mid Yes) = \frac{1}{\sqrt{2\pi}(5)} e^{-\frac{(80-90)^2}{2(25)}} = 0.01$$

Example

- Record

  X = (Refund = Yes, Status = Single, Income =80K)

- We compute:
  - P(C = Yes|X) = P(C = Yes)*P(Refund = Yes |C = Yes)

    *P(Status = Single |C = Yes)

    *P(Income =80K |C= Yes)

    = 3/10* 0 * 2/3 * 0.01 = 0
  - P(C = No|X) = P(C = No)*P(Refund = Yes |C = No)

    *P(Status = Single |C = No)

    *P(Income =80K |C= No)

    = 7/10 * 3/7 * 2/7 * 0.0062 = 0.0005

Example of Naïve Bayes Classifier

- Creating a Naïve Bayes Classifier, essentially means to compute counts:

Total number of records: N = 10

Class No:
Number of records: 7
Attribute Refund:
- Yes: 3
- No: 4

Attribute Marital Status:
- Single: 2
- Divorced: 1
- Married: 4

Attribute Income:
- mean: 110
- variance: 2975

Class Yes:
Number of records: 3
Attribute Refund:
- Yes: 0
- No: 3

Attribute Marital Status:
- Single: 2
- Divorced: 1
- Married: 0

Attribute Income:
- mean: 90
- variance: 25

Example of Naïve Bayes Classifier

Given a Test Record:

$$X = (\text{Refund} = \text{Yes}, \text{Status} = \text{Single}, \text{Income} = 80K)$$

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1

P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/7
P(Marital Status=Divorced|Yes)=1/7
P(Marital Status=Married|Yes) = 0

For taxable income:

If class=No: sample mean=110
sample variance=2975

If class=Yes: sample mean=90
sample variance=25

- P(X|Class=No) = P(Refund=Yes|Class=No) × P(Married| Class=No) × P(Income=120K| Class=No) = 3/7 * 2/7 * 0.0062 = 0.00075
- P(X|Class=Yes) = P(Refund=No| Class=Yes) × P(Married| Class=Yes) × P(Income=120K| Class=Yes) = 0 * 2/3 * 0.01 = 0
- P(No) = 0.3, P(Yes) = 0.7

Since P(X|No)P(No) > P(X|Yes)P(Yes)

Therefore P(No|X) > P(Yes|X)

=> Class = No

Naïve Bayes Classifier

- If one of the conditional probability is zero, then the entire expression becomes zero

- Laplace Smoothing:

$$P(A_i = a|C = c) = \frac{N_{ac} + 1}{N_c + N_i}$$

  - N_i: number of attribute values for attribute A_i

Example of Naïve Bayes Classifier

Given a Test Record:

With Laplace Smoothing

X = (Refund = Yes, Status = Single, Income =80K)

naive Bayes Classifier:

P(Refund=Yes|No) = 4/9
P(Refund=No|No) = 5/9
P(Refund=Yes|Yes) = 1/5
P(Refund=No|Yes) = 4/5

P(Marital Status=Single|No) = 3/10
P(Marital Status=Divorced|No)=2/10
P(Marital Status=Married|No) = 5/10
P(Marital Status=Single|Yes) = 3/6
P(Marital Status=Divorced|Yes)=2/6
P(Marital Status=Married|Yes) = 1/6

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- P(X|Class=No) = P(Refund=No|Class=No)
  $\times$ P(Married| Class=No)
  $\times$ P(Income=120K| Class=No)
  = 4/9 $\times$ 3/10 $\times$ 0.0062 = 0.00082

- P(X|Class=Yes) = P(Refund=No| Class=Yes)
  $\times$ P(Married| Class=Yes)
  $\times$ P(Income=120K| Class=Yes)
  = 1/5 $\times$ 3/6 $\times$ 0.01 = 0.001

- P(No) = 0.7, P(Yes) = 0.3
- P(X|No)P(No) = 0.0005
- P(X|Yes)P(Yes) = 0.0003

=> Class = No

Implementation details

- Computing the conditional probabilities involves multiplication of many very small numbers
  - Numbers get very close to zero, and there is a danger of numeric instability
- We can deal with this by computing the logarithm of the conditional probability

$$\log P(C|A) \sim \log P(A|C) + \log P(A)$$
$$= \sum_i \log(A_i|C) + \log P(A)$$

Naïve Bayes for Text Classification

- Naïve Bayes is commonly used for text classification
- For a document with k terms $d = (t_1, \ldots, t_k)$

Fraction of documents in c

$$P(c|d) = P(c)P(d|c) = P(c)\prod_{t_i \in d} P(t_i|c)$$

- $P(t_i|c)$= Fraction of terms from all documents in c that are t_i.

Number of times t_i appears in some document in c

Laplace Smoothing

$$\boldsymbol{P(t_i|c) = \frac{N_{ic} + 1}{N_c + T}}$$

Total number of terms in all documents in c

Number of unique words (vocabulary size)

- Easy to implement and works relatively well
- Limitation: Hard to incorporate additional features (beyond words).
  - E.g., number of adjectives used.

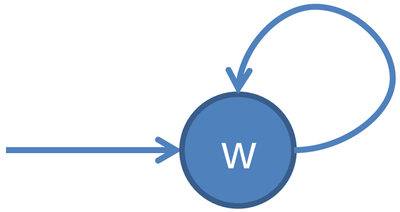
Multinomial document model

- Probability of document $d = (t_1, \ldots, t_k)$ in class c:

$$P(d|c) = P(c) \prod_{t_i \in d} P(t_i|c)$$

- This formula assumes a multinomial distribution for the document generation:
  - If we have probabilities $p_1, \ldots, p_T$ for events $t_1, \ldots, t_T$ the probability of a subset of these is

$$P(d) = \frac{N}{N_{t_1}!\, N_{t_2}! \cdots N_{t_T}!} {p_1}^{N_{t_1}} {p_2}^{N_{t_2}} \cdots {p_T}^{N_{t_T}}$$



- Equivalently: There is an automaton spitting words from the above distribution

```
TrainMultinomialNB(ℂ, 𝔻)
V ← ExtractVocabulary(𝔻)
N ← CountDocs(𝔻)
for each c ∈ ℂ
do N_c ← CountDocsInClass(𝔻, c)
    prior[c] ← N_c / N
    text_c ← ConcatenateTextOfAllDocsInClass(𝔻, c)
    for each t ∈ V
    do T_ct ← CountTokensOfTerm(text_c, t)
    for each t ∈ V
    do condprob[t][c] ← (T_ct + 1) / Σ_t'(T_ct' + 1)
return V, prior, condprob

ApplyMultinomialNB(ℂ, V, prior, condprob, d)
W ← ExtractTokensFromDoc(V, d)
for each c ∈ ℂ
do score[c] ← log prior[c]
    for each t ∈ W
    do score[c] += log condprob[t][c]
return arg max_{c∈ℂ} score[c]
```

► **Figure 13.2** Naive Bayes algorithm (multinomial model): Training and testing.

Example

News titles for Politics and Sports

	Politics	Sports
documents	"Obama meets Merkel" "Obama elected again" "Merkel visits Greece again"	"OSFP European basketball champion" "Miami NBA basketball champion" "Greece basketball coach?"
	P(p) = 0.5	P(s) = 0.5
terms Vocabulary size: 14	obama:2, meets:1, merkel:2, elected:1, again:2, visits:1, greece:1	OSFP:1, european:1, basketball:3, champion:2, miami:1, nba:1, greece:1, coach:1
	Total terms: 10	Total terms: 11

New title: X = "Obama likes basketball"

P(Politics|X) ~ P(p)*P(obama|p)*P(likes|p)*P(basketball|p)
= 0.5 * 3/(10+14) *1/(10+14) * 1/(10+14) = 0.000108

P(Sports|X) ~ P(s)*P(obama|s)*P(likes|s)*P(basketball|s)
= 0.5 * 1/(11+14) *1/(11+14) * 4/(11+14) = 0.000128

Naïve Bayes (Summary)

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
  - Use other techniques such as Bayesian Belief Networks (BBN)
- Naïve Bayes can produce a probability estimate, but it is usually a very biased one
  - Logistic Regression is better for obtaining probabilities.

SUPERVISED LEARNING

Learning

- Supervised Learning: learn a model from the data using labeled data.
  - Classification and Regression are the prototypical examples of supervised learning tasks. Other are possible (e.g., ranking)
- Unsupervised Learning: learn a model – extract structure from unlabeled data.
  - Clustering and Association Rules are prototypical examples of unsupervised learning tasks.
- Semi-supervised Learning: learn a model for the data using both labeled and unlabeled data.

Supervised Learning Steps

- Model the problem
  - What is you are trying to predict? What kind of optimization function do you need? Do you need classes or probabilities?
- Extract Features
  - How do you find the right features that help to discriminate between the classes?
- Obtain training data
  - Obtain a collection of labeled data. Make sure it is large enough, accurate and representative. Ensure that classes are well represented.
- Decide on the technique
  - What is the right technique for your problem?
- Apply in practice
  - Can the model be trained for very large data? How do you test how you do in practice? How do you improve?

Modeling the problem

- Sometimes it is not obvious. Consider the following three problems
  - Detecting if an email is spam
  - Categorizing the queries in a search engine
  - Ranking the results of a web search

Feature extraction

- Feature extraction, or feature engineering is the most tedious but also the most important step
  - How do you separate the players of the Greek national team from those of the Swedish national team?

- One line of thought: throw features to the classifier and the classifier will figure out which ones are important
  - More features, means that you need more training data
- Another line of thought: Feature Selection: Select carefully the features using various functions and techniques
  - Computationally intensive

Training data

- An overlooked problem: How do you get labeled data for training your model?
  - E.g., how do you get training data for ranking?
- Usually requires a lot of manual effort and domain expertise and carefully planned labeling
  - Results are not always of high quality (lack of expertise)
  - And they are not sufficient (low coverage of the space)
- Recent trends:
  - Find a source that generates the labeled data for you.
  - Crowd-sourcing techniques

Dealing with small amount of labeled data

- Semi-supervised learning techniques have been developed for this purpose.
- Self-training: Train a classifier on the data, and then feed back the high-confidence output of the classifier as input
- Co-training: train two “independent” classifiers and feed the output of one classifier as input to the other.
- Regularization: Treat learning as an optimization problem where you define relationships between the objects you want to classify, and you exploit these relationships
  - Example: Image restoration

Technique

- The choice of technique depends on the problem requirements (do we need a probability estimate?) and the problem specifics (does independence assumption hold? do we think classes are linearly separable?)
- For many cases finding the right technique may be trial and error
- For many cases the exact technique does not matter.

Big Data Trumps Better Algorithms

- If you have enough data then the algorithms are not so important

- The web has made this possible.
  - Especially for text-related tasks
  - Search engine uses the collective human intelligence

Google lecture:
[Theorizing from the Data]

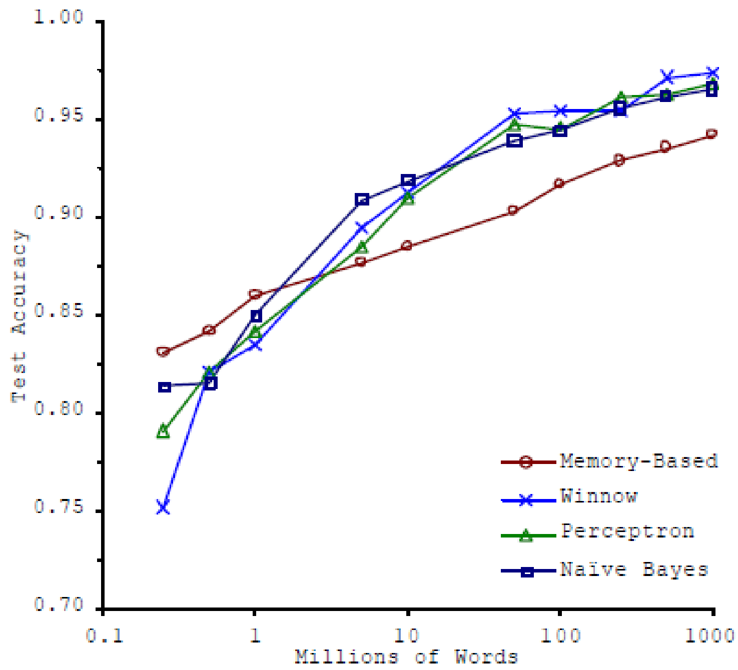


Figure 1. Learning Curves for Confusion Set Disambiguation

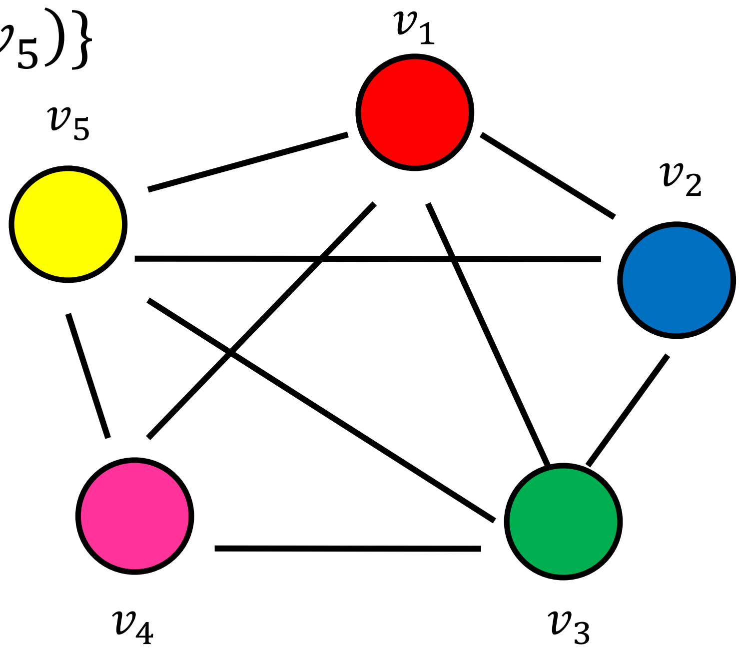
Apply-Test

- How do you scale to very large datasets?
  - Distributed computing – map-reduce implementations of machine learning algorithms (Mahut, over Hadoop)
- How do you test something that is running online?
  - You cannot get labeled data in this case
  - A/B testing
- How do you deal with changes in data?
  - Active learning

GRAPHS AND LINK ANALYSIS RANKING

Graphs - Basics

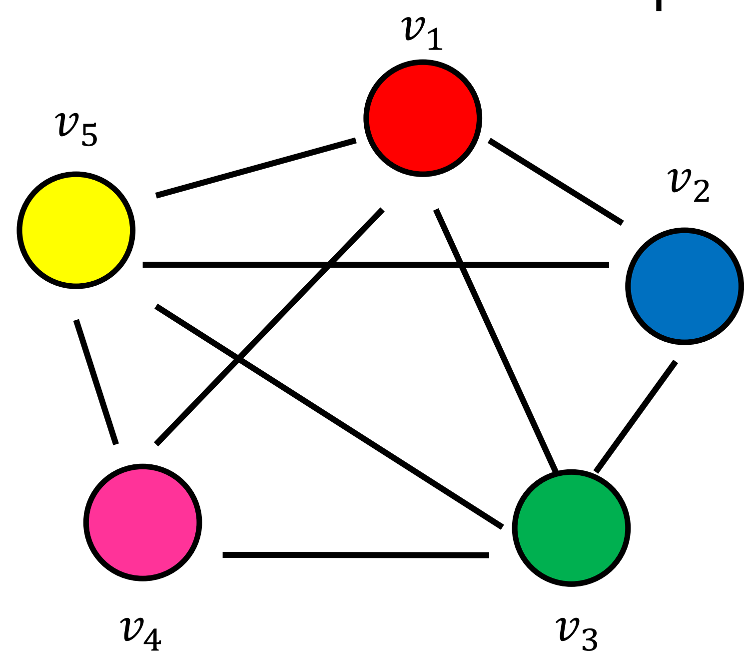
- A graph is a powerful abstraction for modeling entities and their pairwise relationships.
- G = (V,E)
  - Set of nodes $V = \{v_1, \dots, v_5\}$
  - Set of edges $E = \{(v_1, v_2), \dots (v_4, v_5)\}$
- Examples:
  - Social network
  - Twitter Followers
  - Web
  - Collaboration graphs



Undirected Graphs

- Undirected Graph: The edges are undirected pairs – they can be traversed in any direction.
- Degree of node: Number of edges incident on the node
- Path: A sequence of edges from one node to another
  - We say that the node is reachable
- Connected Component: A set of nodes such that there is a path between any two nodes in the set

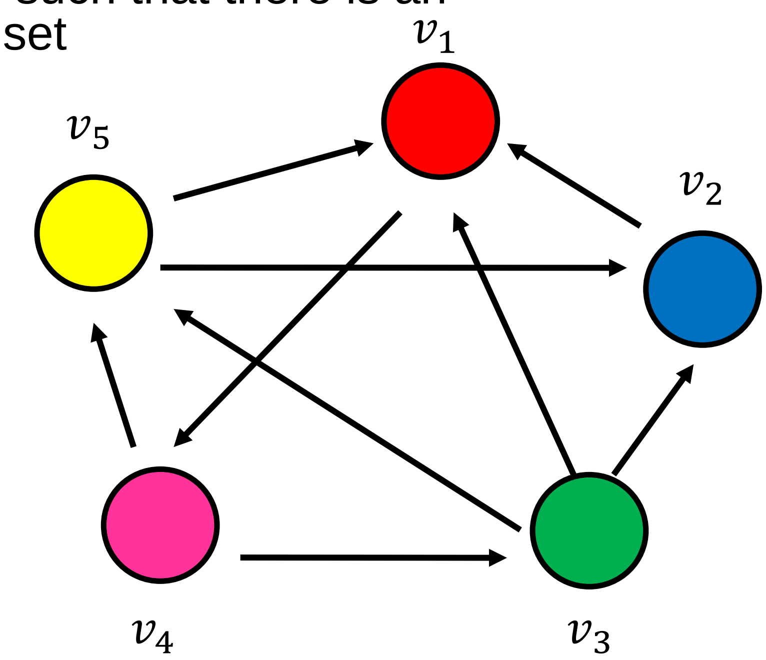
$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$



Directed Graphs

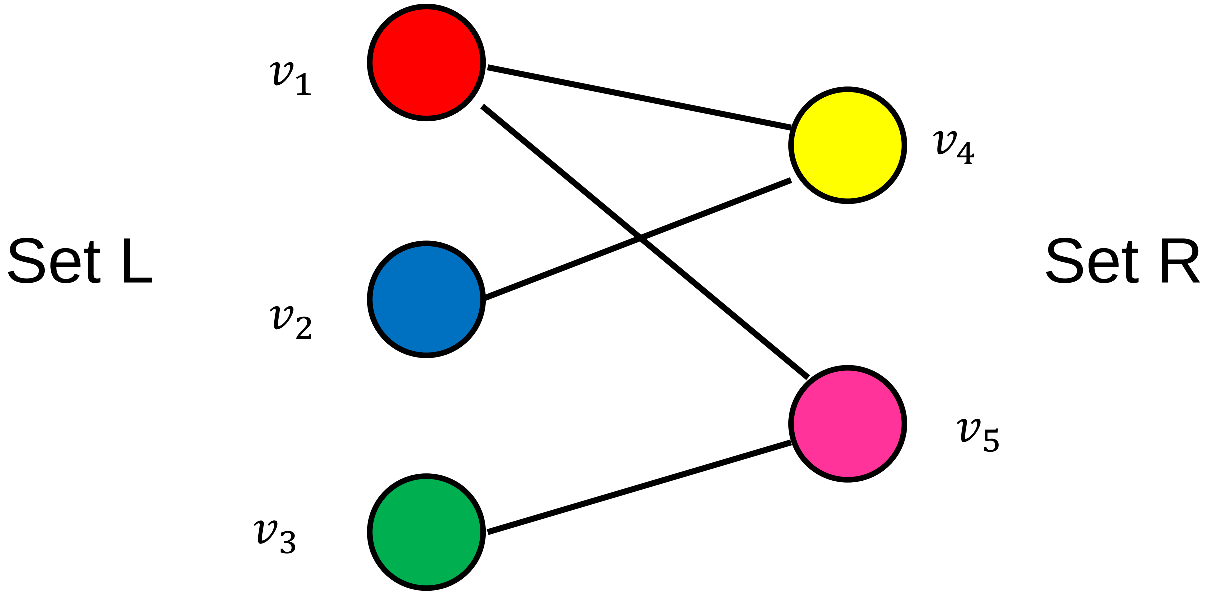
- Directed Graph: The edges are ordered pairs – they can be traversed in the direction from first to second.
- In-degree and Out-degree of a node.
- Path: A sequence of directed edges from one node to another
  - We say that the node is reachable
- Strongly Connected Component: A set of nodes such that there is a directed path between any two nodes in the set
- Weakly Connected Component: A set of nodes such that there is an undirected path between any two nodes in the set

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$



Bipartite Graph

- A graph where the vertex set V is partitioned into two sets V = {L,R}, of size greater than one, such that there is no edge within each set.



Mining the graph structure

- A graph is a combinatorial object, with a certain structure.
- Mining the structure of the graph reveals information about the entities in the graph
  - E.g., if in the Facebook graph I find that there are 100 people that are all linked to each other, then these people are likely to be a community
    - The community discovery problem
  - By measuring the number of friends in the facebook graph I can find the most important nodes
    - The node importance problem
- We will now focus on the node importance problem

Importance problem

- What are the most important nodes in the graph?
  - What are the most authoritative pages on the web
  - Who are the important users in Facebook?
  - What are the most influential Twitter accounts?

Link Analysis

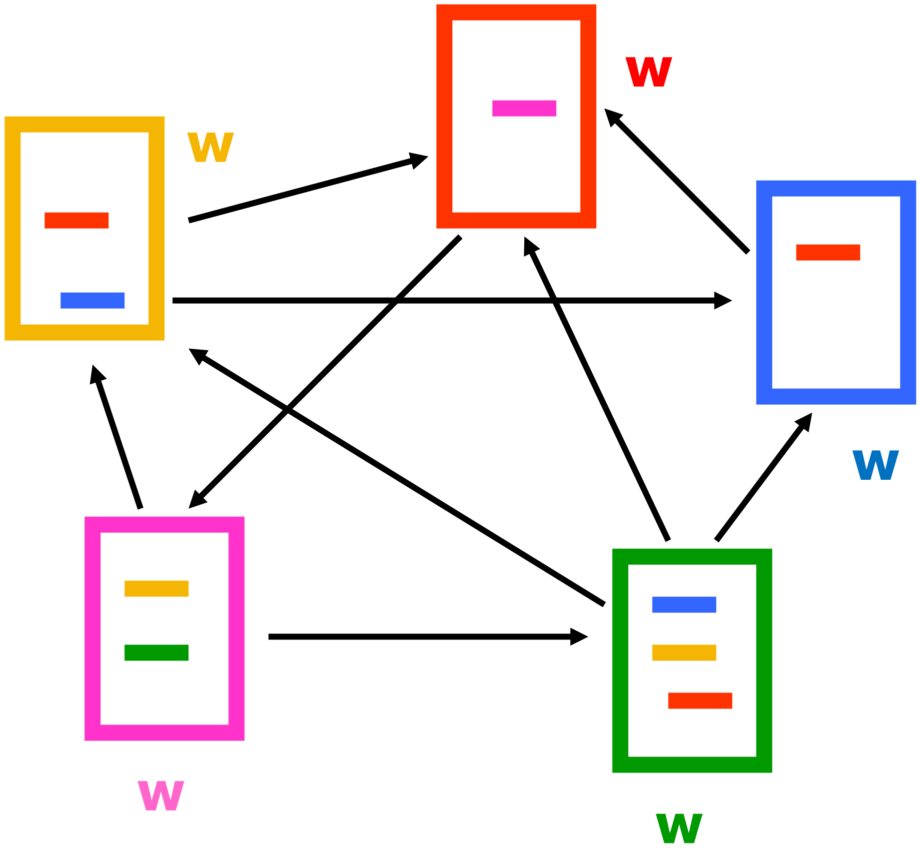
- First generation search engines
  - view documents as flat text files
  - could not cope with size, spamming, user needs
- Second generation search engines
  - Ranking becomes critical
  - shift from relevance to authoritativeness
    - authoritativeness: the static importance of the page
  - use of Web specific data: Link Analysis of the Web graph
  - a success story for the network analysis + a huge commercial success
  - it all started with two graduate students at Stanford

Link Analysis: Intuition

- A link from page p to page q denotes endorsement
  - page p considers page q an authority on a subject
  - use the graph of recommendations
  - assign an authority value to every page

- The same idea applies to other graphs as well
  - Twitter graph, where user p follows user q

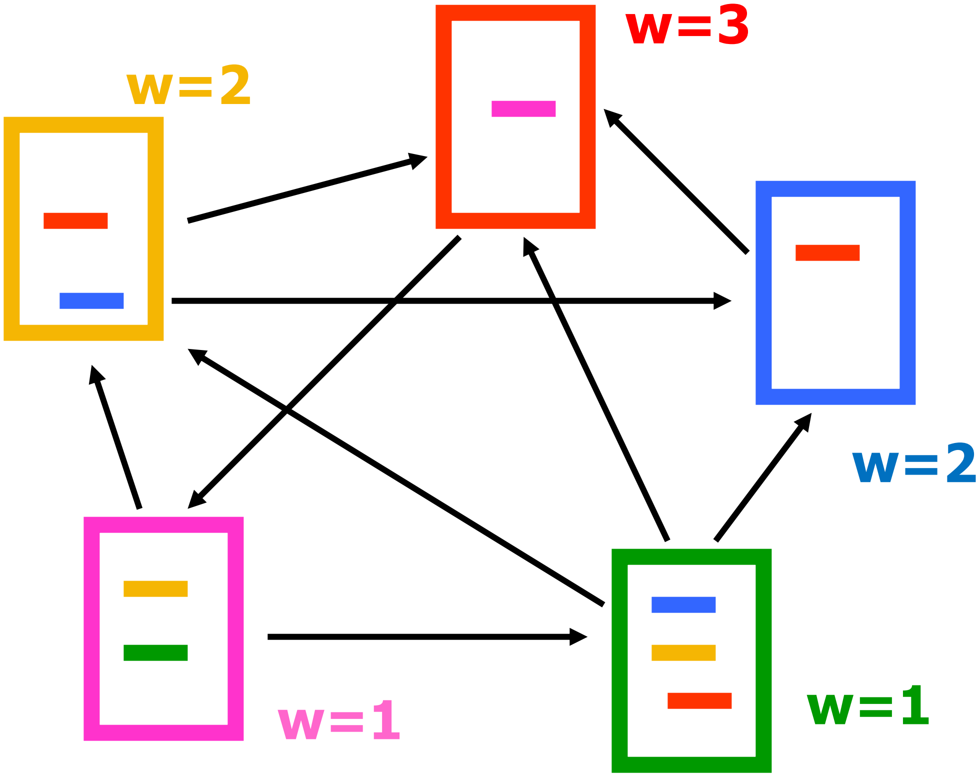
Constructing the graph



- Goal: output an authority weight for each node
  - Also known as centrality, or importance

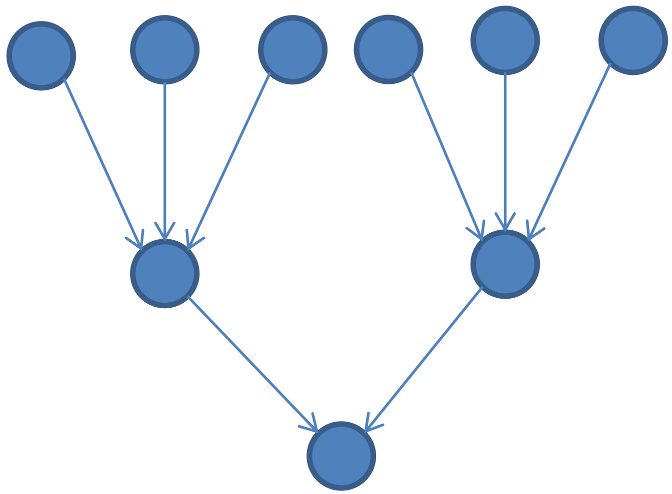
Rank by Popularity

- Rank pages according to the number of incoming edges (in-degree, degree centrality)



1. Red Page
2. Yellow Page
3. Blue Page
4. Purple Page
5. Green Page

Popularity

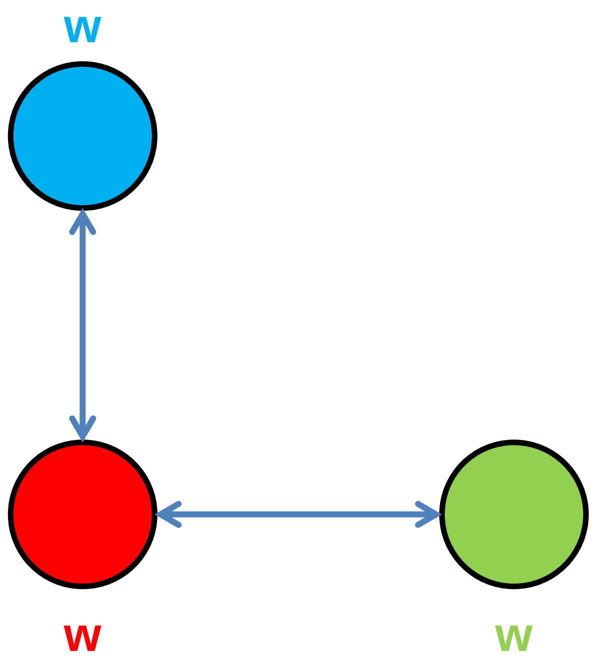


- It is not important only how many link to you, but how important are the people that link to you.
- Good authorities are pointed by good authorities
  - Recursive definition of importance

PageRank

- Good authorities should be pointed by good authorities
  - The value of a page is the value of the people that link to you

- How do we implement that?
  - Assume that we have a unit of authority to distribute to all nodes.
  - Each node distributes the authority value they have to their neighbors
  - The authority value of each node is the sum of the authority fractions it collects from its neighbors.
  - Solving the system of equations we get the authority values for the nodes
    - $w = \frac{1}{2}$, $w = \frac{1}{4}$, $w = \frac{1}{4}$



$$w + w + w = 1$$

$$w = w + w$$

$$w = \tfrac{1}{2}\, w$$

$$w = \tfrac{1}{2}\, w$$

A more complex example

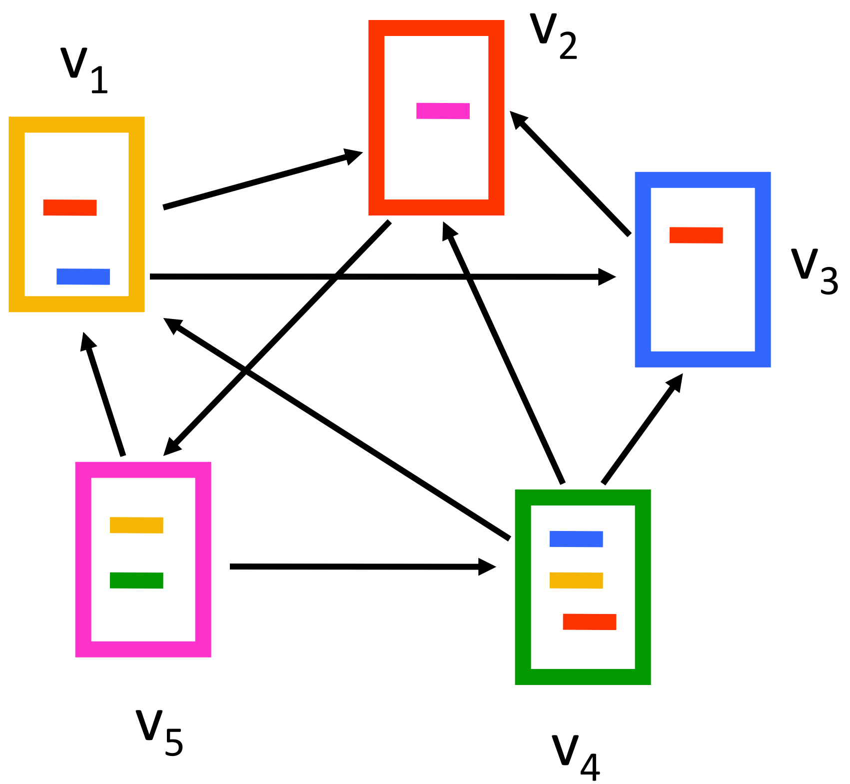
$w_1 = 1/3\ w_4 + 1/2\ w_5$

$w_2 = 1/2\ w_1 + w_3 + 1/3\ w_4$

$w_3 = 1/2\ w_1 + 1/3\ w_4$

$w_4 = 1/2\ w_5$

$w_5 = w_2$



$$PR(p) = \sum_{q \to p} \frac{PR(q)}{|Out(q)|}$$

Random Walks on Graphs

- What we described is equivalent to a random walk on the graph

- Random walk:
  - Start from a node uniformly at random
  - Pick one of the outgoing edges uniformly at random
  - Repeat.

Random walks on graphs

- Question: what is the probability of being at a specific node?
  - p_i: probability of being at node i at this step
  - p_i': probability of being at node i in the next step

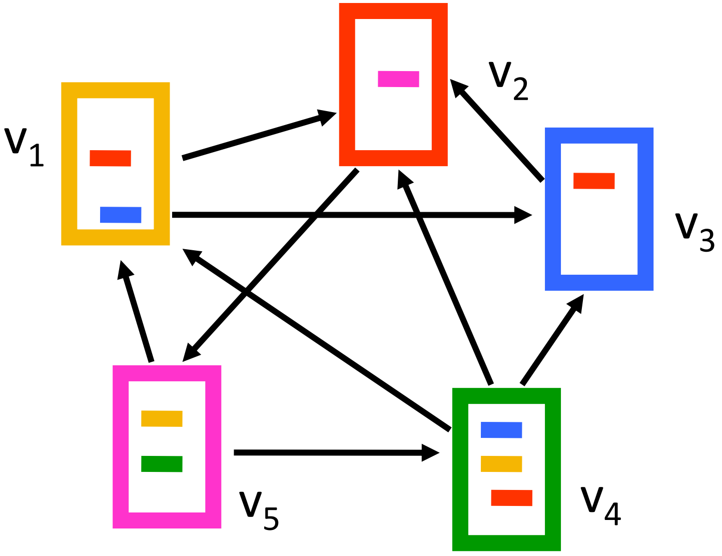
$$p'_1 = 1/3\ p_4 + 1/2\ p_5$$

$$p'_2 = 1/2\ p_1 + p_3 + 1/3\ p_4$$

$$p'_3 = 1/2\ p_1 + 1/3\ p_4$$

$$p'_4 = 1/2\ p_5$$

$$p'_5 = p_2$$



- After many steps the probabilities converge to the stationary distribution of the random walk.