Advanced Studies – Machine Learning and NLP (Research and Applications)

Dr. Kevin Koidl
School of Computer Science and Statistic Trinity College Dublin
ADAPT Research Centre
Research Fellow Trinity College Dublin and ADAPT research centre. **Lecturing** focus on ML/NLP. Student supervision in ML/NLP related areas.

Main **research** areas are related to **ML/NLP** usage in Web in Personalisation and Recommendation Systems. Other research areas are Data Awareness (bigfoot.ie) and e-Learning (edtech).

Focus on conceptualisation of **Cognitive ‘Intelligent’ Content** -> multi-dimensional, **context-aware** content.

**Research Commercialisation** in ML/NLP Trinity College Dublin Spin-out (wripl.com) and International Crowed Sourced Short Literature Translation Project (Kaffeehouse.com).

**Social Enterpreneurship Advisor** Enactus Global and Startup/Research Commercialisation Advisor Ireland.

Speaker on Privacy, Data-awareness and AI impact on society.
Lecture Topics and Dates

Lecturing Dates: 6/3, 13/3, 20/3, 27/3
Time: 2 p.m. – 4 p.m.
Lab: 20/3 and 27/3

6/3 – Introduction, Overview and basic concepts NLP/ML
13/3 – Refresher and Advanced NLP/ML Concepts
20/3 – Application Areas I and Lab
27/3 – Application Areas II and Lab

Classes are partially based on course given in TCD
https://www.scss.tcd.ie/kevin.koidl/cs4062/
An (artificial) agent may help users cope with increasing information:

An agent is a computer system that is situated in some environment and that is capable of autonomous action in its environment in order to meet its design objectives. [Wooldridge, 2002]

Definition of a Rational Agent:

A rational agent should select an action that is expected to maximize its performance measure, given the evidence provided. [Peter Norvig, 2003]
Learning

- ML has been studied from various perspectives (AI, control theory, statistics, information theory, ...)
- From an AI perspective, the general definition is formulated in terms of agents and tasks. E.g.:

  [An agent] is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with $E$.

  [Mitchell, 1997, p. 2]

- Statistics, model-fitting, ...
Design Implications

- **Agent** = *architecture + program*
- Architecture = Sensors and Actuators
- Program = Decision Process
- Examples are: Simple Reflex Agents, Model-based reflex Agents, Goal-based Agents, Utility-based Agents and Learning (Intelligent) Agents.

- Main design decisions:
  - Training experience: How will the system access and use data?
  - Target function: What exactly should be learnt?
  - Hypothesis representation: How will we represent the concepts to be learnt?
  - Inductive inference: What specific algorithm should be used to learn the target concepts?
A Learning Agent

- **Performance element**: responsible for selecting appropriate actions
- **Learning element**: responsible for making improvements
- **Critic**: evaluates action selection against a performance standard
- **Problem generator**: suggests actions that might lead to new and instructive experiences
Main Machine Learning Categories:

- Supervised Learning: Computer receives input and output data aka 'labelled' data and creates a 'mapping' between both.

- Unsupervised Learning: Input data has no labels are given. Learning algorithm has to identify structure in the input data.

Supervised Machine Learning Problem Categories:

- Regression Problem: Continuous Output. For example predict percentage grade (e.g. 76%) based on hours studied.

- Classification Problem: Discrete Output. For example predict grade (e.g. A) based on hours studied.
The target function specifies the concept to be learnt.

In supervised learning, the target function is assumed to be specified through annotation of training data or some form of feedback:

- a corpus of words annotated for word senses, e.g. $f : W \times S \rightarrow \{0, 1\}$
- a database of medical data
- user feedback in spam filtering
- assessment of outcomes of actions by a situated agent

Induction task as search for a hypothesis (or model) that fits the data and sample of the target function available to the learner, in a large space of hypotheses.

The choice of learning algorithm is conditioned to the choice of representation.

Since the target function is not completely accessible to the learner, the algorithm needs to operate under the inductive learning assumption that:

\[
\text{an approximation that performs well over a sufficiently large set of instances will perform well on unseen data.}
\]
Example 1

Customer purchase behavior:

<table>
<thead>
<tr>
<th>Customer103: (time=t0)</th>
<th>Customer103: (time=t1)</th>
<th>...</th>
<th>Customer103: (time=tn)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex: M</td>
<td>Sex: M</td>
<td></td>
<td>Sex: M</td>
</tr>
<tr>
<td>Age: 53</td>
<td>Age: 53</td>
<td></td>
<td>Age: 53</td>
</tr>
<tr>
<td>Income: $50k</td>
<td>Income: $50k</td>
<td></td>
<td>Income: $50k</td>
</tr>
<tr>
<td>Own House: Yes</td>
<td>Own House: Yes</td>
<td></td>
<td>Own House: Yes</td>
</tr>
<tr>
<td>MS Products: Word</td>
<td>MS Products: Word</td>
<td></td>
<td>MS Products: Word</td>
</tr>
<tr>
<td>Computer: 386 PC</td>
<td>Computer: Pentium</td>
<td></td>
<td>Computer: Pentium</td>
</tr>
<tr>
<td>Purchase Excel?: ?</td>
<td>Purchase Excel?: ?</td>
<td></td>
<td>Purchase Excel?: Yes</td>
</tr>
</tbody>
</table>

Process optimization:

<table>
<thead>
<tr>
<th>Product72: (time=t0)</th>
<th>Product72: (time=t1)</th>
<th>...</th>
<th>Product72: (time=tn)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage: mix</td>
<td>Stage: cook</td>
<td></td>
<td>Stage: cool</td>
</tr>
<tr>
<td>Mixing-speed: 60rpm</td>
<td>Temperature: 325</td>
<td></td>
<td>Fan-speed: medium</td>
</tr>
<tr>
<td>Viscosity: 1.3</td>
<td>Viscosity: 3.2</td>
<td></td>
<td>Viscosity: 1.3</td>
</tr>
<tr>
<td>Fat content: 15%</td>
<td>Fat content: 12%</td>
<td></td>
<td>Fat content: 12%</td>
</tr>
<tr>
<td>Density: 2.8</td>
<td>Density: 1.1</td>
<td></td>
<td>Density: 1.2</td>
</tr>
<tr>
<td>Spectral peak: 2600</td>
<td>Spectral peak: 3200</td>
<td></td>
<td>Spectral peak: 3100</td>
</tr>
<tr>
<td>Product underweight?: ?</td>
<td>Product underweight?: ?</td>
<td></td>
<td>Product underweight?: Yes</td>
</tr>
</tbody>
</table>
Example 2

- **ALVINN [Pomerleau, 1994]:** drives 70 mph
Example 3

- A target function for a draughts (checkers) player:
  - $f : \text{Board} \rightarrow \mathbb{R}$
  - if $b$ is a final board state that is won, then $f(b) = 100$
  - if $b$ is a final board state that is lost, then $f(b) = -100$
  - if $b$ is a final board state that is drawn, then $f(b) = 0$
  - if $b$ is a not a final state in the game, then $f(b) = f(b')$, where $b'$ is the best final board state that can be achieved starting from $b$ and playing optimally until the end of the game.

- How feasible would it be to implement it?
  - Not very feasible...

- ... and how can we find intermediate game states?
Target Function Approximation

- collection of rules? neural network? polynomial function of board features? ...
- Approximation as a linear combination of features:

$$\hat{f}(b) = w_0 + w_1 \cdot bp(b) + w_2 \cdot rp(b) + w_3 \cdot bk(b) + w_4 \cdot rk(b) + w_5 \cdot bt(b) + w_6 \cdot rt(b)$$

- where:
  - $bp(b)$: number of black pieces on board $b$
  - $rp(b)$: number of red pieces on $b$
  - $bk(b)$: number of black kings on $b$
  - $rk(b)$: number of red kings on $b$
  - $bt(b)$: number of red pieces threatened by black (i.e., which can be taken on black's next turn)
  - $rt(b)$: number of black pieces threatened by red
Estimation of true target function

- Distinctions:
  - \( f(b) \): the true target function
  - \( \hat{f}(b) \): the learnt function
  - \( f_{\text{train}}(b) \): the training value
  - A training set containing instances and its corresponding training values

- Problem: How do we estimate training values?
- A simple rule for estimating training values:
  - \( f_{\text{train}}(b) \leftarrow \hat{f}(\text{Successor}(b)) \)
  - Successor\((b)\) denotes the next board state following the programs move and the opponent's response.
  - Note: (Successor\((b)\) is an 'estimation' of the value of board state \(b\).)
  - Does the \( \hat{f}(b) \) tend to become more or less accurate for board states closer to the end of the game?
[An agent] is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with $E$.

[Mitchell, 1997]

- Notation (based on [Sebastiani, 2002]):
  - $D$: a set (domain) of documents
    $$D = \{d_1, ..., d_{|D|}\}$$
  - $C$: a set of categories
    $$C = \{c_1, ..., c_{|C|}\}$$

- Text categorisation is the task of assigning a boolean value to each pair $<d_i, c_j>$, s.t.:

\[
\begin{align*}
  <d_i, c_j> &= T & \text{if } d_i \text{ is filed under } c_j \\
  <d_i, c_j> &= F & \text{if } d_i \text{ is NOT filed under } c_j
\end{align*}
\]  

(1)
The goal of TC is to approximate the (unknown) target function:

$$f : D \times C \rightarrow \{T, F\}$$

which classifies (as defined in (1)) each document correctly into one or more categories in $C$.

Classification is (actually) done by a classifier function

$$\hat{f} : D \times C \rightarrow \{T, F\}$$

For evaluation purposes, a target function could be assumed to describe, for instance, a set of hand-annotated documents.
Evaluation of Performance

<table>
<thead>
<tr>
<th>selected</th>
<th>correct</th>
<th>not correct</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>tp</td>
<td>fp</td>
</tr>
<tr>
<td>not selected</td>
<td>fn</td>
<td>tn</td>
</tr>
</tbody>
</table>

- Evaluation compares actual performance ($\hat{f}$) to ideal performance ($f$).
- The most commonly used metrics:
- Recall: how good the system is at "finding" relevant documents for a given category ($\rho$):
  \[
  \rho = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}
  \]
  \[
  (2)
  \]
- Precision: the "quality" of the classified data ($\pi$):
  \[
  \pi = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}
  \]
  \[
  (3)
  \]
A categorisation function $\hat{f}: D \times C \rightarrow \{T, F\}$ performs what we call “hard” categorisation.

Another possibility is, “soft” categorisation, i.e. to allow $\hat{f}$ to range over real values in the interval $[0, 1]$ (i.e. $\hat{f}(d, c) = \nu$, where $0 \leq \nu \leq 1$).

This is equivalent to a ranking of the categories of $C$ according to their appropriateness to each document.

Ranking CPC is also possible.

In supervised TC, the corpus available to the developers is annotated, that is, it is a set $\Omega = \{d_1, ..., d_{|\Omega|}\} \subset D$, such that a target function $f$ is known.

In creating a classifier, $\Omega$ is normally divided into three sub-sets:

- a training set ($Tr$), based on which the initial classifier is inductively built.
- a validation set ($Tv$), also known as hold-out set, on which the initial classifier is optimised, and
- a test set ($Te$), on which the classifier is evaluated.
For example, the TC system to be implemented in this course:

Unannotated corpus

Annotated corpus, $\mathcal{D}$ \rightarrow Target function: $f : \mathcal{D} \times \mathcal{C} \rightarrow [0, 1]$

Trainning experience:

Stream of documents:
- Document browsing

$f : \mathcal{D} \rightarrow \mathcal{C}$

$\{\text{set of words} \rightarrow \ldots, \text{bag of words} \rightarrow \{\text{NB, NN, ID3, SVM, KNN, ...}\}, \text{prob. vectors} \rightarrow \ldots, \text{feature sets} \rightarrow \ldots\}$

Repres.
A 3d term set:
\[ \mathcal{T} = \{football, politics, economy\} \]

IR: calculate distances between vectors (e.g. via cosine matching)

TC: High dimensionality may be problematic
- Document 1: 'Economy and Politics is important for Football.'
- Document 2: 'In Economy the Politics of Football counts.'
- Classification towards three labels 'Football, Economy, Politics'
- Typical approach: Bag of Words and Frequency.
- Resulting Vectors for D1 and d2: (1,1,1) and (1,1,1)
- Trivial Example
Document 1 and 2 are longer documents with (5,5,3) and (5,3,3) in keyword frequency.

How do we compare both Vectors?

Normalisation: Unit Vector (on the surface of the unit hypersphere)

\[
\vec{u} = \frac{\vec{v}}{\|\vec{v}\|}
\]

\[
\|\vec{v}\| = \sqrt{5^2 + 5^2 + 3^2}, \left(\frac{5}{7.68}, \frac{5}{7.68}, \frac{3}{7.68}\right)
\]
Example Author Classification

Three novels. SaS: Sense and Sensibility, PaP: Pride and Prejudice, WH: Wuthering Heights?

<table>
<thead>
<tr>
<th>Term</th>
<th>SaS</th>
<th>PaP</th>
<th>WH</th>
</tr>
</thead>
<tbody>
<tr>
<td>affection</td>
<td>115</td>
<td>58</td>
<td>20</td>
</tr>
<tr>
<td>jealous</td>
<td>10</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>gossip</td>
<td>2</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>wuthering</td>
<td>0</td>
<td>0</td>
<td>38</td>
</tr>
</tbody>
</table>

Log Frequency = 1 + log (term frequency)

<table>
<thead>
<tr>
<th>Term</th>
<th>SaS</th>
<th>PaP</th>
<th>WH</th>
</tr>
</thead>
<tbody>
<tr>
<td>affection</td>
<td>3.06</td>
<td>2.76</td>
<td>2.30</td>
</tr>
<tr>
<td>jealous</td>
<td>2.00</td>
<td>1.85</td>
<td>2.04</td>
</tr>
<tr>
<td>gossip</td>
<td>1.30</td>
<td>0</td>
<td>1.78</td>
</tr>
<tr>
<td>wuthering</td>
<td>0</td>
<td>0</td>
<td>2.58</td>
</tr>
</tbody>
</table>

Length Normalized Values

<table>
<thead>
<tr>
<th>Term</th>
<th>SaS</th>
<th>PaP</th>
<th>WH</th>
</tr>
</thead>
<tbody>
<tr>
<td>affection</td>
<td>0.789</td>
<td>0.832</td>
<td>0.524</td>
</tr>
<tr>
<td>jealous</td>
<td>0.515</td>
<td>0.555</td>
<td>0.465</td>
</tr>
<tr>
<td>gossip</td>
<td>0.335</td>
<td>0</td>
<td>0.405</td>
</tr>
<tr>
<td>wuthering</td>
<td>0</td>
<td>0</td>
<td>0.588</td>
</tr>
</tbody>
</table>
Example Author Classification

\[ \cos(d, e) = \frac{d \cdot e}{\|d\| \times \|e\|} \]

\[ \text{dist}(d, e) = \sqrt{\sum_{i=1}^{\|T\|} (d_i - e_i)^2} \] (4)

\[ \cos(\text{SaS, PaP}) \approx 0.789 \times 0.832 + 0.515 \times 0.555 + 0.335 \times 0.0 + 0.0 \times 0.0 \approx 0.94 \]

\[ \cos(\text{SaS, WH}) \approx 0.79 \]

\[ \cos(\text{PaP, WH}) \approx 0.69 \]
Dimensionality Reduction

- DR: a processing step whose goal is to reduce the size of the vector space from $|\mathcal{T}|$ to $|\mathcal{T}'| \ll |\mathcal{T}|$.
- $\mathcal{T}'$ is called Reduced Term Set
- Benefits of DR:
  - Lower computational cost for ML
  - Help avoid overfitting (training on constitutive features, rather than contingent ones)
- A rule-of-thumb: overfitting is avoided if the number of training examples is proportional to the size of $\mathcal{T}'$ (For TC, experiments have suggested a ratio of 50-100 texts per feature).
# Entropy and Information Gain

<table>
<thead>
<tr>
<th>Function</th>
<th>Notation</th>
<th>Mathematical definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DIA factor</strong></td>
<td>( z(T_k, C_i) )</td>
<td>( P(c_i</td>
</tr>
<tr>
<td><strong>Information Gain, AKA</strong></td>
<td>( IG(T_k, C_i) )</td>
<td>( \sum_{c \in {c_i, \overline{c_i}}} \sum_{t \in {t_k, \overline{t_k}}} P(t, c) \log \frac{P(t, c)}{P(t)P(c)} )</td>
</tr>
<tr>
<td><strong>Expected Mutual Information</strong></td>
<td>( I(T_k; C_i) )</td>
<td>( \sum_{c \in {c_i, \overline{c_i}}} \sum_{t \in {t_k, \overline{t_k}}} P(t, c) \log \frac{P(t, c)}{P(t)P(c)} )</td>
</tr>
<tr>
<td><strong>Mutual information</strong></td>
<td>( MI(T_k, C_i) )</td>
<td>( P(t_k, c_i) \log \frac{P(t_k, c_i)}{P(t_k)P(c_i)} )</td>
</tr>
<tr>
<td><strong>Chi-square</strong></td>
<td>( \chi^2(T_k, C_i) )</td>
<td>(</td>
</tr>
<tr>
<td><strong>NLG coefficient</strong></td>
<td>( NGL(T_k, C_i) )</td>
<td>( \sqrt{</td>
</tr>
<tr>
<td><strong>Relevancy score</strong></td>
<td>( RS(T_k, C_i) )</td>
<td>( \log \frac{P(t_k</td>
</tr>
<tr>
<td><strong>Odds ratio</strong></td>
<td>( OR(t_k, c_i) )</td>
<td>( \frac{P(t_k</td>
</tr>
<tr>
<td><strong>GSS coefficient</strong></td>
<td>( GSS(T_k, C_i) )</td>
<td>( P(t_k, c_i)P(\overline{t_k}, \overline{c_i}) - P(t_k, \overline{c_i})P(\overline{t_k}, c_i) )</td>
</tr>
</tbody>
</table>

from [Sebastiani, 2002]
Entropy, AKA self-information, measures the average amount of uncertainty in a probability mass distribution. In other words, entropy is a measure of how much we learn when we observe an event occurring in accordance with that distribution.

The entropy of a random variable measures the amount of information in that variable (we will always be using \( \log \) base 2 unless stated otherwise):

\[
H(X) = H(p) = - \sum_{x \in X} p(x) \log p(x)
\]

\[
= \sum_{x \in X} p(x) \log \frac{1}{p(x)}
\]

N.B.: We define \( 0 \log(0) = 0 \)
Information Gain

- We may also quantify the reduction of uncertainty of a random variable due to knowing about another. This is known as Expected Mutual Information:

\[ I(X; Y) = IG(X, Y) = \sum_{x,y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \]  

(9)

- Entropies of different probability functions may also be compared by calculating the so called Information Gain. In decision tree learning, for instance:

\[ G(D, F) = H(t) - \sum_{i=1}^{n} p_i H(t_i) \]  

(10)

where \( t \) is the distribution of the mother, \( t_i \) the distribution of daughter node \( i \), and \( p_i \) the proportion of texts passed to node \( i \) if term \( F \) is used to split corpus \( D \).
Information Gain

- **Basic intuition:** the *best features* for a category are those *distributed most differently* on sets of positive and negative instances of documents filed under that category.

- **Pointwise mutual information:**

\[
PMI(T_i, C_j) = \log \frac{P(t_i, c_j)}{P(t_i)P(c_j)}
\]  

(7)

- Calculations to be performed: *co-occurrence* of terms and categories in the training corpus (*Tr*), and *frequency of occurrence* of words and categories in *Tr*. 
Given an email classification system

Consider Document D whose Class is given by C

For a Spam filter there are two classes C = S (SPAM) and C = H (HAM)

We classify D as the class with the highest posterior probability \( P(C \mid D) \) expressed by the Bayes Theorem:

\[
P(C \mid D) = \frac{P(D \mid C)P(C)}{P(D)}
\]

Documents are usually represented as probabilistic models and represented as bag of words.

Models use feature vectors whose components correspond to word vectors.

For a Vocabulary V containing \(|V|\) word types the feature vector has the dimension \(d = |V|\)
Bernoulli Document Model

A document is represented by a feature vector with binary elements taking value 1 if the corresponding word is present in the document and 0 if the word is not present.

Multinomial Document Model

A document is presented by a feature vector with integer elements whose value is the frequency of that word in the document.

Example based on following Vocabulary \( V = \{ \text{blue, red, dog, cat, biscuit, apple} \} \) hence \( |V| = d = 6 \).

A document for example “the blue dog ate a blue biscuit’ has following model representations

\[
\begin{align*}
    d(\text{Bernoulli}) &= (1, 0, 1, 0, 1, 0) \\
    d(\text{Multinomial}) &= (2, 0, 1, 0, 1, 0)
\end{align*}
\]
The Bernoulli model represents a document via a binary vector (a point in the space of words -> Vector Space).

A vocabulary $V$ contains a set of $|V|$ words. In this case dimension $t$ of the document vector corresponds with word $t$ in the vocabulary.

For example: If $b$ is the feature vector for document $d$ (out of a set of documents) then element $t$ out of $b$ is either 0 or 1 representing the absence or presence of word $w$ in the document (also represented as $b_{it}$ being either 0 or one with $b_i$ being the feature vector for the $i$th document $D_i$)

Further, $P(w_t|C)$ is the probability of word $w_t$ occurring in a document of class $C$.

Probability of $w_t$ not occurring in a document of this class is given by $(1 - P(w_t|C))$

Naïve assumption: The probability of each word occurring in the document is independent of the occurrences of the other words.
Naïve assumption: The probability of each word occurring in the document is independent of the occurrences of the other words. This means $P(D_i|C)$ can be calculated in terms of individual word likelihoods $P(w_t|C)$:

$$
P(D_i|C) \sim P(b_i|C) = \prod_{t=1}^{|V|} \left[ b_{it} P(w_t|C) + (1 - b_{it})(1 - P(w_t|C)) \right]
$$

This product runs over every word in the vocabulary. If word $w_t$ is present then $b_{it}$ results in 1 and the probability is $P(w_t|C)$ and visa versa for word not present.

This can be viewed as a model that creates feature vectors modelled on $|V|$ and based on weighted coin tosses with $P(w_t|C)$ for each word.
Steps for calculation

1. Define the vocabulary $V$; the number of words in the vocabulary defines the dimensions of the feature vector.

2. Count the following in the training set:
   - $N$ the total number of documents
   - $N_k$ the number of documents labelled with class $C = k$, for $k=1,\ldots,K$
   - $n_k(w_t)$ the number of documents of class $C=k$ containing words $w_t$ for every class and for each word in the vocabulary.

3. Estimate the likelihoods $P(w_t | C = k)$

4. Estimate the priors $P(C = k)$

To classify an unknown document $D_j$ we estimate the posterior probability for each class

$$P(C | D_j) = P(C | b_j) = P(b_j | C) P(C) = P(C) \prod_{t=1}^{V} \left[ b_{it} P(w_t | C) + (1 - b_{it})(1 - P(w_t | C)) \right]$$
Example

Given two classes Sports (S) and Informatics (I) and a training set of 11 documents. Classify unknown document as NB estimation using Bernoulli Document Model.

Define vocabulary $V = \{\text{goal, tutor, variance, speed, drink, defence, performance, field}\}$

Each document is a 8 dimensional binary vector (row = document)

<table>
<thead>
<tr>
<th>B(Sport)</th>
<th>B(Inf)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 0 0 1 1 1 1</td>
<td>0 1 1 0 0 0 1 0</td>
</tr>
<tr>
<td>0 0 1 0 1 1 0 0</td>
<td>1 1 0 1 0 0 1 1</td>
</tr>
<tr>
<td>0 1 0 1 0 1 1 0</td>
<td>0 1 1 0 0 1 0 0</td>
</tr>
<tr>
<td>1 0 0 1 0 1 0 1</td>
<td>0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>1 0 0 0 1 0 1 1</td>
<td>0 0 1 0 1 0 1 0</td>
</tr>
<tr>
<td>0 0 1 1 0 0 0 1 1</td>
<td></td>
</tr>
</tbody>
</table>
1. Define vocabulary

2. Count total number of documents in the training set $N = 11$; $N(S) = 6$; $N(I) = 5$ and estimate the prior probabilities $P(S) = 6/11$ and $P(I) = 5/11$

Word count in the training data for sport: $n(w_1) = 3$, $n(w_2) = 1$, $n(w_3) = 2$, $n(w_4) = 3$, $n(w_5) = 3$, $n(w_6) = 4$, $n(w_7) = 4$, $n(w_8) = 4$

Word count in the training data for informatics: $n(w_1) = 1$, $n(w_2) = 3$, $n(w_3) = 3$, $n(w_4) = 1$, $n(w_5) = 1$, $n(w_6) = 1$, $n(w_7) = 3$, $n(w_8) = 1$

3. Estimate likelihoods ($P(w_t | C = k) \rightarrow$ number of words in class / total number of documents of that class)

Sport: $P(w_1|S) = \frac{3}{6} = \frac{1}{2}$; $P(w_2|S) = \frac{1}{6}$; $\frac{1}{3}$; $\frac{1}{2}$; $\frac{2}{3}$; $\frac{2}{3}$; $\frac{2}{3}$

Inf: $\frac{1}{5}$; $\frac{3}{5}$; $\frac{3}{5}$; $\frac{1}{5}$; $\frac{1}{5}$; $\frac{1}{5}$; $\frac{3}{5}$; $\frac{1}{5}$
Calculation of posterior probabilities

\[ B_1 = (1,0,0,1,1,1,0,1) \rightarrow P(S) \prod_{t=1}^{V} [ b_{it} P(w_t|S) + (1 - b_{it})(1 - P(w_t|S)) ] \]

\[ P(S|b_1) = \frac{6}{11} \left( \frac{1}{2} \times \frac{5}{6} \times \frac{2}{3} \times \frac{1}{2} \times \frac{1}{2} \times \frac{2}{3} \times \frac{1}{3} \times \frac{2}{3} \right) = \frac{5}{891} = 0.0056\ldots \]

\[ P(I|b_1) = \frac{5}{11} \left( \frac{1}{5} \times \frac{2}{5} \times \frac{2}{5} \times \frac{1}{5} \times \frac{1}{5} \times \frac{1}{5} \times \frac{2}{5} \times \frac{1}{5} \right) = \frac{8}{859375} = 0.00000\ldots \]

Classify Document as S.

\[ P(S|b_2) = \ldots = \frac{12}{14256} = 0.00084\ldots \]

\[ P(I|b_2) = \ldots = \frac{34560}{4296875} = 0.008\ldots \]

Classify Document as I.
Takes frequency of words into account and is not affected by words that do not occur in the document. However, zero counts result in Zero-Probability Problem needs to be addressed by using ‘add one smoothing’.

Bernoulli distribution – probability of a word being present in the model can be seen as weighted coin flip with the probability $P(w_t | C)$

Multinomial distribution – finding the next word in a document by rolling a $|V|$ sided dice with the probability $P(w_t | C)$

Bernoulli has a better performance with short documents than Multinomial
Both are ’okay’ with long documents.
You are a company that is selling books. You want to use advanced NLP/ML to boost your SEO (Search Engine Optimization) and make your books easier to find.

a) Why is this challenge related to NLP/ML?

b) What is the fundamental research area that SEO underlies?

c) List the main challenges you see in boosting SEO with NLP/ML. What data do you need and where can you get it?