AMERICAN UNIVERSITY OF BEIRUT

FACULTY OF ARTS & SCIENCES

Department of Computer Science

CMPS 262 – Data Science
Assignment 3 – Model Fitting & Improvement
December 03rd, 2023

By – Ralph Mouawad,
Alice Karadjian & Lea Bou Sleiman

To – Dr. Fatima Abu Salem

Please Refer to the Python Colab file for the codes.

Introduction:

First, we took the imputed dataset that was cleaned and prepared in assignment 2, we removed the columns that were casual rented bikes and registered rented bikes and kept only the total number of bikes rented because this is what we care about the most, and it'll be simpler to work with. Also note that our data is already normalized so we can begin our work immediately.

Part I:

Question a -

Model	Hyperparameter	Justification
Linear Regression	None	Linear Regression is a simple model that tries to find a weight for each input of the feature vector. These weights are computed in a way that minimizes a certain error (MSE, RMSE...) by using optimization algorithms such as Gradient or Stochastic Gradient Descent (faster, evaluating gradient at only one data point). No hyperparameters are used in this case.
Regression Tree	The maximum depth of the tree, as well as min samples split and min samples leaf	We can control the maximum depth of regression trees. The deeper the regression tree is, the more complex relation it can detect in our feature vector. For min samples split, it controls the limit for splitting the node (if the sample is less than the min required, there won't be splitting, and that node will be a terminal leaf). As for min samples leaf, if creating a split results in a leaf node with fewer samples, the split is not allowed. This hyperparameter helps control the size of the leaves and can prevent overfitting. The problem with complex regression trees is that it can suffer from Overfitting and fail on the testing data set. This is why we added these hyperparameters to try to avoid overfitting. We

		can still add additional hyperparameters, but it'll be too costly to compute results
Random Forest	Number of trees in the forest, min sample split and min sample leaf	We can control the number of trees in the random forest to find the best relations between the feature vectors of x. It is better to start with a moderate number of trees and then tune the model. As for min sample split and min sample leaf, they have the same impact as in the decision tree.
Neural Network	Learning Rate, Number of layers & neurons, & batch size	The learning rate is the step-size used during the optimization process with Stochastic Gradient Descent (or another optimization algorithm). If it is very small, it will take a lot of time to reach a solution and can get stuck. On the other hand, if it's very high, it might diverge. The number of neurons & layers will help catch the complex relations in our network. Finally, the batch size is here to adjust on how many samples we want to evaluate the Gradient. We will adjust these parameters until we reach the best results.

Question b -

Please refer to the Python code on Google Collab. We also computed the standard deviation of average scores, which is in our case for the RMSE. In the decision tree regressor, we got it to be 16. It is acceptable because there's not a high deviation between our RMSE scores at each validation. For random forest regressor, the std deviation was only 5.

Notes related to Neural Networks:
We used the Stochastic Gradient Descent algorithm, with 100 epochs and 4 layers, but we were getting very bad results. We replaced SGD with rmsprop algorithm and tuned the learning rate, added some layers, neurons and epochs. We got better results, for example R2 = 0.85 and RMSE = 769. There were some fluctuations in the decrease in error. We changed some of the parameters to obtain better results (see next answers). We tried to use grid search, but it was giving us an error, so we computed the results by trial & error.

The RMSE was not changing too much when tuning our hyperparameters, it was always between [710,760].

Question c -

Let's define each of the evaluation metrics computed in our regression case:

- R^2 error: It measures the proportion of the variance in the dependent variable explained by the model. It takes a value between [0,1]. It is a little bit like accuracy in classification, in the sense that the higher value we have, the better.
- RMSE: It's the average magnitude of errors between predicted and actual values. It measures the spread of errors, and we always want it to be smaller.
- MSE: Like RMSE but we didn't take the square root of the error.
- MAE: The MAE value represents the average absolute deviation between predicted and actual values.

In our case, we tried to search for the best hyperparameters once for the best R^2 and another time for the best RMSE. The results were very similar: When setting for the best RMSE, we were getting a similar value of R^2 as if it was assigned to be the scoring metric. We kept it as considering the scoring metric to be RMSE to deal with large errors. We're trying to predict a certain value, and we would like to obtain a predicted value not too different from the actual one. For neural network, we searched for the best MSE because it didn't run while searching for RMSE, but it shouldn't be a problem as we computed it after.

Question d -

We ensured that no data leakage is taking place in our pipeline by doing the following:

- Training & Testing splits: By splitting the data into training and testing, we're making sure to keep part of the data unseen to test our models on it.
- Cross-Validation: This is used by repeatedly splitting the data into training and validation. This is used to prevent Overfitting by randomly validating another split of the training data. We're also taking into consideration Cross-Validation while doing the Grid Search.

Our data has been correctly split into training and testing without including future information in the training dataset. Also, to ensure no data leakage, in the data cleaning & preparation part, one should be aware of not normalizing & working with the testing dataset to not get optimistic results. The testing data should remain untouched and unseen to not learn anything about it. This work should be done in the pre-processing phase (assignment 2).

Part II:

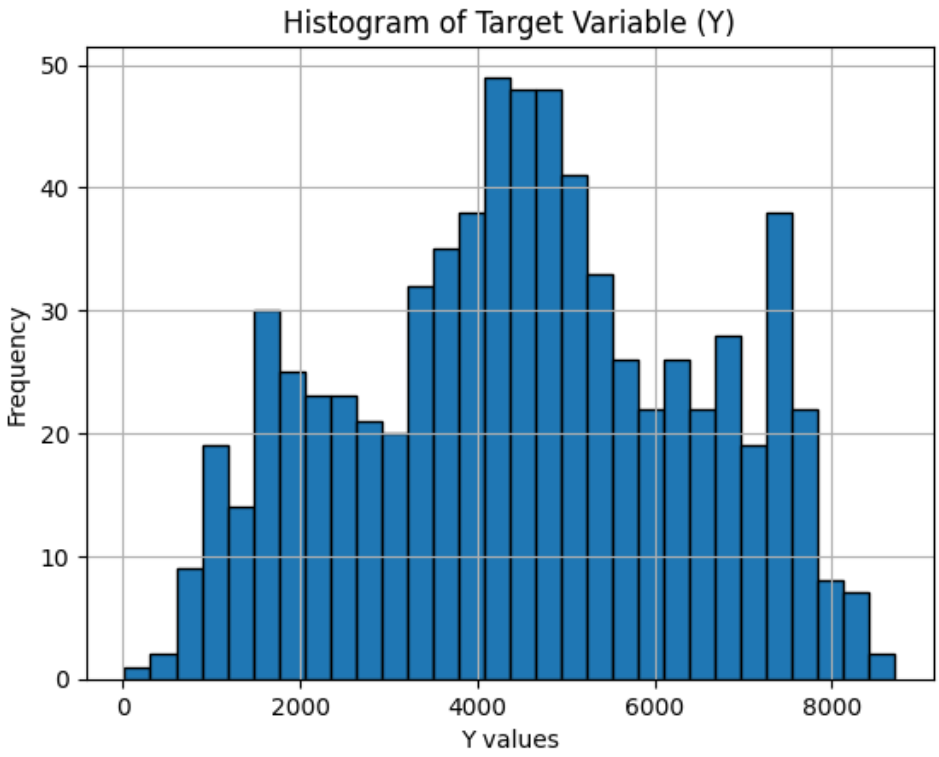
Question a -

We will analyze the improvement of the coefficient of determination between each model used on our dataset;

1- Linear Regression – Since this is a simple model that doesn't involve hyperparameters but only fitting a line to find a linear relation between our data points, R^2 wasn't at its best value. It was equal to 0.82765 (82.76%), which means that 82.76% of the variability of our dependent variable (here, number of bikes rented) has been explained by the independent variables in our regression model. This is relatively high, but we can still perform better by using more complex models.
2- Decision Tree Regressor – This model is more complex than Linear Regression, but here it didn't perform a lot better regarding the coefficient of determination, and it scored 0.83302 (83.3% of variability explained). But this model performed better regarding the Mean Absolute Error (560 vs 620) which indicates that this model can slightly improve our predictions.
3- Random Forest Regressor: Here, we're using a more complex "Ensemble" method which computes many decision trees and tries a lot of combinations of splitting... And this enabled us to obtain a better R^2 of 0.8841 (88.41% of variability explained). This is expected because we're using a more complex model, but at the expense of an expensive computation time: In fact, running the decision tree regressor would take few seconds, but the random forest regressor took several minutes because of its complexity.
4- Basic Feed Forward Neural Networks: We obtained an R2 = 0.87255 (87% of variability explained). Since our grid search didn't run, we tried to compute the best set hyperparameters & it gave us these results. With better tuning, we could achieve a higher R2. That answer was the best one we obtained; we then changed some hyperparameters for the plotting.

Question b -

Let's plot a histogram displaying our targeted value:



We can see that the distribution of our target variable is not 'very' normal, since we have some intervals that are rare outcomes. We can see that the number of bikes rented with the higher frequencies are in the interval [3500, 5500]. The rare outcomes are for the interval [1500,1750] and [7000, 7500]. Even though these intervals are not in the most common range, the fact that they still have high frequencies suggests that there might be significant and impactful events. Understanding and correctly predicting these rare outcomes could be crucial depending on the application. We can consider them as rare outcomes that are costly if underestimated.

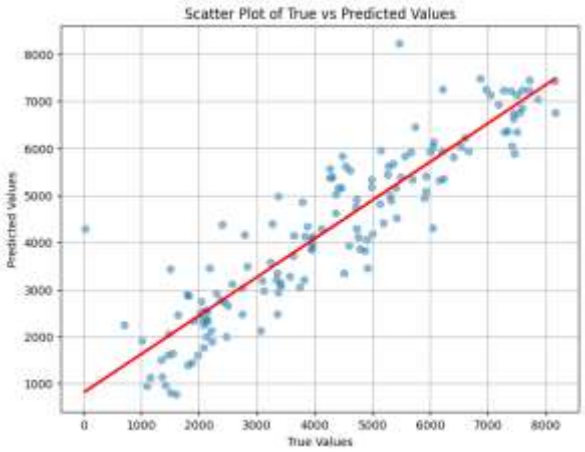
The average of the target variable (here the number of bikes rented) is 4505. Let's look at the Root Mean Squared Error and see its percentage in terms of the average.

1- Linear Regression: This simple model returned an RMSE of 832, meaning that on average we have an average error of 832 bikes while computing them. 832/4505 = 18.46% of error in the predicting values.
2- Decision Tree Regressor: The RMSE here is a little less than the one from Linear Regression: RMSE = 819. 819/4505 = 18.1 %

3- Random Forest Regressor: We got a better performance, with RMSE = 687 bikes. This decrease in the error is due to the complexity of our model that was able to catch more complex relationships between independent variables and dependent variables. 687/4505 = 15.2% This is better than before, and we can still try to find a better set of hyperparameters that would minimize this error and let us get better predictions.
4- Basic Feed Forward Neural Networks: RMSE = 702. 764/4505 = 15.7% of the average target variable. This was the best error that we got.

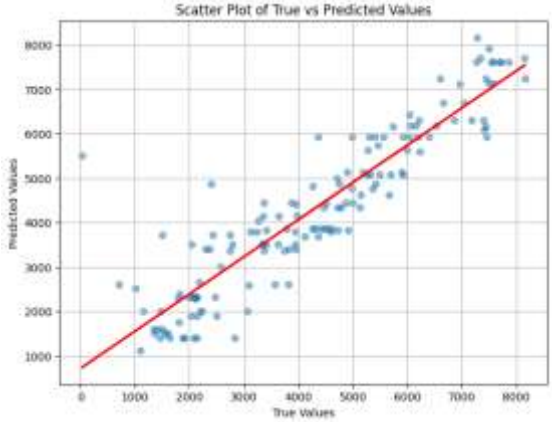
Question c -

1- Linear Regression – slope = 0.81



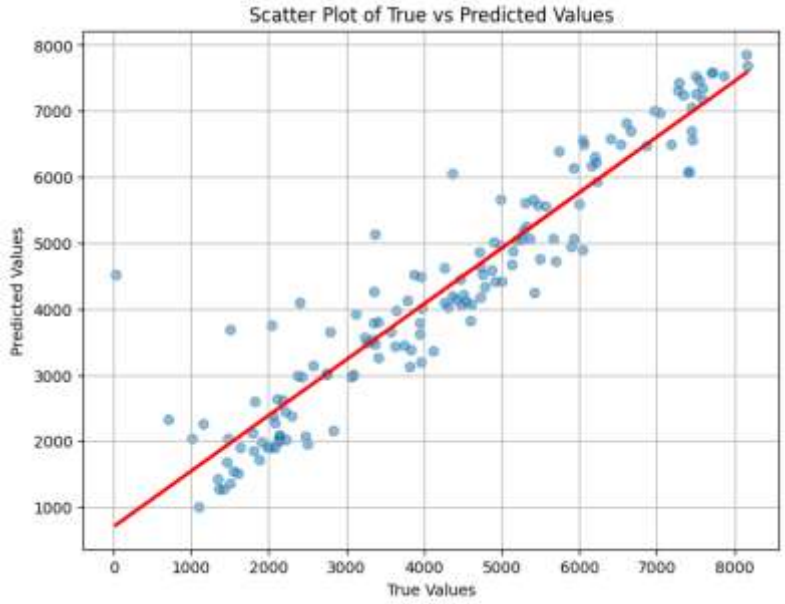
We can see that the line fitting the datapoints doesn't capture them all. The points are not very close to the line.

2- Decision Tree Regressor: slope = 0.836. The plot is:



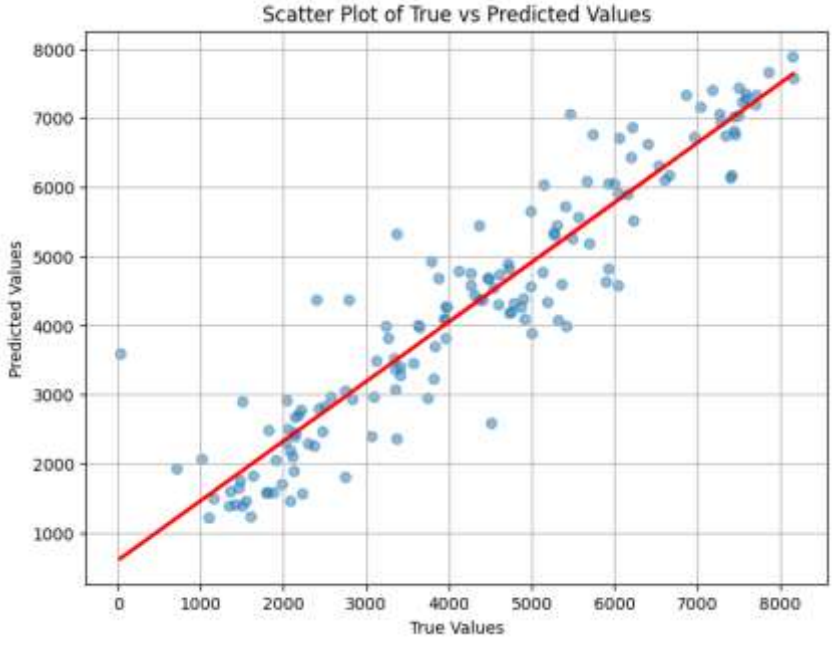
The points are closer to the line, and the true values are a little bit closer to the predicted values.

3- Random Forest: slope = 0.844. The plot:



The line fits most of the points, in a better way than the others. The slope is also a little greater than before.

4- Neural Networks:



The slope here is 0.863. Some points are not very close to the line, but it is still better than linear regression and decision tree.

The slope essentially quantifies the degree and direction of the linear association between the two variables. When the slope approaches 1, it means that the predicted and real values are closer to each other. One common thing that is anomalous is the predicted value of 4200 while the true value is 0. There should be an anomaly while testing because we don't think there's a 0 value in the dataset.

The closer the points are to the line, the better the approximation of predicted values is. Here, random forest is the best one.

Question d:

Pearson's Correlation - It is a measure of the strength and direction of a linear relationship between two variables. It ranges from -1 to 1, where:

1: Perfect positive linear correlation

−1: Perfect negative linear correlation

0: No linear correlation

In the context of making predictions, a high positive correlation indicates that the model's predictions are in line with the actual data, while a high negative correlation suggests an inverse relationship.

Let's see the correlations in our models:

1- Linear Regression: 0.9099
2- Decision Tree Regressor: 0.9128 (better than before)
3- Random Forest: 0.9405
4- Neural Network: 0.9368

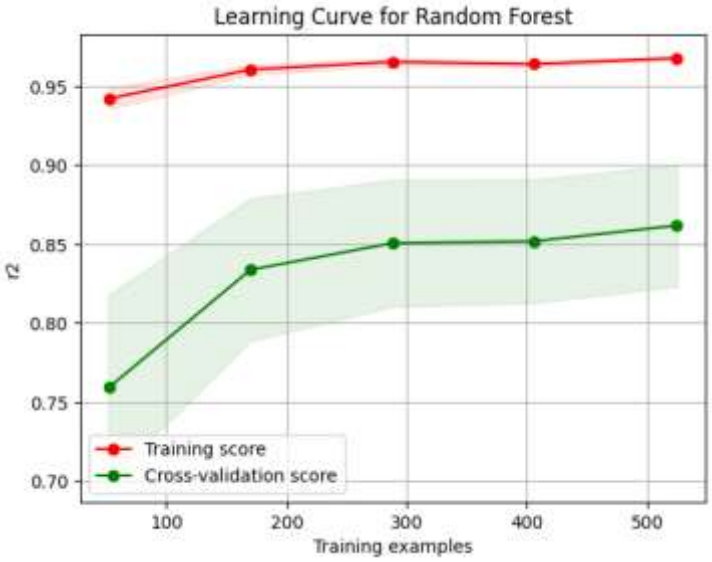
We can see that the random forest and neural network gave the best correlations because they were able to return better predicted values. This is mainly because of their complex structure that was able to capture better relations.

Question e -

Learning Curve for Decision Tree:

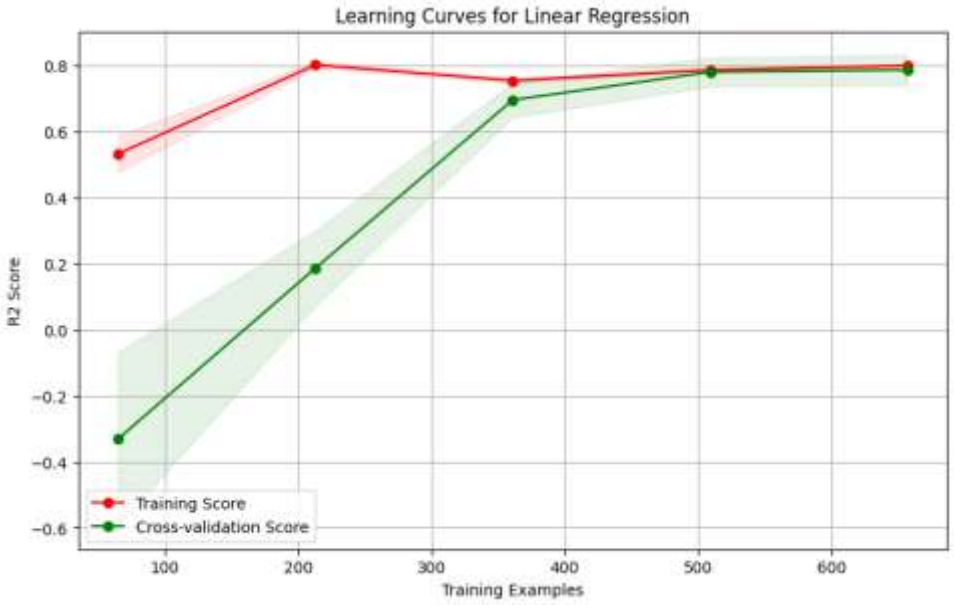
The R2 improved for the cross-validation score but still needs to reach a higher score.

Learning Curve for Random Forest:



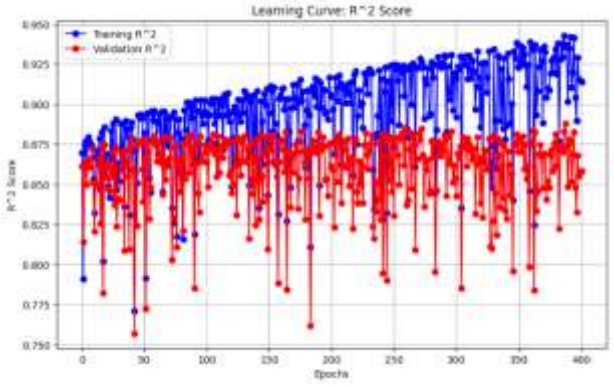
We can see here a better improvement of the R2 score for the cross validation, but it is still less than the Training Score.

Learning Curve for Linear Regression:



We can see here that the learning curve for cross validation improved a lot and attained the same level of R2 of Training. We can deduce here that the Linear Regression Model performed well with our data. However, it didn't reach a high score as other models did, because of the simplicity of the model. For example, the training score for random forest was 0.95 and validation score was 0.85, which is much less than those obtained with Linear Regression.

Learning Curve for Neural Networks:



I tried many times to change the hyperparameters to get a better learning curve. We can see that the cross-validation score is plateauing with the improvement of our training score, and we're at risk of suffering from Overfitting. There's also a lot of fluctuations in both scores.

Conclusion for the learning curves:

- The Linear Regression model got a good learning curve since training and validation scores were very close to each other at the end. However, the score itself was less than other models because Linear Regression is a simple model that can't catch very complex relations in the data.
- Moving on to the Decision Tree regressor, the learning curve wasn't perfect because there was a gap between the final training and validation R2 scores. However, the results were slightly better than linear regression in terms of R2 scores. We are seeing a little bit of overfitting. Also, the training R2 is around 0.93 but the validation is around 0.83 as well as the testing score. We also must find a better set of hyperparameters, but the results aren't very bad.
- For the Random Forest Regressor, the R2 scores were higher than others, but there's also a gap between training and validation scores: 0.96 vs 0.85. The validation score was also plateauing at a certain time, then regained some improvement.
- Regarding Neural Network, the validation score was plateauing and less than the training data. We tried to change the model a little bit but didn't see better results.

We can say that when we have simple models, training and validation scores are close to each other, with no suffering from overfitting or plateauing, but at the expense of a weaker score. Moving to more complex models, we can see some improvements in terms of numbers, but we are facing gaps between training and validation scores, as well as slower improvement of validation scores or plateauing. We must find better hyperparameters to achieve better results, such as reducing the depth of decision trees, or reducing the number of estimators for random forest, and tuning the number of layers, or neurons as well as learning rate for the Neural Networks Model.

Part III:

Question a – The coefficients for each model are:

1) For Linear Regression:

   Coefficients: season: 525.8016 - yr: 2026.8505 - mnth: -39.4844 - holiday: -404.6873 - weekday: 74.1871 - workingday: 161.9908 - weathersit: -625.3008 - temp: 2111.0230 - atemp: 3459.9844 - hum: -831.7712 - windspeed: -2113.4059 - Intercept: 1227.0738

   This model assigns the most important features to be year (2026), temperature (2111), feel-like temperature (3459), and windspeed (-2113).

   The other secondary features are season (525), holiday (-404.68), humidity (-831.77) and weather situation (-625)

   Month, weekday and workingday are not really affecting our target variable.

   These weights mean that each unit increase in each feature increases or decreases the target variable by the weight assigned.

   These results are relevant, and we can say that the second year of our data saw a very high increase in the number of rented bikes, which can be easily visualized. The most important features are temperature, feel-like temperature and windspeed: this makes sense because when the temperature & feel-like temperature get higher, people will most likely use their bikes. As for windspeed, it is very logical to have a negative number because with an increased windspeed, people will be less likely to go ride their bikes.

   For humidity, the lower it is, the better (negative relationship), and also the season here plays a minor role (during summer, we have better weather hence better situation to rent a bike). The negative relation for holiday is not very interesting because I would personally expect people to rent more bikes during holidays. Maybe they are used for other purposes than having a good time (maybe people use them to go to their job).

   The other variables are not important like the month, weekday and workingday because they're not the main factors contributing to the number of bikes rented.

2) Decision Tree Regressor;

season: 0.056148421288565 48 yr: 0.32357549203582747 mnth: 0.018616142916399417 holiday: 0.0 weekday: 0.007428228575751874 workingday: 0.0011273791104784326 weathersit: 0.01880658867534575 temp: 0.43730012581434025 atemp: 0.06197183473188558 hum: 0.04886246660900268 windspeed: 0.02616332024240318

These coefficients show us that the most important features are Year and Temperature. The variables with moderate importance are month, feel-like temperature, humidity and windspeed, unlike the Linear Regression Model. These results are logical because the temperature is a main contributor to the number of bikes rented, and windspeed and humidity play also a role. However, I find that windspeed and humidity should have a more important role, but this is what the decision tree found.

3) Random Forest Regressor:

season: 0.0497 yr: 0.2815 mnth: 0.0309 holiday: 0.0010 weekday: 0.0163 workingday: 0.0048 weathersit: 0.0158 temp: 0.3678 atemp: 0.1440 hum: 0.0580 windspeed: 0.0303

Same as before, the leading important variables are the year, the temperature and the feel-like temperature. The moderate features are humidity, windspeed, season and month. Weekday, holiday, workingday, and weathersit are not really important.

Same explanation as before, but here the feel-like temperature has more importance than its score with decision tree regression.

4) Feed Forward Neural Networks:

We can't extract the weights because of the large number of layers, hidden nodes... It is not as straightforward as for the other models.

Question b -

Let’s recall the feature importance given from the decision tree regressor (previous question):

season: 0.056148421288565 48 yr: 0.32357549203582747 mnth: 0.018616142916399417 holiday: 0.0 weekday: 0.007428228575751874 workingday: 0.0011273791104784326 weathersit: 0.01880658867534575 temp: 0.43730012581434025 atemp: 0.06197183473188558 hum: 0.04886246660900268 windspeed: 0.02616332024240318

The sum of all these features is equal to 1 (normalized). The fact that temperature has the value 0.4373 means that it is a very crucial part for predicting the number of bikes. It means that 43% of the importance in predicting the number of bikes is attributed to the temperature. The year also has an impact, with a value of 0.32. These 2 features combined represent 0.75 or 75% of the importance in the prediction of number of bikes. From the linear regression model, we found that an increase in the temperature will result in an increase in the number of bikes, and the second year will also indicate a higher number.

We can conclude here that there is a relationship between these two features: maybe in the second year there was a better temperature, which resulted in an increased number of rented bikes.

For future work, if we were to predict the number of bikes for another year, we should investigate the temperature if we’re focusing our attention on the decision tree regressor.