Artificial Neural networks

**Case Study: SONAR — Detecting Mines vs. Rocks**

**1️⃣ Business Objective**

**Goal:**  
To build an intelligent system that can automatically detect whether an underwater sonar signal is reflected from a **metallic mine** (potentially dangerous) or a harmless **rock**.

This is vital for:

* **Maritime safety:** Prevent ships and submarines from colliding with mines.
* **Naval defense:** Identify and safely remove underwater mines.
* **Resource exploration:** Distinguish between useful metal structures and natural seabed objects.

**2️⃣ Problem Statement**

In underwater environments, sonar (sound navigation and ranging) is used to detect objects. However, raw sonar signals can be noisy and difficult for humans to interpret consistently.

This dataset:

* Contains 208 sonar returns.
  + 111 are from **metal cylinders** (mines).
  + 97 are from **rocks**.
* Each sonar return is represented by **60 numeric features**, each measuring the energy of the signal in a frequency band.

The problem:  
👉 To **train a Deep learning model** that can learn the difference in signal patterns and **classify new sonar signals** as either *Mine* (M) or *Rock* (R) — accurately and reliably.

**Dataset: "sonardataset.csv"**

**Features (Inputs)**

* There are **60 numerical variables**, each representing the **energy in a specific frequency band** of the sonar signal.
* In the original dataset, they’re just unnamed columns **V1, V2, ..., V60** — you can keep it clear and simple:

**2️⃣ Target (Output)**

* The label is a single categorical variable indicating:
  + "M" for **Mine**
  + "R" for **Rock**

================================================================================

**Tasks**

**1. Data Exploration and Preprocessing**

* Begin by loading and exploring the "Alphabets\_data.csv" dataset. Summarize its key features such as the number of samples, features, and classes.
* Execute necessary data preprocessing steps including data normalization, managing missing values.

**2. Model Implementation**

* Construct a basic ANN model using your chosen high-level neural network library. Ensure your model includes at least one hidden layer.
* Divide the dataset into training and test sets.
* Train your model on the training set and then use it to make predictions on the test set.

**3. Hyperparameter Tuning**

* Modify various hyperparameters, such as the number of hidden layers, neurons per hidden layer, activation functions, and learning rate, to observe their impact on model performance.
* Adopt a structured approach like grid search or random search for hyperparameter tuning, documenting your methodology thoroughly.

**4. Evaluation**

* Employ suitable metrics such as accuracy, precision, recall, and F1-score to evaluate your model's performance.
* Discuss the performance differences between the model with default hyperparameters and the tuned model, emphasizing the effects of hyperparameter tuning.

**Evaluation Criteria**

* Accuracy and completeness of the implementation.
* Proficiency in data preprocessing and model development.
* Systematic approach and thoroughness in hyperparameter tuning.
* Depth of evaluation and discussion.
* Overall quality of the report.

**Additional Resources**

* [TensorFlow Documentation](https://www.tensorflow.org/)
* [Keras Documentation](https://keras.io/)

We wish you the best of luck with this assignment. Enjoy exploring the fascinating world of neural networks and the power of hyperparameter tuning!