DAT-LA–8 Homework 1: Python & Linear Regression

Submit by: Monday, Sept. 14
Submit completed Python file to: DAT-LA–8-STUDENTS GitHub repository.

In this homework, you will see how easy it is to implement one of the most fundamental machine learning algorithms – linear regression. That said, the main objective of this homework is to practice Python, write functions, and better understand the summation operator.

The Summation Operator

In this class, every model makes predictions based on a finite number of data points. So, instead of evaluating integrals on continuous data, we evaluate sums on discrete data.

Given a sequence of N numbers, $X = \{x_1, x_2, \ldots, x_N\}$, the summation operator $\sum$ is defined as:

$$\sum_{i=1}^{N} x_i = x_1 + x_2 + \ldots + x_N.$$

Note that mathematicians tend to be one-indexed and loop from 1 up to and including N.

However, Python is zero-indexed. So, thinking about the summation operator as a `for` loop, we would loop from 0 up to and including $N - 1$.

Writing summations in Python

Given `X = [1, 2, 3, 4, 5]`, the above summation is simply: `sum(X)`.

For a more complex example, suppose we wish to find $\sum_{i=1}^{N} x_i^2 + 2x_i + 1$. Then, we can write a list comprehension or generator that computes the inner function of the summation. Thinking about it in this way, evaluating this sum is still easy:

```
sum(xi*xi + 2*xi + 1 for xi in X)
```

When implementing math formulas in Python, you can drastically simplify your mental model. The following suggestions do not precisely follow PEP 8, but they will make your code much more readable when checking your work against formulas. Try to:

1. Break down the formula into pieces, each represented by a variable. For example, the formula may be composed of meaningful subformulas. If part of the formula is the variance of `X`, perhaps your variable will be `vx` or `varx`. The slope formula in the exercises below was rewritten in terms of means to simplify it, removing most of the summations in the original text.
2. If the formula has no meaningful subformulas, consider literally translating parts of the formula into variable names, e.g. $\sum x_i y_i z_i$ might be represented by the variable `Sxyz`.
3. Use names that mirror the names of the formula's variables. For example, in the summation above we used notation consistent with the math formula – the variable `X` is consistent with a set, and `Sx` is consistent with being the summation of elements of `X`.
4. Make your final line look almost identical to the math formula. For example, the variables `Sxyz` and `Sx` might represent each summation in $\sum x_i y_i z_i - \sum x_i$, making the final Python line very similar: `Sxyz - Sx`.

Linear Regression (non-matrix)

To approach a machine learning problem, three choices must be made: (1) the model, (2) the method of fitting the model, and (3) the validation method.

For this homework, we choose our model to be the linear equation **y = mx + b**. To fit the model, we choose to find m and b which **minimize the sum of squared residuals**. To validate, we choose to **visually graph the model**.

Fitting the Model.

For this discussion, suppose we have N data points: $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$.

A **residual** is the difference between the observed value and the predicted value. In this case, given a predicted value $\hat{y} = mx + b$, the residual is $y_i - \hat{y}$. You may already suspect (based on the name) that the sum of squared residuals is:

$$\sum_{i=1}^{N} (y_i - \hat{y}_i)^2.$$

Suppose we choose this as our measure of "goodness of fit". Then, surprisingly (or not), we can solve for the `m` and `b` exactly which minimize it, using the equations in the below exercises.

The derivation of these formulas uses only basic calculus and algebra. In class, we will derive a similar version with a single parameter. But, if you are curious how the below formulas are derived, work through this excellent step-by-step derivation:

http://isites.harvard.edu/fs/docs/icb.topic515975.files/OLSDerivation.pdf

Homework 1

For each of the below exercises, after you write a function, test the function on the sample lists `X_test` and `Y_test`. Hand-compute what the function should return so you can provide evidence your function works.

1. Write a function `mean(X)` that returns the mean value of a list of numbers, as follows:

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N}$$

2. Write a function `intercept(X, Y, m)` that returns the y-intercept `b`. Make sure to test your function with a test `m` value to ensure it works. What should it return with `X_test`, `Y_test`, and `m = 1.0`? (Hint: Use `mean(X)` to calculate $\bar{x}$ and $\bar{y}$.)

$$b = \bar{y} - m\bar{x}$$

3. Write a function `slope(X, Y)` to return the slope m. If the slope is not what you expect, double-check each intermediate variable by hand-computing what it should be for `X_test` and `Y_test`. Place `print()` statements inside your function to view the intermediate variables. Note that the second terms in each subtraction are not inside the summation operators.

$$m = \frac{\sum_{i=1}^{N} x_i y_i - N\bar{x}\bar{y}}{\sum_{i=1}^{N} x_i^2 - N(\bar{x})^2}$$

4. Graph the data with the predicted values. Does it look like a best-fit line? See if your slope and y-intercept are the same as what is returned using Excel or another statistics program.

5. **Bonus problem!** Using the `csv` module, read in the `hw1-mammals.csv` file, which contains mammal name, body size (kg), and brain weight (g). Is there a linear relationship between body weight and brain weight? Is there a linear relationship between log(body weight) and log(brain weight)? Can you write a formula which given a body weight, predicts the brain weight? (Hint: To retrieve a column of data from a 2-D list, consider using list comprehensions!)