CPSC 532P / LING 530A: Deep Learning for Natural Language Processing (DL-NLP)

Muhammad Abdul-Mageed

muhammad.mageed@ubc.ca

Natural Language Processing Lab

The University of British Columbia

Table of Contents

ML Defined, Sort of...

Definitional Thoughts

- A machine learning algorithm is an algorithm that is able to learn from data.
- Mitchell (1997): ‘‘`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."`

The Task, T

On Tasks and Examples/Instances

- Scientifically (and philosophically!), ML is interesting: understanding of the principles that underlie intelligence
- Tasks beyond human capacity, e.g., searching the entire web
- Speech recognition, machine translation (MT), etc.
- We teach a machine how to process an example
- An example is a collection of **features** representing some object or event
- Represented as a vector $\mathbf{x} \in \mathbb{R}^n$ where each entry $\mathbf{x}_i$ of the vector is a feature.

Classification

On Classification

- **Classification:** Labeling an input with one or more of k categories
- **Binary classification:** Picking from $k = 2$
- **Multi-class classification:** $k > 2$
- **Multi-label classification:** Assign $> k$ to a single input
- To learn, an algorithm is asked to **produce a function** $f : \mathbb{R}^n \rightarrow 1, ..., k$
- When $y = f(x)$, the model assigns an input described by vector x to a category identified by **numeric code** y.

Illustrations

- **Binary:** Eng. vs. Russian (langid)
- **Multi-class:** Eng. vs. Russian vs. Swedish,. . . (langid)
- **Multi-label:** Eng. & Russian (code-switching)

Regression

- **Regression**: predict a numerical value given some input
- algorithm is asked to **output a function** $f : \mathbb{R}^n \rightarrow \mathbb{R}$.
- Note: **Format of output is different** from classification

Regression Example

The prediction of the **expected claim amount that an insured person will make** (used to set insurance premiums).

Experience, E

What is E?

- **Supervised ML:** The learner *experiences* an entire **labeled** dataset, to predict label(s)
- **Unsupervised ML**: Learner experiences a dataset containing many features, then learns **useful properties of the structure of this dataset**
- **In deep learning:** Aim to learn the **entire probability distribution that generated a dataset explicitly:** as in density estimation; **implicitly:** tasks like synthesis or denoising
- **Clustering:** dividing the dataset into clusters of similar examples

Word Clustering Example I

Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays Saturdays
June March July April January December October November September August
people guys folks fellows CEOs chaps doubters commies unfortunates blokes
down backwards ashore sideways southward northward overboard aloft downwards adrift
water gas coal liquid acid sand carbon steam shale iron
great big vast sudden mere sheer gigantic lifelong scant colossal
man woman boy girl lawyer doctor guy farmer teacher citizen
American Indian European Japanese German African Catholic Israeli Italian Arab
pressure temperature permeability density porosity stress velocity viscosity gravity tension
mother wife father son husband brother daughter sister boss uncle
machine device controller processor CPU printer spindle subsystem compiler plotter
John George James Bob Robert Paul William Jim David Mike
anyone someone anybody somebody
feet miles pounds degrees inches barrels tons acres meters bytes
director chief professor commissioner commander treasurer founder superintendent dean custodian

Figure: Word clusters (Brown et al., 1992).

Word Clustering Example II

	Binary path	Top words (by frequency)
A1	111010100010	lmao lmfao lmaoo lmaooo hahahahaha lool ctfu rofl loool lmfaoo lmfaooo lmaoooo lmbo **lololol**
A2	111010100011	haha hahaha hehe hahahaha hahah aha hehehe ahaha hah hahahah kk hahaa ahah
A3	111010100100	yes yep yup nope yess yesss yessss ofcourse yeap likewise yepp yesh yw yuup yus
A4	111010100101	yeah yea nah naw yeahh nooo yeh noo noooo yeaa **ikr** nvm yeahhh nahh nooooo
A5	11101011011100	**smh** jk #fail #random #fact smfh #smh #winning #realtalk smdh #dead #justsaying
B	011101011	**u** yu yuh yhu uu yuu yew y0u yuhh youh yhuu iget yoy yooh yuo [illegible] yue juu Ʊ dya youz yyou
C	11100101111001	w fo fa fr fro ov fer **fir** whit abou aft serie fore fah fuh w/her w/that fron isn agains
D	111101011000	facebook **fb** itunes myspace skype ebay tumblr bbm flickr aim msn netflix pandora
E1	0011001	tryna gon finna bouta trynna boutta gne fina gonn tryina fenna qone trynaa qon
E2	0011000	gonna gunna gona gna guna gnna ganna qonna gonnna gana qunna gonne goona
F	0110110111	soo sooo soooo sooooo soooooo sooooooo soooooooo sooooooooo soooooooooo
G1	11101011001010	;) :p :-) xd ;-) ;d (; :3 ;p =p :-p =)) ;] xdd #gno xddd >:) ;-p >:d 8-) ;-d
G2	11101011001011	:) (: =) :)) :] ☺ :') =] ^_^ :))) ^.^ [: ;)) 😊 ((: ^__^ (= ^-^ :))))
G3	1110101100111	:(:/ -_- -.- :-(:'(d: :| :s -__- =(=/ >.< -___- :-/ </3 :\ -____- ;(/: :((>_< =[:[#fml
G4	111010110001	<3 ♥ xoxo <33 xo <333 ❤ ♡ #love s2 <URL-twitition.com> #neversaynever <3333

Figure: Word clusters (Owoputi et al. ACL 2013).

Measure, P

Accuracy / Error Rate

- For **classification**, we measure the accuracy of the model
- **Accuracy**: the proportion of examples for which the model produces the correct output
- Equivalently: measure the error rate
- **error rate**: the proportion of examples for which the model produces an incorrect output
- Error rate: Referred to as the expected 0-1 loss.
- The 0-1 loss on a particular example is 0 if it is correctly classified and 1 if it is not
- For tasks such as **density estimation**: use a metric that gives the model a continuous-valued score: average log-probability the model assigns to some examples

Harmonic Mean

- One type of Pythagorean means.
- Appropriate when we need the **average of rates**
- Expressed as the **reciprocal of the arithmetic mean of the reciprocals of observations**

1: Harmonic Mean

$$(\frac{1^{-1} + 4^{-1} + 4^{-1}}{3})^{-1} = \frac{3}{\frac{1}{1} + \frac{1}{4} + \frac{1}{4}} = \frac{3}{1.5} = 2$$

Precision, Recall, Accuracy & F_1-Score

		Prediction	
		true	false
Gold standard	positive	a	b
	negative	c	d

Eval Metrics

Precision: % of positive classifier predictions that are correct: $a/a+c$
Recall: % of positive cases the classifier caught: $a/a+b$
Accuracy: % of classifier predictions that are correct: $a+d/a+b+c+d$
F_1 **Score**: The harmonic mean of precision and recall: $2 * \frac{prec*rec}{prec+rec}$

Design Matrix of a Dataset

Data Representation Matrix

- Each **row** represents an input data example (a vector)
- Each **column** is a different feature
- **Classification**: each example (row) is assigned a corresponding label.
- **Label**: 0 or 1 (e.g., democrat vs. republican for perspective classification)

Matrix Dimensions

A **dataset of 100 examples and 5 features**: will have a matrix $X \in \mathbb{R}^{100x5}$.

Linear Regression

Linear Regression Set Up

- **Goal:** Build a system that takes **a vector** $\mathbf{x} \in \mathbb{R}^n$ and outputs a scalar $y \in \mathbb{R}$ as its output.
- In **linear regression**, the output is a `linear` function of the input.
- We call y the **gold** value (or **ground truth**) and the **predicted** value $\hat{y}$.
- We define the system as: $\hat{y} = \mathbf{w}^{\mathrm{T}}\mathbf{x}$ where $\mathbf{w} \in \mathbb{R}^n$ is a **vector** of parameters.
- We can think of $\mathbf{w}$ as a set of **weights** that determine how each feature w_i affects the behavior of the system.

Evaluating Model Performance

- **Test set**: $\mathbf{X}^{(test)}$: matrix of **m** examples for evaluating how well the model performs
- **Labels**: $\mathbf{y}^{(test)}$: a vector of regression targets providing the correct value of y for each example in m
- **Predictions**: $\hat{\mathbf{y}}^{(test)}$: predictions of the model

2: Mean Squared Error (MSE)

$$MSE_{test} = \frac{1}{m} \sum_i (\hat{\mathbf{y}}^{(test)} - \mathbf{y}^{(test)})_i^2.$$

Minimizing Error

Improving Model & Bias Term

- To improve, we want to **minimize error** (MSE_{train}) as the model learns from the training data
- We can do this by simply solving where the gradient of MSE_{train} is **0**. (See DL Book, p. 108).
- **Intercept b**: In linear regression, we use one additional parameter-**an intercept** term **b**
- This makes it an **affine function** (one composed of a linear function plus a constant), that allows us to move the line: $\hat{y} = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b$.
- Called **bias** since the output of the transformation is *biased* toward being **b** in the absence of any input
- Practically, this means augmenting **x** with an extra entry that is always equal to 1. (We optimize that value during training)

Train vs. Test Error

Generalization

- **Generalization**: We need to perform well on *new, previously unseen* data
- **Test error**: in addition to reducing **train error**, we want to reduce **test error**
- Making **assumptions about how the training and test are collected**: help us achieve some generalization
- **Statistical learning theory**: assume existence of a **data generating process**
- **i.i.d assumptions:**
  - Assume examples are **independent** from one another.
  - Assume examples are **identically distributed**, drawn from the same probability distribution as one another.

Model Capacity, Overfitting & Underfitting

- **Goals**: 1) Decrease training error & 2) decrease gap between training and test errors.
- **Underfitting:** model is not able to obtain a 'sufficiently low' error value on training set
- **Overfitting** gap between training error and test error is 'too large'
- **Model capacity**: ability of the model to fit a wide variety of functions (from a given **hypothesis space**). Can be **changed** by, e.g.:
  - changing its number of input features
  - adding new parameters associated with those features (e.g., use a two degree polynomial [quadratic function] of input features, instead of a one degree polynomial [$x^2 vs. x$]).
  - Using non-linear functions

Regularization & Weight Decay

- **Regularization**: a modification to a learning algorithm to **reduce its generalization error but not its training error**
- **Weight decay**: we can modify the training criterion for linear regression to include weight decay.
- Below, λ is a value chosen ahead of time that controls the strength of our preference for **smaller weights**

3: Linear Regression With Weight Decay

$$J(\mathbf{w}) = MSE_{train} + \lambda \mathbf{w}^{\mathrm{T}} \mathbf{w}.$$

Weight Decay

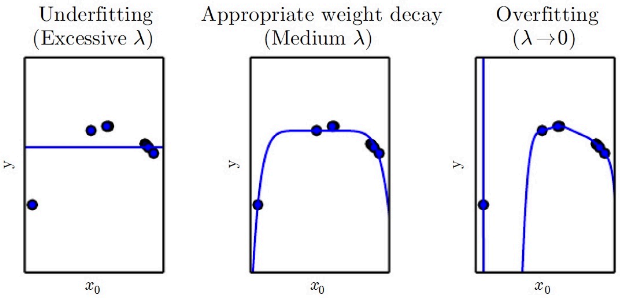

Figure: Linear Regression w/ & w/t Weight Decay. [Goodfellow et al., 2016]

- Multiple other **regularization methods**, some of which particularly developed for deep learning. More on these later . . .

Hyperparameters and Validation Sets

Hyperparameters

- **Hyperparameters**: settings that control the behavior of a learner
- **Examples**: (1) the **degree of the polynomial** used in linear regression and (2) the **value of** λ used to control weight decay
- hyperparameter are chosen on a **validation** set
- If chosen on a **training set**, they will always **default to maximum model capacity**, resulting in **overfitting**.
- We construct the validation set *exclusively* from training data
- As such, we **learn hyperparameters from the train split** and **tune them on the validation split**.

More on Validation Set

Data Splitting & Shuffling

- **Typical splits**: we use 80% of training data as `train` and 20% as `validation` (aka `development` data).
- **Another way**: 80% `train`, 10% `dev` and 10% `test`
- **Data shuffling**: if the problem solution does not depend on sequence of instances (e.g., dialect id or sentiment analysis on tweet level)
- NO **data shuffling**: if sequence is important (e.g., pos tagging, with word-level prediction)
- **Benchmarks**: With benchmarked data, we must use the exact same data splits as others for comparison
- Test data: always **fully blind**. We NEVER see test split.

Design of Data Matrix
(For Your Reference)

Data Matrix Design Illustration 1

Study_hrs	Ling_bg	Math_bg	Attendance	...	Grade
100	6	6	7	...	75
120	7	6	9	...	85
125	7	8	8	...	87
115	8	6	6	...	80
...	...	...	...	...	...

Figure: Grade outcome by student study hours, linguistics background, math background, attendance, etc.

Data Matrix Design Illustration 2

	Study_hrs	Ling_bg	Math_bg	Attendance	...	Grade
	X_1	X_2	X_3	X_4	X_j	y
X^1	100	6	6	7	...	75
X^2	120	7	6	9	...	85
X^3	125	7	8	8	...	87
X^4	115	8	6	6	...	80
X^n	...	...	...	...	...	...

Figure: Each row is an input point (representing a student), and each column is a value of a given **feature**. A row is a **vector**. Last column is the outcome or **gold label** “Y".

Data Matrix Design Illustration 3

	Study_hrs	Ling_bg	Math_bg	Attendance	...	Grade
	X_1	X_2	X_3	X_4	X_j	y
X^1	100	6	6	7	...	75
X^2	120	7	6	9	...	85
X^3	125	7	8	8	...	87
X^4	115	8	6	6	...	80
X^n	...	...	...	...	...	...

Figure: Row 3 is a data point. We can refer to it as a **feature vector** representing a given student.

Data Matrix Design Illustration 4

	Study_hrs	Ling_bg	Math_bg	Attendance	...	Grade
	x_1	x_2	x_3	x_4	x_j	y
x^1	100	6	6	7	...	75
x^2	120	7	6	9	...	85
x^3	125	7	8	8	...	87
x^4	115	8	6	6	...	80
x^n	...	...	...	...	...	...

$$x^3 = \begin{bmatrix} 125 \\ 7 \\ 8 \\ 8 \\ \ldots \end{bmatrix}$$

Figure: Row 3 as **feature vector**. $x^3 \in R^{nx1}$. Note: We can add one entry to x^3 for **bias** and so it will be $\in R^{n+1x1}$.

Data Matrix Design Illustration 5

	Study_hrs	Ling_bg	Math_bg	Attendance	...	Grade
	x_1	x_2	x_3	x_4	x_j	y
x^1	100	6	6	7	...	75
x^2	120	7	6	9	...	85
x^3	125	7	8	8	...	87
x^4	115	8	6	6	...	80
x^n	...	...	...	...	...	...

$$x^3 = \begin{bmatrix} 125 \\ 7 \\ 8 \\ 8 \\ \dots \end{bmatrix}$$

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ \dots \end{bmatrix}$$

Figure: **w** is a vector of free parameters that we hope to learn. We can initiate it randomly, or under some distribution (e.g. Gaussian). We also can add a **bias** to it and so it will be $\in R^{n+1x1}$.

Data Matrix Design Illustration 6

	Study_hrs	Ling_bg	Math_bg	Attendance	...	Grade
	x_1	x_2	x_3	x_4	x_j	y
x^1	100	6	6	7	...	75
x^2	120	7	6	9	...	85
x^3	125	7	8	8	...	87
x^4	115	8	6	6	...	80
x^n	...	...	...	...	...	...

$$x^3 = \boxed{\begin{matrix} 125 \\ 7 \\ 8 \\ 8 \\ \ldots \end{matrix}}$$

$$w = \boxed{\begin{matrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ \ldots \end{matrix}}$$

$$w^T = \boxed{w^1 \; w^2 \; w^3 \; w^4 \; \ldots}$$

Figure: To multiply, we transpose **w** to get w^T. As such, we will have $w^T \in R^{1xn+1}$.

Data Matrix Design Illustration 7

	Study_hrs	Ling_bg	Math_bg	Attendance	...	Grade
	x_1	x_2	x_3	x_4	x_j	y
x^1	100	6	6	7	...	75
x^2	120	7	6	9	...	85
x^3	125	7	8	8	...	87
x^4	115	8	6	6	...	80
x^n	...	...	...	...	...	...

$$X^3 = \begin{bmatrix} 125 \\ 7 \\ 8 \\ 8 \\ \vdots \end{bmatrix} \qquad w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ \vdots \end{bmatrix}$$

$$\boxed{\tilde{y} = w^T x + b}$$

$$w^T = \begin{bmatrix} w^1 & w^2 & w^3 & w^4 & \dots \end{bmatrix}$$

Figure: Now we can predict. The problem is that we do not know the best values of our weights. We can either get these values **analytically** or learn them e.g., via **gradient descent**.