Machine Learning

Today: - Machine learning
        - Density estimation

Next time: - Regression
Machine learning

• Statistical machine learning
  - principles, methods, and algorithms for learning and prediction on the basis of past experience
  - already everywhere: speech recognition, handwritten character recognition, information retrieval, operating systems, security, defense applications, biology ...
Dogs

Dogs

Dogs

Oops...
Steps in learning

• select a (biased) set of possibilities
• adjust predictions based on feedback
• rethink the set of possibilities
Learning and predictions

- We make predictions all the time but rarely investigate the processes underlying our predictions.

- In carrying out scientific research we are also governed by how theories are evaluated.

- To automate the process of making predictions we need to understand in addition how we search and refine “theories.”
Key steps in learning

Data and Assumptions
• what data is available for the learning task?
• what can we assume about the problem?

Representation
• how should we represent the examples to be classified?

Method and Estimation
• what are the possible hypotheses?
• how do we adjust our predictions based on the feedback?

Evaluation
• how well are we doing?

Model Selection
• can we rethink the approach to do even better?
Data and Assumptions

– is this an image classification or dog identification problem?

– how are the examples generated/labeled?
Representation

- There are many ways to represent the same information

- The choice of representation may determine whether the learning task is very easy or very difficult
# Representation

<table>
<thead>
<tr>
<th>GIF</th>
<th>L-JPEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Lempel-Ziv compression</td>
<td>• Two dimensional context</td>
</tr>
<tr>
<td>• Sequence based method</td>
<td>• Image specific method</td>
</tr>
<tr>
<td>• Works well for manually generated images (smooth)</td>
<td>• Works well natural pictures</td>
</tr>
</tbody>
</table>

• Similarly, classification can be performed on both representations, with different success rates depending on the image type
Methods and Estimation

• Examples (binary vectors of length $d = 64$)
  $x = [111111100 \ldots 0000110001101111111]^T$

• Labels $y \in \{-1,1\}$ ("no","yes")

• A linear classifier
  $$\hat{y} = \text{sign}(\theta \cdot x) = \sum \theta_i x_i$$
  where $\Theta$ is a vector of parameters we have to learn.
Methods and Estimation

- 011111100111001000000010000000100111111011101111100 +1
- 00011111100000011000000111000001100111111011111100111 +1
- 111111100000011000000110001111110000001111000001111 -1
- ......

- How do we adjust the parameters $\Theta$ based on the labels?
  \[ \hat{y} = \text{sign}(\theta \cdot x) \]
  We can simply update the parameters whenever we make a mistake:
  \[ \theta \leftarrow \theta + yx \] when prediction was wrong
Evaluation

• How do we measure how well the method is working?

For example: average classification/prediction error as a function of the number of examples seen so far
Model selection

• Can we rethink the approach to do even better?
  – our classifier is limited, can we make it more flexible?
  – is there an entirely different type of classifier that would be more suitable?

• One of the important decisions we have to make is how to model the joint distribution of our variables
Where are we?

- We have recalled the fundamentals of probability
- We have become content with what JDs are and how to use them
- And we even know how to learn JDs from data.
Density estimation

• Density estimators can do many good things…
  – Can spot weird records (anomaly detection)
  – Can be used for clustering (later in course)
  – Ingredient for Bayes Classifiers (later in course)
Density Estimation

- Our Joint Distribution learner is our first example of something called Density Estimation
- A Density Estimator learns a mapping from a set of attributes to a Probability
Density Estimation

- Compare it against the two other major kinds of models:

  - **Classifier**
    - Prediction of categorical output
    - Input Attributes

  - **Density Estimator**
    - Probability
    - Input Attributes

  - **Regressor**
    - Prediction of real-valued output
    - Input Attributes
Evaluating Density Estimation

Test-set criterion for estimating performance on future data*
* See the Decision Tree or Cross Validation lecture for more detail

Input Attributes

Classifier
Prediction of categorical output

Input Attributes

Density Estimator
Probability

Input Attributes

Regressor
Prediction of real-valued output

Test set Accuracy

Test set Accuracy
Evaluating a density estimator

- Given a record $x$, a density estimator $M$ can tell you how likely the record is:

  $$\hat{P}(x|M)$$

- Given a dataset with $R$ records, a density estimator can tell you how likely the dataset is:

  (Under the assumption that all records were independently generated from the Density Estimator’s JD)

$$\hat{P}(\text{dataset}|M) = \hat{P}(x_1 \land x_2 \ldots \land x_R|M) = \prod_{k=1}^{R} \hat{P}(x_k|M)$$
A small dataset: Miles Per Gallon

<table>
<thead>
<tr>
<th>mpg</th>
<th>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>75to78</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>europe</td>
</tr>
</tbody>
</table>

192 Training Set Records

From the UCI repository (thanks to Ross Quinlan)
A small dataset: Miles Per Gallon

<table>
<thead>
<tr>
<th>mpg</th>
<th>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>75to78</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>europe</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mpg</th>
<th>modelyear</th>
<th>maker</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
<td>0.27551</td>
</tr>
<tr>
<td></td>
<td></td>
<td>asia</td>
<td>0.0255102</td>
</tr>
<tr>
<td></td>
<td></td>
<td>europe</td>
<td>0.0153061</td>
</tr>
<tr>
<td>75to77</td>
<td>america</td>
<td>0.153061</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td>0.0255102</td>
<td></td>
</tr>
<tr>
<td></td>
<td>europe</td>
<td>0.0357143</td>
<td></td>
</tr>
<tr>
<td>78to83</td>
<td>america</td>
<td>0.0561224</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td>Never</td>
<td></td>
</tr>
<tr>
<td></td>
<td>europe</td>
<td>Never</td>
<td></td>
</tr>
<tr>
<td>good</td>
<td>70to74</td>
<td>america</td>
<td>0.0102041</td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td>0.0306122</td>
<td></td>
</tr>
<tr>
<td></td>
<td>europe</td>
<td>0.0459184</td>
<td></td>
</tr>
<tr>
<td>75to77</td>
<td>america</td>
<td>0.0306122</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td>0.0408163</td>
<td></td>
</tr>
<tr>
<td></td>
<td>europe</td>
<td>0.0357143</td>
<td></td>
</tr>
<tr>
<td>78to83</td>
<td>america</td>
<td>0.112245</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asia</td>
<td>0.0714286</td>
<td></td>
</tr>
<tr>
<td></td>
<td>europe</td>
<td>0.0357143</td>
<td></td>
</tr>
</tbody>
</table>

192 Training Set Records
A small dataset: Miles Per Gallon

192 Training Set Records

<table>
<thead>
<tr>
<th>mpg</th>
<th>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good 75to78</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad  70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad  75to78</td>
<td>europe</td>
<td></td>
</tr>
<tr>
<td>bad  70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad  70to74</td>
<td>america</td>
<td></td>
</tr>
<tr>
<td>bad  70to74</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad  70to74</td>
<td>asia</td>
<td></td>
</tr>
<tr>
<td>bad  75to78</td>
<td>america</td>
<td></td>
</tr>
</tbody>
</table>

\[
\hat{P}(\text{dataset}|M) = \hat{P}(x_1 \land x_2 \ldots \land x_R|M) = \prod_{k=1}^{R} \hat{P}(x_k|M) \\
= (\text{in this case}) = 3.4 \times 10^{-203}
\]
Log Probabilities

Since probabilities of datasets get so small we usually use log probabilities

\[ \log \hat{P}(\text{dataset}|M) = \log \prod_{k=1}^{R} \hat{P}(x_k|M) = \sum_{k=1}^{R} \log \hat{P}(x_k|M) \]
A small dataset: Miles Per Gallon

<table>
<thead>
<tr>
<th>mpg</th>
<th>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>good</td>
<td>75to78</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>europe</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>asia</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>75to78</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>79to83</td>
<td>america</td>
</tr>
<tr>
<td>bad</td>
<td>70to74</td>
<td>america</td>
</tr>
<tr>
<td>good</td>
<td>75to78</td>
<td>europe</td>
</tr>
</tbody>
</table>

\[
\log \hat{P}(\text{dataset}|M) = \log \prod_{k=1}^{R} \hat{P}(x_k|M) = \sum_{k=1}^{R} \log \hat{P}(x_k|M)
\]

\[
= (\text{in this case}) = -466.19
\]
Summary

• Good news: We have a way to learn a Density Estimator from data.
• Bad news: Density estimation by directly learning the joint is trivial, mindless and dangerous
Using a test set

<table>
<thead>
<tr>
<th>Set Size</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Set</td>
<td>196</td>
</tr>
<tr>
<td>Test Set</td>
<td>196</td>
</tr>
</tbody>
</table>

An independent test set with 196 cars has a worse log likelihood

(actually it’s a billion quintillion quintillion quintillion quintillion times less likely)

…Density estimators can overfit. And the full joint density estimator is the overfittiest of them all!
Overfitting Density Estimators

If this ever happens, it means there are certain combinations that we learn are impossible.

\[
\log \hat{P}(\text{testset}|M) = \log \prod_{k=1}^{R} \hat{P}(x_k|M) = \sum_{k=1}^{R} \log \hat{P}(x_k|M) = -\infty \text{ if for any } k \hat{P}(x_k|M) = 0
\]
Using a test set

<table>
<thead>
<tr>
<th>Set</th>
<th>Size</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Set</td>
<td>196</td>
<td>-466.1905</td>
</tr>
<tr>
<td>Test Set</td>
<td>196</td>
<td>-614.6157</td>
</tr>
</tbody>
</table>

The only reason that our test set didn’t score -infinity is that my code is hard-wired to always predict a probability of at least one in $10^{20}$

*We need Density Estimators that are less prone to overfitting*
Naïve Density Estimation

The problem with the Joint Estimator is that it just mirrors the training data.

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.
Independently Distributed Data

- Let \( x[i] \) denote the \( i \)'th field of record \( x \).
- The independently distributed assumption says that for any \( i, v, u_1 u_2 \ldots u_{i-1} u_{i+1} \ldots u_M \)

\[
P(x[i] = v \mid x[1] = u_1, x[2] = u_2, \ldots x[i-1] = u_{i-1}, x[i+1] = u_{i+1}, \ldots x[M] = u_M) = P(x[i] = v)
\]

- Or in other words, \( x[i] \) is independent of \( \{x[1], x[2], \ldots x[i-1], x[i+1], \ldots x[M]\} \)

- This is often written as

\[
x[i] \perp \{x[1], x[2], \ldots x[i-1], x[i+1], \ldots x[M]\}
\]
Multivalued Independence

For multivalued Random Variables $A$ and $B$, 

$$A \perp B$$

if and only if

$$\forall u, v: P(A = u \mid B = v) = P(A = u)$$

from which you can then prove things like...

$$\forall u, v: P(A = u \land B = v) = P(A = u)P(B = v)$$

$$\forall u, v: P(B = v \mid A = v) = P(B = v)$$
Back to Naïve Density Estimation

- Let $x[i]$ denote the $i$’th field of record $x$:
- Naïve DE assumes $x[i]$ is independent of \{x[1], x[2],..x[i-1], x[i+1],…x[M]\}
- Example:
  - Suppose that each record is generated by randomly shaking a green dice and a red dice
    - Dataset 1: A = red value, B = green value
    - Dataset 2: A = red value, B = sum of values
    - Dataset 3: A = sum of values, B = difference of values
  - Which of these datasets violates the naïve assumption?
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 $P(A^\sim B^\sim C^\sim D)$?
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 \( P(A^{\sim}B^{\sim}C^{\sim}D) \)?

\[
= P(A|\sim B^{\sim}C^{\sim}D) \cdot P(\sim B^{\sim}C^{\sim}D) \\
= P(A) \cdot P(\sim B^{\sim}C^{\sim}D) \\
= P(A) \cdot P(\sim B|C^{\sim}D) \cdot P(C^{\sim}D) \\
= P(A) \cdot P(\sim B) \cdot P(C^{\sim}D) \\
= P(A) \cdot P(\sim B) \cdot P(C|\sim D) \cdot P(\sim D) \\
= P(A) \cdot P(\sim B) \cdot P(C) \cdot P(\sim D)
\]
Naïve Distribution General Case

• Suppose \( x[1], x[2], \ldots x[M] \) are independently distributed.

\[ P(x[1] = u_1, x[2] = u_2, \ldots x[M] = u_M) = \prod_{k=1}^{M} P(x[k] = u_k) \]

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

• So we can do any inference

• But how do we learn a Naïve Density Estimator?
Learning a Naïve Density Estimator

\[ \hat{P}(x[i] = u) = \frac{\# \text{records in which } x[i] = u}{\text{total number of records}} \]

Another trivial learning algorithm!
## Contrast

<table>
<thead>
<tr>
<th>Joint DE</th>
<th>Naïve DE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can model anything</td>
<td>Can model only very boring distributions</td>
</tr>
<tr>
<td>No problem to model “C is a noisy copy of A”</td>
<td>Outside Naïve’s scope</td>
</tr>
<tr>
<td>Given 100 records and more than 6 Boolean attributes</td>
<td>Given 100 records and 10,000 multivalued attributes</td>
</tr>
<tr>
<td>will screw up badly</td>
<td>will be fine</td>
</tr>
</tbody>
</table>
Empirical Results: “Hopeless”

The “hopeless” dataset consists of 40,000 records and 21 Boolean attributes called a, b, c, … u. Each attribute in each record is generated 50-50 randomly as 0 or 1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Model</th>
<th>Parameters</th>
<th>LogLike</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model1</td>
<td>joint</td>
<td>submodel=gauss gausstype=general</td>
<td>-272625</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+/- 301.109</td>
</tr>
<tr>
<td>Model2</td>
<td>naive</td>
<td>submodel=gauss gausstype=general</td>
<td>-58225.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+/- 0.554747</td>
</tr>
</tbody>
</table>

Despite the vast amount of data, “Joint” overfits hopelessly and does much worse.

Average test set log probability during 10 folds of k-fold cross-validation*

Described in a future lectures
Empirical Results: “Logical”

The “logical” dataset consists of 40,000 records and 4 Boolean attributes called a, b, c, d where a, b, c are generated 50-50 randomly as 0 or 1. \( D = A \land \neg C \), except that in 10% of records it is flipped.

The DE learned by “Joint”

The DE learned by “Naive”
Empirical Results: “Logical”

The “logical” dataset consists of 40,000 records and 4 Boolean attributes called a, b, c, d where a, b, c are generated 50-50 randomly as 0 or 1. D = A^¬C, except that in 10% of records it is flipped.

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.11335</td>
</tr>
<tr>
<td>1</td>
<td>0.0126</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.11357</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.01225</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.11297</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.01202</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.1112</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.01215</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.1125</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.111375</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.012275</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.11325</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The DE learned by “Joint”:

Name | Model | Parameters | LogLike
---|---|---|---
Model1 | joint | submodel=gauss, gausstype=general | -9613.79 +/- 26.6781
Model2 | naive | submodel=gauss, gausstype=general | -10763.4 +/- 11.0538

The DE learned by “Naive”:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.500325</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.499675</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.50045</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.49955</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.50165</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.49835</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.69945</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.30055</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Empirical Results: “MPG”
The “MPG” dataset consists of 392 records and 8 attributes.
**Empirical Results: “MPG”**

The “MPG” dataset consists of 392 records and 8 attributes.

<table>
<thead>
<tr>
<th>mpg cylinders</th>
<th>displacement</th>
<th>horsepower</th>
<th>weight</th>
<th>acceleration</th>
<th>modelyear</th>
<th>maker</th>
</tr>
</thead>
<tbody>
<tr>
<td>bad 3</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>70to74</td>
<td>america Never</td>
</tr>
<tr>
<td>good</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A tiny part of the DE learned by "Joint".

<table>
<thead>
<tr>
<th>Name</th>
<th>Model</th>
<th>Parameters</th>
<th>LogLike</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model1</td>
<td>joint</td>
<td>submodel=gauss</td>
<td>-472.486 +/- 77.2184</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gausstype=general</td>
<td></td>
</tr>
<tr>
<td>Model2</td>
<td>naive</td>
<td>submodel=gauss</td>
<td>-257.212 +/- 3.02246</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gausstype=general</td>
<td></td>
</tr>
</tbody>
</table>

The DE learned by "Naive".

<table>
<thead>
<tr>
<th>acceleration</th>
<th>low</th>
<th></th>
<th>0.459184</th>
</tr>
</thead>
<tbody>
<tr>
<td>high</td>
<td></td>
<td></td>
<td>0.540816</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>modelyear</th>
<th>70to74</th>
<th></th>
<th>0.382653</th>
</tr>
</thead>
<tbody>
<tr>
<td>75to77</td>
<td></td>
<td></td>
<td>0.326531</td>
</tr>
<tr>
<td>78to83</td>
<td></td>
<td></td>
<td>0.290816</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>maker</th>
<th>america</th>
<th></th>
<th>0.625</th>
</tr>
</thead>
<tbody>
<tr>
<td>asia</td>
<td></td>
<td></td>
<td>0.201531</td>
</tr>
<tr>
<td>europe</td>
<td></td>
<td></td>
<td>0.173469</td>
</tr>
</tbody>
</table>
Empirical Results: “Weight vs. MPG”
Suppose we train only from the “Weight” and “MPG” attributes

<table>
<thead>
<tr>
<th>mpg</th>
<th>weight</th>
<th>DE Joint</th>
<th>DE Naive</th>
</tr>
</thead>
<tbody>
<tr>
<td>bad</td>
<td>low</td>
<td>0.193878</td>
<td>0.602041</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>0.408163</td>
<td>0.397959</td>
</tr>
<tr>
<td>good</td>
<td>low</td>
<td>0.380102</td>
<td>0.57398</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>0.017857</td>
<td>0.42602</td>
</tr>
</tbody>
</table>

The DE learned by “Joint”

The DE learned by “Naive”
Empirical Results: “Weight vs. MPG”
Suppose we train only from the “Weight” and “MPG” attributes

<table>
<thead>
<tr>
<th>mpg</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>bad</td>
<td>low</td>
</tr>
<tr>
<td></td>
<td>high</td>
</tr>
<tr>
<td>good</td>
<td></td>
</tr>
</tbody>
</table>

| mpg  | bad     | 0.602041 |
|      | good    | 0.397959 |

**Models**

<table>
<thead>
<tr>
<th>Name</th>
<th>Model</th>
<th>Parameters</th>
<th>LogLike</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model1</td>
<td>joint</td>
<td>submodel=gauss gausstype=general</td>
<td>-44.3562 +/- 2.27547</td>
</tr>
<tr>
<td>Model2</td>
<td>naive</td>
<td>submodel=gauss gausstype=general</td>
<td>-53.2231 +/- 0.610411</td>
</tr>
</tbody>
</table>

**The DE learned by “Joint”**

**The DE learned by “Naive”**
Reminder: The Good News

- We have two ways to learn a Density Estimator from data.
- *In other lectures we’ll see vastly more impressive Density Estimators (Mixture Models, Bayesian Networks, Density Trees, Kernel Densities and many more)

- Density estimators can do many good things…
  - Can spot weird records (anomaly detection)
  - Can be used for clustering (later in course)
  - Ingredient for Bayes Classifiers (later in course)
Acknowledgment

These slides are based on slides from previous machine learning classes taught by Andrew Moore at CMU and Tommi Jaakkola at MIT. I thank Andrew and Tommi for letting me use their slides.