**Assignment 3**

**Part 1- Simple Linear Regression**

Linear regression is among the simplest regression methods. One of the main advantages of using it is ease of interpreting results. Simple linear regression is special case of regression where target feature is dependent on single variable, and then we find the best fitting line.

$y = m*x + c$

Dataset

The dataset is available at

**https://github.com/mishravipul/data/raw/main/simple_linear_data.csv**

Dataset 'Student Performance' provided by UCI Machine Learning repository.

Original dataset: https://archive.ics.uci.edu/ml/datasets/student+performance

Features (X)

- G2 - second year math grades (numeric: from 0 to 100)

Target (y)

- G3 - third year math grades (numeric: from 0 to 100, output target)

Objective

To gain understanding of single linear regression through implementing the model from scratch

Tasks

- Read the data from above mentioned dataset and define X and y as numpy array
- Add column at position 0 with all values=1 (pandas.DataFrame.insert function)
- Print rows from 40 to 55.
- Print the shape and datatype of both X and y
- Implement simple linear regression from scratch
    - Write hypothesis function to predict values
    - Write function for calculating mean_squared_error
    - Write function to return gradients for given weights

    - Perform gradient descent taking help of above functions
- Remove outliers, train again and see the difference in error.
- Replace "*weights = np.random.rand(2)*" line in gradient descent with below line, train again and visualize results.
- Play with learning rate and max_iterations
- Generalize the code for multivariate(multiple) linear regression

Resources

- Linear regression maths: https://www.youtube.com/watch?v=ZkjP5RJLQF4
- Simple linear regression: https://www.youtube.com/watch?v=iAgYLRy7e20
- Tutorial: https://machinelearningmastery.com/implement-simple-linear-regression-scratch-python

**Part 2: Multiple linear regression**

Multiple linear regression is simply the linear regression extended to problems where the dependent or output variable is determined by more than one independent variable.

y^ (w, x) = w_0 + w_1 * x_1 + … + w_p * x_ps

Dataset

The dataset is available at

**https://github.com/mishravipul/data/raw/main/multiple_linear_data.csv**

This is the **modified version** of the dataset *'Student Performance'* provided by UCI Machine Learning repository.
Original dataset: https://archive.ics.uci.edu/ml/datasets/student+performance

Features (X)

1. age - student's age (numeric: from 15 to 22)
2. address - student's home address type (binary: 'U' - urban or 'R' - rural)
3. famsize - family size (binary: 'LE3' - less or equal to 3 or 'GT3' - greater than 3)
4. reason - reason to choose this school (nominal: close to 'home', school 'reputation', 'course' preference or 'other')
5. studytime - weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
6. failures - number of past class failures (numeric: n if 1<=n<3, else 4)

7. schoolsup - extra educational support (binary: yes or no)
8. famsup - family educational support (binary: yes or no)
9. paid - extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)
10. activities - extra-curricular activities (binary: yes or no)
11. higher - wants to take higher education (binary: yes or no)
12. internet - Internet access at home (binary: yes or no)
13. romantic - with a romantic relationship (binary: yes or no)
14. freetime - free time after school (numeric: from 1 - very low to 5 - very high)
15. goout - going out with friends (numeric: from 1 - very low to 5 - very high)
16. health - current health status (numeric: from 1 - very bad to 5 - very good)
17. absences - number of school absences (numeric: from 0 to 93)
18. G1 - first year math grades (numeric: from 0 to 100)
19. G2 - second year math grades (numeric: from 0 to 100)

Output target (Y)

20. G3 - final year math grades (numeric: from 0 to 100, output target)

Objective

To learn multiple linear regression and practice handling categorical features

Tasks

- To load the data and print first 5 rows
- Transform categorical features into numerical features. Use either one hot encoding, label encoding or any other suitable preprocessing technique.
- Define X matrix (independent features) and y vector (target feature)
- Train Linear Regression Model (sklearn.linear_model.LinearRegression class)
- Print 'Mean Squared Error' (MSE) obtained on the same dataset i.e. same X and y (sklearn.metrics.mean_squared_error function)
- Predict on a numpy array defined by you

```
>>> new_data = np.array([1,0,1,.....,30,20]).reshape(1,-1)
>>> print("Predicted grade:",model.predict(new_data))
```

**Challenge yourself:-**

- Train LassoRegression and RidgeRegression as well. Read about them from scikit-learn user guide.
- *Step-up challenge*: Get down the MSE (mean squared error) below 3.25 using linear models
- Implement multiple linear regression from scratch
- Plot loss curve (Loss vs number of iterations)

Helpful links

- Scikit-learn documentation for linear regression: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
- Read till where you feel comfortable: https://jakevdp.github.io/PythonDataScienceHandbook/05.06-linear-regression.html