Homework: Description

Overview

In this assignment, you will revisit the **pre-processed "Wine Quality" dataset** from Homework #1 and apply **regression** techniques to uncover relationships between different variables (features) in the dataset. The assignment is divided into **three main parts**:

1. **Single-Feature Linear Regression**
2. **Multiple Linear Regression (3 Features)**
3. **Polynomial Regression**

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
7.4	0.7	0.0	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
7.8	0.88	0.0	2.6	0.098	25.0	67.0	0.9968	3.2	0.68	9.8	5
7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.997	3.26	0.65	50.0	5
11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.998		0.58	9.8	6
7.4	0.7	0.0	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
7.4	0.66	0.0	1.8	0.075	13.0	40.0	0.9978	3.51	0.56	9.4	5

Homework: Description

1. Single-Feature Linear Regression

Objective

- Select **one** dependent variable (output) and **one** independent variable (feature) from the dataset.
- Train a **simple linear regression** model to predict the output from the single chosen feature.

Deliverables

- A **plot** with data points and the regression line.
- A **short write-up** explaining your training procedure, final parameters, final loss, and observations.

Homework: Description

1. Single-Feature Linear Regression

Tasks/Steps

Data Selection:

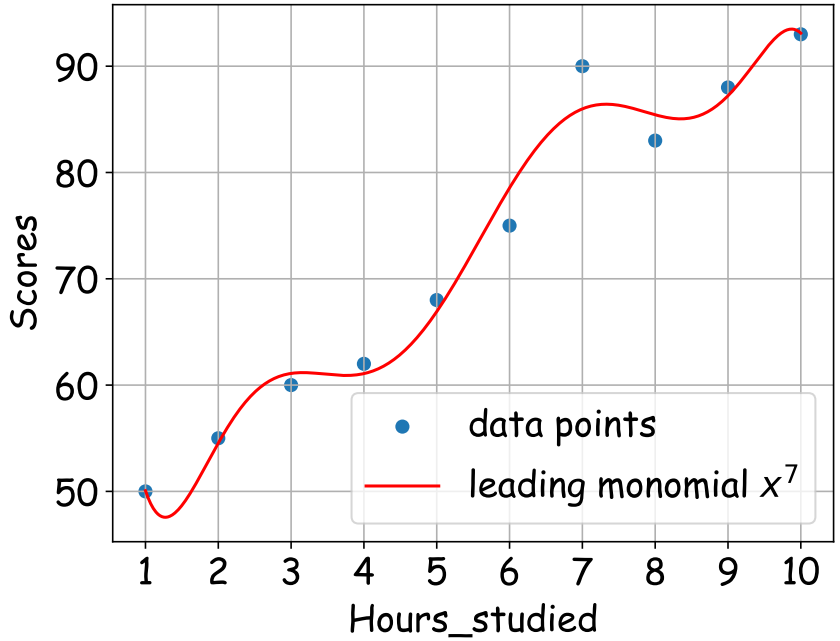
Justify which single **feature** and which **output** you chose.

Implementation in PyTorch:

Initialize model parameters.

Forward pass and loss function (MSE).

Optimization algorithm (gradient descent).



Visualization:

Plot the **data points** (scatter plot).

Plot the **best-fit line** learned by your model on the same figure.

Analysis:

Summarize the training process.

Discuss any difficulties or anomalies you observed when fitting the line.

Interpret how well the linear model fits the data visually and numerically (final loss, etc.).

Homework: Description

2. Multiple-Feature Linear Regression

Objective

- Select **3 features** and **1 output** from the dataset (may be the same output variable as in Part 1 or a different one).
- Train a **simple linear regression** model to predict the output from the chosen features.

Deliverables

- If it is an achievable task, the deliverables should be similar with the previous Single-Feature Linear Regression; follow the tasks in the next page.
- If you think it is not an achievable task, provide the analysis to show the reason; just ignore the tasks in the next page.

Homework: Description

2. Multiple-Feature Linear Regression

Tasks/Steps

Data Selection:

Justify which three **features** and which **output** you chose; Provide a brief rationale.

Implementation in PyTorch:

Build a multi-feature linear model.

Train and optimize (MSE; gradient descent).

Results:

Show the **final loss** (training error; MSE).

If the model **successfully** converges, report your final set of learned weights and bias; if the model **fails** to converge or you encounter difficulties, **analyze** and explain potential reasons.

Interpretation:

Discuss whether the multi-feature regression model appears to be a better fit than a single-feature model.

Reflect on any new challenges that arose when using multiple features.

Homework: Description

3. Polynomial Regression

Objective

- Using the same 3 features and 1 output from Part 2, implement polynomial regression of at least three different polynomial degrees (e.g., degree=2, degree=4, degree=6).
- Train a **polynomial regression** model to predict the output from the chosen features.

Deliverables

- A **summary table** or short discussion comparing performance for each chosen polynomial degree.
- Plots or numeric results illustrating how well each polynomial model fits.
- A **reflection** on potential risks of higher-degree polynomials (e.g., overfitting).

Homework: Description

3. Polynomial Regression

Tasks/Steps

Feature Transformation:

- Explain how you generated polynomial terms (e.g., by manually expanding each feature or using a PyTorch mechanism for polynomial features).
- Decide how you handle interactions (only single-feature powers vs. cross-terms).

Training & Model Comparison:

- Train a polynomial regression model for each degree (≥ 3 degrees).
- Compare the training losses across different degrees.

Analysis:

- Discuss any overfitting or under-fitting you observe.
- Identify which polynomial degree produced the most favorable result based on **loss** or other metrics.
- Provide any insights into runtime or complexity differences.

Homework: Description

4. Binary Classification

Tasks/Steps

Choose a Binary Label:

Construct a **binary classification** label from the dataset (if there is no direct available binary labels found, you need to create one first, e.g., citric acid = 0 or not.)

Implement Logistic Regression in PyTorch:

Loss function: Binary Cross-Entropy (BCE).

Data Splitting & Preprocessing:

Clearly split your data into **training** and **testing** (or validation) sets.

Model Training & Evaluation:

Train on the training set for a certain number of epochs or until convergence.

Report & Visualization

Summarize final **training loss**, **test performance metrics**, and any interesting findings.

(Optional) Provide a **decision boundary** plot if feasible (for a single or two-feature scenario), or a confusion matrix heat-map to illustrate predictions vs. ground truth.

Homework: Description

Implementation Requirements

1. PyTorch Only
   - You must implement the regression logic (forward pass, gradient updates, etc.) in PyTorch.
2. Data Handling
   - You may use pandas or plain Python to load the dataset from CSV or other formats.
   - Feel free to do any necessary feature engineering or transformations to handle missing values, scaling, etc.
3. Plots & Visualization
   - matplotlib or seaborn is recommended for plotting.
   - Clearly label axes, legend, and titles for each figure.
4. Written Report
   - Provide your observations, interpretations, and analysis for each part.
   - Discuss any difficulties or additional experiments you performed.

Homework: Description

Report & Grading

1. Organization (20%)
   - Is your submission clearly structured? Are code, plots, and analysis sections logically presented?
2. Correctness & Implementation (40%)
   - Proper usage of PyTorch for linear and polynomial regression.
   - Evidence of correct gradient-based training for each part.
3. Analysis & Interpretation (40%)
   - Clarity in explaining results, including final losses, potential reasons for success or failure.
   - Depth of insight into overfitting, data distribution, or hyperparameter choices.
4. Extra Credit / Deep Thinking (up to +10%)
   - If your **report is well-organized**, provides **deeper insights** or **additional experiments** (e.g., trying different regularization, comparing different subsets of features, exploring other polynomial expansions), you may receive extra points.