**Báo cáo bài thực hành số 3**

1. **Giới thiệu bài báo cáo**

Bài thực hành gồm có 4 bài

**Bài 1**: Phương pháp hồi quy tuyến tính với Numpy trên dữ liệu California House Price:

* Mục đích: Xây dựng một mô hình hồi quy tuyến tính để dự đoán giá trị trung bình của nhà ở California, sử dụng dữ liệu về giá nhà và các thuộc tính liên quan.
* Phương pháp: Sử dụng thư viện Numpy để thực hiện việc xây dựng mô hình và tối ưu hóa các tham số.

**Bài 2:** Phương pháp Logistic Regression với Gradient Descent trên dữ liệu Predict students’ dropout and academic success:

* Mục đích: Áp dụng Logistic Regression để dự đoán khả năng học viên rời bỏ hoặc thành công trong học tập.
* Phương pháp: Sử dụng Gradient Descent để tối ưu hóa tham số mô hình và đánh giá mức độ chính xác của mô hình.

**Bài 3:** Phương pháp Softmax Regression với Gradient Descent trên dữ liệu Predict students’ dropout and academic success:

* Mục đích: Sử dụng Softmax Regression để phân loại học viên thành các nhóm: graduate, dropout và enroll.
* Phương pháp: Sử dụng Gradient Descent để huấn luyện mô hình và đánh giá hiệu suất thông qua việc trực quan hóa giá trị của hàm mất mát.

**Bài 4:** Sử dụng các thư viện Machine Learning để thực thi lại Linear Regression,

Logistic Regression và Softmax Regression:

* Mục đích: So sánh kết quả và hiệu suất giữa việc thực hiện các mô hình từ đầu (sử dụng Numpy và Gradient Descent) và việc sử dụng thư viện Machine Learning như Scikit-learn hoặc Skorch.
* Phương pháp: Sử dụng các thư viện ML để tái thực hiện các phương pháp hồi quy và phân loại đã được thực hiện từ trước.

1. **Nội dung bài báo cáo**

**Bài 1:**

**Đọc dữ liệu và tiền xử lý:**

* + Sử dụng thư viện Pandas để đọc tập dữ liệu từ file CSV.
  + Loại bỏ cột 'ocean\_proximity' vì không cần thiết cho mô hình hồi quy tuyến tính.

**Chuẩn bị dữ liệu:**

* + Phân chia dữ liệu thành features (X) và target (Y).
  + Sử dụng **train\_test\_split** từ **sklearn** để chia dữ liệu thành tập huấn luyện và tập kiểm tra với tỷ lệ 80-20.

**Thêm cột bias (độ bias) vào dữ liệu:**

* + Thêm cột toàn giá trị 1 vào đầu mỗi sample để xử lý bias trong mô hình hồi quy tuyến tính.

**Huấn luyện mô hình:**

* + Sử dụng phương pháp hồi quy tuyến tính thông qua việc tính toán theta (tham số mô hình) bằng công thức dựa trên ma trận.
  + Theta được tính bằng cách sử dụng công thức của ma trận nghịch đảo:

*θ =* (.)-1. .

**Đánh giá mô hình:**

* + Sử dụng hàm **MSE** để tính Mean Squared Error (MSE) giữa giá trị dự đoán và giá trị thực tế trên tập kiểm tra.
  + In ra giá trị MSE để đánh giá hiệu suất của mô hình trên tập kiểm tra.
  + Kết quả MSE bạn đã tính được là khoảng 4,705,519,646.39, cho thấy mức độ lỗi của mô hình. Đây là giá trị trung bình của bình phương sai số giữa giá trị dự đoán và giá trị thực tế trên tập kiểm tra.

**Bài 2:**

**Đọc dữ liệu và tiền xử lý:**

* + Sử dụng thư viện Pandas để đọc tập dữ liệu từ file CSV.
  + Ánh xạ các giá trị trong cột Target bao gồm “Dropout”, “Graduate” và “Enrolled” thành các giá trị số theo thứ tự lần lượt là 0, 1 và 0

**Chuẩn bị dữ liệu:**

* Sử dụng **train\_test\_split** từ **sklearn** để chia dữ liệu thành tập huấn luyện và tập kiểm tra với tỷ lệ 70-30.

**Định nghĩa mô hình Logistic Regression:**

* Xây dựng hàm sigmoid để sử dụng trong quá trình huấn luyện mô hình.
* Định nghĩa hàm **model** để huấn luyện mô hình Logistic Regression sử dụng gradient descent.
* Mô hình sử dụng các tham số như learning rate và số vòng lặp để tối ưu hóa trọng số.

**Huấn luyện mô hình:**

* Huấn luyện mô hình trên tập huấn luyện với số vòng lặp và learning rate được chỉ định.
* In ra giá trị cost sau mỗi 1000 iterations để theo dõi quá trình học của mô hình.

**Vẽ biểu đồ cost:**

* Vẽ biểu đồ để hiển thị giá trị cost theo số lần lặp để kiểm tra quá trình học của mô hình.

**Bài 3:**

**Nhập dữ liệu và Tiền xử lý**

* Nhập bộ dữ liệu từ một tệp CSV (students.csv).
* Kiểm tra thông tin về bộ dữ liệu bằng cách sử dụng data.info().

Chuyển đổi nhãn văn bản trong cột 'Target' ('Dropout', 'Graduate', 'Enrolled') thành giá trị số (0, 1, 2).

**Định nghĩa và Huấn luyện Mô hình**

* Định nghĩa các hàm trợ giúp (augment\_feature\_vector, vectorized\_compute\_probabilities, gradient\_descent\_iteration, predict, compute\_accuracy) để thực hiện hồi quy softmax.
* Thực hiện các vòng lặp gradient descent (softmax\_regression) để huấn luyện mô hình trên dữ liệu huấn luyện (x\_train, y\_train).

**Quá trình Huấn luyện Lặp lại**

* Vòng lặp huấn luyện chạy trong 1000 lần, cập nhật trọng số của mô hình trong mỗi lần lặp bằng gradient descent. Quá trình này nhằm mục tiêu làm giảm hàm mất mát và tối ưu hóa các tham số (theta) để có dự đoán tốt hơn.
* Đánh giá Mô hình
* Đánh giá mô hình đã được huấn luyện bằng cách tính toán độ chính xác trên bộ test (x\_test, y\_test) bằng cách sử dụng hàm compute\_accuracy.

**Bài 4:**

**1. Linear Regression**

**Nhập dữ liệu và tiền xử lý**

* Dữ liệu được đọc từ tệp 'housing.csv' và hiển thị ra DataFrame.
* Cột 'ocean\_proximity' không được sử dụng cho việc dự đoán và bị loại bỏ khỏi DataFrame.

**Chuẩn bị dữ liệu**

* + Sử dụng **train\_test\_split** từ **sklearn** để chia dữ liệu thành tập huấn luyện và tập kiểm tra với tỷ lệ 80-20.
  + Đối với bộ dữ liệu, chỉ có các giá trị số được sử dụng cho việc dự đoán, do đó, chúng được chuẩn hóa bằng **StandardScaler**.

**Huấn luyện mô hình Linear Regression:**

* + Mô hình Linear Regression được tạo và huấn luyện trên tập huấn luyện sau khi dữ liệu đã được chuẩn hóa.
  + Các trọng số tốt nhất (weights) và bias tốt nhất được in ra màn hình.

**Dự đoán và đánh giá**

* + Dự đoán được thực hiện trên tập kiểm tra và so sánh với nhãn thực tế.
  + Các giá trị dự đoán và nhãn thực tế được in ra cho 100 mẫu đầu tiên.
  + Các độ đo đánh giá như Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), và R-squared (R²) được tính toán và hiển thị.

**So sánh với Normal Equation**:

* + Các độ đo đánh giá từ Linear Regression được so sánh với Normal Equation để kiểm tra tính chính xác của mô hình.

**2.Logistic Regression**

**Khác biệt về Tiền Xử lý Dữ Liệu:**

* + Có thể có sự khác biệt trong cách dữ liệu được tiền xử lý hoặc biến đổi giữa việc thực hiện thủ công và Scikit-learn. Đảm bảo rằng việc chuẩn hóa đặc trưng, xử lý giá trị bị thiếu, mã hóa biến phân loại, v.v., được thực hiện một cách nhất quán giữa hai cách tiếp cận.

**Tham Số Hoặc Cấu Hình Mô Hình:**

* + Mô hình **LogisticRegression** trong Scikit-learn có thể sử dụng các siêu tham số hoặc cấu hình khác so với thực hiện thủ công của bạn. Kiểm tra xem bạn đã thiết lập các tham số tương tự như sức mạnh của việc điều chuẩn (**C**), loại phạt (**l1** hoặc **l2**), loại bộ giải quyết, v.v.

**Phân Chia Tập Huấn Luyện/Thử Nghiệm:**

* + Sự chênh lệch cũng có thể xuất phát từ sự khác biệt trong cách tập dữ liệu huấn luyện và thử nghiệm được chia giữa việc thực hiện thủ công và Scikit-learn. Đảm bảo cùng một trạng thái ngẫu nhiên hoặc seed được sử dụng cho cả hai cách tiếp cận để đảm bảo cùng một cách phân chia.

**Lỗi Trong Thực Hiện Thủ Công:**

* + Xem xét thực hiện Logistic Regression thủ công để kiểm tra xem có lỗi nào trong việc tính toán trọng số, sai số hoặc dự đoán không. Có thể có sai sót trong cách thực hiện tự tạo của bạn.

**3.Softmax\_Regression**

**Đọc và Chuẩn bị Dữ liệu**

* + Đọc dữ liệu từ tệp CSV**:** Sử dụng thư viện Pandas để đọc dữ liệu từ tệp 'students.csv' và lưu nó vào DataFrame có tên là **df**.
  + Loại bỏ dữ liệu thiếu: Dùng **df.dropna()** để loại bỏ các dòng trong DataFrame chứa dữ liệu thiếu.

**Chuẩn bị Dữ liệu Đầu Vào và Đầu Ra**

* + Encoding cột 'Target'**:** Dữ liệu trong cột 'Target' được chuyển đổi từ dạng văn bản ('Dropout', 'Graduate', 'Enrolled') sang dạng số nguyên bằng LabelEncoder() từ thư viện Scikit-learn.
  + Chia dữ liệu: Tập dữ liệu sau khi được chuẩn bị được chia thành hai phần: **X** chứa dữ liệu đầu vào và **y** chứa dữ liệu đầu ra. **X** chứa tất cả các cột trừ cột cuối cùng ('Target') và **y** chỉ chứa cột cuối cùng ('Target').

**Chuẩn hóa Dữ liệu Đầu Vào**

* + StandardScaler: Dữ liệu đầu vào X được chuẩn hóa sử dụng StandardScaler() để đảm bảo các đặc trưng có cùng đơn vị đo lường và giúp mô hình học tốt hơn.

**Xây dựng và Huấn luyện Mô hình**

* + Logistic Regression Model: Sử dụng mô hình Logistic Regression từ thư viện Scikit-learn (**LogisticRegression**), với **multi\_class='multinomial'** và **solver='lbfgs'**.

**Huấn luyện Mô hình:**

* + Dùng **logistic\_reg\_model.fit()** để huấn luyện mô hình trên dữ liệu huấn luyện (**X\_train**, **y\_train**).

**Đánh giá Mô hình**

* + Tính toán các trọng số, sai số và độ chính xác của mô hình sử dụng **logistic\_reg\_model.coef\_**, **logistic\_reg\_model.intercept\_**, và **accuracy\_score()**.

**Dự đoán và Kiểm tra**

* + Sử dụng mô hình đã huấn luyện để dự đoán lớp của dữ liệu trong tập kiểm tra (**X\_test**) bằng **logistic\_reg\_model.predict()**.

**Kết quả**

* + **Độ chính xác của mô hình:** Tính toán độ chính xác của mô hình trên tập kiểm tra bằng **accuracy\_score(y\_test, y\_pred)**.

1. **Nhận xét**

**Nhận xét về sự khác biệt và hạn chế:**

* Từ đầu và sử dụng thư viện: Xây dựng mô hình từ đầu yêu cầu kiến thức sâu rộng hơn về cơ chế hoạt động của mô hình và thuật toán tối ưu. Sử dụng thư viện Machine Learning giúp tiết kiệm thời gian và công sức, nhất là đối với các tác vụ phổ biến.
* Hạn chế của việc sử dụng thư viện: Sử dụng thư viện có thể làm giảm sự hiểu biết sâu rộng về cơ chế hoạt động của mô hình. Đôi khi, việc điều chỉnh và tinh chỉnh mô hình có thể trở nên mập mờ do việc sử dụng các hàm tích hợp sẵn trong thư viện.

**Cải thiện:**

* Đối với việc xây dựng từ đầu: Cần thực hành nhiều hơn, tăng cường hiểu biết về các thuật toán và tối ưu hóa, cũng như làm việc với dữ liệu thực tế.
* Đối với việc sử dụng thư viện: Nắm vững kiến thức cơ bản để hiểu rõ hơn về cách thức làm việc của các hàm tích hợp sẵn trong thư viện, và tinh chỉnh mô hình một cách chính xác và hiệu quả.