INTRODUCTION TO MACHINE LEARNING

Dr. Hilman F. Pardede

Research Center for Informatics
Indonesian Institute of Sciences

The slides are compiled from the following resources:

- https://github.com/joaquinvanschoren/ML-course
- https://www.cse.iitk.ac.in/users/piyush/courses/ml_autumn16/ML.html
- http://sli.ics.uci.edu/Classes/2015W-273a

GOALS

- Understand how various machine learning algorithms work
- Implement them (and, hopefully, their variants/improvements) on your own
- Look at a real-world problem and identify if ML is an appropriate solution
- If so, identify what types of algorithms might be applicable
- Feel inspired to work on and learn more about Machine Learning :-)
- Have familiarity of applying machine learning algorithm using Python

MACHINE LEARNING BASICS

WHAT IS MACHINE LEARNING

- Creating programs that can automatically learn rules from data

WHAT IS MACHINE LEARNING

- Creating programs that can automatically learn rules from data
- Traditional algorithms vs Machine Learning algorithms:

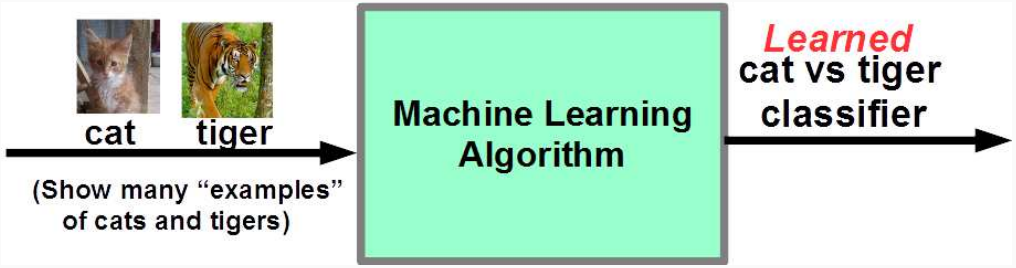
WHAT IS MACHINE LEARNING

- Creating programs that can automatically learn rules from data
- Traditional algorithms vs Machine Learning algorithms:
  - Traditional: Write programs using hard-coded (fixed) rules

WHAT IS MACHINE LEARNING

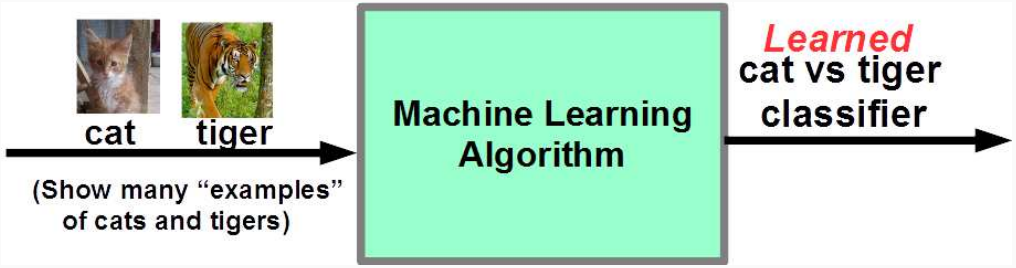
- Creating programs that can automatically learn rules from data
- Traditional algorithms vs Machine Learning algorithms:
  - Traditional: Write programs using hard-coded (fixed) rules

  - Machine Learning (ML): Learn rules by looking at some training data



- Creating programs that can automatically learn rules from data
- Traditional algorithms vs Machine Learning algorithms:
  - Traditional: Write programs using hard-coded (fixed) rules

  - Machine Learning (ML): Learn rules by looking at some training data



  - Learned rules must generalize (do well) on future "test" data (generalization capability)

MACHINE LEARNING IN REAL WORLD

Broadly applicable in many domains (e.g., internet, robotics, healthcare and biology, computer vision, NLP, databases, computer systems, finance, etc.).

- Information retrieval (text, visual, and multimedia searches)
- Machine Translation
- Question Answering
- Social networks
- Recommender systems (Amazon, Netflix, etc.)
- Speech/handwriting/object recognition
- Ad placement on websites
- Credit-card fraud detection
- Weather prediction
- Autonomous vehicles (self-driving cars)
- Healthcare and life-sciences .
- .. and many more applications in sciences and engineering

WHY MACHINE LEARNING

- Recent progress in algorithms and theory
- Growing flood of online data
- Computational power is available
- Budding industry

TYPES OF LEARNING IN ML

TYPES OF MACHINE LEARNING

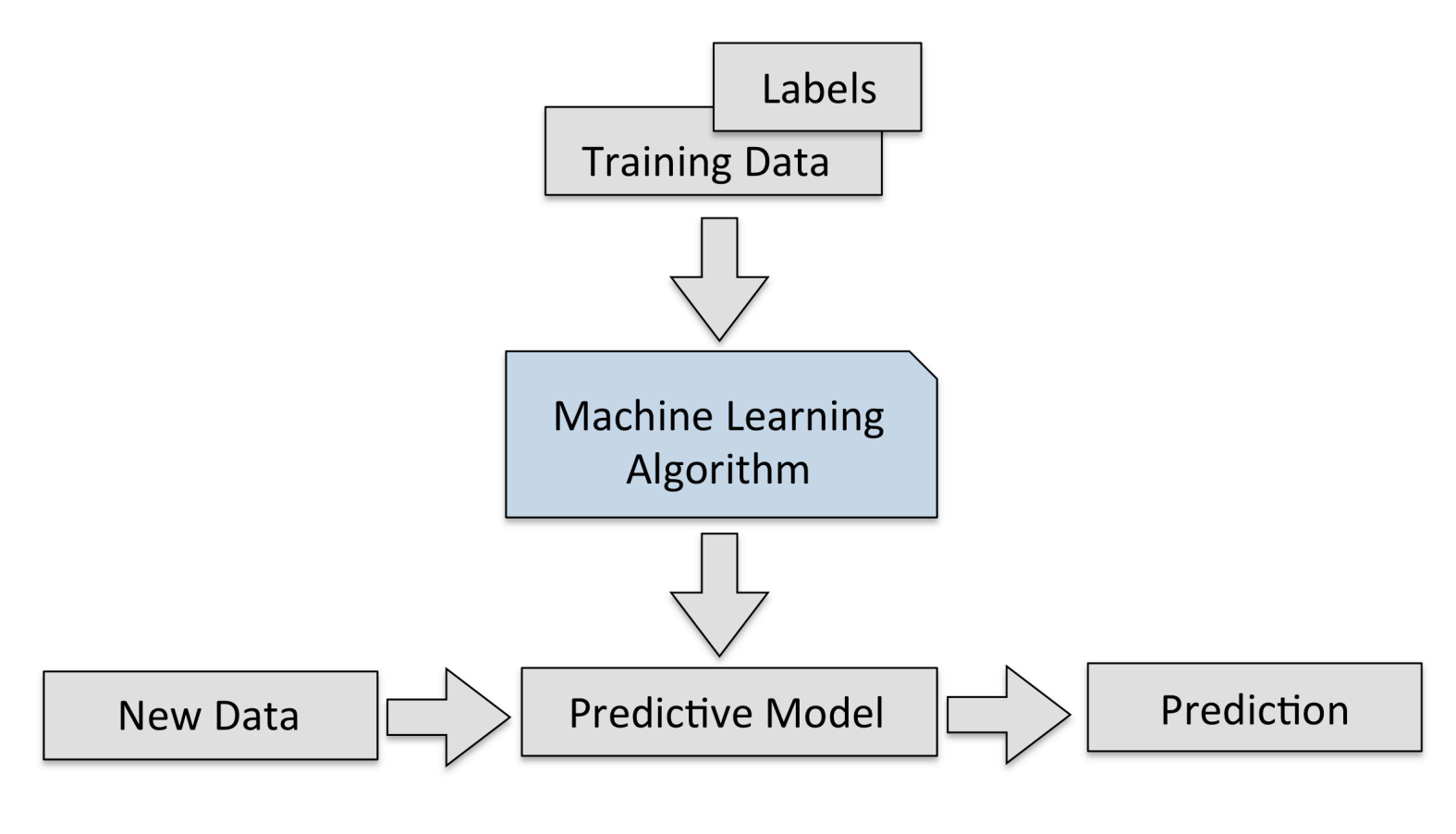
- Supervised Learning: learn a model from labeled training data, then make predictions
- Unsupervised Learning: explore the structure of the data to extract meaningful information
- Reinforcement Learning: develop an agent that improves its performance based on interactions with the environment

Note:

- Semi-supervised methods combine the first two.
- ML systems can combine many types in one system.

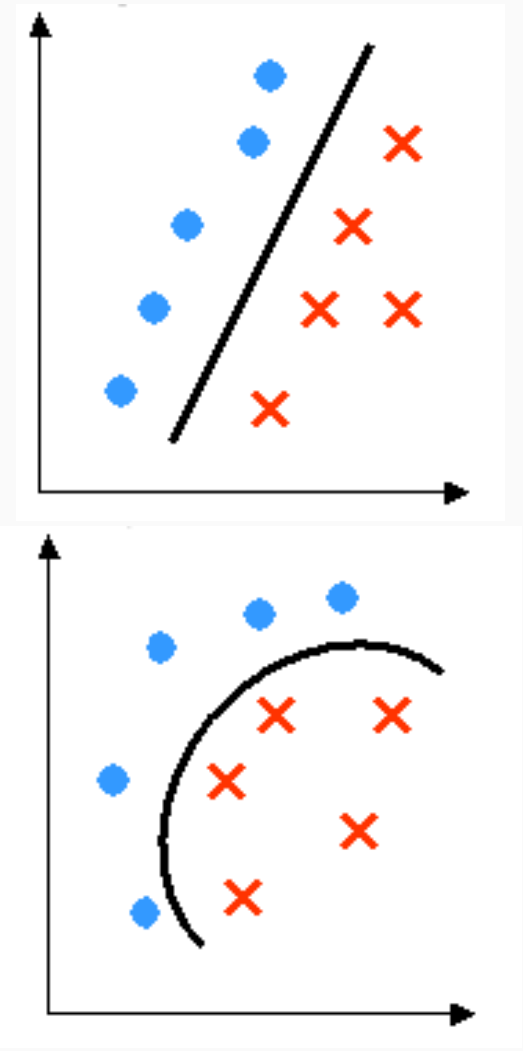
- Learn a model from labeled training data, then make predictions
- Supervised: we know the correct/desired outcome (label)
- Given: Training data as labeled instances $(x_1, y_1), ..., (x_N, y_N)$
- Goal: Learn a rule $(f : x \rightarrow y)$ to predict outputs y for new inputs x
- 2 subtypes:
  - Classification: predict a *class label* (category). e.g. spam/not spam - Many classifiers can also return a *confidence* per class
  - Regression: predict a continuous value, e.g. temperature

SUPERVISED LEARNING:FRAMEWORK



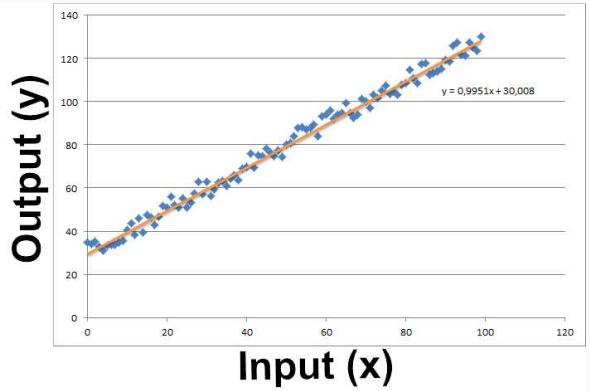
CLASSIFICATION

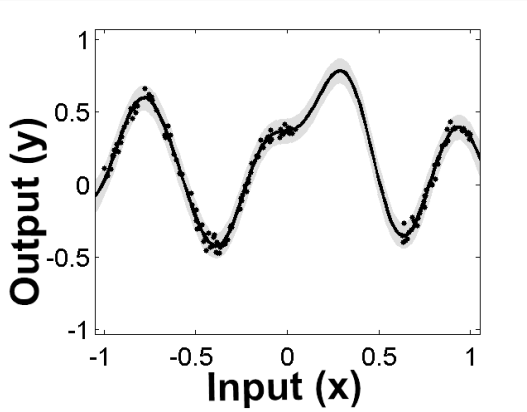
- Class labels are discrete, unordered
- Can be binary (2 classes) or multi-class (e.g. letter recognition)
- Dataset can have any number of predictive variables (predictors)
  - Also known as the dimensionality of the dataset
- The predictions of the model yield a decision boundary separating the classes



REGRESSION

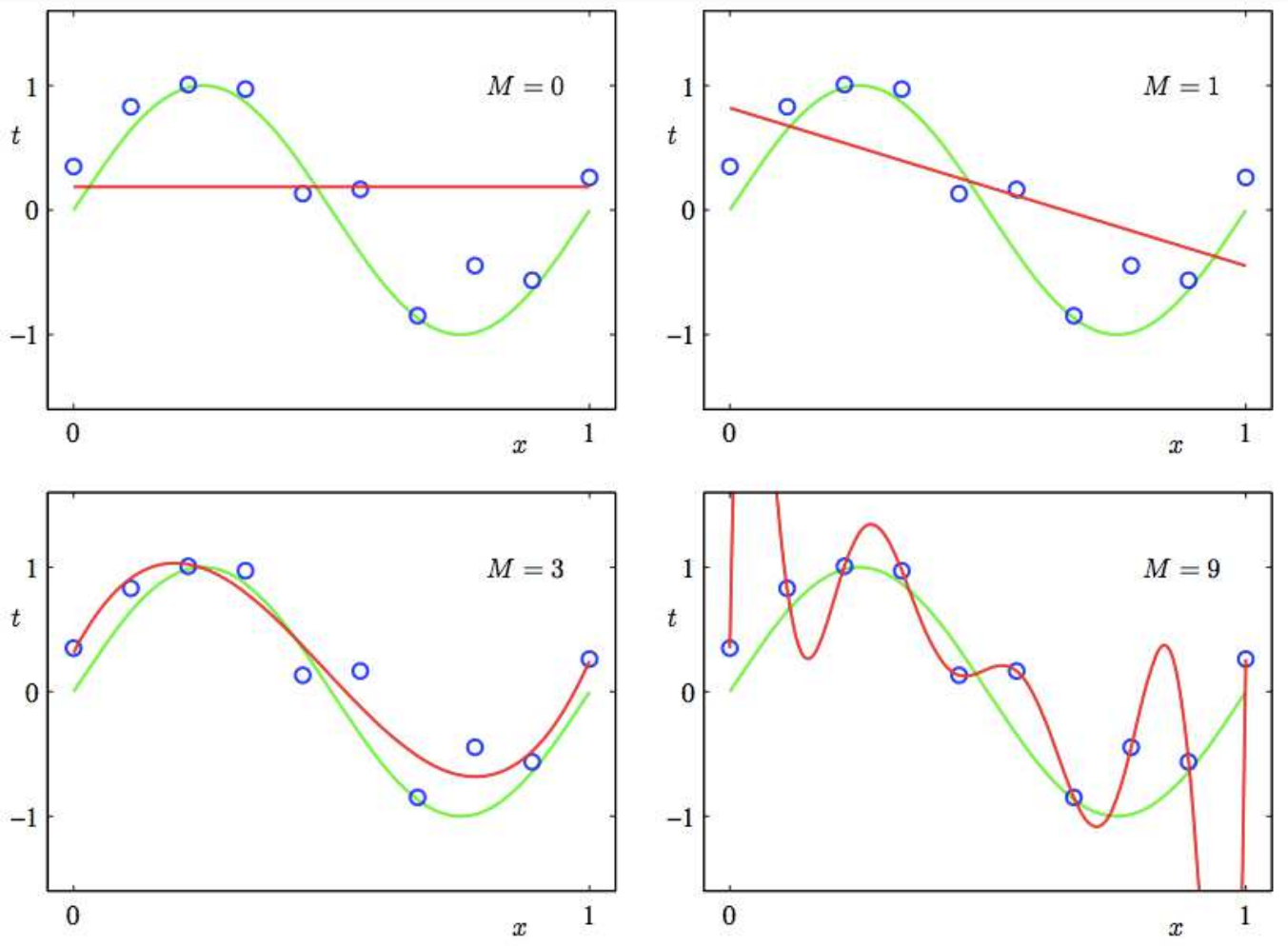
- Target variable is numeric
- Find the relationship between predictors and the target.
  - E.g. relationship between hours studied and final grade
- Linear regression (fits a straight line) or Nonlinear



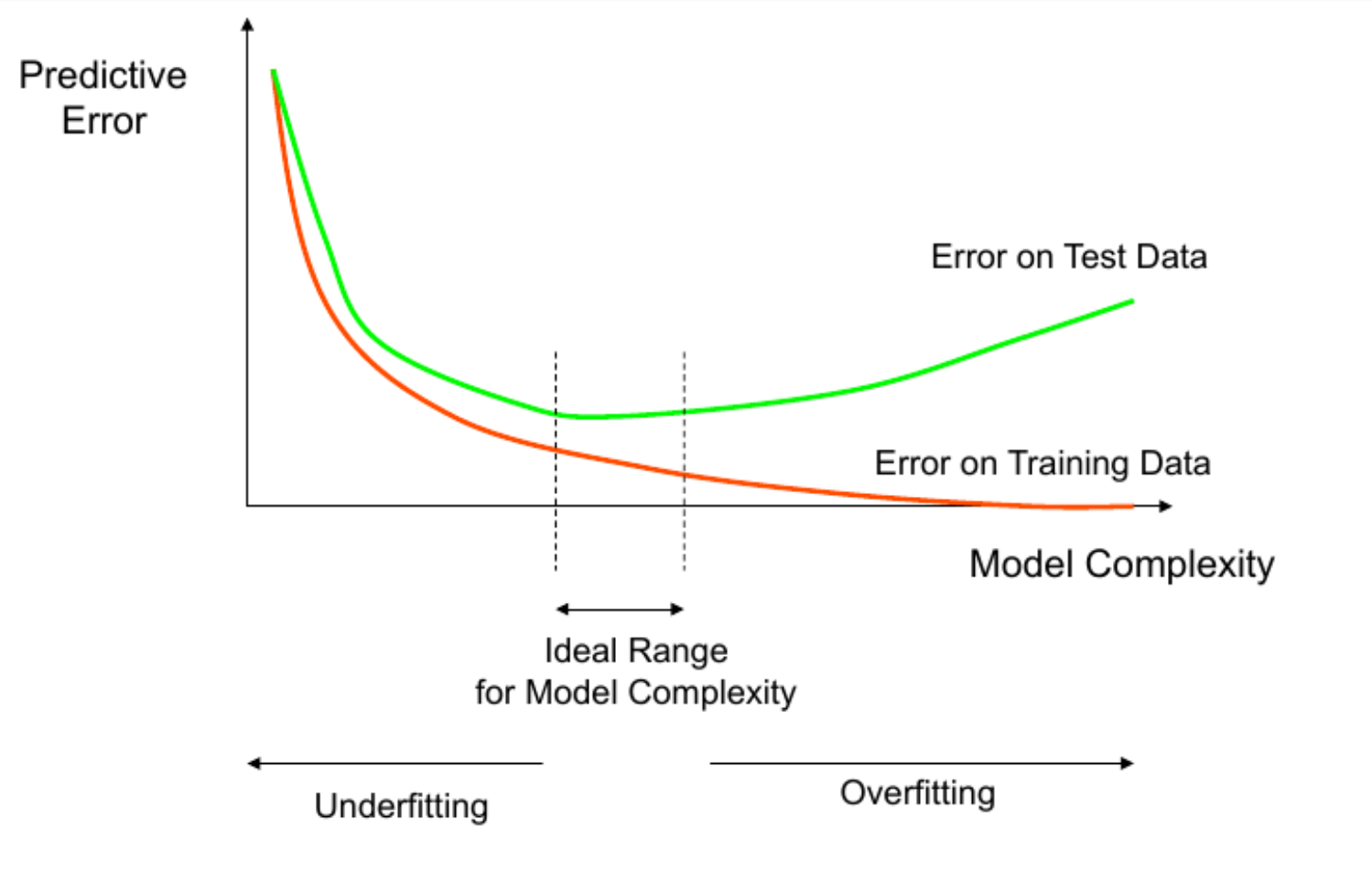


GENERALIZATION

- Generalization is crucial (must do well on test data)
- Finding the right model complexity



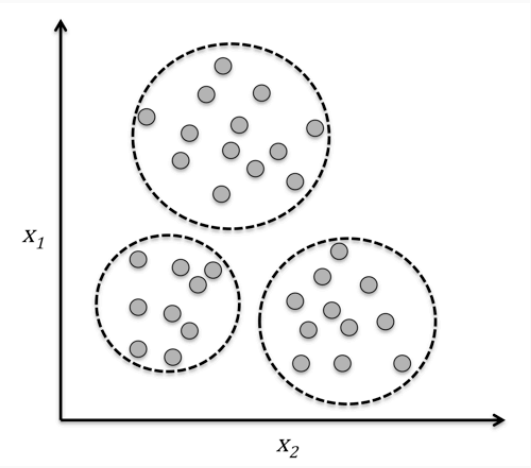
OVERFITTING VS UNDERFITTING



- Given: Training data in form of unlabeled instances $\{x_1, ..., x_N\}$
- Goal: Explore the structure of the data to extract information or to explain data
- Many types such as dimensionality reduction, clustering, association, etc.
- Also used as a preprocessing step for many supervised learning algorithms (e.g., to learn/extract good features, to speed up the algorithms, etc.)

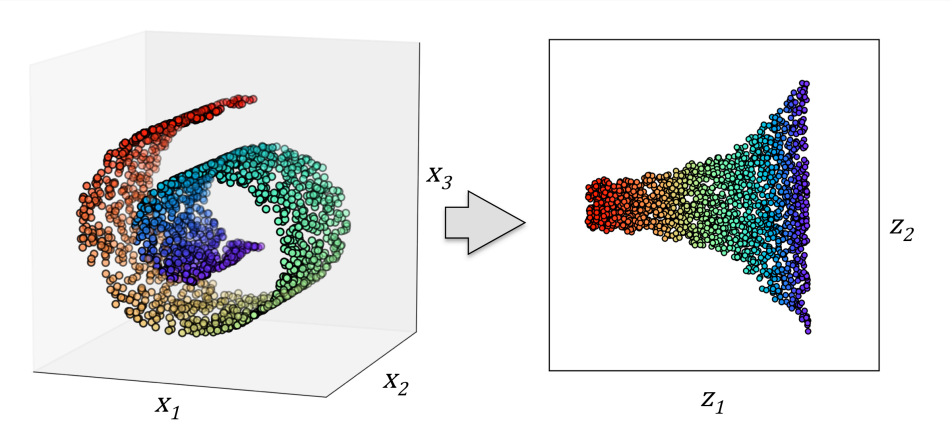
CLUSTERING

- Organize information into meaningful subgroups (clusters)
- Objects in cluster share certain degree of similarity (and dissimilarity to other clusters)
- Example: distinguish different types of customers



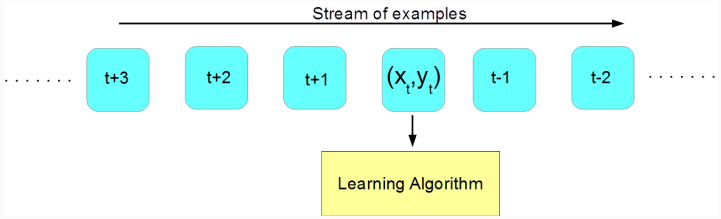
DIMENSIONALITY REDUCTION

- Data can be very high-dimensional and difficult to understand, learn from, store,...
- Dimensionality reduction can compress the data into fewer dimensions, while retaining most of the information
- Not the same as feature selection, which is typically supervised
- Is often useful for visualization (e.g. compress to 2D)

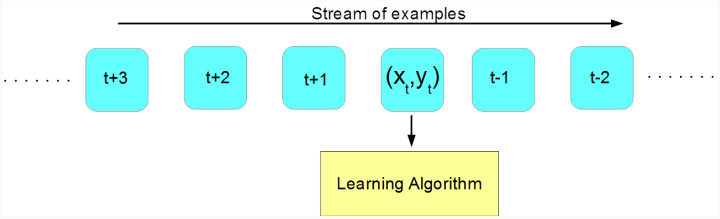


OTHER LEARNINGS

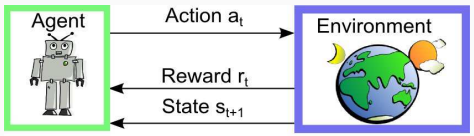
- Online Learning: Learning with one example (or a small minibatch of examples) at a time



- Online Learning: Learning with one example (or a small minibatch of examples) at a time

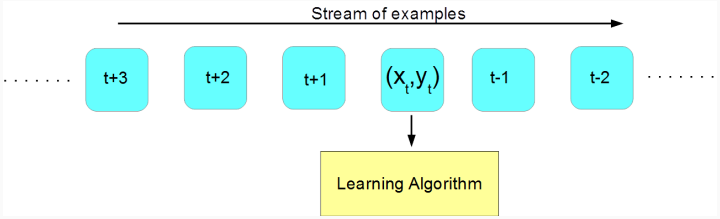


- Reinforcement Learning: Learning a "policy" by performing actions and getting rewards

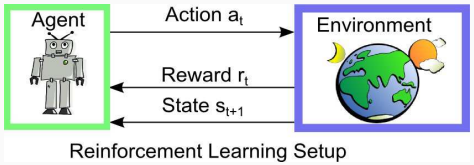


Reinforcement Learning Setup

- Online Learning: Learning with one example (or a small minibatch of examples) at a time



- Reinforcement Learning: Learning a "policy" by performing actions and getting rewards



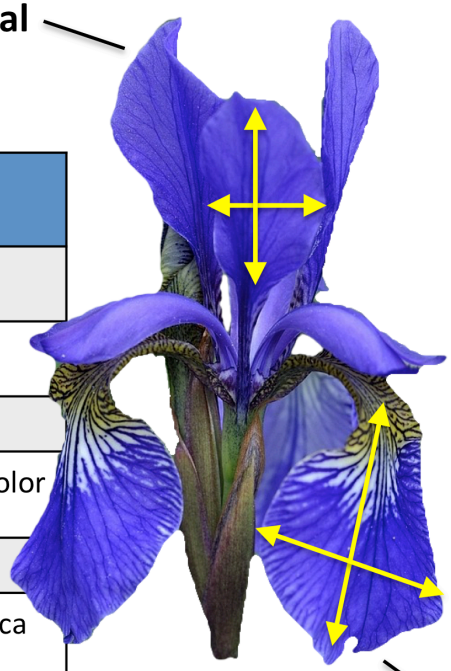
- Transfer/Multitask Learning Leveraging knowledge of solving one problem to solve a new problem

BUILDING ML SYSTEMS

BASIC TERMINOLOGY IN ML

Samples
(instances, observations)

	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica



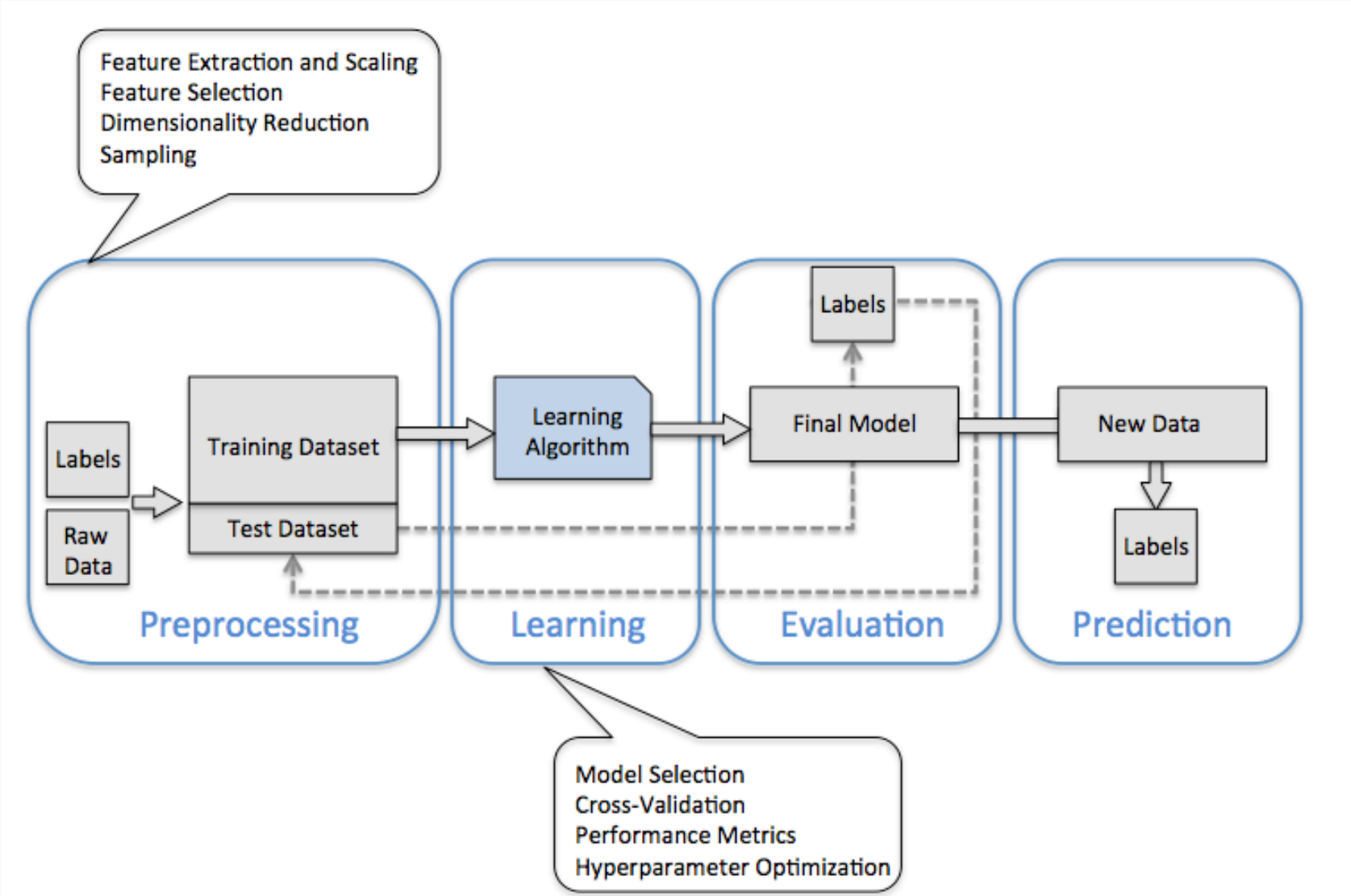
Features
(attributes, measurements, dimensions)

Class labels
(targets)

- Preprocessing: Raw data is rarely ideal for learning
  - Feature scaling: bring values in same range
  - Encoding: make categorical features numeric
  - Discretization: make numeric features categorical
  - Feature selection: remove uninteresting/correlated features
  - Dimensionality reduction can also make data easier to learn

- Learning and model selection
  - Every algorithm has its own biases
  - No single algorithm is always best (No Free Lunch)
  - Model selection compares and selection the best models
    - Different algorithms
    - Every algorithm has different options (hyperparameters)
  - Split data in training and test sets

Together they form a workflow of pipeline



- Supervised learning
  - Linear models (Ridge, Lasso, Elastic Net, ...)
  - Support Vector Machines
  - Tree-based methods (Classification/Regression Trees, Random Forests,...)
  - Nearest neighbors
  - Neural networks
  - Gaussian Processes
  - Feature selection

- Unsupervised learning
  - Clustering (KMeans, ...)
  - Matrix Decomposition (PCA, ...)
  - Manifold Learning (Embeddings)
  - Density estimation
  - Outlier detection
- Model selection
  - Cross-validation
  - Grid-search
  - Lots of metrics

- Python: Scikit-learn, Numpy, MatPlotLib, SciPy, Tensorflow, Keras,...
- Matlab pr Octave (free)
- R: Used mainly in statistics
- C++: For performance, not prototyping
- no coding required: Weka, Knime, Rapidminer, etc

Note:

- Weka are java-based. Library from weka can also be integrated with java.

TOPICS LIST (TENTATIVE)

- Nearest-neighbors methods
- Decision trees
- Naive Bayes
- Linear/non-linear regression and classification (including SVM)
- Ensemble Methods
- Model selection
- Deep Learning

We could not cover unsupervised learning, feature selection, dealing with sequence data (markov model, etc) and many more topics in ML.

BUILDING ML SYSTEMS WITH NEAREST NEIGHBOR

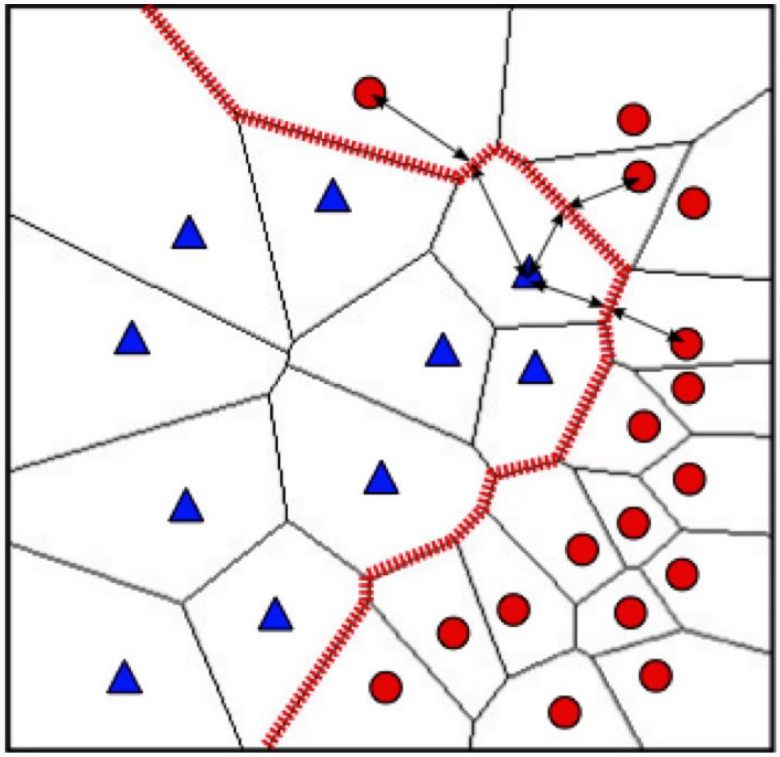
DISTANCE BASED LEARNING: NEAREST NEIGHBOR METHOD

- A distance-based supervised learning method, e.g. Euclidean distance or similarity score (for NLP)
- A simple yet very effective method in practice (if given lots of training data)
  - Probably has an error-rate that is no worse than twice of the "Bayes optimal" classifier which assumes knowledge of the true data distribution
- Also called a memory-based or instance-based or non-parametric method
- No "model" is learned here. Prediction step uses all the training data

- Requires lots of storage (need to keep all the training data at test time)
- Prediction can be slow at test time
  - For each test point, need to compute its distance from all the training points
  - Clever data-structures or data-summarization techniques can provide speed-ups
- Need to be careful in choosing the distance function to compute distances (especially when the data dimension D is very large)
- The 1-NN can suffer if data contains outliers or if amount of training data is small. Using more neighbors ($K > 1$) is usually more robust

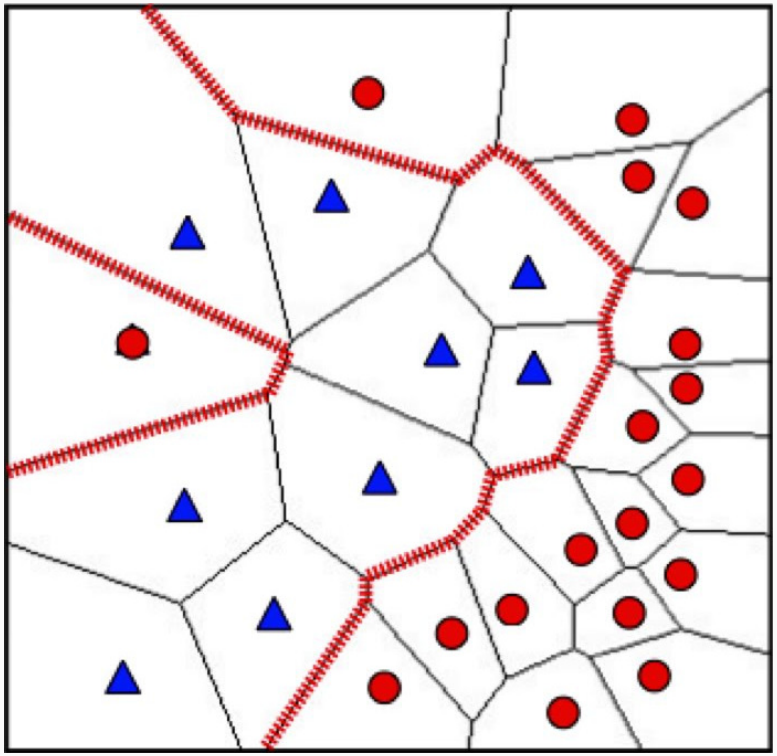
1-NN

- 1-NN induces a Voronoi tessellation of the input space
- The decision boundary is composed of hyperplanes that form perpendicular bisectors of pairs of points from different classes



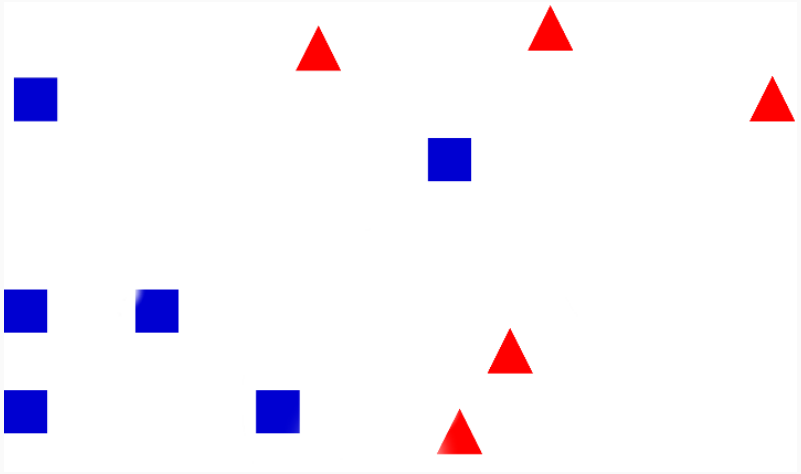
1-NN

- 1-NN induces a Voronoi tessellation of the input space
- The decision boundary is composed of hyperplanes that form perpendicular bisectors of pairs of points from different classes
- The decision boundary can drastically change when the data contains some outliers



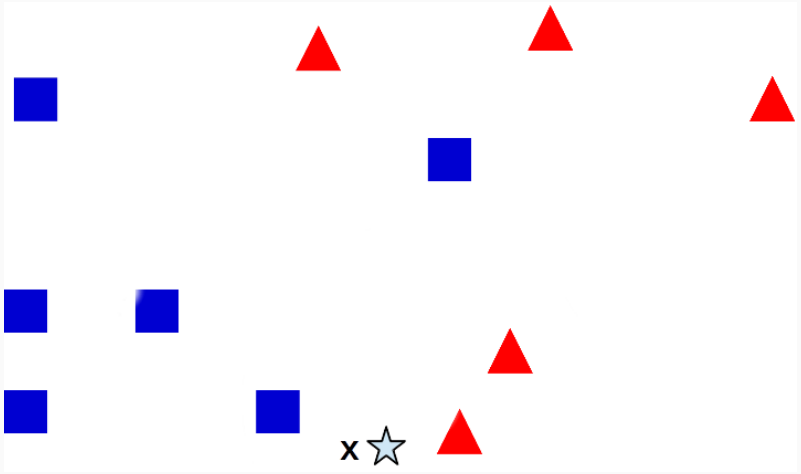
K-NN

- Makes one-nearest-neighbor more robust by using more than one neighbor
- The K -NN prediction rule: Take a majority vote (or average) of the labels of K $>$ 1 neighbors in the training data



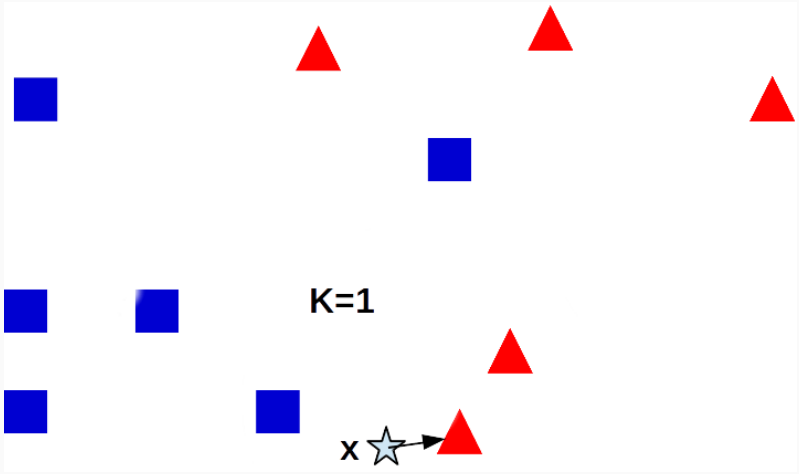
K-NN

- Makes one-nearest-neighbor more robust by using more than one neighbor
- The K -NN prediction rule: Take a majority vote (or average) of the labels of K $>$ 1 neighbors in the training data



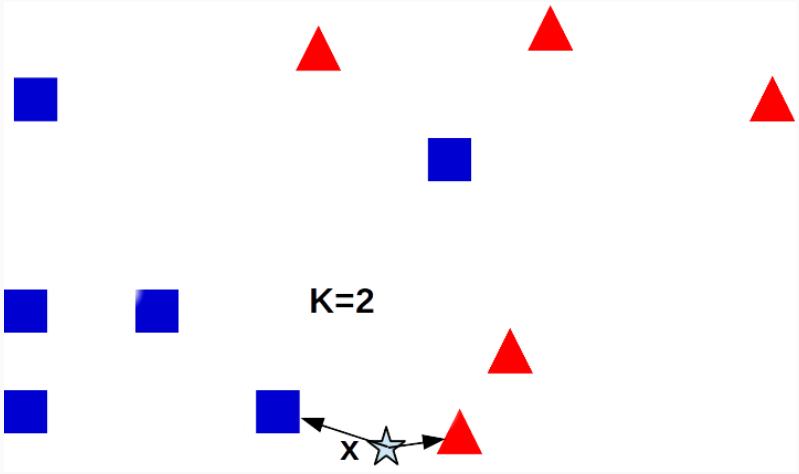
K-NN

- Makes one-nearest-neighbor more robust by using more than one neighbor
- The K -NN prediction rule: Take a majority vote (or average) of the labels of K $>$ 1 neighbors in the training data



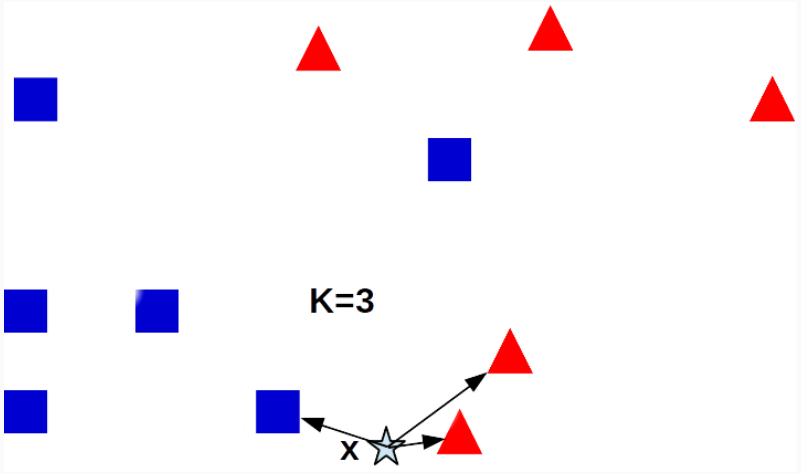
K-NN

- Makes one-nearest-neighbor more robust by using more than one neighbor
- The K -NN prediction rule: Take a majority vote (or average) of the labels of K $>$ 1 neighbors in the training data



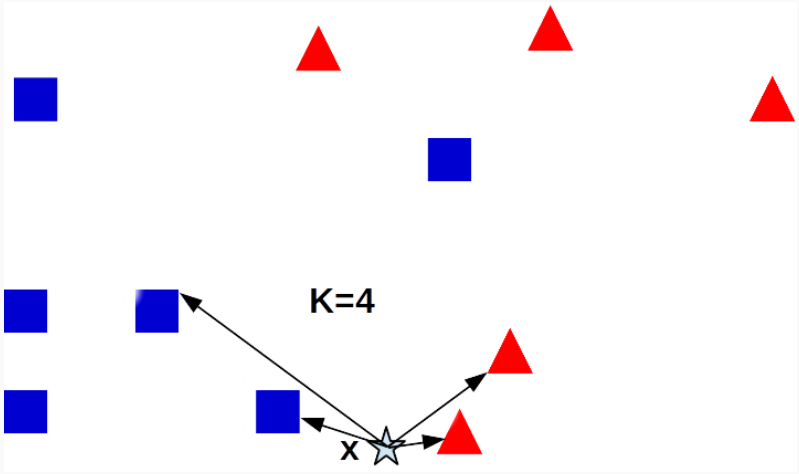
K-NN

- Makes one-nearest-neighbor more robust by using more than one neighbor
- The K -NN prediction rule: Take a majority vote (or average) of the labels of K $>$ 1 neighbors in the training data



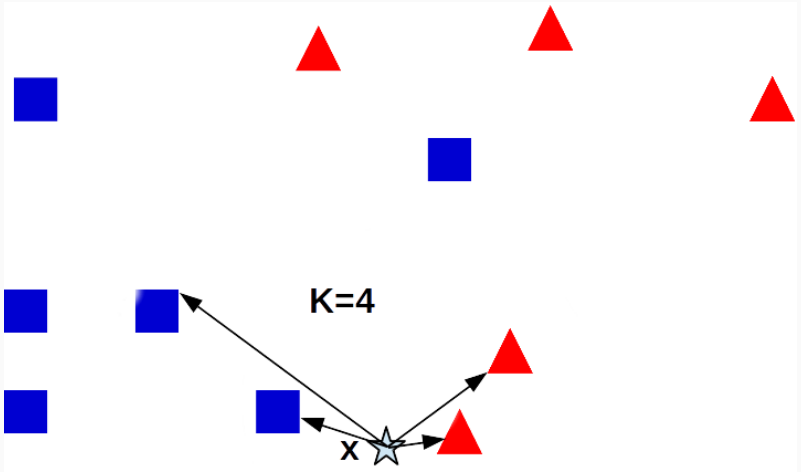
K-NN

- Makes one-nearest-neighbor more robust by using more than one neighbor
- The K -NN prediction rule: Take a majority vote (or average) of the labels of K $>$ 1 neighbors in the training data



K-NN

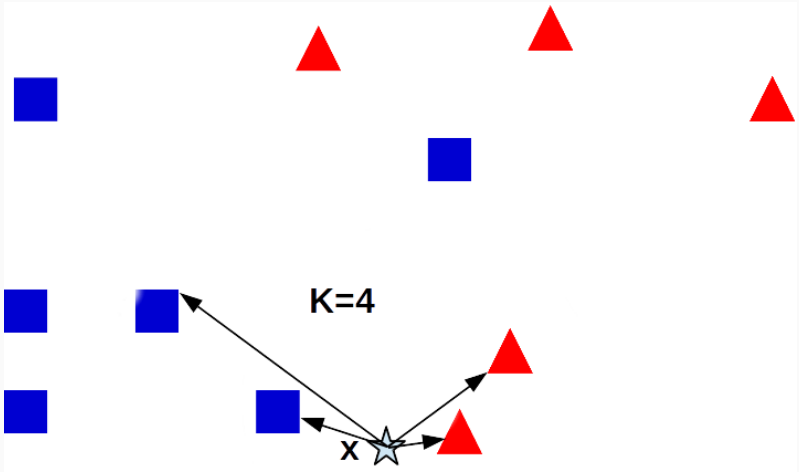
- Makes one-nearest-neighbor more robust by using more than one neighbor
- The K -NN prediction rule: Take a majority vote (or average) of the labels of K $>$ 1 neighbors in the training data



- For classification, we usually take the majority labels from the K neighbors
- For regression, we usually average the real-valued labels of the K neighbors

K-NN

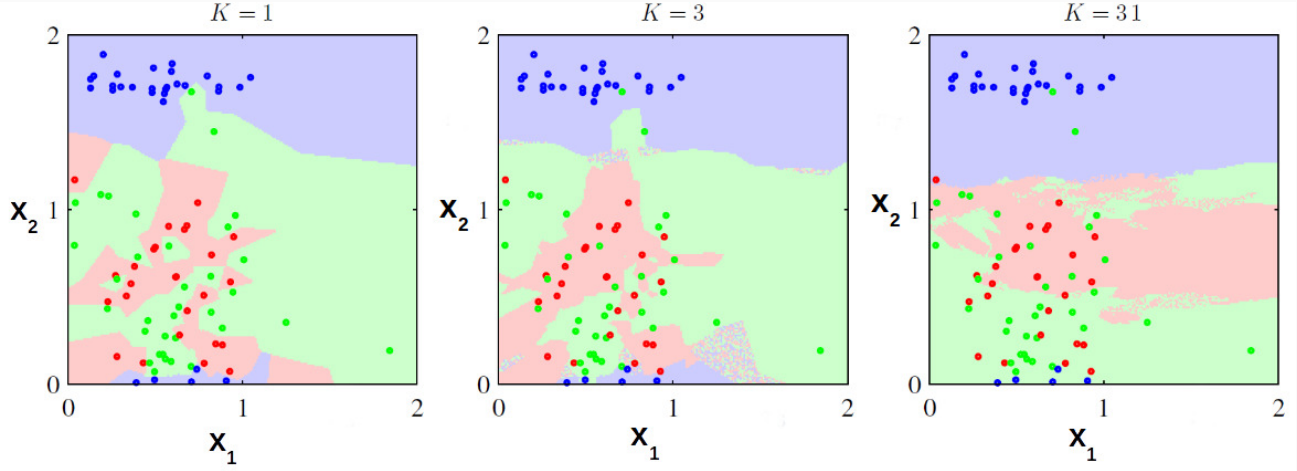
- Makes one-nearest-neighbor more robust by using more than one neighbor
- The K -NN prediction rule: Take a majority vote (or average) of the labels of K $>$ 1 neighbors in the training data



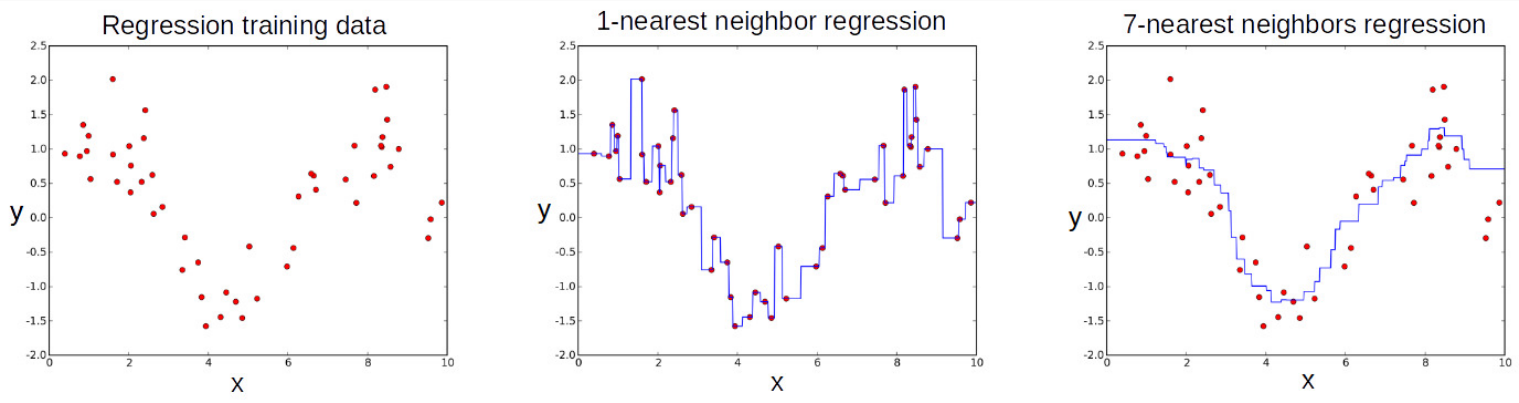
- For classification, we usually take the majority labels from the K neighbors
- For regression, we usually average the real-valued labels of the K neighbors
- The "right" value of K needs to be selected (e.g., via cross-validation)

K-NN: DECISION BOUNDARIES

Larger K leads to smoother decision boundaries



K-NN FOR REGRESSION



Now we build k-NN based ML systems using scikit-learn

OBSERVE...

- Investigate and analyse how the value of k affects the performance
- Investigate when some features are used vs when all features are used
- Comments on overfitting and underfitting

CONCLUSION

- We've covered the main machine learning concepts
- We used scikit-learn to build a first model
- We met our first algorithm (kNN)