Discussion Section - Week 1

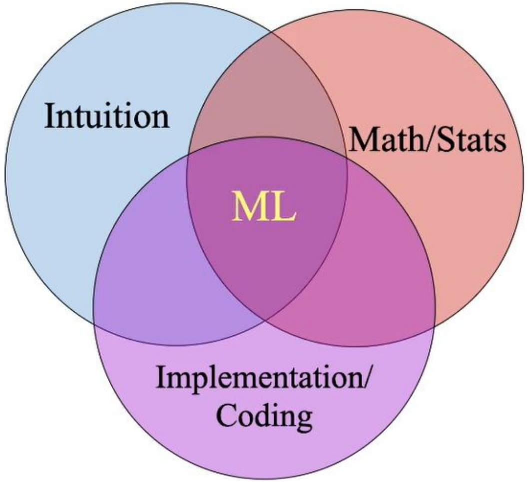
10/07/2020

What we have learnt so far?

1. Machine learning - pick an appropriate model that minimizes a loss function such that it best fits the problem (or data).
2. How to formulate a ML system?

What we have learnt so far?

1. Machine learning - pick an appropriate model that minimizes a loss function such that it best fits the problem (or data).
2. How to formulate a ML system?



\+ Good Data!

What we have learnt so far?

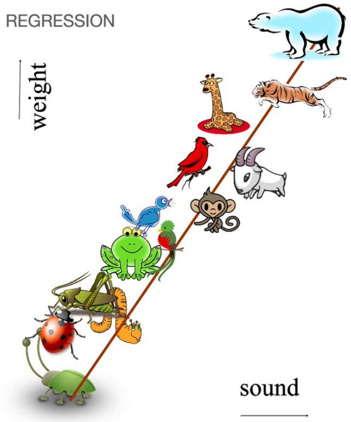
3. Broad classification of ML algorithms
    a. **Supervised - labeled dataset (classification & regression)**
    b. Unsupervised - unlabeled dataset (clustering)

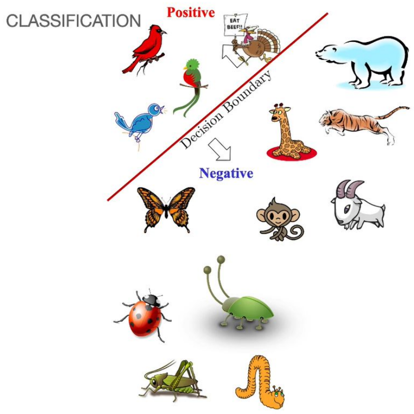
Supervised learning, $\hat{y} =< \mathbf{w}, \mathbf{x} > +b$ such that $\hat{y} \approx y$

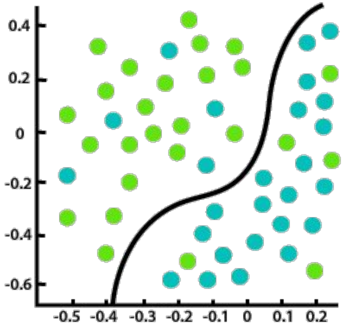
4. How to build a good ML system?
    a. Formulate the problem you want to solve
    b. Understand and process the data
    c. Choose an appropriate ML algorithm & parameters
    d. Train & test the model's generalizability

Step 1 - Formulate the problem

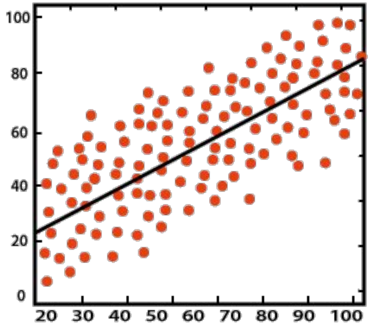
- What do you want to learn from the data?
- Is it a classification or regression problem?







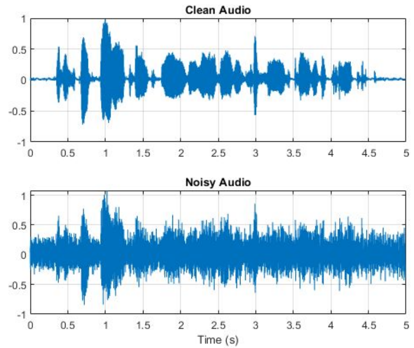
Classification



Regression

Step 2 - Understand and process the data

- What are the features?
- Which features are important?
- Data cleaning - VERY IMPORTANT!
  - Removing datapoints with missing features or out of range values
  - The type of cleaning depends on the data.
    - For example, in case of audio/images it could be background noise suppression.
- Data standardization -
  - Why standardize?
    - When features have different ranges, the models will bias to features with large values.
  - Z-Score normalization (mean is 0 and std. dev is 1)
    - sklearn.preprocessing.StandardScaler()



	Name	Age	Gender	Height	Date
0	lynda	10.0	F	125	5/21/2018
1	tom	NaN	M	135	7/21/2018
2	nick	15.0	F	99	6/21/2018
3	juli	14.0	NaN	120	1/21/2018
4	juli	19.0	NaN	140	10/21/2018
5	juli	18.0	NaN	170	9/21/2018

Step 2 - Understand and process the data

- One-hot encoding - to represent categorical data in a computer readable format.
  - Eg. {“Male”,“Female”}, {“Category 1”, “Category 2”,..... ,”Category N”}
  - “Male” = [1 0]
  - “Female” = [0 1]
  - “Category 2” = [0 1 0 … 0] (N elements)

	Name	Age	Gender	Height	Date
0	lynda	10.0	F	125	5/21/2018
1	tom	NaN	M	135	7/21/2018
2	nick	15.0	F	99	6/21/2018
3	juli	14.0	NaN	120	1/21/2018
4	juli	19.0	NaN	140	10/21/2018
5	juli	18.0	NaN	170	9/21/2018

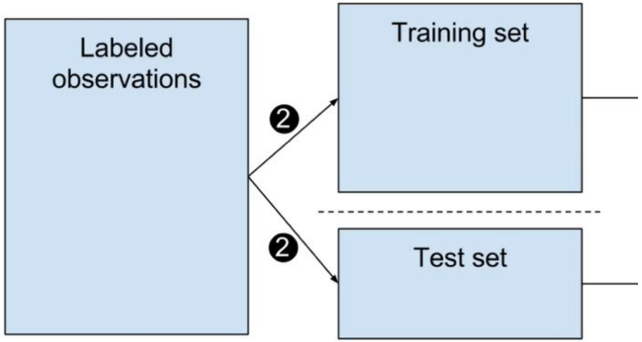
Step 3 - Choose an appropriate ML model

- DNN? SVM?...Plethora of options to choose from!
- How deep should the network be?
- What are the parameters of the model? How to initialize the parameters?
- What is the loss function that we are optimizing?
- ...

Step 5 - Train & test the model’s generalizability

- Any model will eventually perfectly fit to the training data (even if you train it on garbage data!)
- What really matters - the model’s ability to perform well on unseen data.

“GENERALIZABILITY”



Performance on training set

Performance on test set

What we are learning today?

Derivatives with vectors

Supervised learning, $\hat{y} =< \mathbf{w}, \mathbf{x} > +b$ such that $\hat{y} \approx y$

i.e., find the model (w,b) that minimizes the error between the true and estimated labels.

- Easy to compute (w,b) when there are few datapoints.
- But, ML systems are data-driven with very large number of datapoints and feature dimensions.
- Inefficient to use loops :(
- Solution - operate on matrices and vectors :)
- Efficiently minimize the error.

Derivatives with vectors (Numerator layout)

$$\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_m \end{bmatrix}^\mathsf{T} \qquad \mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^\mathsf{T}$$

$$\frac{\partial y}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} & \frac{\partial y}{\partial x_2} & \cdots & \frac{\partial y}{\partial x_n} \end{bmatrix} \qquad \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

"Jacobian formulation"

https://en.wikipedia.org/wiki/Matrix_calculus

Derivatives with vectors (Denominator layout)

$$\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_m \end{bmatrix}^\mathsf{T} \qquad \mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^\mathsf{T}$$

$$\frac{\partial y}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \frac{\partial y_2}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

“Hessian formulation”

https://en.wikipedia.org/wiki/Matrix_calculus

Let’s solve!

Given, **A** is a matrix, **x** and **a** are column vectors.

- $\frac{\partial \mathbf{a}^T \mathbf{x}}{\partial \mathbf{x}} = \frac{\partial \mathbf{x}^T \mathbf{a}}{\partial \mathbf{x}} = \mathbf{a}$,
- $\frac{\partial \mathbf{x}^T \mathbf{A} \mathbf{x}}{\partial \mathbf{x}} = (\mathbf{A} + \mathbf{A}^T)\mathbf{x}$. If $\mathbf{A}$ is symmetric, $\frac{\partial \mathbf{x}^T \mathbf{A} \mathbf{x}}{\partial \mathbf{x}} = 2\mathbf{A}\mathbf{x}$.

1. Why are the partial derivatives of **a’x** and **x’a** equal?
2. Which layout do the above rules adopt?
3. Can you prove the second rule given above?

Let’s solve!

Given, **A** is a matrix, **x** and **a** are column vectors.

- $\frac{\partial \mathbf{a}^T\mathbf{x}}{\partial \mathbf{x}} = \frac{\partial \mathbf{x}^T\mathbf{a}}{\partial \mathbf{x}} = \mathbf{a}$,
- $\frac{\partial \mathbf{x}^T\mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = (\mathbf{A} + \mathbf{A}^T)\mathbf{x}$. If $\mathbf{A}$ is symmetric, $\frac{\partial \mathbf{x}^T\mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = 2\mathbf{A}\mathbf{x}$.

1. Why are the partial derivatives of **a’x** and **x’a** equal?
   Because, **a’x = <a,x> = <x,a> = x’a**
2. Which layout do the above rules adopt?
   Denominator layout - we are differentiating w.r.t. x’.
3. Can you prove the second rule given above?
   Solve using the first rule and chain rule!

Let’s solve!

$f(\mathbf{x}) = \lambda - \mathbf{x}^T(\mathbf{A} + \mathbf{A}^T)\mathbf{x}$ where $\mathbf{A}$ is a symmetric matrix and λ is a constant scalar, derive $\dfrac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$.

Let's solve!

$f(\mathbf{x}) = \lambda - \mathbf{x}^T(\mathbf{A} + \mathbf{A}^T)\mathbf{x}$ where $\mathbf{A}$ is a symmetric matrix and λ is a constant scalar, derive $\dfrac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$.

$f(x) = \lambda - 2x^TAx$ ($\because$ A is symmetric)

$\dfrac{\partial f(x)}{\partial x} = 0 - 2\dfrac{\partial (x^TAx)}{\partial x}$

$= -2(A + A^T)x$ (from rule 2)

$= -4Ax$ ($\because$ A is symmetric)

Let’s solve!

$f(\mathbf{x}) = (\mathbf{a} + \mathbf{x})^T(\mathbf{a} + \lambda\mathbf{x})$ where λ is a constant scalar, derive $\dfrac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$.

Let's solve!

$f(\mathbf{x}) = (\mathbf{a}+\mathbf{x})^T(\mathbf{a}+\lambda\mathbf{x})$ where λ is a constant scalar, derive $\dfrac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$.

$$f(x) = a^Ta + \lambda a^Tx + x^Ta + \lambda x^Tx$$

$$\frac{\partial f(x)}{\partial x} = 0 + \lambda a + a + \underbrace{2\lambda x}_{\text{by chain rule}}$$

$$= (\lambda+1)a + 2\lambda x$$

Coding refresher!

- Get comfortable with Google colab or Anaconda.
- Check out these libraries
  - Numpy
  - Pandas
  - Scikit Learn
  - Matplotlib