This lecture roughly follow:
Admin

• **Assignment 1** is due next Friday: start early.
• Waiting list people: you should be registered soon-ish.
  – Start on the assignment now.
• Bookmark the course webpage:
• Sign up for a CS undergrad account:
  – [https://www.cs.ubc.ca/getacct](https://www.cs.ubc.ca/getacct)
  – Needed to access lectures, notes, assignment, and to submit assignment.
• Sign up for the course [Piazza](http://piazza.com/ubc.ca/winterterm12018/cpsc340) group:
• Tutorials start next week.
• Office hours start next week (see the calendar).
• Auditing: message instructors on Piazza if you want to audit.
Data Mining: Bird’s Eye View

1) Collect data.
2) Data mining!
3) Profit?

Unfortunately, it’s often more complicated...
Data Mining: Some Typical Steps

1) Learn about the application.
2) Identify data mining task.
3) Collect data.
4) Clean and preprocess the data.
5) Transform data or select useful subsets.
6) Choose data mining algorithm.
7) Data mining!
8) Evaluate, visualize, and interpret results.
9) Use results for profit or other goals.
   (often, you’ll go through cycles of the above)
Data Mining: Some Typical Steps

1) Learn about the application.
2) Identify data mining task.
3) Collect data.
4) Clean and preprocess the data.
5) Transform data or select useful subsets.
6) Choose data mining algorithm.
7) Data mining!
8) Evaluate, visualize, and interpret results.
9) Use results for profit or other goals.
   (often, you’ll go through cycles of the above)
What is Data?

- We’ll define data as a collection of **examples**, and their **features**.

<table>
<thead>
<tr>
<th>Age</th>
<th>Job?</th>
<th>City</th>
<th>Rating</th>
<th>Income</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>Yes</td>
<td>Van</td>
<td>A</td>
<td>22,000.00</td>
</tr>
<tr>
<td>23</td>
<td>Yes</td>
<td>Bur</td>
<td>BBB</td>
<td>21,000.00</td>
</tr>
<tr>
<td>22</td>
<td>No</td>
<td>Van</td>
<td>CC</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>Yes</td>
<td>Sur</td>
<td>AAA</td>
<td>57,000.00</td>
</tr>
<tr>
<td>19</td>
<td>No</td>
<td>Bur</td>
<td>BB</td>
<td>13,500.00</td>
</tr>
<tr>
<td>22</td>
<td>Yes</td>
<td>Van</td>
<td>A</td>
<td>20,000.00</td>
</tr>
<tr>
<td>21</td>
<td>Yes</td>
<td>Ric</td>
<td>A</td>
<td>18,000.00</td>
</tr>
</tbody>
</table>

- Each row is an “example”, each column is a “feature”.
  - Examples are also sometimes called “samples”.

Examples are also sometimes called “samples”.
Types of Data

• **Categorical features** come from an unordered set:
  – Binary: job?
  – Nominal: city.

• **Numerical features** come from ordered sets:
  – Discrete counts: age.
  – Ordinal: rating.
  – Continuous/real-valued: height.
Converting to Numerical Features

- Often want a real-valued example representation:
  - This is called a “1 of k” encoding.
  - We can now interpret examples as points in space:
    - E.g., first example is at (23,1,0,0,22000).

<table>
<thead>
<tr>
<th>Age</th>
<th>City</th>
<th>Income</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>Van</td>
<td>22,000.00</td>
</tr>
<tr>
<td>23</td>
<td>Bur</td>
<td>21,000.00</td>
</tr>
<tr>
<td>22</td>
<td>Van</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>Sur</td>
<td>57,000.00</td>
</tr>
<tr>
<td>19</td>
<td>Bur</td>
<td>13,500.00</td>
</tr>
<tr>
<td>22</td>
<td>Van</td>
<td>20,000.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Age</th>
<th>Van</th>
<th>Bur</th>
<th>Sur</th>
<th>Income</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>22,000.00</td>
</tr>
<tr>
<td>23</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>21,000.00</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>25</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>57,000.00</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>13,500.00</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>20,000.00</td>
</tr>
</tbody>
</table>
Approximating Text with Numerical Features

- **Bag of words** replaces document by word counts:

  The *International Conference on Machine Learning* (ICML) is the leading international *academic conference* in *machine learning*

<table>
<thead>
<tr>
<th>ICML</th>
<th>International</th>
<th>Conference</th>
<th>Machine</th>
<th>Learning</th>
<th>Leading</th>
<th>Academic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- Ignores order, but often captures general theme.
- You can compute ‘distance’ between documents.
Approximating Images and Graphs

• We can think of other data types in this way:
  – Images:
  – Graphs:
Data Cleaning

• ML+DM typically assume ‘clean’ data.
• Ways that data might not be ‘clean’:
  – Noise (e.g., distortion on phone).
  – Outliers (e.g., data entry or instrument error).
  – Missing values (no value available or not applicable)
  – Duplicated data (repetitions, or different storage formats).
• Any of these can lead to problems in analyses.
  – Want to fix these issues, if possible.
  – Some ML methods are robust to these.
  – Often, ML is the best way to detect/fix these.
The Question I Hate the Most...

• How much data do we need?

• A difficult if not impossible question to answer.

• My usual answer: “more is better”.
  – With the warning: “as long as the quality doesn’t suffer”.

• Another popular answer: “ten times the number of features”.
A Simple Setting: Coupon Collecting

• Assume we have a categorical variable with 50 possible values:
  – {Alabama, Alaska, Arizona, Arkansas,...}.

• Assume each category has probability of 1/50 of being chosen:
  – How many examples do we need to see before we expect to see them all?

• Expected value is \(\sim 225\).

• Coupon collector problem: \(O(n \log n)\) in general.
  – Gotta Catch’em all!

• Obvious sanity check, is need more samples than categories:
  – Situation is worse if they don’t have equal probabilities.
  – Typically want to see categories more than once to learn anything.
Feature Aggregation

• Feature aggregation:
  – Combine features to form new features:

<table>
<thead>
<tr>
<th>Van</th>
<th>Bur</th>
<th>Sur</th>
<th>Edm</th>
<th>Cal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BC</th>
<th>AB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

• More province information than city information.
Feature Selection

- Feature Selection:
  - Remove features that are not relevant to the task.

<table>
<thead>
<tr>
<th>SID:</th>
<th>Age</th>
<th>Job?</th>
<th>City</th>
<th>Rating</th>
<th>Income</th>
</tr>
</thead>
<tbody>
<tr>
<td>3457</td>
<td>23</td>
<td>Yes</td>
<td>Van</td>
<td>A</td>
<td>22,000.00</td>
</tr>
<tr>
<td>1247</td>
<td>23</td>
<td>Yes</td>
<td>Bur</td>
<td>BBB</td>
<td>21,000.00</td>
</tr>
<tr>
<td>6421</td>
<td>22</td>
<td>No</td>
<td>Van</td>
<td>CC</td>
<td>0.00</td>
</tr>
<tr>
<td>1235</td>
<td>25</td>
<td>Yes</td>
<td>Sur</td>
<td>AAA</td>
<td>57,000.00</td>
</tr>
<tr>
<td>8976</td>
<td>19</td>
<td>No</td>
<td>Bur</td>
<td>BB</td>
<td>13,500.00</td>
</tr>
<tr>
<td>2345</td>
<td>22</td>
<td>Yes</td>
<td>Van</td>
<td>A</td>
<td>20,000.00</td>
</tr>
</tbody>
</table>

- Student ID is probably not relevant.
Feature Transformation

• Mathematical transformations:
  – *Discretization* (binning): turn numerical data into categorical.

<table>
<thead>
<tr>
<th>Age</th>
<th>&lt; 20</th>
<th>&gt;= 20, &lt; 25</th>
<th>&gt;= 25</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

• Only need consider 3 values.
Feature Transformation

• Mathematical transformations:
  – **Discretization** (binning): turn numerical data into categorical.
  – Square, exponentiation, or take logarithm.
Feature Transformation

• Mathematical transformations:
  – Discretization (binning): turn numerical data into categorical.
  – Square, exponentiation, or take logarithm.
  – Scaling: convert variables to comparable scales
    (E.g., convert kilograms to grams.)
Feature Transformation

• Mathematical transformations:
  – **Discretization** (binning): turn numerical data into categorical.
  – Square, exponentiation, or take logarithm.
  – Scaling: convert variables to comparable scales.
  – Fourier coefficients, spectrograms, and wavelets (signal data).

https://en.wikipedia.org/wiki/Fourier_transform
https://en.wikipedia.org/wiki/Spectrogram
(pause)
Exploratory Data Analysis

• You should always ‘look’ at the data first.

• But how do you ‘look’ at features and high-dimensional examples?
  – Summary statistics.
  – Visualization.
  – ML + DM (later in course).
Categorical Summary Statistics

• Summary statistics for a categorical variable:
  – Frequencies of different classes.
  – Mode: category that occurs most often.
  – Quantiles: categories that occur more than t times.

Frequency: 13.3% of Canadian residents live in BC.
Mode: Ontario has largest number of residents (38.5%)
Quantile: 6 provinces have more than 1 million people.
Continuous Summary Statistics

• Measures of location:
  – **Mean**: average value.
  – **Median**: value such that half points are larger/smaller.
  – **Quantiles**: value such that ‘t’ points are larger.

• Measures of spread:
  – **Range**: minimum and maximum values.
  – **Variance**: measures how far values are from mean.
    - Square root of variance is “standard deviation”.
  – **Intequantile ranges**: difference between quantiles.
Continuous Summary Statistics

• Data: [0 1 2 3 3 5 7 8 9 10 14 15 17 200]
• Measures of location:
  – Mean(Data) = 21
  – Mode(Data) = 3
  – Median(Data) = 7.5
  – Quantile(Data,0.5) = 7.5
  – Quantile(Data,0.25) = 3
  – Quantile(Data,0.75) = 14
• Measures of spread:
  – Range(Data) = [0 200].
  – Std(Data) = 51.79
  – IQR(Data,.25,.75) = 11
• Notice that mean and std are more sensitive to extreme values ("outliers").
Entropy as Measure of Randomness

• Another common summary statistic is **entropy**.
  – Entropy measures “randomness” of a set of variables.
    • Roughly, another measure of the “spread” of values.
    • Formally, “how many bits of information are encoded in the average example”.

  – For a categorical variable that can take ‘k’ values, entropy is defined by:

\[
\text{entropy} = - \sum_{c=1}^{k} p_c \log p_c
\]

where \( p_c \) is the proportion of times you have value ‘c’.

  – Low entropy means “very predictable”.
  – High entropy means “very random”.
  – Minimum value is 0, maximum value is \( \log(k) \).

    • We use the convention that \( 0 \log 0 = 0 \).
Entropy as Measure of Randomness

Low entropy means “very predictable”  High entropy means “very random”

• For categorical features: uniform distribution has highest entropy.
• For continuous densities with fixed mean and variance:
  – Normal distribution has highest entropy (not obvious).
• Entropy and Dr. Seuss (words like “snunkoople” increase entropy).
Distances and Similarities

• There are also summary statistics between features ‘x’ and ‘y’.
  
  – **Hamming distance:**
    • Number of elements in the vectors that aren’t equal.
  
  – **Euclidean distance:**
    • How far apart are the vectors?
  
  – **Correlation:**
    • Does one increase/decrease linearly as the other increases?
    • Between -1 and 1.
Distances and Similarities

- There are also summary statistics between features ‘x’ and ‘y’.
  - Rank correlation:
    - Does one increase/decrease non-linearly as the other increases?

- Distances/similarities between other examples:
  - Jaccard coefficient (distance between sets):
    - \( \frac{\text{size of intersection of sets}}{\text{size of union of sets}} \)
  - Edit distance (distance between strings):
    - How many characters do we need to change to go from x to y?
    - Computed using dynamic programming (CPSC 320).

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Limitations of Summary Statistics

- On their own summary statistic can be misleading.
- Why not to trust statistics

- Amcomb’s quartet:
  - Almost same means.
  - Almost same variances.
  - Almost same correlations.
  - Look completely different.
- Datasaurus dozen.

https://en.wikipedia.org/wiki/Anscombe%27s_quartet
(pause)
Visualization

• You can learn a lot from 2D plots of the data:
  – Patterns, trends, outliers, unusual patterns.

<table>
<thead>
<tr>
<th>Lat</th>
<th>Long</th>
<th>Temp</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>30.1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>29.8</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>29.9</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>30.1</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>29.9</td>
</tr>
</tbody>
</table>

vs.

https://en.wikipedia.org/wiki/Temperature
Basic Plot

• Visualize one variable as a function of another.

[Graph 1: Projection Time against Vector Length (Sparsity = 10%)]

[Graph 2: Assaults Reported in the BJSVictimization Survey]

• Fun with plots.
Histograms display distribution of a variable.
Box Plot

- This whisker shows the lowest value.
- This line shows the lower quartile.
- This line shows the median.
- This line shows the upper quartile.
- This whisker shows the highest value.

The width of the box shows the interquartile range.

Maximum daily temperature in Melbourne:

http://www.bbc.co.uk/schools/gcsebitesize/maths/statistics/representingdata3hi
http://r.ramganalytics.com/r/facebook-likes-and-analytics/
Box Plot

- Photo from CTV Olympic coverage in 2010:
Matrix Plot

• If our features are real-valued, we can view data as a picture:
  – “Matrix plot”.
  – May be able to see trends in features.
Matrix Plot

- A matrix plot of all similarities (or distances) in a data set:

![Matrix Plot Diagram](image-url)
Scatterplot

• Look at distribution of two features:
  – Feature 1 on x-axis.
  – Feature 2 on y-axis.
  – Basically a “plot without lines” between the points.

- Shows correlation between “personality” score and “looks” score.

http://cdn.okccdn.com/blog/humanexperiments/looks-v-personality.png
Scatterplot

• Look at distribution of two features:
  – Feature 1 on x-axis.
  – Feature 2 on y-axis.
  – Basically a “plot without lines” between the points.

- Shows correlation between “personality” score and “looks” score.
- But scatterplots let you see more complicated patterns.

https://en.wikipedia.org/wiki/Anscombe%27s_quartet
Scatterplot Arrays

- For multiple variables, can use scatterplot array.

- Colors can indicate a third categorical variable.
Scatterplot Arrays

- For multiple variables, can use **scatterplot array**.

- Colors can indicate a third categorical variable.

http://www.ats.ucla.edu/stat/r/pages/layout.htm
Scatterplot Arrays

• For multiple variables, can use scatterplot array.

• Colors can indicate a third categorical variable.

http://www.ats.ucla.edu/stat/r/pages/layout.htm
(pause)
Motivating Example: Food Allergies

• You frequently start getting an upset stomach

• You suspect an adult-onset food allergy.

http://www.cliparthut.com/upset-stomach-clipart-cn48e5.html
Motivating Example: Food Allergies

• To solve the mystery, you start a food journal:

<table>
<thead>
<tr>
<th></th>
<th>Egg</th>
<th>Milk</th>
<th>Fish</th>
<th>Wheat</th>
<th>Shellfish</th>
<th>Peanuts</th>
<th>...</th>
<th>Sick?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.7</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7</td>
<td>0</td>
<td>0.6</td>
<td>0</td>
<td>0.01</td>
<td>0</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7</td>
<td>1.2</td>
<td>0</td>
<td>0.10</td>
<td>0.01</td>
<td>0.01</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>0.3</td>
<td>0</td>
<td>1.2</td>
<td>0.3</td>
<td>0.10</td>
<td>0.01</td>
<td>0.01</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

• But it’s hard to find the pattern:
  – You can’t isolate and only eat one food at a time.
  – You may be allergic to more than one food.
  – The quantity matters: a small amount may be ok.
  – You may be allergic to specific interactions.
Supervised Learning

• We can formulate this as **supervised learning**:

<table>
<thead>
<tr>
<th>Egg</th>
<th>Milk</th>
<th>Fish</th>
<th>Wheat</th>
<th>Shellfish</th>
<th>Peanuts</th>
<th>...</th>
<th>Sick?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.7</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7</td>
<td>0</td>
<td>0.6</td>
<td>0</td>
<td>0.01</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7</td>
<td>1.2</td>
<td>0</td>
<td>0.10</td>
<td>0.01</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>0.3</td>
<td>0</td>
<td>1.2</td>
<td>0.3</td>
<td>0.10</td>
<td>0.01</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

• Input for an **example** (day of the week) is a set of **features** (quantities of food).
• Output is a desired **class label** (whether or not we got sick).
• **Goal of supervised learning**:
  – Use data to find a model that outputs the right label based on the features.
  – Model predicts whether foods will make you sick (even with new combinations).
Supervised Learning

• General **supervised learning** problem:
  – Take features of examples and corresponding labels as inputs.
  – Find a model that can accurately *predict the labels of new examples*.

• This is the **most successful machine learning technique**:
  – Spam filtering, optical character recognition, Microsoft Kinect, speech recognition, classifying tumours, etc.

• We’ll first focus on **categorical labels**, which is called “**classification**”.
  – The model is a called a “**classifier**”.
Naïve Supervised Learning: “Predict Mode”

- A very naïve supervised learning method:
  - Count how many times each label occurred in the data (4 vs. 1 above).
  - Always predict the most common label, the “mode” (“sick” above).
- This ignores the features, so is only accurate if we only have 1 label.
- We want to use the features, and there are MANY ways to do this.
  - Next time we’ll consider a classic way known as decision tree learning.
Summary

• **Typical data mining steps:**
  – Involves data collection, preprocessing, analysis, and evaluation.

• **Example-feature representation** and categorical/numerical features.
  – Transforming non-vector examples to vector representations.

• **Feature transformations:**
  – To address coupon collecting or simplify relationships between variables.

• **Exploring data:**
  – Summary statistics and data visualization.

• **Supervised learning:**
  – Using data to write a program based on input/output examples.

• **Post-lecture bonus slides:** other visualizations, parallel/distributed.

• **Next week:** let’s start some machine learning...
Histogram

• Histogram with grouping:

[Image of a histogram showing the cause of death in America from 1900 to 2010]

Box Plots

• Box plot with grouping:
Map Coloring

• Color/intensity can represent feature of region.

Popularity of naming baby “Evelyn” over time:

But not very good if some regions are very small.

http://waitbutwhy.com/2013/12/how-to-name-baby.html

Canadian Income Mobility
Map Coloring

• Variation just uses fixed-size blocks and tries to arrange geographically:

[Diagram showing a map with states and their population percentages claiming American ancestry]

[Source: U.S. Census Bureau, 2015 ACS 1-Year Estimates]

http://www.visualcapitalist.com/america-immigration-1820/
Contour Plot

• Colour visualizes ‘z’ as we vary ‘x’ and ‘y’.
Treemaps

- Area represents attribute value:
Cartogram

• Fancier version of treemaps:

http://www-personal.umich.edu/~mejn/cartograms/
Stream Graph

This is every name in the Top 1,000 for each decade since 1880, all stacked on top of each other.

In 1950, 95% of all babies were given a Top 1,000 name.

In 2012, only 73% of baby names fell into the Top 1,000.

http://waitbutwhy.com/2013/12/how-to-name-baby.html
Stream Graph

http://www.babynamewizard.com/d3js-voyager/popup.html#prefix=ca&sw=both&exact=false
Videos and Interactive Visualizations

• For data recorded over time, videos can be useful:
  – Map colouring over time.

• There are also lots of neat interactive visualization methods:
  – Sale date for most expensive paintings.
  – Global map of wind, weather, and oceans.
  – Many examples here.
Hamming Distance vs. Jaccard Coefficient

- These vectors agree in 2 positions.
  - Normalizing Hamming distance by vector length, similarity is 2/9.

- If we’re really interested in predicting 1s, we could find set of 1s in both and compute Jaccard:
  - A -> \{1,2,3,6\}, B -> \{4,5,9\}
  - No intersection so Jaccard similarity is actually 0.
Hamming Distance vs. Jaccard Coefficient

- Let’s say we want to find the tumour in an MR image.
- We have an expert label (top) and a prediction from our ML system (bottom).

- The normalized Hamming distance between the predictions at each pixel is 0.91. This sounds good, but since there are so many non-tumour pixels this is misleading.
- The ML system predicts a much bigger tumour so hasn’t done well. The Jaccard coefficient between the two sets of tumour pixels is only 0.11 so reflects this.
Coupon Collecting

- Consider trying to collect 50 uniformly-distributed states, drawing at random.
- The probability of getting a new state if there ‘x’ states left: \( p = \frac{x}{50} \).
- So expected number of samples before next “success” (getting a new state) is \( \frac{50}{x} \).
  (mean of geometric random variable with \( p = \frac{x}{50} \))
- So the expected number of draws is the sum of \( \frac{50}{x} \) for \( x = 1:50 \).
- For ‘n’ states instead of 50, summing until you have all ‘n’ gives:
  \[
  \sum_{i=1}^{n} \frac{n}{i} = n \sum_{i=1}^{n} \frac{1}{i} \leq n \left( 1 + \log(n) \right) = O(n \log n)
  \]
Huge Datasets and Parallel/Distributed Computation

• Most sufficient statistics can be computed in linear time.
• For example, the mean of ‘n’ numbers is computed as:

\[
\text{mean}(x_1, x_2, x_3, \ldots, x_n) = \frac{x_1 + x_2 + x_3 + \cdots + x_n}{n}
\]

• This costs \(O(n)\), which is great.

• But if ‘n’ is really big, we can go even faster with parallel computing...
Huge Datasets and Parallel/Distributed Computation

- Computing the mean with **multiple cores**:
  - Each of the ‘c’ cores computes the sum of $O(n/c)$ of the data:

\[
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix}
\rightarrow
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_{n/c}
\end{bmatrix}
\rightarrow \text{Sum 1}
\]

\[
\begin{bmatrix}
x_{n/c+1} \\
x_{n/c+2} \\
\vdots \\
x_{2n/c}
\end{bmatrix}
\rightarrow \text{Sum 2}
\]

\[
\begin{bmatrix}
x_{(c-1)n/c+1} \\
x_{(c-1)n/c+2} \\
\vdots \\
x_n
\end{bmatrix}
\rightarrow \text{Sum C}
\]
Huge Datasets and Parallel/Distributed Computation

• Computing the mean with multiple cores:
  – Each of the ‘c’ cores computes the sum of $O(n/c)$ of the data:
  – Add up the ‘c’ results from each core to get the mean.

$$\begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_n \end{bmatrix} \rightarrow \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n/c} \end{bmatrix} \rightarrow \text{Sum}_1$$

$$\begin{bmatrix} x_{n/c+1} \\ x_{n/c+2} \\ \vdots \\ x_{2n/c} \end{bmatrix} \rightarrow \text{Sum}_2$$

$$\begin{bmatrix} x_{(c-1)n/c+1} \\ x_{(c-1)n/c+2} \\ \vdots \\ x_n \end{bmatrix} \rightarrow \text{Sum}_C$$

$$\text{mean}(x) = \frac{\text{sum}_1 + \text{sum}_2 + \cdots + \text{sum}_C}{n}$$
Huge Datasets and Parallel/Distributed Computation

• Computing the mean with multiple cores:
  – Each of the ‘c’ cores computes the sum of $O(n/c)$ of the data.
  – Add up the ‘c’ results from each core to get the mean.
  – Cost is only $O(n/c + c)$, which can be much faster for large ‘n’.

• This assumes cores can access data in parallel (not always true).
• Can reduce cost to $O(n/c)$ by having cores write to same register.
  – But need to “lock” the register and might effectively cost $O(n)$. 
Huge Datasets and Parallel/Distributed Computation

• Sometimes ‘n’ is so big that data can’t fit on one computer.

• In this case the data might be distributed across ‘c’ machines:
  – Hopefully, each machine has $O(n/c)$ of the data.

• We can solve the problem similar to the multi-core case:
  – “Map” step: each machine computes the sum of its data.
  – “Reduce” step: each machine communicates sum to a “master” computer, which adds them together and divides by ‘n’.
Huge Datasets and Parallel/Distributed Computation

• Many problems in DM and ML have this flavour:
  – “Map” computes an operation on the data on each machine (in parallel).
  – “Reduce” combines the results across machines.

https://dzone.com/articles/how-hadoop-mapreduce-works
Huge Datasets and Parallel/Distributed Computation

• Many problems in DM and ML have this flavour:
  – “Map” computes an operation on the data on each machine (in parallel).
  – “Reduce” combines the results across machines.
  – These are standard operations in parallel libraries like MPI.

• Can solve many problems almost ‘c’ times faster with ‘c’ computers.

• To make it up for the high cost communicating across machines:
  – Assumes that most of the computation is in the “map” step.
  – Often need to assume data is already on the computers at the start.
Huge Datasets and Parallel/Distributed Computation

• Another challenge with “Google-sized” datasets:
  – You may need so many computers to store the data, that it’s inevitable that some computers are going to fail.

• Solution to this is a distributed file system.

• Two popular examples are Google’s MapReduce and Hadoop DFS:
  – Store data with redundancy (same data is stored in many places).
    • And assume data isn’t changing too quickly.
  – Have a strategy for restarting “map” operations on computers that fail.
  – Allows fast calculation of more-fancy things than sufficient statistics:
    • Database queries and matrix multiplications.
Data Clean and the Duke Cancer Scandal

• See the Duke cancer scandal:

• Basic sanity checks for data cleanliness show problems in these (and many other) studies:
  – E.g., flipped labels, off-by-one mistakes, switched columns etc.