Université Kasdi Merbah Ouargla

Faculté des nouvelles technologies de l'information et de la communication

Département d'informatique et technologie d'information

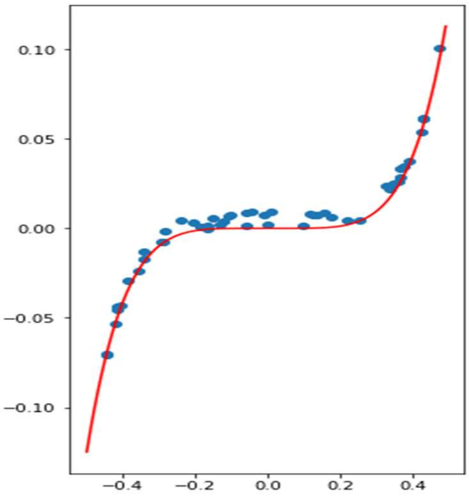
Module : Machine Learning I

Academic Year: 2024/20025

Lab Sheet

Exercise 1:

In this exercise, you will generate a data sample *(x,y)* as show in the following figure. Additionally, you will use two approaches to tune the value of *a:* brute-force approach and gradient descent. You will then calculate the average number of iterations required in each approach until achieving the threshold mean squared error (MSE).



1. **Generate Data Sample:**
   - Use NumPy to generate a data sample *(x,y)* consisting of 20 data points.
   - The independent variable x should be generated as random values uniformly distributed between [−0.5, 0.5].
   - The dependent variable y should be generated using the formula $y = ax^3 + random\ noise$, where a is arbitrary value chosen by you and the random noise is added to simulate real-world data.
2. **Brute-Force Approach:**
   - Randomly sample values of a within the specified range [0, 100] to test.
   - Calculate the MSE for each randomly sampled value of a.
   - Repeat this process until the MSE falls below a specified threshold=0.015.
   - Calculate the average number of iterations needed to reach the threshold MSE.
   - How many sampling operations are theoretically required to find the value of *a*, given that a is sampled from a normal distribution with a mean of 0 and standard deviation of 1? Calculate for the true a = 0.1 and a = 0.31, assuming values are rounded to two decimals; What are the best and the worst scenario for the value of a?
3. **Gradient Descent to Tune *a*:**
   - Initialize a random value for a.

- Perform gradient descent to iteratively update the value of a such that it minimizes the MSE between the predicted and actual values of *y*.
- Use a learning rate and a predefined number of iterations for gradient descent.
- Calculate the average number of iterations required until achieving the threshold MSE = 0.015.

4. **Visualize Data Sample and Predictions:**

- Plot the generated data sample *(x,y)* using a scatter plot.
- Plot the final predictions obtained after tuning a using both approaches.

5. **Reflection:**

- Reflect on the differences between the brute-force approach and gradient descent in terms of computational efficiency and effectiveness in finding the optimal value of a.
- Discuss the implications of the average number of iterations on the performance and scalability of each approach.

Exercise:

In this exercise, you will implement the gradient descent algorithm to solve a linear regression problem. Given a dataset consisting of *(x, y)* coordinates, your task is to iteratively update the parameter *a* to minimize the cost function using both Mean Squared Error (MSE) and Mean Absolute Error (MAE) losses. Additionally, you will estimate the configurations (*a*) for the first two iterations of the gradient descent algorithm.

- **Dataset:**You are provided with a dataset containing 8 samples with the following *x* and *y* coordinates:
  - X = [-0.3, -0.1, -0.4, 0.2, 0.1, -0.0 , 0.4, -0.05]
  - Y = [-0.9, 0. , -4.4, 0.5, 0.3, 0.9, 7.7, 0.6]
- **Tasks:**
  - Calculate the Cost $J(\theta)$ using both MSE and MAE losses.
  - Define the objective function to be minimized.
  - Solve the linear regression problem iteratively using the gradient descent algorithm.
  - Estimate the configurations (*a*) for the two first iterations.
  - Transform this solution into Python code.