Lecture Notes for

Machine Learning in Python

Professor Eric Larson

Table Data using Numpy, Pandas

Class Logistics and Agenda

- Canvas? Anaconda Installs?
- In-person versus Zoom and other classes
- My introduction!
- Agenda:
  - Finish Table Data, Numpy
  - Data Quality
  - Attributes Representation
    - documents
  - The Pandas eco-system
    - loading and manipulating attributes
- Needing some more help?
  - **fast.ai** has great, free resources

Class Overview, by topic

Table Data Visualization

Numpy, Pandas, Seaborn
Overviews with some in-depth discussion

Dimension Reduction and Image Processing

Scikit-learn, Scikit Image,
Intuition only, Some mathematics

Linear and Logistic Regression

Numpy, Recreate API for Scikit-learn
Detailed mathematics for simple optimization
intuition for advanced optimization

Neural Networks and Back Prop.

Numpy
Detailed mathematics for NN operations

Wide and Deep Networks

Convolutional Networks

Recurrent Networks

Keras, Tensorflow
Intuition, Detailed implement.

Ethics in Language Models

ConceptNet
Case studies

Table Data

- **Table Data**: Collection of data **instances** and their **features**
  - **Python:** Pandas Dataframe
  - **R:** Data.frame
  - **Matlab:** Table Class
  - **C++**: Trick Question

Objects, records, rows, points, **samples**, cases, entities, **instances**

Attributes, columns, variables, fields, characteristics, **Features**

TID	Pregnant	BMI	Age	Diabetes
1	Y	33.6	41-50	positive
2	N	26.6	31-40	negative
3	Y	23.3	31-40	positive
4	N	28.1	21-30	negative
5	N	43.1	31-40	positive
6	Y	25.6	21-30	negative
7	Y	31.0	21-30	positive
8	Y	35.3	21-30	negative
9	N	30.5	51-60	positive
10	Y	37.6	51-60	positive

Feature Type Representation

	Attribute	Representation Transformation	Comments
Discrete	**Nominal**	Permutation of values only. **one hot encoding or hash function**	If all **employee ID** numbers were reassigned, would it make any difference?
	Ordinal	Order must be preserved `new_value = f(old_value)` where *f* is a monotonic function. **integer**	An attribute encompassing the notion of **good, better best** can be represented equally well by the values {1, 2, 3} or by { 0.5, 1, 10}.
Continuous	**Interval**	`new_value =f(old_value) + b` *f* is monotonic through origin **float**	Thus, the **Fahrenheit** and **Celsius** temperature scales differ in terms of where their zero value is and the size of a unit (degree).
	Ratio	`new_value = f(old_value)` *f* is monotonic through origin **float**	**Length** can be measured in meters or feet, but **zero is zero**

from Tan et al. Introduction to Data Mining

Demo

“Finish”
Jupyter Notebooks

`01_Numpy and Pandas Intro.ipynb`

Data Quality

Review of Feature Data

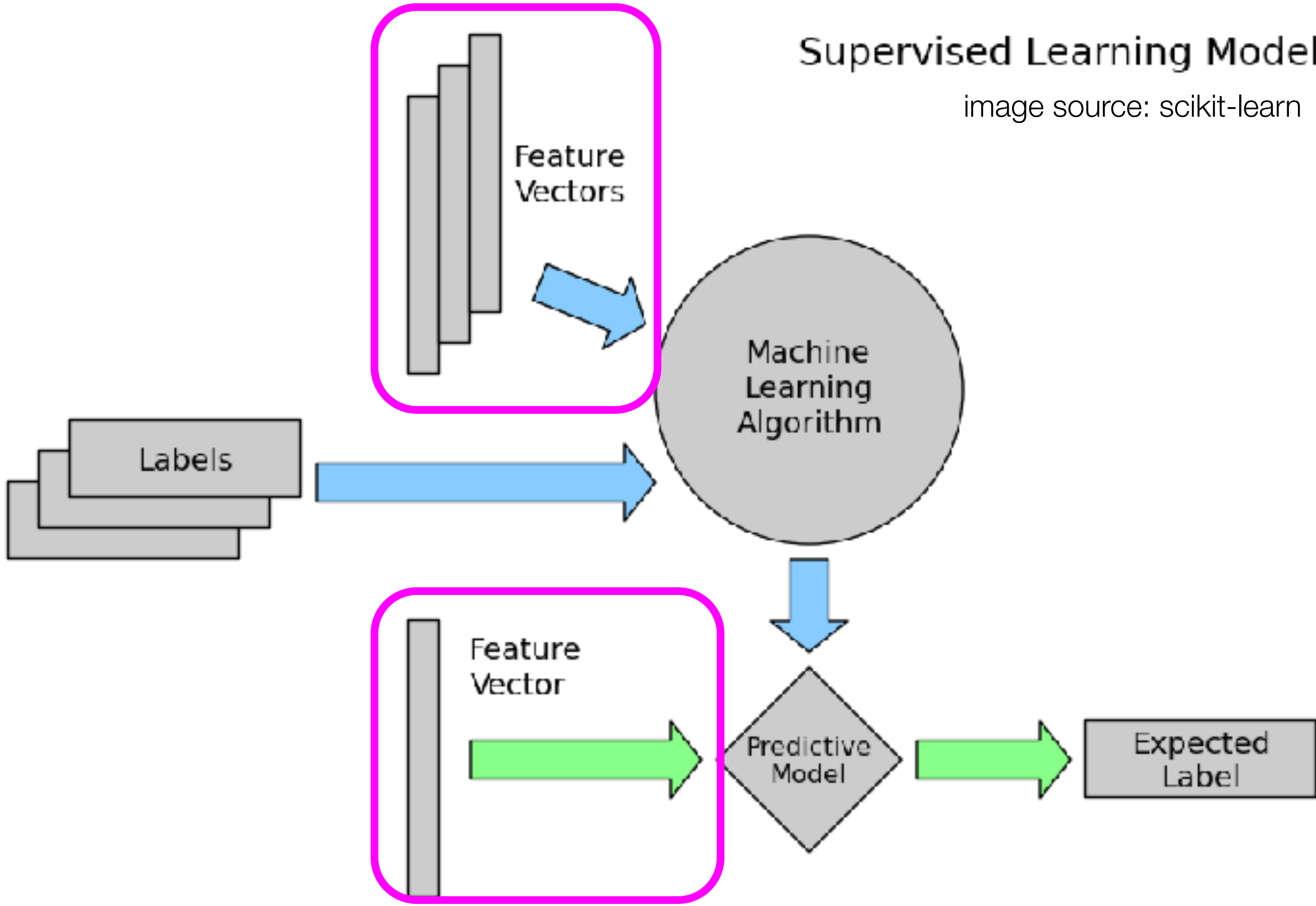

image source: scikit-learn

Data Quality Problems

- Missing
  - Easy to find, NaNs
- Duplicated
  - Easy to find, hard to verify
- Noise or Outlier
  - Hard to define
  - Hard to catch

Information is not collected (e.g., people decline to give their age and weight)

Features **not applicable** (e.g., annual income for children)

UCI ML Repository: 90% of repositories have missing data

TID	*Hair Color*	*Height*	*Age*	*Arrested*
1	Brown	5’2”	23	no
2	Hazel	1.5m	12	no
3	Bl	5	999	no
4	Brown	5’2”	23	no

Handling Issues with Data Quality

- **Eliminate** Instance or Feature
- **Ignore** the Missing Value During Analysis Replace with all possible values (talk about later)
- **Impute** Missing Values

How?

Stats?
mean
median
mode

Imputation

- When is it probably fine to impute missing data:
  - (A) When there is not much missing data
  - (B) When the missing feature is mostly predictable from another feature
  - (C) When there is not much missing data for each subgroup of the data
  - (D) When it is the class you want to predict

Split-Impute-Combine

TID	Pregnant	BMI	Age	Diabetes
1	Y	33.6	41-50	positive
2	N	26.6	31-40	negative
3	Y	23.3	?	positive
4	N	28.1	21-30	negative
5	N	43.1	31-40	positive
6	Y	25.6	21-30	negative
7	Y	31.0	21-30	positive
8	Y	35.3	?	negative
9	N	30.5	51-60	positive
10	Y	37.6	51-60	positive

split: pregnant
split: BMI > 32

TID	Pregnant	BMI	Age	Diabetes
1	Y	>32	41-50	positive
8	Y	>32	?	negative
10	Y	>32	51-60	positive

Mode: none, can't impute

TID	Pregnant	BMI	Age	Diabetes
3	Y	<32	?	positive
6	Y	<32	21-30	negative
7	Y	<32	21-30	positive

Mode: 21-30

K-Nearest Neighbors Imputation

TID	Pregnant	BMI	Age	Diabetes
1	Y	33.6	41-50	positive
2	N	26.6	31-40	negative
3	Y	23.3	?	positive
4	?	28.1	21-30	negative
5	N	43.1	31-40	positive
6	Y	25.6	21-30	negative
7	Y	31.0	21-30	positive
8	Y	35.3	?	negative
9	N	30.5	51-60	positive
10	Y	37.6	51-60	positive

For K=3, find 3 closest neighbors

TID	Preg nant	BMI	Age	Diabetes	Distance
3	Y	23.3	?	positive	0
6	Y	25.6	21-30	negative	(0 + 2.3 + 1)/3
2	N	26.6	31-40	negative	(1 + 3.3 + 1)/3
4	?	28.1	21-30	negative	(4.8 + 1)/2

Imputed Age: 21-30

How to calculate distance?

- Difference for valid features only
- May need to normalize ranges
- Or weight neighbors differently
- Or have min # of valid features
- Euclidean, city-block, etc.

For Next Lecture

- Before next class:
  - verify installation of seaborn, plotly, (and/or bokeh if you want)
  - look at pandas table data and additional tutorials

- Next time: Documents, Data Imputation Demo

Lecture Notes for

Machine Learning in Python

Professor Eric Larson

Table Data using Numpy, Pandas