Summary SeoulBikeData

Methodology

* The input variable of “Functional Day” was eliminated due to every instance being ‘Fun’ and “Date” was eliminated.
* For “Seasons” variables, each season was given a value (1,2,3,4) for Winter, Spring, Summer, Autumn. For “Holiday”, the no or yes response was changed to 1 and 2 respectively. As well “Hour”, “Solar Radiation”, “Rainfall” and “Snowfall” all had a numerical one added to each instance in order to eliminate zeros for proper use of some of the models.
* The original data file was edited into “Bikes.csv” to be read into Scala for easier access to the dataframe. Changes include: removing column headers, moving response variable to the last column, and the changes mentioned above and in the Report.
* In Python, the package of Keras used to build and run the Neural Nets.
* Both in Scala and Python, mean squared error (MSE) was used as the loss function.
* In Python, the Adam optimizer was used. In Scala, the Optimizer Object was used. Both languages the batch size was set to 32.
* In Python a preliminary run of each Neural Net was done with epochs set at 500 to find the most optimal epoch range for a given activation function and learning rate. This was determined by finding the lowest MSE value at a given epoch. An example is shown below.
* In Scala, the epochs were set to 200 as the findings from the Python version of the code showed 200 obtained a reasonable minimum for each neural net.
* In Python, the number of units in each layer was set to 24 while in Scala, Scalation used a formula to determine the number of units in each layer.
* In both languages, the activation functions of sigmoid, reLU and eLU were used at 0.001,0.01 and 0.1 learning rates.
* In both languages, a comparison was made between performing Forward Feature selection on the model prior to training and testing the Neural Net or directly feeding the data set into the Neural Net.

Feature Indexes:

* 0 - Hour: Hour of the Day
* 1 - Temperature: Celsius
* 2 - Humidity: Percent
* 3 - Wind Speed: m/s
* 4 - Visibility: 10 m
* 5 - Dew point Temperature: Celsius
* 6 - Solar Radiation: MJ/m2
* 7 - Rainfall: mm
* 8 - Snowfall: cm
* 9 - Seasons: Winter, Spring, Summer Autumn
* 10 - Holiday: Holiday or No Holiday

**Table 1.** 2 Layer Neural Net with Forward Feature Selection in Scala

| Activation Fn | Learning Rate | Feature Index Selected | R2 Bar | R2 CV |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .501844 | 50.4362 |
| sigm | 0.01 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .360732 | 47.4264 |
| sigm | 0.001 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | -0.300891 | 17.9233 |
| relu | 0.1 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .505213 | 50.322 |
| relu | 0.01 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .502789 | 50.4079 |
| relu | 0.001 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .50296 | 50.3436 |
| elu | 0.1 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .514268 | 51.468 |
| elu | 0.01 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .515857 | 51.5715 |
| elu | 0.001 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .511207 | 51.3661 |

**Table 2.** 2 Layer Neural Net in Scala

| Activation Fn | Learning Rate | CV R2 | R2 Bar | MSE |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 0.502934 | 0.502309 | 0.0168841 |
| sigm | 0.01 | 0.368212 | 0.367417 | 0.02146 |
| sigm | 0.001 | -0.00844695 | -0.009715 | 0.0342545 |
| relu | 0.1 | 0.498375 | 0.497745 | 215460 |
| relu | 0.01 | 0.503742 | 0.503118 | 213155 |
| relu | 0.001 | 0.506537 | 0.505917 | 211954 |
| elu | 0.1 | 0.519847 | 0.519243 | 206238 |
| elu | 0.01 | 0.520959 | 0.520357 | 205760 |
| elu | 0.001 | 0.515101 | 0.514491 | 208276 |

**Table 3**. 3 Layer Neural Net with Forward Selection in Scala

| Activation Fn | Learning Rate | Feature Index Selected | CV R2 | R2 Bar |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | -0.303067 | -0.32576 |
| sigm | 0.01 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | -0.556089 | 23.3939 |
| sigm | 0.001 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .61241 | 61.017 |
| relu | 0.1 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | -0.102869 | -0.338607 |
| relu | 0.001 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .444303 | 44.4209 |
| relu | 0.0005 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .511345 | 51.2737 |
| elu | 0.001 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .596135 | 59.6135 |
| elu | 0.0005 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .552576 | 55.2108 |
| elu | 0.0001 | 0, 10, 1, 2, 9, 6, 7, 5, 3, 4, 8 | .559078 | 55.5521 |

**Table 4**. 3 Layer Neural Net in Scala

| Activation Fn | Learning Rate | CV R2 | R2 Bar | MSE |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | -0.00253855 | -0.00379903 | 430615 |
| sigm | 0.01 | -0.00396020 | -0.00522246 | 431226 |
| sigm | 0.001 | 0.555405 | 0.554846 | 190964 |
| relu | 0.1 | -0.00251909 | -0.00377954 | 430607 |
| relu | 0.001 | 0.50016 | 0.499531 | 214694 |
| relu | 0.0005 | 0.504961 | 0.504339 | 212631 |
| elu | 0.001 | 0.545773 | 0.545202 | 195102 |
| elu | 0.0005 | 0.563219 | 0.562669 | 187608 |
| elu | 0.0001 | 0.560156 | 0.559603 | 188924 |

**Table 5**. 4 Layer Neural Net with Forward Selection in Scala

| Activation Fn | Learning Rate | Cols | CV R2 | R2 Bar |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 0, 1, 2, 9, 5, 6, 7, 3, 8, 10, 4 | .500562 | 54.1981 |
| sigm | 0.01 | 0, 2, 8, 7, 9, 10, 1, 6, 3, 4, 5 | -.0266937 | 33.2825 |
| sigm | 0.001 | 0, 2, 10, 8, 7, 3, 4, 6, 5, 1, 9 | -.0745524 | -6.16859 |
| relu | 0.001 | 0, 1, 7, 5, 9, 6, 3, 8, 2, 4, 10 | .513462 | NaN |
| relu | 0.0005 | 0, 1, 7, 5, 9, 6, 3, 8, 2, 4, 10 | .473907 | 46.9689 |
| relu | 0.0001 | 0, 1, 5, 8, 4, 7, 2, 3, 10, 6, 9 | .634369 | 63.5974 |
| elu | 0.01 | 0, 7, 4, 5, 6, 2, 9, 1, 3, 8, 10 | -1.19965 | -119.785 |
| elu | 0.001 | 0, 1, 2, 3, 8, 10, 5, 9, 7, 6, 4 | .519093 | 54.9044 |
| elu | 0.0005 | 0, 1, 7, 4, 8, 2, 9, 5, 6, 3, 10 | .557576 | 55.2775 |

**Table 6**. 4 Layer Neural Net in Scala

| Activation Fn | Learning Rate | CV R2 | R2 Bar | MSE |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 0.501259 | 0.500632 | 0.016941 |
| sigm | 0.01 | -0.0355665 | -0.0368685 | 0.0351757 |
| sigm | 0.001 | -0.0778174 | -0.0791726 | 0.0366109 |
| relu | 0.001 | 0.508972 | 0.508355 | 210909 |
| relu | 0.0005 | 0.441828 | 0.441126 | 239749 |
| relu | 0.0001 | 0.495275 | 0.49464 | 216792 |
| elu | 0.01 | 0.241332 | 0.240378 | 325867 |
| elu | 0.001 | 0.67902 | 0.67944 | 137515 |
| elu | 0.0005 | 0.647423 | 0.64698 | 151441 |

**Table 7**. 2 Layer Neural Net with Forward Selection in Python

| Activation Fn | Learning Rate | Feature Index Selection | CV R2 | R2 Bar |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 4, 7,10,11,13 | 0.453744 | 0.455087 |
| sigm | 0.01 | 4, 7,10,11,13 | 0.454573 | 0.452701 |
| sigm | 0.001 | 4, 7,10,11,13 | 0.446831 | 0.439162 |
| relu | 0.1 | 4, 7,10,11,13 | 0.458654 | 0.466091 |
| relu | 0.01 | 4, 7,10,11,13 | 0.454473 | 0.457288 |
| relu | 0.001 | 4, 7,10,11,13 | 0.447147 | 0.442085 |
| elu | 0.1 | 4, 7,10,11,13 | 0.458433 | 0.463272 |
| elu | 0.01 | 4, 7,10,11,13 | 0.454121 | 0.460203 |
| elu | 0.001 | 4, 7,10,11,13 | 0.446999 | 0.447517 |

**Table 8**. 2 Layer Neural Net in Python

| Activation Fn | Learning Rate | R2 | CV R2 | R2 Bar |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 0.452911 | 0.456469 | 0.452129 |
| sigm | 0.01 | 0.457368 | 0.459942 | 0.456593 |
| sigm | 0.001 | 0.411292 | 0.42054 | 0.41045 |
| relu | 0.1 | 0.459215 | 0.450429 | 0.458442 |
| relu | 0.01 | 0.461984 | 0.463735 | 0.461215 |
| relu | 0.001 | 0.448571 | 0.449919 | 0.447782 |
| elu | 0.1 | 0.45795 | 0.455129 | 0.457175 |
| elu | 0.01 | 0.460305 | 0.461319 | 0.459534 |
| elu | 0.001 | 0.4492 | 0.449487 | 0.448412 |

**Table 9**. 3 Layer Neural Net with Forward Selection in Python

| Activation Fn | Learning Rate | Feature Index Selection | CV R2 | R2 Bar |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 4, 7,10,11,13 | 0.528658 | 0.533327 |
| sigm | 0.01 | 4, 7,10,11,13 | 0.604328 | 0.608135 |
| sigm | 0.001 | 4, 7,10,11,13 | 0.554394 | 0.570927 |
| reLu | 0.1 | 4, 7,10,11,13 | 0.587736 | 0.605214 |
| reLu | 0.01 | 4, 7,10,11,13 | 0.641509 | 0.641436 |
| reLu | 0.001 | 4, 7,10,11,13 | 0.516563 | 0.517335 |
| elu | 0.1 | 4, 7,10,11,13 | 0.616315 | 0.635646 |
| elu | 0.01 | 4, 7,10,11,13 | 0.643841 | 0.664568 |
| elu | 0.001 | 4, 7,10,11,13 | 0.528907 | 0.539323 |

**Table 10.** 3 Layer Neural Net in Python

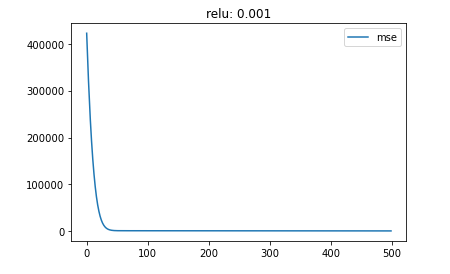
| Activation Fn | Learning Rate | R2 | CV R2 | R2 Bar |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 0 | -0.004442 | -0.005737 |
| sigm | 0.01 | 0.373401 | 0.360641 | 0.369806 |
| sigm | 0.001 | 0.423411 | 0.41102 | 0.420103 |
| relu | 0.1 | 0.509153 | 0.520797 | 0.506337 |
| relu | 0.01 | 0.631602 | 0.625762 | 0.629488 |
| relu | 0.001 | 0.517862 | 0.507335 | 0.515095 |
| elu | 0.1 | 0.549012 | 0.510165 | 0.546424 |
| elu | 0.01 | 0.655157 | 0.652716 | 0.653178 |
| elu | 0.001 | 0.5545 | 0.547766 | 0.551944 |

**Table 11**. 4 Layer Neural Net with Forward Selection in Python

| Activation Fn | Learning Rate | Feature Index Selection | CV R2 | R2 Bar |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 4, 7,10,11,13 | -0.001462 | -0.000571 |
| sigm | 0.01 | 4, 7,10,11,13 | 0.683007 | 0.684934 |
| sigm | 0.001 | 4, 7,10,11,13 | 0.650572 | 0.66785 |
| relu | 0.1 | 4, 7,10,11,13 | 0.592627 | 0.625404 |
| relu | 0.01 | 4, 7,10,11,13 | 0.669759 | 0.689212 |
| relu | 0.001 | 4, 7,10,11,13 | 0.621595 | 0.633178 |
| elu | 0.1 | 4, 7,10,11,13 | 0.608192 | 0.60617 |
| elu | 0.01 | 4, 7,10,11,13 | 0.67114 | 0.686083 |
| elu | 0.001 | 4, 7,10,11,13 | 0.613021 | 0.610114 |

**Table 12**. 4 Layer Neural Net in Python

| Activation Fn | Learning Rate | R2 | CV R2 | R2 Bar |
| --- | --- | --- | --- | --- |
| sigm | 0.1 | 0 | -0.002266 | -0.00143 |
| sigm | 0.01 | 0.384433 | 0.403236 | 0.383553 |
| sigm | 0.001 | 0.560498 | 0.55331 | 0.55987 |
| relu | 0.1 | 0.420477 | 0.339963 | 0.419649 |
| relu | 0.01 | 0.623527 | 0.618848 | 0.622989 |
| relu | 0.001 | 0.66664 | 0.613223 | 0.666163 |
| elu | 0.1 | 0.503256 | 0.504111 | 0.502546 |
| elu | 0.01 | 0.648599 | 0.615523 | 0.648097 |
| elu | 0.001 | 0.670766 | 0.676281 | 0.670295 |



**Figure 1**: An example of seeing how many epochs it takes to minimize the MSE (the loss function used in each model). With the default number of epochs set at 500, this example shows that for the reLU activation function and 0.001 learning rate, the around 200 epochs is where the minimization in loss becomes stagnant.