Weather Prediction

Requirements and Test Document

Simardeep Khinda

04/07/2023

CS 225, Spring 2023

Embry-Riddle Aeronautical University

Daytona Beach campus

1 Aerospace Boulevard

Daytona Beach, FL 32114

**Introduction:**

The purpose of this document is to review the requirements for the development of the weather forecaster. The weather forecaster uses 2 different algorithms to attempt a prediction for the future weather. The input is a minimum and maximum temperature for each day for X number of days. The more data that is provided, the more accurate the data will likely be. Similarly, not enough data may lead to inaccuracies for the predicted data. For more information on the functionality of the weather forecaster, please refer to the SDD (Software Design Document).

**Background information:**

To complete the requirements for Sprint 1, the only essentiality is that the code shall compile. Currently, there is no data being trained or predicted as that will be developed in future sprints. As of right now, the class FileManager is used to begin reading from a file so the software can use the read data to calculate and predict future data. For User Story 1, there will be no test cases as the only method to make sure the user story has been met is for the software to compile without errors. Unlike other user stories, there is only one possible requirement and therefore, there cannot be 3 different test cases to verify the succession.

For Sprint 2, the data that is read from the CSV file is now able to be passed to each model for their respective training and predictions. For this sprint, the models, linear regression, and polynomial regression have been complete. As described more thoroughly in the SDD, linear regression looks for a linear trend and bases its’ predictions on those. This is implemented into the program using the formula where **m** is the slope and **b** is the y-intercept. Polynomial regression works similarly but instead of finding a linear trend, it aims to identify a non-linear trend. In summary, polynomial regression does almost the same thing as linear regression but it plants a curved line rather than a linear line to train the model and make the prediction.

**Requirements:**

The requirements are listed below. These requirements have been created to assure that the user stories created earlier (available in the ATS) are successfully met. Regardless of how the software is constructed, if the requirements are met, that will lead to the user stories being fulfilled which in return, provides a software that works with the expected functionality. The requirements are provided below in Table 1.

**Table 1: Requirement Specifications**

|  |  |
| --- | --- |
| **ID** | **Requirement Specification** |
| 1 | As a developer, I want to be able to test my software. |
| 1.1 - The software shall compile without errors. |
| 2 | As a developer, I want to be able to receive user input. |
| 2.1 – There shall be a method in FileManager to read data. |
| 3 | As a developer, I want to be able to pass the user input to each model to predict data. |
| 3.1 – The data shall be readable in the prediction model classes. |
| 4 | As a developer, I want to be able to predict data using polynomial regression. |
| 4.1 – The software shall be able to train and predict data using polynomial regression. |
| 5 | As a developer, I want to be able to predict data using linear regression. |
| 5.1 - The software shall be able to train and predict data using linear regression. |

**Test Cases:**

The test cases below have been formed from the requirements above. If all test cases are to pass, then software is working as expected. There should not be any different behavior that can be called successful. The actual output should match the expected output provided below in Table 2.

Table 2: Test Cases and Results

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req’t**  **ID** | **Test**  **Case**  **ID** | **Initial**  **Conditions**  **And Input** | **Expected Behavior**  **Or Output** | **Actual**  **Behavior**  **Or Output** | **Pass**  **Fail** |
| 2.1 | 2.1.1 | The header of the CSV file (input.csv) shall be “Day, Minimum, Maximum” | “Day, Minimum, Maximum” |  |  |
| 2.1 | 2.1.2 | The row beneath the header shall be “1, 99, 100” | “Day 1| Min: 99| Max: 100” |  |  |
| 2.1 | 2.1.3 | The row beneath header shall be “1, 99” | IndexOutOfBoundsException  “Index 2 out of bounds for length 2” |  |  |
| 3.1 | 3.1.1 | For just one day, set minimum to 0 and maximum to 99. Print old Min and Max data in Controller class. | “Minimum: [0]  Maximum:[99]” |  |  |
| 3.1 | 3.1.2 | For just one day, set minimum to 0 and maximum to 99. Print old Min and Max data in LinearRegression class. | “LR Data: [0]  LR Data: [99]” |  |  |
| 3.1 | 3.1.3 | For just one day, set minimum to 0 and maximum to 99. Print old Min and Max data in PolynomialRegression class. | “PR Data: [0]  PR Data: [99]” |  |  |
| 4.1 | 4.1.1 | Input this to input.csv: “Day,Minimum,Maximum  1,0,0  2,0,0  3,0,0”  Print Polynomial Regression results. | PR Data: [0.0, 0.0, 0.0]  Predicted data using PR: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,0.0]  PR Data: [0.0, 0.0, 0.0]  Predicted data using PR: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |  |  |
| 4.1 | 4.1.2 | Input this to input.csv: “Day,Minimum,Maximum  1,0,1  2,1,2  3,2,3”  Print Polynomial Regression results. | PR Data: [0.0, 1.0, 2.0]  Polynomial: [3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0]  PR Data: [1.0, 2.0, 3.0]  Polynomial: [4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0] |  |  |
| 4.1 | 4.1.3 | Input this to input.csv: “Day,Minimum,Maximum  1,30,40  2,35,41  3,32,39.5”  Print Polynomial Regression results. | PR Data: [30.0, 35.0, 32.0]  Polynomial: [49.0, 55.33, 61.67, 68.0, 74.33, 80.67, 87.0, 93.33, 99.67, 106.0]  PR Data: [40.0, 41.0, 39.5]  Polynomial: [44.25, 45.67, 47.08, 48.5, 49.92, 51.33, 52.75, 54.17, 55.58, 57.0] |  |  |
| 5.1 | 5.1.1 | Input this to input.csv: “Day,Minimum,Maximum  1,0,0  2,0,0  3,0,0”  Print Linear Regression results. | LR Data: [0.0, 0.0, 0.0]  Predicted data using LR: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]  LR Data: [0.0, 0.0, 0.0]  Predicted data using LR: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0] |  |  |
|  |  | Input this to input.csv: “Day,Minimum,Maximum  1,0,1  2,1,2  3,2,3”  Print Linear Regression results. | “LR Data: [0.0, 1.0, 2.0]  Predicted data using LR: [3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0]  LR Data: [1.0, 2.0, 3.0]  Predicted data using LR: [4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0]” |  |  |
|  |  | Input this to input.csv: “Day,Minimum,Maximum  1,30,40  2,35,41  3,32,39.5”  Print Linear Regression results. | LR Data: [30.0, 35.0, 32.0]  Predicted data using LR: [34.33, 35.33, 36.33, 37.33, 38.33, 39.33, 40.33, 41.33, 42.33, 43.33]  LR Data: [40.0, 41.0, 39.5]  Predicted data using LR: [39.67, 39.42, 39.17, 38.92, 38.67, 38.42, 38.17, 37.92, 37.67, 37.42] |  |  |

**References:**

All sources cited previous sections are listed in this section. If the project required no sources, keep this section but leave it blank. Sources might be papers and texts in the general problem domain of the project, code snippets, libraries incorporated in the project, or even algorithmic solutions to specific parts of the project.

* <https://www.mathsisfun.com/data/least-squares-regression.html>
* https://www.analyticsvidhya.com/blog/2021/07/all-you-need-to-know-about-polynomial-regression/
* https://commons.apache.org/proper/commons-math/javadocs/api-3.6.1/org/apache/commons/math3/fitting/WeightedObservedPoints.html
* https://commons.apache.org/proper/commons-math/userguide/fitting.html

**Appendices:**

This is optional, but may include external sources, source code, input data files, or other related material.