Computer Science 5703-001

Machine Learning Practice

Andrew H. Fagg
Symbiotic Computing Laboratory
School of Computer Science

Constructing Models

- Start with observations (data) drawn from the world
  – Motion of an object, force applied to that object

- Models relate different types of observations to one-another

$$F = m \times a$$

What Makes a Good Model?

What Makes a Good Model?

A good model:

- Is simple
- Explains the observations that have already been made
- Is predictive of future observations

Machine Learning

What is it about?

Machine Learning

Fundamentally: ML is about using data to automatically construct a model. We would like:

- The model to produce meaningful output given novel situations
- The model to give us insights into the nature of the data and the underlying phenomena

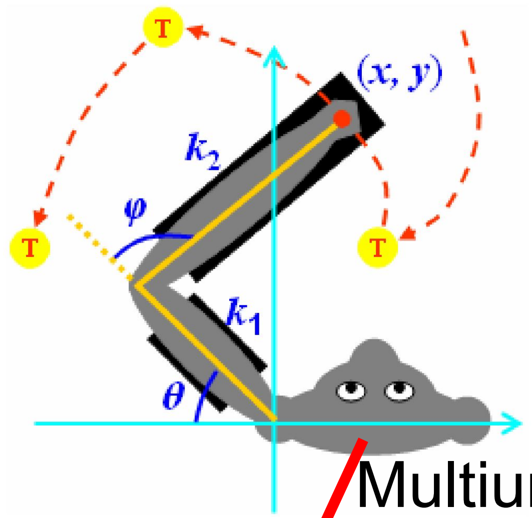
Example: Brain-Machine Interfaces

- Goal: to develop a direct connection from the brain to an advanced prosthetic device
- Approach:
  - Electrodes in the primary motor cortex “listen” to individual neurons or small clusters of neurons
  - Cortical neurons communicate by emitting sequences of pulses (“spikes” or “action potentials”) at different rates
  - Use a model to decode these pulses in terms of the intent to move the arm

Brain-Machine Interfaces

Estimate of intended movement

Command prosthetic arm



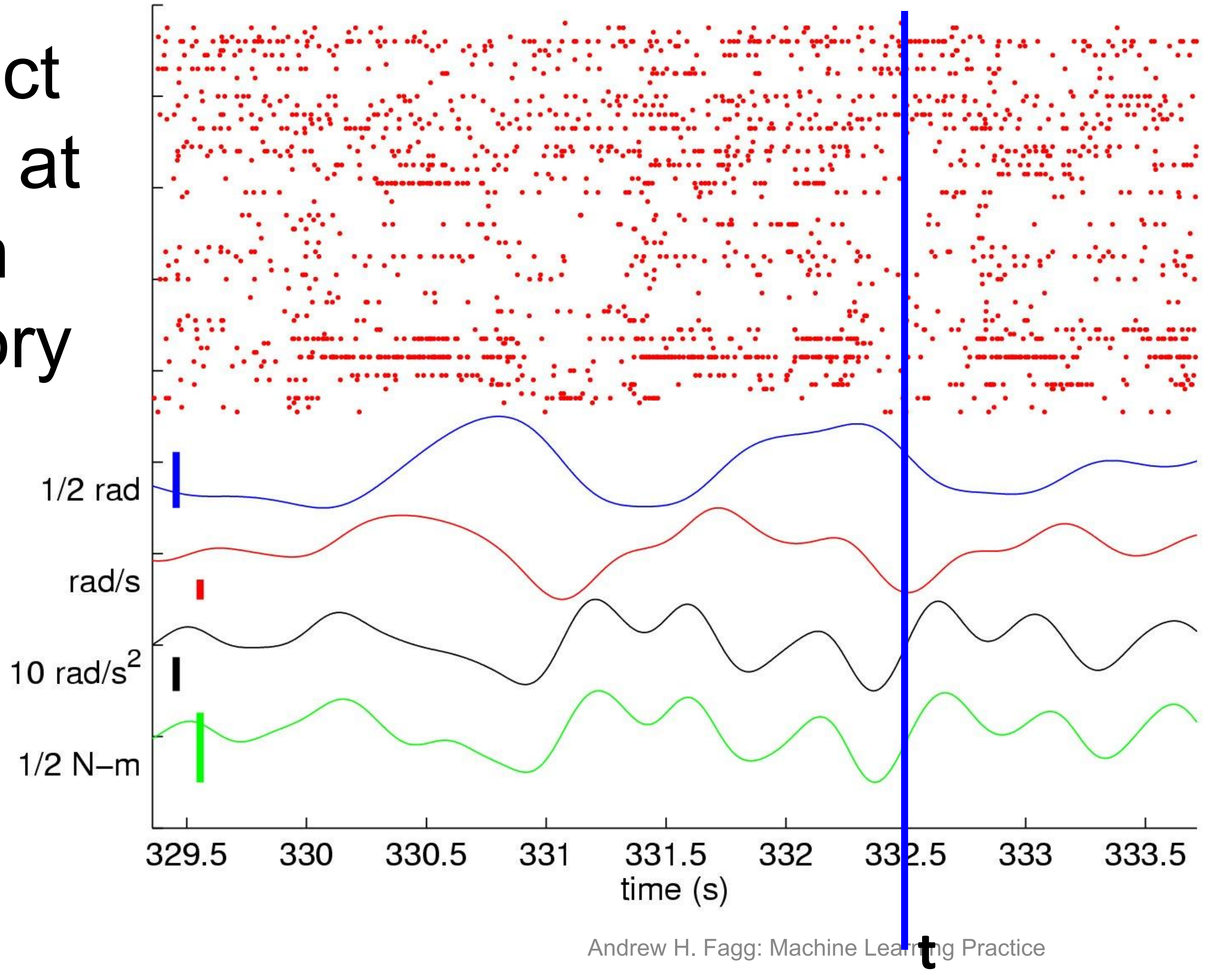
Multiunit recording

Predictive model

In collaboration with Nicholas G. Hatsopoulos and Lee E. Miller

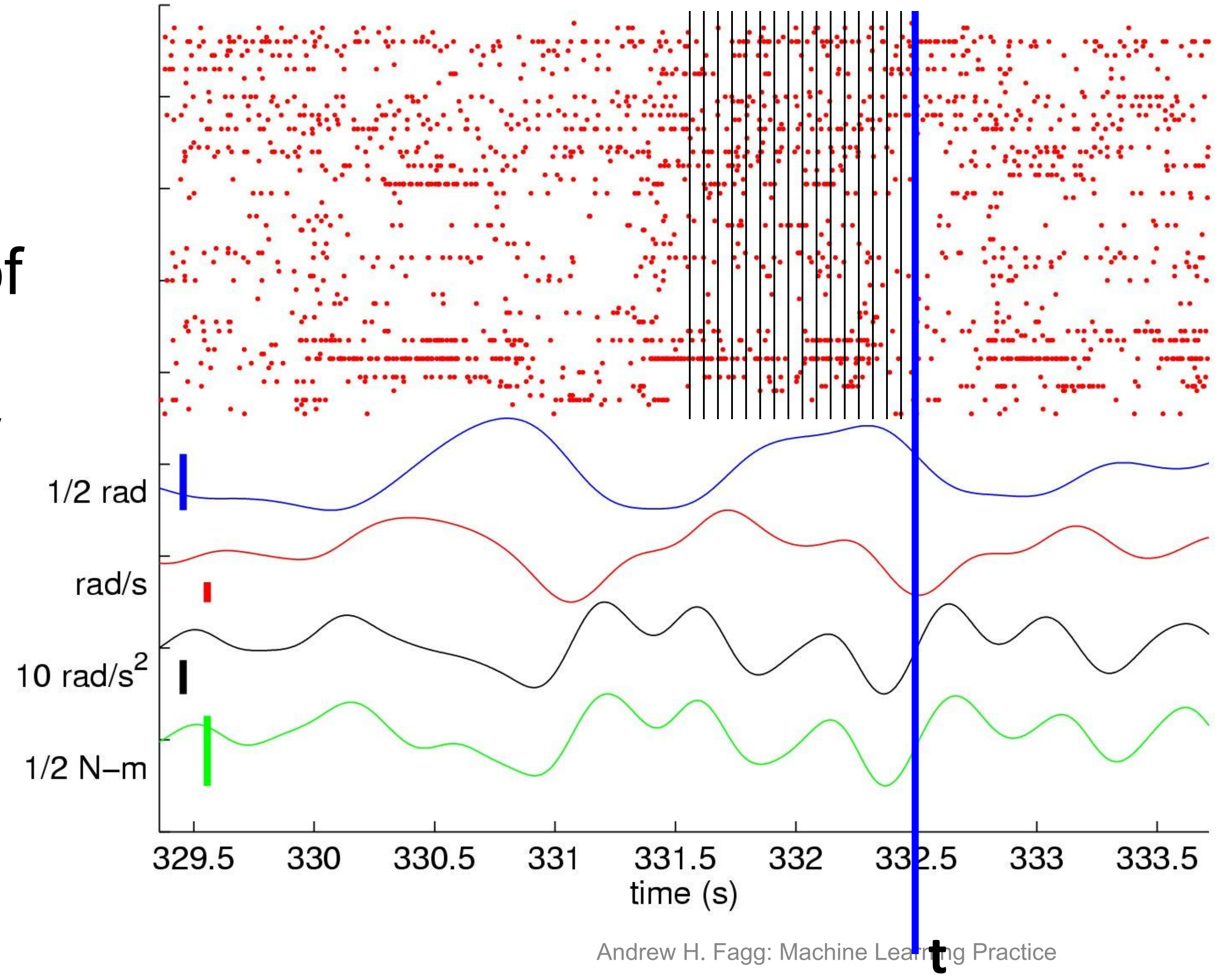
Decoding Arm State

Want to predict arm motion at time **t** given recent history of spiking behavior



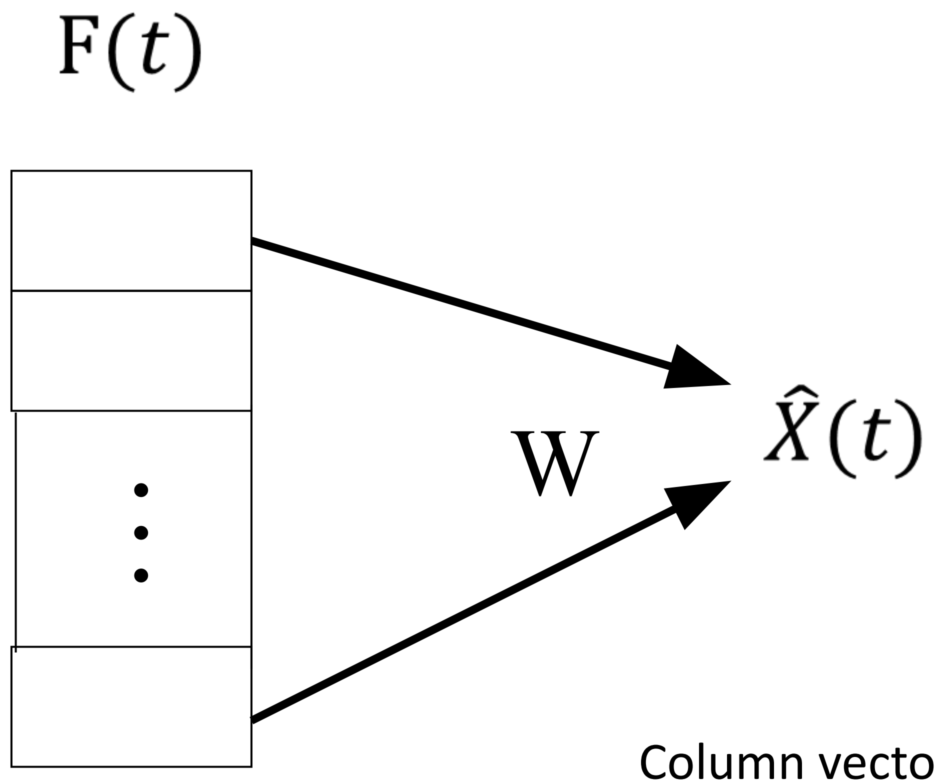
Decoding Arm State

50ms bins: 20 descriptors of neural activation for each cell



Linear Model

Each feature (F_i) is a count of spikes by a neuron for a 50 ms bin

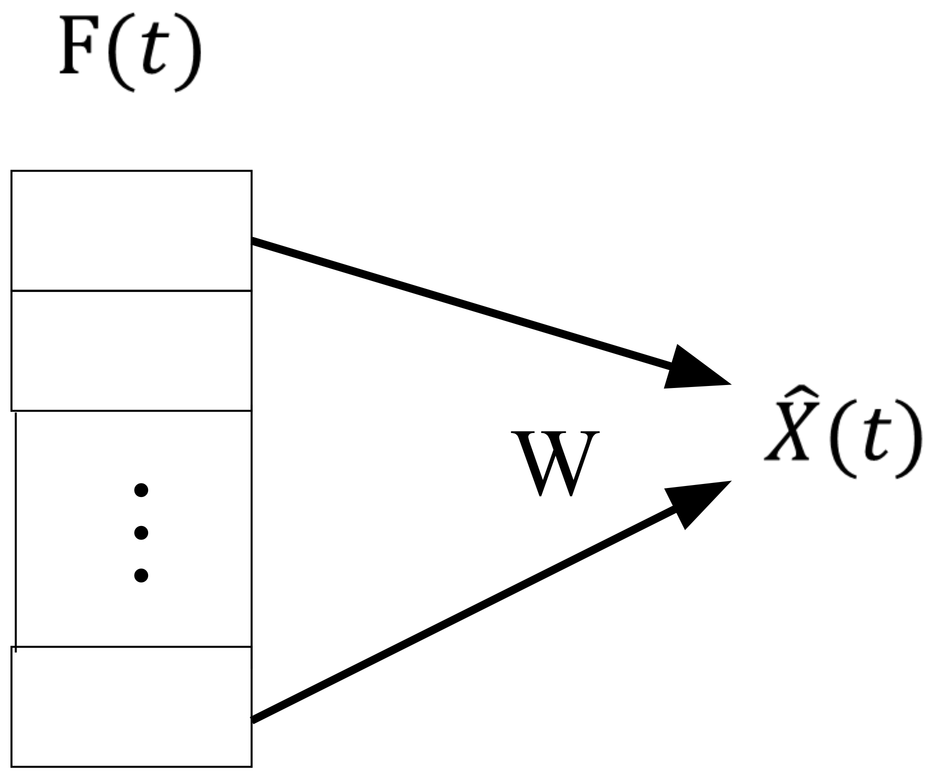


Column vector encoding spike counts for N cells at T taps up to time t

$$\hat{X}(t) = g_W(F(t)) = W^T F(t)$$

Linear Model

Each feature (F_i) is a count of spikes by a neuron for a 50 ms bin



$$\hat{X}(t) = g_W\big(F(t)\big) = W^T F(t) = \sum_{i=0}^{N-1} w_i \times F_i(t)$$

Training a Linear Model

Gathering the data:

- Monkey makes a sequence of reaches
- Simultaneously observe the movement of the monkey’s arm and the neural activity
- This provides a set of example input / output examples for our model

Training a Linear Model

- Linear model works well for this problem:

$$\hat{X}(t) = \sum_{i=0}^{N-1} w_i \times F_i(t)$$

Training a Linear Model

- Linear model works well for this problem:

$$\hat{X}(t) = \sum_{i=0}^{N-1} w_i \times F_i(t)$$

- Cost function:

$$E = \frac{1}{n}\sum_{t}\left(X(t) - \hat{X}(t)\right)^2$$

- Learning algorithm: pick the w_i's so as to minimize E

Using Our Model

Given new observations of neural spiking patterns, we can:

- Predict how the monkey will move her arm
- Use these predictions to drive the motion of the prosthesis

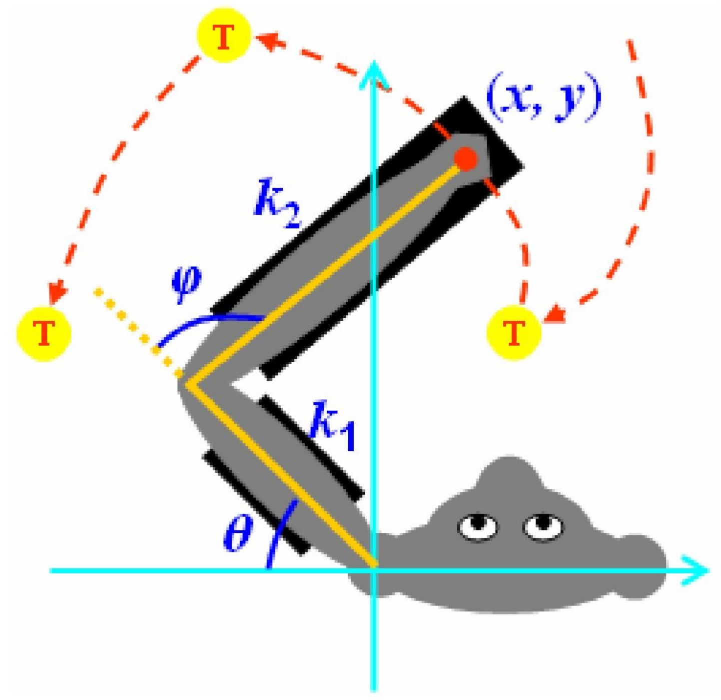
Classes of Models

Defined by the data type of the output. Very broadly:

- Continuous output: regression-type models
- Categorical output: classifier models

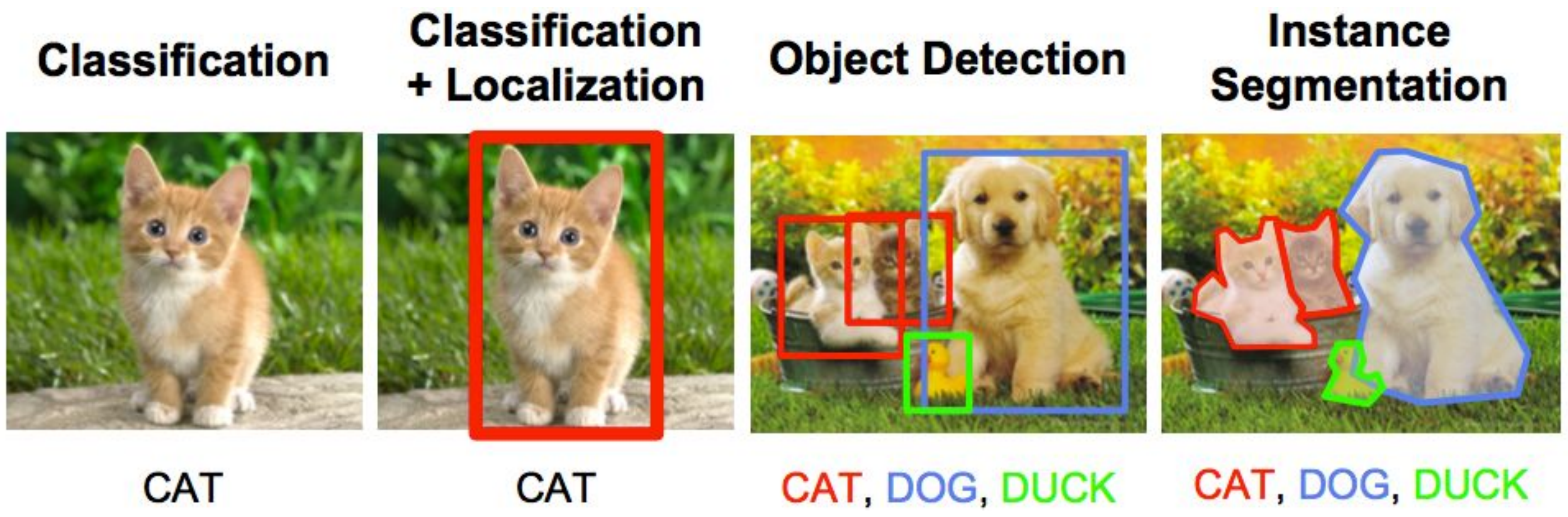
Regression-Type Models

- Continuous output
- In our brain-machine interface example: what velocity should the arm be moving at given the recent history of neural activity patterns?



Classification-Type Models

- Classification: given an input, which one of several classes does the input belong to?
- Can be crisp (choose exactly one class)
- Or can be probabilistic (each class is assigned a probability)



https://i.stack.imgur.com

Classes of Machine Learning Problems

What information is provide at the time of training?

Classes of Machine Learning Problems

Supervised learning:

- Training set contains input / output (labels) pairs
- Outputs could be continuous, probabilistic or categorical

Classes of Machine Learning Problems

Unsupervised learning:

- The training set contains only inputs
- Fundamental question: what is the structure of these inputs?
  - A common case: algorithm assigns categorical labels to each of the inputs (this is called *clustering*)
  - But we can also ask continuous questions. For example: are there linear or nonlinear manifolds that the data live on?

...

Classes of Machine Learning Problems

Semi-Supervised learning:

- Part of the training set contains input / output pairs
- The rest of the training set contains only inputs

- Using all of the data can yield a better model than if we only used the labeled data

Classes of Machine Learning Problems

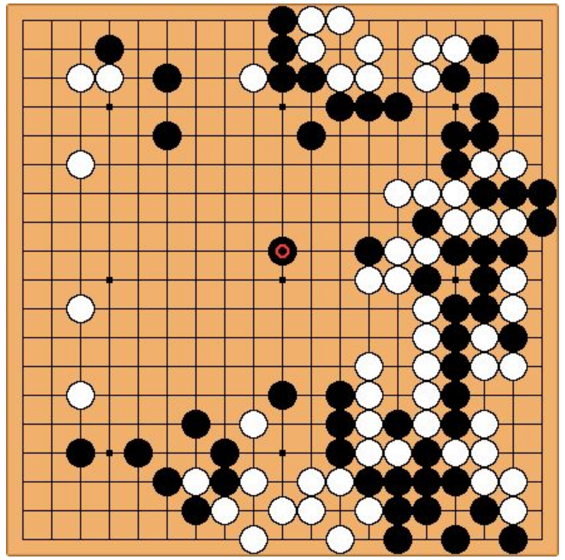
Reinforcement learning:

- Different than direct prediction or classification: RL is about taking sequences of actions in some environment
- At each step:
  - In response to an input, the model (agent) produces some action
  - The feedback signal is an evaluation of the results of this and previous actions

Classes of Machine Learning Problems

Reinforcement learning:

- Common reward types:
  - – How much time did it take to execute an action?
  - – How much energy did an action take?
  - – Did the agent win the game?



- Learning problem: for a given state, what is the action that maximizes the expected sum of rewards over time?

https://senseis.xmp.net

Practical Challenges

Modeling Choices:

- Right model and learning algorithm
  - Worry about computational complexity in training or querying a model
- Hyper-parameters
- Selecting a data set to train from
  - Data can be expensive to collect
  - Different algorithms require different amounts of data

Practical Challenges

Overfitting

- Model matches the training data set well, but does not perform well on independent data

…

Practical Challenges

Overfitting

- Model matches the training data set well, but does not perform well on independent data

- How do we detect this?
- How do we mitigate this?
  - Some algorithms will handle this automatically
  - In some cases, we have to be careful about how we choose our hyperparameters and our training set

Practical Challenges

Comparing models and algorithms

- Measuring performance of a model
- Performance is inherently a random variable
  - Must acknowledge this when we are comparing two models
  - This implies that comparison/selection is an empirical process
  - Also must acknowledge this issue when selecting hyper-parameters

Course Topics

Preliminaries:

- Python
- Jupyter
- Pandas
- Numpy
- Scikit-Learn
- Python best practices

Course Topics

- Classifiers
  - Logistic regression, support vector machines, decision trees
  - Feature importance
- Regression
  - Linear and non-linear
  - Polynomial / kernel regression, support vector regression and decision tree regression
- Decision Trees: individual trees, and ensemble methods (including random forests)

Course Topics

Unsupervised Methods

- Principal component analysis
- Local linear embeddings
- Multidimensional scaling
- ISOmap
- t-SNE
- UMap

- Clustering: K-Means, Mixture Models

Course Topics

Tuning Models

- Detecting and mitigating overfitting
- Choosing hyperparameters
- Comparing algorithm types in a statistically sound way

Course Delivery

This is the in-person version of the class

- Course number: CS 5703-001
- This course has a different schedule and different assignments than the online version of the class, so they are not interchangeable
- Slides will be posted to the main course web site

Computing Environment

- All homework assignments will be done in Python
- We are using a Jupyter Hub for the class
  - Key packages pre-installed
  - Data, code skeletons and assignments will be made available inside of your personal Jupyter environment
  - You are also welcome to work on your local machine or use tools like Google Colab, if you wish

What I am assuming about you…

- Programming background:
  - Experience with object-oriented programming
  - Python is not a necessary prerequisite, but is a bonus

- Statistical Methods:
  - Linear regression
  - Hypothesis testing

Resources

- Course web page:
  `https://symbiotic-computing.org/fagg_html/classes/mlp_live_2024`
- Canvas: grade book, announcements
- Slack: primary discussion platform
- GatherTown and in person: office hours
- Text: Aurélien Géron (2020) **Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow (Concepts, Tools, and Techniques to Build Intelligent Systems)**, 2nd edition, ISBN-13: 978-1492032649, O'Reilly Media
- Web resources: documentation, tutorials, papers (linked from the schedule or announced on Canvas)

Grading

- Homework: 85%
  - 11 assignments: explore different ML methods and data sets
    - Drop lowest of these scores
  - Criteria:
    - Success in solving the problem
    - Cleanliness of the code (yes, we expect documentation)
  - Late policy:
    - Up to 1 day late: 10%; up to 2 days late: 20%
- In-class exercises: 15%
  - Must be in class to receive this credit
  - Drop lowest of these scores
- No final exam or end-of-semester project

Proper Academic Conduct

- Homework assignments are to be your own work
- Do not communicate homework solutions in any form with anyone other than the instructor or TA
- Code available on the Internet or generated by a Large Language Model may be used to help you understand the syntax or semantics of the Python tools that we will be using. However, turning in code that solves the specific homework problems that you did not write yourself is prohibited

- General communication with each other is okay

Keys to Success

- Stay on top of lectures and homework assignments
- Learn to read the API documentation
- Most assignments will not be doable in the day before the deadline. Start early
- The net is filled with lots of advice about how to do things
  - Much of the advice is poor or down-right wrong
  - Even when the advice is correct, you should still be able to write your own code
- Ask plenty of questions

For Next Time

- For today: chapter 1
- Prepare for next time:
  - Start of chapter 2
  - Python for programmers (see link on Canvas)
  - Python tutorials - review as necessary

- Lecture on Thursday: we will get you started on Jupyter, Pandas, and Numpy