# ANN Implementation

## Implementation of the MLP Algorithm

By implementing the multi-layer perception algorithm, I can construct and train an artificial neural network which can estimate the index flood of an area, given several parameters as input values to the model.

## Data Pre-processing

The supplied data has the following columns of data available:

* Input parameters
  + AREA - Catchment area in km2
  + BFIHOST - Base flow index
  + FARL - Flood attenuation due to reservoirs and lakes
  + FPEXT - Flood plain extent
  + LDP - Longest drainage path
  + PROPWET - Proportion of wet days
  + RMED-1D - Median annual maximum 1-day rainfall
  + SAAR - Standard annual average rainfall
* Predictand
  + Index Flood - Median of the annual maximum series of catchment flow in m3/s

### Data Cleaning

Before parsing the input data to the network, I developed several stages in MS Excel to check the data for anomalous or erroneous values and select which data should be used in the model.

1. Create a correlation table of each input parameter against the flood index.

This step is designed to highlight which columns should be considered as useful input parameters. It utilises the CORREL function in MS Excel and shows if and how much each column correlates with the index flood.

The (absolute) output from this step is shown in table X.

|  |  |
| --- | --- |
| **Input Parameter** | **Correlation to Index Flood (%)** |
| AREA | 72.1% |
| BFIHOST | 29.4% |
| FARL | 2.1% |
| FPEXT | 2.1% |
| LDP | 66.4% |
| PROPWET | 40.4% |
| RMED-1D | 18.0% |
| SAAR | 23.9% |

This step clearly highlights two columns which have a low correlation rate; FARL and FPEXT. This would suggest that these two columns could be removed from the data set to improve the accuracy that can be achieved with the training algorithm, thus both columns were removed from the data set.

1. Create a validation matrix for each column of data.

This step is designed to highlight any statistical errors that exist in the data set. The matrix has been implemented by utilising the following measures:

**Measure**: Is value numerical?  
**Formula**: =SUMPRODUCT(ISNUMBER(*[Column]*) + 0) < (COUNTA(*[Column]*)-1)**Description**: This formula will check if each cell in the column of data is numeric, returning a Boolean array. The array is then summed and checked against the number of elements in the column, returning TRUE if there exist any cells which are not numeric or FALSE if all cells are numeric.

**Measure**: Is value textual?  
**Formula**: =SUMPRODUCT(ISTEXT([*Column]*) + 0) > 0  
**Description**: This formula will check if each cell in the column of data is textual, returning a Boolean array. The array is then summed, returning TRUE if there exist any cells which contain text or FALSE if no textual cells are found.

**Measure**: Is value blank?  
**Formula**: =SUMPRODUCT(ISBLANK([*Column]*) + 0) > 0  
**Description**: This formula will check if each cell in the column of data is empty, returning a Boolean array. The array is then summed, returning TRUE if there exist any cells which are empty or FALSE if no empty cells are found.

**Measure**: Is value negative?  
**Formula**: =IF(COUNTIF([*Column]*,"<0") > 0, TRUE, FALSE)  
**Description**: This formula will check if each cell in the column of data contains a negative value, returning a Boolean array. The array is then summed, returning TRUE if there exist any cells which are negative or FALSE if no negative cells are found.

Applying these conditions on each column created the validation matrix show in table X.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **AREA** | **BFIHOST** | **LDP** | **PROPWET** | **RMED-1D** | **SAAR** | **Index flood** |
| **Numeric** | FALSE | FALSE | FALSE | TRUE | FALSE | TRUE | FALSE |
| **Textual** | FALSE | FALSE | FALSE | TRUE \*2 | FALSE | FALSE | FALSE |
| **Blank** | FALSE | FALSE | FALSE | TRUE \*5 | FALSE | TRUE \*3 | FALSE |
| **Negative** | TRUE \*1 | FALSE | FALSE | FALSE | FALSE | FALSE | TRUE \*4 |
| **Valid?** | **Failed** | **Passed** | **Passed** | **Failed** | **Passed** | **Failed** | **Failed** |

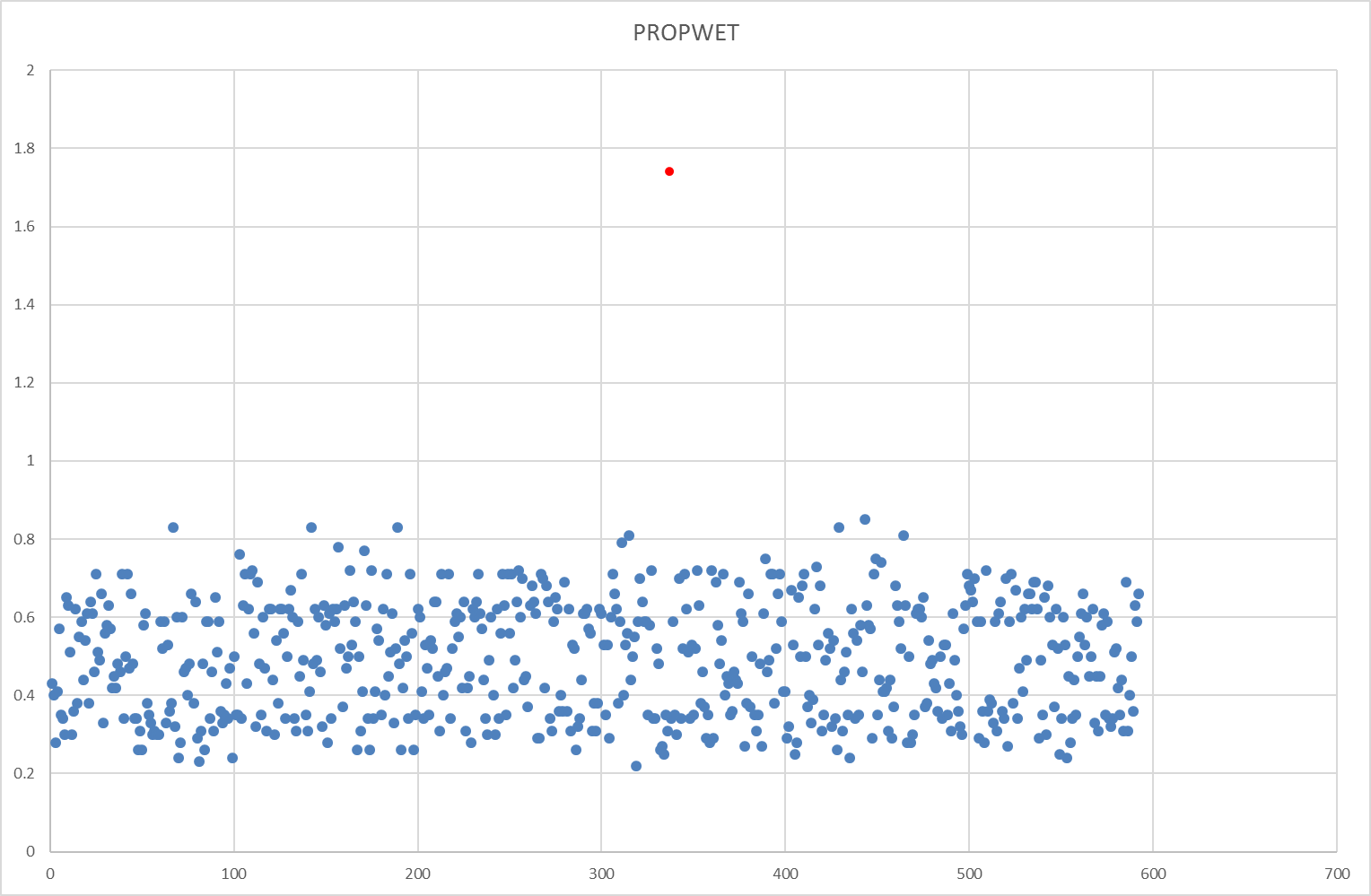
Following the creation of this matrix, I manually searched out the erroneous values until each cell in the valid row reported that it had passed the validation. My manual method involved sorting suspect columns that reported an error so the error would float to either the top or the bottom of the column.

Each row that contained an erroneous value was removed from the data set as it was not possible to extrapolate any useful information from the row unless the entire row is complete and valid. This method highlighted the following rows:

* \*1 - Row 79: ‘-999’ value for AREA
* \*2 - Row 114: ‘a’ value for PROPWET
* \*3 - Row 538: Missing value for SAAR
* \*4 - Row 548: ‘-999’ value for Index flood
* \*5 - Row 587: Missing value for PROPWET

1. Plot a scatter graph for each column of data.

This step is designed to highlight any outliers that exist in the data. One possible outliers was identified through this method.



The point highlighted in red were identified as a potential outlier in the PROPWET column. Research into the column revealed that values should fall between 0.2 and 0.8. The rest of the data set seems to fit this range with a few extremes existing, but none out of the range as far as this point. Therefore, I decided that the point was anomalous and removed the row of data from the data set.

1. Remove trends

This step is designed to minimise the standard deviation of each column of data to dampen the effects of any trends and seasonal components in the data. This was done by sorting each column of data and calculating whether each data point was within 10% of another data point. An example using a sample from the Area column is shown in table X for which a value would need to be within 458.534 of the next value for it to lie within 10%.

|  |  |  |
| --- | --- | --- |
| **Area** | **First Difference** | **Within 10%?** |
| 4586.97 | 188.31 | True |
| 4398.66 | 68.83 | True |
| 4329.83 | 313.41 | True |
| 4016.42 | 536.41 | False |
| 3480.01 | 134.27 | True |
| 3345.74 | 44.74 | True |
| 3300.8 | 89.69 | True |
| 3211.11 | 358.71 | True |

In this case, the rows of data where the Area is 4016.42 and above were truncated from the data set. This results in a lower standard deviation for this column, dropping from 562.33 to 456.58.

This method results in the following rows of data being removed from the data set:

* Row 572: Area and Index flood high
* Row 139: Area and Index flood high
* Row 22: Area and LDP high
* Row 297: Area and LDP high
* Row 407: RMED-1D high
* Row 126: RMED-1D high
* Row 12: Index flood high
* Row 529: Index flood high
* Row 297: Index flood high
* Row 501: RMED-1D low
* Row 515: RMED-1D low
* Row 171: RMED-1D low
* Row 143: RMED-1D low

### Data Standardisation

As the input parameters represent different physical quantities and the numeric values have different scales of magnitude, I standardised the data before training the neural network.

The data was standardised to between 0.1 and 0.9 using the following formula:

Where *S* is the standardised value, *R* is the raw value and Min and max are the minimum and maximum values over the column of data.