# CSE4/574 Coding Assignment One[*]

## Warm Up: Data Analysis

In this part, you will explore how to load, process, visualize and analyze different types of datasets. Data preprocessing and visualizing are some of the core parts of the machine learning pipeline.

Select any THREE datasets from the list below and provide a data analysis. All datasets are provided in the folder.

- Amazon Top Selling Book dataset
- Netflix dataset
- Diamond dataset
- Penguin dataset
- Titanic dataset
- Wine Quality dataset
- Insurance dataset

Practice the following steps:

- Pick a dataset from the list above.
- Create a Jupyter Notebook or .py script.
- Load a dataset(e.g.pd.read_csv())
- Using inbuilt functions in 'pandas' library, extract the main statistics about the dataset (more details). Use at least 5 functions (e.g. describe(), head()).
- Perform data preprocessing:
  - Calculate the sum of missing entries in the dataset (e.g. isnull().sum(axis=0))
  - If applied,for missing values:
    - Fill the missing values with the most frequent value or
    - Drop the rows with missing values (e.g. dropna()).
- If applied: convert the columns with data type string to data type categorical.
- Using any data visualization library (e.g. matplotlib, seaborn, plotly), draw at least 5 visualization graphs related to your dataset. You can utilize any columns or a combination of columns in your dataset to generate graphs, e.g. correlation matrix, features vs the target, counts of categorical features vs the target.
- Repeat

[*]In reference to Alina Vreshchaka's coding assignment for CSE4/574.

## Linear Regression on the winequality-red Dataseet [40% points]

In this part, you will implement linear regression using the mean square error (MSE) on the winequality-red dataset:

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \mathbf{w}^T\,\mathbf{x}_i)^2 \;.$$

In matrix notation, the loss is rewritten as

$$J(\mathbf{w}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\,\mathbf{w})^T(\mathbf{y} - \mathbf{X}\,\mathbf{w}) \;,$$

where $\mathbf{X}$ is the input data matrix, $\mathbf{y}$ is the target vector, and $\mathbf{w}$ is the weight vector for regression, as defined in the slides.

**Dataset** The dataset consists of 12 columns, with attributes as: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, quality.

**Steps**:

- Import required libraries (not allowed: sklearn or any other libraries with in-built functions that help to implement ML).
- Do appropriate data preprocessing as described above.
- Choose which features to be included as input features, and which one serves as the output scale.
- Create the data matrices for $\mathbf{X}$ (input) and $\mathbf{y}$ (target) in a shape $X \in \mathbb{R}^{N\times d}$ and $\mathbf{y} \in \mathbb{R}^N$, where $N$ is a number of data samples and $d$ is a number of features.
- Divide the dataset into training and test, as 80%-training and 20%-testing dataset.
- Print the shape of your $\mathbf{X}$_train, $\mathbf{y}$_train, $\mathbf{X}$_test, $\mathbf{y}$_test.
- Use the closed-form solution to solve the linear regression problem:

$$\mathbf{w} = (\mathbf{X}^T\,\mathbf{X})^{-1}\,\mathbf{X}^T\,\mathbf{y} \;.$$

- Get the predictions and calculating the sum of squared errors on testing data:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \mathbf{w}^T\,\mathbf{x}_i)^2 \;.$$

- Plot the predictions vs the actual data values (ground truth).

In your report, you should:

- Describe the lienar regression problem, the dataset, and how your pre-process and construct your dataset for training and testing.
- Describe the steps of your experiments.
- Provide your loss value for training data and the final weight vector.
- Show the plot comparing the predictions vs the actual test data.
- Discuss the benefits/drawbacks of the method you used for computing weights.

## Logistic Regression on the penguins Dataset [60% points]

In this part, you will work on logistic regression and will use a logistic function to model a binomial (Binary/Bernoulli) output variable. Logistic regression model predicts that the observation belongs to a particular category. To generate these probabilities, logistic regression uses the sigmoid function. This function maps a real number to a value between 0 and 1.

**Dataset** You will use Palmer Archipelago (Antarctica) penguin dataset. It contains three penguin species and includes measurements of bill length, bill depth, flipper length and body mass. Overall, we are provided with 344 data samples. The dataset consists of 7 columns:

- species: penguin species (Chinstrap, Adelie, or Gentoo)
- culmen_length_mm: culmen length (mm)
- culmen_depth_mm: culmen depth (mm)
- flipper_length_mm: flipper length (mm)
- body_mass_g: body mass (g)
- island: island name (Dream, Torgersen, or Biscoe) in the Palmer Archipelago (Antarctica)
- sex: penguin sex (female, male)

**Steps**:

- Import required libraries (not allowed: sklearn or any other libraries with in-built functions that help to implement ML).
- Read, preprocess and print main statistic about the dataset as described above.
- Convert features with string datatype to categorical (species,island,sex).
- Normalize non-categorical features (culmen_length_mm, culmen_depth_mm, flipper_length_mm, body_mass_g):
  - Find the min and max values for each column
  - Rescale dataset columns to the range from 0 to 1
- Choose your target Y. For this dataset, there is a number of options:
  - We can use binary classifier to predict which gender a penguin belongs to (female or male). In this case column sex can be used as $y$ (target).
  - We can use binary classifier to predict if a penguin location is Adelie island or not. In this case column island can be used as $y$ (target).
  - Create the data matrices for $\mathbf{X}$ (input) and $\mathbf{y}$ (target) in a shape $\mathbf{X} \in \mathbb{R}^{N \times d}$ and $\mathbf{y} \in \mathbb{R}^{N}$, where $N$ is a number of data samples and $d$ is a number of features.
  - Divide the dataset into training and test, as 80%-training, 20%-testing dataset.
  - Print the shape of your $\mathbf{X}$_train, $\mathbf{y}$_train, $\mathbf{X}$_test, $\mathbf{y}$_test.

Use the following structure of the code to define logistic regression:

```
class LogitRegression()
    def __init__()
        # Takes as an input hyperparameters: learning rate and the
        number of iterations.

    def fit():
        # This method performs the training.
        # Initialize weights
        # For a number of iterations
          # Call gradient_descent function
          # Call cost function and keep it in an array,
          # e.g., loss.append()

    def sigmoid():
        # Define a sigmoid function as
        #           $\sigma(z) = \frac{1}{1+e^{-z}}$

    def cost():
        # Implement the loss function for Logistic Regression,
        # which is available in page 42 of the slides

    def gradient_descent():
        # Implement the gradient formula, which is available
        # in page 42 of the slides

    def predict(self, X):
        # Return the predicted result in the binary form
        # $\hat{y} = \sigma(\mathbf{w}^T\mathbf{x}+b) = +1$ if $\sigma(z) \geq 0.5$; and 0 otherwise.
```

- Train the model:
  - Defile a model by calling LogitRegression class and passing hyperparameters, e.g., model = LogitRegression(learning_rate, iterations)
  - Train the model, by calling fit function and passing your training dataset, e.g., model.fit(X_train, y_train)
  - Suggested hyperparameters:
    learning_rate = 1e-6
    iterations =1 00000
    weights = np.random.uniform(0, 1)
- Make a prediction on test dataset by counting how many correct/incorrect predictions your model makes and print your accuracy.
- Print out the loss values over each iterations.

You can make binary (e.g. male/female) or multiclass (e.g. Dream/Torgersen/Biscoe) classification. Accepted accuracy rate for penguin dataset is above 64%.

In the report, you should

- Describe logistic regression.
- Describe the dataset: how you preprocess your dataset and how to apply it to logistic regression.
- Provide your best accuracy and the weight vector.
- Include loss graph and provide a short description.
- Explain how hyperparameters influence the accuracy of the model. Provide at least 3 different setups with learning rate and #iterations and discuss the results.
- Discuss the benefits/drawbacks of using a Logistic Regression model.

**Submission** You need to write a report containing the two models above. You need to creat a Github to put your runable code, and include the Github link to your report (see more details in the syllabus). And submit to UBLearns.

**Report template**: https://www.overleaf.com/7651663239dcdnbfhkmrfs

**Submission deadline**: 11:59PM, October 27, 2023.