Introduction to Machine Learning

Chantri Polprasert

Outline

- Why and What?
  - Traditional programming vs machine learning
  - Key Types of Machine Learning
- Supervised Machine Learning
  - Key Concepts
  - K-Nearest Neighbors Classification (KNN)
- Hands on

Type of Analytics

Source: IBM's Introduction to Data Analytics on Coursera

Why Machine Learning?
Early days “intelligent” applications

- Hand Coded rules (rule-based) of “if ” and “else” decisions to process data or adjust to user input.
  - e.g. Spam filter from blacklist words

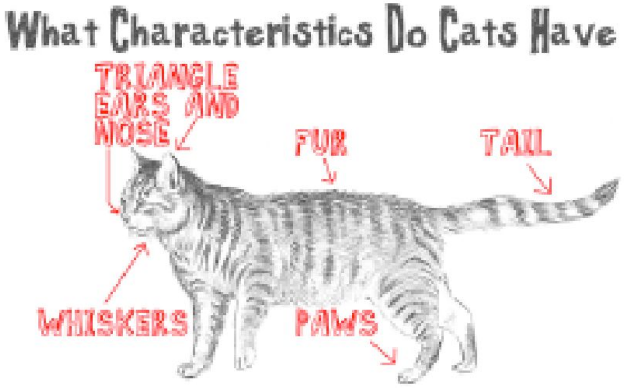
……… free offer
Guarantee ……..
……… win lottery!

- Disadvantages:
  - Static in nature
  - Difficulty in Handling Complexity
  - Manual Updates Required
  - Immutability: changing the tasks slightly might require a rewrite of the whole system

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

- Machine Learning is…… learning from data on its own
- … discovering hidden patterns
- … data-driven decisions
- can adapt to new data



Rule-based vs Machine Learning Intelligent system

	Rule-based	Machine Learning
Description	Well-defined problems, rules are clear and well-understood, structured input, outcomes can be predicted using if-then rules	complex problems involving multiple factors, rules or patterns that change over time
Advantages	• High precision • Ease of use (easy to generate, use , debug and understand) • Speed • Good for static problems	• Adaptability • Self-learning • Scalability • High recall
Drawbacks	• Limited scope (lack learning capabilities) • Immutability (Static and unscalable by nature, can introduce expensive complications when introducing new rules) • Restricted intelligence (as good as the rules set) • Need domain experts to manage rules	• Need high quality labeled data • Complexity • Precision-Recall tradeoff • Black-box

Exercise1

For a task involving sorting a list of numbers, which approach would be the most appropriate?

A. Merge Sort

B. Random Forest model trained on different ranges of number

C. Reinforcement learning

D. Genetic algorithms

Exercise2

You are building a system to recognize handwritten digits (like those on checks or forms). Which method would perform better?

A. Writing custom logic to analyze each pixel in the image

B. A convolutional neural network (CNN)

C. Manually identifying features of each digit

D. Traditional image processing techniques

Exercise3

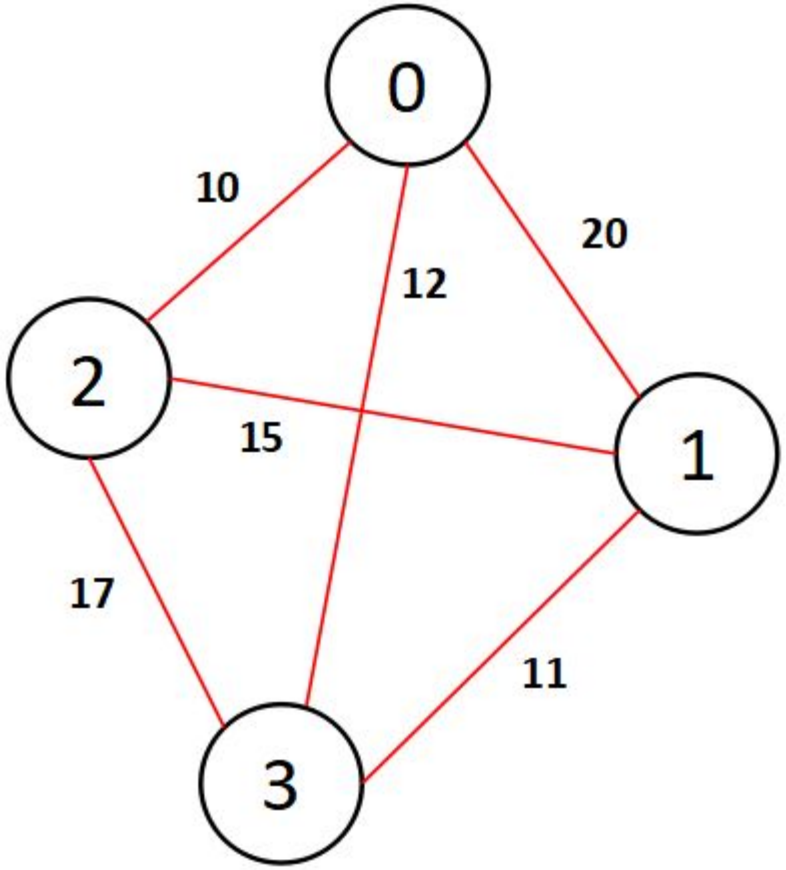
The salesman must travel to 20 cities once before returning home. The distance between each city is given and is assumed to be the same on both directions. To reduce cost, the salesman must find the route that minimizes the total distance travelled. Which method would perform better to find the shortest route.

A. Brute force

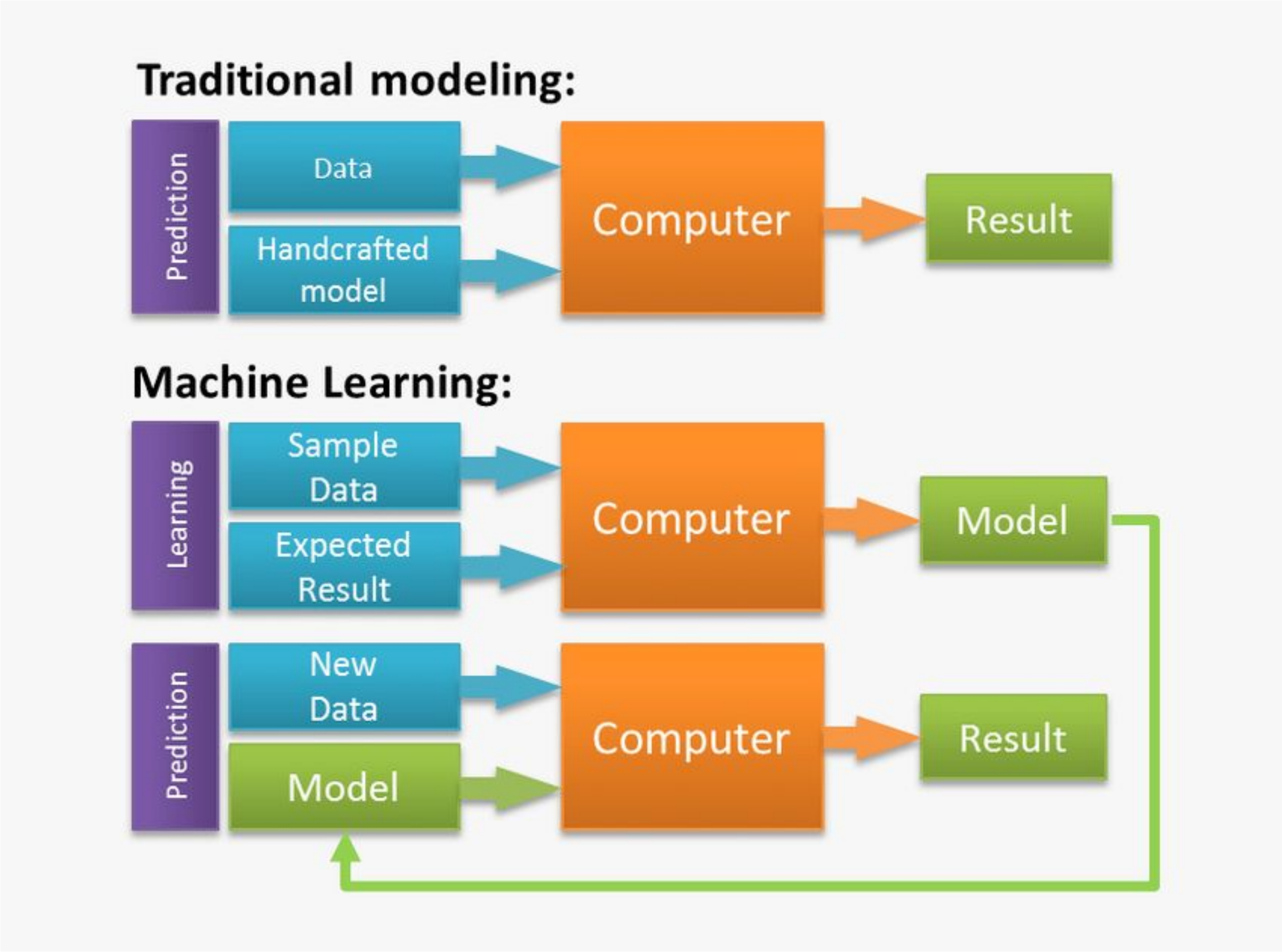
B. Graph neural network trained on the structure of the topology

C. Greedy

D. KNN



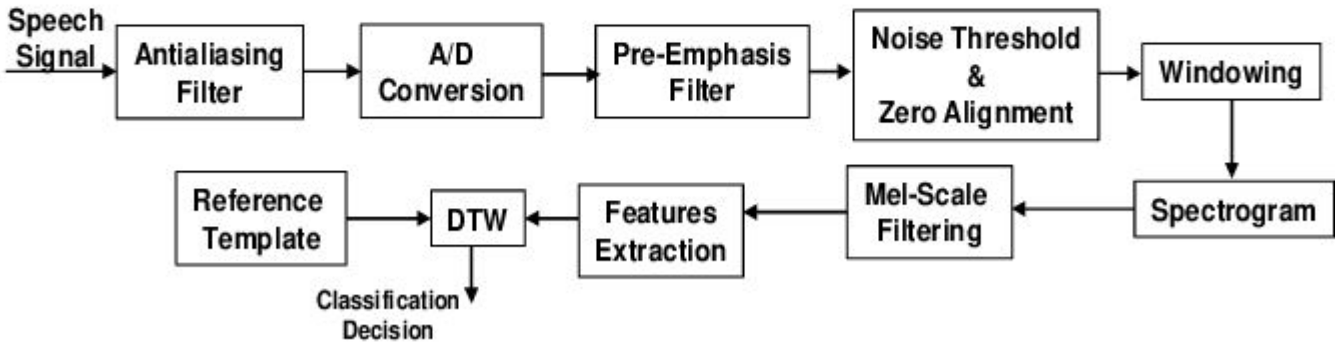
Traditional Programming Vs. Machine Learning



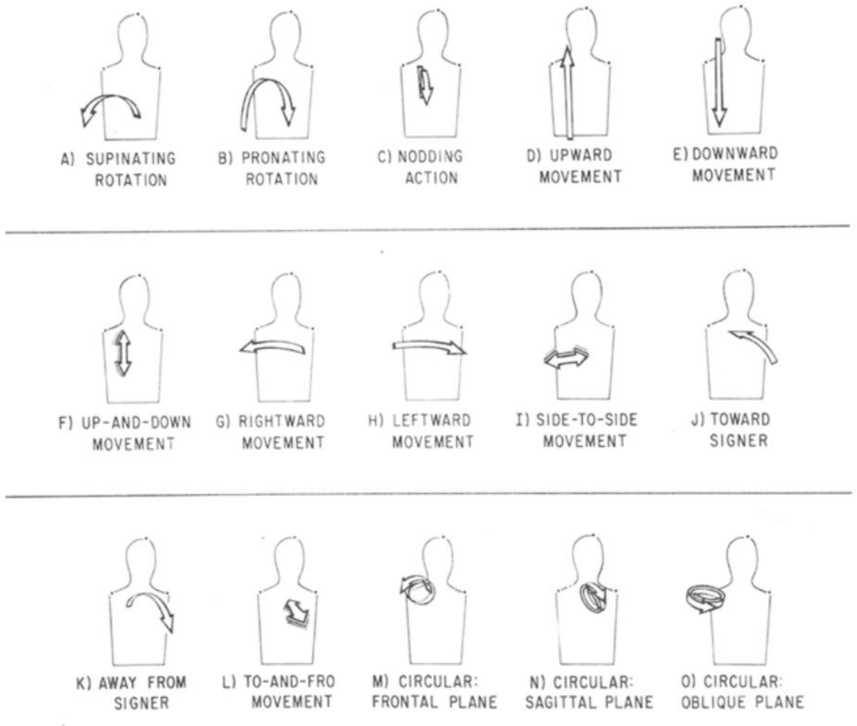
Machine Learning models can learn by example

- Algorithms learn rules from <u>labelled examples.</u>
- A set of labelled examples used for learning is called <u>training data</u>.
- The learned rules should also be able to <u>generalize</u> to correctly recognize or predict new examples not in the training set.
- Require signal processing algorithm to process the signal (e.g. Fast Fourier Transform)

Training dataset: Audio signal	Output text
	Yes
	No
	Happy



Machine Learning models can learn by example



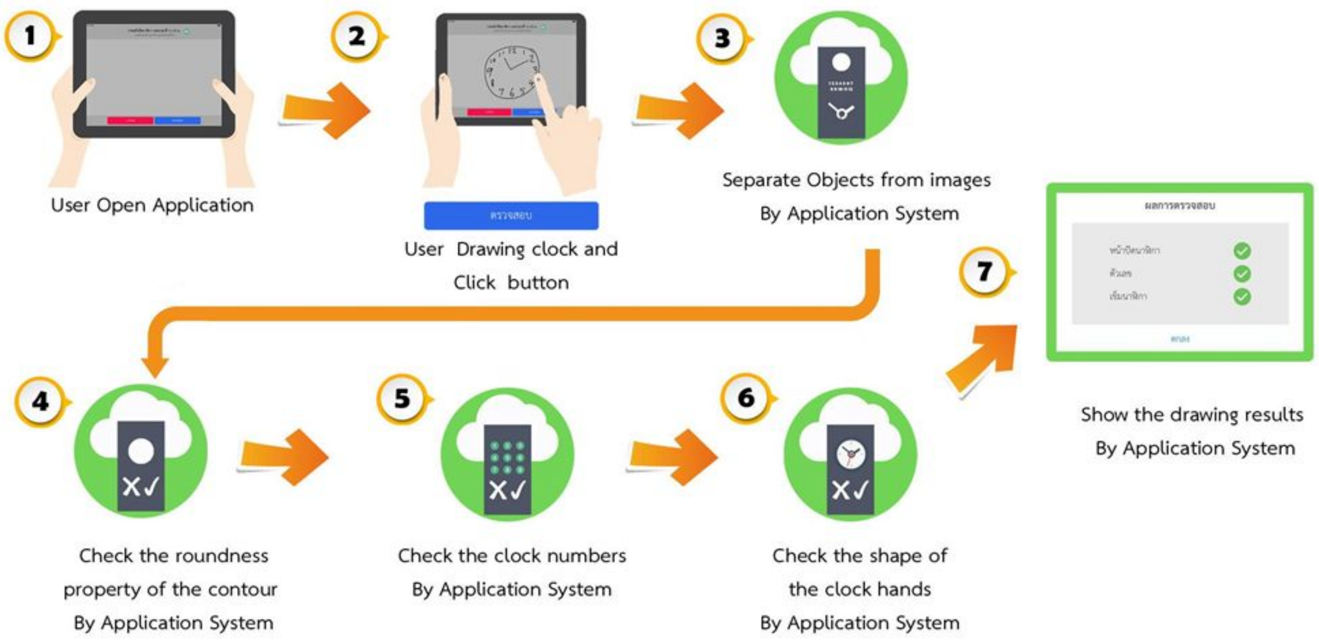
Machine Learning models learn from experience

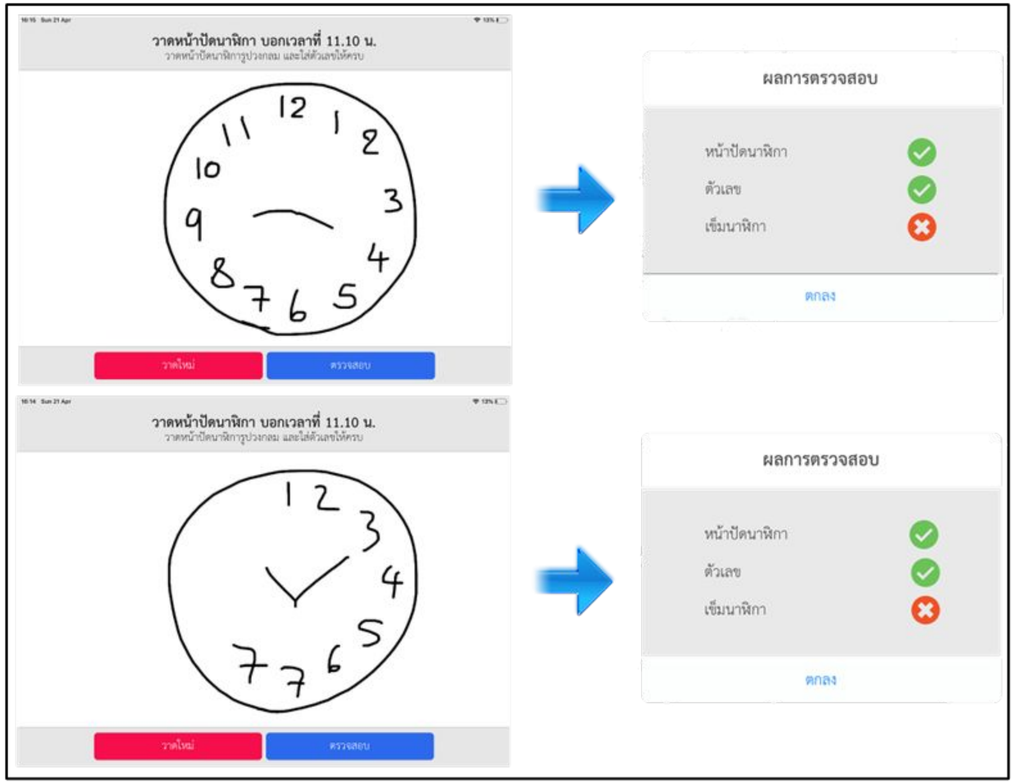
- Labelled examples (Email spam detection)

- User feedback (Clicks on a search page)

- Surrounding environment (self-driving cars)

Machine learning to detect Alzheimer's disease





Other machine learning applications

- Detect fraud transaction
- Handwritten recognition
- Insurance: which claim is fraud?
- Health: does this patient have cancer?
- Customer segmentation: How many groups of wine drinker?

Is a horse an acerous or non-acerous?

Acerous Vs. Non-Acerous

https://youtu.be/TeFF9wXiFfs Udacity's Intro to Machine Learning

Solution - Acerous Vs. Non-Acerous

https://youtu.be/M5pj2CrO-2w Udacity's Intro to Machine Learning

Good data vs good model: which one is better?

Step 1: Acquire Data

- Identify data sets
- Retrieve data
- Query data

Step 3: Analyze Data

- Select analytical techniques
- Build models

Good data vs good model

Good data: refers to high-quality information that is accurate, relevant, and reliable for the specific context in which it is used

- Accuracy
- Completeness
- Consistency
- Relevance
- Timeliness

Good model: a mathematical or computational representation that effectively captures the relationships within the data and can make accurate predictions or classifications.

- Generalization
- Simplicity
- Accuracy
- Robustness
- Explainability

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)
- Unsupervised machine learning: Find structure in unlabeled data
  - Find groups of similar instances in the data (clustering)
  - Finding unusual patterns (anomaly detection)
  - Find rules to capture associations between items. (association Analysis)
- Semi-supervised machine learning: Learn with partially labelled data and a lot of unlabelled data
  - Recognizes a person from a pool of photos using the combination of supervised and unsupervised machine learning algorithm
- Reinforcement machine learning: Learn strategies to maximize final reward

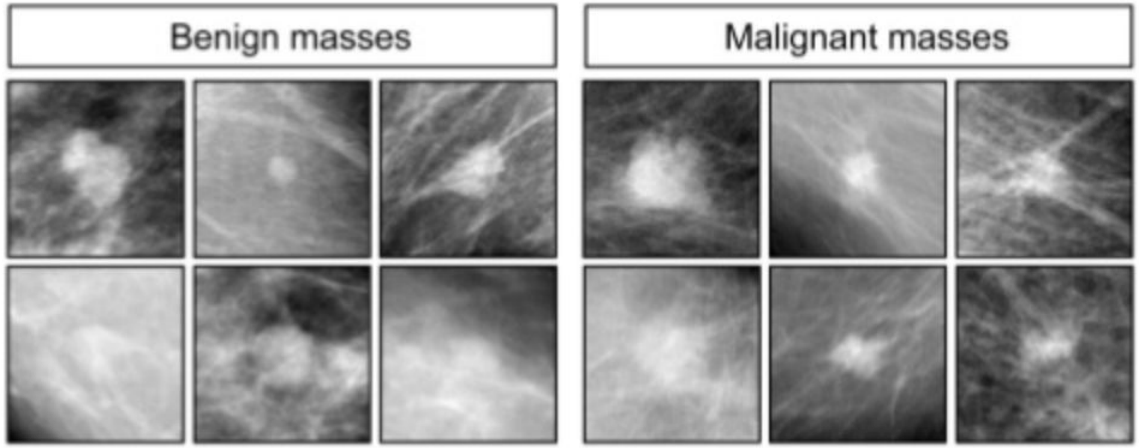
Supervised machine learning: Learn to predict target values from labelled data.

Labelled Data

ID	Clump	UnifSize	UnifShape	MargAdh	SingEpiSize	BareNuc	BlandChrom	NormNucl	Mit	Class
1000025	5	1	1	1	2	1	3	1	1	benign
1002945	5	4	4	5	7	10	3	2	1	benign
1015425	3	1	1	1	2	2	3	1	1	malignant
1016277	6	8	8	1	3	4	3	7	1	benign
1017023	4	1	1	3	2	1	3	1	1	benign
1017122	8	10	10	8	7	10		7	1	malignant
1018099	1	1	1	1	2	10	3	1	1	benign
1018561	2	1	2	H	2	1	3	1	1	benign
1033078	2	1	1	1	2	1	1	1	5	benign
1033078	4	2	1	1	2	1	2	1	1	benign

labels

Benign or malignant tumor: Classification or regression?

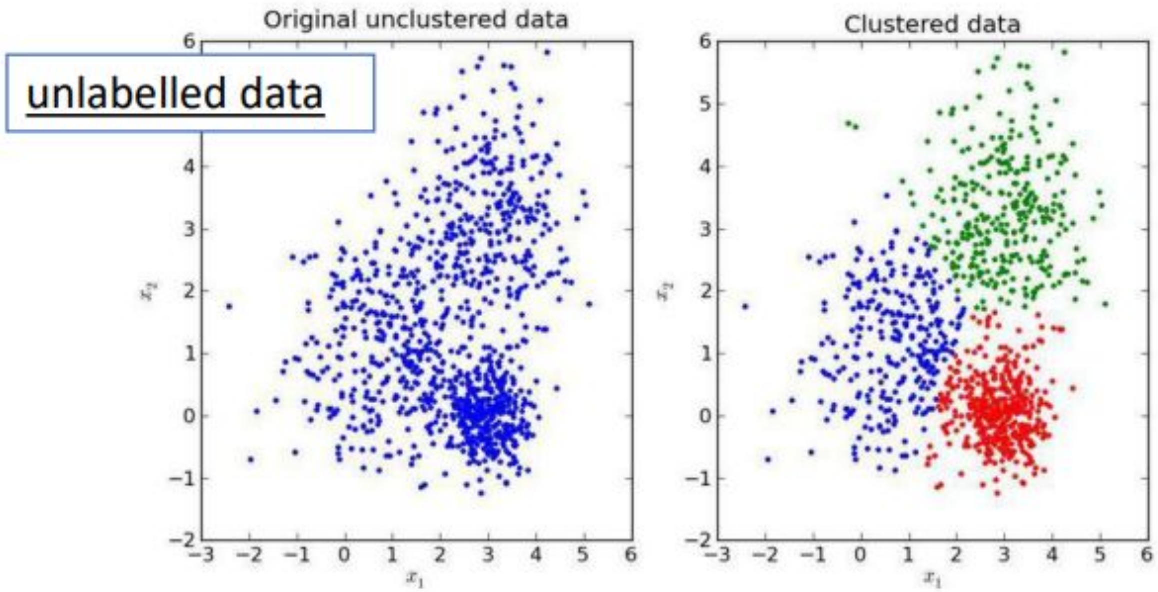


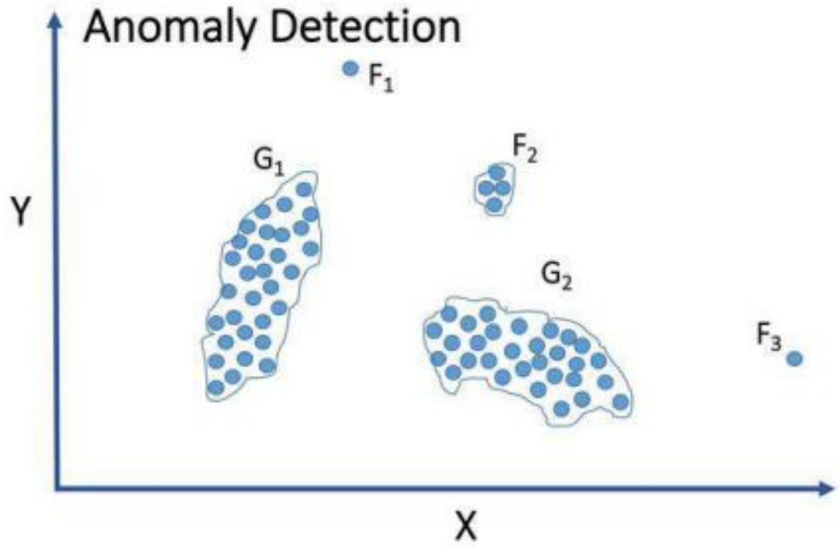
House price prediction: Classification or regression?

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2
5	0.02985	0.0	2.18	0.0	0.458	6.430	58.7	6.0622	3.0	222.0	18.7	394.12	5.21	28.7

Key types of Machine Learning problems

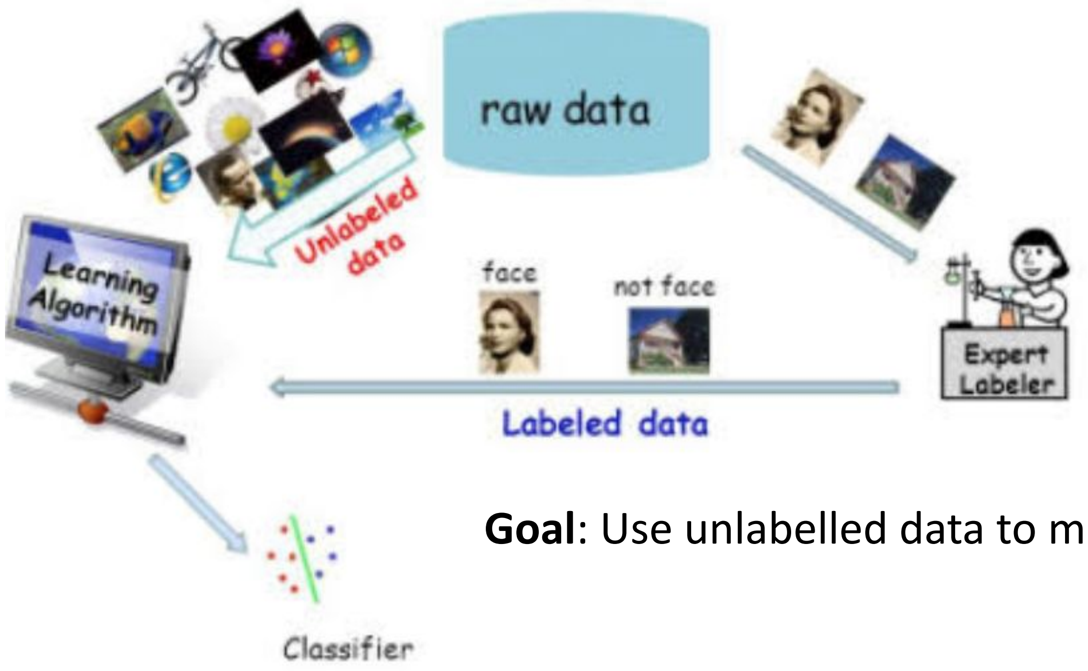
- Unsupervised machine learning: Find structure in unlabelled data
    - Find groups of similar instances in the data (clustering)
    - Finding unusual patterns (anomaly detection)
    - Find rules to capture associations between items. (association Analysis)





Key types of Machine Learning problems

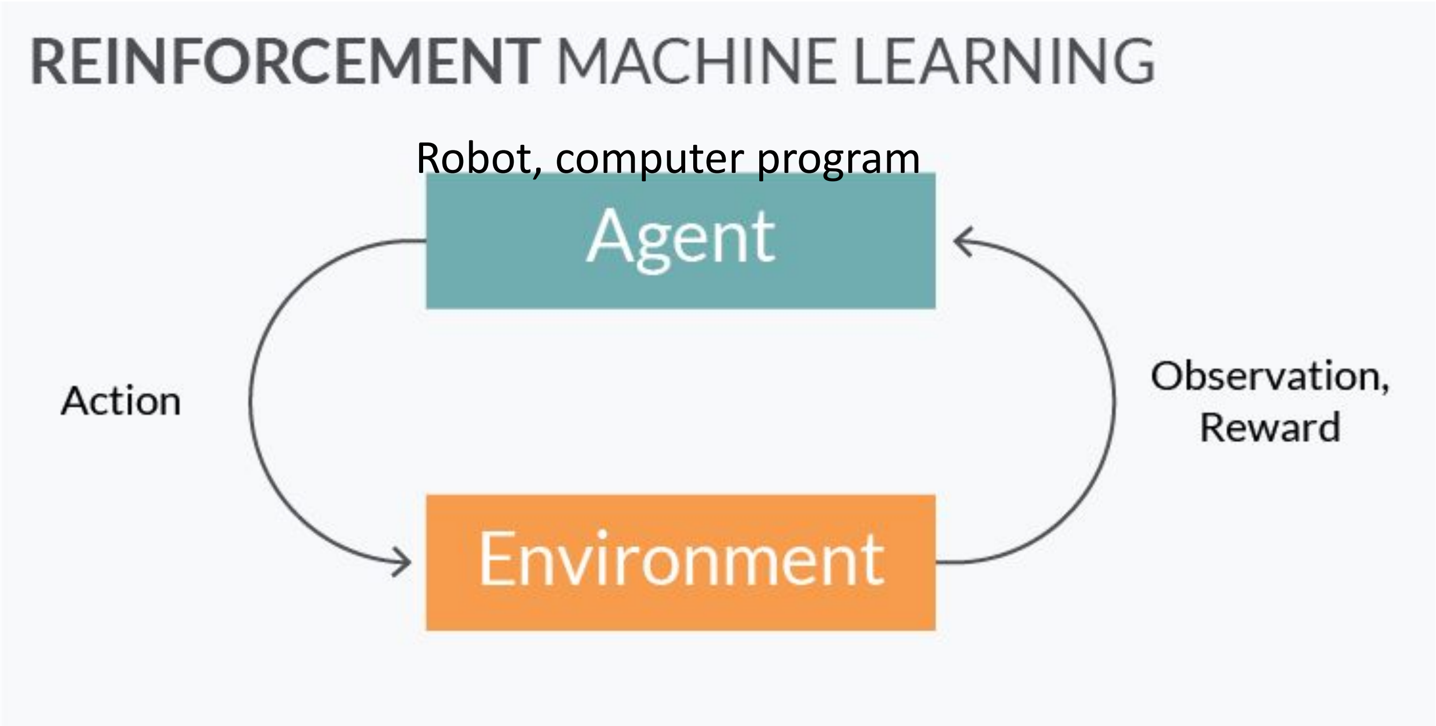
- Semi-supervised machine learning: Learn with partially labelled data and a lot of unlabelled data
  - Recognizes a person from a pool of photos using the combination of supervised and unsupervised machine learning algorithm

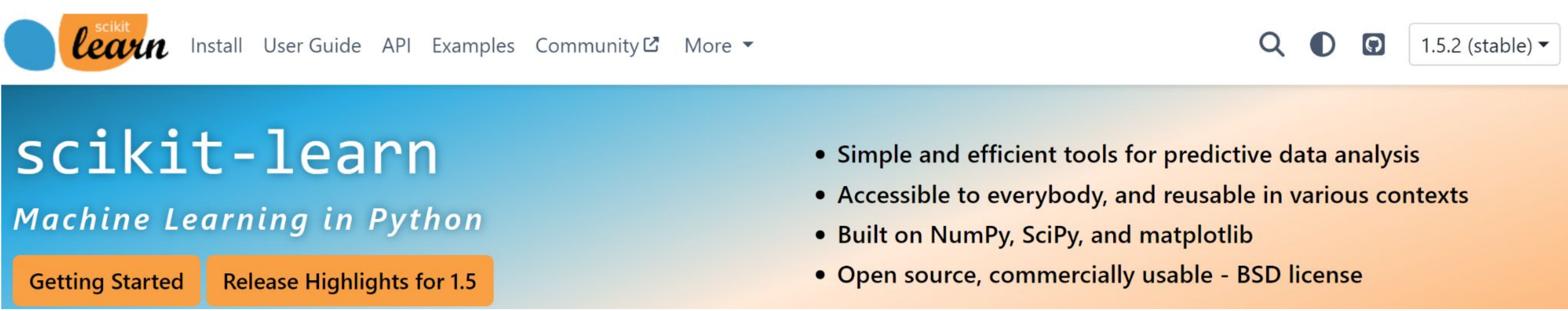


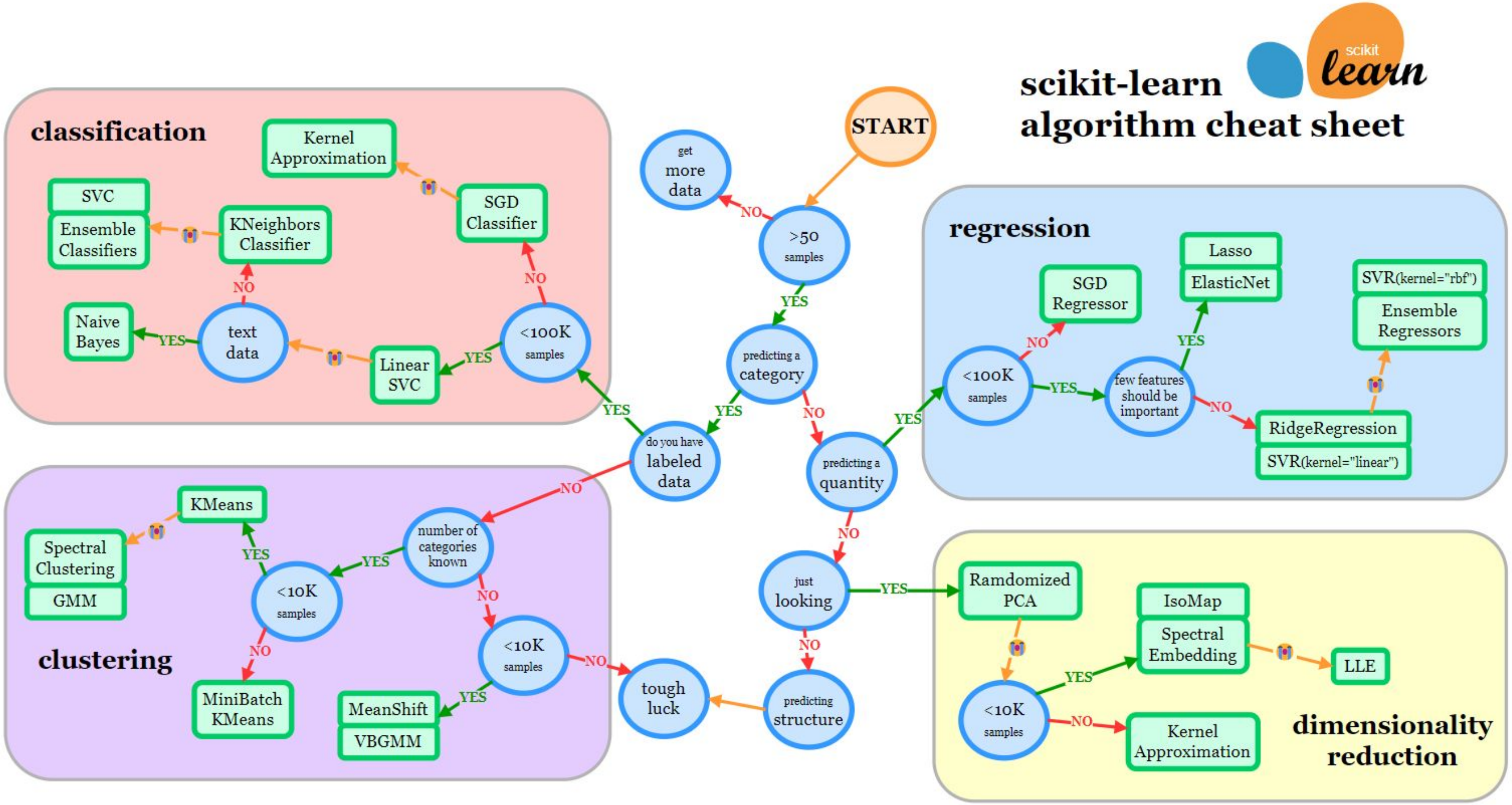
Goal: Use unlabelled data to make supervised learning better

Key types of Machine Learning problems

- Reinforcement machine learning: Learn strategies to maximize final reward within the environment.







- Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python.
- It provides a selection of efficient tools for machine learning and statistical modeling via a consistent interface in Python.
- This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

SVR: Support Vector Machine for Regression
SVC: Support vector machines for Classification

Supervised Machine Learning

Key Concepts

Supervised machine learning

- A prominent approach within the field of machine learning that utilizes labeled datasets to train algorithms for predicting outcomes or classifying data.
- This method is characterized by its reliance on input-output pairs, where the input data (features) is associated with known outputs (targets).
- The primary goal is for the model to learn the relationship between these features and targets so it can make accurate predictions on new, unseen data (generalization)
- Labeled Data: a data that contains both the Features (X variables) and the Target (y variable).
- Training or Fitting: the algorithm iteratively learns to predict the target variable given the features and modifies for the proper response in order to "learn" from the training dataset

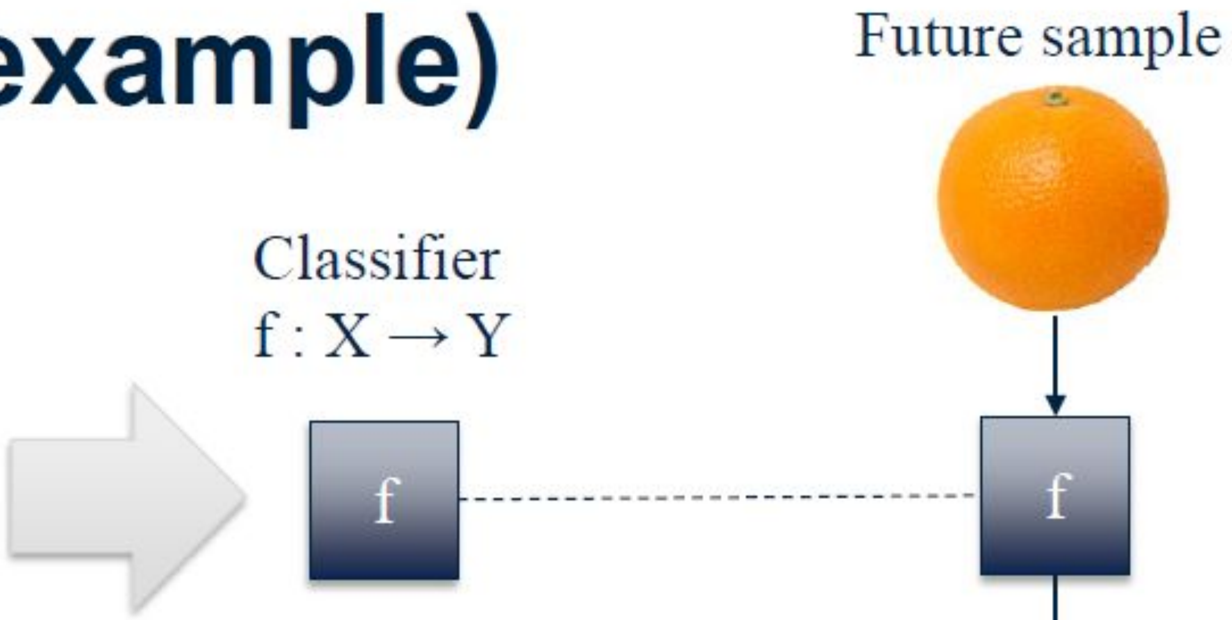
Supervised machine learning

- **Types of problems**: classification and regression
- **Learning process**:
  - Training: The algorithm is trained using a labeled dataset. During this phase, it learns to map inputs to outputs by adjusting its internal parameters based on the errors it makes in predictions.
  - Validation: After training, the model's performance is evaluated using a separate validation dataset to ensure it generalizes well to new data.
  - Testing: Finally, the model is tested on an unseen dataset to assess generalization
- **Common algorithm**: KNN, Linear regression, logistic regression, Decision tree, Random forest, SVM etc…

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

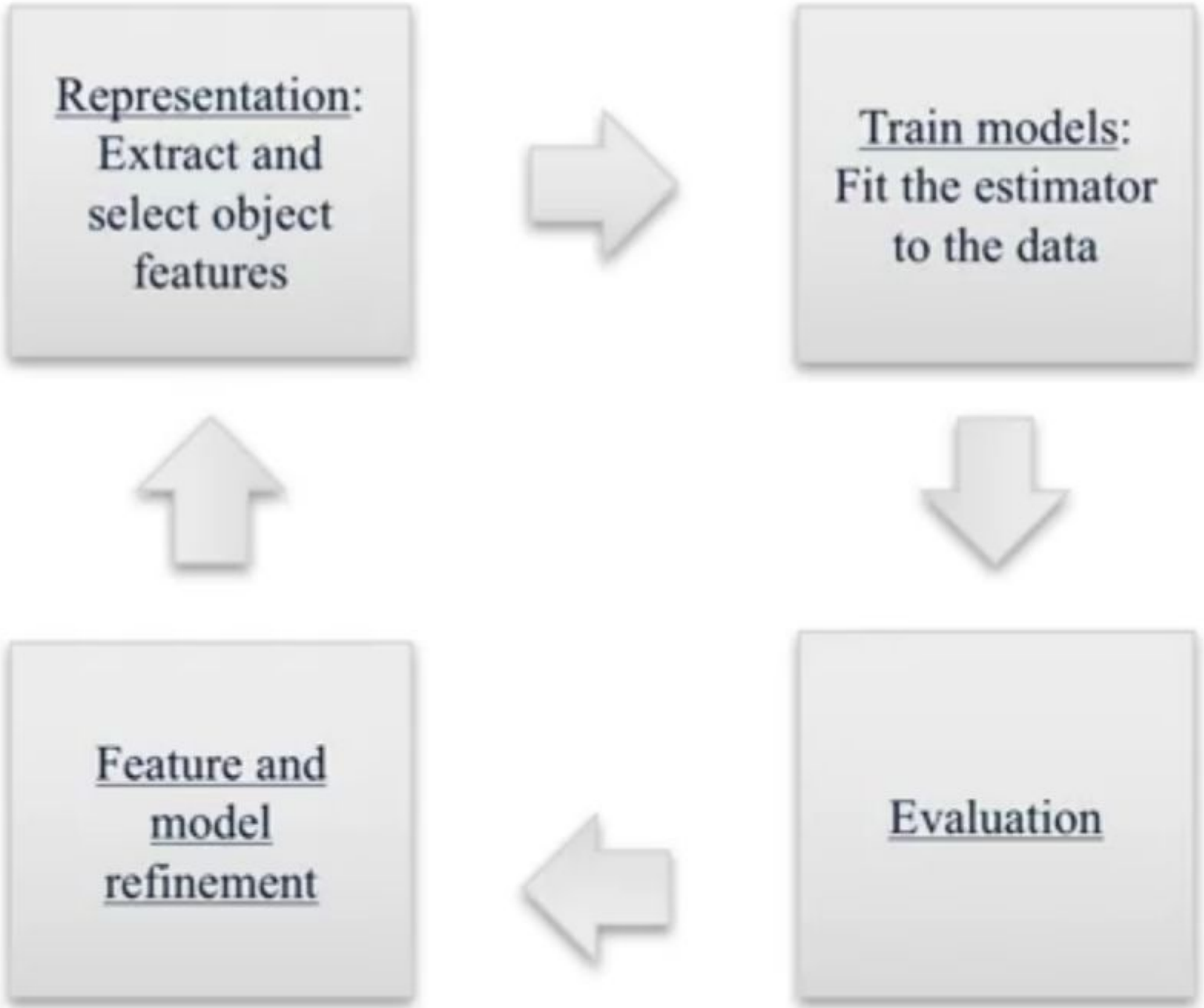


Label: Orange

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

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

Represent / Train / Evaluate / Refine Cycle



Feature Representations

Challenge of getting a good set of features: feature extraction and feature engineering

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

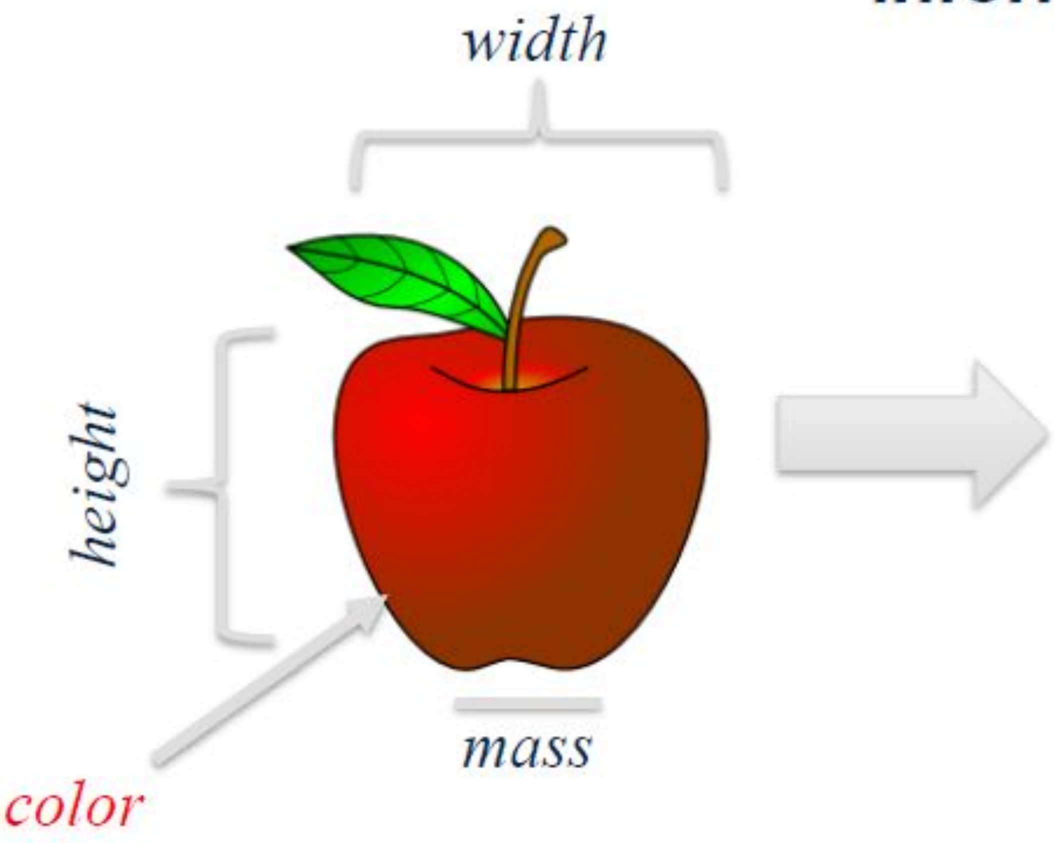
A matrix of color values (pixels)

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

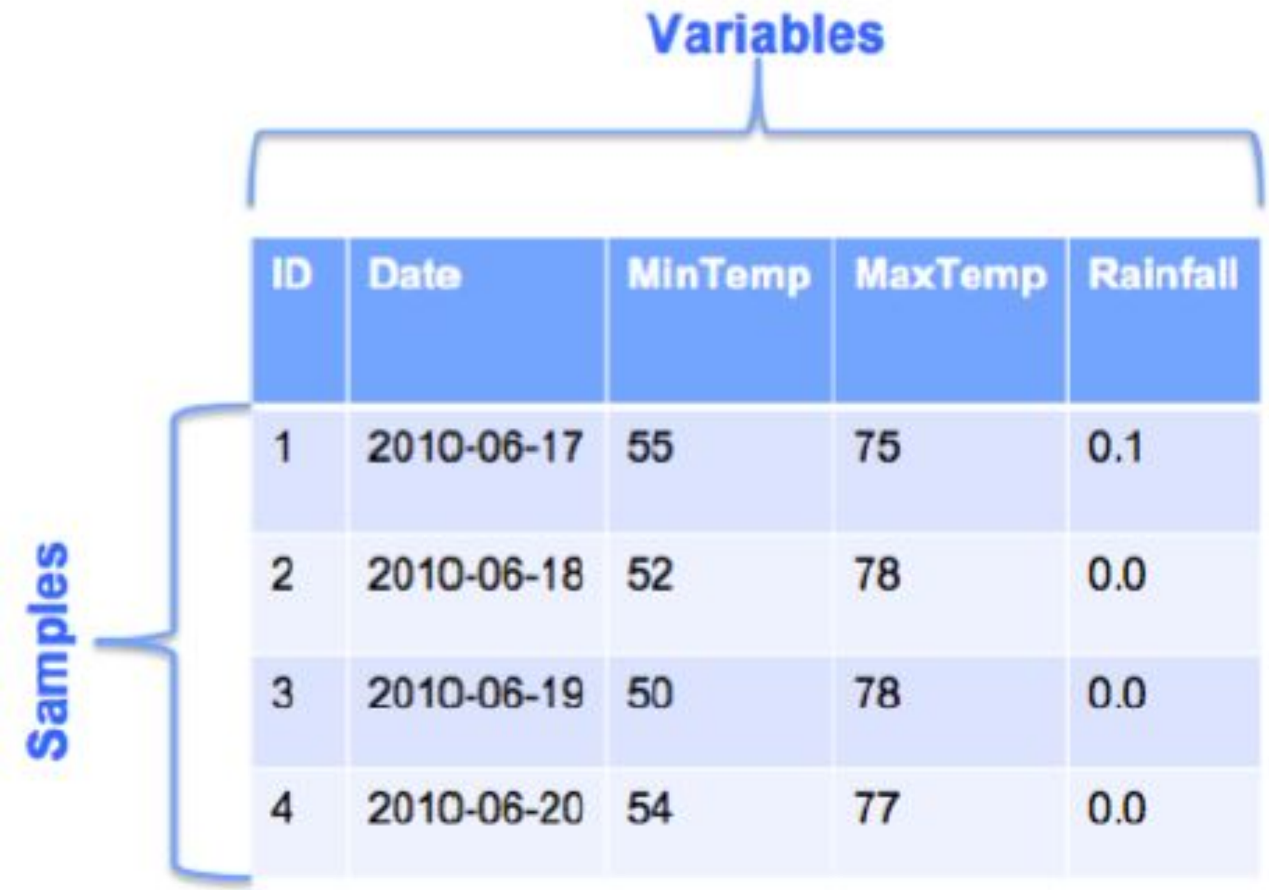
	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
18	1	apple	cripps_pink	162	7.5	7.1	0.83

2. Learning model

Classifier

Predicted class (apple)

Terms to Describe Data



ID	Date	MinTemp	MaxTemp	Rainfall
1	2010-06-17	55	75	0.1
2	2010-06-18	52	78	0.0
3	2010-06-19	50	78	0.0
4	2010-06-20	54	77	0.0

- Other Names for ‘Sample’
  - Sample
  - Row
  - Record
  - Example
  - Instance
  - Observation

Terms to Describe Data

Variables

Samples

ID	Date	MinTemp	MaxTemp	Rainfall
1	2010-06-17	55	75	0.1
2	2010-06-18	52	78	0.0
3	2010-06-19	50	78	0.0
4	2010-06-20	54	77	0.0

- Other Names for ‘Variables’
  - Column
  - Attribute
  - Field
  - Feature
  - Dimension

Supervised Machine Learning Notation

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

y_1

$$X_1 = \begin{bmatrix} 162 \\ 7.5 \\ 7.1 \\ 0.83 \end{bmatrix}$$

Datasets: $(\overline{\boldsymbol{x}}_i, y_i)$

Input: $\overline{\boldsymbol{x}}_i \in \mathbb{R}^p$ p is dimension

Output: $y_i \in \mathbb{R}$

Estimator: $f(x_i) \approx y_i$

What's the assumption of $(\overline{\boldsymbol{x}}_i, y_i)$?

Representing data

one sample

$$X = \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix} \qquad y = \begin{pmatrix} 1.6 \\ 2.7 \\ 4.4 \\ 0.5 \\ 0.2 \\ 5.6 \\ 6.7 \end{pmatrix}$$

one feature

outputs / labels

Train/Test Set

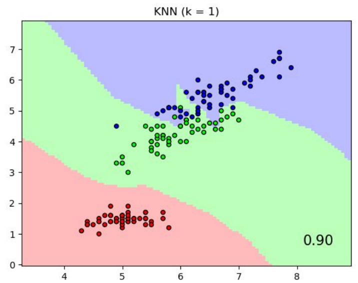
How would we know if its predictions were likely to be accurate?

training set

$$X = \begin{pmatrix} 1.1 & 2.2 \\ 6.7 & 0.5 \\ 2.4 & 9.3 \\ 1.5 & 0.0 \\ 0.5 & 3.5 \\ 5.1 & 9.7 \\ 3.7 & 7.8 \end{pmatrix} \qquad y = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

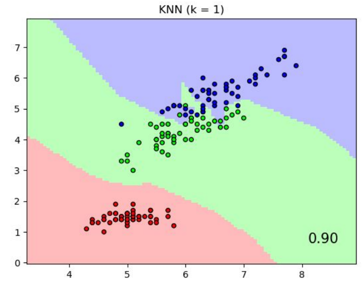
test set

Nearest Neighbors



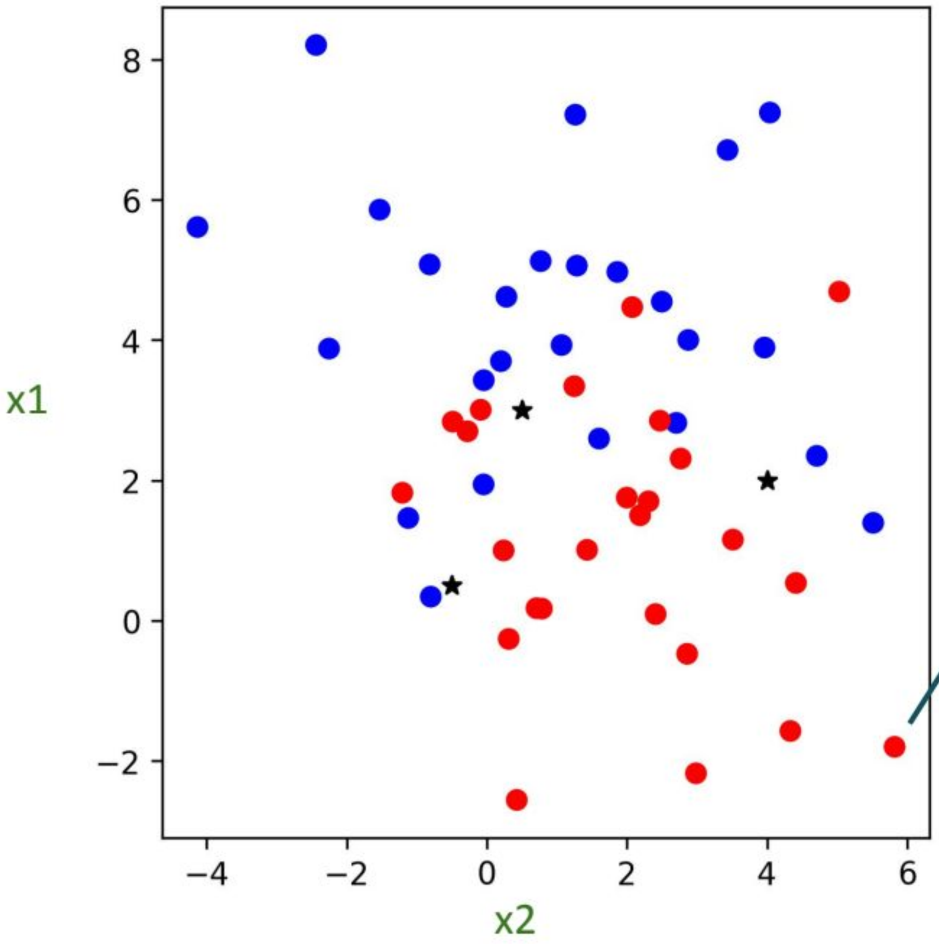
- sklearn.neighbors provides functionality for supervised neighbors-based learning methods
  - Supervised: classification and regression
- Principles: find a predefined number of training samples closest in distance to the new point, and predict the label from these
- The number of samples can be a user-defined constant (k-nearest neighbor learning), or vary based on the local density of points (radius-based neighbor learning)
- The distance can be any metric measure: standard Euclidean distance (most common), Dynamic time warping.
- Neighbors-based methods (non-generalizing machine learning methods), since they simply "remember" all of its training data

Nearest Neighbors



- Despite its simplicity, nearest neighbors has been successful in a large number of classification and regression problems (handwritten digits and satellite image scenes)
- **No assumption on data distribution**: Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular.
- The classes in sklearn.neighbors can handle either NumPy arrays or scipy.sparse matrices as input.
- Instance-based learning (lazy learning)

K Nearest Neighbor (K=1)

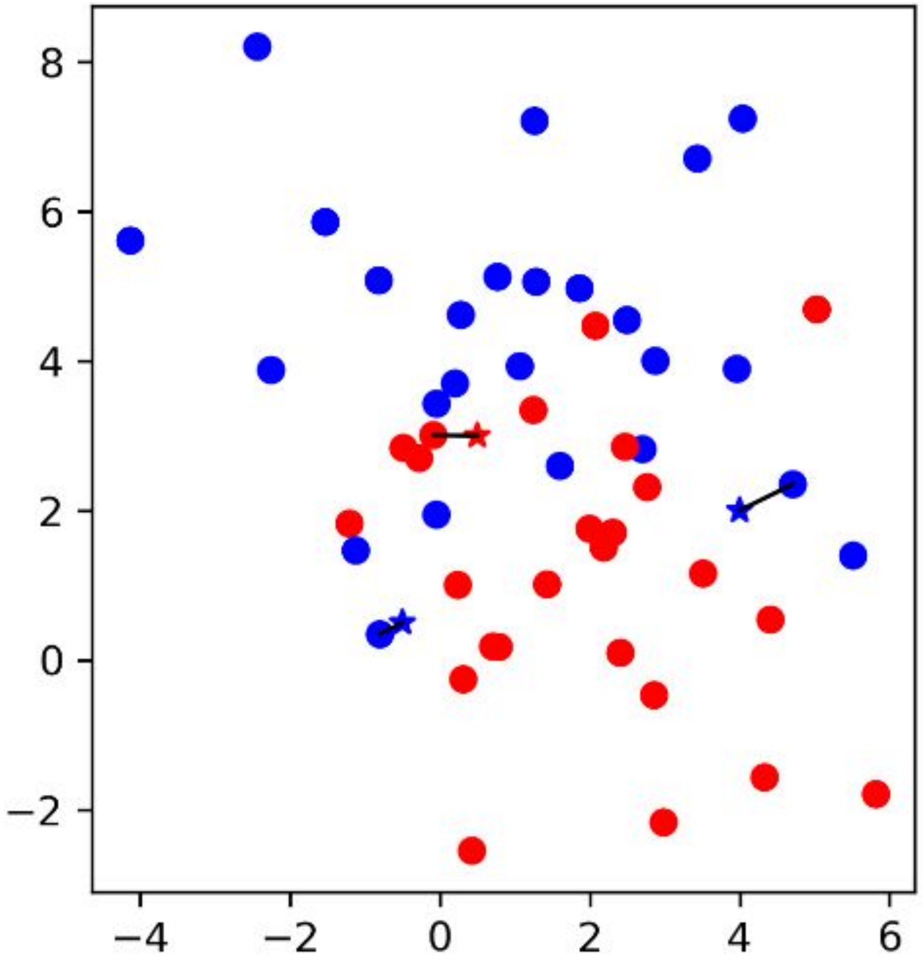


Features		label
x1	x2	y
-2	6	red

34

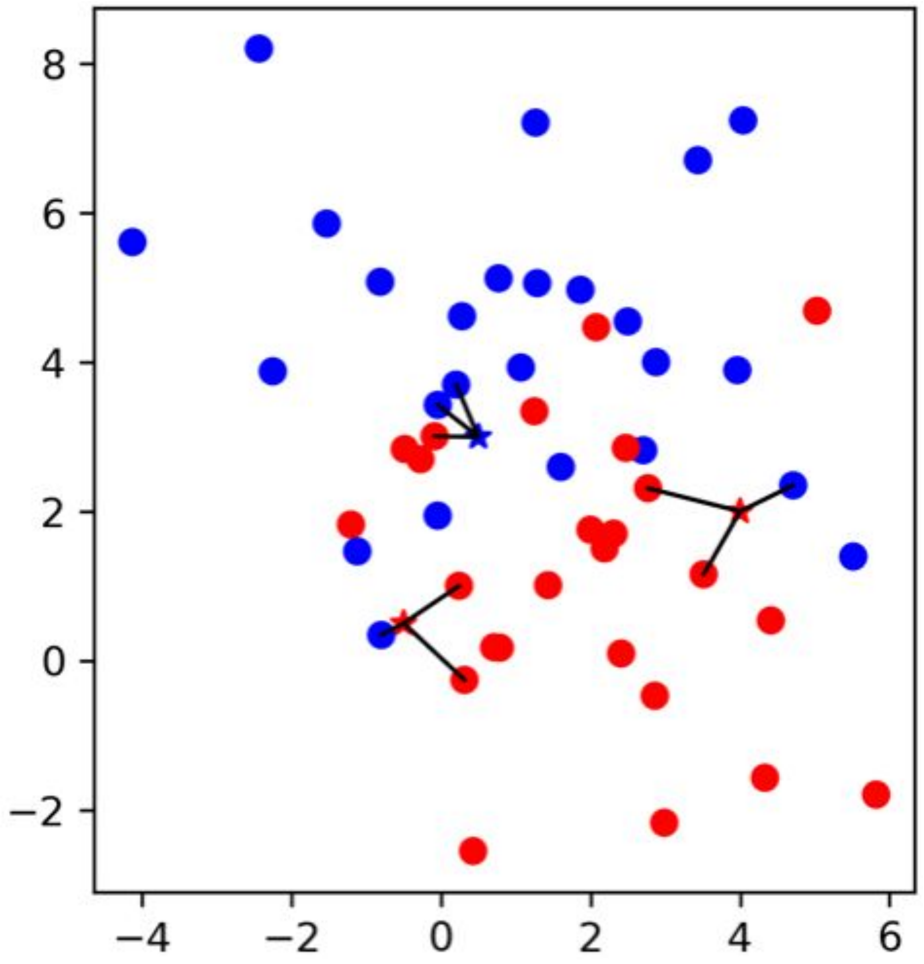
More neighbors (K=1)

Red or Blue?



More neighbors (K=3)

Red or Blue?



KNN with scikit-learn (K=1)

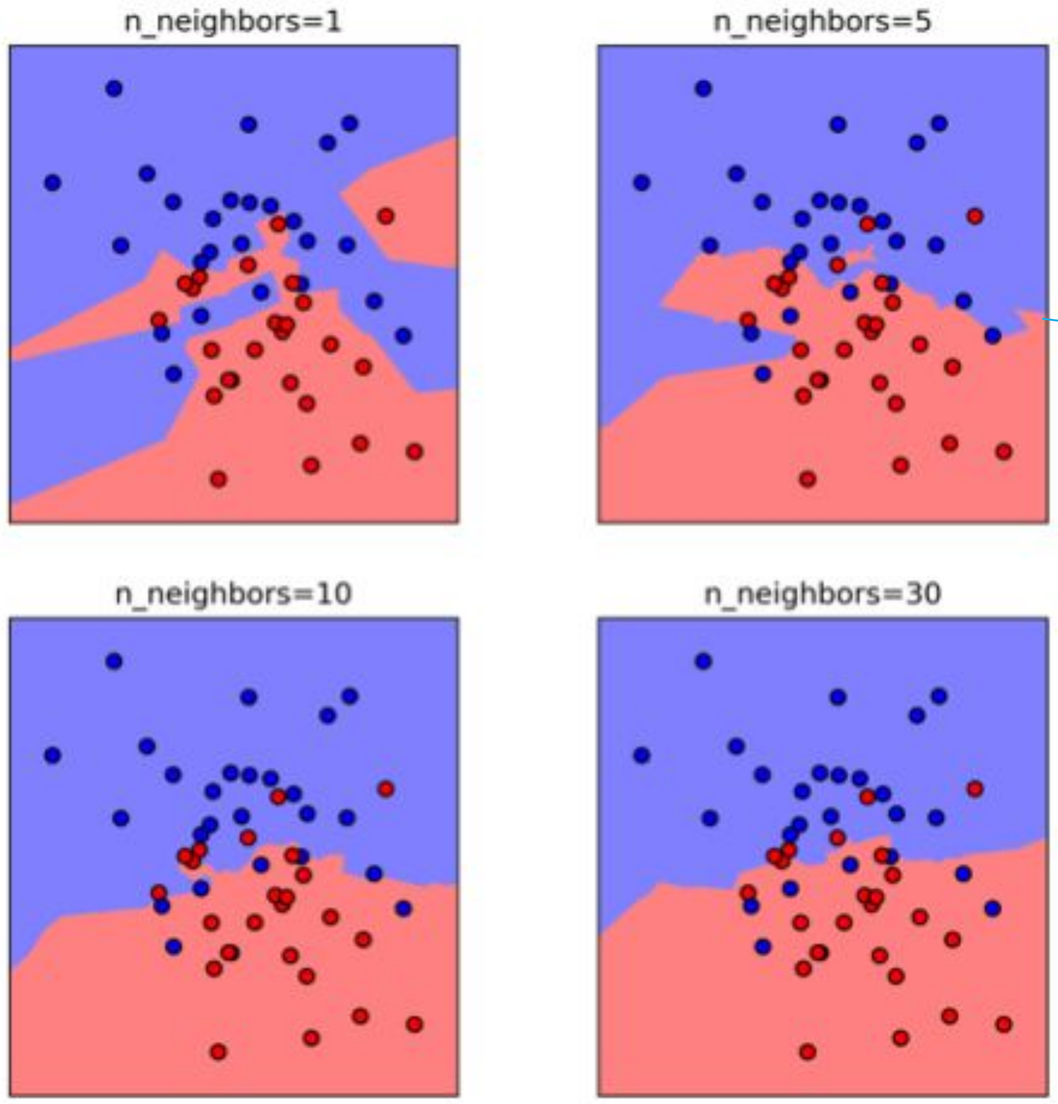
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y)

from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X_train, y_train)
print("accuracy: {:.2f}".format(knn.score(X_test, y_test)))
y_pred = knn.predict(X_test)
```

accuracy: 0.77

Hyperparameter
(n_neighbors (which is k))

Influence of n_neighbors

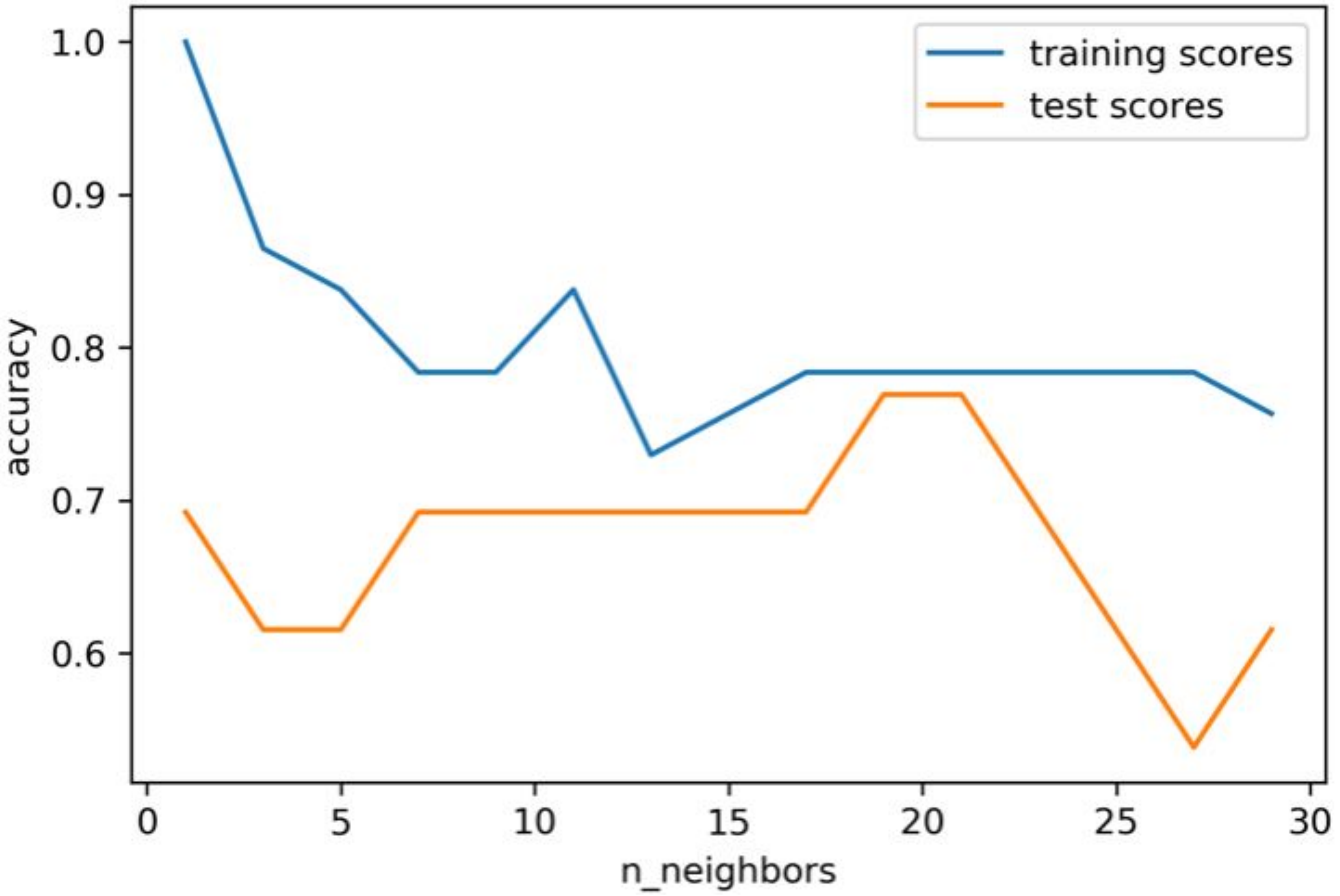


- K=1 prone to noise, outliers, mislabeled of classes, and considerable variations in **decision boundaries**.

- K=30 is more robust.

- KNN is good to be a baseline.
- Since it is an instance-based model, when the training data has many instances, or each instance has lots of features, this can really slow down the performance.

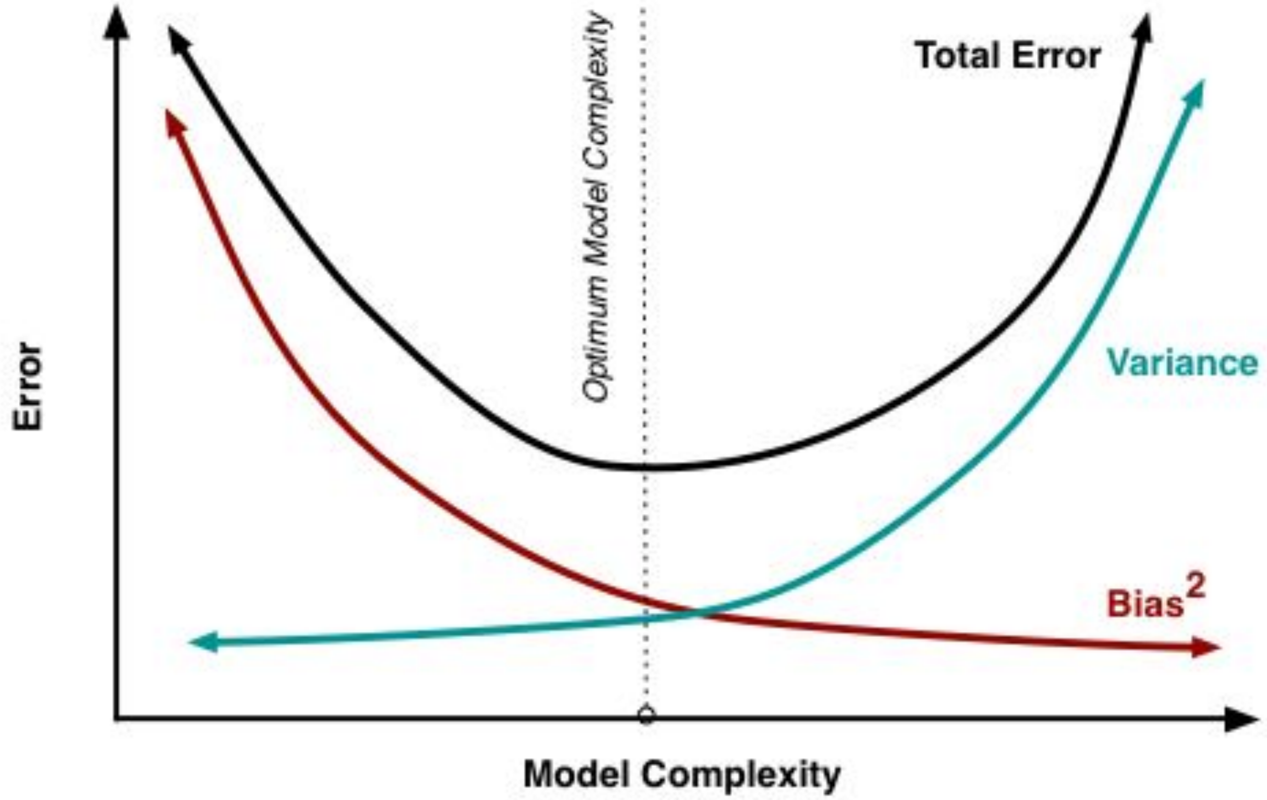
Model complexity



Small n_neighbors

- More complexed
- Fit the training more
- High variance, low bias

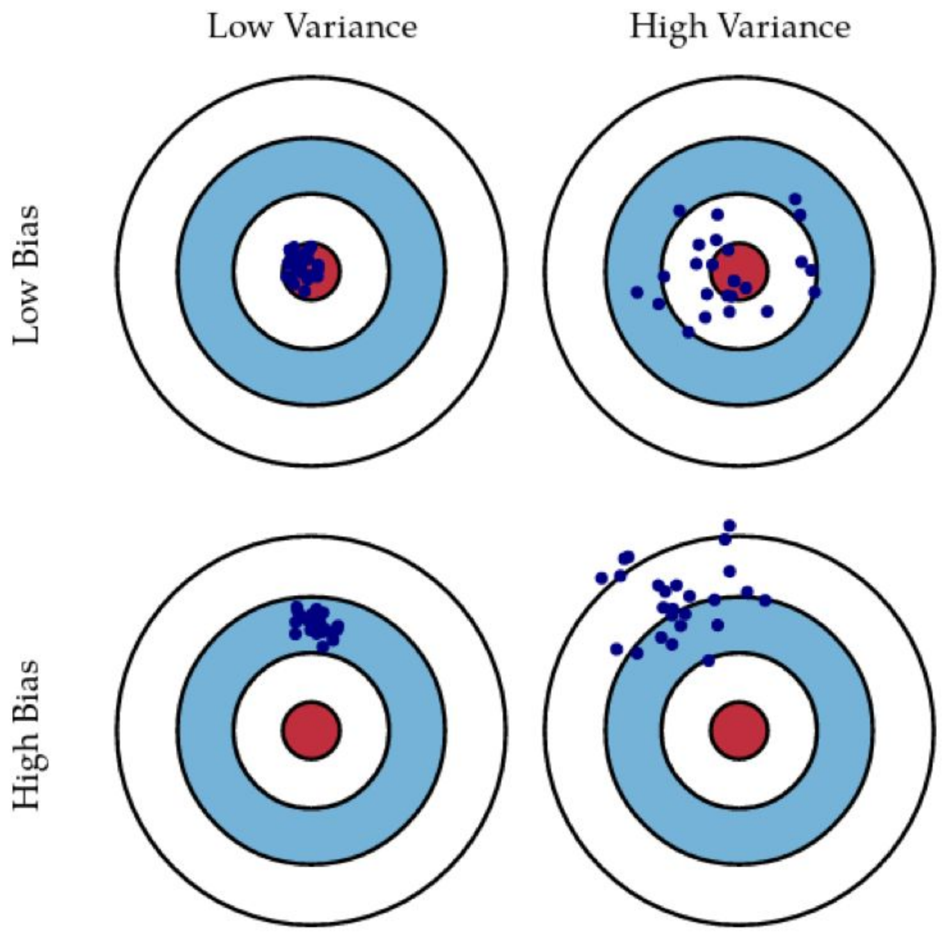
Bias-Variance Tradeoff



http://scott.fortmann-roe.com/docs/BiasVariance.html

Bias and Variance

- Center of the target is a model that perfectly predicts the correct values.
- Each hit represents an individual realization of our model (with the test set), given the chance variability in the training data we gather.

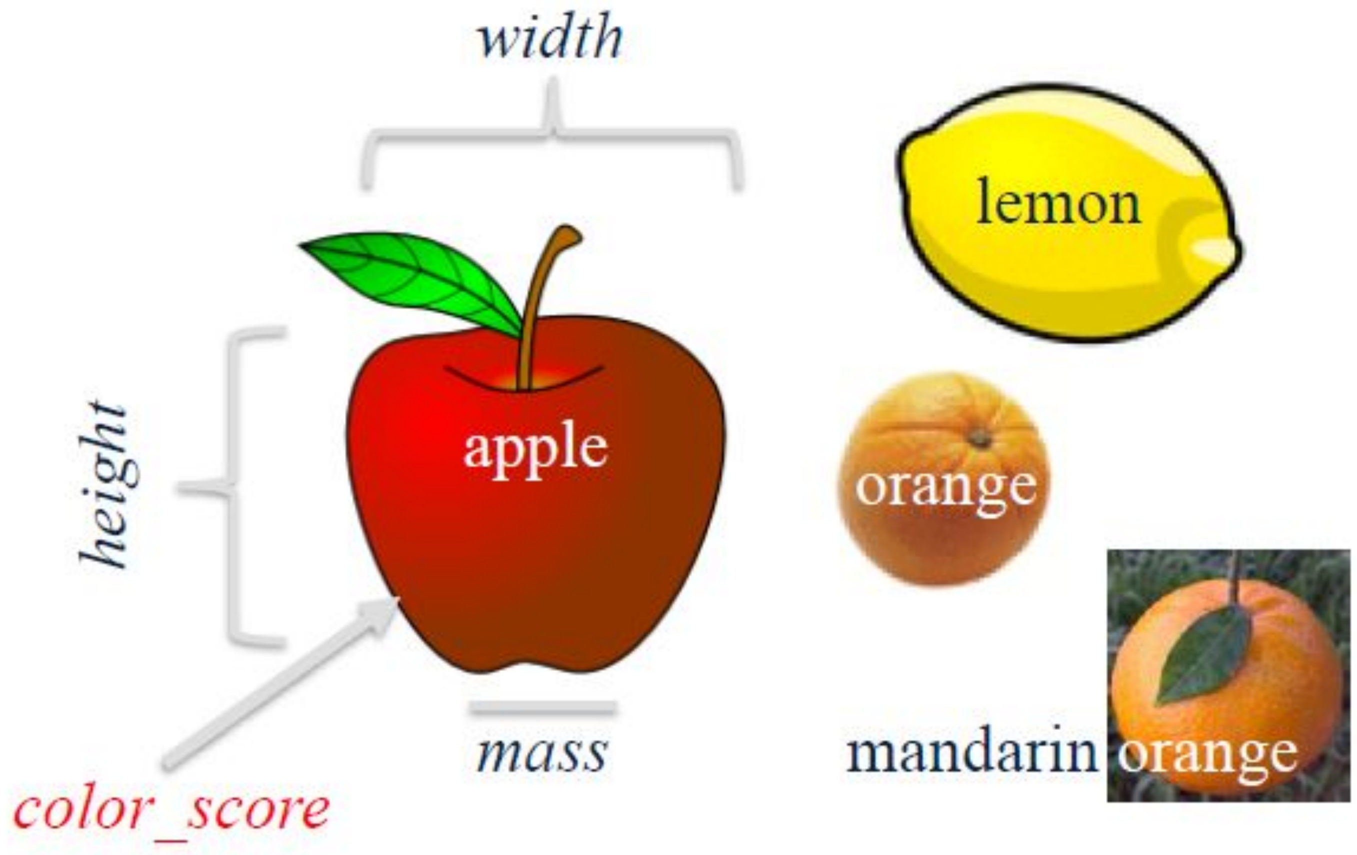


http://scott.fortmann-roe.com/docs/BiasVariance.html

Hands-On

Fruit Classifier

The Fruit Dataset: Identify a type of fruit based on height, weight, mass and color



	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)

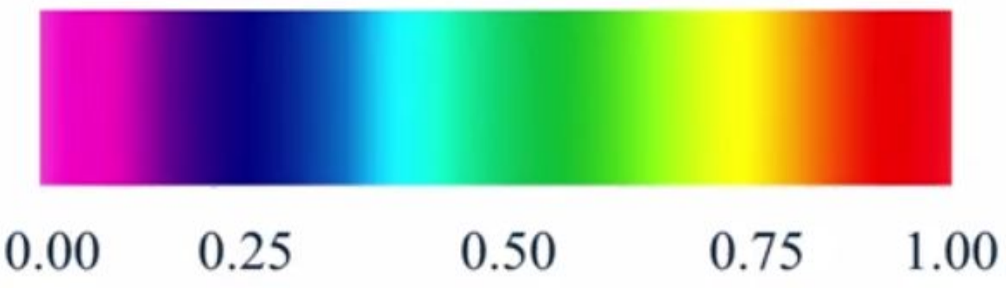
	fruit_label	uit_name	fruit_subtype	mass	width	height	color_score
0	1	pple	granny_smith	192	8.4	7.3	0.55
1	1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
2	1	pple	granny_smith	176	7.4	7.2	0.60
3	2	andarin	mandarin	86	6.2	4.7	0.80
4	2	andarin	mandarin	84	6.0	4.6	0.79
5	2	andarin	mandarin	80	5.8	4.3	0.77
6	2	andarin	mandarin	80	5.9	4.3	0.81
7	2	andarin	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
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

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

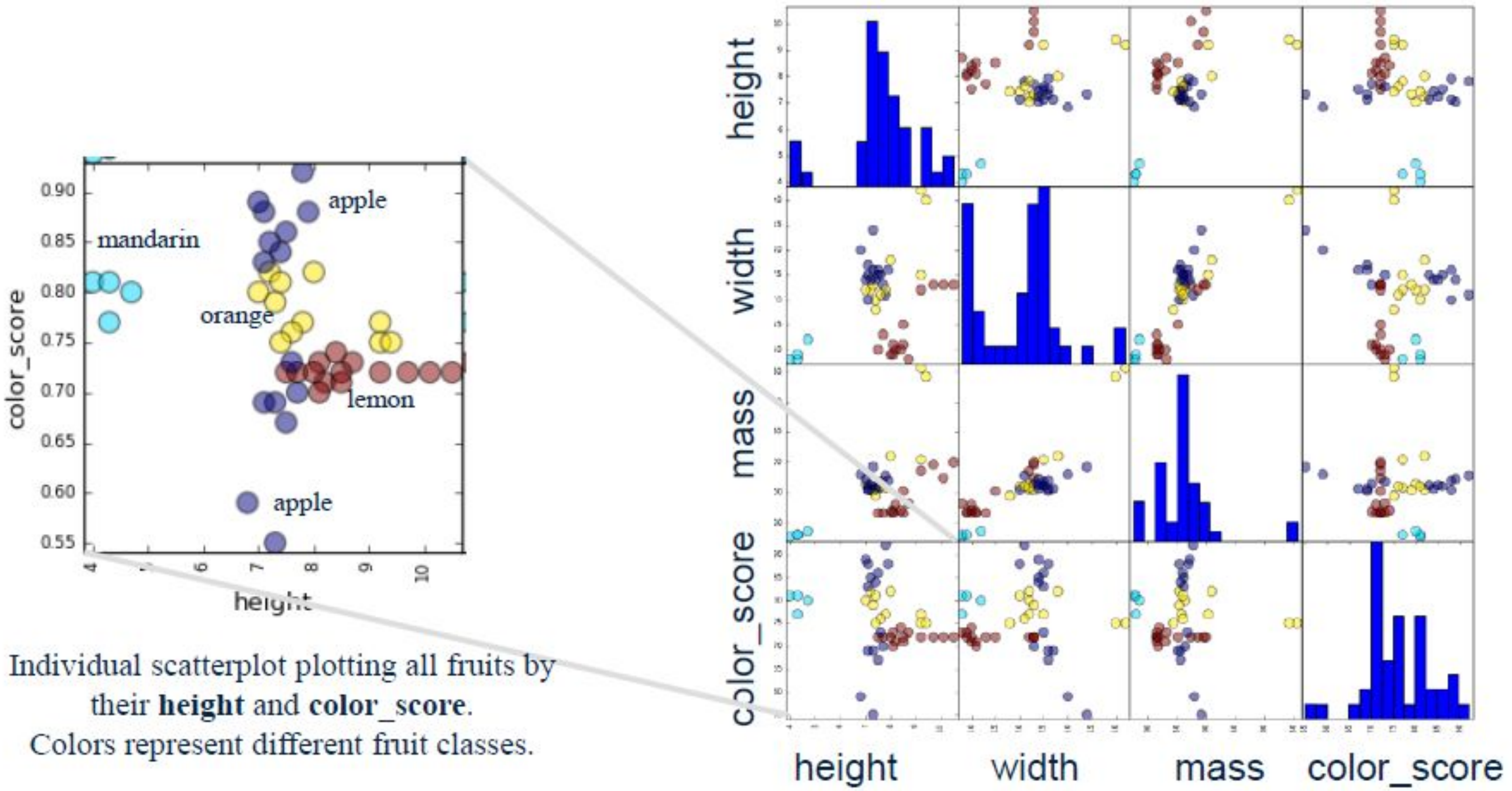
The scale for the (simplistic) color_score feature used in the fruit dataset



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

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.

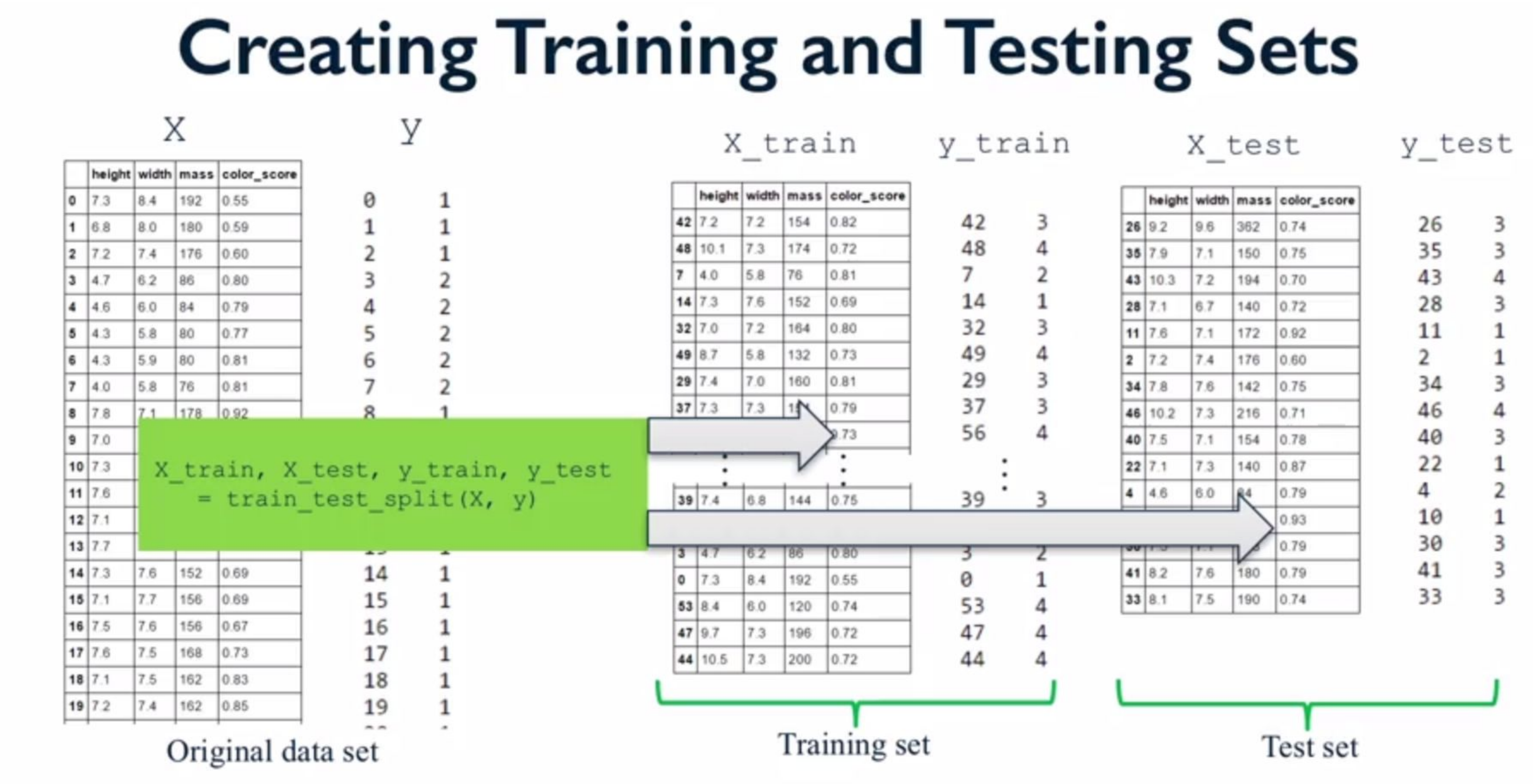
Some reasons why looking at the data initially is important (Why EDA)

- Inspecting feature values **may help identify what cleaning or pre-processing** 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

Creating Training and Testing Sets



Building Your First Model: k-Nearest Neighbors

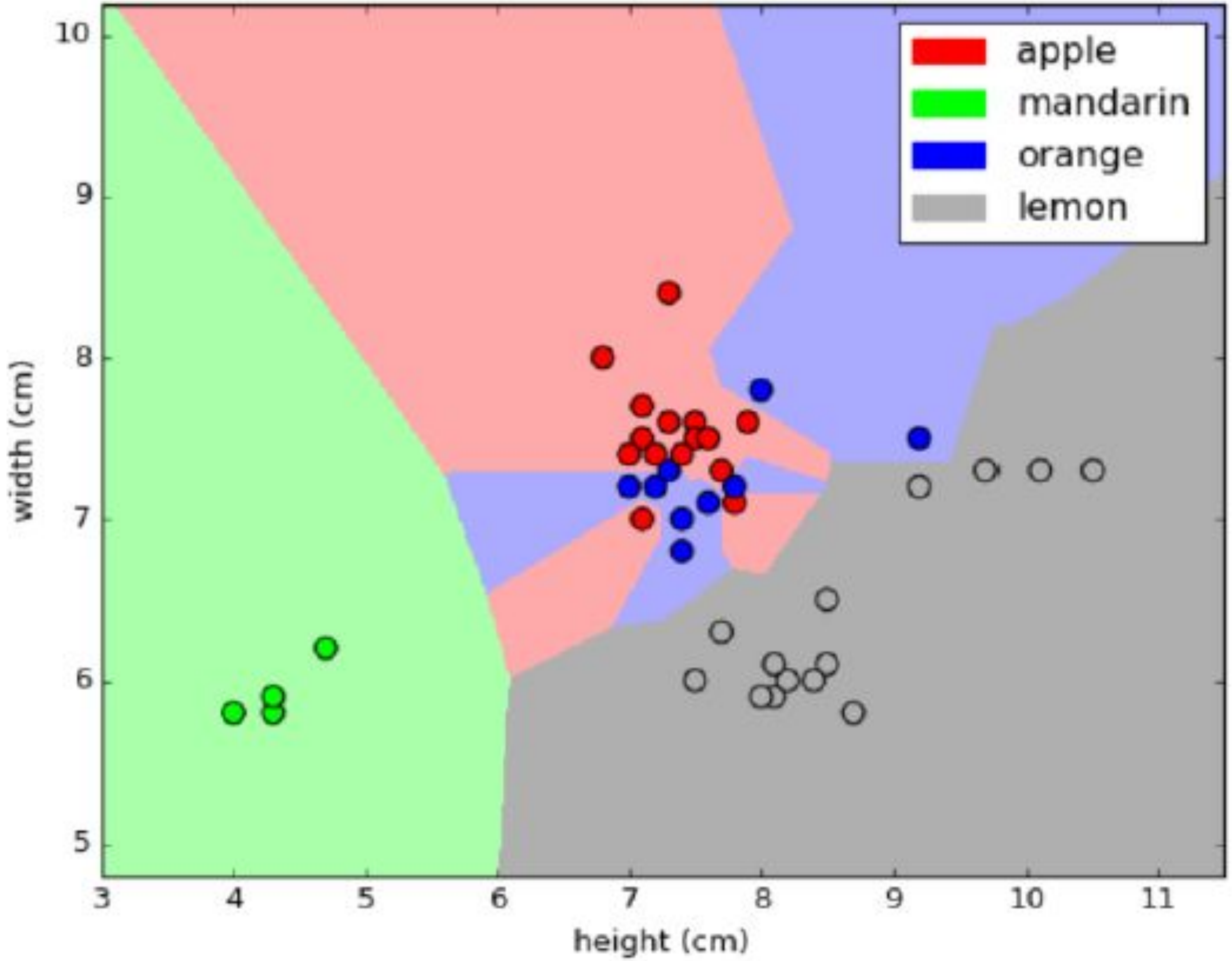
Given a training set X_train with labels y_train, and

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 classification



Fruit dataset
Decision boundaries
with k = 1

KNeighborsClassifier

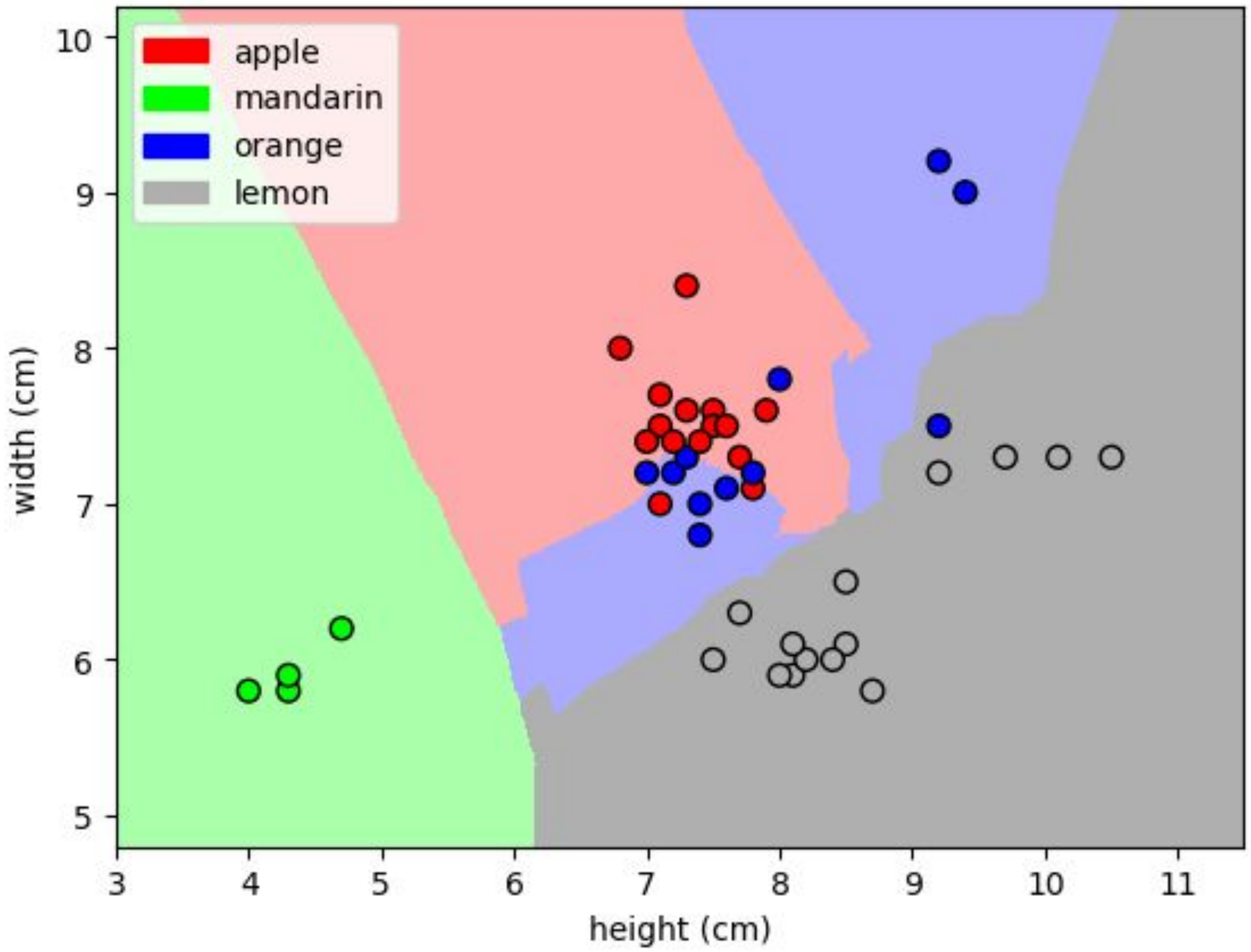
```
class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, *,
weights='uniform', algorithm='auto', leaf_size=30, p=2,
metric='minkowski', metric_params=None, n_jobs=None) #
```

Classifier implementing the k-nearest neighbors vote.

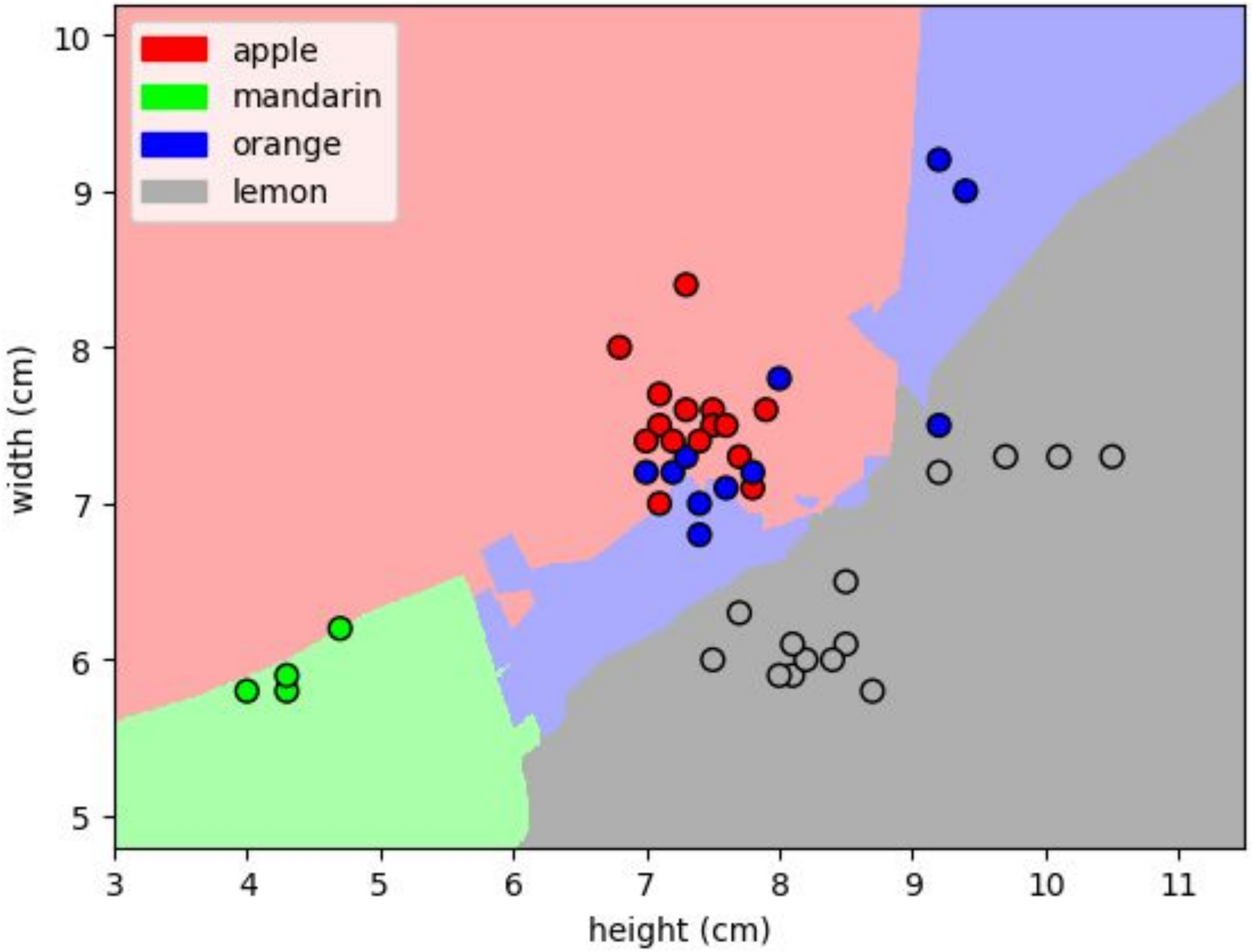
Parameters:

- n_neighborsint, default=5 Number of neighbors to use by default for kneighbors queries.
- p, default=2: Power parameter for the Minkowski metric. When p = 2, this is euclidean_distance
- weights{‘uniform’, ‘distance’}, callable or None, default=’uniform’ Weight function used in prediction. Possible values: ‘uniform’. All points in each neighborhood are weighted equally.
- algorithm: ‘auto’, ‘ball_tree’, ‘brute’

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

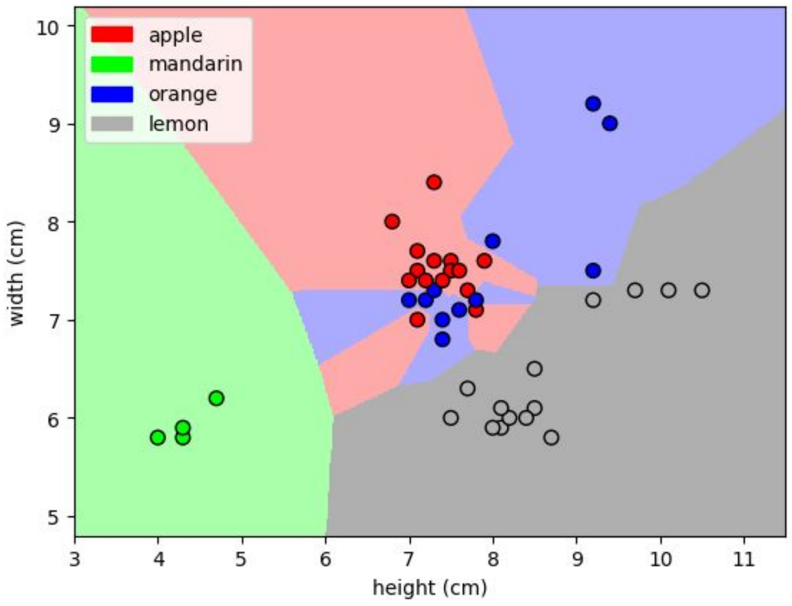


K-nearest neighbors (k=10) for fruit dataset



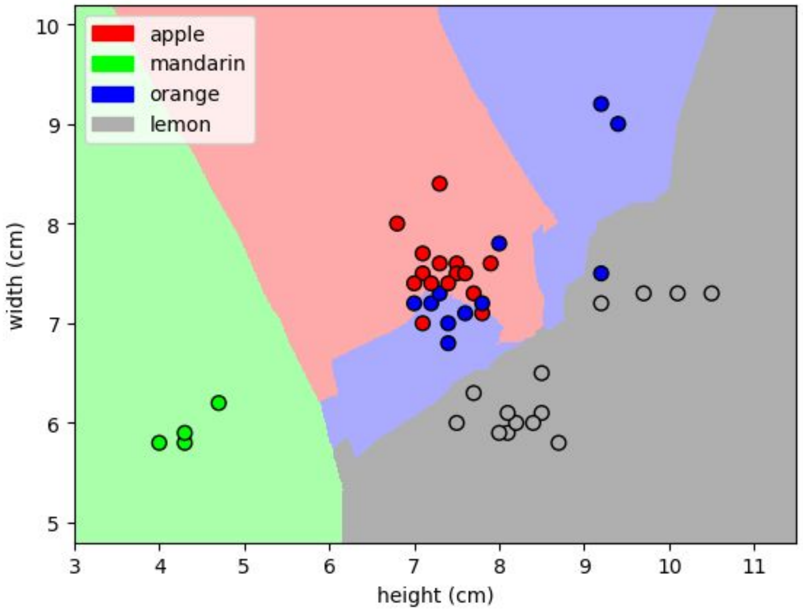
Overfitting in KNN

Variance vs Bias

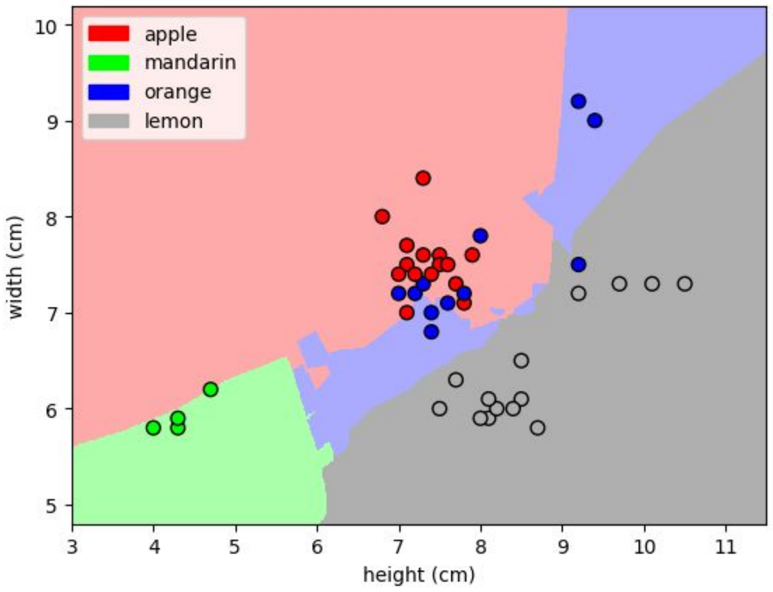


k=1

prone to noise, outliers, mislabeled of classes. and considerable variations in decision boundaries.



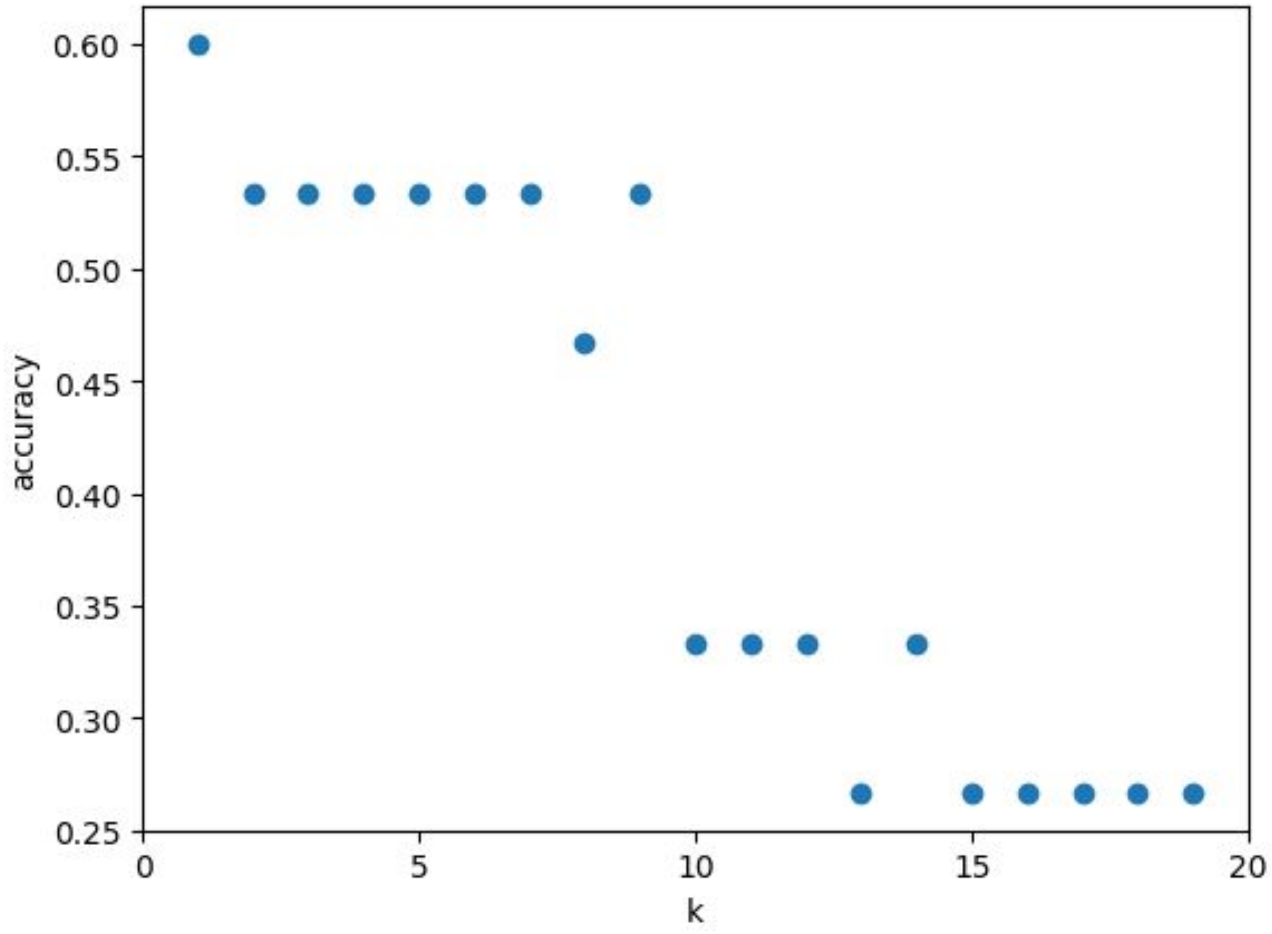
k=5



k=10

Smoother boundaries, more robust

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



References

- https://scikit-learn.org/stable/index.html
- Andreas C. Müller, **COMS W4995 Applied Machine Learning**, Columbia University, Spring 2020.
- Andreas C. Müller and Sarah Guido. ***Introduction to Machine Learning with Python: A Guide for Data Scientists***. O'Reilly Media; 1 edition.
- https://github.com/Starignus/AppliedML_Python_Coursera/tree/master
- Udacity's Intro to Machine Learning