**CS 5588 Big Data Analytics and Application**  
**January 30, 2025**

### **Hands-On Session Plan: Classification and Regression Models**

This session focuses on building feedforward neural networks for **both regression and classification tasks** using PyTorch. The instructor will guide students through **both classification and regression tasks** using the **Diabetes Dataset**, while students will independently perform similar tasks using the **California Housing Dataset**. The session concludes with GitHub documentation and a reflective survey.

### **Submission**

* [**Survey Form**](https://forms.gle/thgEd6GNoDSVM7Xc9)
* Students must complete a survey at the end of the session and submit their completed notebooks to GitHub.

### **High-Level Guidelines**

#### 1. Objectives

* Build and train feedforward neural networks for **both classification and regression tasks**.
* Leverage GPU acceleration in Google Colab for faster training.
* Gain experience in documenting and sharing work via GitHub.

#### 2. Datasets

* **Diabetes Dataset (Instructor-Led):**
  + **Features:** 10 baseline variables for 442 diabetes patients (e.g., age, BMI, blood pressure).
  + **Tasks:**
    - Regression: Predict a quantitative measure of disease progression.
    - Classification: Categorize patients into "low progression" or "high progression" based on a threshold.
  + **Dataset URL:** [Diabetes Dataset on Kaggle](https://www.kaggle.com/datasets/mathchi/diabetes-data-set).
* **California Housing Dataset (Student Activity):**
  + **Features:** Census data (e.g., median income, housing median age, average occupancy).
  + **Tasks:**
    - Regression: Predict median house values.
    - Classification: Categorize houses as "low value" or "high value" based on a threshold.
  + **Dataset URL:** [California Housing Dataset on Kaggle](https://www.kaggle.com/datasets/camnugent/california-housing-prices).

### **Session Flow**

#### 1. Google Colab Setup:

* Initialize Google Colab and enable GPU acceleration.
* Test GPU availability using torch.cuda.is\_available().

#### 2. Diabetes Dataset (Instructor-Led):

##### **Regression Task:**

* **Objective:** Predict the quantitative progression of diabetes.
* **Steps:**
  1. **Dataset Preparation:**
     + Load the Diabetes Dataset using load\_diabetes() from sklearn.datasets.
     + Preprocess the data using StandardScaler.
     + Split the data into training and test sets using train\_test\_split.
     + Convert the data into PyTorch tensors.
  2. **Neural Network Definition:**
     + Build a feedforward neural network:
       - Input layer size = number of features (10).
       - Two hidden layers with 64 and 32 units, using ReLU activation.
       - Output layer with one neuron for regression.
  3. **Training:**
     + Use MSELoss as the loss function and Adam as the optimizer.
     + Train for 100 epochs, logging the loss every 10 epochs.
  4. **Evaluation:**
     + Calculate **mean squared error** and **R-squared**.
     + Visualize predictions vs. true values using matplotlib.

##### **Classification Task:**

* **Objective:** Classify patients into "low progression" or "high progression" based on a threshold (e.g., median progression value).
* **Steps:**
  1. **Dataset Preparation:**
     + Use the same Diabetes Dataset but create binary labels:
       - Label patients as 1 (high progression) or 0 (low progression).
       - Use the median progression value as the threshold.
     + Split the data into training and test sets and preprocess it.
     + Convert the data into PyTorch tensors.
  2. **Neural Network Definition:**
     + Build a feedforward neural network:
       - Input layer size = number of features (10).
       - Two hidden layers with 64 and 32 units, using ReLU activation.
       - Output layer with 2 neurons for binary classification (softmax activation).
  3. **Training:**
     + Use CrossEntropyLoss as the loss function and Adam as the optimizer.
     + Train for 100 epochs, logging the loss every 10 epochs.
  4. **Evaluation:**
     + Calculate metrics such as accuracy, precision, recall, and F1-score.
     + Generate a confusion matrix and visualize it.

#### 3. California Housing Dataset (Student Activity):

##### **Regression Task:**

* **Objective:** Predict median house values.
* **Steps:**
  1. **Dataset Preparation:**
     + Load the California Housing Dataset using fetch\_california\_housing() from sklearn.datasets.
     + Preprocess the data using StandardScaler.
     + Split the data into training and test sets.
     + Convert the data into PyTorch tensors.
  2. **Neural Network Definition:**
     + Replicate the neural network used for the Diabetes Dataset regression task.
     + Input layer size = number of features (8 for California Housing Dataset).
  3. **Training:**
     + Use MSELoss as the loss function and Adam as the optimizer.
     + Train for 100 epochs and log the loss every 10 epochs.
  4. **Evaluation:**
     + Calculate **mean squared error** and **R-squared**.
     + Visualize predictions vs. true values.

##### **Classification Task:**

* **Objective:** Classify houses into "low value" or "high value" based on a threshold (e.g., median house value).
* **Steps:**
  1. **Dataset Preparation:**
     + Use the same California Housing Dataset but create binary labels:
       - Label houses as 1 (high value) or 0 (low value) based on the median value threshold.
     + Preprocess and split the data as in the regression task.
     + Convert the data into PyTorch tensors.
  2. **Neural Network Definition:**
     + Replicate the neural network used for the Diabetes Dataset classification task.
     + Input layer size = number of features (8 for California Housing Dataset).
  3. **Training:**
     + Use CrossEntropyLoss as the loss function and Adam as the optimizer.
     + Train for 100 epochs and log the loss every 10 epochs.
  4. **Evaluation:**
     + Calculate metrics such as accuracy, precision, recall, and F1-score.
     + Generate a confusion matrix and visualize it.

### **GitHub Submission**

1. Save the completed notebooks:
   * Diabetes\_Classification\_Regression.ipynb (Instructor-Led).
   * California\_Housing\_Classification\_Regression.ipynb (Student Activity).
2. Create a GitHub repository:
   * Add the notebooks and a descriptive README file explaining the tasks and results.
3. Share the repository link for review.

### **Wrap-Up**

#### 1. Key Takeaways:

* Hands-on experience building feedforward neural networks for regression and classification tasks.
* Understanding evaluation metrics like MSE, R-squared, accuracy, precision, and recall.
* Familiarity with GitHub for professional documentation.

#### 2. Survey:

* Students complete a short survey reflecting on:
  + Challenges faced.
  + Lessons learned.
  + Suggestions for improving future sessions.