# Take-Home Coding Assessment: L2 Assistant Data Scientist

This programming assessment is designed to take less than 3 hours and has three short parts: applying a regression model, implementing a simple edit-distance algorithm, and working with text data.

Please return the following materials for review in a zip file to [datascience.air@gmail.com](mailto:datascience.air@gmail.com):

1. Python or R code file[[1]](#footnote-1)
2. Written documentation and visualizations (where required)

You **MUST WRITE CODE** for all three section (so do not do anything by hand) and are encouraged to use existing libraries and packages (particularly for part 1).

Make sure that the code you turn is clean and well-documented and includes everything needed to run it (including import statements for packages or libraries that you use).

## 1. Regression Modelling

For this exercise, we are interested in better understanding the shapes of iris flowers. Specifically, we are interested in whether the petal length and sepal length are related. We will use the “iris” data set which is available in both R and Python (and also attached as a csv, “Iris\_Data.csv”) which includes the petal and sepal lengths and widths and the species of iris to which each example belongs.

1. How many irises belong to each species?
2. Make a scatterplot of petal length vs sepal length. Color the dots according to species. Document your observations (2-3 sentences)
3. Fit a regression model predicting sepal length based on petal length, petal width and sepal width (you do not need to test any of the regression assumptions).
4. Describe the results of your regression, focusing on the relationship between sepal length and petal length.
5. **Extra Credit**: Fit a regression model predicting sepal length based on petal length, petal width, sepal width and species (you do not need to test for any of the “classical” regression assumptions). This is the same as part c but also with species as a predictor. Describe the results.

## 2. Implementing an Edit-Distance Algorithm

Write a program to calculate a variant of the Hamming distance with two key modifications to the standard algorithm. In information theory, the Hamming distance is a measure of the distance between two text strings. This is calculated by adding one to the Hamming distance for each character that is different between the two strings. For example, “**k**itten" and “**m**itten" have a Hamming distance of 1. See <https://en.wikipedia.org/wiki/Hamming_distance> for more information.

Modifications to the standard Hamming distance algorithm for the purposes of this exercise include:

1. Add .5 to the Hamming distance if a capital letter is switched for a lower case letter unless it is in the first position. Examples include:
   1. "Kitten" and "kitten" have a distance of **0**
   2. "kitten" and "Ki**T**ten" have a Hamming distance of .5.
   3. "Puppy" and "P**O**ppy" have a distance of 1.5 (1 for the different letter, additional .5 for the different capitalization).
2. Consider S and Z (and s and z) to be the same letter. For example, "analyze" has a distance of 0 from "analyse".

Test cases with expected outputs:

|  |  |  |
| --- | --- | --- |
| **First Word** | **Second Word** | **Distance Score** |
| make | Mage | 1 |
| MaiSY | MaiZy | .5 |
| Eagle | Eager | 2 |
| Sentences work too | Sentences wAke too | 3.5 |

Use the program you wrote to score the following strings:

1. "data Science" to "Data Sciency"
2. "organizing" to "orGanising"
3. "AGPRklafsdyweIllIIgEnXuTggzF" to "AgpRkliFZdiweIllIIgENXUTygSF")

Then:

1. Describe a scenario (3-4 sentences) where implementing the standard Hamming distance algorithm would be applicable.

## 3. Data Cleaning

Perform some data cleaning using the provided file, “patent\_drawing.csv”. “Patent\_drawing.csv” contains a list of patents and a short description of each drawing included with a patent grant. For example, patent number 0233365 (<https://patents.google.com/patent/US20030233365A1/en>) has 16 images. For each image, there is a brief description of the drawings. The description is included in the “text” field in patent\_drawing.csv.

Let’s say that we are interested in understanding:

1. How many of the field descriptions reference a perspective that is not standard (i.e. viewed from the top, bottom, front or rear)? Specifically, write code to count how many of the rows have the words "view" or "perspective" but do not include "bottom", "top", "front" or "rear" in the text field?
2. What is the average number of drawing descriptions per patent?

1. You must answer all questions programmatically and turn in your code. You should not do anything by hand. [↑](#footnote-ref-1)