# Practical Machine Learning
## ENGR 491/891

# Programming Assignment 3

## Spring 2022

## Logistic Regression

### Assignment Goals

The goal of this assignment is to solve multi-class classification problems, visualization of image samples, and analyzing errors.

- **Part A**: Multi-class Classification – Structured Data
- **Part B**: Multi-class Classification – Unstructured Data & Analysis of Model Errors

### Assignment Instructions

Note: You must use Scikit-Learn to create the models. You are allowed to use Python libraries such as Pandas, NumPy, and Matplotlib.

i. The code should be written in a Jupyter notebook. Use the following naming convention.
   <lastname1>_ <lastname2>_ assignment3.ipynb
ii. The Jupyter notebook should be submitted via Canvas.

The programming code will be graded on **both implementation, correctness and quality of the results**.

## Score Distribution

ENGR 891: 100 points

## **Part A:** Multi-class Classification – Structured Data

You will perform multi-class classification on the following dataset using the Logistic Regression and KNN models. Your goal is to:

- Maximize test accuracy as well as precision, recall & F1 score for each class

**Note**: grading will be based on the quality of the obtained results. Only the model implementation will not earn full credit. You must try every possible way to improve the test performance.

**Dataset**:
The dataset, given in the *winequality-white.csv* file, is related to the white variants of the Portuguese "Vinho Verde" wine. It provides the physicochemical (inputs) and sensory (the output) variables. The dataset consists of characteristics of white wine (e.g., alcohol content, density, amount of citric acid, pH, etc.) with target variable "quality" representing rating of wine. The target variable "quality" ranges from 3 to 9. Higher rating indicates better quality of a wine. The classes are ordered and not balanced (e.g., there are much more normal wines than excellent or poor ones).

Input features (based on physicochemical tests):

- fixed acidity
- volatile acidity
- citric acid
- residual sugar
- chlorides
- free sulfur dioxide
- total sulfur dioxide
- density
- pH
- sulphates
- alcohol

**Pre-processing**: [**3 pts**]

- Load the CSV file as a Pandas DataFrame object.
- Create a data frame object for the features and another data frame object for the target.
- Create a **bar plot** to display the class distribution.
- Convert the above two DataFrame objects into two NumPy arrays (you may use the NumPy "asarray" function).
- Convert the target array type into "int".
- Partition the dataset into training & test subsets: 80% training & 20% test (you may use Scikit-Learn's train_test_split() function).

**Feature Selection**:
You need to use the optimal set of features for the Logistic Regression models, which may consist of all features or a subset. Note that dropping the poorly correlated features with the target may not necessarily improve the test performance. You need to determine the optimal feature set empirically.

[**4 pts**]

**Experiments**:
You will perform multi-class classification using the following models. If a model requires the data to be standardized, you must do so prior training. You must perform hyprparameter tuning. For experiment 3, use the Pipeline object that will augment features, standardize (if required), and create a model.

- Experiment 1) K-Nearest Neighbors (K-NN)
- Experiment 2) Logistic Regression (use batch Gradient Descent)
- Experiment 3) Polynomial Logistic Regression (use batch Gradient Descent)

**Report**:
- Optimal set of features in experiments 2 and 3. If you used all features, report that as well.
- For each optimal model report training accuracy, test accuracy, test confusion matrix, weighted average of the test precision, recall, F1 scores, degree of the optimal polynomial model (experiments 3), and optimal values of the model hyperparameters. Use the following table in your report.

[**30 pts**]

| Model | Train Accuracy | Test Accuracy | Test Precision (weighted avg) | Test Recall (weighted avg) | Test F1 (weighted avg) | Optimal Hyperparameter values | Wall time for hyperparameter tuning |
|---|---|---|---|---|---|---|---|
| K-NN | | | | | | | |
| Logistic Regression | | | | | | | |
| Polynomial Logistic Regression | | | | | | Degree = | |

Answer the following questions.
- Q-1) Which model did you find most **effective** (optimal test performance)? Explain why this model performed better than other models in your experiments.
- Q-2) Which model did you find most **efficient** (less time for hyperparameter tuning)? Explain why it is efficient.

[**3 + 3 pts**]

## **Part B:** Multi-class Classification – Unstructured Data & Analysis of Errors

You will create a Logistic Regression classifier (using Scikit-Learn) to perform multi-class on the following unstructured dataset. You will also analyze the misclassifications error.

**Note**: grading will be based on the quality of the obtained results and visualization. Only the model implementation will not earn full credit. You must try every possible way to improve the test performance.

**Dataset**:
The Fashion MNIST dataset contains 70,000 28 x 28 grayscale images of 10 fashion categories. The train set contains 60,000 images, and the test set contains 10,000 images.

You may directly load this dataset using the Keras API:
https://keras.io/api/datasets/fashion_mnist/

**Pre-processing**: **[5 pts]**
You will need to perform some pre-processing steps, i.e., reshaping and scaling.

- **Reshape the data**: The dimension of the training set is 60000 x 28 x 28 and test set is 10000 x 28 x 28.

Before you use this data for the ML model, you need to flatten each sample (i.e., 28 x 28 = 784) such the dimension of training and test set becomes:
- 60000 x 784
- 10000 x 784

Use the NumPy "reshape" function for this purpose.
https://numpy.org/doc/stable/reference/generated/numpy.reshape.html

Example: X_train = X_train.reshape((X_train.shape[0], 784))

- **Scale the data**: use the min-max scaling technique, i.e., by divide the training & test data matrix by 255.0.

Example: X_train = X_train/255.0

**Visualization**:
Display 15 random images from each class as shown below. Below each image, show its label. See the labels and their names from this link:

https://keras.io/api/datasets/fashion_mnist/

To visualize an image, use the matplotlib.pyplot.imshow function, as shown in the following notebook.
https://github.com/rhasanbd/Logistic-Regression-Comparative-Understanding/blob/master/Logistic%20Regression-6-Gradient%20Descent%20Optimization%20Techniques.ipynb

For creating a grid of images (example above), use the matplotlib.pyplot.subplot function. See an example from the following notebook on how to create a grid of images.
https://github.com/rhasanbd/Study-of-Analogy-based-Learning-Image-Classification/blob/main/Study%20of%20Analogy%20based%20Learning-Image%20Classification-1.ipynb

**[10 pts]**

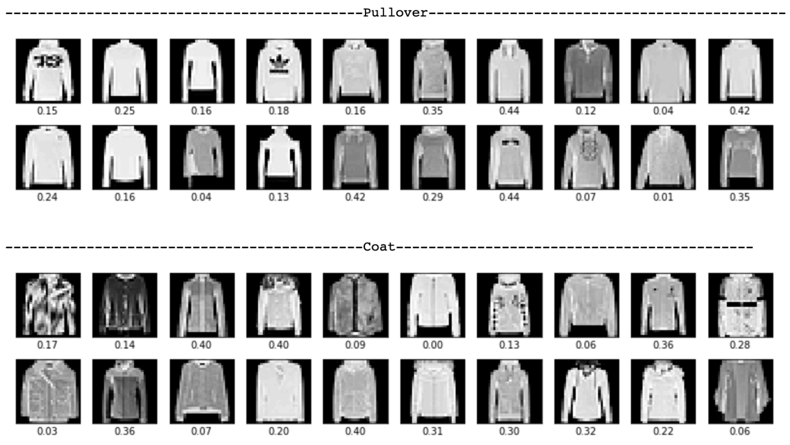
**Experiments**:

- Experiment 4) Perform multi-class classification by optimally training a Logistic Regression classifier using the batch gradient descent algorithm. Report train accuracy, test accuracy, and test classification report. You MUST try to optimize the test performance of the model by using suitable values for the following hyperparameters of the Scikit-Learn LogisticRegression class: penalty, C, solver, max_iter. Following notebook could be useful:

[https://github.com/rhasanbd/Logistic-Regression-Comparative-Understanding/blob/master/Logistic%20Regression-3-Multiclass%20Classification-Batch%20Gradient%20Descent.ipynb](https://github.com/rhasanbd/Logistic-Regression-Comparative-Understanding/blob/master/Logistic%20Regression-3-Multiclass%20Classification-Batch%20Gradient%20Descent.ipynb)

**[20 pts]**

- Visualize a set of at least 20 misclassified test samples and their predicted probabilities for the following two classes (class 5: Sandal and class 7: Sneaker). First, store the misclassified test samples and their predicted probabilities for the two classes in four lists. Then, use the lists for visualization. The misclassified test images and their predicted probabilities should be displayed in a grid as shown below for two sample classes.



**[10 pts]**

- Q-3) For each set of misclassified images, select at least 2 images from each class with predicted probabilities less than 0.15 and explain why your model failed to make accurate predictions on these samples. Copy the images in your report followed by your explanation, as shown below.

  Example:
  Following is a misclassified test image of a coat. The model misclassified this image as " " (find out what class the model predicted) because …

**[12 pts]**

**Deliverables**:

You will submit two products.

- A single Jupyter notebook containing all experiments. For each experiment show the required performance measures. If it’s convenient, you may split your experiments in multiple notebooks and submit. Your code must be clearly annotated. Also add header descriptions for each block.
- A PDF copy of the written report with the cover page (get it from Canvas). The report must include the following items.

a) Part A: Optimal set of features for the Logistic Regression model.
b) Part A: Using the table (given in Part A) report training accuracy, test accuracy, weighted average of the test precision, recall, F1 scores, degree of the optimal polynomial model (experiments 3), and optimal values of the model hyperparameters.
c) Part A: Answers to Q-1 to Q-2.
d) Part B: Display 15 random images from each class.
e) Part B: Report train accuracy, test accuracy, and test classification report for experiment 4.
f) Part B: Answers to Q-3.