Summary statistics

(You should be familiar with most of these already...)

- The *frequency* of an attribute value is the percentage of objects which take that value for that attribute.
- The *mode* of an attribute is the value with highest frequency.
- The $p^{th}$ *percentile* $x_p$ of an attribute $x$ is a value such that $p\%$ of the observed values of $x$ are less than $x_p$.
- The *mean* value of an attribute is the sum divided by the number of records.
- The *median* value is the 50$^{th}$ percentile.
- The *range* is the maximum value minus the minimum value.
- The *variance* is the mean squared deviation from the mean, and the *standard deviation* is the square root of the variance.
- The *covariance matrix* of a random vector contains the variances of the components on the diagonal and the covariances in off-diagonal elements.
Visualization

- Goal of visualization:
  *Make (some aspect of) data easily understandable to a person*

- Why is visualization a useful tool?
  - The human visual system is good at quickly detecting patterns
  - Utilizes domain knowledge of experts that may be hard to formalize
  - Example: Graph of social network (on Facebook)
Examples: Box plot

- Outlier
- 10th percentile
- 75th percentile
- 50th percentile
- 25th percentile
- 10th percentile

Following figure shows the basic part of a box plot:

Another way of displaying the distribution of data invented by J. Tukey.
Examples: Histogram (1d)
- Bin width and placing important
Examples: Scatterplot

- Attributes values determine the position
- Two-dimensional scatter plots most common, but can have three-dimensional scatter plots
- Often additional attributes can be displayed by using the size, shape, and color of the markers that represent the objects
- It is useful to have arrays of scatter plots to compactly summarize the relationships of several pairs of attributes
Examples: Histogram (2d)
Examples: Contour plot

- Note: Usage of color here is questionable because color space is cyclic and not perceptually linear.
Examples: Correlation matrix

- Correlations (or distances) between *pairs of objects* or *pairs of attributes*
- A good ordering is crucial
  (so this is most useful with labels or after clustering)
- Again, usage of color scale here is questionable
Examples: Parallel coordinates

- Ordering of attributes affects the visualization
- Color can be used to indicate class or cluster (as in this example)
Classification: Basic concepts
Classification: Basic concepts

- The classification problem
- Cost functions and performance measures
- k Nearest Neighbors (kNN)
- Bayesian classifier
- Naive Bayes
- Generalization performance
The classification problem

- Observe pairs \((x_1, y_1) \ldots (x_N, y_N)\) where \(x_i \in \mathcal{X}\), with \(\mathcal{X}\) an arbitrary set, and \(y_i \in \mathcal{Y}\), with \(\mathcal{Y}\) a finite (typically small) set.

- Given a set \(\{x_{N+1} \ldots x_{N+M}\}\) of new objects (each one also \(\in \mathcal{X}\)), predict the corresponding classes \(y_{N+1} \ldots y_{N+M}\) (each one also \(\in \mathcal{Y}\)). (Typically the new objects are predicted one at a time, independently of each other.)
Example 1: spam filter

- $\mathcal{X}$ is the set of all possible emails (strings)
- $\mathcal{Y}$ is the set \{ spam, non-spam \}

<table>
<thead>
<tr>
<th>From: <a href="mailto:medshop@spam.com">medshop@spam.com</a></th>
<th>Subject: viagra cheap meds...</th>
<th>spam</th>
</tr>
</thead>
<tbody>
<tr>
<td>From: <a href="mailto:my.professor@helsinki.fi">my.professor@helsinki.fi</a></td>
<td>Subject: important information here’s how to ace the test...</td>
<td>non-spam</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>From: <a href="mailto:mike@example.org">mike@example.org</a></td>
<td>Subject: you need to see this how to win $1,000,000...</td>
<td>?</td>
</tr>
</tbody>
</table>
Example 2: face recognition

- $\mathcal{X}$ is the set of all possible images
- $\mathcal{Y}$ is the set \{patrik, doris, antti\}
Example 3: handwritten digit recognition

- $\mathcal{X}$ is the set of all possible images of fixed (small) size
- $\mathcal{Y}$ is the set \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}
**Example 4: credit card fraud detection**

- $\mathcal{X}$ is the set of all possible ‘last three transactions’
- $\mathcal{Y}$ is the set \{fraud, normal\}

<table>
<thead>
<tr>
<th>Cash advance $1,000</th>
<th>fraud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cash advance $10,000</td>
<td></td>
</tr>
<tr>
<td>Flight out of the country $1,200</td>
<td></td>
</tr>
<tr>
<td>Candy bar $1.20</td>
<td>normal</td>
</tr>
<tr>
<td>Groceries $67.10</td>
<td></td>
</tr>
<tr>
<td>Restaurant $35.82</td>
<td></td>
</tr>
<tr>
<td>Flight out of the country $380</td>
<td></td>
</tr>
<tr>
<td>Hotel booking $210</td>
<td></td>
</tr>
<tr>
<td>Groceries $69.20</td>
<td></td>
</tr>
</tbody>
</table>
Where do the correct labels come from?

i.e. How do we get labeled training data?

- Predicting the future: time eventually hands us the answer
  - Example: Predicting $Y = \{ \text{rain, no rain} \}$ tomorrow, based on weather today and yesterday

- Automating something that humans can do: We can use human-labeled data to train the computer to perform the same task
  - Example: Spam filtering, face recognition, ...

- Creating a simple predictor to replace an expensive or slow test: Use the expensive or slow test to build the training data
  - Example: Predicting disease based on symptoms or cheap tests
Prediction type

Given a new input $x'$, our prediction for the corresponding $y'$ can come in at least one of three different forms:

1. Single class (‘forced choice’): $\hat{y}' \in \mathcal{Y}$

2. Single class or “don’t know”: $\hat{y}' \in (\mathcal{Y} \cup \{“don’t know”\})$

3. Probability distribution over classes: $\hat{P}(y)$ with
   $\forall y \in \mathcal{Y}: \hat{P}(y) \geq 0$ and $\sum_y \hat{P}(y) = 1$

Note: Unless otherwise stated, the default in the remainder of this course will be prediction type (1), i.e. ‘forced choice’.
Prediction type: Example

Predicting whether tomorrow it will

A. Rain a lot (more than 5mm)
B. Rain a little (less than 5mm but more than 0mm)
C. Not rain (0mm)

Prediction type:

1. Forced choice:

<table>
<thead>
<tr>
<th>date</th>
<th>prediction</th>
<th>outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>2.1</td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>3.1</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>4.1</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>5.1</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
2. Allowing for “don’t know” answer:

<table>
<thead>
<tr>
<th>date</th>
<th>prediction</th>
<th>outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>2.1</td>
<td>don’t know</td>
<td>A</td>
</tr>
<tr>
<td>3.1</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>4.1</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>5.1</td>
<td>don’t know</td>
<td>C</td>
</tr>
</tbody>
</table>

2. Probability distribution over the classes:

<table>
<thead>
<tr>
<th>date</th>
<th>prediction</th>
<th>outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>(A: 0.6, B: 0.3, C: 0.1)</td>
<td>A</td>
</tr>
<tr>
<td>2.1</td>
<td>(A: 0.5, B: 0.45, C: 0.05)</td>
<td>A</td>
</tr>
<tr>
<td>3.1</td>
<td>(A: 0.8, B: 0.1, C: 0.1)</td>
<td>B</td>
</tr>
<tr>
<td>4.1</td>
<td>(A: 0.3, B: 0.34, C: 0.36)</td>
<td>C</td>
</tr>
<tr>
<td>5.1</td>
<td>(A: 0.2, B: 0.4, C: 0.4)</td>
<td>C</td>
</tr>
</tbody>
</table>

... ... ...
Illustrative 2d classification example

- Point clouds in 2d useful for illustrating basic ideas and approaches...

- ...nevertheless, always keep in mind that real problems can have hundreds (or thousands) of dimensions!

![Illustration of a 2d classification example with points colored in red and green, one test example, and annotations for training examples of class 1 and 2, as well as the test example.]
More 2d classification examples

- Separable, linearly separable, non-separable, multiclass, etc
'Confusion' matrix

- For prediction type (1) and (2), we can analyze classification performance as follows:
  - For each data object, store both the true class \((y)\) and the predicted class \((\hat{y})\)
  - Count how many times each pair happened

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>don't know</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td>82</td>
<td>13</td>
<td>3</td>
<td>16</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>2</td>
<td>12</td>
<td>17</td>
<td>11</td>
</tr>
<tr>
<td><strong>C</strong></td>
<td>5</td>
<td>24</td>
<td>92</td>
<td>19</td>
</tr>
</tbody>
</table>

- true positive (TP)
- true negative (TN)
- false positive (FP)
- false negative (FN)
Performance metrics for binary classification

- Accuracy = \( \frac{TP + TN}{TP + TN + FP + FN} \)
- Error rate = \( 1 - \text{Accuracy} \)
- True positive rate (‘sensitivity’) = \( \frac{TP}{TP + FN} \)
- True negative rate (‘specificity’) = \( \frac{TN}{TN + FP} \)
- False positive rate = \( \frac{FP}{TN + FP} \)
- False negative rate = \( \frac{FN}{TP + FN} \)
- Recall = \( \frac{TP}{TP + FN} \)
- Precision = \( \frac{TP}{TP + FP} \)
Application-specific cost/utility function

- In some applications all correct answers are equally good and all wrong answers equally bad ⇒ ‘0/1 cost’

- In many applications different types of mistakes have different costs
  
  - Spam filtering: A spam message in the inbox is only a minor annoyance; an important message in the trash is very bad!
  
  - HIV test: A false positive only results in some additional tests, while a false negative can delay treatment and be fatal

- Formulating problem in terms of ‘utility’ is equivalent to using ‘cost’ (just change the sign!)

- Maximize expected utility = minimize expected cost (gives best performance in the long run)
Cost function for forced choice

- $C = \text{func(true class, predicted class)}$
  - i.e. a matrix of size $K \times K$, for $K$ classes

- Cost matrix not symmetric in general

- Costs can be dollars, but this assumes ‘utility’ linear in money

- Goal: minimize expected cost
  - (= average cost over many examples)
Expected cost

Prerequisites reminder: (see Appendix C in the textbook)

- The *expected value* of a random variable $C$ is given by

$$E\{C\} = \sum_c c \, P(C = c) \quad (6)$$

For instance, if the distribution of $C$ is $P(C = \$0) = 0.2$, $P(C = \$1) = 0.5$, and $P(C = \$8) = 0.3$, then we have

$$E\{C\} = \$0 \times 0.2 + \$1 \times 0.5 + \$8 \times 0.3 = \$(0 + 0.5 + 2.4) = \$2.9$$

- This is the *long run average cost*: When sampling $C$ from its distribution a large number of times the average cost converges to the expected cost.
The total cost obtained by the classifier is obtained by elementwise multiplication of the confusion matrix with the cost matrix. In the example below, the total cost is $0 \times 38 + $1 \times 6 + $30 \times 3 + $0 \times 27 = $96.

The optimal ‘forced choice’ answer can be computed from probabilistic predictions, and is not always the most likely class (example on next slide).
Example:

Using the cost matrix below, if the probabilistic prediction is $P(y = '−') = a$, and $P(y = '+') = 1 - a$, then

Expected cost of predicting ‘−’ = $a \cdot 0 + (1 - a) \cdot 30 = 30(1 - a)$
Expected cost of predicting ‘+’ = $a \cdot 1 + (1 - a) \cdot 0 = a$

Setting these two equal yields $30(1 - a) = a$ which is solved by $a = 30/31 \approx 0.968$. Whenever the probability of the negative class is higher than this, one should predict negative; in all other cases, predicting positive is the appropriate decision.

\[
\begin{array}{c|cc}
\text{predicted class} & - & + \\
\hline
\text{actual class} & \$0 & \$1 \\
- & \$30 & \$0 \\
\end{array}
\]
Cost function allowing for “don’t know” answer

- \( C = \text{func(true class, predicted class)} \)
  i.e. a matrix of size \( K \times (K + 1) \), for \( K \) classes

- Otherwise exactly as for the ‘forced choice’ case. Using probabilistic predictions we could compute the best answer, which would typically be “don’t know” when we are not sure enough.

<table>
<thead>
<tr>
<th>predicted class</th>
<th></th>
<th></th>
<th>don’t know</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual class</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>$0$</td>
<td>$1$</td>
<td>$0.50$</td>
</tr>
<tr>
<td>+</td>
<td>$30$</td>
<td>$0$</td>
<td>$0.50$</td>
</tr>
</tbody>
</table>

\( K \) classes
Cost function for probabilistic predictions

- $C = \text{func}(y_t, \hat{P})$, where $y_t$ takes any value in $\mathcal{Y}$, and $\hat{P}$ is a length $K$ unit sum non-negative vector that specifies the predicted probabilities for each of the $K$ classes

- Example 1: linear cost $C = 1 - \hat{P}(y_t)$

- Example 2: logarithmic cost $C = -\log \hat{P}(y_t)$
A cost function for probabilistic predictions is *proper* if, whenever the true probability distribution over the classes is $P$, the expected cost is minimized for $\hat{P} = P$.

The linear cost is *not* proper (and hence should be used only with caution):

Say the true distribution is $P(y = 0) = 1/4$, $P(y = 1) = 3/4$, and set the predicted distribution to $\hat{P}(y = 0) = b$, $\hat{P}(y = 1) = 1 - b$. The expected cost is

$$E\{C\} = E_{y_t}\{1 - \hat{P}(y = y_t)\}$$

$$= \frac{1}{4} (1 - b) + \frac{3}{4} (1 - (1 - b))$$

$$= \frac{1}{4} + \frac{1}{2} b$$

which is minimized for $b = 0$, not the true probability $b = 1/4$. 
The logarithmic cost *is proper* (only binary case shown below):

Say the true probability of class \( y = 0 \) is \( a \) (so the true probability of class \( y = 1 \) is \( 1 - a \)). Assume \( 0 < a < 1 \).

Our prediction is that the probability of \( y = 0 \) is \( b \) (so we are also predicting that the probability of \( y = 1 \) is \( 1 - b \)). Assume \( 0 < b < 1 \).

The expected logarithmic loss is:

\[
E\{C\} = -a \log b - (1 - a) \log(1 - b)
\]

Let’s find the value of \( b \) that minimizes the expected logarithmic loss. Taking the derivative with respect to \( b \) gives:

\[
\frac{d}{db} E\{C\} = -a \frac{1}{b} - (1 - a) \frac{1}{1 - b} (-1) = -\frac{a}{b} + \frac{1 - a}{1 - b}
\]

Setting this to zero gives:

\[
\frac{1 - a}{1 - b} = \frac{a}{b} \iff b - ab = a - ab \iff b = a
\]
Take the second derivative with respect to $b$:

$$\frac{d^2}{db^2}E\{C\} = \frac{a}{b^2} - \frac{1-a}{(1-b)^2}(-1) = \frac{a}{b^2} + \frac{1-a}{(1-b)^2} > 0$$

Hence $b = a$ is the unique minimum, and so the logarithmic cost is proper according to the definition.