2/3/2026

In-class Lecture Notes — Scaling, Data Selection, etc

Use BCEWithLogitsLoss when:
↳ Binary classification with NO activation function on the end

Use BCELoss when
↳ Binary classification WITH sigmoid/tanh etc at the end

Higher loss switching from MSE → BCE does NOT mean MSE was better
(you can't relate loss scores from different loss functions)

NOTE: Training for too long can cause overfitting;
standard epochs ≈ 20

Methods for removing/unifying misinputted/formatted data (in pandas)

- Drop duplicates (df.dropna() drops nulls)
- Drop columns (df.drop(column=...))

What if our data unevenly represents a certain group?
↳ Fix data by:
- Removing extra data from overrepresented category
- Duplicating underrepresented data

SCALING → If one feature has much larger values compared to another feature, model can cheat on outputs to get a low loss

```
from sklearn.preprocessing import
```

scale with: sklearn library OR MinMax scaling

2/3/2026

Lecture Notes (cont.-)

Feature Selection

↳ Why remove data?

- Some features may be irrelevant
  (i.e. id number)
- Some features may be unavailable in the target environment
  (i.e. # of nearby schools present in training data but we can't access it when the model is being used)

ONE-HOTS

If cat=1 and bird=2 and fish=3...
we know fish-bird=cat??
↳ Model makes incorrect relationships

So we use one-hots to stop the model drawing these incorrect conclusions

[1, 0, 0] → DOG!
[0, 1, 0] → CAT! or bird or whatever
[0, 0, 1] → FISH!

Pandas can do this for us with pd.get_dummies

BATCHING

↳ Can speed up training
↳ Process only a portion of data at a time

TORCH DATASET CLASS

from torch.utils.data import Dataset, parent class

2/5/2026

Post-Pre-lab response

↳ Can we expect the model to perform equally well on testing/training data?

A: No, the model generally is expected to perform slightly better on testing data, but in a good setup, the testing and training data should present similar patterns to minimize the gap.

Matplotlib Intro

→ We graph data to:
↳ Identify outliers
↳ Visualize complexity

Matplotlib activity

↳ Used plt.subplots to have several graphs side-by-side
`fig, axs = plt.subplots(2, 2)`
`axs[i,j].plot(...)`

LINE PLOTS

- Good for graphing things related to time
- Connect each "scattered" datapoint w/ a line
- Inputs & outputs must be same length

SCATTER PLOTS

- Scatters data as points.
- Inputs & outputs must be the same length

BAR CHART

- Visualizes number of occurences in discrete categories
- Could be used for student grades

HISTOGRAM

- Like a bar chart but for continuous data

Others:
- Pie Charts
- Box Plots
- 3D Plots

Configure params in plt:
`plt.figure()` ← all in one figure
`plt.scatter(x, y, ...)`
`plt.plot(x, y, ...)`
`plt.etc()[...]`

2/5/2026

Validation

- We can't tell if a model is overfitting data without seeing how it does on the testing data

SO: u can set aside another piece of data as mini testing data, and test the model on the mini testing data after each epoch

↳ If you observe the model overfitting (loss on training decreases but mini-test loss stays the same/increases) we can stop early

Manu Shawar
2/6/2026

Discussion Section Notes

Reviews:

Precision: Of all instances predicted positives, how many are truly positive

Recall: Minimizes false negatives as opposed to false positives
↳ Of all actual positives, how many did it correctly predict

Resources for HW #5

- Python classes etc. available in "course resources" doc in modules
- Week 3 Day 1: Python for Data could be useful to review
- Week 4: Day 1 Activity: Pytorch Review (Logits etc)
- This weeks slides
- Discord!!!
- Read the FREAKING docs yo.
  - Pytorch, NumPy, Sklearn

2/18/2026
Manu Jhawar

Homework #5

PART 0

Plan for Dataset:
- Use <class TrainingData(Dataset):> to initialize
- Pass in scaled data
- Use W5D1 slides if stuck

Plan for DataLoader:
- Use torch.utils.data.DataLoader for each data set
  - ↳ Creates batches
- Use W5D1 slides if stuck

Plan for Train/Test Loop Modifications:
- Update to pull data from DataLoaders instead of from train_inputs / train_outputs
- Use for X_batch, y_batch in <loader>:
  ... (to extract dataloader data)

PART 1

Confusion Matrix:

ACTUAL \ PREDICTED	Healthy	Wilting	Dead
Healthy	5	2	1
Wilting	1	2	2
Dead	1	2	2

Using the confusion matrix, we can see the model does well at predicting healthy plants (very high precision, quite high recall)
- ↳ The model is slightly above average at classifying dead plants
- ↳ The model is bad at classifying wilting plants.

PART 5

- ↳ Scaling the data did slightly improve the model's performance. This is because it allows the model to be less biased towards a specific input type just because it arbitrarily works with larger values.
- ↳ Batching did not noticeably improve the model's performance. This is because batching is primarily optimization and efficiency and practicality-focused, not performance enhancing.
- ↳ Accuracy was the most useful metric (compared to loss) for assessing the model's performance.

35

2/8/2026
Manu Jhawar

Week 5 Day 1 Pre-lab

Precision & Recall

- Accuracy can be a bad measurement if the data set has uneven output representation.
- We can use precision:
  - ↳ How precise are your predictions for a specific class?
- We can use recall:
  - ↳ Out of all datapts in a class, how many did I get right?
- Use cases for each depend on scenario
- Spam emails would rather have high precision

Feature Selection

- Some features are irrelevant to the output data
  - i.e. id number
- We can selectively remove these features to help the model predict on only relevant data.

2/8/2026
Manu Sharma

Week 5: Day 2 Prelab

Graphs

- Pie Chart
  - ↳ Visualize proportions of categorical data
- Bar chart
  - ↳ Visualize distributions of categorical data
- Histogram
  - ↳ Visualize distributions of continuous data
- Line Graph
  - ↳ Visualize trends in data
- Scatter plot
  - ↳ Visualize arbitrary relationships in input-output data

2/18/26

Extra Entry: sklearn review

sklearn.metrics.precision_score

→ y_true, y_pred
Computes precision: $\frac{tp}{tp+fp}$ (true positives / false positives)
Best value is 1, worst value is 0

sklearn.metrics.recall_score

→ y_true, y_pred
Computes recall: $\frac{tp}{tp+fn}$ (true positives / false negatives)
Best value is 1, worst value is 0

sklearn.preprocessing.StandardScaler

→ Standardizes features by removing the mean and scaling to unit variance
- Can also pass in with_mean=bool and with_std=bool

Intro to Machine Learning Engineering Notebook

CMPM17-02

Manu Jhawar
(650) 250-3558
mjhawar@vcsc.edu

Table of Contents

0