CP640 Machine Learning - Assignment 2
Due Date: Nov 20, 2023 at 11:59 PM

Assignment Submission Guidelines

1. **File Naming**: Ensure your assignment file is named in the following format: your network login followed by the assignment number. For instance, if the username is "barn4520" and you're submitting Assignment 1, the file should be named "barn4520_a01.ipynb".
2. **Assignment Format**: All assignments should be completed using Jupyter Notebook. Once you're done, ensure you run all the cells to verify their functionality, then save your work as a ".ipynb" file. For theoretical or conceptual queries, provide your responses within the notebook using markdown cells. Coding segments must be well-documented for clarity.
3. **Submission Platform**: All submissions must be made via the MyLearningSpace website. We do not accept assignments through email.
4. **Late Submissions**: If you submit your assignment within 24 hours post the deadline, your grade will be reduced by 50%. Unfortunately, we cannot accept submissions made beyond 24 hours from the deadline, and such submissions will receive a grade of 0.
5. **Plagiarism Policy**: All submitted code will undergo a plagiarism check. Engaging in plagiarism can have significant academic consequences. Using generative AI to aid in or fully complete your coursework will be considered academic misconduct.

1 Concept Question

1.1 Support Vector Machines: 3 points

As shown in Figure 1, there are 4 training samples in a 2-dimensional space. $x_1 = (0, 0)$ and $x_2 = (2, 2)$ are being positive, while $x_3 = (h, 1)$ and $x_4 = (0, 3)$ are being negative. h is a parameter whose value falls in the range of $[0, 3]$.

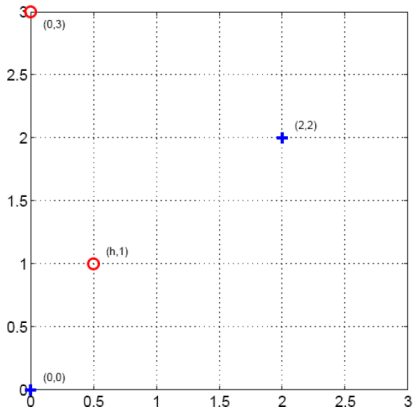

Figure 1: Support Vector Machines

1. How large can $h \geq 0$ be so the training examples are still linearly separable?
2. Will the direction of the maximum margin decision boundary change as a function of h when the samples are separable? Explain your answer with one sentence.

3. What will be the margin obtained by the maximum margin boundary as a function of h? Note that the margin as a function of h is actually a linear function.

1.2 Neural Nets: 2 points

Consider a neural net for a binary classification which has one hidden layer as shown in Figure 2. We use a linear activation function $h(z) = cz$ at hidden units and a sigmoid activation function $g(z) = \frac{1}{1+e^{-z}}$ at the output unit to learn the function for $P(y = 1|x, w)$, where $x = (x_1, x_2)$ and $w = (w_1, w_2, \ldots, w_9)$.

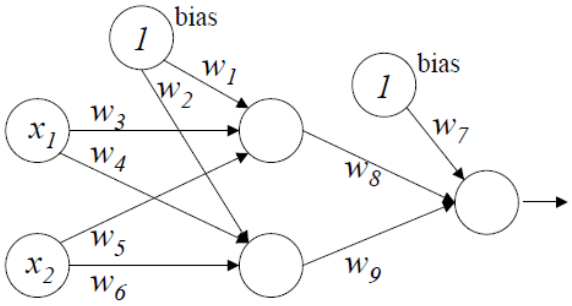

Figure 2: Neural Nets.

1. What is the output $P(y = 1|x, w)$ from the above neural net? Express it in terms of x_i, c and weights w_i.
2. Is it true that any multi-layered neural net with linear activation functions at hidden layers can be represented as a neural net without any hidden layer? Briefly explain your answer.

1.3 Bayesian Rules and Bayesian Networks: 3 points

1. Consider the Bayesian network shown in Figure 3. All the variables are boolean. Write the expression for the joint likelihood of the network in its factored form (1 points).

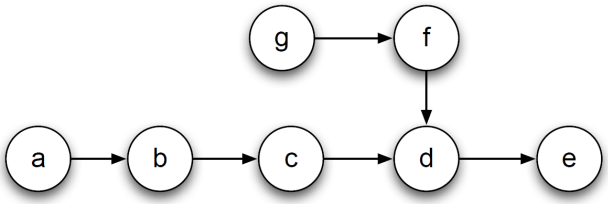

Figure 3: Bayesian Networks

2. Suppose you are given the following set of data shown in Figure 4 with three Boolean input variables a, b, and c, and a single Boolean output variable K. Assume we are using a Naive Bayesian classifier to predict the value of K from the values of the other variables. What is $P(K = 1|a = 1, b = 1, c = 0)$, and $P(K = 0|a = 1, b = 1)$? (2 points)

2 Product Classification

In this exercise, we'll delve into a dataset from the Otto Classification Challenge, a competition hosted by Kaggle in 2015 [1]. The Otto Group stands as a global e-commerce giant, boasting subsidiaries across over 20 countries. This includes renowned brands like Crate & Barrel (USA), Otto.de (Germany), and 3 Suisses (France). Given their expansive daily sales volume worldwide, a systematic product performance analysis becomes vital. However, a challenge arises as their expansive global setup often leads to identical products being categorized differently. Thus, the depth and quality of product analysis largely hinge on the precision of product grouping. Efficient classification paves the way for deeper insights into their product assortment.

[1] https://www.kaggle.com/c/otto-group-product-classification-challenge

a	b	c	K
1	0	1	1
1	1	1	1
0	1	1	0
1	1	0	0
1	0	1	0
0	0	0	1
0	0	0	1
0	0	1	0

Figure 4: Naive Bayesian Networks

The provided dataset comprises 93 features spanning over 200,000 products. The crux of this exercise is to construct various predictive models capable of discerning primary product categories. Utilizing the sklearn library, you'll implement and juxtapose the efficacy of diverse algorithms discussed in our sessions. The dataset, named "otto.csv", is accessible for download via MyLS.

2.1 Data Loading and Preprocessing: 1 points

1. Load the data;
2. The target variable in this dataset is already in a categorical format suitable for multi-class classification. If it were in a string format and needed numerical encoding, you could use label encoding;
3. Extract features and target;
4. Standardize the feature values;
5. Randomly split the data into 70% training and 30% testing.

2.2 Classifier Construction and Evaluation: 4 points

In this question, you will be developing the following different predictive models with *sklearn*.

1. Logistic Regression;
2. Neural Network (Multi Layer Perceptron);
3. Naive Bayes;
4. Linear SVM

2.3 Performance Comparison: 1 points

To evaluate the performance of different methods, please compute the F_1 measure of each algorithm and conclude which method works best in your experiment.

3 COVID Diagnosis

The 2019 coronavirus outbreak (COVID-19) has brought forward several distinguishing characteristics. Although its diagnosis is validated through polymerase chain reaction (PCR), patients inflicted with pneumonia due to the virus might exhibit certain patterns on chest X-rays and CT scans that aren't immediately obvious to the naked eye. In the early phases of 2020, researchers shed light on the clinical and paraclinical aspects of COVID-19, noting that many patients showed irregularities in chest CT scans, often affecting both lungs.

In the realm of computer vision, Convolutional Neural Networks (CNNs) have marked a revolutionary stride. Surpassing traditional methods, they've set new performance benchmarks. These networks have consistently demonstrated their efficacy across varied real-world scenarios, including image categorization. Your mission here is to harness these X-ray visuals to craft a CNN using Keras, aiming to diagnose and interpret the infection. Such a tool could empower doctors to make more informed decisions while awaiting a radiologist's review, offering them a virtual second opinion to corroborate their evaluation of a patient.

The provided dataset in MyLS encompasses 188 chest X-ray images, capturing both COVID-19 affected patients and healthy individuals. This dataset is bifurcated into training and testing subsets. Each subset is further divided into two categories: "NORMAL" and "PNEUMONIA", intended for model development and assessment, respectively. To bring this predictor to life, the outlined steps in this assignment need to be followed.

1. Image pre-processing (2 points). Image pre-processing stands as a foundational and pivotal step when working with image data. In this phase, it's essential to resize all images to our target dimensions (64x64) and normalize them by dividing by 255. Based on the computational capacity available, images can be represented using all three RGB channels or just one channel, signifying grayscale. Optionally, further data augmentation can be employed to enhance the dataset's richness. Relevant functionalities can be found in the Keras' ImageDataGenerator module [2].

   - ImageDataGenerator
   - flow_from_directory

   A sample setting of converting the image is shown below. Explore your own for best result.

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
datagen=ImageDataGenerator(
    zoom_range=0.2, # Zooming rate of the image
    horizontal_flip=True, # Make a horizontal copy
    rescale=1.0/255.0, # Normalize the new images
    width_shift_range=0.10, # Percentage of width shifting
    height_shift_range=0.10, # Percentage of height shifting)
```

2. Model building (3 points). You have the flexibility to craft our convolution layers. Consider the architecture depicted in Figure 3 as an illustration. Here, the design commences with a pair of convolutional layers, succeeded by two max-pooling layers. Post flattening the output, it integrates a fully connected dense layer, culminating with an output layer. Each layer employs the Relu activation function, with the exception of the final output layer which utilizes the sigmoid function. To further improve the predictor's performance, you can also **optionally** add a tensorflow.keras.callbacks

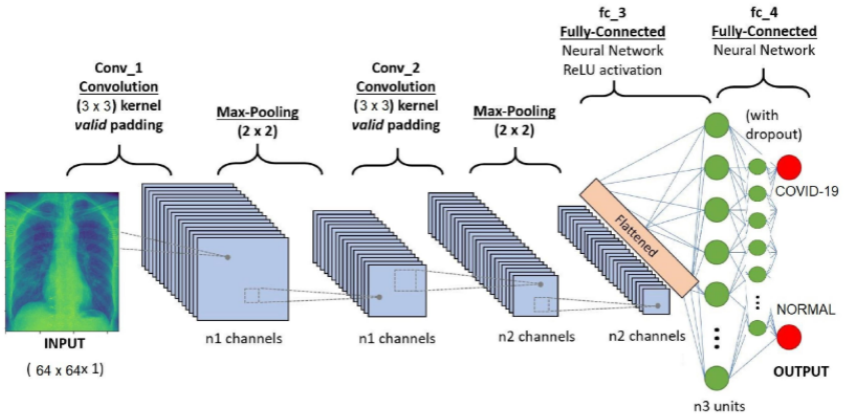

Figure 5: CNN structure

module. Callback is a strategy to reduce over fitting and save time. For example, you can use EarlyStopping if the accuracy does not improve for certain iterations. This can be implemented as below.

```
from tensorflow.keras.callbacks import EarlyStopping
earlystop=EarlyStopping(patience=6)
```

You can also **optionally** add dropout layers to further reduce overfitting [3] and introduce Adam to optimize the SGD process in learning parameters[4].

[2] https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator
[3] https://machinelearningmastery.com/how-to-reduce-overfitting-with-dropout-regularization-in-keras/
[4] https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

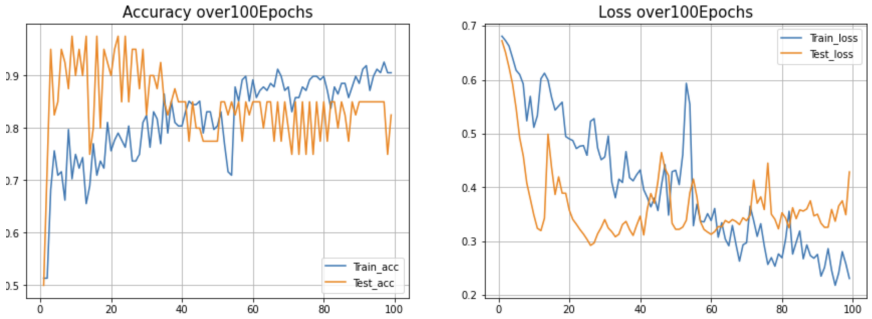

Figure 6: Performance evaluation

3. Model evaluation (3 points). To gauge the proficiency of our predictor, you are tasked with employing evaluation metrics such as Accuracy and Losses (Binary Cross Entropy), tracking their evolution over 20 epochs. Further, to gain deeper insights into the learning trajectory, it's pivotal to display outcomes for both training and testing datasets. Submissions should incorporate visualizations akin to those in Figure 5, illustrating Accuracy and Losses. Concluding, you'll furnish an analysis rooted in your experimental findings, delving into discussions around potential overfitting.