

# Computational Systems Biology Deep Learning in the Life Sciences

6.802 20.390 20.490 HST.506

6.874 Area II TQE (AI)

David Gifford  
Lecture 1  
February 4, 2019



Massachusetts  
Institute of  
Technology

<http://mit6874.github.io>

[mit6874.github.io](https://mit6874.github.io)  
[6.874staff@mit.edu](mailto:6.874staff@mit.edu)

Please use Piazza or the staff email for any questions

You should have received the Google Cloud coupon URL  
in your email

# Teaching Staff



**David Gifford**  
**Gifford@mit.edu**



**Manolis Kellis**  
**manoli@mit.edu**



**Tim Truong**  
**ttruong@mit.edu**



**Sachit Saksena**  
**sachit@mit.edu**



**Corban Swain**  
**c\_swain@mit.edu**

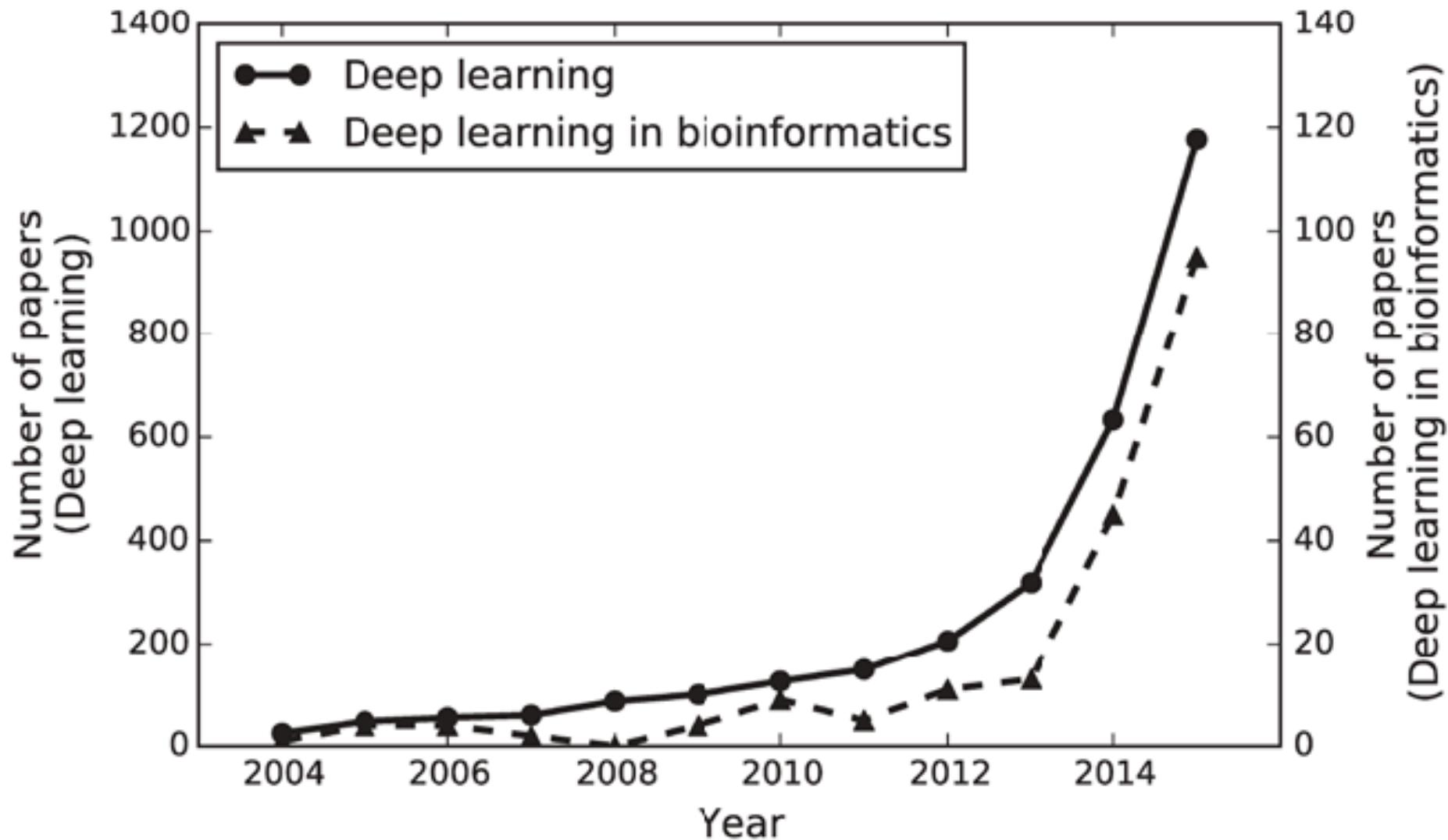
Recitations (this week)

Thursday 4 - 5pm 36-156

Friday 4 - 5pm 36-156

Office hours are after recitation at 5pm in  
same room  
(PS1 help and advice)

Approximately 8% of deep learning publications  
are in bioinformatics



# Welcome to a new approach to life sciences research

- Enabled by the convergence of three things
  - Inexpensive, high-quality, collection of large data sets (sequencing, imaging, etc.)
  - New machine learning methods (including ensemble methods)
  - High-performance Graphics Processing Unit (GPU) machine learning implementations
- Result is completely transformative

# Your background

- Calculus, Linear Algebra
- Probability, Programming
- Introductory Biology

```

def loadstable(ver):
    return _loadversion(ver, prefix="_stable_")

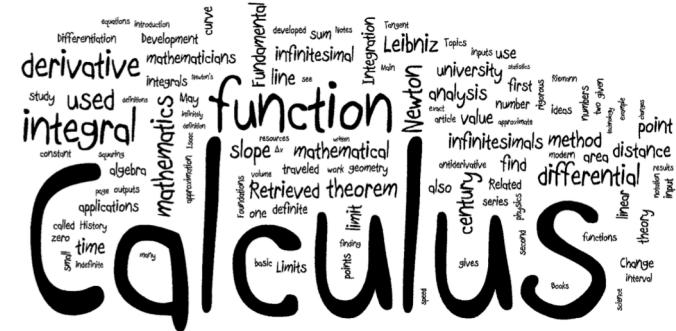
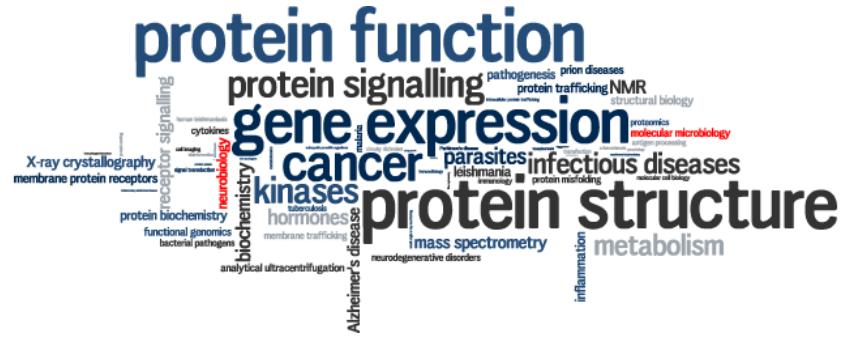
def loadunstable(ver):[...]
def loadexact(ver):[...]
def _loadversion(ver, prefix):
    targetname = prefix + ver.replace('.', '_')
    mainpackage = __original__import__("tools", globals(), locals(),
                                       [targetname])
    global currentversion
    currentversion = getattr(mainpackage, targetname)

# Let users change versions after choosing this one
currentversion.loadstable = loadstable
currentversion.loadunstable = loadunstable
currentversion.loadexact = loadexact

return currentversion

currentversion = None

```



## eigenvalues

Linear Dependence of Vectors  
The vectors  $\vec{x}_1, \vec{x}_2, \vec{x}_3, \dots, \vec{x}_n$  are said to be linearly dependent if there exists scalars  $c_1, c_2, c_3, \dots, c_n$  not all zero such that

$$c_1\vec{x}_1 + c_2\vec{x}_2 + c_3\vec{x}_3 + \dots + c_n\vec{x}_n = \vec{0}$$

If it holds only when  $c_1 = c_2 = c_3 = \dots = c_n = 0$

then the vectors  $\vec{x}_1, \vec{x}_2, \vec{x}_3, \dots, \vec{x}_n$  are called linearly independent.

vectors

## Linear Algebra

determinants

## eigenvectors

Linear Dependence of Vectors  
If  $\vec{x}_1, \vec{x}_2, \vec{x}_3, \dots, \vec{x}_n$  are linearly dependent then one vector can be expressed as a linear combination of the others.

$c_1\vec{x}_1 + c_2\vec{x}_2 + c_3\vec{x}_3 + \dots + c_n\vec{x}_n = \vec{0}$  or the same thing can be written as

$$c_1\vec{x}_1 = -(c_2\vec{x}_2 + c_3\vec{x}_3 + \dots + c_n\vec{x}_n)$$

$$\vec{x}_1 = -\frac{c_2}{c_1}\vec{x}_2 + \frac{c_3}{c_1}\vec{x}_3 + \dots + \frac{c_n}{c_1}\vec{x}_n$$

## Statistics

## Probability

Mean  
Median  
Mode  
Sample  
Population  
Experiment  
Observation  
Independence  
Randomness  
Conditional chance

Standard Deviation  
Skewed  
Histogram  
Pie chart  
Dot plot  
Random  
Conditional  
Chance  
Probability tree  
Intersection  
Box plot  
Observation  
Union  
Normal  
Independence  
Randomness  
Dotplot  
Pie chart  
Conditional  
Chance  
Random  
Conditional  
Chance  
Histogram  
Bar graph  
Pie chart  
Dot plot  
Random  
Conditional  
Chance  
Standard Deviation  
Skewed  
Mean  
Median  
Mode  
Population  
Experiment  
Observation  
Independence  
Randomness  
Conditional chance

# Grade contributions

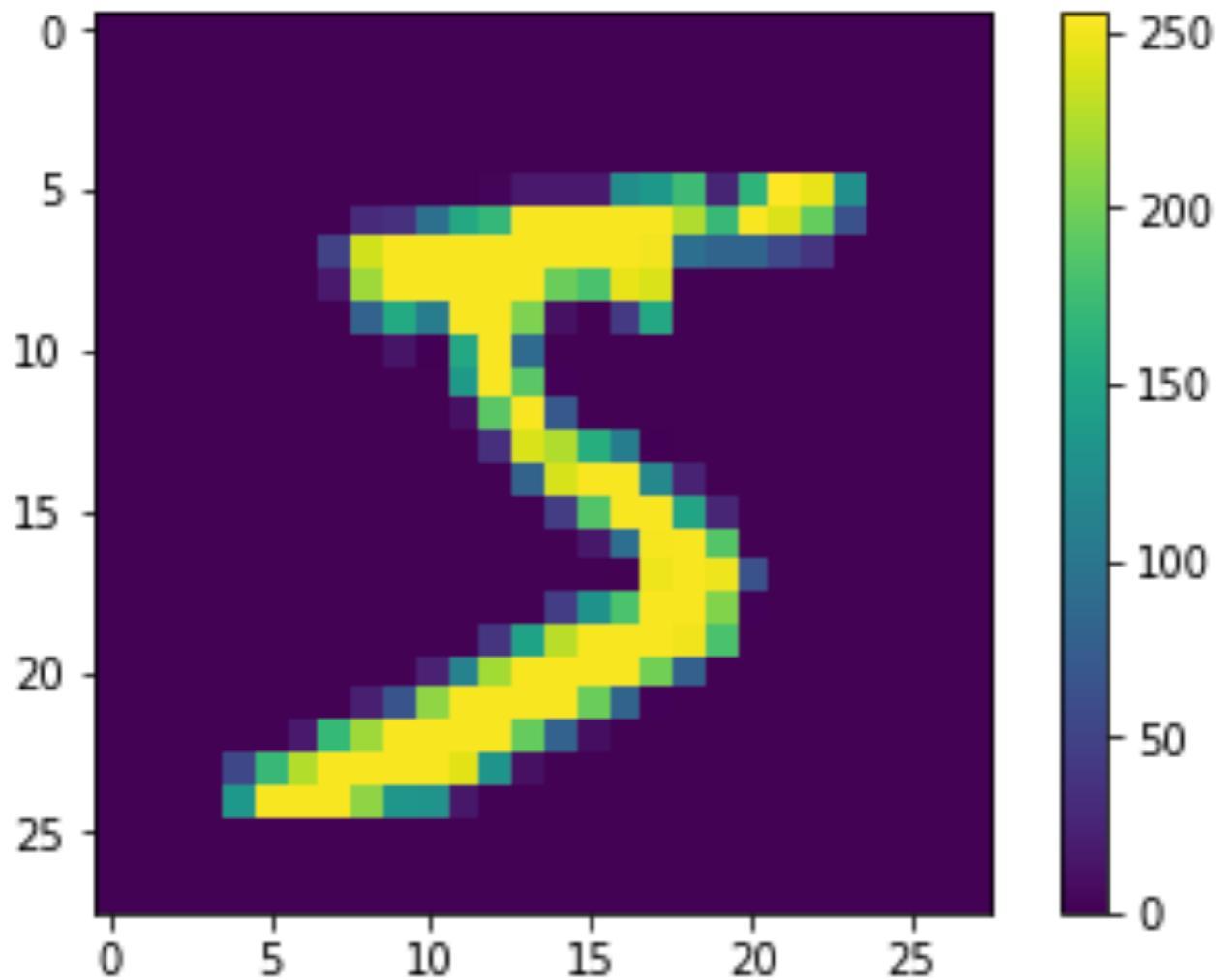
- Four Problem Sets (40%)
  - Individual contribution
  - Done using Google Cloud, Jupyter Notebook
- Two quizzes (1.5 hours), one sheet of notes (30%)
- Final Project (25%)
  - Done in teams of two
- Scribing (5%)

# Alternative MIT subjects

- 6.047 / 6.878 Computational Biology: Genomes, Networks, Evolution
- 6.S897/HST.956: Machine Learning for Healthcare (2:30pm 4-270)
- 8.592 Statistical Physics in Biology
- 7.09 Quantitative and Computational Biology
- 7.32 Systems Biology
- 7.33 Evolutionary Biology: Concepts, Models and Computation
- 7.57 Quantitative Biology for Graduate Students
- 18.417 Introduction to Computational Molecular Biology
- 20.482 Foundations of Algorithms and Computational Techniques in Systems Biology

Psets	Week	Date	Module	Lec/Rec	Description
PS1: Softmax warmup (MNIST) (out: Tue 2/6, due: Fri 2/21)	1	Tuesday, February 4, 2020	Module 1: ML models and interpretation	Lecture 1	Scope of the subject, ML Intro
		Thursday, February 6, 2020		Lecture 2	Learning MLPs
		Friday, February 7, 2020		Recitation 1	ML and Google notebook overview
	2	Tuesday, February 11, 2020		Lecture 3	Model capacity hypothesis space, Neural Networks
		Thursday, February 13, 2020		Lecture 4	Convolutional neural networks, Recurrent neural networks
		Friday, February 14, 2020		Recitation 2	Neural Networks Review
	3	Tuesday, February 18, 2020			(Holiday - President's Day)
		Thursday, February 20, 2020		Lecture 5	ML model interpretation I (SIS) (Brandon Carter Guest Lecture)
		Friday, February 21, 2020		Recitation 3	Interpreting ML models
PS2: TF Binding, ChIP, Motifs (out: Fri 2/21, Due: Fri 3/13)	4	Tuesday, February 25, 2020	Module 2: Chromatin structure / Model selection and uncertainty	Lecture 6	Chromatin accessibility
		Thursday, February 27, 2020		Lecture 7	Protein-DNA interactions and ChIP seq motif discovery
		Friday, February 28, 2020		Recitation 4	Chromatin and gene regulation
	5	Tuesday, March 3, 2020		Lecture 8	Model uncertainty and experiment design
		Thursday, March 5, 2020		Lecture 9	Generative models (gradients, VAEs, GANs)
		Friday, March 6, 2020		Recitation 5	Model uncertainty
	6	Tuesday, March 10, 2020		Lecture 10	Chromatin interactions and 3D genome organization
		Thursday, March 12, 2020		Lecture 11	Dimensionality reduction (PCA, t-SNE, autoencoders)
		Friday, March 13, 2020		Recitation 6	Regulatory element models
PS3: scRNA-seq tSNE analysis (out: Thu 3/12, due Fri 4/3)	7	Tuesday, March 17, 2020	Module 3: Expressed Genome / Dimensionality reduction	Lecture 12	The expressed genome and RNA splicing (RNA-seq)
		Thursday, March 19, 2020		Lecture 13	Quiz 1
		Friday, March 20, 2020		Recitation 7	No recitation
	8	Tuesday, March 24, 2020			(Spring Vacation)
		Thursday, March 26, 2020			
		Friday, March 20, 2020			
	9	Tuesday, March 31, 2020		Lecture 14	scRNA-seq and cell labeling
		Thursday, April 2, 2020		Lecture 15	Manifolds, manifold mapping, word2vec
		Friday, April 3, 2020		Recitation 8	Dimensionality reduction
PS4: Disease, genetics, diagnostics (Out: Fri 4/3, Due: Fri 4/17)	10	Tuesday, April 7, 2020	Module 4: Human Genetics - Genotype -> Phenotype	Lecture 16	Deep learning in Disease Studies and Human Genetics
		Thursday, April 9, 2020		Lecture 17	eQTL prediction and variant prioritization
		Friday, April 10, 2020		Recitation 9	Genetics
	11	Tuesday, April 14, 2020		Lecture 18	STARR-seq and GWAS studies
		Thursday, April 16, 2020		Lecture 19	High-throughput experimentation
		Friday, April 17, 2020		Recitation 10	Protein structure prediction
No other psets	12	Tuesday, April 21, 2020	Module 5: Therapeutics and Diagnostics	Lecture 20	Therapeutic design
		Thursday, April 23, 2020		Lecture 21	Imaging and genotype to phenotype (Guest: Adrian Dalca)
		Friday, April 24, 2020		Recitation 11	
	13	Tuesday, April 28, 2020	Projects	Lecture 22	Quiz 2
		Thursday, April 30, 2020		Lecture 23	How to write, how to present
	14	Tuesday, May 5, 2020		Recitation 12	(Project work)
		Thursday, May 7, 2020		Lecture 24	Project Presentations I
	15	Tuesday, May 12, 2020		Lecture 25	Project Presentations II

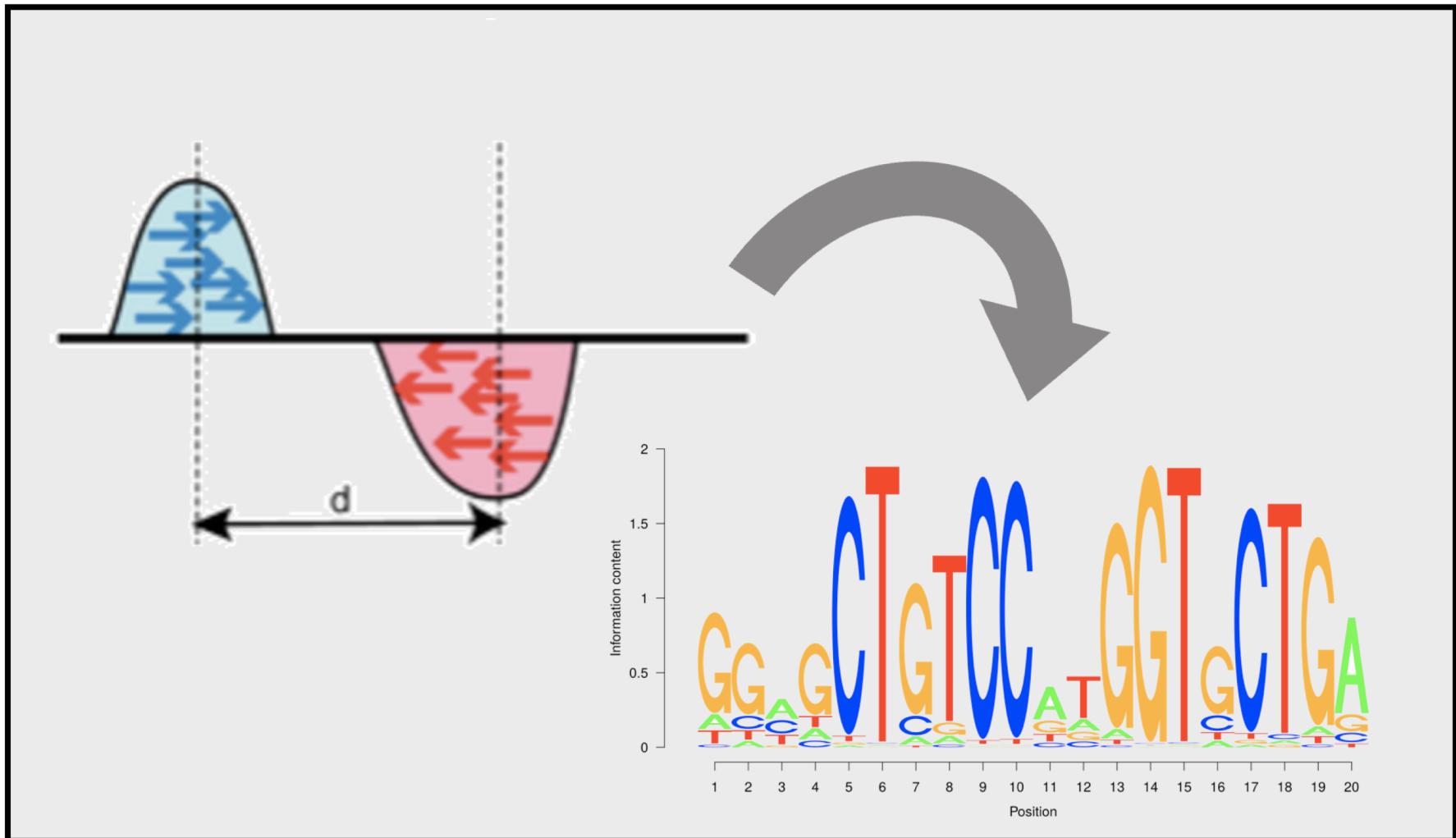
# PS 1: Tensor Flow Warm Up



ground truth: 5

Psets	Week	Date	Module	Lec/Rec	Description
PS1: Softmax warmup (MNIST) (out: Tue 2/6, due: Fri 2/21)	1	Tuesday, February 4, 2020	Module 1: ML models and interpretation	Lecture 1	Scope of the subject, ML Intro
		Thursday, February 6, 2020		Lecture 2	Learning MLPs
		Friday, February 7, 2020		Recitation 1	ML and Google notebook overview
	2	Tuesday, February 11, 2020		Lecture 3	Model capacity hypothesis space, Neural Networks
		Thursday, February 13, 2020		Lecture 4	Convolutional neural networks, Recurrent neural networks
		Friday, February 14, 2020		Recitation 2	Neural Networks Review
	3	Tuesday, February 18, 2020			(Holiday - President's Day)
		Thursday, February 20, 2020		Lecture 5	ML model interpretation I (SIS) (Brandon Carter Guest Lecture)
		Friday, February 21, 2020		Recitation 3	Interpreting ML models
PS2: TF Binding, ChIP, Motifs (out: Fri 2/21, Due: Fri 3/13)	4	Tuesday, February 25, 2020	Module 2: Chromatin structure / Model selection and uncertainty	Lecture 6	Chromatin accessibility
		Thursday, February 27, 2020		Lecture 7	Protein-DNA interactions and ChIP seq motif discovery
		Friday, February 28, 2020		Recitation 4	Chromatin and gene regulation
	5	Tuesday, March 3, 2020		Lecture 8	Model uncertainty and experiment design
		Thursday, March 5, 2020		Lecture 9	Generative models (gradients, VAEs, GANs)
		Friday, March 6, 2020		Recitation 5	Model uncertainty
	6	Tuesday, March 10, 2020		Lecture 10	Chromatin interactions and 3D genome organization
		Thursday, March 12, 2020		Lecture 11	Dimensionality reduction (PCA, t-SNE, autoencoders)
		Friday, March 13, 2020		Recitation 6	Regulatory element models
PS3: scRNA-seq tSNE analysis (out: Thu 3/12, due Fri 4/3)	7	Tuesday, March 17, 2020	Module 3: Expressed Genome / Dimensionality reduction	Lecture 12	The expressed genome and RNA splicing (RNA-seq)
		Thursday, March 19, 2020		Lecture 13	Quiz 1
		Friday, March 20, 2020		Recitation 7	No recitation
	8	Tuesday, March 24, 2020			(Spring Vacation)
		Thursday, March 26, 2020			
		Friday, March 20, 2020			
	9	Tuesday, March 31, 2020		Lecture 14	scRNA-seq and cell labeling
		Thursday, April 2, 2020		Lecture 15	Manifolds, manifold mapping, word2vec
		Friday, April 3, 2020		Recitation 8	Dimensionality reduction
PS4: Disease, genetics, diagnostics (Out: Fri 4/3, Due: Fri 4/17)	10	Tuesday, April 7, 2020	Module 4: Human Genetics - Genotype -> Phenotype	Lecture 16	Deep learning in Disease Studies and Human Genetics
		Thursday, April 9, 2020		Lecture 17	eQTL prediction and variant prioritization
		Friday, April 10, 2020		Recitation 9	Genetics
	11	Tuesday, April 14, 2020		Lecture 18	STARR-seq and GWAS studies
		Thursday, April 16, 2020		Lecture 19	High-throughput experimentation
		Friday, April 17, 2020		Recitation 10	Protein structure prediction
No other psets	12	Tuesday, April 21, 2020	Module 5: Therapeutics and Diagnostics	Lecture 20	Therapeutic design
		Thursday, April 23, 2020		Lecture 21	Imaging and genotype to phenotype (Guest: Adrian Dalca)
		Friday, April 24, 2020		Recitation 11	
	13	Tuesday, April 28, 2020	Projects	Lecture 22	Quiz 2
		Thursday, April 30, 2020		Lecture 23	How to write, how to present
	14	Tuesday, May 5, 2020		Recitation 12	(Project work)
		Thursday, May 7, 2020		Lecture 24	Project Presentations I
	15	Tuesday, May 12, 2020		Lecture 25	Project Presentations II

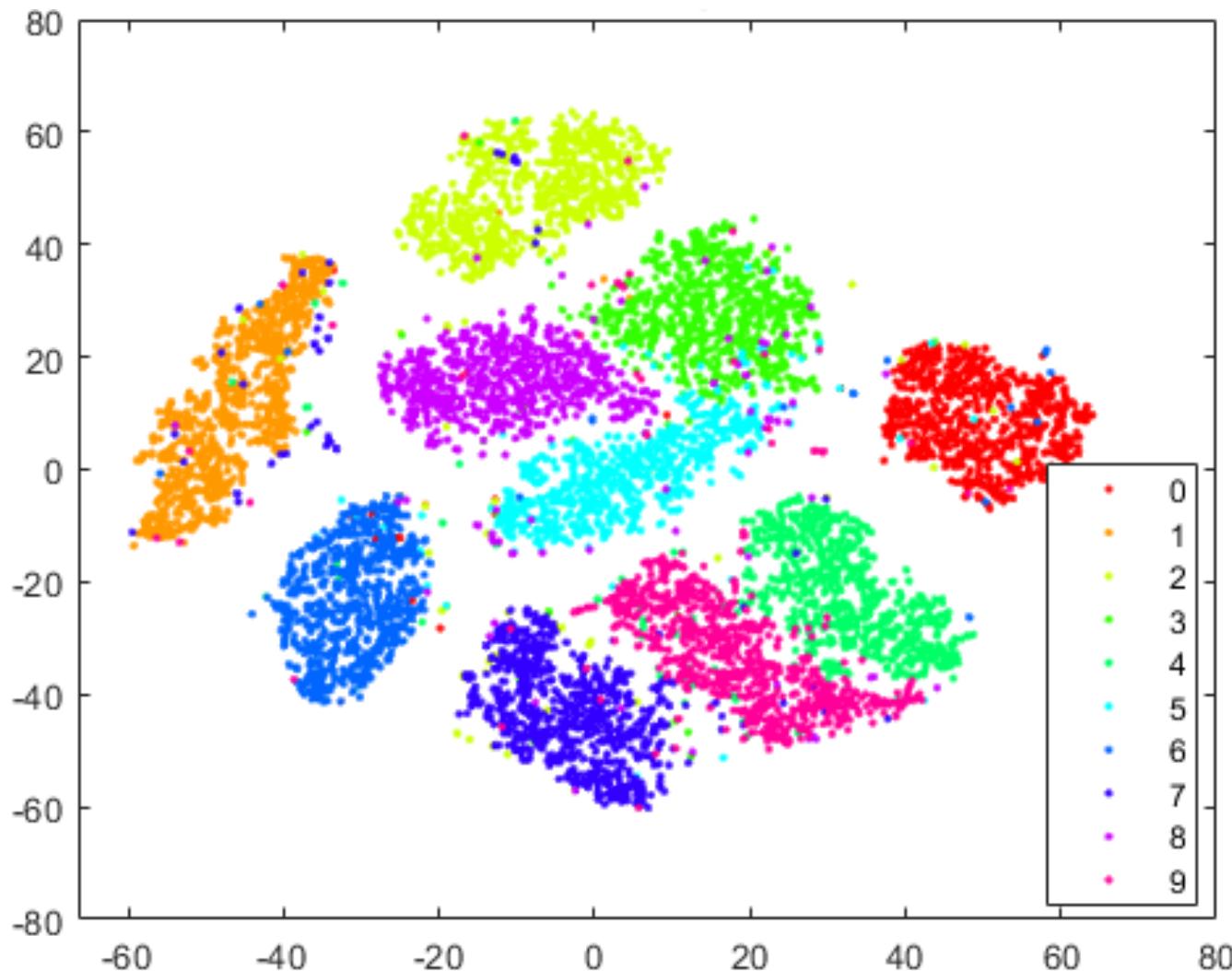
# PS 2: Genomic regulatory codes



Psets	Week	Date	Module	Lec/Rec	Description
PS1: Softmax warmup (MNIST) (out: Tue 2/6, due: Fri 2/21)	1	Tuesday, February 4, 2020	Module 1: ML models and interpretation	Lecture 1	Scope of the subject, ML Intro
		Thursday, February 6, 2020		Lecture 2	Learning MLPs
		Friday, February 7, 2020		Recitation 1	ML and Google notebook overview
	2	Tuesday, February 11, 2020		Lecture 3	Model capacity hypothesis space, Neural Networks
		Thursday, February 13, 2020		Lecture 4	Convolutional neural networks, Recurrent neural networks
		Friday, February 14, 2020		Recitation 2	Neural Networks Review
	3	Tuesday, February 18, 2020			(Holiday - President's Day)
		Thursday, February 20, 2020		Lecture 5	ML model interpretation I (SIS) (Brandon Carter Guest Lecture)
		Friday, February 21, 2020		Recitation 3	Interpreting ML models
PS2: TF Binding, ChIP, Motifs (out: Fri 2/21, Due: Fri 3/13)	4	Tuesday, February 25, 2020	Module 2: Chromatin structure / Model selection and uncertainty	Lecture 6	Chromatin accessibility
		Thursday, February 27, 2020		Lecture 7	Protein-DNA interactions and ChIP seq motif discovery
		Friday, February 28, 2020		Recitation 4	Chromatin and gene regulation
	5	Tuesday, March 3, 2020		Lecture 8	Model uncertainty and experiment design
		Thursday, March 5, 2020		Lecture 9	Generative models (gradients, VAEs, GANs)
		Friday, March 6, 2020		Recitation 5	Model uncertainty
	6	Tuesday, March 10, 2020		Lecture 10	Chromatin interactions and 3D genome organization
		Thursday, March 12, 2020		Lecture 11	Dimensionality reduction (PCA, t-SNE, autoencoders)
		Friday, March 13, 2020		Recitation 6	Regulatory element models
PS3: scRNA-seq tSNE analysis (out: Thu 3/12, due Fri 4/3)	7	Tuesday, March 17, 2020	Module 3: Expressed Genome / Dimensionality reduction	Lecture 12	The expressed genome and RNA splicing (RNA-seq)
		Thursday, March 19, 2020		Lecture 13	Quiz 1
		Friday, March 20, 2020		Recitation 7	No recitation
	8	Tuesday, March 24, 2020			(Spring Vacation)
		Thursday, March 26, 2020			
		Friday, March 20, 2020			
	9	Tuesday, March 31, 2020		Lecture 14	scRNA-seq and cell labeling
		Thursday, April 2, 2020		Lecture 15	Manifolds, manifold mapping, word2vec
		Friday, April 3, 2020		Recitation 8	Dimensionality reduction
PS4: Disease, genetics, diagnostics (Out: Fri 4/3, Due: Fri 4/17)	10	Tuesday, April 7, 2020	Module 4: Human Genetics - Genotype -> Phenotype	Lecture 16	Deep learning in Disease Studies and Human Genetics
		Thursday, April 9, 2020		Lecture 17	eQTL prediction and variant prioritization
		Friday, April 10, 2020		Recitation 9	Genetics
	11	Tuesday, April 14, 2020		Lecture 18	STARR-seq and GWAS studies
		Thursday, April 16, 2020		Lecture 19	High-throughput experimentation
		Friday, April 17, 2020		Recitation 10	Protein structure prediction
No other psets	12	Tuesday, April 21, 2020	Module 5: Therapeutics and Diagnostics	Lecture 20	Therapeutic design
		Thursday, April 23, 2020		Lecture 21	Imaging and genotype to phenotype (Guest: Adrian Dalca)
		Friday, April 24, 2020		Recitation 11	
	13	Tuesday, April 28, 2020	Projects	Lecture 22	Quiz 2
		Thursday, April 30, 2020		Lecture 23	How to write, how to present
	14	Tuesday, May 5, 2020		Recitation 12	(Project work)
		Thursday, May 7, 2020		Lecture 24	Project Presentations I
	15	Tuesday, May 12, 2020		Lecture 25	Project Presentations II

# PS 3: Parametric tSNE

## Single Cell RNA-seq data



Psets	Week	Date	Module	Lec/Rec	Description
PS1: Softmax warmup (MNIST) (out: Tue 2/6, due: Fri 2/21)	1	Tuesday, February 4, 2020	Module 1: ML models and interpretation	Lecture 1	Scope of the subject, ML Intro
		Thursday, February 6, 2020		Lecture 2	Learning MLPs
		Friday, February 7, 2020		Recitation 1	ML and Google notebook overview
	2	Tuesday, February 11, 2020		Lecture 3	Model capacity hypothesis space, Neural Networks
		Thursday, February 13, 2020		Lecture 4	Convolutional neural networks, Recurrent neural networks
		Friday, February 14, 2020		Recitation 2	Neural Networks Review
	3	Tuesday, February 18, 2020			(Holiday - President's Day)
		Thursday, February 20, 2020		Lecture 5	ML model interpretation I (SIS) (Brandon Carter Guest Lecture)
		Friday, February 21, 2020		Recitation 3	Interpreting ML models
PS2: TF Binding, ChIP, Motifs (out: Fri 2/21, Due: Fri 3/13)	4	Tuesday, February 25, 2020	Module 2: Chromatin structure / Model selection and uncertainty	Lecture 6	Chromatin accessibility
		Thursday, February 27, 2020		Lecture 7	Protein-DNA interactions and ChIP seq motif discovery
		Friday, February 28, 2020		Recitation 4	Chromatin and gene regulation
	5	Tuesday, March 3, 2020		Lecture 8	Model uncertainty and experiment design
		Thursday, March 5, 2020		Lecture 9	Generative models (gradients, VAEs, GANs)
		Friday, March 6, 2020		Recitation 5	Model uncertainty
	6	Tuesday, March 10, 2020		Lecture 10	Chromatin interactions and 3D genome organization
		Thursday, March 12, 2020		Lecture 11	Dimensionality reduction (PCA, t-SNE, autoencoders)
		Friday, March 13, 2020		Recitation 6	Regulatory element models
PS3: scRNA-seq tSNE analysis (out: Thu 3/12, due Fri 4/3)	7	Tuesday, March 17, 2020	Module 3: Expressed Genome / Dimensionality reduction	Lecture 12	The expressed genome and RNA splicing (RNA-seq)
		Thursday, March 19, 2020		Lecture 13	Quiz 1
		Friday, March 20, 2020		Recitation 7	No recitation
	8	Tuesday, March 24, 2020			(Spring Vacation)
		Thursday, March 26, 2020			
		Friday, March 20, 2020			
	9	Tuesday, March 31, 2020		Lecture 14	scRNA-seq and cell labeling
		Thursday, April 2, 2020		Lecture 15	Manifolds, manifold mapping, word2vec
		Friday, April 3, 2020		Recitation 8	Dimensionality reduction
PS4: Disease, genetics, diagnostics (Out: Fri 4/3, Due: Fri 4/17)	10	Tuesday, April 7, 2020	Module 4: Human Genetics - Genotype -> Phenotype	Lecture 16	Deep learning in Disease Studies and Human Genetics
		Thursday, April 9, 2020		Lecture 17	eQTL prediction and variant prioritization
		Friday, April 10, 2020		Recitation 9	Genetics
	11	Tuesday, April 14, 2020		Lecture 18	STARR-seq and GWAS studies
		Thursday, April 16, 2020		Lecture 19	High-throughput experimentation
		Friday, April 17, 2020		Recitation 10	Protein structure prediction
No other psets	12	Tuesday, April 21, 2020	Module 5: Therapeutics and Diagnostics	Lecture 20	Therapeutic design
		Thursday, April 23, 2020		Lecture 21	Imaging and genotype to phenotype (Guest: Adrian Dalca)
		Friday, April 24, 2020		Recitation 11	
	13	Tuesday, April 28, 2020	Projects	Lecture 22	Quiz 2
		Thursday, April 30, 2020		Lecture 23	How to write, how to present
	14	Tuesday, May 5, 2020		Recitation 12	(Project work)
		Thursday, May 7, 2020		Lecture 24	Project Presentations I
	15	Tuesday, May 12, 2020		Lecture 25	Project Presentations II

# Your programming environment

## Problem 2

In this problem, we wish to use CNN to learn the motif of CTCF from sequences with similar di-nucleotide frequency. The positive samples are 101bp sequences centered at CTCF ChIP-seq peaks from GM12878 cell line. The negative sequences are generated by permuting the nucleotides in the positive sequences while keeping the di-nucleotide frequency.

We will provide functions for loading data, training and testing. You will:

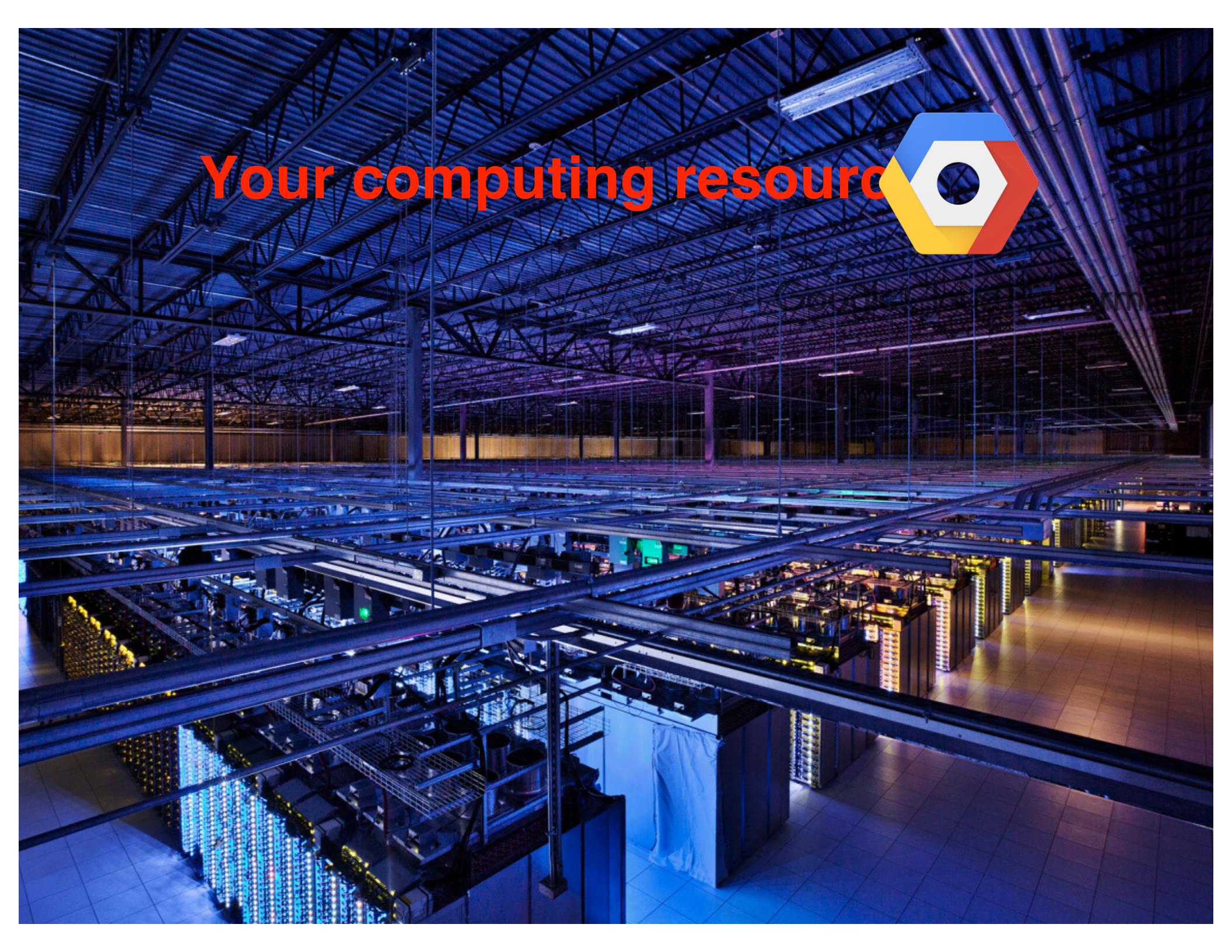
- implement a CNN model with given specifications
- specify the initialization of parameters in the model
- train the model and evaluate on the test set

All the places where you need to fill in begins with "TODO" and ends with "END OF YOUR CODE".

```
In [1]: import tensorflow as tf, sys, numpy as np, h5py
from os.path import join, dirname, basename, exists, realpath
from os import makedirs
from tensorflow.examples.tutorials.mnist import input_data
from sklearn.metrics import roc_auc_score
```

```
In [2]: data_folder = '../data/motif_disc'
batch_size = 128
valid_size = 2000
epochs = 20
best_model_file = join('../output', basename(data_folder), 'best_model.ckpt')
if not exists(dirname(best_model_file)):
    makedirs(dirname(best_model_file))
```

```
In [3]: # Function to load the data embedded in the previous problem and their labels
def load_data(mydir):
    train = h5py.File(join(mydir, 'train.h5'), 'r')
```



# Your computing resource



# Cloud Platform Education Grants

Use credits provided to you via the Google Cloud Platform Education Grants program to access Google Cloud Platform. Get what you need to build and run your apps, websites and services.

Thank you for your interest in Google Cloud Platform Education Grants. Please fill out the form below to receive a coupon code for credit to use on Google Cloud Platform.

**First Name**

**Last Name**

**School Email**

 @mit.edu

If you do not see your domain listed, please contact your course instructor: [gifford@mit.edu](mailto:gifford@mit.edu)

By clicking "Submit" below, you agree that we may share the following information with your educational institution and course instructor ([gifford@mit.edu](mailto:gifford@mit.edu)): (1) personal information that you provide to us on this form and (2) information regarding your use of the coupon and Google Cloud Platform products.

**Submit**

# What is Machine Learning?

[Shalev-Shwartz and Ben-David, 2014]:

*“Learning is the process of converting experience into expertise or knowledge.”*

## What is Machine Learning?

[Shalev-Shwartz and Ben-David, 2014]:

*“Learning is the process of converting experience into expertise or knowledge.”*

[Mohri et al., 2012]:

*“Machine learning can be broadly defined as computational methods using experience to improve performance or to make accurate predictions.”*

## What is Machine Learning?

[Shalev-Shwartz and Ben-David, 2014]:

*“Learning is the process of converting experience into expertise or knowledge.”*

[Mohri et al., 2012]:

*“Machine learning can be broadly defined as computational methods using experience to improve performance or to make accurate predictions.”*

[Murphy, 2012]:

*“The goal of machine learning is to develop methods that can automatically detect patterns in data, and then to use the uncovered patterns to predict future data or other outcomes of interest.”*

## What is Machine Learning?

[Shalev-Shwartz and Ben-David, 2014]:

*“Learning is the process of converting experience into expertise or knowledge.”*

[Mohri et al., 2012]:

*“Machine learning can be broadly defined as computational methods using experience to improve performance or to make accurate predictions.”*

[Murphy, 2012]:

*“The goal of machine learning is to develop methods that can automatically detect patterns in data, and then to use the uncovered patterns to predict future data or other outcomes of interest.”*

[Hastie et al., 2001]:

*“[...] state the learning task as follows: given the value of an input vector  $\mathbf{x}$ , make a good prediction of the output  $\mathbf{y}$ , denoted by  $\hat{\mathbf{y}}$ ”*

## What is Machine Learning?

A computer program is said to learn from  
**experience E**

with respect to some  
**class of tasks T**

and  
**performance measure P,**

if its performance at tasks in T, as measured by P, improves with experience E.

[Mitchell, 1997]

# What is Machine Learning?

A computer program is said to learn from  
**experience E**

with respect to some  
**class of tasks T**

and  
**performance measure P,**

if its performance at tasks in T, as measured by P, improves with experience E.

[Mitchell, 1997]

## Problem Set 1

- experience E: training set of images of handwritten digits with labels (training set)
- task T: classifying handwritten digits within new images (test set)
- performance measure P: percent of test set digits correctly classified in new images (test set)

## What is Machine Learning?

A computer program is said to learn from  
**experience E**

with respect to some  
**class of tasks T**

and  
**performance measure P,**

if its performance at tasks in T, as measured by P, improves with experience E.

[Mitchell, 1997]

# What is Machine Learning?

A computer program is said to learn from  
**experience E**

with respect to some  
**class of tasks T**

and  
**performance measure P,**

if its performance at tasks in T, as measured by P, improves with experience E.

[Mitchell, 1997]

## Problem Set 1

- experience E: training set of images of handwritten digits with labels (training set)
- task T: classifying handwritten digits within new images (test set)
- performance measure P: percent of test set digits correctly classified in new images (test set)

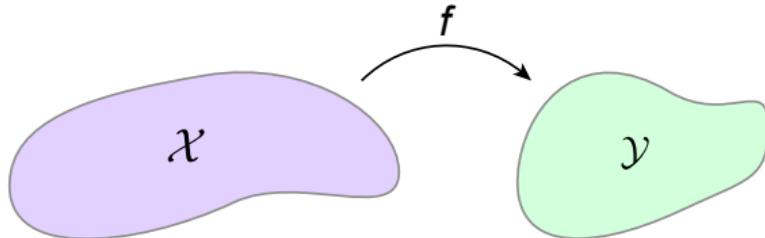
## Notation

$a, b, c_i$	scalar (slanted, lower-case)
$\mathbf{a}, \mathbf{b}, \mathbf{c}$	vector (bold, slanted, lower-case)
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	matrix (bold, slanted, upper-case)
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	tensor (bold, upright, upper-case)
$\mathcal{A}, \mathcal{B}, \mathcal{C}$	set (calligraphic, slanted, upper-case)

## Notation

$a, b, c_i$	scalar (slanted, lower-case)
$\mathbf{a}, \mathbf{b}, \mathbf{c}$	vector (bold, slanted, lower-case)
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	matrix (bold, slanted, upper-case)
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	tensor (bold, upright, upper-case)
$\mathcal{A}, \mathcal{B}, \mathcal{C}$	set (calligraphic, slanted, upper-case)
$\mathcal{X}$	input space or feature space
$\mathbf{X}, \mathbf{x}$	dataset example matrix or tensor
$\mathbf{x}^{(i)}$	$i$ th example of dataset, one row of $\mathbf{X}$
$x_j^{(i)}, x_j$	feature $j$ of example $\mathbf{x}^{(i)}$
$\mathcal{Y}$	label space
$\mathbf{y}^{(i)}$	label of example $i$
$\hat{\mathbf{y}}^{(i)}$	predicted label of example $i$

## Terminology



Input  $\mathbf{X} \in \mathcal{X}$ :

- **features** (in machine learning)
- predictors (in statistics)
- independent variables (in statistics)
- regressors (in regression models)
- input variables
- covariates

Output  $\mathbf{y} \in \mathcal{Y}$ :

- **labels** (in machine learning)
- responses (in statistics)
- dependent variables (in statistics)
- regressand (in regression models)
- target variables

Training set  $S_{\text{training}} = \{(\mathbf{X}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N \in \{\mathcal{X}, \mathcal{Y}\}^N$ , where  $N$  is number of training examples

An example is a collection of features (and an associated label)

Training: use  $S_{\text{training}}$  to learn functional relationship  $f : \mathcal{X} \rightarrow \mathcal{Y}$

# Terminology

$$f : \mathcal{X} \rightarrow \mathcal{Y}$$
$$f(\mathbf{x}; \boldsymbol{\theta}) = \hat{\mathbf{y}}$$

$\theta$ :

- **weights** and **biases** (intercepts)
- coefficients  $\beta$
- parameters

$f$ :

- model
- hypothesis  $h$
- classifier
- predictor
- discriminative models:  $P(Y|X)$
- generative models:  $P(X, Y)$

## Problem Set 1

$$\mathbf{x} \in [0, 1]^{784}$$

$$\hat{\mathbf{y}} \in [0, 1]^{10}$$

$$\mathbf{W} \in \mathbb{R}^{784 \times 10}$$

$$\mathbf{b} \in \mathbb{R}^{10}$$

$$f(\mathbf{x}; \mathbf{W}, \mathbf{b}) = \phi_{\text{softmax}}(\mathbf{W}^T \mathbf{x} + \mathbf{b})$$

## Problem Set 1

input space:

$$\mathcal{X} = \{0, 1, \dots, 255\}^{28 \times 28}$$

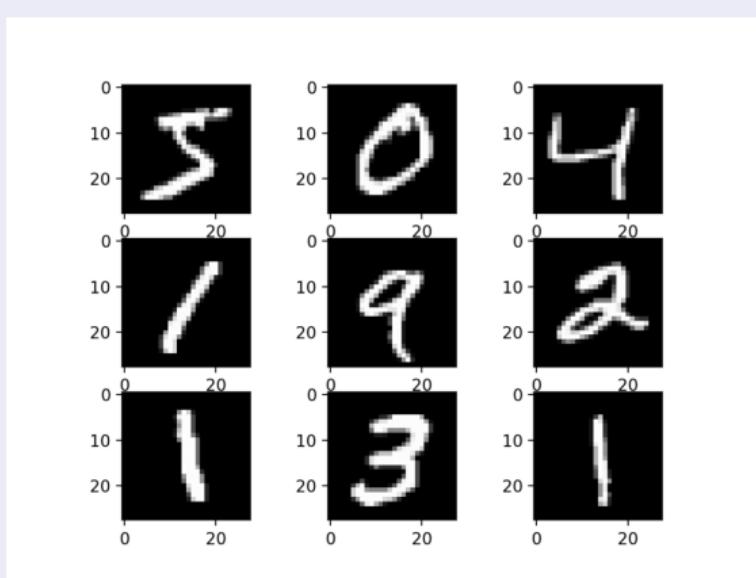
after rescaling:

$$\mathcal{X}' = [0, 1]^{28 \times 28}$$

after flattening:

$$\mathcal{X}'' = [0, 1]^{784}$$

## Classification



# Data in PS1

## Problem Set 1

input space:

$$\mathcal{X} = \{0, 1, \dots, 255\}^{28 \times 28}$$

after rescaling:

$$\mathcal{X}' = [0, 1]^{28 \times 28}$$

after flattening:

$$\mathcal{X}'' = [0, 1]^{784}$$

integer-encoded label space:

$$\mathcal{Y}_i = \{0, 1, \dots, 9\}$$

one-hot-encoded label space:

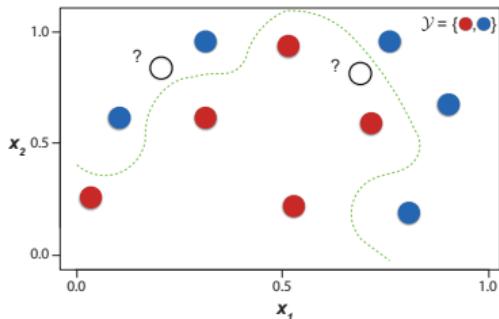
$$\mathcal{Y}_h = [0, 1]^{10}$$

$$\underbrace{\begin{matrix} & & & \\ 1 & 2 & \cdots & 28 \\ & & & \end{matrix}}_{\mathbf{x}^{(i)} \in \mathcal{X}} \quad \begin{matrix} x_{1,1} & x_{1,2} & \cdots & x_{1,28} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,28} \\ \vdots & \vdots & \ddots & \vdots \\ x_{28,1} & x_{28,2} & \cdots & x_{28,28} \end{matrix}$$

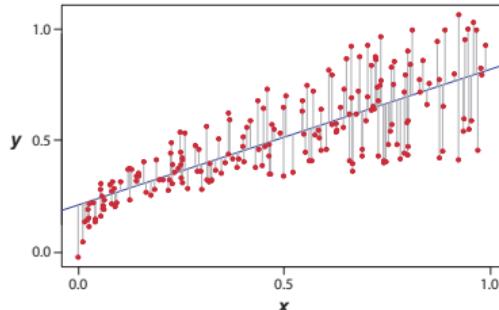
$$\underbrace{\begin{matrix} & & & \\ 1 & 2 & \cdots & 10 \\ & & & \end{matrix}}_{\mathbf{y}^{(i)} \in \mathcal{Y}_h} \quad \begin{bmatrix} y_1 & y_2 & \cdots & y_{10} \end{bmatrix}$$

# Types of Machine Learning

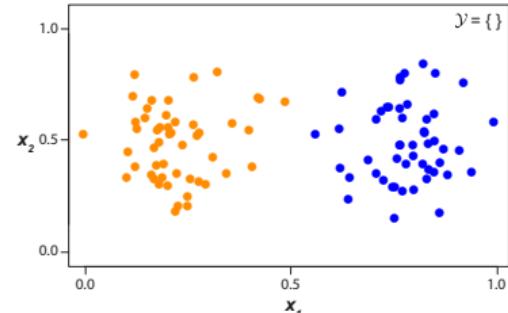
## Classification



## Regression



## Unsupervised learning



$$\mathcal{Y} \neq \emptyset$$

supervised or semi-supervised learning

$$\mathcal{Y} = \mathbb{R}$$

regression

$$\mathcal{Y} = \mathbb{R}^K, K > 1$$

multivariate regression

$$\mathcal{Y} = \{0, 1\}$$

binary classification

$$\mathcal{Y} = \{1, \dots, K\}$$

multi-class classification (integer encoding)

$$\mathcal{Y} = \{0, 1\}^K, K > 1$$

multi-label classification

$$\mathcal{Y} = \emptyset$$

unsupervised learning

# Types of Machine Learning

## Problem Set 1

- task: every  $\mathbf{X}$  has an associated  $\mathbf{y} \Rightarrow$  supervised learning
- subtask:  $\mathcal{Y} = \{0, \dots, 9\} \Rightarrow$  multi-class classification
- method: we use softmax regression (also known as multinomial logistic regression) as multi-class classification method

# Objective functions

An **objective function**  $\mathcal{J}(\Theta)$  is the function that you optimize when training machine learning models. It is usually in the form of (but not limited to) one or combinations of the following:

## Loss / cost / error function $\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y})$ :

### Classification

- 0-1 loss
- cross-entropy loss
- hinge loss

### Regression

- mean squared error (MSE,  $L_2$  norm)
- mean absolute error (MAE,  $L_1$  norm)
- Huber loss (hybrid between  $L_1$  and  $L_2$  norm)

### Probabilistic inference

- Kullback-Leibler divergence (KL divergence)

## Likelihood function / posterior:

- negative log-likelihood (NLL) in maximum likelihood estimation (MLE)
- posterior in maximum a posteriori estimation (MAP)

## Regularizers and constraints

- $L_1$  regularization  $||\Theta||_1 = \lambda \sum_{i=1}^N |\theta_i|$
- $L_2$  regularization  $||\Theta||_2^2 = \lambda \sum_{i=1}^N \theta_i^2$
- max-norm  $||\Theta||_2^2 \leq c$

## Loss functions for classification

**0-1 loss:**

$$\mathcal{L}_{0-1}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^N \mathbb{1}([\hat{y}^{(i)}] \neq y^{(i)}) = \sum_{i=1}^N \begin{cases} 1, & \text{for } \hat{y}^{(i)} \neq y^{(i)} \\ 0, & \text{for } \hat{y}^{(i)} = y^{(i)} \end{cases}$$

where  $[x]$  is the function that rounds  $x$  to the nearest integer.

## Loss functions for classification

**0-1 loss:**

$$\mathcal{L}_{0-1}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^N \mathbb{1}([\hat{y}^{(i)}] \neq y^{(i)}) = \sum_{i=1}^N \begin{cases} 1, & \text{for } \hat{y}^{(i)} \neq y^{(i)} \\ 0, & \text{for } \hat{y}^{(i)} = y^{(i)} \end{cases}$$

where  $[x]$  is the function that rounds  $x$  to the nearest integer.

**Binary cross-entropy loss** (for binary classification):

$\mathcal{L}_{\text{BCE}} = \text{NLL}$  (Negative Log Likelihood)

Likelihood is defined using the Bernoulli distribution

$$p(\hat{y}^{(i)}, y^{(i)}) = (\hat{y}^{(i)})^{y^{(i)}} (1 - \hat{y}^{(i)})^{(1-y^{(i)})}$$

## Loss functions for classification

**0-1 loss:**

$$\mathcal{L}_{0-1}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^N \mathbb{1}([\hat{y}^{(i)}] \neq y^{(i)}) = \sum_{i=1}^N \begin{cases} 1, & \text{for } \hat{y}^{(i)} \neq y^{(i)} \\ 0, & \text{for } \hat{y}^{(i)} = y^{(i)} \end{cases}$$

where  $[x]$  is the function that rounds  $x$  to the nearest integer.

**Binary cross-entropy loss** (for binary classification):

$\mathcal{L}_{\text{BCE}} = \text{NLL}$  (Negative Log Likelihood)

Likelihood is defined using the Bernoulli distribution

$$p(\hat{y}^{(i)}, y^{(i)}) = (\hat{y}^{(i)})^{y^{(i)}} (1 - \hat{y}^{(i)})^{(1-y^{(i)})}$$

$$\begin{aligned} \mathcal{L}_{\text{BCE}}(\hat{\mathbf{y}}, \mathbf{y}) &= \sum_{i=1}^N -y^{(i)} \log(\hat{y}^{(i)}) - (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \\ &= \sum_{i=1}^N \begin{cases} -\log(\hat{y}^{(i)}), & \text{for } y^{(i)} = 1 \\ -\log(1 - \hat{y}^{(i)}), & \text{for } y^{(i)} = 0 \end{cases} \end{aligned}$$

## Loss functions for classification

**Binary cross-entropy loss** (for binary classification):

$$\begin{aligned}\mathcal{L}_{\text{BCE}}(\hat{\mathbf{y}}, \mathbf{y}) &= \sum_{i=1}^N -y^{(i)} \log(\hat{y}^{(i)}) - (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \\ &= \sum_{i=1}^N \begin{cases} -\log(\hat{y}^{(i)}), & \text{for } y^{(i)} = 1 \\ -\log(1 - \hat{y}^{(i)}), & \text{for } y^{(i)} = 0 \end{cases}\end{aligned}$$

$\mathbf{y}$	$\hat{\mathbf{y}}$	$[\hat{\mathbf{y}}]$	$\mathcal{L}_{0-1}(\hat{\mathbf{y}}, \mathbf{y})$	$\mathcal{L}_{\text{BCE}}(\hat{\mathbf{y}}, \mathbf{y})$
[1, 0, 0]	[0.9, 0.2, 0.4]	[1, 0, 0]	0	0.84
[1, 1, 0]	[0.6, 0.4, 0.1]	[1, 0, 0]	1	1.53
[1, 0, 1]	[0.1, 0.7, 0.3]	[0, 1, 0]	3	4.71

## Loss functions for classification

### Problem Set 1

**Categorical cross-entropy loss** (for multi-class classification with  $K$  classes):

$$\mathcal{L}_{\text{CCE}}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^N \sum_{j=1}^K y_j^{(i)} \log(\hat{y}_j^{(i)}),$$

$$\text{where } \hat{y}_j^{(i)} = \frac{\exp(z_j^{(i)})}{\sum_{k=1}^K \exp(z_k^{(i)})} \quad \text{if softmax is used}$$

note:  $y_j^{(i)} = 1$  only if  $\mathbf{x}^{(i)}$  belongs to class  $j$  and otherwise  $y_j^{(i)} = 0$

Probabilistic interpretation:

$\mathcal{L}_{\text{CCE}}$  = NLL, if likelihood is defined using the categorical distribution

## Loss functions for regression

**Mean squared error:**

$$\mathcal{L}_{\text{MSE}}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^N (y^{(i)} - \hat{y}^{(i)})^2$$

Probabilistic interpretation:

$\mathcal{L}_{\text{MSE}}$  = NLL, under the assumption that the noise is normally distributed, with constant mean and variance

## Loss functions for regression

**Mean squared error:**

$$\mathcal{L}_{\text{MSE}}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^N (y^{(i)} - \hat{y}^{(i)})^2$$

Probabilistic interpretation:

$\mathcal{L}_{\text{MSE}}$  = NLL, under the assumption that the noise is normally distributed, with constant mean and variance

**Mean absolute error:**

$$\mathcal{L}_{\text{MAE}}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^N |y^{(i)} - \hat{y}^{(i)}|$$

## Loss functions for regression

**Mean squared error:**

$$\mathcal{L}_{\text{MSE}}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^N (y^{(i)} - \hat{y}^{(i)})^2$$

Probabilistic interpretation:

$\mathcal{L}_{\text{MSE}}$  = NLL, under the assumption that the noise is normally distributed, with constant mean and variance

**Mean absolute error:**

$$\mathcal{L}_{\text{MAE}}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^N |y^{(i)} - \hat{y}^{(i)}|$$

$\mathbf{y}$	$\hat{\mathbf{y}}$	$\mathcal{L}_{\text{MSE}}(\hat{\mathbf{y}}, \mathbf{y})$	$\mathcal{L}_{\text{MAE}}(\hat{\mathbf{y}}, \mathbf{y})$
[3.2, 1.2, 0.3]	[3.1, 1.3, 0.4]	0.01	0.1
[2.1, 0.1, -5.1]	[2.0, -0.1, 1.2]	13.25	2.2
[-0.1, 3.1, 0.5]	[0.1, 3.3, -0.5]	0.36	0.47

## Empirical risk minimization

**Expected risk** (loss) associated with hypothesis  $h(\mathbf{x})$ :

$$\mathcal{R}_{\text{exp}}(h) = \mathbb{E}(\mathcal{L}(h(\mathbf{x}), \mathbf{y})) = \int_{\mathcal{X} \times \mathcal{Y}} \mathcal{L}(h(\mathbf{x}), \mathbf{y}) p(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y}$$

Minimize  $\mathcal{R}_{\text{exp}}(h)$  to find optimal hypothesis  $h$ :

$$h = \operatorname{argmin}_{h \in \mathcal{F}} \mathcal{R}_{\text{exp}}(h)$$

Problem:

- distribution  $p(\mathbf{x}, \mathbf{y})$  unknown
- $\mathcal{F}$  is too large (set of all functions from  $\mathcal{X}$  to  $\mathcal{Y}$ )

## Empirical risk minimization

**Empirical risk** associated with hypothesis  $h(\mathbf{x})$ :

$$\mathcal{R}_{\text{emp}}(h) = \frac{1}{N} \sum_{i=1}^N \mathcal{L}(h(\mathbf{x}^{(i)}), \mathbf{y}^{(i)})$$

Minimize  $\mathcal{R}_{\text{emp}}(h)$  to find  $\hat{h}$ :

$$\hat{h} = \operatorname{argmin}_{h \in \mathcal{H}} \mathcal{R}_{\text{emp}}(h)$$

In practice:

- instead of  $p(\mathbf{x}, \mathbf{y})$ , we use training set  $\mathcal{S}_{\text{training}}$
- instead of  $\mathcal{F}$ , we use  $\mathcal{H} \subset \mathcal{F}$ , e.g., all polynomials of degree 5

# Optimizing objective function

## Gradient descent

- initialize model parameters  $\theta_0, \theta_1, \dots, \theta_m$
- repeat until converge, for all  $\theta_i$

$$\theta_i^t \leftarrow \theta_i^{t-1} - \lambda \frac{\partial}{\partial \theta_i^{t-1}} \mathcal{J}(\Theta),$$

where the objective function  $\mathcal{J}(\Theta)$  is evaluated over all training data  $\{(\mathbf{X}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N$ .

gradient-ds.png

# Training, validation, test sets

## Training set ( $S_{\text{training}}$ ):

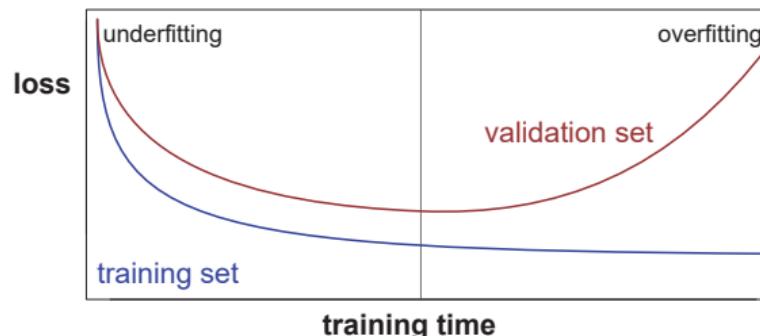
- set of examples used for learning
- usually 60 - 80 % of the data

## Validation set ( $S_{\text{validation}}$ ):

- set of examples used to tune the model hyperparameters
- usually 10 - 20 % of the data

## Test set ( $S_{\text{test}}$ ):

- set of examples used only to assess the performance of fully-trained model
- after assessing test set performance, model must not be tuned further
- usually 10 - 30 % of the data



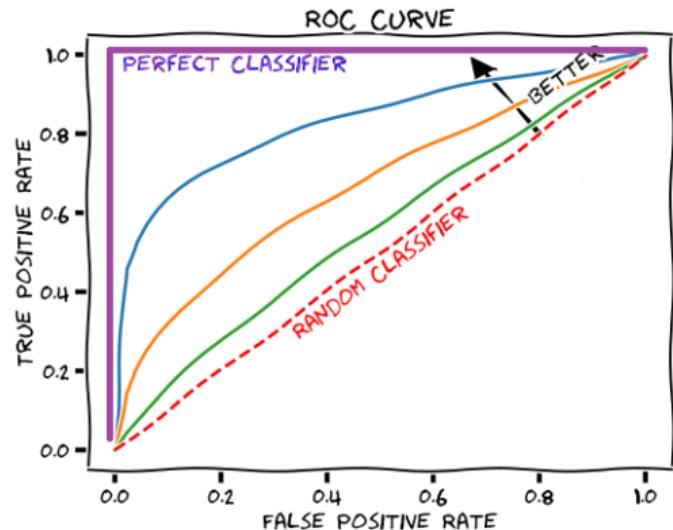
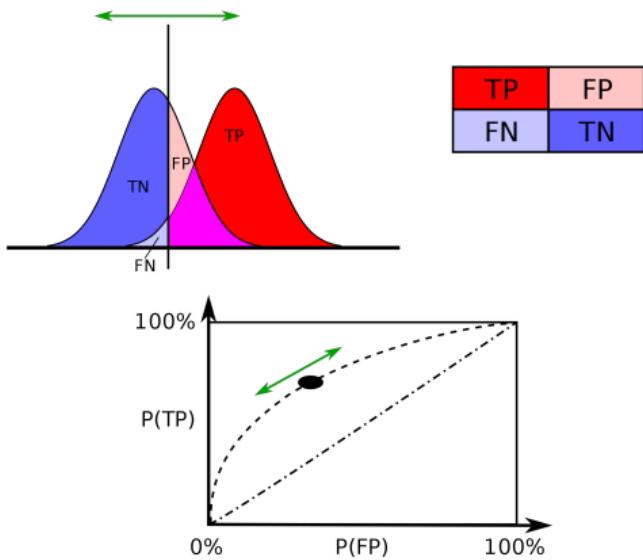
# Confusion matrix and derived metrics

		True condition		Accuracy = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$
Total population		Condition positive	Condition negative	
Predicted condition	Predicted condition positive	True positive, Power	False positive, Type I error	Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$
	Predicted condition negative	False negative, Type II error	True negative	
		Recall, Sensitivity $= \frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	Specificity $= \frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	$F_1$ score = $\frac{1}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}} \cdot 2$

## Problem Set 1

**Accuracy:** proportion of true predictions -  $(TP + TN) / (TP + FP + TN + FN)$

# Receiver Operating Characteristic (ROC) Performance



## Area Under the ROC Curve (AuROC)

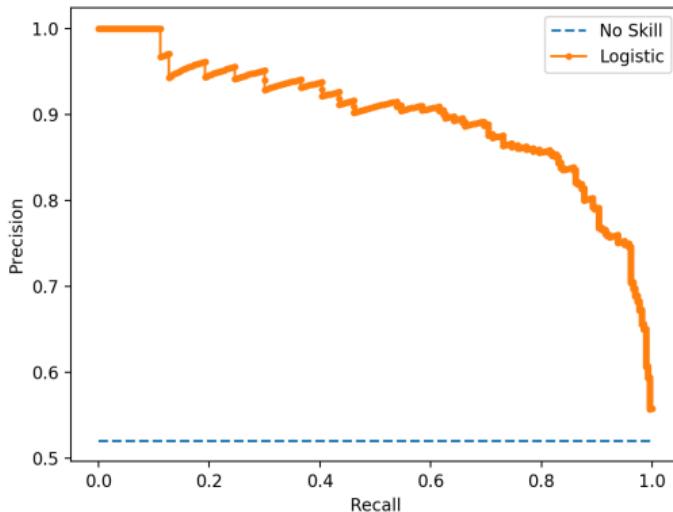
AuROC is a common metric for comparing classification methods

$$TPR = TP / (TP + FN)$$

$$FPR = FP / (FP + TN)$$

**Problematic when we have an unbalanced dataset (example more positives than negatives)**

## Precision Recall Curve (PRC) Performance



### Area Under the PRC (AuPRC)

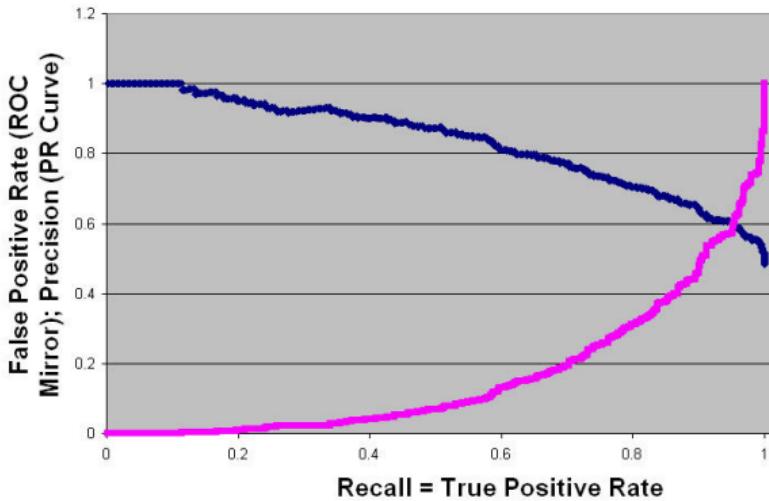
$$\text{Precision} = \text{PPV} = \text{TP} / (\text{TP} + \text{FP}) = 1 - \text{FDR}$$

$$\text{Recall} = \text{TPR} = \text{TP} / (\text{TP} + \text{FN})$$

**Useful when datasets are unbalanced**

## ROC and PRC curves are complementary

Precision Recall Graph (blue) and  
Mirrored ROC Curve (violet)



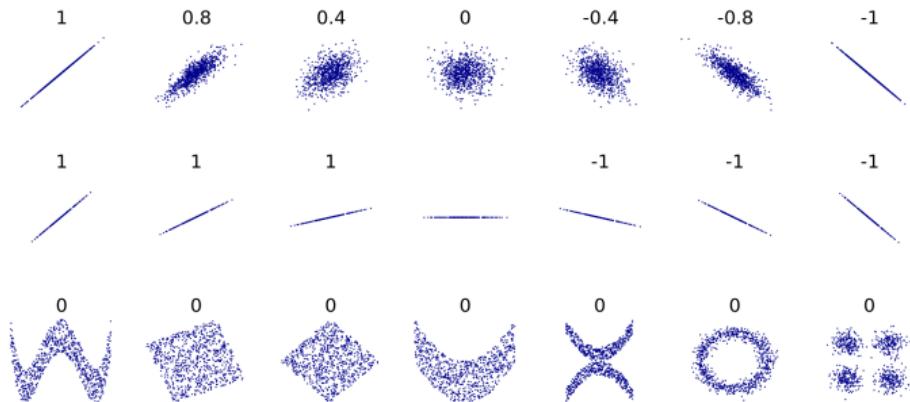
### Recall

$$FPR = FP / (FP + TN)$$

$$\text{Precision} = \text{PPV} = TP / (TP + FP) = 1 - \text{FDR}$$

$$\text{Recall} = \text{TPR} = TP / (TP + FN)$$

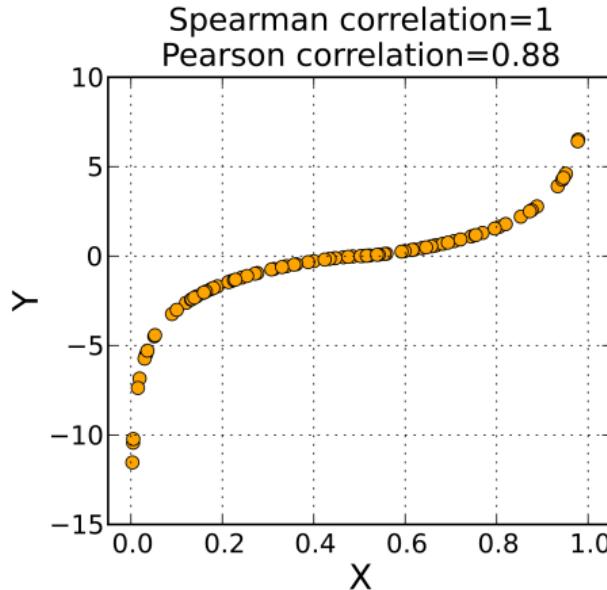
## Regression Metric 1 - Pearson Correlation



Pearson correlation coefficient is  $r$ .  $r^2$  is the fraction of linearly explained variance

$$r = \frac{(x - \bar{x})}{\|x\|} \cdot \frac{(y - \bar{y})}{\|y\|}$$

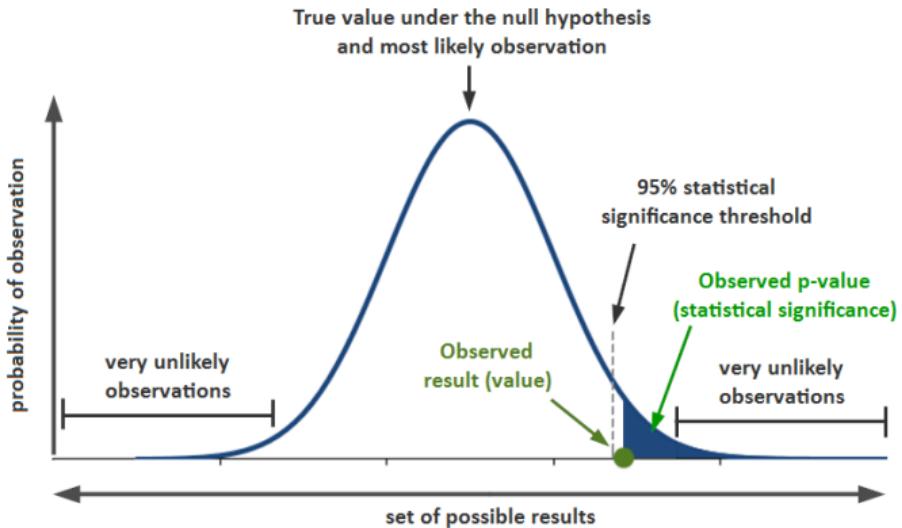
## Regression Metric 2 - Spearman Rank Correlation



Pearson correlation of observation ranks

For ties assign fractional ranks by average rank in ascending order

# Correlation significance tests



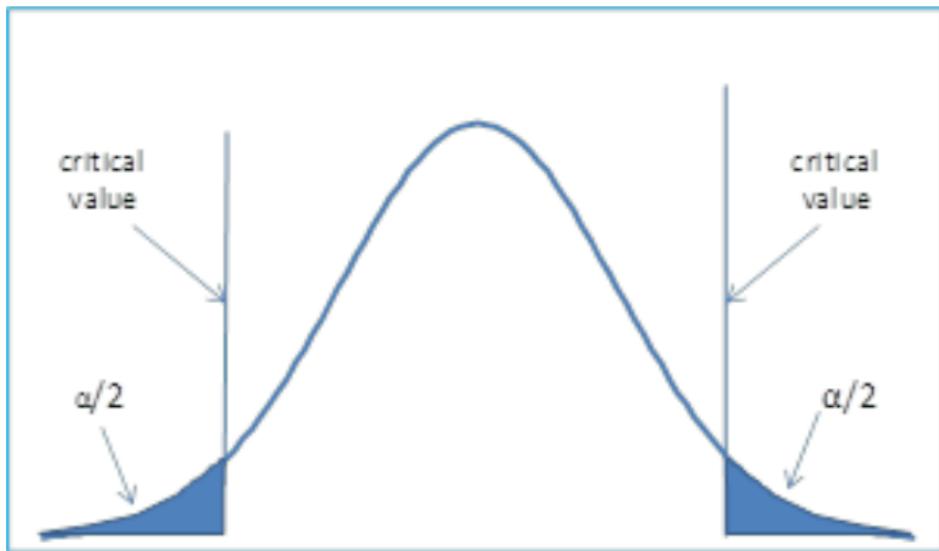
$t$  is distributed as Student's t-distribution with  $n - 2$  degrees of freedom under the null hypothesis

$n$  is the number of observations

$$t = r \sqrt{\frac{n-2}{1-r^2}}$$

Alternatively we can permute values to observe the empirical distribution of null correlations

## One sided vs. two sided test



Two sided tests are used when we are testing for a difference without regard to direction

Two sided tests allocate half the area to each direction

Thus they are more strict if you only wish to test in one direction

## Classifier significance test

### Binomial test for probability that null model would produce observed results

$n$  is the number of observations in test set

$k$  is the number classified correctly test set

$p$  is the probability classifier will make correct choice at random

Probability that exactly  $k$  observations are classified correctly by null

$$Pr(x = k) = \binom{n}{k} p^k (1 - p)^{n-k}$$

## Classifier significance test

### Binomial test for probability that null model would produce observed results

$n$  is the number of observations in test set

$k$  is the number classified correctly test set

$p$  is the probability classifier will make correct choice at random

Probability that exactly  $k$  observations are classified correctly by null

$$Pr(x = k) = \binom{n}{k} p^k (1 - p)^{n-k}$$

Test that  $k$  or greater would have been classified correctly by null

$$p = \sum_{i=k}^n Pr(x = i)$$

This can be approximated by a Chi-squared test

## Multiple hypothesis correction is important

If we ask  $m$  questions we need to adjust our probability that the null is likely

$p_{single}$  Probability one test occurred by chance

## Multiple hypothesis correction is important

If we ask  $m$  questions we need to adjust our probability that the null is likely

$p_{single}$  Probability one test occurred by chance

$p_{corrected} \leq m * p_{single}$  from Boole's inequality results in the Bonferroni correction

## Multiple hypothesis correction is important

If we ask  $m$  questions we need to adjust our probability that the null is likely

$p_{single}$  Probability one test occurred by chance

$p_{corrected} \leq m * p_{single}$  from Boole's inequality results in the Bonferroni correction

$p_{single} \leq \frac{p_{corrected}}{m}$  Filter for significant events

## Multiple hypothesis correction is important

If we ask  $m$  questions we need to adjust our probability that the null is likely

$p_{single}$  Probability one test occurred by chance

$p_{corrected} \leq m * p_{single}$  from Boole's inequality results in the Bonferroni correction

$p_{single} \leq \frac{p_{corrected}}{m}$  Filter for significant events

Benjamini-Hochberg uses a desired false discover rate to provide a relaxed bound

$\alpha$  is our desired false discovery rate (FDR)

$m$  is the number of tests  $H_1 \dots H_m$

$P_1 \dots P_m$  are their p-values in ascending order

- Find the largest  $k$  such that  $P_k \leq \frac{k}{m}\alpha$
- Reject the null hypothesis for all  $H_i$  for  $i = 1, \dots, k$

## Multiple hypothesis correction is important

If we ask  $m$  questions we need to adjust our probability that the null is likely

$p_{single}$  Probability one test occurred by chance

$p_{corrected} \leq m * p_{single}$  from Boole's inequality results in the Bonferroni correction

$p_{single} \leq \frac{p_{corrected}}{m}$  Filter for significant events

Benjamini-Hochberg uses a desired false discover rate to provide a relaxed bound

$\alpha$  is our desired false discovery rate (FDR)

$m$  is the number of tests  $H_1 \dots H_m$

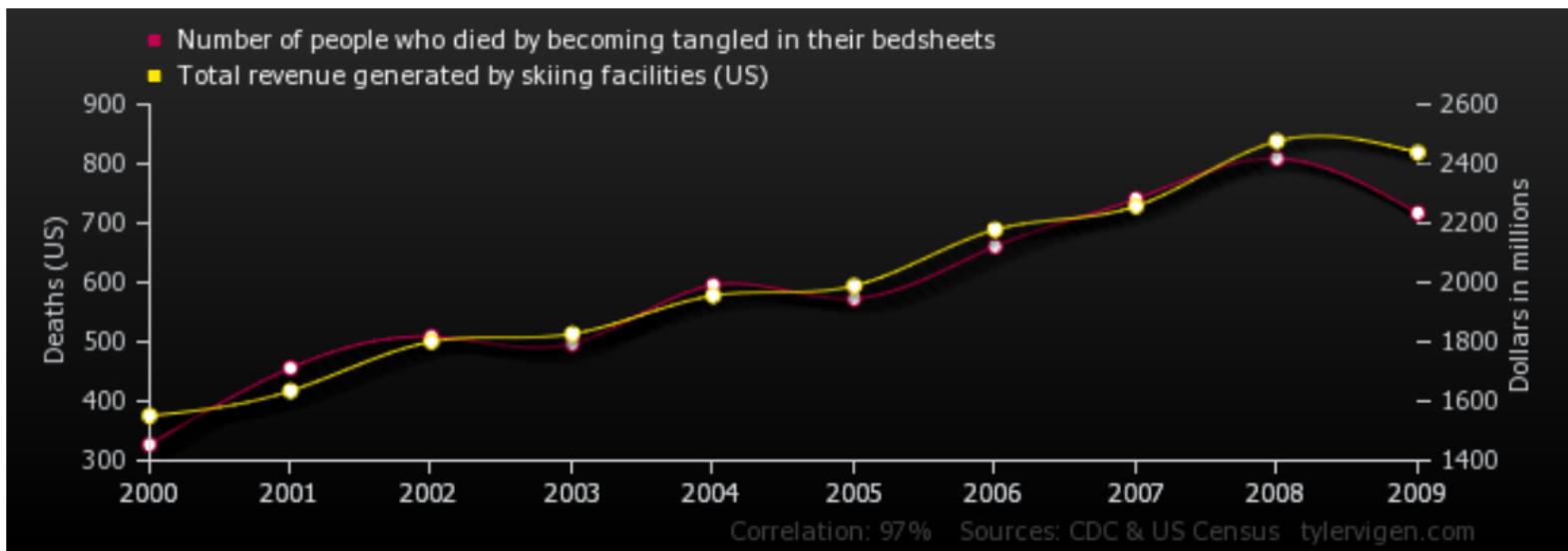
$P_1 \dots P_m$  are their p-values in ascending order

- Find the largest  $k$  such that  $P_k \leq \frac{k}{m}\alpha$
- Reject the null hypothesis for all  $H_i$  for  $i = 1, \dots, k$

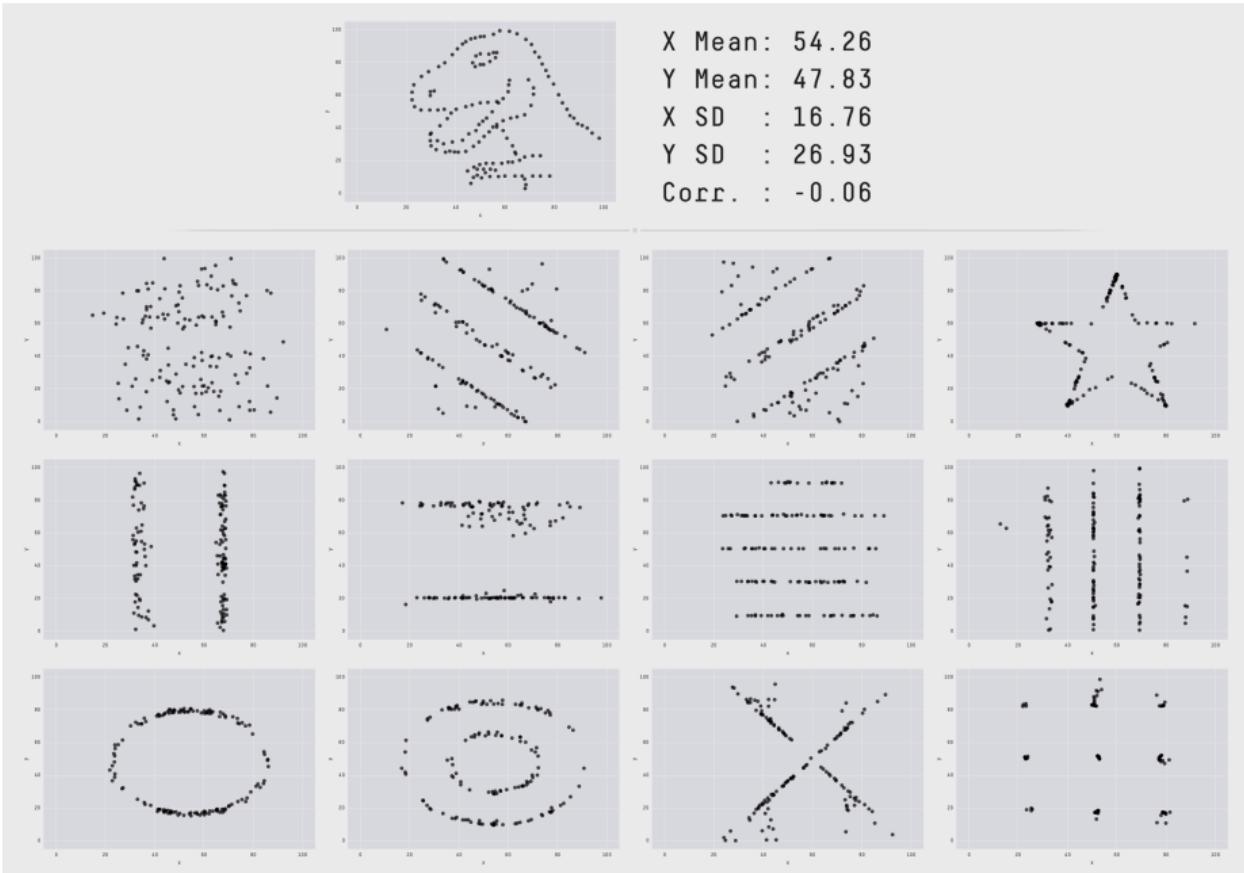
Which transcription factors  $TF_1 \dots TF_5$  bind with a corrected significance of .05?

Single test p-values are 0.003, 0.006, 0.020, 0.045, 0.600

## Correlation is not causation



# The Datasaurus Dozen - J. Matejka, G. Fitzmaurice



# Quo vadis, 6.874?

- neural networks (NNs)
  - convolutional neural networks (CNNs)
  - recurrent neural networks (RNNs)
  - residual neural networks
  - (variational) autoencoders (VAEs)
  - generative adversarial networks (GANs)
- regularization
  - $L_1$  regularization
  - $L_2$  regularization
  - dropout
  - early stopping
- model selection
  - cross-validation (CV)
  - Akaike information criterion (AIC)
  - Bayesian information criterion (BIC)
- model interpretation methods
  - sufficient input subsets (SIS)
  - saliency maps
- model uncertainty
  - identifying out of distribution inputs
  - ensembles and calibrated uncertainty
- dimensionality reduction methods
  - principal component analysis (PCA)
  - t-SNE
  - autoencoders
  - non-negative matrix factorization (NMF)
- hyperparameter optimization and AutoML

## References

-  Hastie, T., Tibshirani, R., and Friedman, J. (2001).  
*The Elements of Statistical Learning.*  
Springer, New York, NY, USA.
-  Mitchell, T. M. (1997).  
*Machine Learning.*  
McGraw-Hill, Inc., New York, NY, USA.
-  Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2012).  
*Foundations of Machine Learning.*  
MIT Press, Cambridge, MA, USA.
-  Murphy, K. P. (2012).  
*Machine learning : a probabilistic perspective.*  
MIT Press, Cambridge, MA, USA.
-  Shalev-Shwartz, S. and Ben-David, S. (2014).  
*Understanding Machine Learning: From Theory to Algorithms.*  
Cambridge University Press, New York, NY, USA.