|  |  |
| --- | --- |
| Name Of The Student | Himanshu |
| Internship Project Topic | TCS iON RIO-210: Build a Classification Model for Drug Trials Dataset |
| Name of the Organization | TCS iON |
| Name of the Industry Mentor | Himdweep Walia |
| Name of the Institute | Amity University |

|  |  |  |
| --- | --- | --- |
| Date | Day # | Hours Spent |
| 28-05-2024 | Day-35 | 3.5 Hours |
| Activities done during the day:  Fitting the data to model using python.  **Data fitting:**  Python is a power tool for fitting data to any functional form. You are no longer limited to the simple linear or polynominal functions you could fit in a spreadsheet program. You can also calculate the standard error for any parameter in a functional fit.  **The basic steps to fitting data are:**   * Import the curve\_fit function from scipy. * Create a list or numpy array of your independent variable (your x values). You might read this data in from another source, like a CSV file. * Create a list of numpy array of your depedent variables (your y values). You might read this data in from another source, like a CSV file. * Create a function for the equation you want to fit. The function should accept as inputs the independent variable(s) and all the parameters to be fit. * Use the function curve\_fit to fit your data. * Extract the fit parameters from the output of curve\_fit. * Use your function to calculate y values using your fit model to see how well your model fits the data. * Graph your original data and the fit equation.   **Fitting x, y Data**  First, import the relevant python modules that will be used.   |  | | --- | | import numpy as np  import matplotlib.pyplot as plt  from scipy.optimize import curve\_fit |   Now we will consider a set of x,y-data. This data has one independent variable (our x values) and one dependent variable (our y values). We will recast the data as numpy arrays, so we can use numpy features when we are evaluating our data. It is often very helpful to look at a plot of the data when deciding what functional form to fit.   |  | | --- | | X\_data = [ -10.0, -9.0, -8.0, -7.0, -6.0, -5.0, -4.0, -3.0, -2.0, -1.0, 0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]  Y\_data = [1.2, 4.2, 6.7, 8.3, 10.6, 11.7, 13.5, 14.5, 15.7, 16.1, 16.6, 16.0, 15.4, 14.4, 14.2, 12.7, 10.3, 8.6, 6.1, 3.9, 2.1]  #Recast x\_data and y\_data into numpy arrays so we can use their handy features  X\_data = np.asarray(x\_data)  Y\_data = np.asarray(y\_data)  plt.plot(x\_data, y\_data, 'o') |   This data could probably be fit to many functional forms. We will try two different functional forms. (Looking at data and knowing what function it might fit is non-trivial and beyond the scope of this lesson. For purposes of this lesson, we will simply fit the data to given functional forms.)  IMG_256  Reference:  <https://milliams.com/courses/applied_data_analysis/Fitting.html/> | | |
|  | | |