Project 1 – Linear Regression

Saturday, November 17, 2018 1:37 PM

In this project, we want to find how various factors contribute to a school-aged child becoming the number one sprinter at their school. First we will plot the (fake) data to see if we can spot a pattern. Then we will generate what is called the "best fit line" using the process of linear regression (sometimes called the "least squares" method).

1. Create a file with a main method called track_star_linear_regression.py.
2. Create a file called InputCsvs that contains a python class by the same name.
    a. Create a constructor that takes a file name as a variable and stores it in an instance variable.
    b. Create a method that reads in the following csv and stores the data in the first three columns in three different lists. The first column contains the runners' speeds; the second column holds the runners' ages and the third column contains the alphabetized positions of the runners' initials with regard to the other runners' initials.
    c. The same method or a different method should return the three lists when called.
    d. **Note that Python doesn't know we are reading in numbers; it sees everything as text strings. You will need to convert the numbers from strings to floats.**

    track_st…

3. Within track_star_linear_regression.py, import the following class and call the show_plot method to display the data. For now, just use the defaults for the last three parameters in show_plot.

    There will be two graphs that display one after the other. On the first, graph the age on the x-axis and the speed on the y-axis. On the second, graph the initials on the x-axis and the speed on the y-axis.

    Plotter

    As an aside, you will note that this class uses ggplot, a package that is normally used in the R programming language. It allows us a few niceties that are difficult to achieve in pyplot. Don't be concerned with these subtleties. You don't need to understand how this class works; just call show_plot().

4. Create a class called Algorithms with a method called find_best_fit.
    a. This method will take two lists as input, one representing the x-data values and the other the y-.
    b. Calculate the slope of the best fit line using the following formula:

    $$\hat{m} = \frac{\sum_{i=1}^{n}(y_i - \overline{y})(x_i - \overline{x})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

    - y-bar is the mean of all the values in the y list.
    - x-bar is the mean of all the values in the x list.
    - y_i represents each value in the y list one by one.
    - x_i represents each value in the x list one by one.

    So what to do?

    - First, find the mean of x and y.
    - Next, create a list called y_bar_diffs of all the y values minus the mean of y. Do the same with x.
    - Multiply each value in the y_bar_diffs list by each value in the x_bar_diffs list and add them all up to get a single number. This is the numerator.
    - Now square each value in the x_bar_diffs list and add them all up to get a number. This is the denominator.
    - Divide the numerator by the denominator and assign it to a variable called m-hat.

    c. Calculate the y intercept of the best fit line using the following formula:
    $\hat{b} = \overline{y} - \hat{m}\overline{x}$

    d. Calculate each y value on the best fit line using the following formula. Store all the values in a list called y_hat_list.
    $\hat{y}_i = \hat{m}x_i + \hat{b}$

    - m-hat is the value you calculated in step b.
    - x_i is each x value in the original list you passed into the method.

- b-hat is the value you calculated in step c.
- y_i-hat is your newly calculated value that you should store in a list called y_hat_list.

e. Calculate the R squared value using the following formula:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}$$

f. Return the slope, the intercept and r-squared from this method.

5. Back in track_star_linear_regression.py, use Algorithms to calculate the best fit lines for age and speed, and initials and speed.
6. Finally pass these values into the show_plot() method, replacing the default values with the newly calculated values.

Neat, huh? Hopefully, you have some visual insight but we will discuss what this all means at the completion of the project.