Applied Machine Learning

Introduction

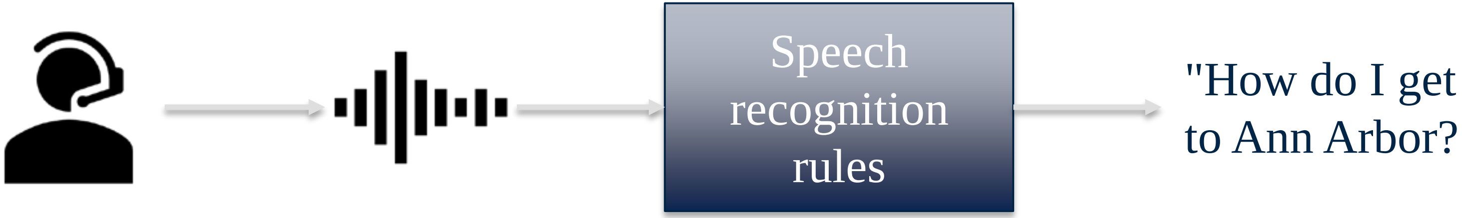
Kevyn Collins-Thompson

Associate Professor of Information & Computer Science
University of Michigan

What is Machine Learning (ML)?

- **The study of computer programs (algorithms) that can learn by example**
- **ML algorithms can generalize from existing examples of a task**
  - *e.g. after seeing a training set of labeled images, an image classifier can figure out how to apply labels accurately to new, previously unseen images*

Speech Recognition



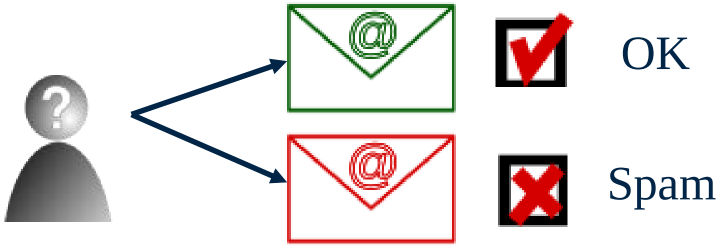
Machine Learning models can learn by example

- Algorithms learn rules from labelled examples
- A set of labelled examples used for learning is called training data.
- The learned rules should also be able to generalize to correctly recognize or predict new examples not in the training set.

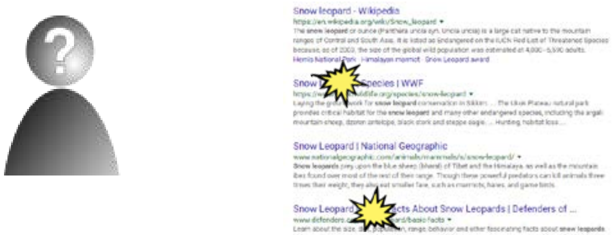
Audio signal	Output text
	How do I get to Ann Arbor?
	Hello!
	Please order me a pizza.

Machine Learning models learn from experience

- **Labeled examples (Email spam detection)**



- **User feedback (Clicks on a search page)**

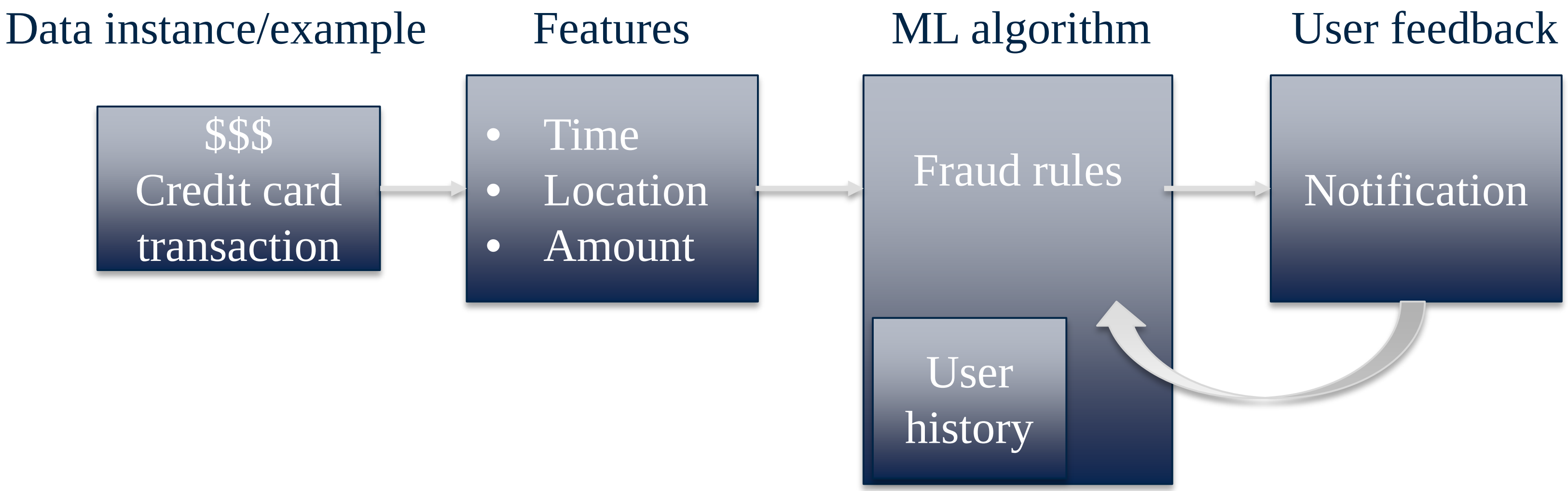


- **Surrounding environment (self-driving cars)**

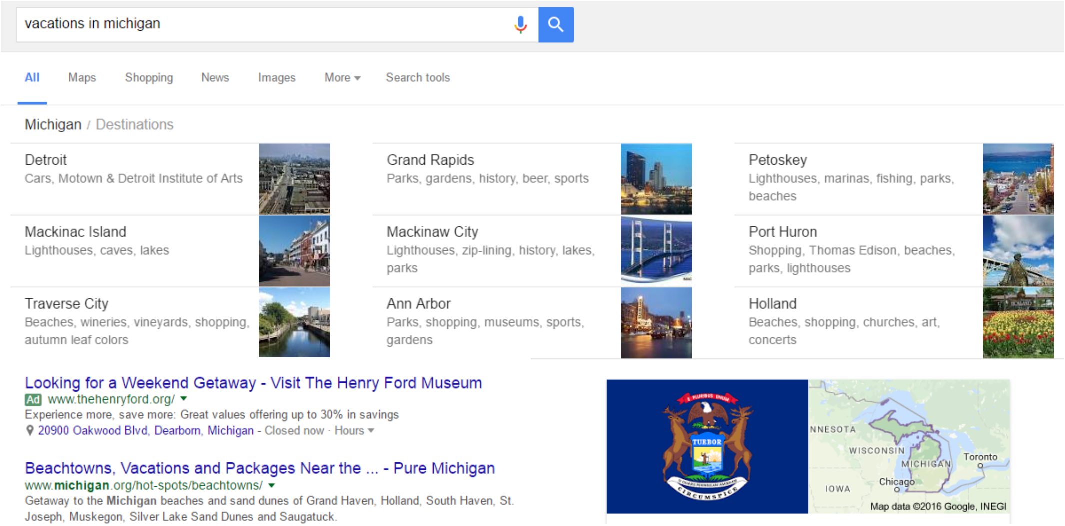
Machine Learning brings together statistics, computer science, and more..

- **Statistical methods**
  - *Infer conclusions from data*
  - *Estimate reliability of predictions*
- **Computer science**
  - *Large-scale computing architectures*
  - *Algorithms for capturing, manipulating, indexing, combining, retrieving and performing predictions on data*
  - *Software pipelines that manage the complexity of multiple subtasks*
- **Economics, biology, psychology**
  - *How can an individual or system efficiently improve their performance in a given environment?*
  - *What is learning and how can it be optimized?*

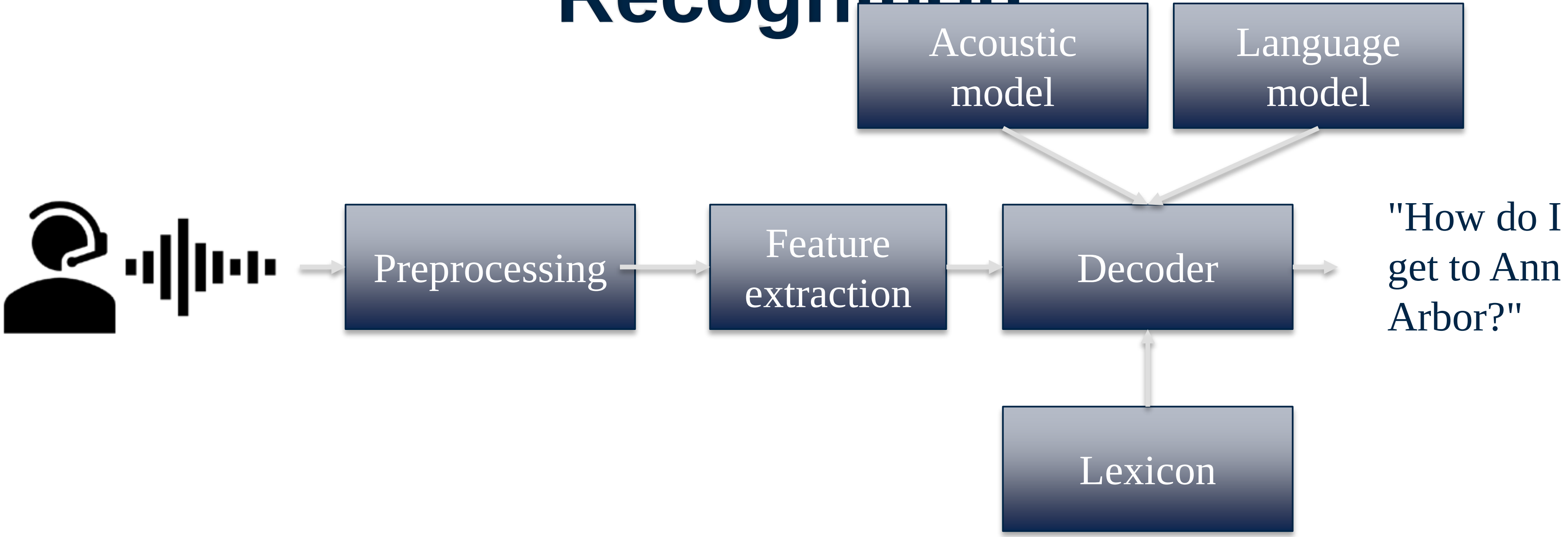
Machine Learning for fraud detection and credit scoring



Web search: query spell-checking, result ranking, content classification and selection, advertising placement



Machine Learning for Speech Recognition



Machine Learning algorithms are at the heart of the information economy

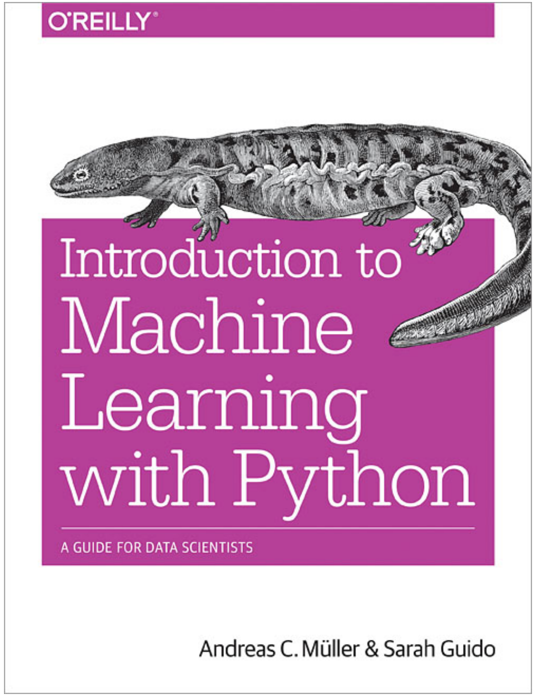
- Finance: fraud detection, credit scoring
- Web search
- Speech recognition
- eCommerce: Product recommendations
- Email spam filtering
- Health applications: drug design and discovery
- Education: Automated essay scoring

What is Applied Machine Learning?

- **Understand basic ML concepts and workflow**
- **How to properly apply 'black-box' machine learning components and features**
- **Learn how to apply machine learning algorithms in Python using the scikit-learn package**

- **What is not covered in this course:**
  - *Underlying theory of statistical machine learning*
  - *Lower-level details of how particular ML components work*
  - *In-depth material on more advanced concepts like deep learning*

Recommended text for this course



Introduction to Machine Learning with Python
A Guide for Data Scientists
By Andreas C. Müller and Sarah Guido

O'Reilly Media

Applied Machine Learning

Key Concepts in Machine Learning

Kevyn Collins-Thompson

Associate Professor of Information & Computer Science
University of Michigan

Key types of Machine Learning problems

Supervised machine learning: Learn to predict target values from labelled data.

- **Classification (target values are discrete classes)**
- **Regression (target values are continuous values)**

Supervised Learning (classification example)

Training set

X Sample		Y Target Value (Label)	
	x_1	Apple	y_1
	x_2	Lemon	y_2
	x_3	Apple	y_3
	x_4	Orange	y_4

Classifier
$f : X \rightarrow Y$

f

At training time, the classifier uses labelled examples to learn rules for recognizing each fruit type.

Future sample

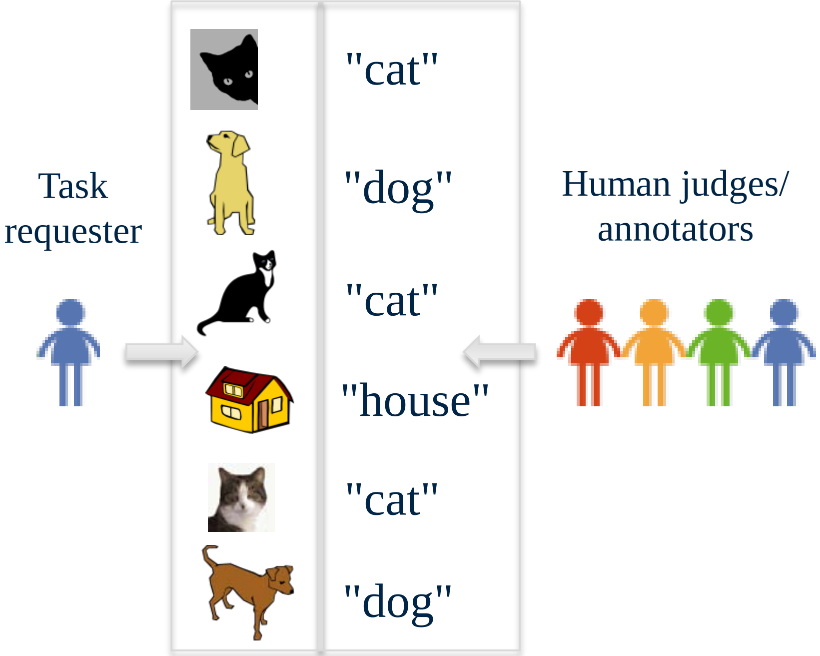
f

Label: Orange

After training, at prediction time, the trained model is used to predict the fruit type for new instances using the learned rules.

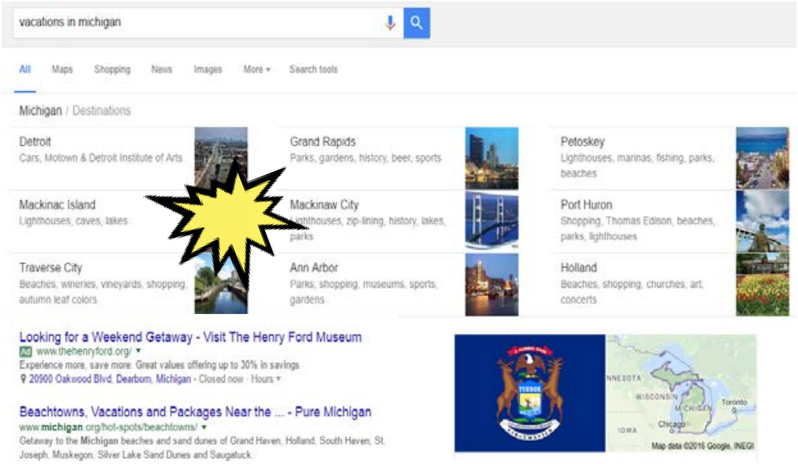
Examples of explicit and implicit label sources

Explicit labels



Crowdsourcing platform

Implicit labels



Clicking and reading the "Mackinac Island" result can be an implicit label for the search engine to learn that "Mackinac Island" is especially relevant for the query `[vacations in michigan]` for that specific user.

Key types of Machine Learning problems

<u>Supervised</u> machine learning: Learn to predict <u>target values</u> from labelled data.

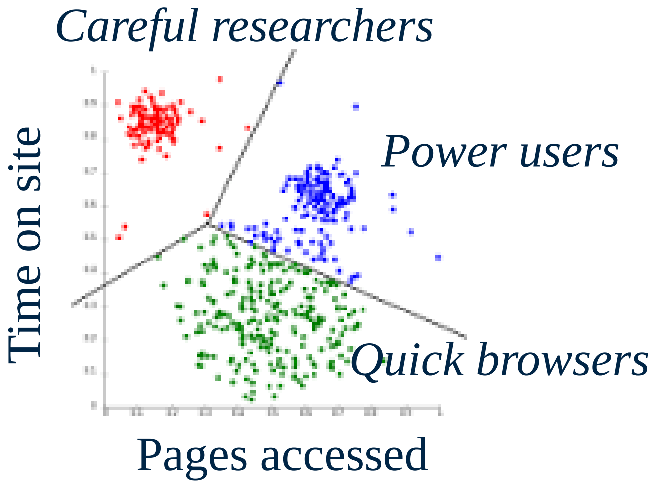
- Classification (target values are discrete classes)
- Regression (target values are continuous values)

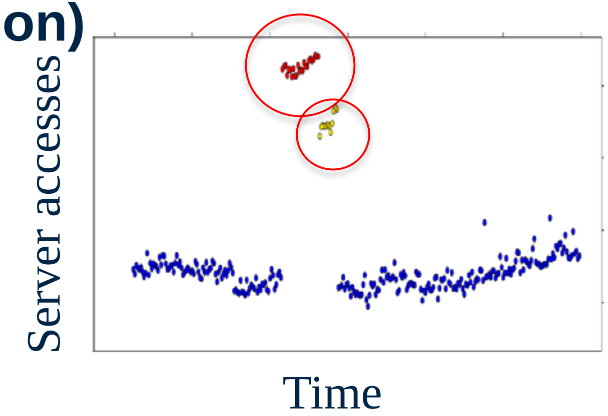
<u>Unsupervised</u> machine learning: Find structure in *unlabeled data*

- Find groups of similar instances in the data (clustering)
- Finding unusual patterns (outlier detection)

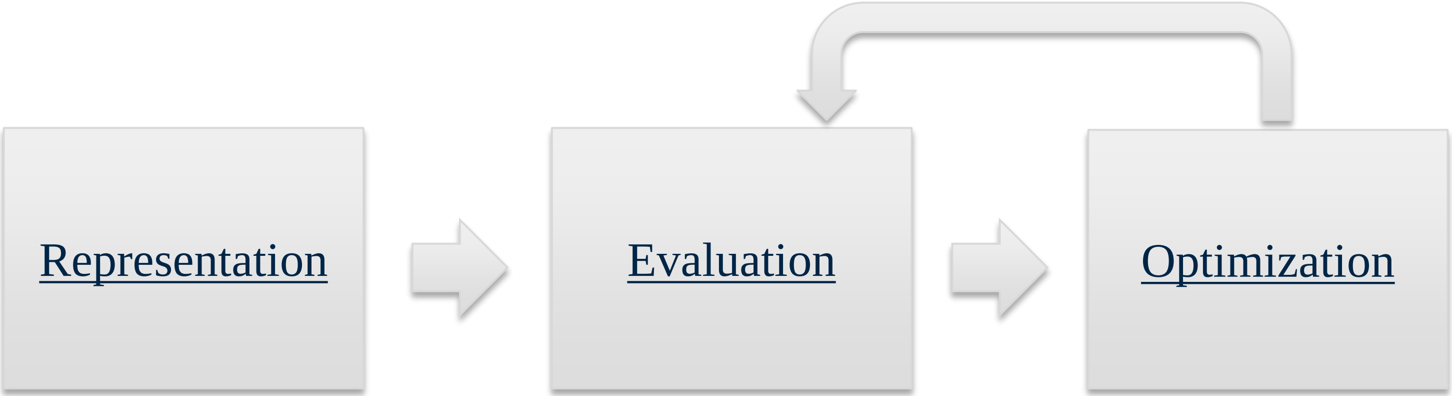
Unsupervised learning: finding useful structure or knowledge in data when no labels are available

- **Finding clusters of similar users (clustering)**
- **Detecting abnormal server access patterns (unsupervised outlier detection)**





A Basic Machine Learning Workflow



Representation

Choose:
- A feature representation
- Type of classifier to use

e.g. image pixels, with
k-nearest neighbor classifier

Evaluation

Choose:
- What criterion distinguishes good vs. bad classifiers?

e.g. % correct predictions on test set

Optimization

Choose:
- How to search for the settings/parameters that give the best classifier for this evaluation criterion

e.g. try a range of values for "k" parameter
in k-nearest neighbor classifier

Feature Representations

Feature representation

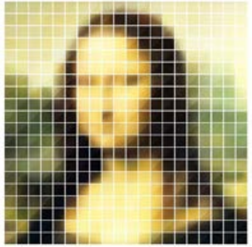
Email

```
To:  Chris Brooks
From:  Daniel Romero
Subject:  Next course offering
Hi Daniel,
Could you please send the outline for the
next course offering?  Thanks! -- Chris
```

Feature	Count
to	1
chris	2
brooks	1
from	1
daniel	2
romero	1
the	2
...	

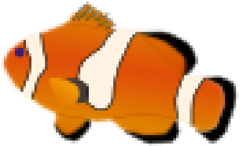
A list of words with their frequency counts

Picture



A matrix of color values (pixels)

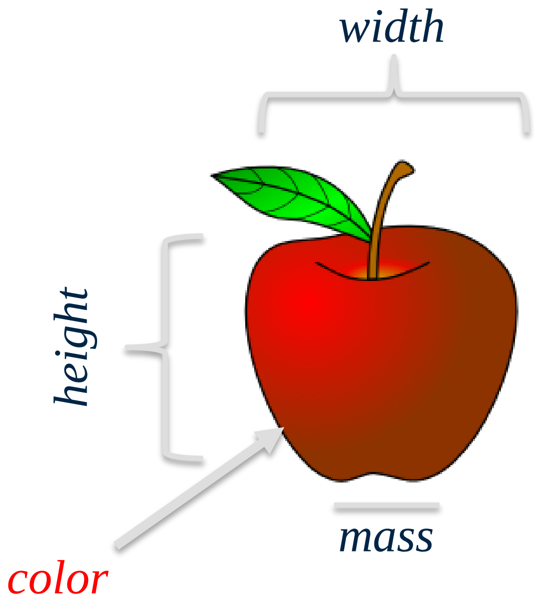
Sea Creatures



Feature	Value
DorsalFin	Yes
MainColor	Orange
Stripes	Yes
StripeColor1	White
StripeColor2	Black
Length	4.3 cm

A set of attribute values

Representing a piece of fruit as an array of features (plus label information)



1. Feature representation

Label information (available in training data only)

Feature representation

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
18	1	apple	cripps_pink	162	7.5	7.1	0.83

2. Learning model

Classifier

Predicted class (apple)

Represent / Train / Evaluate / Refine Cycle

Representation: Extract and select object features

Represent / Train / Evaluate / Refine Cycle

Representation:
Extract and
select object
features

Train models:
Fit the estimator
to the data

Represent / Train / Evaluate / Refine Cycle

Representation: Extract and select object features

Train models: Fit the estimator to the data

Evaluation

Represent / Train / Evaluate / Refine Cycle

Representation: Extract and select object features

Train models: Fit the estimator to the data

Evaluation

Feature and model refinement

Applied Machine Learning

Python Tools for Machine Learning

Kevyn Collins-Thompson

Associate Professor of Information & Computer Science
University of Michigan

scikit-learn: Python Machine Learning Library

- scikit-learn Homepage
  http://scikit-learn.org/
- scikit-learn User Guide
  http://scikit-learn.org/stable/user_guide.html
- scikit-learn API reference
  http://scikit-learn.org/stable/modules/classes.html
- In Python, we typically import classes and functions we need like this:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

SciPy Library: Scientific Computing Tools

http://www.scipy.org/

- **Provides a variety of useful scientific computing tools, including statistical distributions, optimization of functions, linear algebra, and a variety of specialized mathematical functions.**
- **With scikit-learn, it provides support for *sparse matrices*, a way to store large tables that consist mostly of zeros.**
- **Example import:** `import scipy as sp`

NumPy: Scientific Computing Library

http://www.numpy.org/

- Provides fundamental data structures used by scikit-learn, particularly multi-dimensional arrays.
- Typically, data that is input to scikit-learn will be in the form of a NumPy array.
- Example import: `import numpy as np`

Pandas: Data Manipulation and

http://pandas.pydata.org/

- **Provides key data structures like `DataFrame`**
- **Also, support for reading/writing data in different formats**
- **Example import: `import pandas as pd`**

matplotlib and other plotting libraries

matplotlib http://matplotlib.org/

- We typically use matplotlib's **pyplot** module:
```
import matplotlib.pyplot as plt
```
- We also sometimes use the **seaborn** visualization library (http://seaborn.pydata.org/)
```
import seaborn as sn
```
- And sometimes the **graphviz** plotting library:
```
import graphviz
```

Versions of main libraries used in this course

Library name	Minimum version
scikit-learn	0.18.1
scipy	0.19.0
numpy	1.12.1
pandas	0.19.2
matplotlib	2.0.1
seaborn	0.7.1
graphviz	0.7

It’s okay if your versions of these don’t match ours exactly, as long as the version of scikit-learn and other libraries you’re using is the same or greater than listed here.

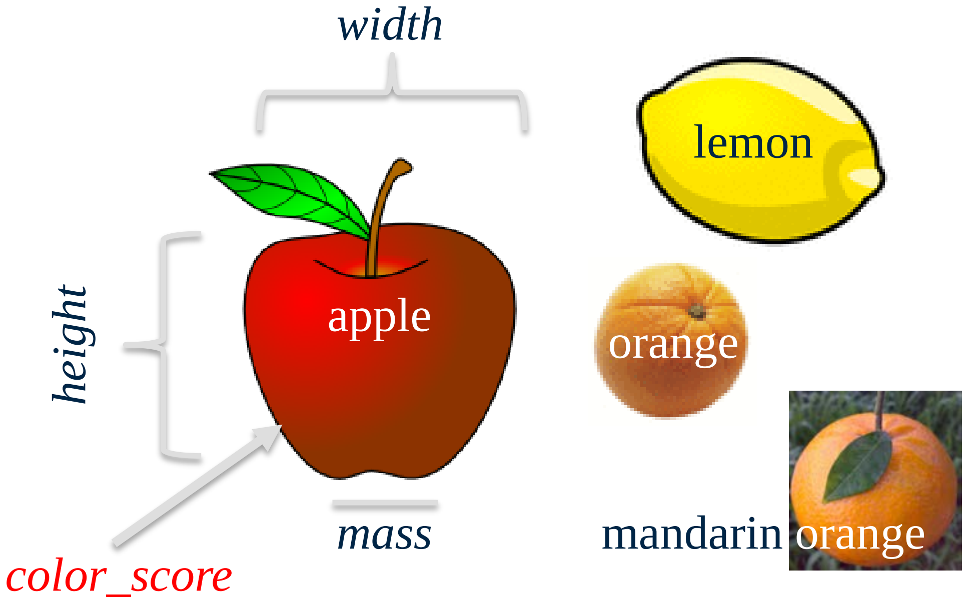
Applied Machine Learning

An Example Machine Learning Problem

Kevyn Collins-Thompson

Associate Professor of Information & Computer Science
University of Michigan

The Fruit Dataset



	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn	172	7.4	7.0	0.89
10	1	apple	braeburn	166	6.9	7.3	0.93
11	1	apple	braeburn	172	7.1	7.6	0.92
12	1	apple	braeburn	154	7.0	7.1	0.88
13	1	apple	golden_delicious	164	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69
15	1	apple	golden_delicious	156	7.7	7.1	0.69
16	1	apple	golden_delicious	156	7.6	7.5	0.67

`fruit_data_with_colors.txt`

Credit: Original version of the fruit dataset created by Dr. Iain Murray, Univ. of Edinburgh

The input data as a table

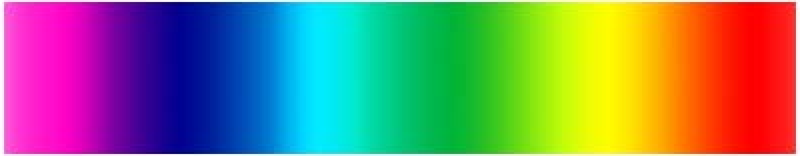
Each row corresponds to a single data instance (sample)

	fruit_label	ruit_name	fruit_subtype	mass	width	height	color_score
0	1	pple	granny_smith	192	8.4	7.3	0.55
1	1	pple	granny_smith	180	8.0	6.8	0.59
2	1	pple	granny_smith	176	7.4	7.2	0.60
3	2	nandarin	mandarin	86	6.2	4.7	0.80
4	2	nandarin	mandarin	84	6.0	4.6	0.79
5	2	nandarin	mandarin	80	5.8	4.3	0.77
6	2	nandarin	mandarin	80	5.9	4.3	0.81
7	2	nandarin	mandarin	76	5.8	4.0	0.81
8	1	pple	braeburn	178	7.1	7.8	0.92
9	1	pple	braeburn	172	7.4	7.0	0.89
1	1	pple	braeburn	166	6.9	7.3	0.93
1	1	pple	braeburn	172	7.1	7.6	0.92
1	1	pple	braeburn	154	7.0	7.1	0.88
1	1	pple	golden_delicious	164	7.3	7.7	0.70
1	1	pple	golden_delicious	152	7.6	7.3	0.69
1	1	pple	golden_delicious	156	7.7	7.1	0.69
1	1	pple	golden_delicious	156	7.6	7.5	0.67
1	1	pple	golden_delicious	168	7.5	7.6	0.73
1	1	pple	cripps_pink	162	7.5	7.1	0.83
1	1	pple	cripps_pink	162	7.4	7.2	0.85
2	1	pple	cripps_pink	160	7.5	7.5	0.86

These four columns contain the features of each data instance (sample)

The `fruit_label` column contains the label for each data instance (sample)

The scale for the (simplistic) `color_score` feature used in the fruit dataset



Color category	`color_score`
Red	0.85 - 1.00
Orange	0.75 - 0.85
Yellow	0.65 - 0.75
Green	0.45 - 0.65

Creating Training and Testing Sets

X

	height	width	mass	color_score
0	7.3	8.4	192	0.55
1	6.8	8.0	180	0.59
2	7.2	7.4	176	0.60
3	4.7	6.2	86	0.80
4	4.6	6.0	84	0.79
5	4.3	5.8	80	0.77
6	4.3	5.9	80	0.81
7	4.0	5.8	76	0.81
8	7.8	7.1	178	0.92
9	7.0			
10	7.3			
11	7.6			
12	7.1			
13	7.7			
14	7.3	7.6	152	0.69
15	7.1	7.7	156	0.69
16	7.5	7.6	156	0.67
17	7.6	7.5	168	0.73
18	7.1	7.5	162	0.83
19	7.2	7.4	162	0.85

y

0	1
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	1
14	1
15	1
16	1
17	1
18	1
19	1

Original data set

```
X_train, X_test, y_train, y_test
   = train_test_split(X, y)
```

X_train

	height	width	mass	color_score
42	7.2	7.2	154	0.82
48	10.1	7.3	174	0.72
7	4.0	5.8	76	0.81
14	7.3	7.6	152	0.69
32	7.0	7.2	164	0.80
49	8.7	5.8	132	0.73
29	7.4	7.0	160	0.81
37	7.3	7.3	15[illegible]	0.79
				0.73
⋮			⋮	
39	7.4	6.8	144	0.75
3	4.7	6.2	86	0.80
0	7.3	8.4	192	0.55
53	8.4	6.0	120	0.74
47	9.7	7.3	196	0.72
44	10.5	7.3	200	0.72

y_train

42	3
48	4
7	2
14	1
32	3
49	4
29	3
37	3
56	4
⋮	
39	3
3	2
0	1
53	4
47	4
44	4

Training set

X_test

	height	width	mass	color_score
26	9.2	9.6	362	0.74
35	7.9	7.1	150	0.75
43	10.3	7.2	194	0.70
28	7.1	6.7	140	0.72
11	7.6	7.1	172	0.92
2	7.2	7.4	176	0.60
34	7.8	7.6	142	0.75
46	10.2	7.3	216	0.71
40	7.5	7.1	154	0.78
22	7.1	7.3	140	0.87
4	4.6	6.0	84	0.79
				0.93
30	7.5	7.1	[illegible]	0.79
41	8.2	7.6	180	0.79
33	8.1	7.5	190	0.74

y_test

26	3
35	3
43	4
28	3
11	1
2	1
34	3
46	4
40	3
22	1
4	2
10	1
30	3
41	3
33	3

Test set

Applied Machine Learning

Examining the Data

Kevyn Collins-Thompson

Associate Professor of Information & Computer Science
University of Michigan

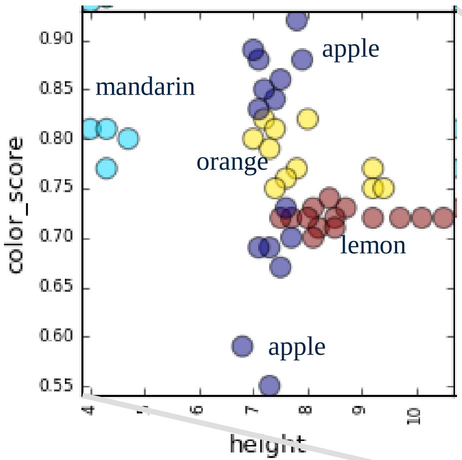
Some reasons why looking at the data initially is important

- Inspecting feature values may help identify what cleaning or preprocessing still needs to be done once you can see the range or distribution of values that is typical for each attribute.
- You might notice missing or noisy data, or inconsistencies such as the wrong data type being used for a column, incorrect units of measurements for a particular column, or that there aren't enough examples of a particular class.
- You may realize that your problem is actually solvable without machine learning.

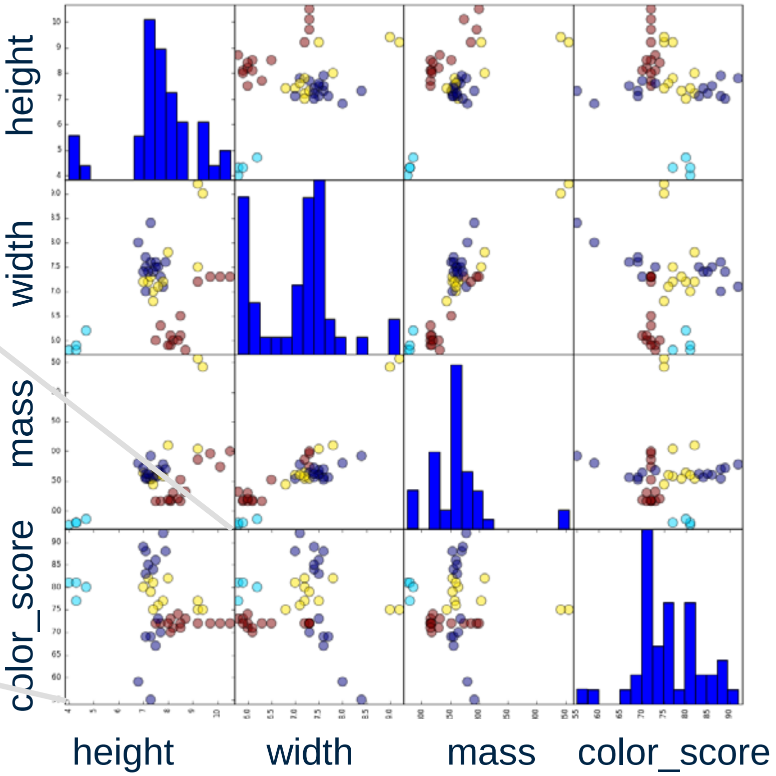
Examples of incorrect or missing feature values

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	192
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	apple	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn		7.4	7.0	0.89
10	1	apple	braeburn		6.9	7.3	0.93
11	1	apple	braeburn		7.1	7.6	0.92
12	1	apple	braeburn		7.0	7.1	0.88
13	1	apple	golden_delicious	164	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69

A pairwise feature scatterplot visualizes the data using all possible pairs of features, with one scatterplot per feature pair, and histograms for each feature along the diagonal



Individual scatterplot plotting all fruits by their **height** and **color_score**.
Colors represent different fruit classes.



```
In [27]: fruits
```

Out[27]:

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn	172	7.4	7.0	0.89
10	1	apple	braeburn	166	6.9	7.3	0.93
11	1	apple	braeburn	172	7.1	7.6	0.92
12	1	apple	braeburn	154	7.0	7.1	0.88
13	1	apple	golden_delicious	164	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69
15	1	apple	golden_delicious	156	7.7	7.1	0.69
16	1	apple	golden_delicious	156	7.6	7.5	0.67

```
In [88]: fruits.shape
Out[88]: (59, 7)
```

```
In [88]: fruits.shape
Out[88]: (59, 7)

In [92]: X_train.shape
Out[92]: (44, 4)

In [93]: X_test.shape
Out[93]: (15, 4)

In [94]: y_train.shape
Out[94]: (44,)

In [95]: y_test.shape
Out[95]: (15,)
```

```
In [96]: X_train
Out[96]:
```

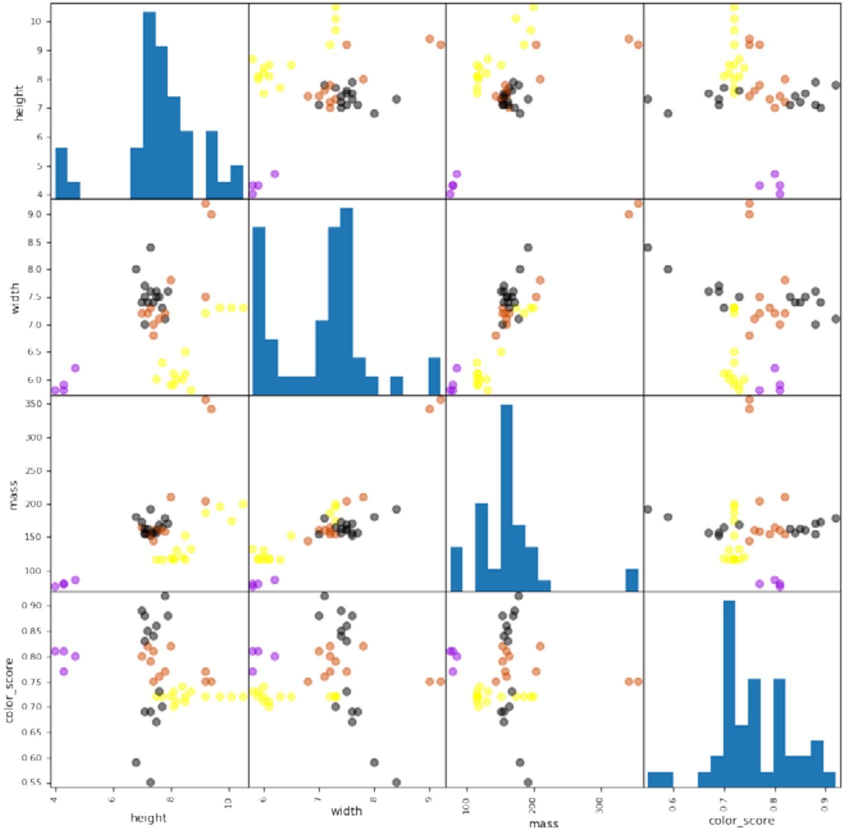
	height	width	mass	color_score
42	7.2	7.2	154	0.82
48	10.1	7.3	174	0.72
7	4.0	5.8	76	0.81
14	7.3	7.6	152	0.69
32	7.0	7.2	164	0.80
49	8.7	5.8	132	0.73
29	7.4	7.0	160	0.81
37	7.3	7.3	154	0.79
56	8.1	5.9	116	0.73
18	7.1	7.5	162	0.83
55	7.7	6.3	116	0.72
27	9.2	7.5	204	0.77
15	7.1	7.7	156	0.69
5	4.3	5.8	80	0.77
31	8.0	7.8	210	0.82
16	7.5	7.6	156	0.67

```
In [98]: y_train
Out[98]: 42    3
48    4
7     2
14    1
32    3
49    4
29    3
37    3
56    4
18    1
55    4
27    3
15    1
5     2
31    3
16    1
50    4
20    1
51    4
8     1
13    1
25    3
17    1
58    4
57    4
52    4
38    3
1     1
12    1
45    4
24    3
6     2
```

```
In [97]: X_test
Out[97]:
```

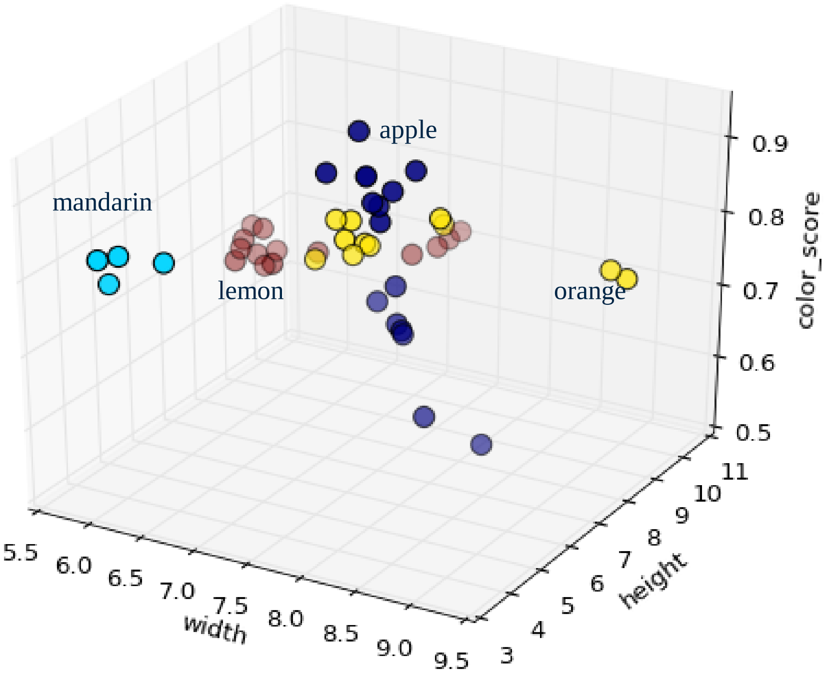
	height	width	mass	color_score
26	9.2	9.6	362	0.74
35	7.9	7.1	150	0.75
43	10.3	7.2	194	0.70
28	7.1	6.7	140	0.72
11	7.6	7.1	172	0.92
2	7.2	7.4	176	0.60
34	7.8	7.6	142	0.75
46	10.2	7.3	216	0.71
40	7.5	7.1	154	0.78
22	7.1	7.3	140	0.87
4	4.6	6.0	84	0.79
10	7.3	6.9	166	0.93
30	7.5	7.1	158	0.79
41	8.2	7.6	180	0.79
33	8.1	7.5	190	0.74

```
In [99]: y_test
Out[99]: 26    3
35    3
43    4
28    3
11    1
2     1
34    3
46    4
40    3
22    1
4     2
10    1
30    3
41    3
33    3
```

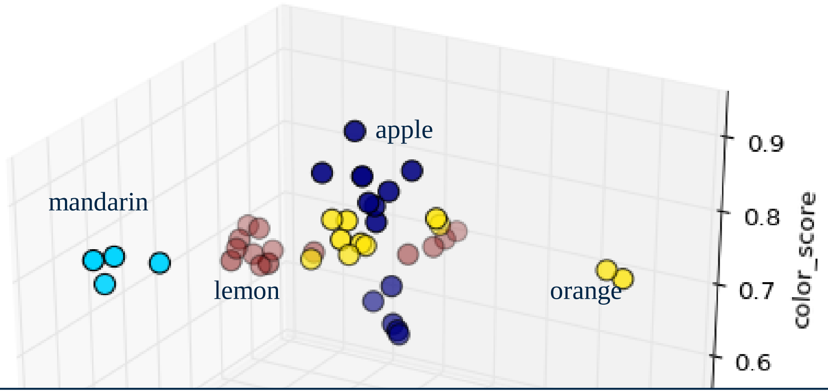



```python
from matplotlib import cm
cmap = cm.get_cmap('gnuplot')
scatter = pd.scatter_matrix(X_train, c= y_train, marker = 'o', s=40, hist_kwds={'bins':15}, figsize=(12,12), cmap=cmap)
```

A three-dimensional feature scatterplot



A three-dimensional feature scatterplot




```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111, projection = '3d')
ax.scatter(X_train['width'], X_train['height'], X_train['color_score'], c = y_train, marker = 'o', s=100)
ax.set_xlabel('width')
ax.set_ylabel('height')
ax.set_zlabel('color_score')
plt.show()
```

Applied Machine Learning

K-Nearest Neighbors Classification

Kevyn Collins-Thompson

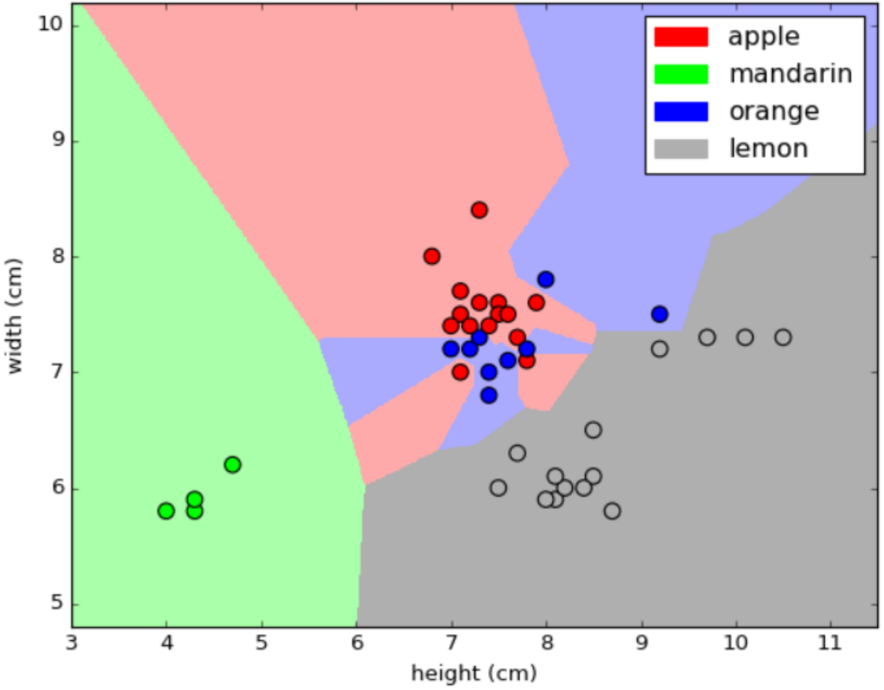
Associate Professor of Information & Computer Science
University of Michigan

The k-Nearest Neighbor (k-NN) Classifier Algorithm

Given a training set X_train with labels y_train, and given a new instance x_test to be classified:

1. Find the most similar instances (let's call them X_NN) to x_test that are in X_train.
2. Get the labels y_NN for the instances in X_NN
3. Predict the label for x_test by combining the labels y_NN
   e.g. simple majority vote

A visual explanation of k-NN



Fruit dataset
Decision boundaries
with k = 1

A nearest neighbor algorithm needs four things specified

1. A distance metric
2. How many 'nearest' neighbors to look at?
3. Optional weighting function on the neighbor points
4. Method for aggregating the classes of neighbor points

A nearest neighbor algorithm needs four things specified

1. A distance metric

   Typically Euclidean (Minkowski with p = 2)

2. How many 'nearest' neighbors to look at?

   e.g. five

3. Optional weighting function on the neighbor points

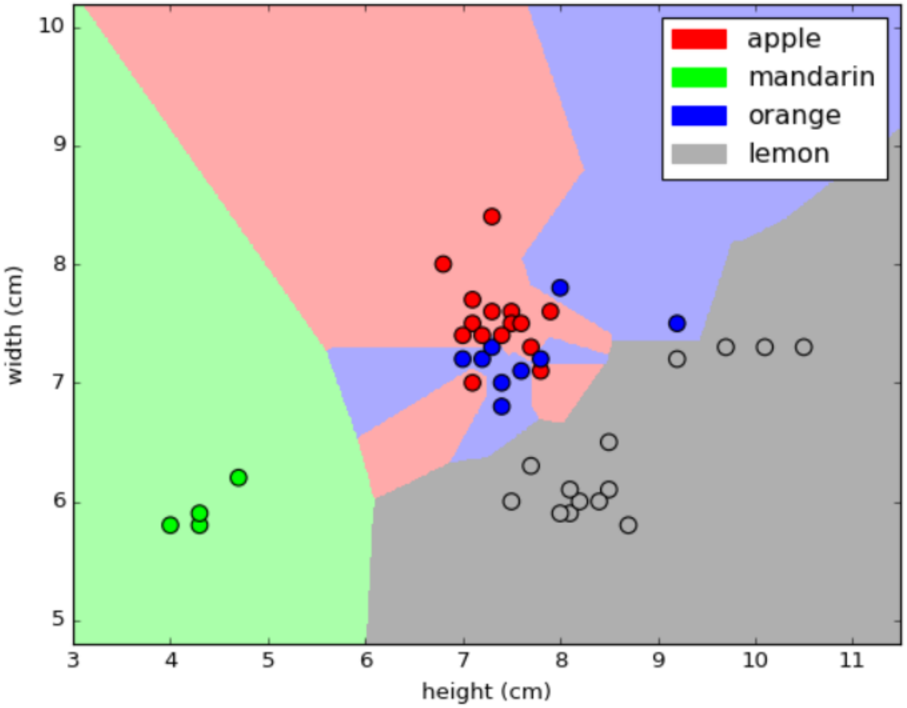
   Ignored

4. How to aggregate the classes of neighbor points

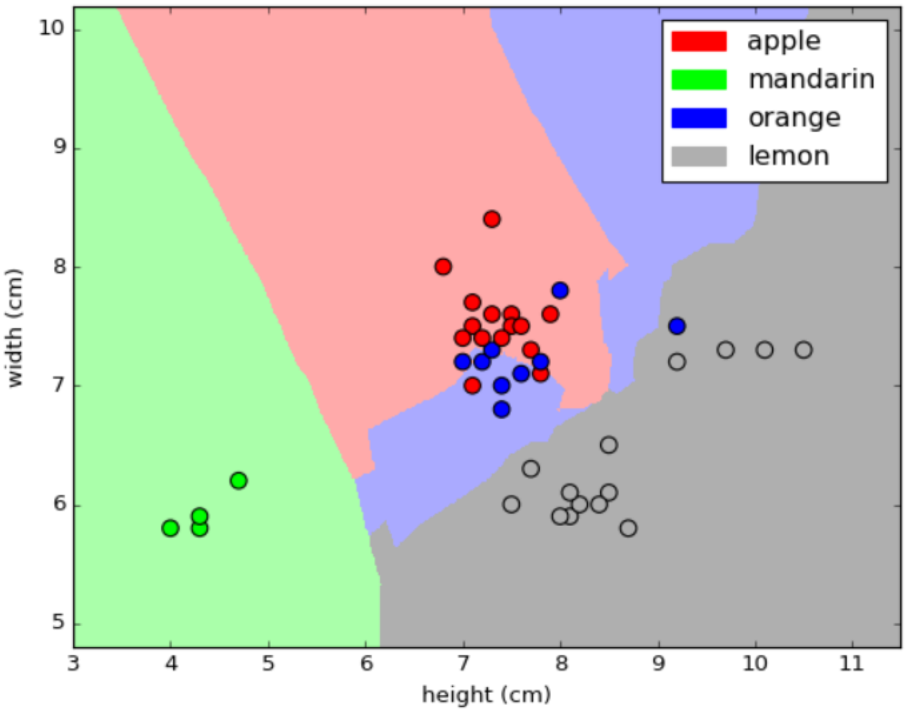
   Simple **majority vote**

   (Class with the most representatives among nearest neighbors)

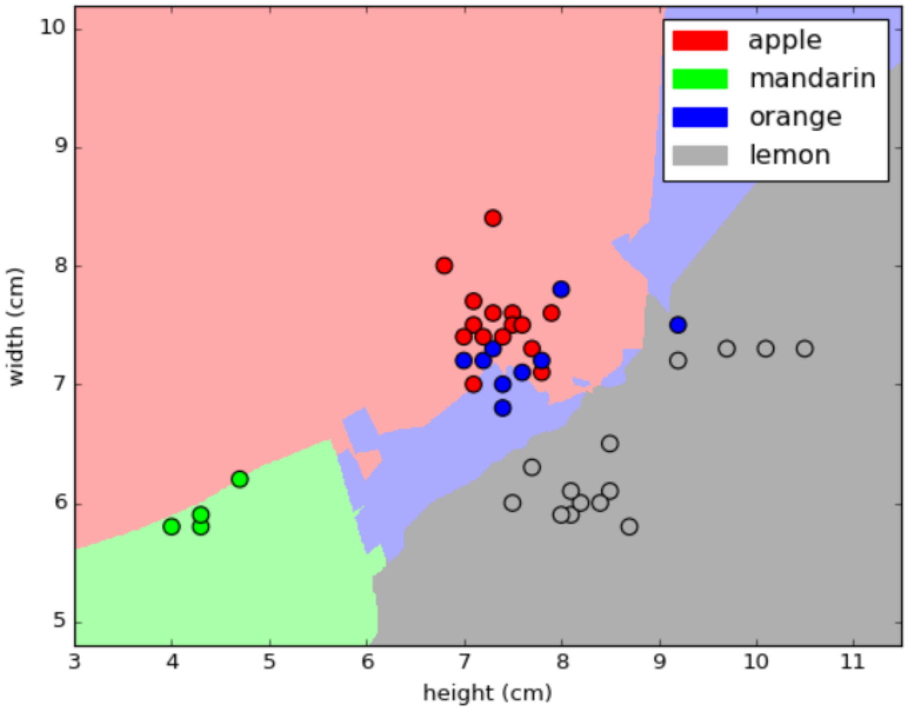
K-nearest neighbors (k=1) for fruit dataset

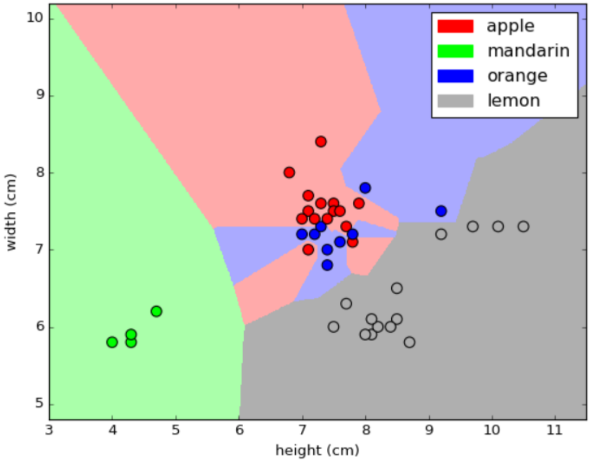


K-nearest neighbors (k=5) for fruit dataset

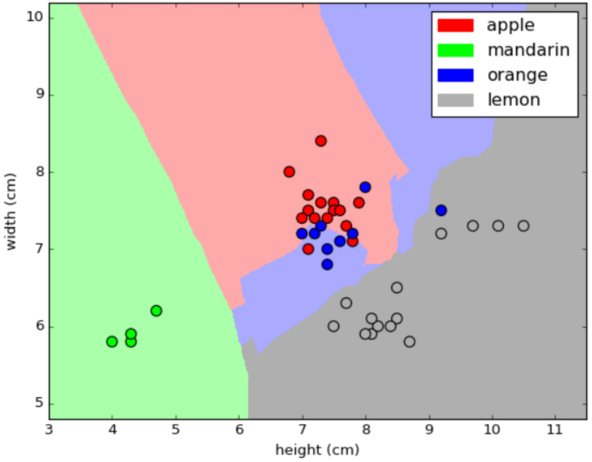


K-nearest neighbors (k=10) for fruit dataset

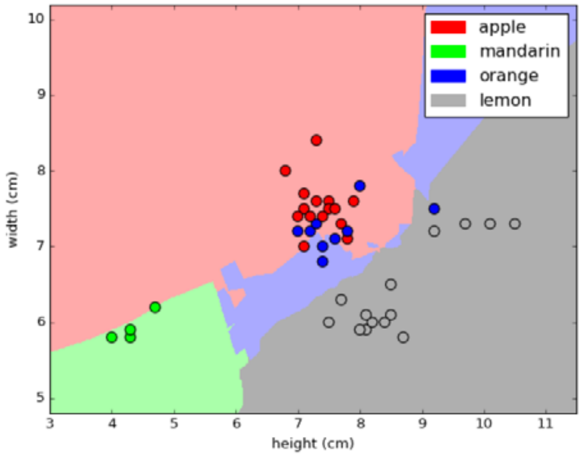



apple
mandarin
orange
lemon
width (cm)
height (cm)
3 4 5 6 7 8 9 10 11
5 6 7 8 9 10

K=1

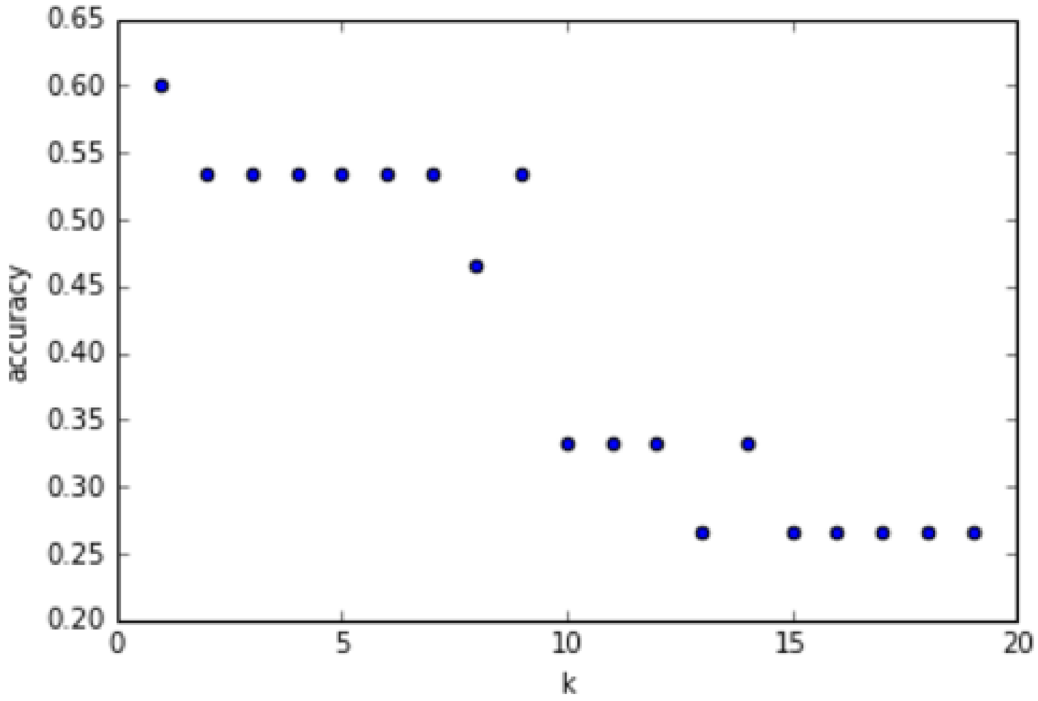

apple
mandarin
orange
lemon
width (cm)
height (cm)
3 4 5 6 7 8 9 10 11
5 6 7 8 9 10

K=5


apple
mandarin
orange
lemon
width (cm)
height (cm)
3 4 5 6 7 8 9 10 11
5 6 7 8 9 10

K=10

How sensitive is k-NN classifier accuracy to the choice of 'k' parameter?



Fruit dataset with 75%/25% train-test split