COMP 204: Python programming for life sciences

Introduction to machine learning

Mathieu Blanchette, based on material from Yue Li, Christopher Cameron, and Carlos Gonzales

Remaining of this course: Advanced topics

The rest of the semester will be spent introducing advanced topics in programming: machine learning, BioPython, etc.
Those topics will be covered in the final exam, but not at the same depth as material covered until now.

Introduction to Machine Learning

Machine learning is a branch of Artificial Intelligence that aims to design systems that can learn from data or from experience.

Until now, all the problems we encountered were solved by the programmer (you) writing programs that describe exactly the sequence of steps
and rules that need to be taken in order to achieve the desired result.

Machine learning programs learn how to automatically adjust their behavior in order to perform a certain task better. ML is data-driven (as opposed to rule-based), leading to novel scientific discoveries.

ML applications are everywhere: Science, medicine, finance, marketing, games, etc. etc.

Problem: cat vs. bird

How would you write a computer program to identify a cat or bird in a photo?

Cats

Birds

Distinguishing features between cats and birds

There are some obvious features to distinguish cats and birds:

- **Cats**: fur, ears, a tail
- **Birds**: beaks, feathers, no teeth

How would you tell a computer to recognize a beak? fur? a tail?

- Writing a classical program to do so would be hugely complicated
- Would fail when the cat/bird has unusual posture, color, etc.

Humans are really really good at distinguishing cats from birds! How do we do it?

- We learn from examples: our parents pointed out cats and birds in real life or books.
- We *automatically* learned what the features of each animal are
- Human learning happens because the connections between neurons in our brain adjust as we learn.

Examples of ML application

character recognition

- categorize images of handwritten characters by the letters represented

face detection

- find faces in images (or indicate if a face is present)

medical diagnosis

- diagnose a patient as a sufferer or non-sufferer of some disease, based on set of symptoms or imaging data
- predict the required dosage for successful treatment

fraud detection

- identify credit card transactions (for instance) which may be fraudulent in nature

Examples of ML application

Detecting disease-causing mutations

- We don't know how to program it because we don't fully understand the functions of our genome
- We have very limited understanding of the physiology underlying most of the complex phenotypes (e.g. Alzheimer's disease, cancers) and how they interact with the environments (e.g., nutrition, exposed to radiation, neighbourhoods)
- There are unknown causal factors that we may not even observe or not yet have a way to measure them (e.g., uncharacterized pathways)

Machine learning can help when:

- We have collected enough example where the mutations and phenotypes are known, so we can learn what mutations cause what diseases

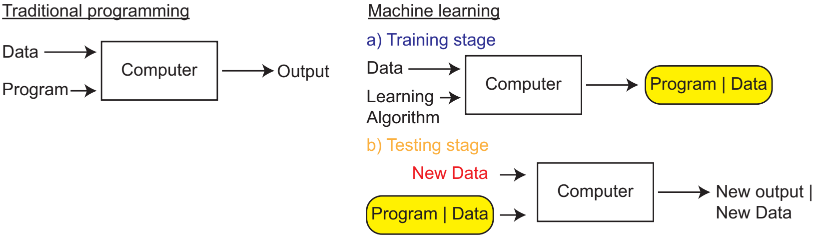
‘Traditional’ programming vs. machine learning

Traditional programming

- Program is written first *independent* of the data
- Program is applied to data to produce an output
- The program does not adapt to the data: it remains the same throughout its execution

Machine learning

- Program (or parameters of the program) adjusts itself automatically to **fit** the data
- End result is a program that is trained to achieve a given task



Types of learning tasks

- Supervised learning:
  - Given examples of inputs (e.g., genotype) and corresponding desired outputs (e.g., disease), predict outputs on future unseen inputs, e.g., classification, regression, time series prediction
  - Often the connotation of machine learning (people often ask how accurate is your model?)
- Unsupervised learning
  - Create a new representation of the input, e.g., form clusters, extract latent continuous features, compression
  - This is the new frontier of machine learning because most big datasets do not come with labels
- Reinforcement learning
  - Learn action to maximize payoff (e.g., robotics, self-driving vehicle)
  - An important research area but not the focus of this class

Supervised learning

In supervised learning, the algorithm is given examples along with their correct labels. This is called the training data.

Image	Label
	Cat
	Bird
	Cat
	Cat
	Bird

Goal: Learning how to classify new images:

 ?

Types of supervised learning tasks

Three general types of prediction tasks:

1. **classification**: the goal is to predict which of a predefined set of classes an example belongs to
   - Cat vs Bird?
   - Cancer vs normal?
   - digit recognition: 0 or 1 or 2 or 3 or 4... ?
2. **regression**: goal is to predict a real value
   - What will the price of oil be tomorrow?
   - How fast will this tumour grow?
3. **probability estimation**: goal is to estimate a probability
   - will it rain tomorrow?
   - will this drug be effective on this patient?

Supervised learning = Learning a function

We can express the goal of learning as being to estimate an unknown function $f(x)$, where

- x is an example (e.g. an image, or the set of symptoms of a patient)
- $f(x)$ is the thing we want to predict
  1. **classification**: $f(x)$ is a class (e.g. Cat or Dog)
  2. **regression**: $f(x)$ is a real value
  3. **probability estimation**: f(x) is a probability

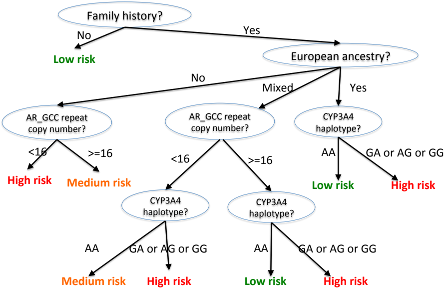
Types of ML algorithms

There are many types of ML algorithms:

- **logistic regression**: https://en.wikipedia.org/wiki/Logistic_regression
- **polynomial regression**: https://en.wikipedia.org/wiki/Polynomial_regression
- **decision tree**: https://en.wikipedia.org/wiki/Decision_tree
- **random forest**: https://en.wikipedia.org/wiki/Random_forest
- **artificial neural network**: https://en.wikipedia.org/wiki/Artificial_neural_network
- **support vector machine**: https://en.wikipedia.org/wiki/Support_vector_machine
- and many more...

Decision tree: prostate risk cancer

Goal: Predict the prostate cancer risk level of an individual
Input data: Family history, ancestry, AR_GCC copy number, CYP3A4 genotype.



Challenge: Having observed patients that developed prostate cancer, and those who didn't, *write a program that learns* what is the best decision tree.

Key elements of ML

Every ML algorithm has three components:

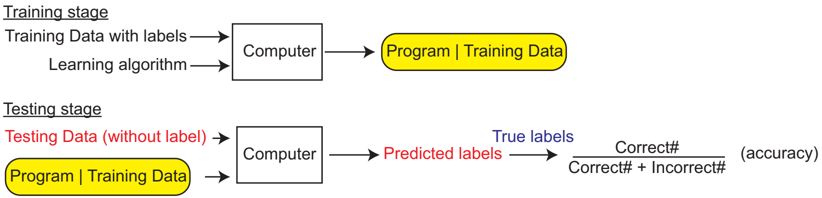
1. **representation**: how to represent knowledge?
   - how should the input information be represented?
   - what type of predictor should be used?
2. **evaluation**: how to evaluate candidate predictors?
   - accuracy, prediction and recall, squared error, likelihood, etc.
3. **optimization**: the process by which we will build our predictive model to optimize performance?
   - there are a lot of possible models (e.g. many different decision trees)
   - how do we select the ideal model?

Evaluating machine learning algorithms

- How can we get an *unbiased* estimate of the accuracy for a learned model?
- Goal: Estimate accuracy of predictor on examples it has not seen as part of its training.

Training data vs Testing data

- split available data into **training** and **testing** datasets
- create a learned model from the training data
- measure accuracy of trained model by applying it to the testing data



Cat vs. bird ML example

total data: labeled pictures of cats and birds (50K each)

training data: labeled pictures of cats and birds (45K each)

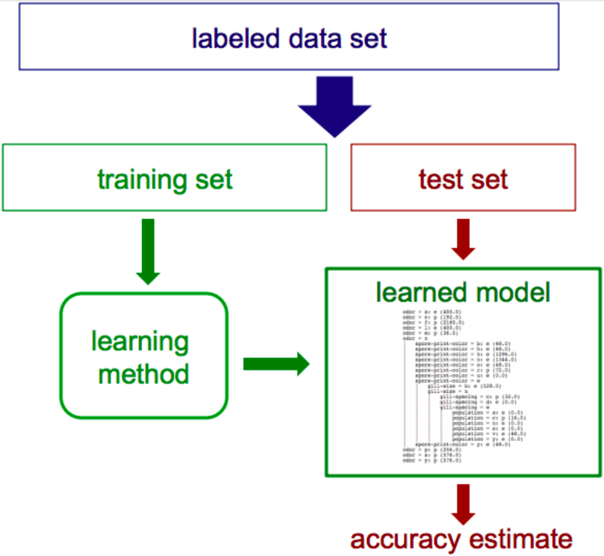
- model input is a representation of the example photo
- label is either '0' (cat) or '1' (bird)

testing data: labeled pictures of cats and birds (5K each)

ML steps:

1. create learned model from examples in training data
    - implement ML algorithm and apply to examples
2. predict on previously unseen examples
    - apply learned model to testing data
3. compare model predictions against known labels
    - calculate accuracy measure

Evaluating ML algorithms #2



Python's scikit-learn module

Over the next two lectures

- we're going to perform some basic machine learning
- using Python's scikit-learn module

scikit-learn API:
http://scikit-learn.org/stable/modules/classes.html

scikit-learn tutorials:
http://scikit-learn.org/stable/

Intro to reinforcement learning

Suppose you have an humanoid robot with legs, arms, etc.
There is a motor at each joint of the robot, and you get to control those motors.
Your task is to write a program that controls a humanoid robot to make it walk.

Walking involves precisely controlling each of the motors in order to move forward.
There are dozens of motors involved, each needing to behave in the right way and at the right time.
It's super complicated!!

Reinforcement learning

Reinforcement learning lets machines learn in the same way humans do: learning from experience, rather than by being shown labeled examples.

Approach:

- We start with a robot that doesn't know how to walk, but moves its "muscles" randomly.
- The goal of the robot it to reach a certain destination (e.g. its "mother" at the other end of the room).
- When it reaches its mother, it gets a reward (satisfaction).
- Over time, it realizes that certain actions seem to lead to better rewards (reaching destination faster).
- It slowly learns to adjust its behavior to maximize its reward
- And eventually, we get this...

Extra - reinforcement learning example

Learning to walk

https://www.youtube.com/watch?v=gn4nRCC9TwQ