Compute Science Department
CS672 – Introduction to Deep Learning (CRN# 74071)
Fall 2023

Project #2 / Due 18-Nov-2023

This exercise will guide you through two important tasks in any Machine Learning / Deep Learning engagement:

- **Perform Exploratory Data Analysis**
- **Device a ML/DL Model (Classification Analysis)** using the **PyTorch** framework.

Performing Exploratory Data Analysis (EDA) on data is of paramount important for every Data Scientist / Data Analyst. Exploratory Data Analysis is often used to uncover various patterns present in the data and to draw conclusions from it. EDA is the core part when it comes to developing a Machine Learning model. This takes place through analysis and visualization of the data which will be fed to the Machine Learning Model. A Machine Learning Model is as good as the training data - you must understand it if you want to understand your model.

Prior commencing your efforts on coding, you must install the PyTorch libraries.
Launch this url: https://pytorch.org/get-started/locally/
Populate the entries and you will be shown the proper Torch library to download/install:

PyTorch Build	Stable (2.1.0)		Preview (Nightly)	
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python		C++ / Java	
Compute Platform	CUDA 11.8	CUDA 12.1	~~ROCm 5.6~~	CPU
Run this Command:	`pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118`			

(A)

Perform an **explanatory data analysis** (**EDA**) on **Breast Cancer (Wisconsin) Diagnostic Classification** obtained from 442 diabetes patients. Besides the need to build a model based on the data provided, you are asked to look for issues in the data and find correlation among the various variables in order to improve breast cancer classifications.

Write **Python/R** (or on any language contained within a notebook) scripts in order to complete the following tasks along with their output. All work should be done and submitted in a single **Notebook (Jupyter or Colab).**

1) Prep the data in order to be ready to be fed to a model.

    Look for missing, null, NaN records.

Find outliers.
Transform data – all entries should be numeric.

2) List all types of data, numeric, categorical, etc.
3) Perform EDA on data

Utilize both:

- Classic approach in EDA (Pandas, NumPy libraries)
- Look for a PyTorch-based EDA tool (if you find one!)

Present dependencies and correlations among the various features in the data.
List the most variables (Feature Importance) that will affect the target label.

(B)
Build a **Deep Learning model** (based on **PyTorch's classification modeling** approaches) that provides reliable and improved accuracy on classifying the correct class of a patient with cancer.

Typical architecture of a classification neural network
The word typical is on purpose.
Because the architecture of a classification neural network can widely vary depending on the problem you're working on.

However, there are some fundamentals all deep neural networks contain:

- An input layer.
- Some hidden layers.
- An output layer.

Much of the rest is up to the data analyst creating the model.
The following are some standard values you'll often use in your classification neural networks.

Hyperparameter	**Binary Classification**	**Multi-class Classification**
Input layer shape	Same as number of features (e.g. 5 for age, sex, height, weight, smoking status in heart disease prediction)	Same as binary classification
Hidden layer(s)	Problem specific, minimum = 1, maximum = unlimited	Same as binary classification
Neurons per hidden layer	Problem specific, generally 10 to 100	Same as binary classification
Output layer shape	1 (one class or the other)	1 per class (e.g. 3 for tree, person or animal photo)
Hidden activation	Usually ReLU (Rectified Linear Unit)	Same as binary classification
Output activation	Sigmoid	Softmax
Loss function	Cross entropy (tf.keras.losses.BinaryCrossentropy in TensorFlow)	Cross entropy (tf.keras.losses.CategoricalCrossentropy in TensorFlow)
Optimizer	SGD (Stochastics Gradient Descent), Adam	Same as binary classification

Perform **binary-class classification modeling analysis** on the ready-to-be-fed data.

Perform the following tasks:

- Evaluating and improving our classification model
- Split the dataset to 80%/20% ratio (training vs test)
- Evaluate your model on the test dataset.
- Plot the graphs of loss vs accuracy curves
- Print the Confusion Matrix

Tune your model on the following parameters:
The **activation parameter** - "relu" & "sigmoid").

The **learning_rate** (also **lr**) parameter – Set a proper range of values for learning rate, usually from 0.1 down to 0.001 or less.
Important Note: You can think of the learning rate as how quickly a model learns. The higher the learning rate, the faster the model's capacity to learn, however, there's such a thing as a too high learning rate, where a model tries to learn too fast and doesn't learn anything. We'll see a trick to find the ideal learning rate soon.

The **number of epochs** - Remember a single epoch is the model trying to learn patterns in the data by looking at it once. Try several sizes: 500, 1000, 2000, etc...

Dataset / Content
The data could be obtained from within scikit-learn library:
https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html#
The columns are as follows; their names are pretty self-explanatory:

```
Breast cancer wisconsin (diagnostic) dataset
--------------------------------------------

**Data Set Characteristics:**

    :Number of Instances: 569

    :Number of Attributes: 30 numeric, predictive attributes and the class

    :Attribute Information:
        - radius (mean of distances from center to points on the perimeter)
        - texture (standard deviation of gray-scale values)
        - perimeter
        - area
        - smoothness (local variation in radius lengths)
        - compactness (perimeter^2 / area - 1.0)
        - concavity (severity of concave portions of the contour)
        - concave points (number of concave portions of the contour)
        - symmetry
        - fractal dimension ("coastline approximation" - 1)

        The mean, standard error, and "worst" or largest (mean of the three
        worst/largest values) of these features were computed for each image,
        resulting in 30 features.  For instance, field 0 is Mean Radius, field
        10 is Radius SE, field 20 is Worst Radius.

        - class:
                - WDBC-Malignant
                - WDBC-Benign
```

scikit learn Install User Guide API Examples Community More

Prev Up Next

scikit-learn 1.3.1
Other versions

Please **cite us** if you use the software.

`sklearn.datasets`.**load_breast_cancer**
`load_breast_cancer`
Examples using `sklearn.datasets.load_breast_canc`

Load and return the breast cancer wisconsin dataset (classification).

The breast cancer dataset is a classic and very easy binary classification dataset.

Classes	2
Samples per class	212(M),357(B)
Samples total	569
Dimensionality	30
Features	real, positive

The copy of UCI ML Breast Cancer Wisconsin (Diagnostic) dataset is downloaded from: https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic