# Text Categorization: Discriminative Classifiers (Part 1)

# Anatomy of Naïve Bayes Classifier (repeated)

Two categories: $\theta_1$ and $\theta_2$

$$score(d) = \log \frac{p(\theta_1 | d)}{p(\theta_2 | d)} = \log \frac{p(\theta_1)\prod_{w \in V} p(w|\theta_1)^{c(w,d)}}{p(\theta_2)\prod_{w \in V} p(w|\theta_2)^{c(w,d)}}$$

$$= \boxed{\log \frac{p(\theta_1)}{p(\theta_2)}} + \sum_{w \in V} \underline{c(w,d)} \boxed{\log \frac{p(w|\theta_1)}{p(w|\theta_2)}}$$

Category bias ($\beta_0$) doesn't depend on d!

Sum over all words (features $\{x_i\}$ )

Feature value: $x_i$=c(w,d)

Weight on each word (feature) $\beta_i$

Generalize

$$d = (x_1, x_2, ..., x_M), \quad x_i \in \Re$$

$$score(d) = \beta_0 + \sum_{i=1}^{M} x_i \beta_i \quad \beta_i \in \Re$$

= Logistic Regression!

# Discriminative Classifier 1: Logistic Regression

**Binary Response Variable: Y ∈{0,1}**

**Predictors:** $X = (x_1, x_2, ..., x_M), \; x_i \in \Re$

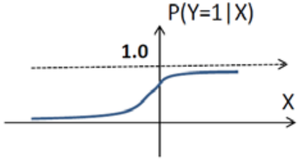
$$Y = \begin{cases} 1 & category(d) = \theta_1 \\ 0 & category(d) = \theta_2 \end{cases}$$

**Modeling p(Y|X) directly**

**Allow many other features than words!**

$$\log \frac{p(\theta_1 | d)}{p(\theta_2 | d)} = \log \frac{p(Y=1|X)}{p(Y=0|X)} = \log \frac{p(Y=1|X)}{1 - p(Y=1|X)} = \beta_0 + \sum_{i=1}^{M} x_i \beta_i \quad \beta_i \in \Re$$

$$p(Y=1|X) = \frac{e^{\beta_0 + \sum_{i=1}^{M} x_i \beta_i}}{e^{\beta_0 + \sum_{i=1}^{M} x_i \beta_i} + 1}$$






# Estimation of Parameters

- Training Data: T={(Xi, Yi)}, i=1,2, ..., |T|
- Parameters: $\vec{\beta} = (\beta_0, \beta_1, ..., \beta_M)$
- Conditional likelihood: $p(T | \vec{\beta}) = \prod_{i=1}^{|T|} p(Y = Y_i | X = X_i, \vec{\beta})$

$Y_i$=1

$$p(Y=1|X) = \frac{e^{\beta_0 + \sum_{i=1}^{M} x_i \beta_i}}{e^{\beta_0 + \sum_{i=1}^{M} x_i \beta_i} + 1}$$

$Y_i$=0

$$p(Y=0|X) = \frac{1}{e^{\beta_0 + \sum_{i=1}^{M} x_i \beta_i} + 1}$$

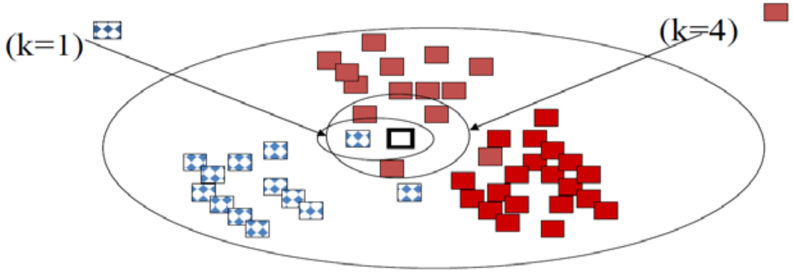
- Maximum Likelihood estimate $\vec{\beta}^* = \arg\max_{\vec{\beta}} p(T | \vec{\beta})$

**Can be computed in many ways (e.g., Newton's method)**

# Discriminative Classifier 2: K-Nearest Neighbors (K-NN)

- Find **k examples** in the training set that are **most similar** to the text object to be classified ("neighbor" documents)
- Assign the **category that is most common** in these neighbor text objects (neighbors vote for the category)
- Can be improved by considering the distance of a neighbor (a closer neighbor has more influence)
- Can be regarded as a way to **directly estimate the conditional probability** of label given data instance, i.e., p(Y|X)
- Need a **similarity function** to measure similarity of two text objects

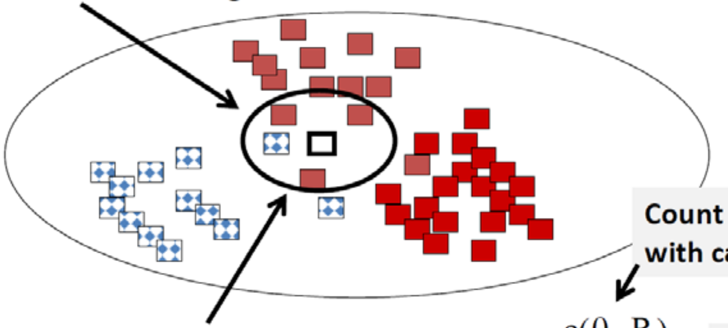
# Illustration of K-NN Classifier




# K-NN as an Estimate of p(Y|X)

Assume $p(\theta_i|d)$ is <u>locally smooth</u>, i.e., the same for all the d's in this region R

$p(\theta_i|d) = p(\theta_i|R)$



Count of d's in R with category $\theta_i$

$$p(\theta_i \mid R) = \frac{c(\theta_i, R)}{|R|}$$

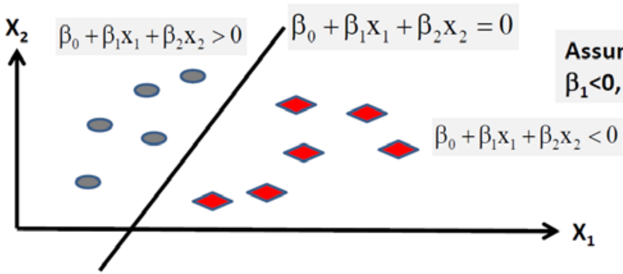
Total # of docs in R

Estimate $p(\theta_i|R)$ based on the known categories in the region

# Discriminative Classifier 3: Support Vector Machine (SVM)

- Consider two categories: $\{\theta_1, \theta_2\}$
- Use a linear separator $f(X) = \beta_0 + \sum_{i=1}^{M} x_i \beta_i \quad \beta_i \in \Re$

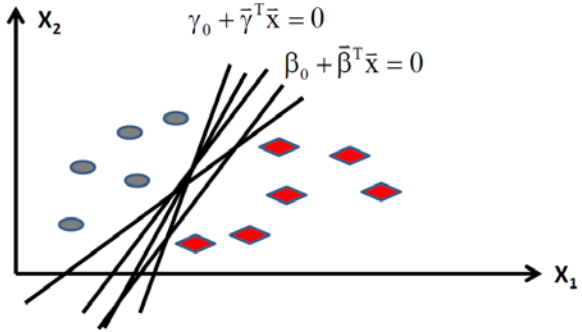
$$f(X) \geq 0 \Rightarrow X \text{ is in category } \theta_1$$
$$f(X) < 0 \Rightarrow X \text{ is in category } \theta_2$$




**Assume**
**$\beta_1<0$, $\beta_2>0$**



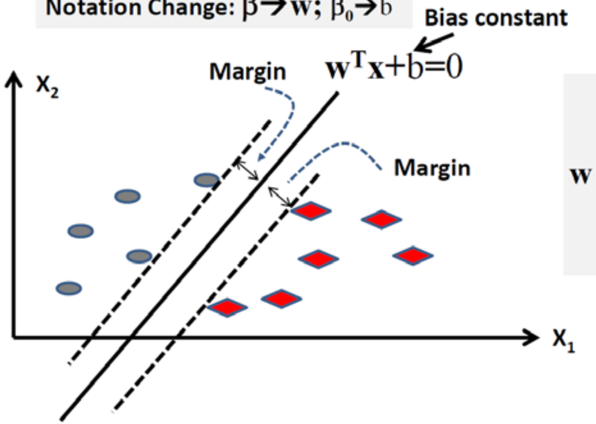
# Which Linear Separator Is the Best?






# Best Separator = Maximize the Margin

**Notation Change: $\beta \rightarrow \mathbf{w}$; $\beta_0 \rightarrow b$**




**Feature Weights**

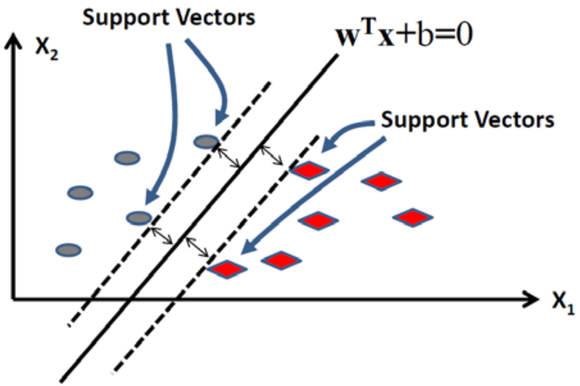
$$\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \ldots \\ w_M \end{pmatrix}$$

**Feature Vector (e.g., word counts)**

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_M \end{pmatrix}$$



# Only the Support Vectors Matter



# Linear SVM

**Classifier:** $f(x)=w^T x+b$

**Parameters:** $w$, $b$

$f(X) \geq 0 \Rightarrow X$ is in category $\theta_1$

$f(X) < 0 \Rightarrow X$ is in category $\theta_2$

**Training Data:** $T=\{(x_i, y_i)\}$, $i=1, \ldots, |T|$. $x_i$ is a feature vector; $y_i \in \{-1, 1\}$

**Goal 1: Correct labeling on training data:**

If $y_i=1$ ➔ $w^T x_i+b \geq 1$

If $y_i=-1$ ➔ $w^T x_i+b \leq -1$

**Constraint**

$$\boxed{\forall i,\ y_i(w^T x_i+b)\geq 1}$$

**Objective**

$$\boxed{\text{Minimize } \Phi(w)=w^T w}$$

**Goal 2: Maximize margin**

Large margin $\Leftrightarrow$ Small $w^T w$

**The optimization problem is quadratic programming with linear constraints**

# Linear SVM with Soft Margin

**Classifier:** $f(x)=w^T x+b > 0?$

**Parameters:** $w$, $b$

**Training Data:** $T=\{(x_i, y_i)\}$, $i=1, \ldots, |T|$.

**Added to allow training errors**

**Find w, b, and $\xi_i$ to minimize** $\Phi(w)=w^T w \boxed{+C\Sigma_{i\in[1,|T|]}\xi_i}$

**Subject to** $\forall i\in[1,|T|],\ y_i(w^T x_i+b)\geq 1 \boxed{-\xi_i, \quad \xi_i\geq 0}$

C>0 is a parameter to control the trade-off between minimizing the errors and maximizing the margin

**The optimization problem is still quadratic programming with linear constraints**

# Summary of Text Categorization Methods

- **Many methods** are available, but no clear winner
  - All require **effective feature representation** (need domain knowledge)
  - It is useful to **compare/combine multiple methods** for a particular problem
- Most techniques rely on **supervised machine learning** and thus can be applied to **any text categorization** problem!
  - **Humans annotate** training data and design features
  - **Computer optimizes** the combination of features
  - Good performance requires 1) **effective features** and 2) **plenty of training data**
  - Performance is generally **(much) more affected** by the **effectiveness of features** than by the choice of a specific classifier
- How to design effective features? (application-specific)
  - **Analyze** the categorization problem and **exploit** domain knowledge
  - Perform **error analysis** to obtain insights
  - Leverage **ML techniques** (e.g., feature selection, dimension reduction, deep learning)
- How to obtain "enough" training examples?
  - Low-quality ("pseudo") training examples may be leveraged
  - Exploit **unlabeled data** (using **semi-supervised** learning techniques)
  - **Domain adaptation/transfer learning** ("borrow" training examples from a related domain/problem)

# Text Categorization: Evaluation (Part 1)

# General Evaluation Methodology

- Have **humans** to create a **test collection** where every document is tagged with the desired categories ("ground truth")
- Generate **categorization** results **using a system** on the test collection
- **Compare** the system categorization **decisions** with the human-made categorization decisions **and quantify** their similarity (or equivalently difference)
  - The higher the similarity is, the better the results are
  - **Similarity** can be measured from **different perspectives** to understand the quality of results in detail (e.g., which category performs better?)
  - In general, **different categorization mistakes** may have a **different cost** that inevitably depends on specific applications, but it is **okay not to consider** such a cost variation for **relative comparison of methods**

## Classification Accuracy (Percentage of Correct Decisions)

| | $c_1$ | $c_2$ | $c_3$ | … | $c_k$ |
|---|---|---|---|---|---|
| $d_1$ | y(+) | y(-) | n(+) | | n(+) |
| $d_2$ | y(-) | n(+) | y(+) | | n(+) |
| $d_3$ | n(+) | n(+) | y(+) | | n(+) |
| … | | | | | |
| $d_N$ | … | … | | | |

**+/- human answer**
(+= correct; - =incorrect)
**y/n system result**
(y=yes; n=no)

$$\text{Classification Accuracy} = \frac{\text{Total number of correct decisions}}{\text{Total number of decisions made}}$$

$$= \frac{\text{count}(y(+)) + \text{count}(n(-))}{kN}$$



## Problems with Classification Accuracy

- **Some decision errors** are **more serious** than others
  - It may be more important to **get the decisions right on some documents** than others
  - It may be more important to **get the decisions right on some categories** than others
  - E.g., spam filtering: **missing a legitimate email** costs more than letting a spam go
- Problem with **imbalanced test set**
  - Skewed test set: 98% in category 1; 2% in category 2
  - Strong baseline: put all instances in category 1 ➔ 98% accuracy!

# Per-Document Evaluation

| | $c_1$ | $c_2$ | $c_3$ | … | $c_k$ |
|---|---|---|---|---|---|
| $d_1$ | y(+) | y(-) | n(+) | | n(+) |
| $d_2$ | y(-) | n(+) | y(+) | | n(+) |
| $d_3$ | n(+) | n(+) | y(+) | | n(+) |

How good are the decisions on $d_i$?

**When the system says "yes," how many are correct?**

**Precision** $= \dfrac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$

| | System ("y") | System ("n") |
|---|---|---|
| **Human (+)** | True Positives TP | False Negatives FN |
| **Human (-)** | False Positives FP | True Negatives TN |

**Recall** $= \dfrac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$

**Does the doc have all the categories it should have?**



# Per-Category Evaluation

| | $c_1$ | $c_2$ | $c_3$ | … | $c_k$ |
|---|---|---|---|---|---|
| $d_1$ | y(+) | y(-) | n(+) | | n(+) |
| $d_2$ | y(-) | n(+) | y(+) | | n(+) |
| $d_3$ | n(+) | n(+) | y(+) | | n(- ) |

How good are the decisions on $c_i$?

**When the system says "yes," how many are correct?**

**Precision** $= \dfrac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$

| | System ("y") | System ("n") |
|---|---|---|
| **Human (+)** | True Positives TP | False Negatives FN |
| **Human (-)** | False Positives FP | True Negatives TN |

**Recall** $= \dfrac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$

**Has the category been assigned to all the docs of this category?**



# Combine Precision and Recall: F-Measure

$$F_\beta = \frac{1}{\frac{\beta^2}{\beta^2+1}\frac{1}{R} + \frac{1}{\beta^2+1}\frac{1}{P}} = \frac{(\beta^2+1)P*R}{\beta^2 P + R}$$

$$F_1 = \frac{2PR}{P+R}$$

**P**: precision

**R**: recall

β: parameter (often set to 1)

Why not 0.5*P+0.5*R?

**What is R if the system says "y" for all category-doc pairs?**

# Per-Category Evaluation

| | $c_1$ | $c_2$ | $c_3$ | … | $c_k$ |
|---|---|---|---|---|---|
| $d_1$ | y(+) | y(-) | n(+) | | n(+) |
| $d_2$ | y(-) | n(+) | y(+) | | n(+) |
| $d_3$ | n(+) | n(+) | y(+) | | n(- ) |

How good are the decisions on $c_i$?

| | System ("y") | System ("n") |
|---|---|---|
| **Human (+)** | True Positives<br>TP | False Negatives<br>FN |
| **Human (-)** | False Positives<br>FP | True Negatives<br>TN |

**When the system says "yes," how many are correct?**

$$\textbf{Precision} = \frac{TP}{TP + FP}$$

$$\textbf{Recall} = \frac{TP}{TP + FN}$$

**Has the category been assigned to all the docs of this category?**

# Text Categorization: Evaluation (Part 2)

## (Macro) Average Over All the Categories

| | $c_1$ | $c_2$ | $c_3$ | … | $c_k$ |
|---|---|---|---|---|---|
| $d_1$ | y(+) | y(-) | n(+) | | n(+) |
| $d_2$ | y(-) | n(+) | y(+) | | n(+) |
| $d_3$ | n(+) | n(+) | y(+) | | n(+) |
| … | | | | | |
| $d_N$ | … | … | | | |

Aggregate

| | | | | | | |
|---|---|---|---|---|---|---|
| **Precision** | p1 | p2 | p3 | … | pk | → **Overall Precision** |
| **Recall** | r1 | r2 | r3 | … | rk | → **Overall Recall** |
| **F-Measure** | f1 | f2 | f3 | … | fk | → **Overall F score** |



## (Macro) Average Over All the Documents

| | $c_1$ | $c_2$ | $c_3$ | … | $c_k$ | **Precision** | **Recall** | **F-Measure** |
|---|---|---|---|---|---|---|---|---|
| $d_1$ | y(+) | y(-) | n(+) | | n(+) | p1 | r1 | f1 |
| $d_2$ | y(-) | n(+) | y(+) | | n(+) | p2 | r2 | f2 |
| $d_3$ | n(+) | n(+) | y(+) | | n(+) | | | |
| … | | | | | | … | … | … |
| $d_N$ | | | | | | pN | rN | fN |

Aggregate

**Overall Precision** **Overall Recall** **Overall F score**



## Micro-Averaging of Precision and Recall

| | $c_1$ | $c_2$ | $c_3$ | … | $c_k$ |
|---|---|---|---|---|---|
| $d_1$ | y(+) | y(-) | n(+) | | n(+) |
| $d_2$ | y(-) | n(+) | y(+) | | n(+) |
| $d_3$ | n(+) | n(+) | y(+) | | n(+) |
| … | | | | | |
| $d_N$ | … | … | | | |

**First pool all decisions, then compute precision and recall**

| | System ("y") | System ("n") |
|---|---|---|
| **Human (+)** | True Positives ( TP) | False Negatives (FN) |
| **Human (-)** | False Positives(FP) | True Negatives(TN) |

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

# Sometimes Ranking Is More Appropriate

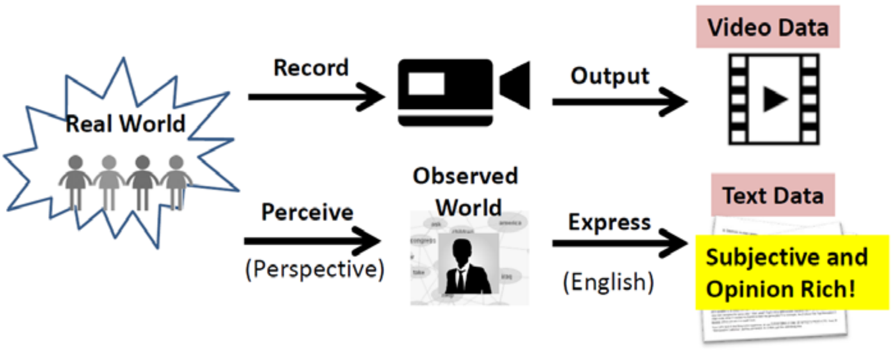
- The **categorization results** are often **passed to a human** for
  - further editing (e.g., correcting system mistakes on news categories)
  - prioritizing a task (e.g., routing an email to the right person for processing)
- In such cases, we can **evaluate the results** as a **ranked list** if the system can give scores for the decisions
  - E.g., discovery of **spam emails** (➔ rank emails for the "spam" category)
  - **Often more appropriate** to frame the problem as a **ranking problem** instead of a categorization problem (e.g., ranking documents in a search engine)

# Summary of Categorization Evaluation

- Evaluation is always very **important**, so **get it right**!
- Measures must **reflect the intended use** of the results for a particular application (e.g., spam filtering vs. news categorization)
  - Consider: How will the results be further processed (by a user)?
  - Ideally associate a different cost with each different decision error
- Commonly used **measures for relative comparison** of different methods:
  - **Accuracy, precision, recall, F score**
  - Variations: per-document, per-category, micro vs. macro averaging
- Sometimes **ranking** may be more appropriate

# Opinion Mining and Sentiment Analysis: Motivation

# Objective vs. Subjective Sensors




**How can we mine and analyze opinion buried in text?**



# A Sentence in News (Implicit Holder and Target)

Target

Holder

"... In an effort to get residents to wake up and pay attention to **Hurricane Sandy**, the **governor of Connecticut** just **said** that **Sandy** might be as **bad** as the **worst hurricane** ever to hit **New England**--the **hurricane of 1938**... "

Negative

Context

Target

**Harder to Mine and Analyze: Need deeper NLP**

Source: Blodget, H. (2012, October 28). Hurricane Sandy is being compared to the worst hurricane ever to hit New England. *Business Insider*. *Business Insider*. Retrieved from http://www.businessinsider.com/hurricane-sandy-vs-hurricane-of-1938-2012-10.



# Opinion Representation (Product Review )

- Basic Opinion Representation
  - Opinion **holder**: Whose opinion is this? **Reviewer X**
  - Opinion **target**: What is this opinion about? **Product: iPhone 6**
  - Opinion **content**: What exactly is the opinion? **Review Text**
- Enriched Opinion Representation
  - Opinion **context**: Under what situation (e.g., time, location) was the opinion expressed? **Year = 2015**
  - Opinion **sentiment**: What does the opinion tell us about the opinion holder's feeling (e.g., positive vs. negative)? **Positive**

**Relatively Easy to Mine and Analyze**

# What Is an Opinion?

**Objective** statement or **Factual** statement **(can be proved right/wrong)**

**Opinion** ≈ a **subjective** statement describing what a **person** **believes** or **thinks** about **something**

Opinion Target

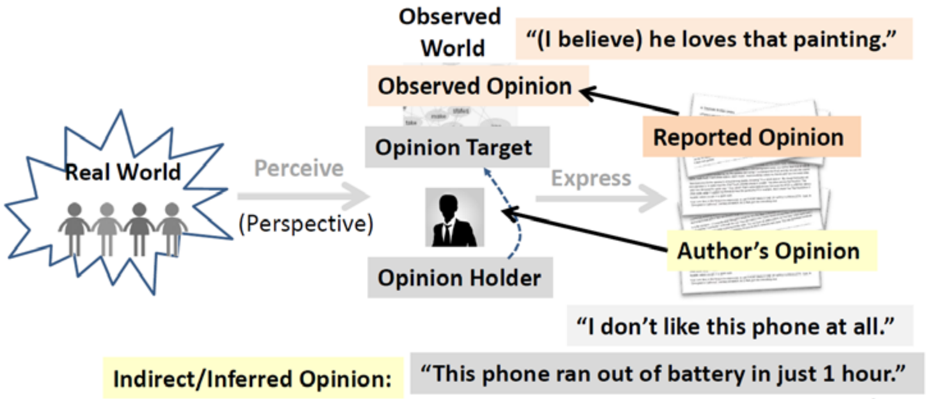
Opinion Holder

Depends on **culture, background, and context**

# Variations of Opinions

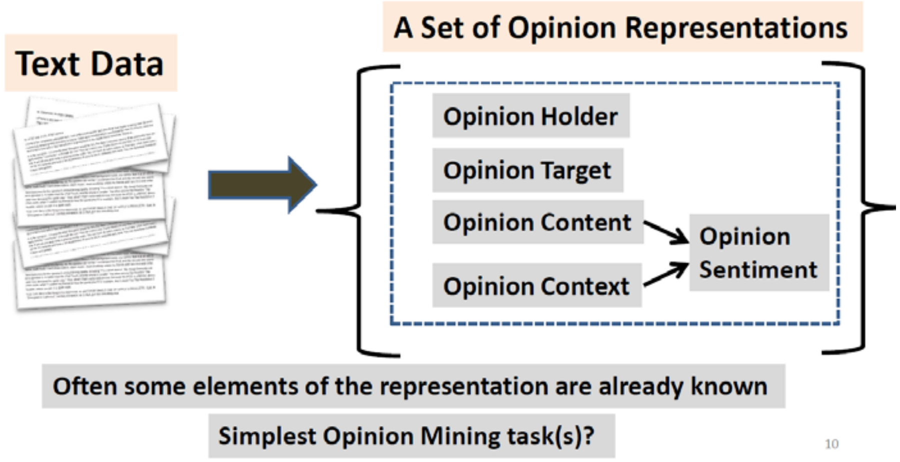
- **Opinion holder**: Individual vs. group
- **Opinion target**: One entity, a group of entities, one attribute of an entity, someone else's opinion, etc.
- **Opinion content**:
  - Surface variation: one sentence/phrase, a paragraph, a whole article
  - Sentiment/emotion variation: positive vs. negative, happy vs. sad, etc.
- **Opinion context**
  - Simple context: Different time, location, etc.
  - Complex context: Potentially includes the entire discourse context of an opinion

# Different Kinds of Opinions in Text Data






# The Task of Opinion Mining






# Why Opinion Mining?

- **Decision Support**
  - Help consumers **choose a product or service**
  - **Help voters** decide whom to vote for
  - **Help policy makers** design new policy
- **Understand People**
  - Help understand **people's preferences** to better serve them (e.g., optimize a product search engine; optimize recommender systems)
  - Help with **advertising** (targeted advertising)
- **"Voluntary Survey" (humans as sensors; aggregated opinions)**
  - **Business intelligence**
  - **Market** research
  - Data-driven **social science research**
  - Gain advantage in **any prediction** (text-based prediction)

# Opinion Mining and Sentiment Analysis: Sentiment Classification

# Sentiment Classification: Task Definition

- Input: An **opinionated text** object
- Output: A **sentiment tag**/label
  - **Polarity analysis**: e.g., categories = {positive, negative, neutral}, or categories ={5, 4, 3, 2, 1}
  - **Emotion analysis** (beyond polarity): e.g., categories ={happy, sad, fearful, angry, surprised, disgusted}
- A **special case of text categorization**! ➔ Any text categorization method can be used to do sentiment classification
- Further improvement comes from
  - **More sophisticated features** appropriate for sentiment tagging
  - Consideration of the **order of the categories** (e.g., ordinal regression)

# Commonly Used Text Features

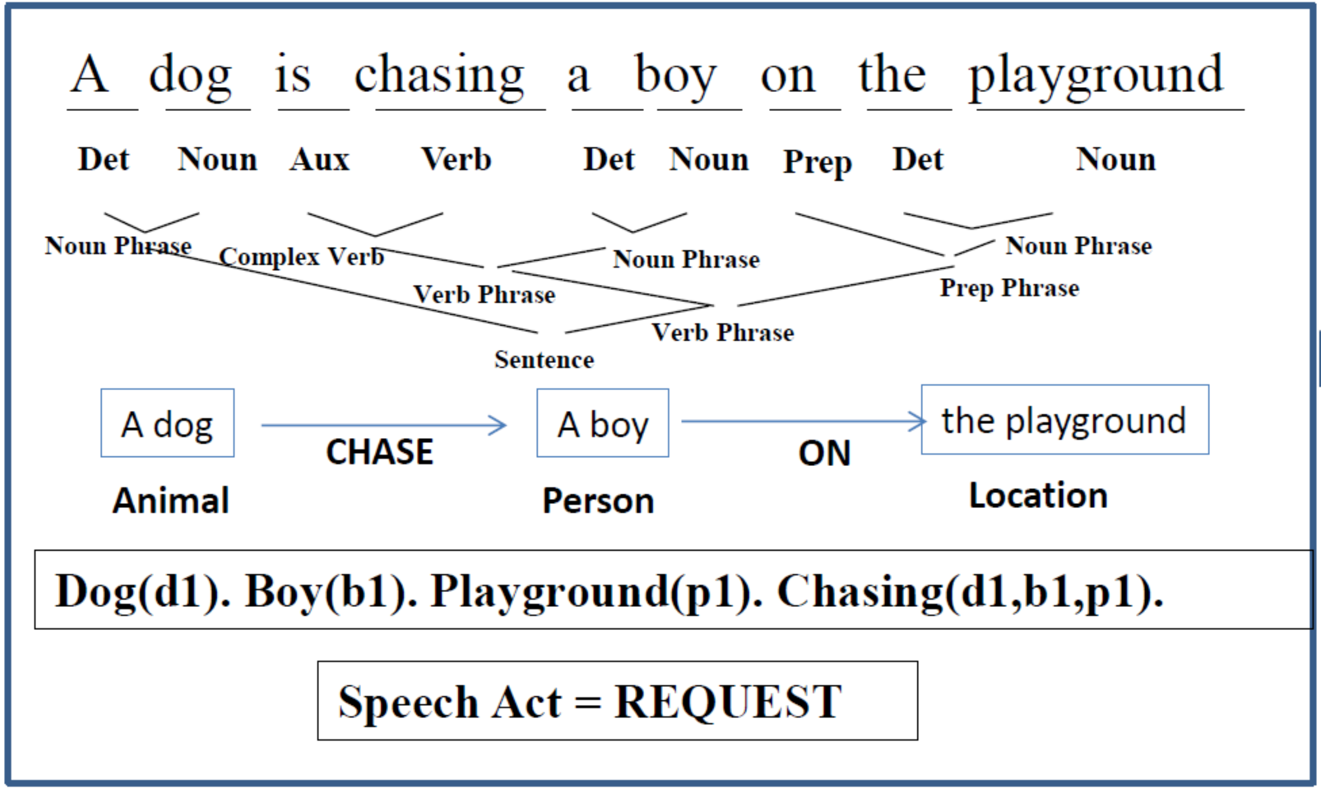
- **Character n-grams**: can be mixed with different n's
  - General and robust to spelling/recognition errors, but less discriminative than words
- **Word n-grams**: can be mixed with different n's
  - Unigrams are often very effective, but not for sentiment analysis (e.g. , "it's not good"  or "it's not as good as")
  - Long n-grams are discriminative, but may cause overfitting
- **POS tag n-grams**: mixed word n-gram and POS tags
  - E.g., "ADJECTIVE NOUN" or "great NOUN"

# Commonly Used Text Features (cont.)

- **Word classes**
  - Syntactic (= POS tags)
  - Semantic Concept: e.g., thesaurus/ontology, recognized entities
  - Empirical word clusters (e.g., cluster of paradigmatically or syntagmatically related words)
- **Frequent patterns** in text (e.g., frequent word set; collocations)
  - More specific/discriminative than words
  - May generalize better than pure n-grams
- **Parse tree-based** (e.g., frequent subtrees, paths)
  - Even more discriminative, but need to avoid overfitting
- **Pattern discovery** algorithms are very useful **for feature construction**

# NLP Enriches Text Representation with Complex Features

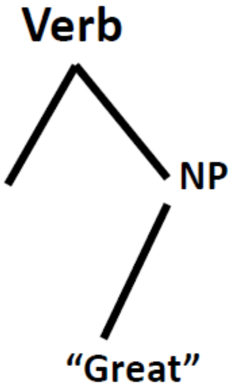
A dog is chasing a boy on the playground




"great NOUN"

"Verb Adv Adj"

...



# Feature Construction for Text Categorization

- **Feature design** affects categorization accuracy significantly
- A **combination** of machine learning, error analysis, and domain knowledge is **most effective**
  - **Domain knowledge** → seed features, feature space
  - **Machine learning** → feature selection, feature learning
  - **Error analysis** → feature validation
- **NLP enriches** text representation ➔ enriches feature space (more likely overfitting!)
- Optimizing the tradeoff between **exhaustivity** and **specificity** is a major goal

high coverage (frequent)

discriminative (infrequent)

# Sentiment Analysis:
# Ordinal Logistic Regression

# Motivation: Rating Prediction

- Input: An **opinionated text** document **d**
- Output: **Discrete rating r** $\in$**{1, 2, …, k}**
- Using **regular text categorization** techniques
  - Doesn't consider the order and dependency of the categories
  - The features distinguishing r=2 from r=1 may be the same as those distinguishing r=k from r=k-1 (e.g., positive words = higher rating)
- Solution: **Add order to a classifier** (e.g., ordinal logistic regression )

# Logistic Regression for Binary Sentiment Classification

**Binary Response Variable: Y $\in$ {0,1}** **Predictors:** $X = (x_1, x_2, \dots, x_M),\ x_i \in \Re$

$$Y = \begin{cases} 1 & \text{X is POSITIVE} \\ 0 & \text{X is NEGATIVE} \end{cases}$$

$$\log \frac{p(Y=1|X)}{p(Y=0|X)} = \log \frac{p(Y=1|X)}{1-p(Y=1|X)} = \beta_0 + \sum_{i=1}^{M} x_i \beta_i \quad \beta_i \in \Re$$

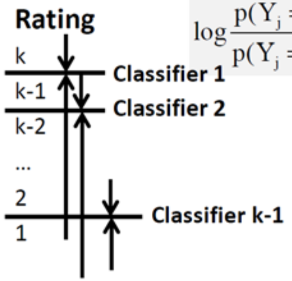
$$p(Y=1|X) = \frac{e^{\beta_0 + \sum_{i=1}^{M} x_i \beta_i}}{e^{\beta_0 + \sum_{i=1}^{M} x_i \beta_i} + 1}$$

# Logistic Regression for Multi-Level Ratings

$$Y_j = \begin{cases} 1 & \text{rating is j or above} \\ 0 & \text{rating is lower than j} \end{cases}$$

**Predictors:** $X = (x_1, x_2, \dots, x_M),\ x_i \in \Re$

**Rating:** **r$\in${1, 2, …, k}**




$$\log \frac{p(Y_j=1|X)}{p(Y_j=0|X)} = \log \frac{p(r \geq j|X)}{1-p(r \geq j|X)} = \alpha_j + \sum_{i=1}^{M} x_i \beta_{ji} \quad \beta_{ji} \in \Re$$

$$p(r \geq j|X) = \frac{e^{\alpha_j + \sum_{i=1}^{M} x_i \beta_{ji}}}{e^{\alpha_j + \sum_{i=1}^{M} x_i \beta_{ji}} + 1}$$



# Rating Prediction with Multiple Logistic Regression Classifiers

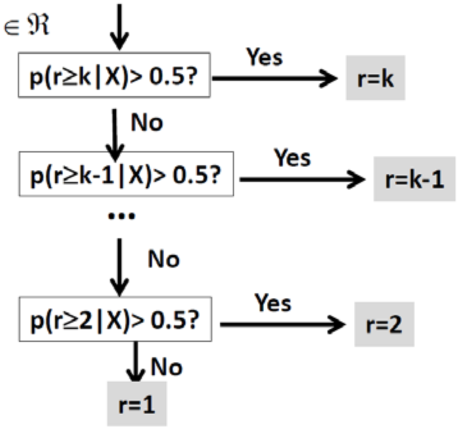
**Text Object:** $X = (x_1, x_2, \dots, x_M),\ x_i \in \Re$

**Rating:** **r$\in${1, 2, …, k}**

**After training k-1 Logistic Regression Classifiers**

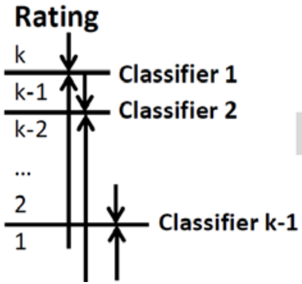
$$p(r \geq j|X) = \frac{e^{\alpha_j + \sum_{i=1}^{M} x_i \beta_{ji}}}{e^{\alpha_j + \sum_{i=1}^{M} x_i \beta_{ji}} + 1}$$

**j=k, k-1, …, 2**



# Problems with k-1 Independent Classifiers?

$$\log\frac{p(Y_j=1|X)}{p(Y_j=0|X)} = \log\frac{p(r\ge j|X)}{1-p(r\ge j|X)} = \alpha_j + \sum_{i=1}^{M} x_i\beta_{ji} \quad \beta_{ji}\in\Re$$




$$p(r\ge j|X) = \frac{e^{\alpha_j+\sum_{i=1}^{M}x_i\beta_{ji}}}{e^{\alpha_j+\sum_{i=1}^{M}x_i\beta_{ji}}+1}$$

How many parameters are there in total? (k-1)*(M+1)

The k-1 classification problems are dependent.
The positive/negative features tend to be similar!

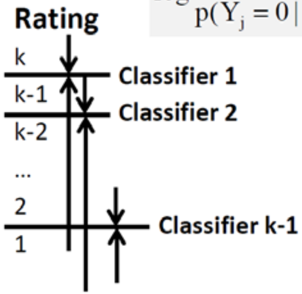


# Ordinal Logistic Regression

**Key Idea: $\forall i = 1, \ldots, M$, $\forall j = 3, \ldots, k$, $\beta_{ji} = \beta_{j-1i}$**

➔ Share training data ➔ Reduce # of parameters

$$\log\frac{p(Y_j=1|X)}{p(Y_j=0|X)} = \log\frac{p(r\ge j|X)}{1-p(r\ge j|X)} = \alpha_j + \sum_{i=1}^{M} x_i\beta_i \quad \beta_i\in\Re$$




$$p(r\ge j|X) = \frac{e^{\alpha_j+\sum_{i=1}^{M}x_i\beta_i}}{e^{\alpha_j+\sum_{i=1}^{M}x_i\beta_i}+1}$$

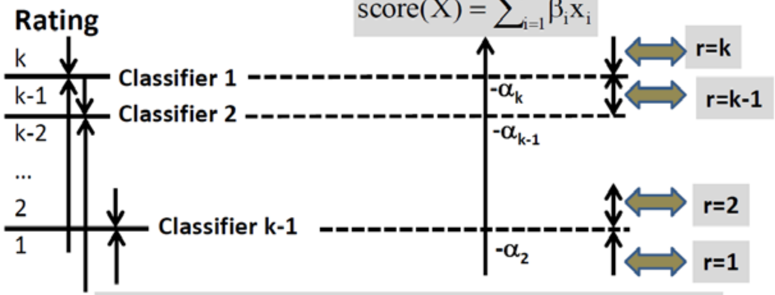
How many parameters are there in total? M+k-1



# Ordinal Logistic Regression: Rating Prediction

$$p(r\ge j|X)\ge 0.5 \Leftrightarrow \frac{e^{\alpha_j+\text{score}(X)}}{e^{\alpha_j+\text{score}(X)}+1}\ge 0.5 \Leftrightarrow \text{score}(X)\ge -\alpha_j$$

$$\text{score}(X) = \sum_{i=1}^{M}\beta_i x_i$$




**r=j ⇔ score $\in[-\alpha_j, -\alpha_{j+1})$, define $\alpha_1=\infty$, $\alpha_{k+1}=-\infty$**