Take-away today

- Text classification: definition & relevance to information retrieval
- Naive Bayes: simple baseline text classifier
- Theory: derivation of Naive Bayes classification rule & analysis
- Evaluation of text classification: how do we know it worked / didn’t work?
Outline

1 Recap

2 Text classification

3 Naive Bayes

4 NB theory

5 Evaluation of TC
A text classification task: Email spam filtering

From: ‘‘’ <takworllld@hotmail.com>
Subject: real estate is the only way... gem oalvgkay
Anyone can buy real estate with no money down
Stop paying rent TODAY!
There is no need to spend hundreds or even thousands for similar courses
I am 22 years old and I have already purchased 6 properties using the
methods outlined in this truly INCREDIBLE ebook.
Change your life NOW!
=================================================================
Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm
=================================================================

How would you write a program that would automatically detect
and delete this type of message?
Formal definition of TC: Training

Given:

- A **document space** $\mathbb{X}$
  - Documents are represented in this space – typically some type of high-dimensional space.

- A fixed set of **classes** $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$
  - The classes are human-defined for the needs of an application (e.g., spam vs. nonspam).

- A **training set** $\mathbb{D}$ of labeled documents. Each labeled document $\langle d, c \rangle \in \mathbb{X} \times \mathbb{C}$

Using a learning method or **learning algorithm**, we then wish to learn a **classifier** $\gamma$ that maps documents to classes:

$$\gamma : \mathbb{X} \rightarrow \mathbb{C}$$
Formal definition of TC: Application/Testing

Given: a description $d \in X$ of a document  
Determine: $\gamma(d) \in C$, that is, the class that is most appropriate for $d$
Topic classification

classes:

REGIONS

UK
China

INDUSTRIES

poultry
coffee

SUBJECT AREAS

elections
sports

regions industries subject areas

γ(d′) = China

training set:

CONGESTION
LONDON

OLYMPICS
BEIJING

FEED
CHICKEN

ROASTING
BEANS

RECOUNT
VOTES

DIAMOND
BASEBALL

PARLIAMENT
BIG BEN

TOURISM
GREAT WALL

PATE
DUCKS

ARABICA
ROBUSTA

SEAT
RUN-OFF

FORWARD
SOCCER

WINDSOR
THE QUEEN

MAO
COMMUNIST

BIRD FLU
TURKEY

KENYA
HARVEST

TV ADS
CAMPAIGN

TEAM
CAPTAIN

test set:

FIRST PRIVATE
CHINESE
AIRLINE

13 / 58
Examples of how search engines use classification

- Language identification (classes: English vs. French etc.)
- The automatic detection of spam pages (spam vs. nonspam)
- Sentiment detection: is a movie or product review positive or negative (positive vs. negative)
- Topic-specific or *vertical* search – restrict search to a “vertical” like “related to health” (relevant to vertical vs. not)
The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.
- We compute the probability of a document \(d\) being in a class \(c\) as follows:

\[
P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)
\]

- \(n_d\) is the length of the document. (number of tokens)
- \(P(t_k|c)\) is the conditional probability of term \(t_k\) occurring in a document of class \(c\)
- \(P(t_k|c)\) as a measure of how much evidence \(t_k\) contributes that \(c\) is the correct class.
- \(P(c)\) is the prior probability of \(c\).
- If a document’s terms do not provide clear evidence for one class vs. another, we choose the \(c\) with highest \(P(c)\).
Our goal in Naive Bayes classification is to find the “best” class.

The best class is the most likely or maximum a posteriori (MAP) class \( c_{\text{map}} \):

\[
c_{\text{map}} = \arg \max_{c \in C} \hat{P}(c|d) = \arg \max_{c \in C} \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)
\]
Taking the log

- Multiplying lots of small probabilities can result in floating point underflow.
- Since $\log(xy) = \log(x) + \log(y)$, we can sum log probabilities instead of multiplying probabilities.
- Since log is a monotonic function, the class with the highest score does not change.
- So what we usually compute in practice is:

$$c_{\text{map}} = \arg \max_{c \in C} \left[ \log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k | c) \right]$$
Naive Bayes classifier

Classification rule:

\[ c_{\text{map}} = \arg \max_{c \in C} \left[ \log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k | c) \right] \]

Simple interpretation:
1. Each conditional parameter \( \log \hat{P}(t_k | c) \) is a weight that indicates how good an indicator \( t_k \) is for \( c \).
2. The prior \( \log \hat{P}(c) \) is a weight that indicates the relative frequency of \( c \).
3. The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class.
4. We select the class with the most evidence.
Parameter estimation take 1: Maximum likelihood

- Estimate parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$ from train data: How?
- Prior:

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

- $N_c$: number of docs in class $c$; $N$: total number of docs
- Conditional probabilities:

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

- $T_{ct}$ is the number of tokens of $t$ in training documents from class $c$ (includes multiple occurrences)
- We’ve made a Naive Bayes independence assumption here: $\hat{P}(t_k|c) = \hat{P}(t_k|c)$, independent of position
The problem with maximum likelihood estimates: Zeros

\[ P(\text{China}|d) \propto P(\text{China}) \cdot P(\text{BEIJING}|\text{China}) \cdot P(\text{AND}|\text{China}) \]
\[ \cdot P(\text{TAIPEI}|\text{China}) \cdot P(\text{JOIN}|\text{China}) \cdot P(\text{WTO}|\text{China}) \]

- If WTO never occurs in class China in the train set:

\[
\hat{P}(\text{WTO}|\text{China}) = \frac{T_{\text{China},\text{WTO}}}{\sum_{t' \in V} T_{\text{China},t'}} = \frac{0}{\sum_{t' \in V} T_{\text{China},t'}} = 0
\]
The problem with maximum likelihood estimates: Zeros (cont)

- If there are no occurrences of WTO in documents in class China, we get a zero estimate:

$$\hat{P}(\text{WTO} | \text{China}) = \frac{T_{\text{China}, \text{WTO}}}{\sum_{t' \in V} T_{\text{China}, t'}} = 0$$

- We will get $P(\text{China} | d) = 0$ for any document that contains WTO!
To avoid zeros: Add-one smoothing

- Before:
  \[ \hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}} \]

- Now: Add one to each count to avoid zeros:
  \[ \hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V} (T_{ct'} + 1)} = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B} \]

- \( B \) is the number of bins – in this case the number of different words or the size of the vocabulary \(|V| = M\)
Naive Bayes: Summary

- Estimate parameters from the training corpus using add-one smoothing
- For a new document, for each class, compute sum of (i) log of prior and (ii) logs of conditional probabilities of the terms
- Assign the document to the class with the largest score
Exercise: Estimate parameters, classify test set

<table>
<thead>
<tr>
<th>docID</th>
<th>words in document</th>
<th>in $c = \text{China}$?</th>
</tr>
</thead>
<tbody>
<tr>
<td>training set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Chinese Beijing Chinese</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>Chinese Chinese Shanghai</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>Chinese Macao</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>Tokyo Japan Chinese</td>
<td>no</td>
</tr>
<tr>
<td>test set</td>
<td>5 Chinese Chinese Chinese Tokyo Japan</td>
<td>?</td>
</tr>
</tbody>
</table>

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

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V} (T_{ct'} + 1)} = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B}$$

($B$ is the number of bins – in this case the number of different words or the size of the vocabulary $|V| = M$)

$$c_{\text{map}} = \arg \max_{c \in C} [\hat{P}(c) \cdot \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)]$$
Example: Parameter estimates

Priors: \( \hat{P}(c) = 3/4 \) and \( \hat{P}(\overline{c}) = 1/4 \) Conditional probabilities:

\[
\begin{align*}
\hat{P}(\text{CHINESE}|c) & = \frac{5 + 1}{8 + 6} = \frac{6}{14} = \frac{3}{7} \\
\hat{P}(\text{TOKYO}|c) & = \hat{P}(\text{JAPAN}|c) = \frac{0 + 1}{8 + 6} = \frac{1}{14} \\
\hat{P}(\text{CHINESE}|\overline{c}) & = \frac{1 + 1}{3 + 6} = \frac{2}{9} \\
\hat{P}(\text{TOKYO}|\overline{c}) & = \hat{P}(\text{JAPAN}|\overline{c}) = \frac{1 + 1}{3 + 6} = \frac{2}{9}
\end{align*}
\]

The denominators are \((8 + 6)\) and \((3 + 6)\) because the lengths of \(text_c\) and \(text_{\overline{c}}\) are 8 and 3, respectively, and because the constant \(B\) is 6 as the vocabulary consists of six terms.
Example: Classification

\[ \hat{P}(c|d_5) \propto \frac{3}{4} \cdot \left(\frac{3}{7}\right)^3 \cdot \frac{1}{14} \cdot \frac{1}{14} \approx 0.0003 \]

\[ \hat{P}(\overline{c}|d_5) \propto \frac{1}{4} \cdot \left(\frac{2}{9}\right)^3 \cdot \frac{2}{9} \cdot \frac{2}{9} \approx 0.0001 \]

Thus, the classifier assigns the test document to \( c = \text{China} \). The reason for this classification decision is that the three occurrences of the positive indicator \textit{Chinese} in \( d_5 \) outweigh the occurrences of the two negative indicators \textit{Japan} and \textit{Tokyo}. 
Outline

1. Recap
2. Text classification
3. Naive Bayes
4. NB theory
5. Evaluation of TC
Now we want to gain a better understanding of the properties of Naive Bayes.

We will formally derive the classification rule . . .

. . . and make our assumptions explicit.
Derivation of Naive Bayes rule

We want to find the class that is most likely given the document:

\[ c_{\text{map}} = \arg \max_{c \in C} P(c \mid d) \]

Apply Bayes rule \( P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \):

\[ c_{\text{map}} = \arg \max_{c \in C} \frac{P(d \mid c)P(c)}{P(d)} \]

Drop denominator since \( P(d) \) is the same for all classes:

\[ c_{\text{map}} = \arg \max_{c \in C} P(d \mid c)P(c) \]
Too many parameters / sparseness

\[
c_{\text{map}} = \arg \max_{c \in \mathbb{C}} P(d|c)P(c) = \arg \max_{c \in \mathbb{C}} P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle|c)P(c)
\]

- There are too many parameters \( P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle|c) \), one for each unique combination of a class and a sequence of words.
- We would need a very, very large number of training examples to estimate that many parameters.
- This is the problem of data sparseness.
Naive Bayes conditional independence assumption

To reduce the number of parameters to a manageable size, we make the Naive Bayes conditional independence assumption:

\[
P(d|c) = P(\langle t_1, \ldots, t_{n_d} \rangle|c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k|c)
\]

We assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities \(P(X_k = t_k|c)\). Recall from earlier the estimates for these conditional probabilities: 
\[
\hat{P}(t|c) = \frac{T_{ct} + 1}{\left( \sum_{t' \in V} T_{ct'} \right) + B}
\]
Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are *badly* violated.

Example:

<table>
<thead>
<tr>
<th></th>
<th>(c_1)</th>
<th>(c_2)</th>
<th>class selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>true probability (P(c</td>
<td>d))</td>
<td>0.6</td>
<td>0.4</td>
</tr>
<tr>
<td>(\hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k</td>
<td>c))</td>
<td>0.00099</td>
<td>0.00001</td>
</tr>
<tr>
<td>NB estimate (\hat{P}(c</td>
<td>d))</td>
<td>0.99</td>
<td>0.01</td>
</tr>
</tbody>
</table>

- Double counting of evidence causes underestimation (0.01) and overestimation (0.99).

- Classification is about predicting the correct class and *not* about accurately estimating probabilities.

- Naive Bayes is terrible for correct estimation . . .

- . . . but if often performs well at accurate prediction (choosing the correct class).
Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods
- Better than methods like decision trees when we have many equally important features
- A good dependable baseline for text classification (but not the best)
- Optimal if independence assumptions hold (never true for text, but true for some domains)
- Very fast
- Low storage requirements
Outline

1 Recap

2 Text classification

3 Naive Bayes

4 NB theory

5 Evaluation of TC
Evaluation on Reuters

classes:

REGIONS
- UK
- China

INDUSTRIES
- poultry
- coffee

SUBJECT AREAS
- elections
- sports

training set:

- CONGESTION
- LONDON
- OLYMPICS
- BEIJING
- FEED
- CHICKEN
- ROASTING
- BEANS
- RECOUNT
- VOTES
- DIAMOND
- BASEBALL
- PARLIAMENT
- BIG BEN
- TOURISM
- GREAT WALL
- PATE
- DUCKS
- ARABICA
- ROBUSTA
- SEAT
- RUN-OFF
- FORWARD
- SOCCER
- WINDSOR
- THE QUEEN
- MAO
- COMMUNIST
- BIRD FLU
- TURKEY
- KENYA
- HARVEST
- TV ADS
- CAMPAIGN
- TEAM
- CAPTAIN

test set:

- FIRST
- PRIVATE
- CHINESE
- AIRLINE

\[ \gamma(d') = \text{China} \]
Example: The Reuters collection

<table>
<thead>
<tr>
<th>symbol</th>
<th>statistic</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>documents</td>
<td>800,000</td>
</tr>
<tr>
<td>$L$</td>
<td>avg. # word tokens per document</td>
<td>200</td>
</tr>
<tr>
<td>$M$</td>
<td>word types</td>
<td>400,000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>type of class</th>
<th>number</th>
<th>examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>region</td>
<td>366</td>
<td>UK, China</td>
</tr>
<tr>
<td>industry</td>
<td>870</td>
<td>poultry, coffee</td>
</tr>
<tr>
<td>subject area</td>
<td>126</td>
<td>elections, sports</td>
</tr>
</tbody>
</table>
Extreme conditions create rare Antarctic clouds

Tue Aug 1, 2006 3:20am ET

SYDNEY (Reuters) - Rare, mother-of-pearl colored clouds caused by extreme weather conditions above Antarctica are a possible indication of global warming, Australian scientists said on Tuesday.

Known as nacreous clouds, the spectacular formations showing delicate wisps of colors were photographed in the sky over an Australian
Evaluating classification

- Evaluation must be done on test data that are independent of the training data, i.e., training and test sets are disjoint.
- It’s easy to get good performance on a test set that was available to the learner during training (e.g., just memorize the test set).
- Measures: Precision, recall, $F_1$, classification accuracy
Precision $P$ and recall $R$

<table>
<thead>
<tr>
<th></th>
<th>in the class</th>
<th>not in the class</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicted to be in</td>
<td>true positives (TP)</td>
<td>false positives (FP)</td>
</tr>
<tr>
<td>the class</td>
<td></td>
<td></td>
</tr>
<tr>
<td>predicted to not be</td>
<td>false negatives (FN)</td>
<td>true negatives (TN)</td>
</tr>
<tr>
<td>in the class</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TP, FP, FN, TN are counts of documents. The sum of these four counts is the total number of documents.

precision:\[P = \frac{TP}{TP + FP}\]

recall:\[R = \frac{TP}{TP + FN}\]
A combined measure: $F$

- $F_1$ allows us to trade off precision against recall.
  
  $$F_1 = \frac{1}{\frac{1}{2}P + \frac{1}{2}R} = \frac{2PR}{P + R}$$

- This is the harmonic mean of $P$ and $R$: $\frac{1}{F} = \frac{1}{2}(\frac{1}{P} + \frac{1}{R})$
Averaging: Micro vs. Macro

- We now have an evaluation measure ($F_1$) for one class.
- But we also want a single number that measures the aggregate performance over all classes in the collection.

**Macroaveraging**
- Compute $F_1$ for each of the $C$ classes
- Average these $C$ numbers

**Microaveraging**
- Compute TP, FP, FN for each of the $C$ classes
- Sum these $C$ numbers (e.g., all TP to get aggregate TP)
- Compute $F_1$ for aggregate TP, FP, FN
### F₁ scores for Naive Bayes vs. other methods

**Table (a):**

<table>
<thead>
<tr>
<th></th>
<th>NB</th>
<th>Rocchio</th>
<th>kNN</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>micro-avg-L (90 classes)</td>
<td>80</td>
<td>85</td>
<td>86</td>
<td>89</td>
</tr>
<tr>
<td>macro-avg (90 classes)</td>
<td>47</td>
<td>59</td>
<td>60</td>
<td>60</td>
</tr>
</tbody>
</table>

**Table (b):**

<table>
<thead>
<tr>
<th></th>
<th>NB</th>
<th>Rocchio</th>
<th>kNN</th>
<th>trees</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>earn</td>
<td>96</td>
<td>93</td>
<td>97</td>
<td>98</td>
<td>98</td>
</tr>
<tr>
<td>acq</td>
<td>88</td>
<td>65</td>
<td>92</td>
<td>90</td>
<td>94</td>
</tr>
<tr>
<td>money-fx</td>
<td>57</td>
<td>47</td>
<td>78</td>
<td>66</td>
<td>75</td>
</tr>
<tr>
<td>grain</td>
<td>79</td>
<td>68</td>
<td>82</td>
<td>85</td>
<td>95</td>
</tr>
<tr>
<td>crude</td>
<td>80</td>
<td>70</td>
<td>86</td>
<td>85</td>
<td>89</td>
</tr>
<tr>
<td>trade</td>
<td>64</td>
<td>65</td>
<td>77</td>
<td>73</td>
<td>76</td>
</tr>
<tr>
<td>interest</td>
<td>65</td>
<td>63</td>
<td>74</td>
<td>67</td>
<td>78</td>
</tr>
<tr>
<td>ship</td>
<td>85</td>
<td>49</td>
<td>79</td>
<td>74</td>
<td>86</td>
</tr>
<tr>
<td>wheat</td>
<td>70</td>
<td>69</td>
<td>77</td>
<td>93</td>
<td>92</td>
</tr>
<tr>
<td>corn</td>
<td>65</td>
<td>48</td>
<td>78</td>
<td>92</td>
<td>90</td>
</tr>
<tr>
<td>micro-avg (top 10)</td>
<td>82</td>
<td>65</td>
<td>82</td>
<td>88</td>
<td>92</td>
</tr>
<tr>
<td>micro-avg-D (118 classes)</td>
<td>75</td>
<td>62</td>
<td>n/a</td>
<td>n/a</td>
<td>87</td>
</tr>
</tbody>
</table>

Naive Bayes does pretty well, but some methods beat it consistently (e.g., SVM).
Take-away today

- Text classification: definition & relevance to information retrieval
- Naive Bayes: simple baseline text classifier
- Theory: derivation of Naive Bayes classification rule & analysis
- Evaluation of text classification: how do we know it worked / didn’t work?
Resources

- Chapter 13 of IIR
- Resources at http://cislmu.org
  - Weka: A data mining software package that includes an implementation of Naive Bayes
  - Reuters-21578 – text classification evaluation set
  - Vulgarity classifier fail
Introduction to Information Retrieval
http://informationretrieval.org

IIR 14: Vector Space Classification

Hinrich Schütze

Center for Information and Language Processing, University of Munich

2013-05-28
Overview

1. Recap
2. Intro vector space classification
3. Rocchio
4. kNN
5. Linear classifiers
6. > two classes
Take-away today

- **Vector space classification**: Basic idea of doing text classification for documents that are represented as vectors
- **Rocchio classifier**: Rocchio relevance feedback idea applied to text classification
- **$k$ nearest neighbor classification**
- **Linear classifiers**
- **More than two classes**
Outline

1. Recap
2. Intro vector space classification
3. Rocchio
4. kNN
5. Linear classifiers
6. > two classes
Recall vector space representation

- Each document is a vector, one component for each term.
- Terms are axes.
- High dimensionality: 100,000s of dimensions
- Normalize vectors (documents) to unit length
- How can we do classification in this space?
Basic text classification setup

classes:
- UK
- China
- poultry
- coffee
- elections
- sports

regions industries subject areas
CONGESTION LONDON
OLYMPICS BEIJING
FEED CHICKEN
ROASTING BEANS
RECOUNT VOTES
DIAMOND BASEBALL

PARLIAMENT BIG BEN
TOURISM GREAT WALL
PATE DUCKS
ARABICA ROBUSTA
SEAT RUN-OFF
FORWARD SOCCER

WINDSOR THE QUEEN
MAO COMMUNIST
BIRD FLU TURKEY
KENYA HARVEST
TV ADS CAMPAIGN
TEAM CAPTAIN

test set:
FIRST PRIVATE CHINESE AIRLINE

$\gamma(d') = \text{China}$
Vector space classification

- As before, the training set is a set of documents, each labeled with its class.
- In vector space classification, this set corresponds to a labeled set of points or vectors in the vector space.
- Premise 1: Documents in the same class form a contiguous region.
- Premise 2: Documents from different classes don't overlap.
- We define lines, surfaces, hypersurfaces to divide regions.
Classes in the vector space

Should the document ⭐ be assigned to China, UK or Kenya? Find separators between the classes. Based on these separators: ⭐ should be assigned to China. How do we find separators that do a good job at classifying new documents like ⭐? – Main topic of today
Aside: 2D/3D graphs can be misleading

Left: A projection of the 2D semicircle to 1D. For the points $x_1, x_2, x_3, x_4, x_5$ at x coordinates $-0.9, -0.2, 0, 0.2, 0.9$ the distance $|x_2x_3| \approx 0.201$ only differs by 0.5% from $|x'_2x'_3| = 0.2$; but $|x_1x_3|/|x'_1x'_3| = d_{true}/d_{projected} \approx 1.06/0.9 \approx 1.18$ is an example of a large distortion (18%) when projecting a large area. Right: The corresponding projection of the 3D hemisphere to 2D.
Outline

1. Recap
2. Intro vector space classification
3. Rocchio
4. kNN
5. Linear classifiers
6. > two classes
kNN classification

- kNN classification is another vector space classification method.
- It also is very simple and easy to implement.
- kNN is more accurate (in most cases) than Naive Bayes and Rocchio.
- If you need to get a pretty accurate classifier up and running in a short time . . .
- . . . and you don’t care about efficiency that much . . .
- . . . use kNN.
kNN classification

- kNN = $k$ nearest neighbors
- **kNN classification rule for** $k = 1$ (1NN): Assign each test document to the class of its *nearest neighbor* in the training set.
- 1NN is not very robust – one document can be mislabeled or atypical.
- **kNN classification rule for** $k > 1$ (kNN): Assign each test document to the *majority class of its* $k$ nearest neighbors in the training set.
- **Rationale of kNN:** contiguity hypothesis
  - We expect a test document $d$ to have the same label as the training documents located in the local region surrounding $d$. 

Probabilistic kNN

- Probabilistic version of kNN: $P(c|d) = \text{fraction of } k \text{ neighbors of } d \text{ that are in } c$
- kNN classification rule for probabilistic kNN: Assign $d$ to class $c$ with highest $P(c|d)$
kNN is based on Voronoi tessellation
Exercise

How is star classified by:

(i) 1-NN  (ii) 3-NN  (iii) 9-NN  (iv) 15-NN  (v) Rocchio?
Curse of dimensionality

- Our intuitions about space are based on the 3D world we live in.
- Intuition 1: some things are close by, some things are distant.
- Intuition 2: we can carve up space into areas such that: within an area things are close, distances between areas are large.
- These two intuitions don’t necessarily hold for high dimensions.
- In particular: for a set of \( k \) uniformly distributed points, let \( d_{\text{min}} \) be the smallest distance between any two points and \( d_{\text{max}} \) be the largest distance between any two points.
- Then

\[
\lim_{d \to \infty} \frac{d_{\text{max}} - d_{\text{min}}}{d_{\text{min}}} = 0
\]
Curse of dimensionality: Simulation

- Simulate
  \[ \lim_{d \to \infty} \frac{d_{\text{max}} - d_{\text{min}}}{d_{\text{min}}} = 0 \]
- Pick a dimensionality \( d \)
- Generate 10 random points in the \( d \)-dimensional hypercube (uniform distribution)
- Compute all 45 distances
- Compute \( \frac{d_{\text{max}} - d_{\text{min}}}{d_{\text{min}}} \)
- We see that intuition 1 (some things are close, others are distant) is not true for high dimensions.
Intuition 2: Space can be carved up

- Intuition 2: we can carve up space into areas such that: within an area things are close, distances between areas are large.
- If this is true, then we have a simple and efficient algorithm for kNN.
- To find the $k$ closest neighbors of data point $<x_1, x_2, \ldots, x_d>$ do the following.
- Using binary search find all data points whose first dimension is in $[x_1 - \epsilon, x_1 + \epsilon]$. This is $O(\log n)$ where $n$ is the number of data points.
- Do this for each dimension, then intersect the $d$ subsets.
Intuition 2: Space can be carved up

- Size of data set \( n = 100 \)
- Again, assume uniform distribution in hypercube
- Set \( \epsilon = 0.05 \): we will look in an interval of length 0.1 for neighbors on each dimension.
- What is the probability that the nearest neighbor of a new data point \( \vec{x} \) is in this neighborhood in \( d = 1 \) dimension?
  - for \( d = 1 \): \( 1 - (1 - 0.1)^{100} \approx 0.99997 \)
- In \( d = 2 \) dimensions?
  - for \( d = 2 \): \( 1 - (1 - 0.1^2)^{100} \approx 0.63 \)
- In \( d = 3 \) dimensions?
  - for \( d = 3 \): \( 1 - (1 - 0.1^3)^{100} \approx 0.095 \)
- In \( d = 4 \) dimensions?
  - for \( d = 4 \): \( 1 - (1 - 0.1^4)^{100} \approx 0.0095 \)
- In \( d = 5 \) dimensions?
  - for \( d = 5 \): \( 1 - (1 - 0.1^5)^{100} \approx 0.0009995 \)
Intuition 2: Space can be carved up

- In $d = 5$ dimensions?
- For $d = 5$: $1 - (1 - 0.1^5)^{100} \approx 0.0009995$
- In other words: with enough dimensions, there is only one “local” region that will contain the nearest neighbor with high certainty: the entire search space.
- We cannot carve up high-dimensional space into neat neighborhoods . . .
- . . . unless the “true” dimensionality is much lower than $d$. 
kNN: Discussion

- No training necessary
  - But linear preprocessing of documents is as expensive as training Naive Bayes.
  - We always preprocess the training set, so in reality training time of kNN is linear.

- kNN is very accurate if training set is large.
- Optimality result: asymptotically zero error if Bayes rate is zero.
- But kNN can be very inaccurate if training set is small.
Outline

1 Recap

2 Intro vector space classification

3 Rocchio

4 kNN

5 Linear classifiers

6 > two classes
How to combine hyperplanes for > 2 classes?
One-of problems

- One-of or multiclass classification
  - Classes are mutually exclusive.
  - Each document belongs to exactly one class.
  - Example: language of a document (assumption: no document contains multiple languages)
One-of classification with linear classifiers

- Combine two-class linear classifiers as follows for one-of classification:
  - Run each classifier separately
  - Rank classifiers (e.g., according to score)
  - Pick the class with the highest score
Any-of problems

- Any-of or multilabel classification
  - A document can be a member of 0, 1, or many classes.
  - A decision on one class leaves decisions open on all other classes.
  - A type of “independence” (but not statistical independence)
  - Example: topic classification
  - Usually: make decisions on the region, on the subject area, on the industry and so on “independently”
Any-of classification with linear classifiers

- Combine two-class linear classifiers as follows for any-of classification:
  - Simply run each two-class classifier separately on the test document and assign document accordingly.
Take-away today

- **Vector space classification**: Basic idea of doing text classification for documents that are represented as vectors
- **Rocchio classifier**: Rocchio relevance feedback idea applied to text classification
- $k$ nearest neighbor classification
- Linear classifiers
- More than two classes
Resources

- Chapter 13 of IIR (feature selection)
- Chapter 14 of IIR
- Resources at http://cis1mu.org
  - Perceptron example
  - Text classification chapter on decision trees and perceptrons: Manning & Schütze (1999)
Resources

- Chapter 14 of IIR (basic vector space classification)
- Chapter 15 of IIR (SVMs)
- Discussion of “how to select the right classifier for my problem” in Russell and Norvig
- Resources at http://cislmu.org