SUMMER SCHOOL 2020

MACHINE LEARNING DAY 2

DEEP LEARNING

Session II: Linear regression

Isaac Ye, HPTC @ York University

Isaac@sharcnet.ca

Session II

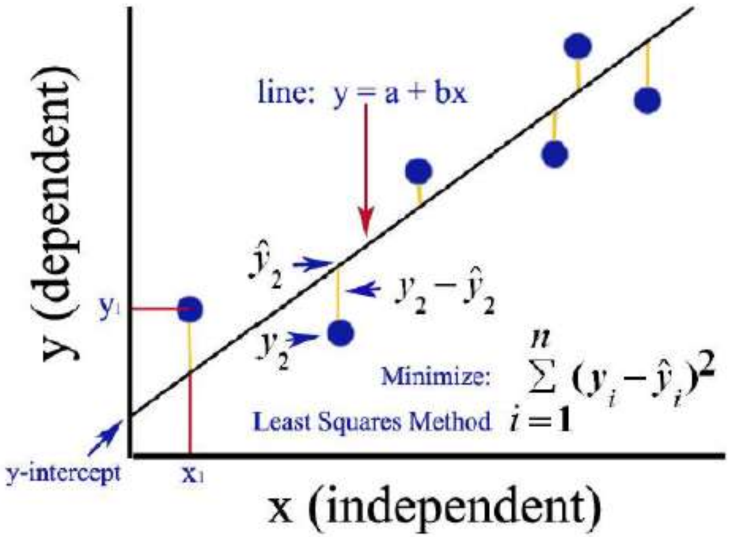
- Linear regression (multi-variables)
- PyTorch model/cost function/optimizer
- *Lab 2A: Multivariable linear regression with PyTorch*
- Running DL in Graham
- *Lab 2B - Working in Graham and running a simple code*

Categories of ML problems

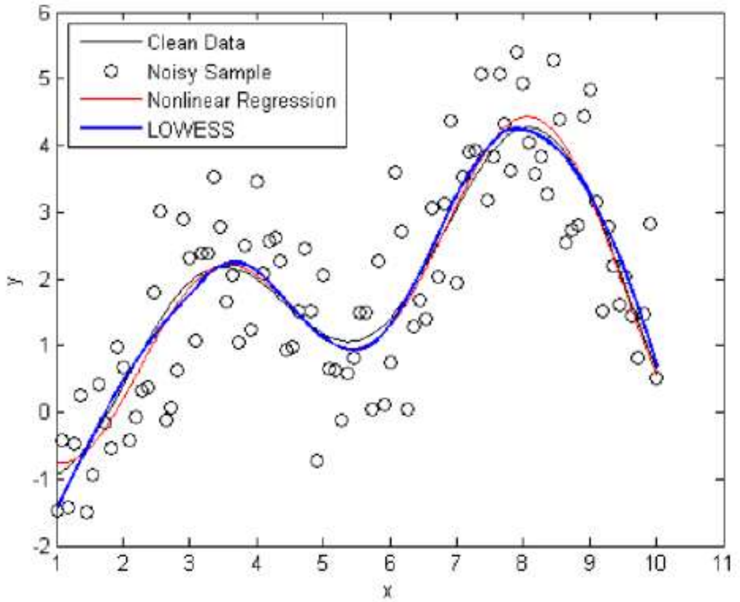
	Supervised	Unsupervised	Reinforcement
Discrete	Classification	Clustering	Action space agent
Continuous	Regression	Dimensionality reduction	Action space agent

Regression problem

Fit the prediction function f(x) to the training data to predict continuous real value



Linear regression



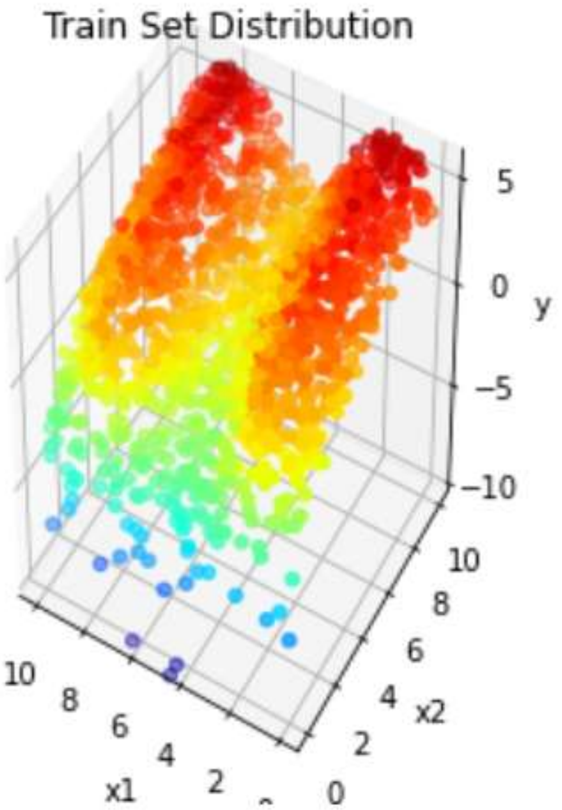
Nonlinear regression

Linear regression: multivariable

Data preparation: Input (x_1,x_2, y)

x_1	x_2	y
3.91870851	2.32626914	0.73817558
2.59194437	6.00656071	4.3940048
6.46991632	3.57514815	0.61488728
:	:	:
4.56486433	2.14296641	3.95964088
1.29483514	1.67730041	3.48018992

```
num_data = 2400
x1 = np.random.rand(num_data) *10
x2 = np.random.rand(num_data) *10
e = np.random.normal(0, 0.5, num_data)
X= np.array([x1,x2]).T  # T for transpose from (2, 2400) to (2400, 2)
y=2*np.sin(x1) + np.log(0.5*x2**2)+e
```



Model (Hypothesis)

$$H(x_1, x_2) = w_1x_1 + w_2x_2 + b$$

For the data with n number of features, it is can be written as

$$H(x_1, x_2, x_3, \ldots, x_n) = w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n + b$$

Expression in matrix

$$H(x_{i1}, x_{i2}) = w_1 x_{i1} + w_2 x_{i2} + b$$

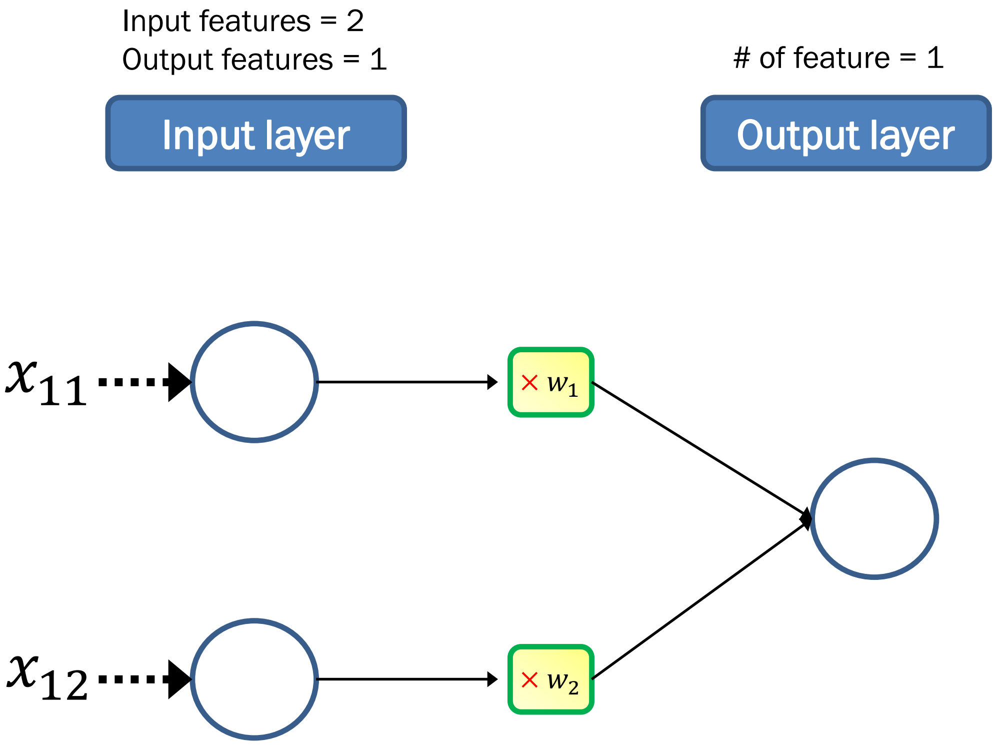
$$\begin{bmatrix} x_1 & x_2 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + b = w_1 x_1 + w_2 x_2 + b$$

$$\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{n1} & x_{n2} \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + b = \begin{bmatrix} w_1 x_{11} + w_2 x_{12} + b \\ w_1 x_{21} + w_2 x_{22} + b \\ \vdots \\ w_1 x_{n1} + w_2 x_{n2} + b \end{bmatrix}$$

$$H(X) = XW + b$$

Layout

$$\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{n1} & x_{n2} \end{bmatrix}$$



$$H(x_{i1}, x_{i2}) = w_1 x_{i1} + w_2 x_{i2}$$

Let's consider a simple

case with W only.

Cost function

$$H(X) = XW + b$$

$$cost = \frac{1}{m}\sum_{i=1}^{m}(H(x_{i1}, x_{i2}) - y_i)^2$$

We want to minimize the cost as well!

Algorithm structure

Data Preparation

Model define

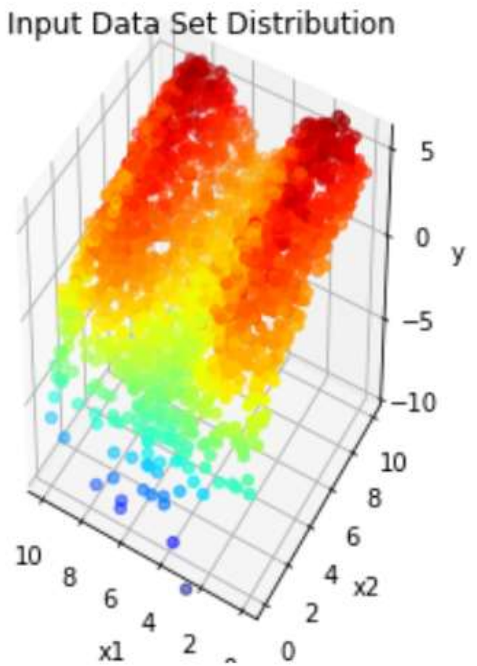
Cost function + optimizer

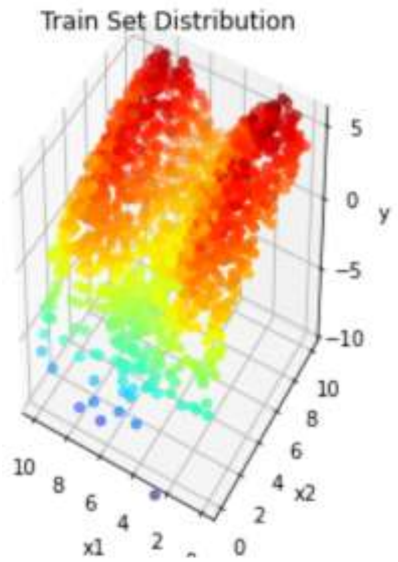
Model Test

Data preparation

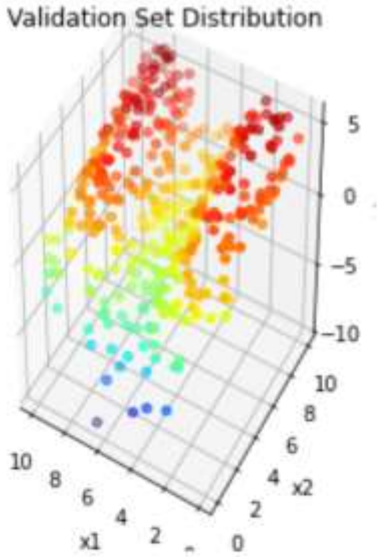
Data Preparation

x_1	x_2	y
3.91870851	2.32626914	0.73817558
2.59194437	6.00656071	4.3940048
6.46991632	3.57514815	0.61488728
:	:	:
4.56486433	2.14296641	3.95964088
1.29483514	1.67730041	3.48018992

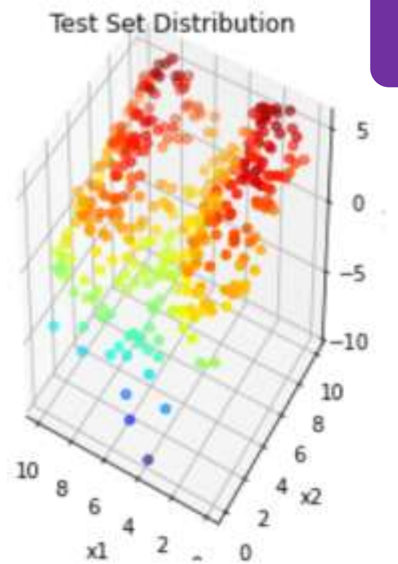




Train set



Validation set



Testing set

In the code

```
train_X, train_y = X[:1600, :], y[:1600]
val_X, val_y = X[1600:2000, :], y[1600:2000]
test_X, test_y = X[2000:, :], y[2000:]
```

Model define: linear regression

Model define

In the code

```
import torch
import torch.nn as nn

class LinearModel(nn.Module):
  def __init__(self):
    super(LinearModel, self).__init__()
    self.linear = nn.Linear(in_features=2, out_features=1, bias=True)

  def forward(self, x):
    return self.linear(x)
```

Linear model in PyTorch

Linear

CLASS `torch.nn.Linear(in_features, out_features, bias=True)` [SOURCE]

Applies a linear transformation to the incoming data: $y = xA^T + b$

Parameters

- **in_features** – size of each input sample
- **out_features** – size of each output sample
- **bias** – If set to `False`, the layer will not learn an additive bias. Default: `True`

Shape:

- Input: $(N, *, H_{in})$ where $*$ means any number of additional dimensions and $H_{in} = \text{in\_features}$
- Output: $(N, *, H_{out})$ where all but the last dimension are the same shape as the input and $H_{out} = \text{out\_features}$.

https://pytorch.org/docs/stable/index.html

Cost function : mean squared error
Optimizer: stochastic gradient descent

Cost
function
+ optimizer

Cost function

In the code

```
reg_loss = nn.MSELoss()
```

MSELoss

CLASS torch.nn.MSELoss(*size_average=None, reduce=None, reduction='mean'*) [SOURCE]

Creates a criterion that measures the mean squared error (squared L2 norm) between each element in the input x and target y.

Optimizer

In the code

```
lr = 0.005
optimizer = optim.SGD(model.parameters(), lr =lr)
```

CLASS torch.optim.SGD(*params, lr=<required parameter>, momentum=0, dampening=0, weight_decay=0, nesterov=False*) [SOURCE]

Implements stochastic gradient descent (optionally with momentum).

Model test

Model Test

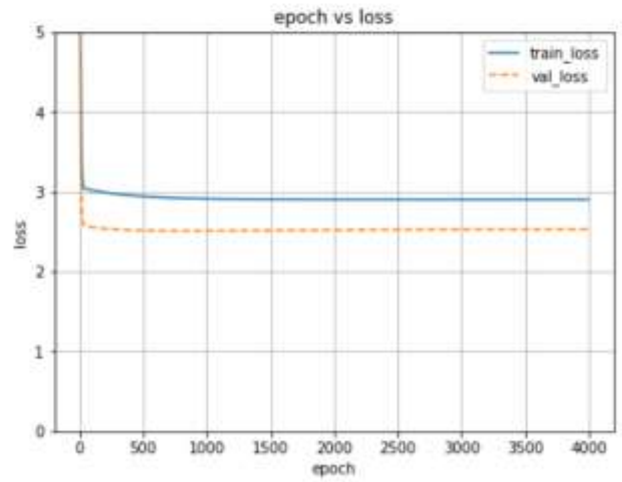
EPOCH=4000

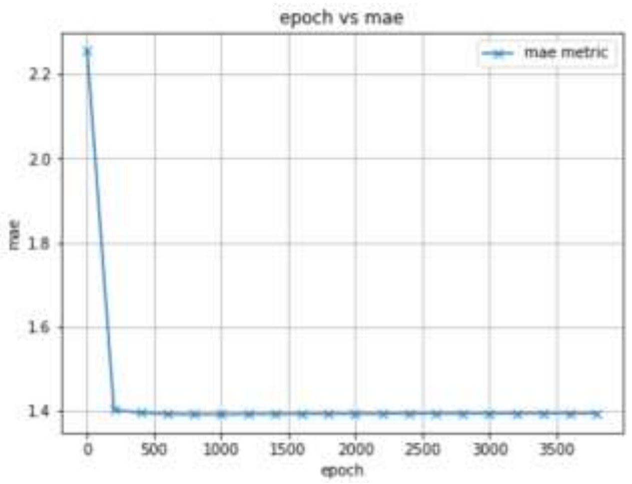
Train set

Validation set

Every 200 epoch

Test set





Lab 2A: Linear regression – multivariable

Exercise 1: different learning rate	Learning rate
Run 1	0.005
Run 2	0.05
Run 3	0.5

Exercise 2: w/ vs w/o bias	Bias
Run 1	Yes
Run 2	No

Exercise 3: different loss function	Learning rate
MSE	0.005
MAE (L1Loss)	0.005

https://pytorch.org/docs/stable/nn.html#loss-functions

You may want to try

1. Increase size of data and re-run it
2. Use different optimizers
   https://pytorch.org/docs/stable/optim.html?highlight=optimizer#torch.optim.Optimizer

Break room

Running a DL code in Graham

SHARCNET™

A consortium of 19 Ontario institutions providing advanced computing resources and support...

Shared
Hierarchical
Academic
Research
Computing
NETwork

computecanada

- Member of Compute Canada and Compute Ontario
- 3,000+ Canadian and international users

- **~50,000 CPU cores**
- **370+ GPUs**
- **10 Gb/s network**
- **100 Gb/s between national centres**

Virtual environment

Allows users to create virtual environments so that one can install Python modules easily

Many versions of same module are possible

```
[isaac@gra-login3 ~]$ module load python
[isaac@gra-login3 ~]$ module list

Currently Loaded Modules:
  1) nixpkgs/16.09   (S)      3) gcccore/.5.4.0  (H)   5) ifort/.2016.4.258 (H)   7) openmpi/2.1.1 (m)   9) python/3.7.4 (t)
  2) imkl/11.3.4.258 (math)   4) icc/.2016.4.258 (H)   6) intel/2016.4      (t)   8) StdEnv/2016.4 (S)

  Where:
   S:     Module is Sticky, requires --force to unload or purge
   m:     MPI implementations / Implémentations MPI
   math:  Mathematical libraries / Bibliothèques mathématiques
   t:     Tools for development / Outils de développement
   H:                Hidden Module



[isaac@gra-login3 ~]$ virtualenv --no-download ~/tf5
Using base prefix '/cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/python/3.7.4'
New python executable in /home/isaac/tf5/bin/python
Installing setuptools, pip, wheel...
done.
[isaac@gra-login3 ~]$ source tf5/bin/activate
(tf5) [isaac@gra-login3 ~]$ deactivate
[isaac@gra-login3 ~]$
```

Lab 2B – Working environment (Graham)

Working environment in Graham

1. Log into graham.computecanada.ca with guest account and p/w : please see [this page] for further details.

   (Use MobaXterm or Putty for Windows / Open terminal in Linux or Mac)

2. Load modules and make a virtual environment: please see [this page] for further details.

```
module load python
module load scipy-stack
virtualenv --no-download ~/ENV
```

3. Activate virtual enviornment and upgrade/install Pip and PyTorch: please see [this page] for further details.

```
source ~/ENV/bin/activate
pip install --no-index --upgrade pip
pip install --no-index torch
pip install --no-index torchvision torchtext torchaudio
pip install sklearn
```

4. (Optional) Deactivate virtual enviornment

```
deactivate
```

Lab 2B – Running simple code

Break room

Running a simple DL code in Graham

1. Clone the repository and change directory to Session_2

```
cd /home/$USER/scratch/$USER
git clone https://github.com/isaacye/SS2020V2_ML_Day2.git
cd SS2020V2_ML_Day2/Session_2
```

2. Activate virtual environment (make sure you load python and scipy-stack module)

```
source ~/ENV/bin/activate
```

3. Run it

```
python SS20_lab2_LRm.py
```

Note that you may want to use a text editor (Nano/emacs/VI): Please see [Nano basic] for further details.

4. File transfer plotting files to your local computer using WinScp or MobaXterm (Windows) / sftp (Linux, Mac) and check it out

Session break:

Please come back by 1:30 PM