Data Summarization and Machine Learning

Kelly Rivers and Stephanie Rosenthal
15-110 Fall 2019
Data Analysis

What kind of analysis is best for your application?

- Counting – how many times does something happen?
- Probabilities – how likely is something to happen?
- Machine Learning – what model can summarize or predict new data?
- Visualization – what does your data look like?

Machine learning is a popular hammer with which to attack problems
NOT ALL DATA ANALYSIS PROBLEMS REQUIRE MACHINE LEARNING!!!
Data Summarization

When you get new data, you should compute some summary information:
• Means (averages)
• Medians (middle value in sorted list)
• Modes (most common value)
• Ranges (low to high, middle half, etc)
• Counts of columns, categories, etc
• Data Types (given and desired)
• Do you have categories? What are they and what do they mean?
• Missing values and why if possible
• Outliers or unexpected values
• Duplicates (most often duplicate rows)
Examples of Summarization in Python

Computing the mean of a list of values (must be numbers):

```python
mean = sum(lst)/len(lst)
```

Computing the median:

```python
median = sorted(lst)[len(lst)//2]
```

Computing the mode:

A) store values (keys) and counts (values) in a dictionary and then iterate through the dictionary to find the largest value

B) import statistics, run mode(lst)
Computing Probabilities

Probability is the likelihood of something happening or some value occurring

\[ P(\text{value}) = \frac{\text{count(value)}}{\text{count(number of rows)}} \]

```python
# lst # of values (e.g., one column of data)
valprob = lst.count(value)/len(lst)

# OR
valcount = 0
for i in lst:
    if i == value:
        valcount += 1
valprob = valcount / len(lst)
```
Computing Probabilities

What is the probability that someone will make a purchase based on the last 6 hours of data?
Computing Joint Probabilities

Sometimes you want to know the likelihood of more than one thing happening at the same time. Typically we look at multiple columns of our data at the same time.

\[ P(v1inCol1 \& v2inCol2) = \frac{\text{count}(v1inCol1 \& v2inCol2)}{\text{count}(\text{number of rows})} \]

\[
\text{col1} \# \text{of values in column1} \\
\text{col2} \# \text{of values in column2 (assume same length as col1)} \\
jointcount = 0 \\
\text{for} \ i \ \text{in} \ \text{range}(\text{len(col1)}):\n\quad \text{if} \ col1[i] == v1inCol1 \ \text{and} \ col2[i] == v2inCol2:\n\qquad jointcount += 1 \\
\text{valprob} = jointcount / \text{len(lst1)}
\]
What is the probability that someone will make a purchase and the time is 11:00?
Computing Conditional Probabilities

Sometimes you want to know the likelihood of something happening or some value occurring **GIVEN** that some other event/value occurred.

$$P(v_{1\text{inCol1}} \mid v_{2\text{inCol2}}) = \frac{\text{count}(v_{1\text{inCol1}} \& v_{2\text{inCol2}})}{\text{count}(v_{2\text{inCol2}})}$$

```python
col1 # of values (e.g., one column of data)
col2 #column2 (same length as col1)
v1v2count = 0
for i in range(len(col2)): #should be the same len as col1
    if col1[i] == v1inCol1 and col2[i] == v2inCol2:
        v1v2count += 1
condprob = v1v2count / col2.count(v2)
```
What is the probability that someone will make a purchase given the time is 11:00?
Summaries and Probabilities

Summarization and probabilities are likely to be the best analysis tools that you can use for most problems.

Always start there. It is needed anyway for most machine learning.
What is Machine Learning?

Study of algorithms that optimize their own performance at some task using experience (data). It is math and statistics applied to data.

Machine Learning is not magic

Goal: learn a mathematical function that best predicts your data
Machine Learning Is Growing

Preferred approach for many problems

• Speech recognition
• Natural language processing
• Medical diagnosis
• Fraud protection
• Advertising
• Weather prediction
• Winning Jeopardy!
Types of Machine Learning

- Classification
- Regression
- Forecasting
- Network Analysis
- Clustering
- Text Analysis
What do we mean by using data?

What is the probability that someone will make a purchase based on the last 6 hours of data?
What do we mean by using data?

What is the probability that someone will make a purchase based on the last 6 hours of data?

<table>
<thead>
<tr>
<th>Time</th>
<th>Purchase Made?</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00</td>
<td>Yes</td>
</tr>
<tr>
<td>10:00</td>
<td>No</td>
</tr>
<tr>
<td>11:00</td>
<td>Yes</td>
</tr>
<tr>
<td>12:00</td>
<td>Yes</td>
</tr>
<tr>
<td>1:00</td>
<td>Yes</td>
</tr>
<tr>
<td>2:00</td>
<td>No</td>
</tr>
</tbody>
</table>
Why is this Machine Learning?

You are learning or approximating a statistic or function that best explains the data

- simple example: overall mean
- based on features that help us make a better estimate
  - Time of day
  - Price of product
Classification

Goal: group data into discrete groups or classes
  • Find most likely class label $y$ given features $X$

Examples
  • Spam filter
  • Text classification
  • Object detection
  • Activity recognition

<table>
<thead>
<tr>
<th>Time of Day</th>
<th>Price</th>
<th>Purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Best Classifier

Idea: compute the probability of label y appearing in the data with the exact features X

Example: What is the probability of a customer buying a $10.00 shirt at 2pm?

Answer: Look at the times when customers looked at $10 at 2pm and count how many purchased.

<table>
<thead>
<tr>
<th>Time of Day</th>
<th>Price</th>
<th>Purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1pm</td>
<td>$5.00</td>
</tr>
<tr>
<td>2</td>
<td>2pm</td>
<td>$10.00</td>
</tr>
<tr>
<td>3</td>
<td>10am</td>
<td>$20.00</td>
</tr>
<tr>
<td>4</td>
<td>11am</td>
<td>$10.00</td>
</tr>
<tr>
<td>5</td>
<td>2pm</td>
<td>$10.00</td>
</tr>
<tr>
<td>6</td>
<td>2pm</td>
<td>$5.00</td>
</tr>
</tbody>
</table>
Best Classifier *(if you have a lot of data)*

Idea: compute the probability of label y appearing in the data with the exact features X

It is hard to have every possible combination of features and you cannot use this method if you do not have every combination.

Question: How many rows of data do you need if you have 10 binary features? 20 binary features?

If you don’t have enough data, then you must use a different algorithm
Types of Classification Algorithms

Naïve Bayes
Logistic Regression
Support Vector Machines
Decision Trees
K-Nearest Neighbors
Neural Networks
... many more...
Logistic Regression

Idea: find a line that divides the data
Instead of counting datapoints, just compare to the dividing line
Logistic Regression

Idea: find a line that divides the data

Works well when a line separates the data

Works well with binary features (0/1’s)
Support Vector Machines

Idea: pick the line that is farthest and equidistant from both classes
Support Vector Machines

Idea: pick the line that is farthest and equidistant from both classes
Support Vector Machines

Idea: pick the line that is farthest and equidistant from both classes
• Assign a penalty to points that are over the line
Support Vector Machines

Idea: pick the line that is farthest and equidistant from both classes

Very popular and accurate classifier

Challenge: can be hard to figure out a good penalty for misclassified points
Decision Trees

Idea: instead of drawing a single complicated line through the data, draw many simpler lines, use a tree structure to represent it.
Decision Trees

Idea: instead of drawing a single complicated line through the data, draw many simpler lines, use a tree structure to represent it.
Decision Trees

Idea: instead of drawing a single complicated line through the data, draw many simpler lines, use a tree structure to represent it.
Decision Trees

Idea: instead of drawing a single complicated line through the data, draw many simpler lines, use a tree structure to represent it.
Decision Trees

Idea: instead of drawing a single complicated line through the data, draw many simpler lines, use a tree structure to represent it.

For best results, make sure tree isn’t very deep.

Many people use “forests” of many trees.
K-Nearest Neighbors

Idea: a new point is likely to share the same label as points around it
K-Nearest Neighbors

Idea: a new point is likely to share the same label as points around it
K-Nearest Neighbors

Idea: a new point is likely to share the same label as points around it

Challenge 1: what does “nearest” mean?
Challenge 2: must compute distance to each point
Your ML Toolbox

Logistic Regression
Support Vector Machine (SVM)
Decision Tree
K-Nearest Neighbors
More Models

Naïve Bayes
Graphical models
HMMs
Neural Networks
Random Forests
Quiz

- Logistic Regression
- Support Vector Machine (SVM)
- Decision Tree
- K-Nearest Neighbors
Quiz

Logistic Regression
Support Vector Machine (SVM)
Decision Tree
K-Nearest Neighbors

<table>
<thead>
<tr>
<th>Time of Day</th>
<th>Color</th>
<th>Purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1pm</td>
<td>Blue</td>
</tr>
<tr>
<td>2</td>
<td>2pm</td>
<td>Green</td>
</tr>
<tr>
<td>3</td>
<td>10am</td>
<td>Blue</td>
</tr>
<tr>
<td>4</td>
<td>11am</td>
<td>Red</td>
</tr>
<tr>
<td>5</td>
<td>2pm</td>
<td>Blue</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>2pm</td>
<td>Blue</td>
</tr>
</tbody>
</table>
Quiz

Logistic Regression
Support Vector Machine (SVM)
Decision Tree
K-Nearest Neighbors
Regression

Tries to draw a trend line through the data

\[ y = \beta_0 + \beta_1 x \]
Regression

Goal: Predict a numerical value or time.

Examples

• Stock market prediction
• Weather temperature prediction
• Webpage Visit/Edit count prediction

<table>
<thead>
<tr>
<th></th>
<th>Light Sensor</th>
<th>Light Sen2</th>
<th>LED</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>230</td>
<td>240</td>
<td>150</td>
</tr>
<tr>
<td>2</td>
<td>300</td>
<td>350</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>4</td>
<td>500</td>
<td>450</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>400</td>
<td>300</td>
<td>200</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>0</td>
<td>50</td>
<td>200</td>
</tr>
</tbody>
</table>
Types of Regression Algorithms

Linear Regression
Support Vector Regression
More, but I won’t talk about them
  • Decision Tree
  • KNN
Regression Basics

\[ y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots \]

y is the **dependent variable**, outcome, response

x’s are the **independent variables**, predictors, or explanatory variables

\( \beta \)'s are the **weights** of the independent variables

We use linear combinations of variables as an **approximation of true model**
Regression: Linear Regression

Idea: Find a line that minimizes the distance of the points to the line
Regression: Support Vector Regression

Idea: The best line has most data points fall within a band around it
SV Regression – most points fall within a band
SV Machine Classifier – most points fall outside of the band
Regression

Linear regression is a very general algorithm and often works well. Support vector regression tends to produce regressions with more but smaller residuals.

Challenges:

• Both algorithms require at least as many data points as there are features to solve for weights.
• Both algorithms assume errors in estimation are independent and have constant variance.
  • May not produce accurate estimations if variance grows with a feature or errors are not independent (due to measurement issues, correlation, etc.)
**Doing Machine Learning**

What do you need in order to do machine learning?

- Your features (columns) computed for all rows of your data
- The expected “ground truth” result that should be computed for each row

Machine learning algorithms need **training data** (experience) to allow it to **optimize** (perfect) the model, compute probabilities, etc.

Because it is likely that you will want to **evaluate it more than once**, people set aside a **validation set** to test iteratively.

You need **testing data** to **evaluate** whether it does a good job on one final distinct set of data.
Rules about Training

• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
• YOU CAN’T USE ALL YOUR DATA TO TRAIN
Why?

• The goal of testing is to determine whether your model is a good fit
• But using all your data to train means that it is of course a good fit (the best possible fit that could be optimized)
• There’s no left over data to check whether your assumptions are true
What do you do?

- 70% of data is for training
- 10-20% is for validation (iterating for good results)
- Remainder is one-time use for testing (actual final testing)
Scikit-Learn Training and Testing

Scikit-Learn is a package (sklearn) that computes the mathematics and statistics that are required for each machine learning algorithm.

You still have to:

• Load your data
• Split your training and testing sets
• Tell it what to train and test on respectively
• Interpret the results
Importing and Instantiating

from sklearn import svm
clf = svm.SVC(gamma=0.001, C=100.)

The library name

clf stands for classifier

The Model Type

The Model Type

Instantiate the SVC (support vector classifier) class with 2 params
Training/Fitting

In sklearn, the word for train is “fit”
Each classifier has a fit function that takes the training data and the labels

\[
\text{clf.fit(digits.data[:-1], digits.target[:-1])}
\]

digits is a built in dataset class
data means features
[-1] means don’t use the last line
target means labels
**Testing/Predicting**

In sklearn, the word for test is “predict”

Each classifier has a predict function that takes some testing data and predicts the labels so you can find the accuracy

```python
clf.predict(digits.data[-1:])
```

digits is a built in dataset class

data means features  
[-1:] means use the last line

It will output the labels
Another Example

```python
>>> iris_X_train = iris_X[indices[:-10]]
>>> iris_y_train = iris_y[indices[:-10]]
>>> iris_X_test = iris_X[indices[-10:]]
>>> iris_y_test = iris_y[indices[-10:]]
```

```python
>>> # Create and fit a nearest-neighbor classifier
>>> from sklearn.neighbors import KNeighborsClassifier
>>> knn = KNeighborsClassifier()
>>> knn.fit(iris_X_train, iris_y_train)
```

KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski', metric_params=None, n_jobs=1, n_neighbors=5, p=2, weights='uniform')

```python
>>> knn.predict(iris_X_test)
array([1, 2, 1, 0, 0, 2, 1, 2, 0])
>>> iris_y_test
array([1, 1, 1, 0, 0, 2, 1, 2, 0])
```
Your ML Toolbox with SciKit-Learn

Naïve Bayes
from sklearn.naive_bayes import GaussianNB
clf = GaussianNB() #clf for classifier

Logistic Regression
from sklearn import linear_model
clf = LogisticRegression(C=1e5)

Support Vector Machine (SVM)
from sklearn import svm
clf = svm.SVC()

Decision Tree
from sklearn import tree
clf = tree.DecisionTreeClassifier()

K-Nearest Neighbors
from sklearn.neighbors import NearestNeighbors
clf = NearestNeighbors(n_neighbors=2)

Linear Regression
from sklearn import linear_model
regr = linear_model.LinearRegression()

Support Vector Regression
from sklearn.svm import SVR
svr = SVR(kernel='linear', C=1e3)
Takeaways

• Lots of data summarization techniques
• Machine learning is the use of statistics to predict or model something about the data using optimization
• Different types of machine learning and each of those types have different modeling techniques
• SciKit-Learn is the package in python to do this for you