Naïve Bayes and Hadoop

Shannon Quinn
The word "sustainable" is unsustainable.

**2061:** "Sustainable" occurs an average of once per page.

**2036:** "Sustainable" occurs an average of once per sentence.

**2109:** All sentences are just the word "sustainable" repeated over and over.

**Frequency of use of the word "sustainable" in US English text, as a percentage of all words, by year.**

Source: Google ngrams

http://xkcd.com/ngram-charts/
Coupled Temporal Scoping of Relational Facts.

Understanding Semantic Change of Words Over Centuries.
### Some of A Joint Distribution

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>is</td>
<td>the</td>
<td>effect</td>
<td>of</td>
<td>the</td>
<td>0.00036</td>
</tr>
<tr>
<td>is</td>
<td>the</td>
<td>effect</td>
<td>of</td>
<td>a</td>
<td>0.00034</td>
</tr>
<tr>
<td>.</td>
<td>The</td>
<td>effect</td>
<td>of</td>
<td>this</td>
<td>0.00034</td>
</tr>
<tr>
<td>to</td>
<td>this</td>
<td>effect</td>
<td>:</td>
<td>“”</td>
<td>0.00034</td>
</tr>
<tr>
<td>be</td>
<td>the</td>
<td>effect</td>
<td>of</td>
<td>the</td>
<td>…</td>
</tr>
<tr>
<td></td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>not</td>
<td>the</td>
<td>effect</td>
<td>of</td>
<td>any</td>
<td>0.00024</td>
</tr>
<tr>
<td></td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>does</td>
<td>not</td>
<td>affect</td>
<td>the</td>
<td>general</td>
<td>0.00020</td>
</tr>
<tr>
<td>does</td>
<td>not</td>
<td>affect</td>
<td>the</td>
<td>question</td>
<td>0.00020</td>
</tr>
<tr>
<td>any</td>
<td>manner</td>
<td>affect</td>
<td>the</td>
<td>principle</td>
<td>0.00018</td>
</tr>
</tbody>
</table>
Abstract: Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset. [Kohavi, 1996]

Number of Instances: 48,842
Number of Attributes: 14 (in UCI's copy of dataset) + 1; 3 (here)
Naïve Density Estimation

The problem with the Joint Estimator is that it just mirrors the training data (and is *hard* to compute). We need something which generalizes more usefully.

The naïve model generalizes strongly:

Assume that each attribute is distributed independently of any of the other attributes.
Using the Naïve Distribution

• Once you have a Naïve Distribution you can easily compute any row of the joint distribution.

• Suppose $A$, $B$, $C$ and $D$ are independently distributed. What is $P(A \land \neg B \land C \land \neg D)$?
Using the Naïve Distribution

• Once you have a Naïve Distribution you can easily compute any row of the joint distribution.

• Suppose A, B, C and D are independently distributed. What is $P(A \land \neg B \land C \land \neg D)$?

$P(A) \cdot P(\neg B) \cdot P(C) \cdot P(\neg D)$
Naïve Distribution General Case

• Suppose $X_1, X_2, \ldots, X_d$ are independently distributed.

$$\Pr(X_1 = x_1, \ldots, X_d = x_d) = \Pr(X_1 = x_1) \cdot \ldots \cdot \Pr(X_d = x_d)$$

• So if we have a Naïve Distribution we can construct any row of the implied Joint Distribution on demand.

• How do we learn this?
Learning a Naïve Density Estimator

\[ P(X_i = x_i) = \frac{\# \text{records with } X_i = x_i}{\# \text{records}} \quad \text{MLE} \]

\[ P(X_i = x_i) = \frac{\# \text{records with } X_i = x_i + mq}{\# \text{records} + m} \quad \text{Dirichlet (MAP)} \]

Another trivial learning algorithm!
Indepependently Distributed Data

• Review: A and B are _independent_ if
  – \( \Pr(A,B) = \Pr(A)\Pr(B) \)
  – Sometimes written:
    \[
    A \perp B
    \]

• A and B are _conditionally independent given C_ if
  \( \Pr(A,B \mid C) = \Pr(A \mid C)\Pr(B \mid C) \)
  – Written
    \[
    A \perp B \mid C
    \]
Bayes Classifiers

• If we can do inference over $\Pr(X,Y)$…
• … in particular compute $\Pr(X \mid Y)$ and $\Pr(Y)$.
  – We can compute

$$
\Pr(Y \mid X_1,\ldots,X_d) = \frac{\Pr(X_1,\ldots,X_d \mid Y) \Pr(Y)}{\Pr(X_1,\ldots,X_d)}
$$
Can we make this interesting? Yes!

- Key ideas:
  - Pick the class variable $Y$
  - Instead of estimating $P(X_1, \ldots, X_n, Y) = P(X_1) \ast \cdots \ast P(X_n) \ast Y$, estimate $P(X_1, \ldots, X_n \mid Y) = P(X_1 \mid Y) \ast \cdots \ast P(X_n \mid Y)$
  - Or, assume $P(X_i \mid Y) = \operatorname{Pr}(X_i \mid X_1, \ldots, X_{i-1}, X_{i+1}, \ldots X_n, Y)$
  - Or, that $X_i$ is conditionally independent of every $X_j$, $j \neq i$, given $Y$.

- How to estimate?

MLE
The Naïve Bayes classifier – v1

• Dataset: each example has
  – A unique id \( id \)
    • Why? For debugging the feature extractor
  – \( d \) attributes \( X_1, \ldots, X_d \)
    • Each \( X_i \) takes a discrete value in \( \text{dom}(X_i) \)
  – One class label \( Y \) in \( \text{dom}(Y) \)

• You have a \textit{train} dataset and a \textit{test} dataset
The Naïve Bayes classifier – v1

• You have a \textit{train} dataset and a \textit{test} dataset
• Initialize an “event counter” (hashtable) C
• For each example \textit{id}, \textit{y}, \textit{x}_1,\ldots,\textit{x}_d\text{ in }\textit{train}:
  – C(“\textit{Y}=\text{ANY}”) ++; \hspace{1em} C(“\textit{Y}=\textit{y}”) ++
  – For \textit{j} in 1..\textit{d}:
    • C(“\textit{Y}=\textit{y} \land \textit{X}_j=\textit{x}_j”) ++
• For each example \textit{id}, \textit{y}, \textit{x}_1,\ldots,\textit{x}_d\text{ in }\textit{test}:
  – For each \textit{y}' \text{ in }\text{dom}(\textit{Y}):
    • Compute \( \Pr(y',x_1,\ldots,x_d) = \left( \prod_{j=1}^{d} \Pr(X_j = x_j \mid Y = y') \right) \Pr(Y = y') \)
      
      \[= \left( \prod_{j=1}^{d} \frac{\Pr(X_j = x_j, Y = y')}{\Pr(Y = y')} \right) \Pr(Y = y') \]
      – Return the best \textit{y}'
The Naïve Bayes classifier – v1

• You have a \textit{train} dataset and a \textit{test} dataset
• Initialize an “event counter” (hashtable) \( C \)
• For each example \( id, y, x_1, \ldots, x_d \) in \textit{train}:
  – \( C(\text{"Y=ANY"}) ++; \quad C(\text{"Y=y"}) ++ \)
  – For \( j \) in 1..d:
    • \( C(\text{"Y=y ^ X_j=x_j"}) ++ \)
• For each example \( id, y, x_1, \ldots, x_d \) in \textit{test}:
  – For each \( y' \) in \( \text{dom}(Y) \):
    • Compute \( \Pr(y',x_1,\ldots,x_d) = \left( \prod_{j=1}^{d} \Pr(X_j = x_j \mid Y = y') \right) \Pr(Y = y') \)
      \[= \left( \prod_{j=1}^{d} \frac{C(X_j = x_j \wedge Y = y')}{C(Y = y')} \right) \frac{C(Y = y')}{C(Y = \text{ANY})} \]
      This will overfit, so …
  – Return the best \( y' \)
The Naïve Bayes classifier – v1

- You have a train dataset and a test dataset
- Initialize an “event counter” (hashtable) C
- For each example id, y, x₁,…..,x₅ in train:
  - C(“Y=ANY”) ++;  C(“Y=y”) ++
  - For j in 1..d:
    • C(“Y=y ^ X_j=x_j”) ++
- For each example id, y, x₁,…..,x₅ in test:
  - For each y’ in dom(Y):
    • Compute Pr(y’,x₁,…..,x₅) = \left( \prod_{j=1}^{d} \Pr(X_j = x_j | Y = y') \right) \Pr(Y = y')

= \left( \prod_{j=1}^{d} \frac{C(X_j = x_j ^ Y = y') + m q_j}{C(Y = y') + m} \right) \frac{C(Y = y') + m q_y}{C(Y = ANY) + m}

where:
\[
q_j = \frac{1}{|dom(X_j)|} \quad q_y = \frac{1}{|dom(Y)|} \quad m=1
\]

- Return the best y’

This will underflow, so ...
The Naïve Bayes classifier – v1

- You have a train dataset and a test dataset
- Initialize an “event counter” (hashtable) $C$
- For each example $id, y, x_1, \ldots, x_d$ in train:
  - $C(“Y=ANY”) ++; \quad C(“Y=y”) ++$
  - For $j$ in $1..d$:
    - $C(“Y=y \land X_j=x_j”) ++$
- For each example $id, y, x_1, \ldots, x_d$ in test:
  - For each $y'$ in $\text{dom}(Y)$:
    - Compute $\log \text{Pr}(y',x_1,\ldots,x_d) =$
      \[
      \left( \sum_{j} \log \frac{C(X_j = x_j \land Y = y') + mq_j}{C(Y = y') + m} \right) + \log \frac{C(Y = y') + mq_j}{C(Y = ANY) + m}
      \]
    - Return the best $y'$

where:
- $q_j = 1/|\text{dom}(X_j)|$
- $q_y = 1/|\text{dom}(Y)|$
- $m=1$
The Naïve Bayes classifier – v2

• For text documents, what features do you use?
• One common choice:
  – $X_1 =$ first word in the document
  – $X_2 =$ second word in the document
  – $X_3 =$ third …
  – $X_4 =$ …
  – …
• But: $\Pr(X_{13}=\text{hockey} \mid Y=\text{sports})$ is probably not that different from $\Pr(X_{11}=\text{hockey} \mid Y=\text{sports})$…so instead of treating them as different variables, treat them as different copies of the same variable
The Naïve Bayes classifier – v1

- You have a \textit{train} dataset and a \textit{test} dataset
- Initialize an “event counter” (hashtable) \( C \)
- For each example \( id, y, x_1, \ldots, x_d \) in \textit{train}:
  - \( C(“Y=\text{ANY}”) \) ++; \( C(“Y=y”) \) ++
  - For \( j \) in 1..d:
    - \( C(“Y=y \land X_j=x_j”) \) ++
- For each example \( id, y, x_1, \ldots, x_d \) in \textit{test}:
  - For each \( y’ \) in \textnormal{dom}(Y):
    - Compute \( \Pr(y’,x_1,\ldots,x_d) = \left( \prod_{j=1}^{d} \Pr(X_j = x_j \mid Y = y’) \right) \Pr(Y = y’) \)
      \[ = \left( \prod_{j=1}^{d} \frac{\Pr(X_j = x_j, Y = y’)}{\Pr(Y = y’)} \right) \Pr(Y = y’) \]
  - Return the best \( y’ \)
The Naïve Bayes classifier – v2

- You have a \textit{train} dataset and a \textit{test} dataset
- Initialize an “event counter” (hashtable) C
- For each example \(id, y, x_1, \ldots, x_d\) in \textit{train}:
  - \(C(\"Y=ANY\") \+++ C(\"Y=y\") \+++\)
  - For \(j\) in 1..d:
    - \(C(\"Y=y \land X_j=x_j\") \+++\)
- For each example \(id, y, x_1, \ldots, x_d\) in \textit{test}:
  - For each \(y'\) in \(\text{dom}(Y)\):
    - Compute \(\Pr(y',x_1,\ldots,x_d) = \left(\prod_{j=1}^{d} \Pr(X_j=x_j \mid Y=y')\right) \Pr(Y=y')\)
    \[= \left(\prod_{j=1}^{d} \frac{\Pr(X_j=x_j,Y=y')}{\Pr(Y=y')}\right) \Pr(Y=y')\]
  - Return the best \(y'\)
The Naïve Bayes classifier – v2

• You have a train dataset and a test dataset
• Initialize an “event counter” (hashtable) C
• For each example id, y, x₁,….,x_d in train:
  – C(“Y=ANY”) ++;  C(“Y=y”) ++
  – For j in 1..d:
    • C(“Y=y ^ X=x_j”) ++
• For each example id, y, x₁,….,x_d in test:
  – For each y’ in dom(Y):
    • Compute Pr(y’,x₁,….,x_d) = \left( \prod_{j=1}^{d} \Pr(X = x_j \mid Y = y') \right) \Pr(Y = y')

= \left( \prod_{j=1}^{d} \frac{\Pr(X = x_j, Y = y')}{\Pr(Y = y')} \right) \Pr(Y = y')

  – Return the best y’
The Naïve Bayes classifier – v2

- You have a \textit{train} dataset and a \textit{test} dataset
- Initialize an “event counter” (hashtable) $C$
- For each example $id, y, x_1, \ldots, x_d$ in \textit{train}:
  - $C(\text{"Y=ANY"})$ $++$; $C(\text{"Y=y"})$ $++$
  - For $j$ in $1..d$:
    - $C(\text{"Y=y ^ X=x_j"})$ $++$
- For each example $id, y, x_1, \ldots, x_d$ in \textit{test}:
  - For each $y'$ in $\text{dom}(Y)$:
    - Compute $\log \text{Pr}(y', x_1, \ldots, x_d) =$
      \[
      \left( \sum_j \log \frac{C(X = x_j ^ Y = y') + mq_x}{C(X = \text{ANY} ^ Y = y') + m} \right) + \log \frac{C(Y = y') + mq_y}{C(Y = \text{ANY}) + m}
      \]
  - Return the best $y'$

where:
- $q_j = 1/|V|$
- $q_y = 1/|\text{dom}(Y)|$
- $m=1$
Complexity of Naïve Bayes

- You have a *train* dataset and a *test* dataset
- Initialize an “event counter” (hashtable) \( C \)
- For each example \( i, y, x_1, \ldots, x_d \) in *train*:
  - \( C(\text{"Y=ANY"}) \) ++; \( C(\text{"Y=y"}) \) ++
  - For \( j \) in 1..\( d \):
    - \( C(\text{"Y=y \land X=x_j"}) \) ++
- For each example \( i, y, x_1, \ldots, x_d \) in *test*:
  - For each \( y' \) in \( \text{dom}(Y) \):
    - Compute \( \log \Pr(y', x_1, \ldots, x_d) = \)
      \[
      \left( \sum_j \log \frac{C(X = x_j \land Y = y') + mq_x}{C(X = \text{ANY} \land Y = y') + m} \right) + \log \frac{C(Y = y') + mq_y}{C(Y = \text{ANY}) + m}
      \]
  - Return the best \( y' \)

\[ \text{Complexity: O}(n), \quad n=\text{size of train} \]

Sequential reads

Sequential reads

where:
- \( q_j = 1/|V| \)
- \( q_y = 1/|\text{dom}(Y)| \)
- \( m=1 \)

\[ \text{Complexity: O}(|\text{dom}(Y)| \times n'), \quad n'=\text{size of test} \]
MapReduce!

And Now For Something Completely Different
Inspiration not Plagiarism

• This is not the first lecture ever on Mapreduce
• I borrowed from William Cohen, who borrowed from Alona Fyshe, and she borrowed from:
  • Jimmy Lin
  • Google
    • http://code.google.com/edu/submissions/mapreduce-minilecture/listing.html
    • http://code.google.com/edu/submissions/mapreduce/listing.html
  • Cloudera
    • http://vimeo.com/3584536
Introduction to MapReduce

- 3 main phases
  - Map (send each input record to a key)
  - Sort (put all of one key in the same place)
    - Handled behind the scenes in Hadoop
  - Reduce (operate on each key and its set of values)
- Terms come from functional programming:
  - `map(lambda x:x.upper(), ["shannon","p","quinn"])->[‘SHANNON’, ‘P’, ‘QUINN’]`
  - `reduce(lambda x,y:x+-y, ["shannon","p","quinn"])->”shannon-p-quinn”`
MapReduce overview

Map
Shuffle/Sort
Reduce
MapReduce in slow motion

- Canonical example: Word Count
- Example corpus:

  Joe likes toast

  Jane likes toast with jam

  Joe burnt the toast
MR: slow motion: Map

Input

Joe likes toast  
Map 1

Jane likes toast with jam  
Map 2

Joe burnt the toast  
Map 3

Output

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Joe</td>
<td>1</td>
</tr>
<tr>
<td>likes</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
<tr>
<td>Jane</td>
<td>1</td>
</tr>
<tr>
<td>likes</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
<tr>
<td>with</td>
<td>1</td>
</tr>
<tr>
<td>jam</td>
<td>1</td>
</tr>
<tr>
<td>Joe</td>
<td>1</td>
</tr>
<tr>
<td>burnt</td>
<td>1</td>
</tr>
<tr>
<td>the</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
</tbody>
</table>
MR: slow motion: Sort

**Input**

<table>
<thead>
<tr>
<th>Joe</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>likes</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jane</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>likes</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
<tr>
<td>with</td>
<td>1</td>
</tr>
<tr>
<td>jam</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Joe</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>burnt</td>
<td>1</td>
</tr>
<tr>
<td>the</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
</tbody>
</table>

**Output**

<table>
<thead>
<tr>
<th>Joe</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jane</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>likes</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>likes</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>toast</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>toast</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>with</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>jam</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>burnt</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>the</th>
<th>1</th>
</tr>
</thead>
</table>
### MR: slow mo: Reduce

#### Input

<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joe</td>
<td>1</td>
</tr>
<tr>
<td>Joe</td>
<td>1</td>
</tr>
<tr>
<td>Jane</td>
<td>1</td>
</tr>
<tr>
<td>likes</td>
<td>1</td>
</tr>
<tr>
<td>likes</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
<tr>
<td>toast</td>
<td>1</td>
</tr>
<tr>
<td>with</td>
<td>1</td>
</tr>
<tr>
<td>jam</td>
<td>1</td>
</tr>
<tr>
<td>burnt</td>
<td>1</td>
</tr>
<tr>
<td>the</td>
<td>1</td>
</tr>
</tbody>
</table>

#### Output

<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joe</td>
<td>2</td>
</tr>
<tr>
<td>Jane</td>
<td>1</td>
</tr>
<tr>
<td>likes</td>
<td>2</td>
</tr>
<tr>
<td>toast</td>
<td>3</td>
</tr>
<tr>
<td>with</td>
<td>1</td>
</tr>
<tr>
<td>jam</td>
<td>1</td>
</tr>
<tr>
<td>burnt</td>
<td>1</td>
</tr>
<tr>
<td>the</td>
<td>1</td>
</tr>
</tbody>
</table>
**Hadoop job 201301231150_0778 on hadoopjt**

User: wcohen
Job Name: streamjob6055532903853567038.jar
Job Setup: Successful
Status: Failed
Started at: Wed Jan 30 11:46:47 EST 2013
Failed in: 41sec
Job Cleanup: Successful
Black-listed TaskTrackers: 2

Job Scheduling information: 5 running map tasks using 5 map slots, 0 running reduce tasks using 0 reduce slots.

<table>
<thead>
<tr>
<th>Kind</th>
<th>% Complete</th>
<th>Num Tasks</th>
<th>Pending</th>
<th>Running</th>
<th>Complete</th>
<th>Killed</th>
<th>Failed/Killed Task Attempts</th>
</tr>
</thead>
<tbody>
<tr>
<td>map</td>
<td>100.00%</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>35 / 5</td>
</tr>
<tr>
<td>reduce</td>
<td>00%</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>0 / 0</td>
</tr>
</tbody>
</table>

Job Counters:

<table>
<thead>
<tr>
<th>Counter</th>
<th>Map</th>
<th>Reduce</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rack-local map tasks</td>
<td>0</td>
<td>0</td>
<td>38</td>
</tr>
<tr>
<td>Launched map tasks</td>
<td>0</td>
<td>0</td>
<td>40</td>
</tr>
<tr>
<td>Data-local map tasks</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Failed map tasks</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Map Completion Graph - close
### Hadoop map task list for **job 201301231150 0778** on **hadoop**

#### All Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Complete</th>
<th>Status</th>
<th>Start Time</th>
<th>Finish Time</th>
<th>Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>task 201301231150 0778 m 000000</td>
<td>0.00%</td>
<td>30-Jan-2013 11:47:01</td>
<td>30-Jan-2013 11:47:25 (24sec)</td>
<td><a href="#">Multiple java.lang.RuntimeException: PipeMap errors</a></td>
<td><a href="#">Multiple java.lang.RuntimeException: PipeMap errors</a></td>
</tr>
</tbody>
</table>
### All Task Attempts

<table>
<thead>
<tr>
<th>Task Attempts</th>
<th>Machine</th>
<th>Status</th>
<th>Progress</th>
<th>Start Time</th>
<th>Finish Time</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>attempt_201301231150_0778_m_000000_0</td>
<td>/default-rack/cloud3u12.opencloud</td>
<td>FAILED</td>
<td>0.00%</td>
<td>30-Jan-2013 11:47:01</td>
<td>30-Jan-2013 11:47:06 (4sec)</td>
<td>java.</td>
</tr>
<tr>
<td>attempt_201301231150_0778_m_000000_1</td>
<td>/default-rack/cloud2u28.opencloud</td>
<td>FAILED</td>
<td>0.00%</td>
<td>30-Jan-2013 11:47:07</td>
<td>30-Jan-2013 11:47:11 (4sec)</td>
<td>java.</td>
</tr>
<tr>
<td>Name</td>
<td>Errors</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>java.lang.RuntimeException: PipeMapRed.waitOutputThreads(): subprocess failed with code 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>at org.apache.hadoop.streaming.PipeMapRed.waitOutputThreads(PipeMapRed.java:311)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>at org.apache.hadoop.streaming.PipeMapRed.mapRedFinished(PipeMapRed.java:540)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>at org.apache.hadoop.streaming.PipeMapper.close(PipeMapper.java:132)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>at org.apache.hadoop.mapred.MapRunner.run(MapRunner.java:57)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>at org.apache.hadoop.streaming.PipeMapRunner.run(PipeMapRunner.java:36)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>at org.apache.hadoop.streaming.MapTask.runOldMapper(MapTask.java:358)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>at org.apache.hadoop.streaming.MapTask.run(MapTask.java:307)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>at org.apache.hadoop.streaming.Child.main(Child.java:170)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task Logs</th>
<th>Counters</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last 4KB</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Last 8KB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>All</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Task Logs: 'attempt_201301231150_0778_m_000000_0'

stdout logs

stderr logs

Exception in thread "main" java.lang.NoClassDefFoundError: com/wcohen/StreamNB
Caused by: java.lang.ClassNotFoundException: com.wcohen.StreamNB
    at java.net.URLClassLoader$1.run(URLClassLoader.java:202)
    at java.security.AccessController.doPrivileged(Native Method)
    at java.net.URLClassLoader.findClass(URLClassLoader.java:190)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:306)
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:301)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:247)
Could not find the main class: com.wcohen.StreamNB. Program will exit.
public static void main(String[] args) throws Exception {

    Configuration conf = new Configuration();

    Job job = new Job(conf, "wordcount");

    job.setMapperClass(Map.class);
    job.setReducerClass(Reduce.class);

    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);

    job.setOutputKeyClass(Text.class);

    job.setOutputValueClass(IntWritable.class);

    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));

    job.waitForCompletion(true);
}
public static class Map extends Mapper<LongWritable, Text, Text, IntWritable> {

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(LongWritable key, Text value, Context context) throws <stuff> {
        String line = value.toString();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()) {
            word.set(tokenizer.nextToken());
            context.write(word, one);
        }
    }
}
public static class Reduce extends Reducer<Text, IntWritable, Text, IntWritable> {

public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
        sum += val.get();
    }
    context.write(key, new IntWritable(sum));
}
}
Is any part of this wasteful?

• Remember - moving data around and writing to/reading from disk are very expensive operations

• No reducer can start until:
  • all mappers are done
  • data in its partition has been sorted
Common pitfalls

• You have no control over the order in which reduces are performed
• You have “no” control over the order in which you encounter reduce values
  – Sort of (more later).
• The only ordering you should assume is that Reducers always start after Mappers
Common pitfalls

• You should assume your Maps and Reduces will be taking place on different machines with different memory spaces

• Don’t make a static variable and assume that other processes can read it
  – They can’t.
  – It appear that they can when run locally, but they can’t
  – No really, don’t do this.
Common pitfalls

• Do not communicate between mappers or between reducers
  – overhead is high
  – you don’t know which mappers/reducers are actually running at any given point
  – there’s no easy way to find out what machine they’re running on
    • because you shouldn’t be looking for them anyway
Assignment 1

- Naïve Bayes on Hadoop
- Document classification
- Due Thursday, Jan 29 by 11:59:59pm
Other reminders

• If you haven’t yet signed up for a student presentation, do so!
  – I will be assigning slots starting Friday at 5pm, so sign up before then
  – You need to be on BitBucket to do this

• Mailing list for questions

• Office hours Mondays 9-10:30am