Intelligent Systems Project

Bike rentals in Washington D.C.

Gagliano Giuseppe

06/07/2015

Index

Picture index

1. Introduction

The aim of this project is to generate Intelligent Systems that help in finding relations between the bike rental of Washington D.C. and its environment variables (i.e. weather situations, seasons, period, etc) to make forecasts about the future of the rentals.

The target datasets used are contained into the file *Bike-Sharing-Dataset.zip*. It also contains a *Readme.txt* file with all the specification of the datasets. Notice that the dataset has a lack of data due to a natural disaster (hurricane) in which there was not rental activity. This "hole" is considered as useful information to see how robust is the systems w.r.t. outliers.

In the chapter ***Fitting*** of this document it will covered the first part of the project which consists in the fitting of the bike rentals using a subset of features, from the provided dataset, as inputs. This was done either for the daily dataset and the hourly dataset.

In particular, first I've tried to find the best subset of features that returns the smallest error, in terms of Mean Squared Error, using a MLP Neural Network.

Then I used this subset of features to evaluate how the performances changes using a RBF Neural Network, a Mamdani Fuzzy Inference System (FIS) and an ANFIS.

In the chapter ***Forecasting*** instead, I've tried to get some forecasts starting from trained NN. In particular, in the first part I've used an open loop scheme (outputs depends only from inputs and its predecessors), in the second part I've used a closed loop scheme (outputs depends either from the inputs, its predecessors and from previous output).

In all the cases I've tried to get best results, in terms of MSE, varying opportunely the different parameters.

2. Part I – Fitting

A preliminary operation needed is the rearrangement of the dataset to better handle data in Matlab. In particular what was done is the extraction of the day feature from the "dteday" field and the deleting of the "registered" and "casual" fields because, as assumption, I preferred to focus only on the "count" feature.

I. Neural fitting model development and feature selection

1. MLP NN

To find the best subset there are different approaches, the best approach that can be used depends on different factors (size of dataset, available computational power, target precision, etc). For this analysis, I've considered two different approaches:

The first one is used for the daily dataset and it evaluates the MSE of all the possible combination of features. It's very computational expensive and for this reason it's not used for the hourly dataset which is 23 times larger.

The script ***day1_min_mse.m*** generates a csv file named ***day1_min_mse.csv*** which contains the MSE of each MLP with different subsets of features.

The second approach is more engineering and it's based on the observation of the behavior of the system. I noticed that, starting with one feature and increasing by one each step taking the feature with the best MSE, MSE decreased until a certain step. I considered the best subset, the combination that gave me the smallest MSE.

The script ***hour1_min_mse.m*** contains the Matlab code modified for the last simulation (biggest subset combination). The results are aggregated into the file ***hour1_min_mse.xlsx***.

These are the results:

Data Series	Optimal MSE	Features
day.csv	4,18E+005	3 – season 4 – yr 5 – mnth 7 – weekday 9 – weathersit 11 – atemp
hour.csv	3,04E+03	3 – season 4 – yr 5 – mnth 6 – hr 9 – workingday 10 – weathersit 11 – temp

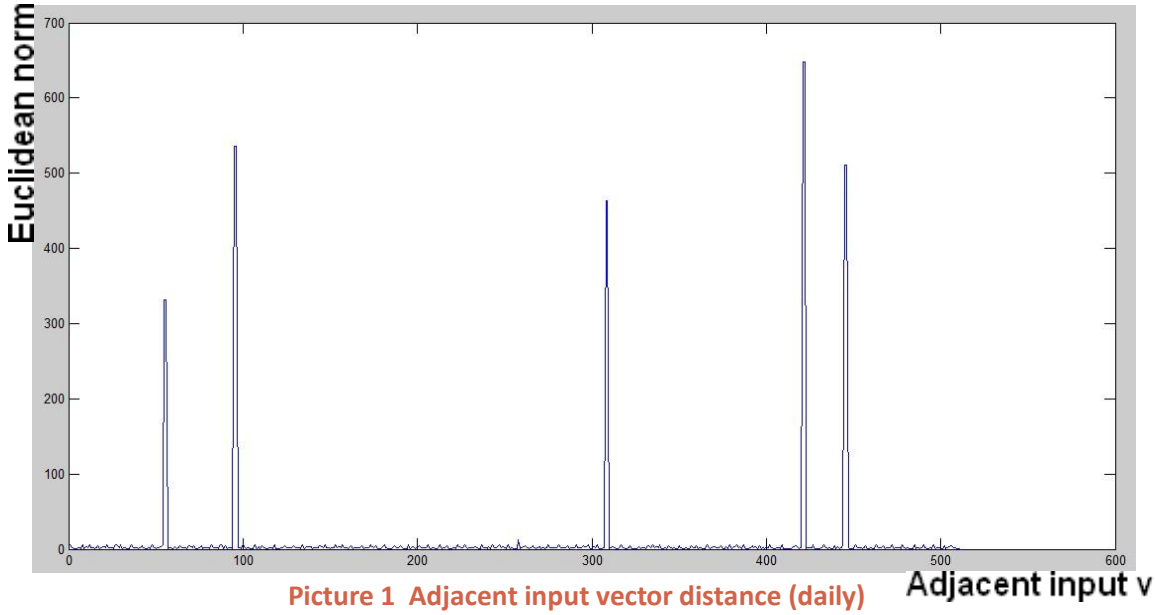
Notice that, even if the MSEs appear as large numbers, they can be considered good if we look at the count variability (variance of count is 3,75E+006 for the daily and 3,3114E+003 for the hourly dataset).

2. RBF NN (DAILY)

The script ***day2_rbf.m*** create and train an RBF (for the daily dataset) that takes as input the above subset and as target the *count* field.

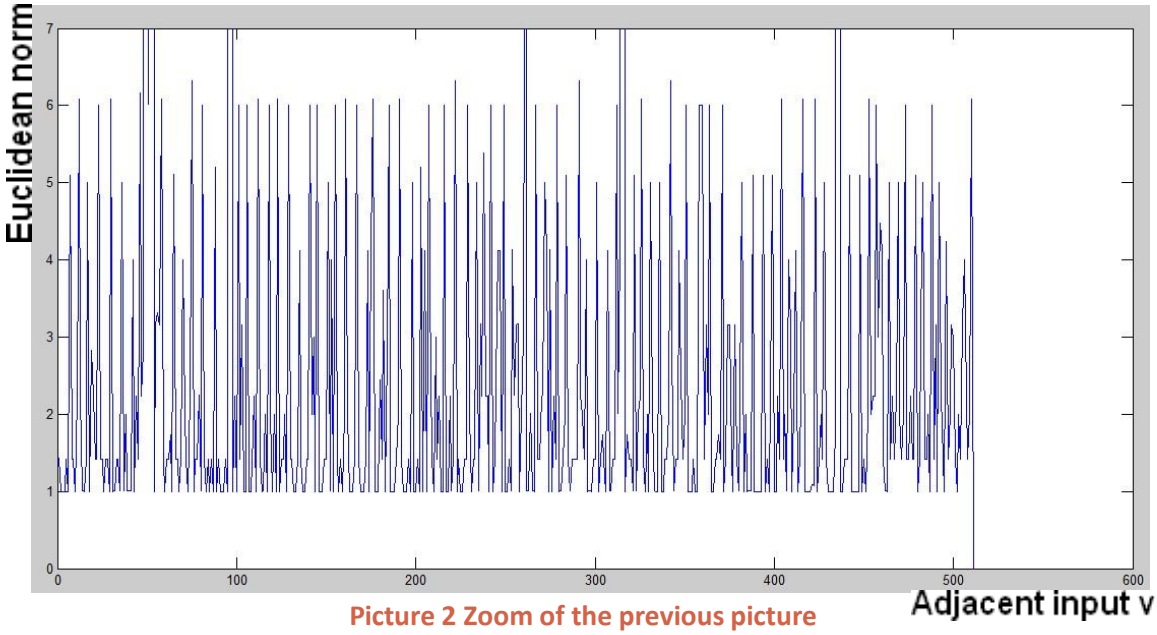
First thing to do is randomly divide the set in 2 subset: training and testing. This is done by the ***dividerand*** function. I've chosen 70% of samples for the training 30% for the testing

Then it is needed to choose the target error and the spreading constant (SC) to define the RBF network with the function ***newrb***. The first is set to a value near to zero (0.02).
For the second, I've chosen considering that to a get good generalization the spread constant has to be larger than the distance between adjacent input vectors, but smaller than the distance across the whole input space. Computing the Euclidean norm I obtained the following results:



Picture 1 Adjacent input vector distance (daily)

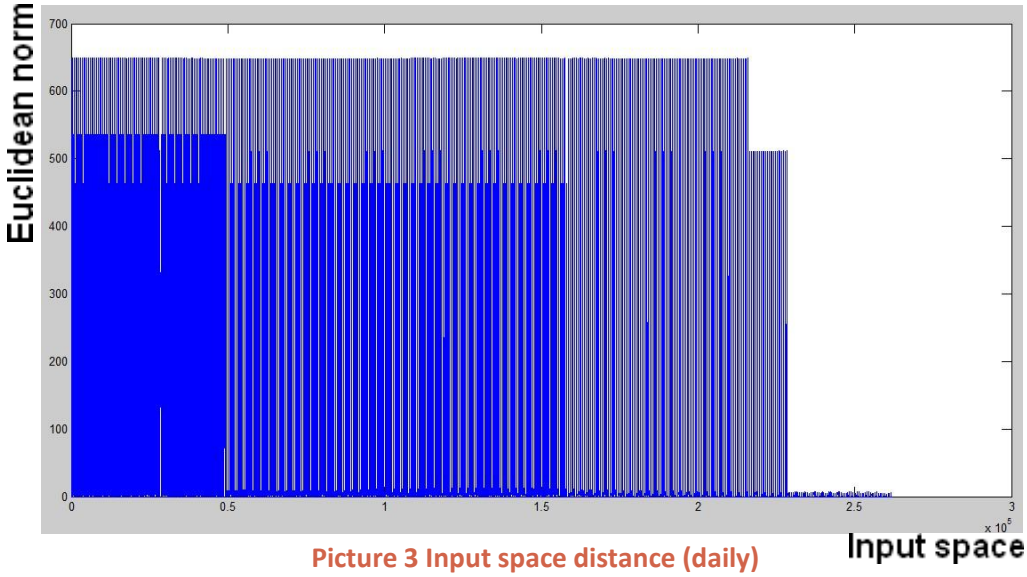
In the latter it can be seen the plot for adjacent input vectors. Notice that there are 5 outliers that has distance of over 300, the others are below 7, as it is shown in the following plot:



Picture 2 Zoom of the previous picture

So, I've considered 7 as lower bound for the SC.

Whereas the following plot shows the distance of the input space:



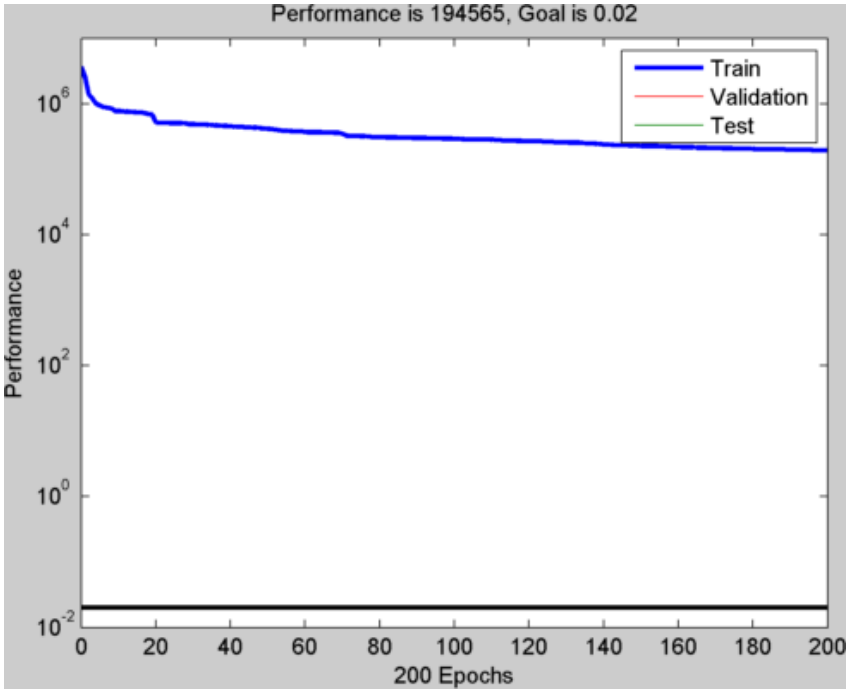
Picture 3 Input space distance (daily)

In this case I've considered as upper bound the value of 650.

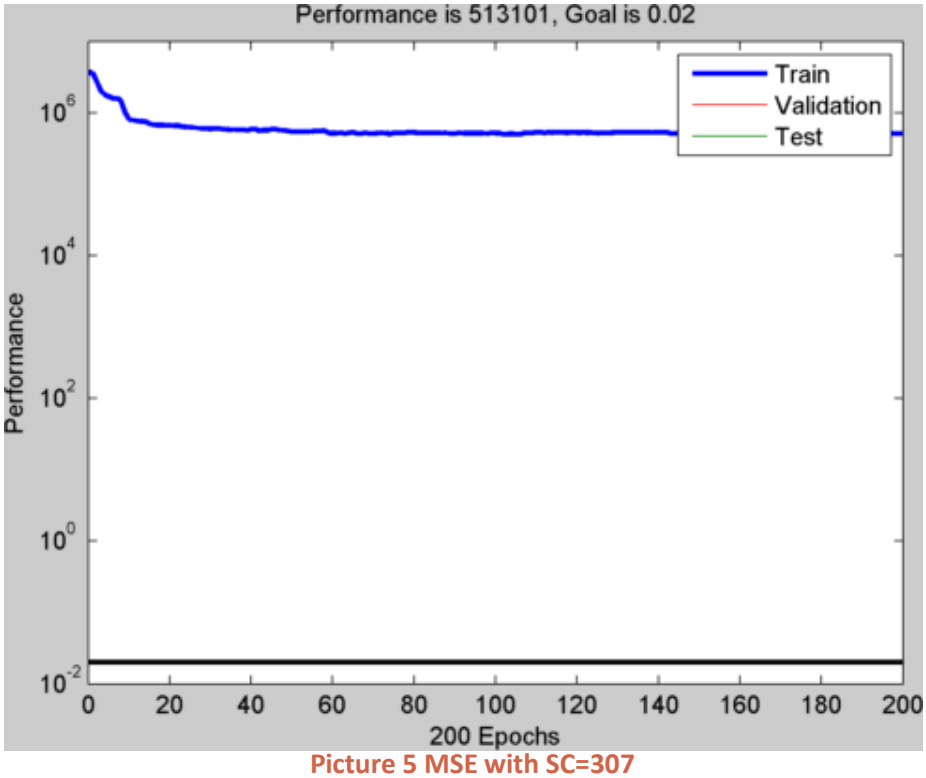
So, the SC should be between 7 and 650.

Simulating with different SC I obtained different results.

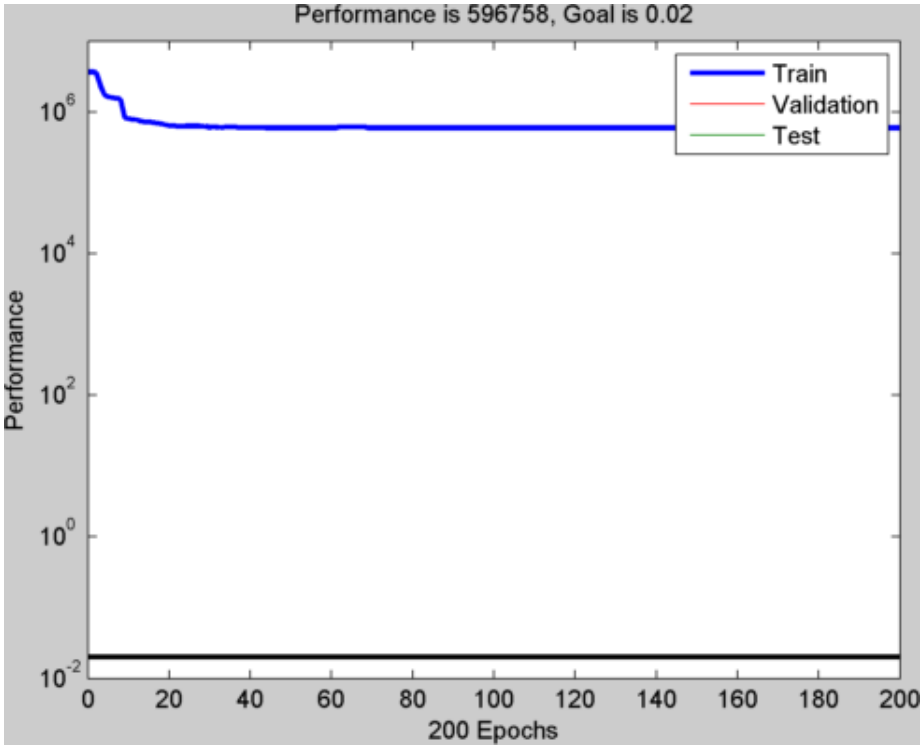
To sum up I noticed that increasing the SC the MSE increase but needs more epochs to reach a stable value. The following plots shows three cases with a sc of 7, 307 and 657:



Picture 4 MSE with SC=7



Picture 5 MSE with SC=307



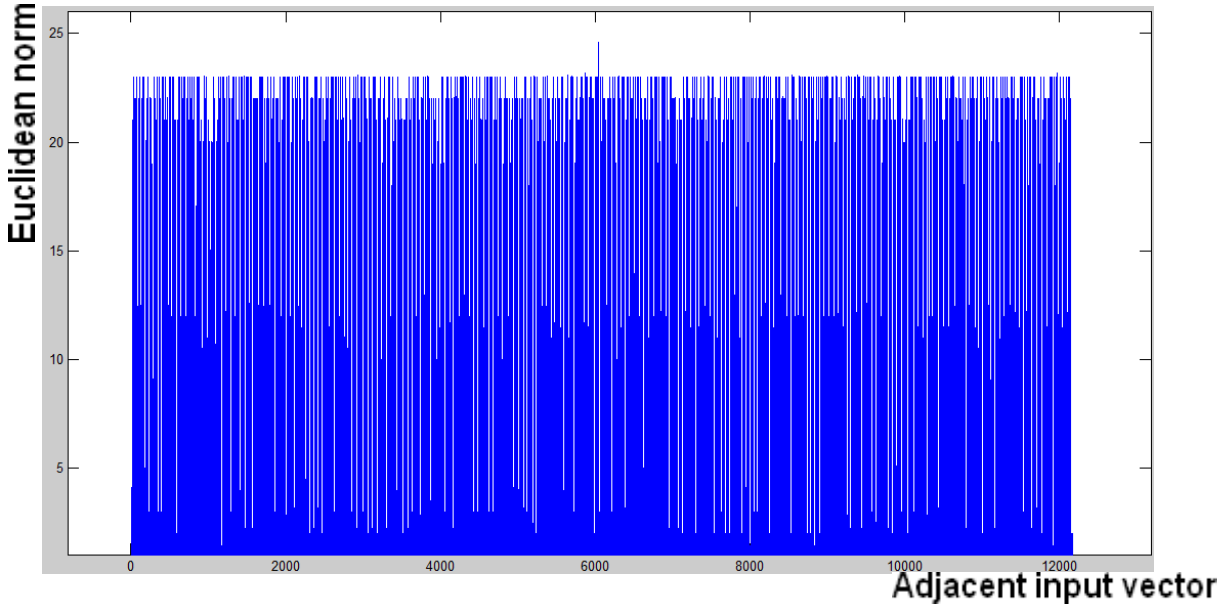
Picture 6 MSE with SC=657

It can be seen that, in all the cases, when it is reached a steady state the MSE is pretty constant to a value with an order of magnitude of 10^5.

Trying with SC= 57, the MSE for the test set is $6.5575 \cdot 10^5$, this means that the performances are approximately the same to the MLP for the day dataset.

3. RBF NN (HOURLY)

For the hour dataset the adjacent distances are:



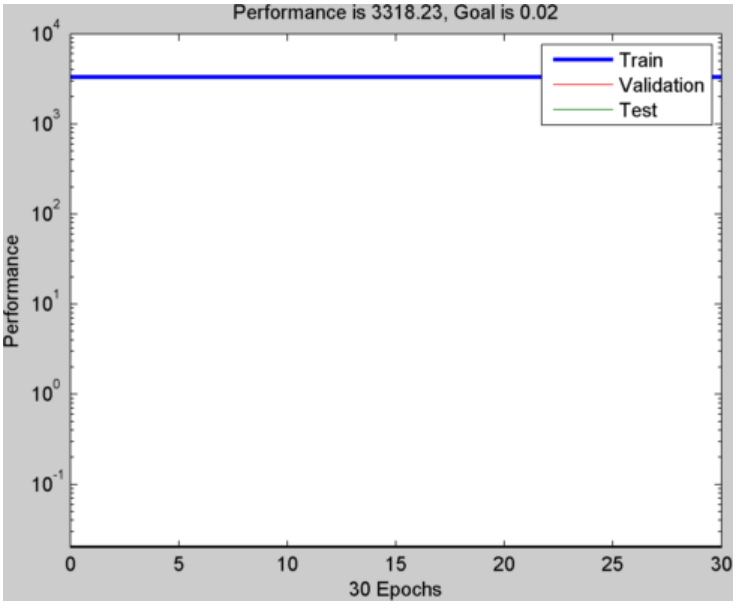
Picture 7 Adjacent input vectors distances (hourly)

Notice that either in this case there is an outlier (distance>23). In this case I considered as minimum distance 23.

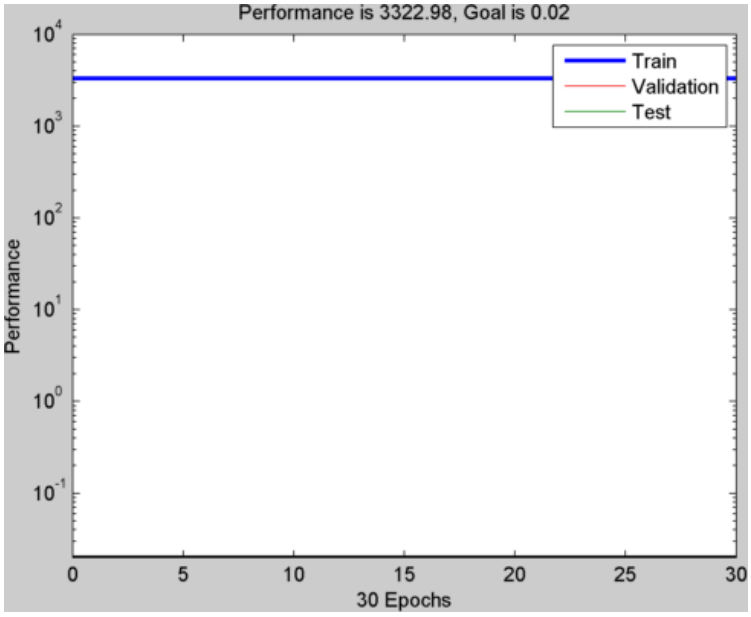
For the maximum distance, using the *dist* function It's not a good idea because of the great amount of data. I used the following approach: if I consider the input as a hypersphere centered at 0, with a radius equal to the longest input vector, in the worst case (opposite vectors), two vectors will be distant 2 times the longest vector. So, I computed the norm of all the training inputs (which is the length of the vectors with origin at 0 and pointing to the input values), then I computed the maximum value of this set of values and I considered its double (i.e. 53).

The final result is that the SC should be between 23 and 53.

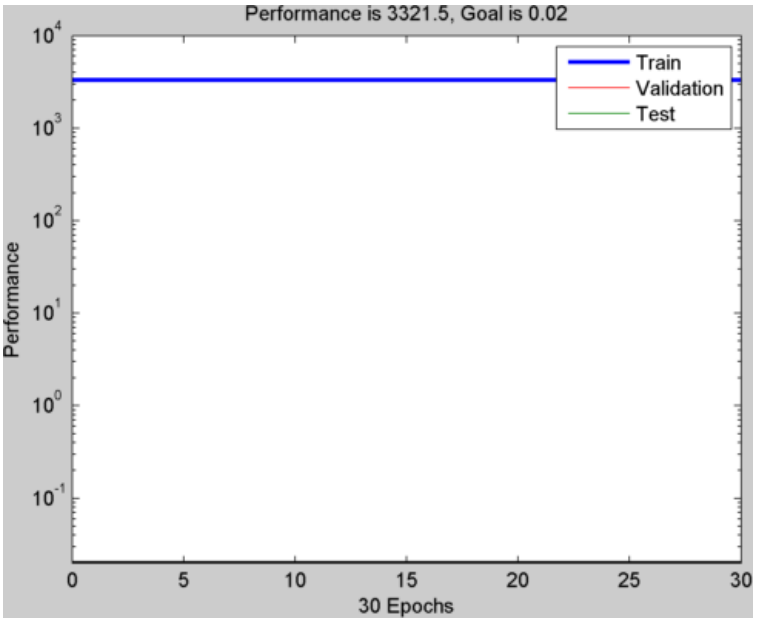
The results are the following (with SC={23, 38, 53}):



Picture 8 MSE with SC=23



Picture 9 MSE with SC=38



Picture 10 MSE with SC=53

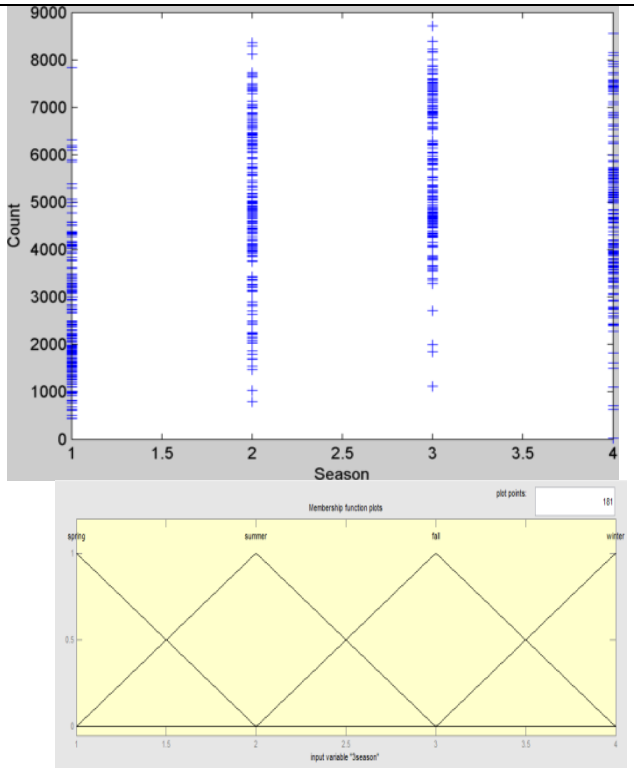
The MSE is pretty constant to 10^3 in all the cases. Considering SC=38 and MN=50 (maximum number of neurons), the MSE for the testing is equals to 3.4666e+003.

II. Fuzzy fitting model development

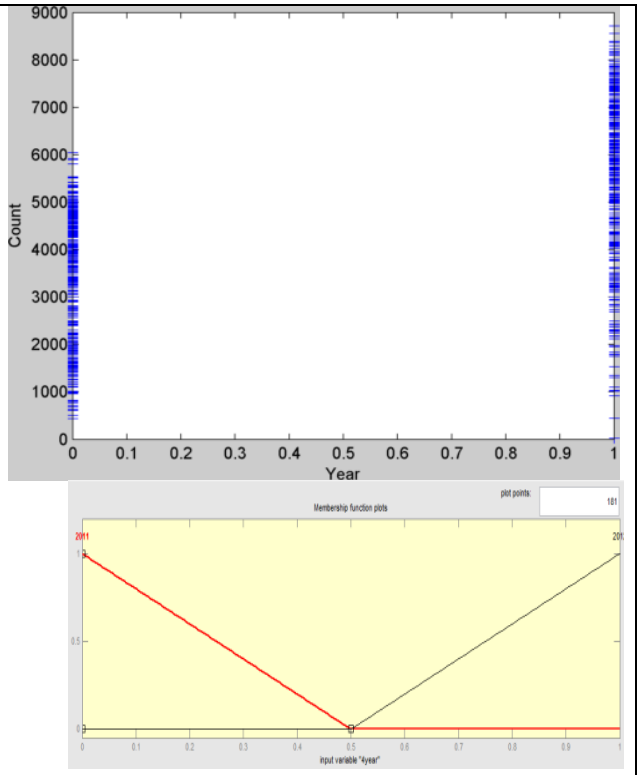
1. Mamdany fuzzy system (Daily)

In this case we have to estimate the bike rental with a Fuzzy Inference System with the same subset of the previous sections which are taken as input.

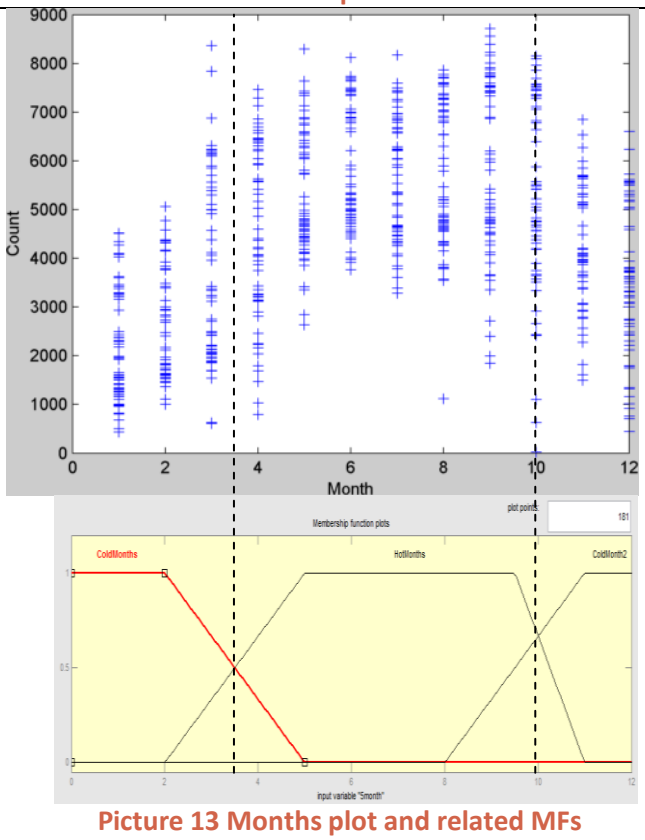
The problem here is to understand how the features affects the output. In the following pictures I show the different features with the related chosen membership functions for the input of the Mamdani FIS.



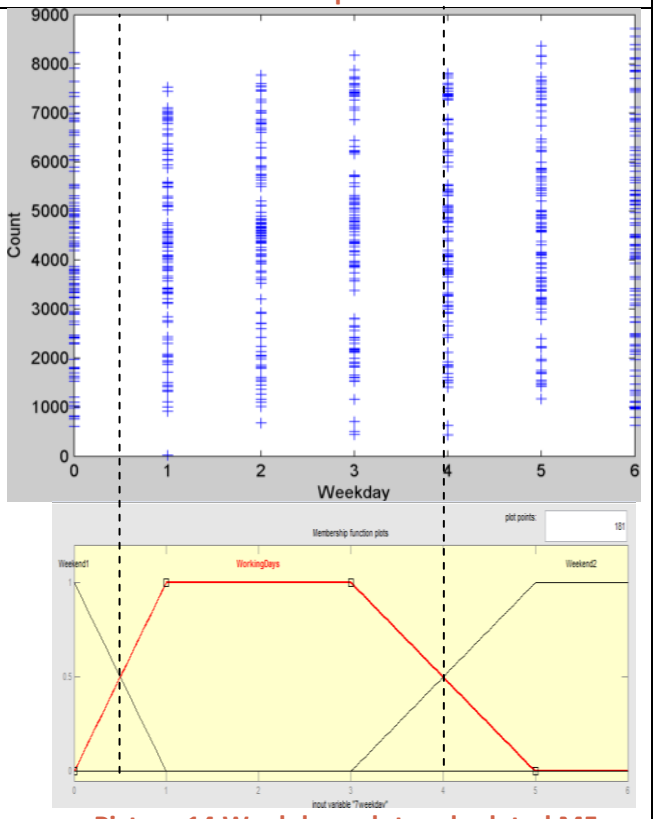
Picture 11 Seasons plot and related MFs



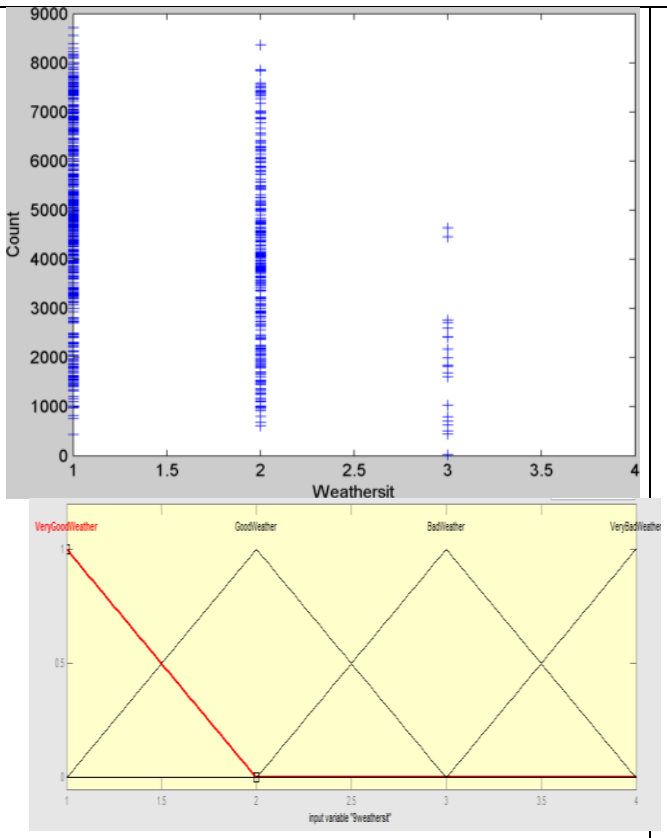
Picture 12 Years plot and related MFs



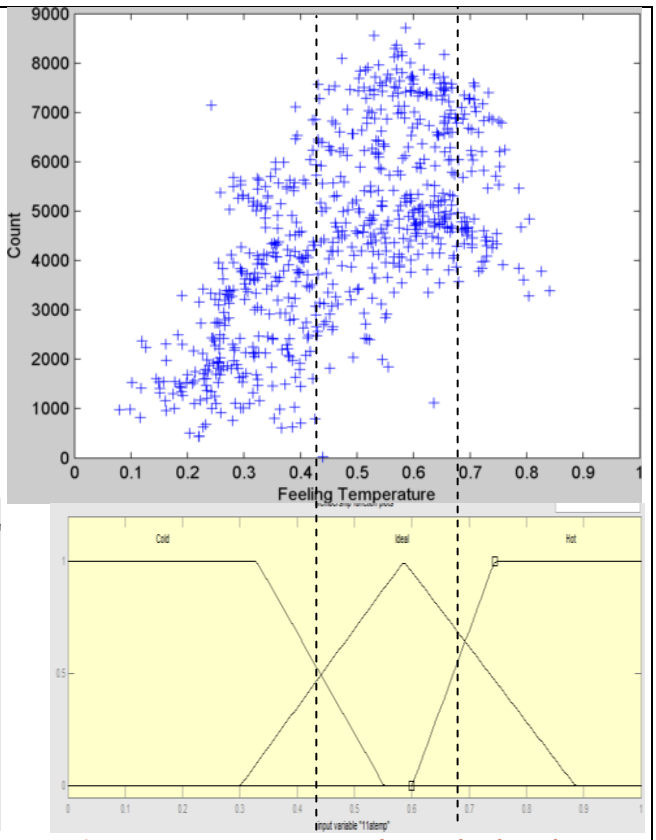
Picture 13 Months plot and related MFs



Picture 14 Weekdays plot and related MFs



Picture 15 Weathersit plot and related MFs

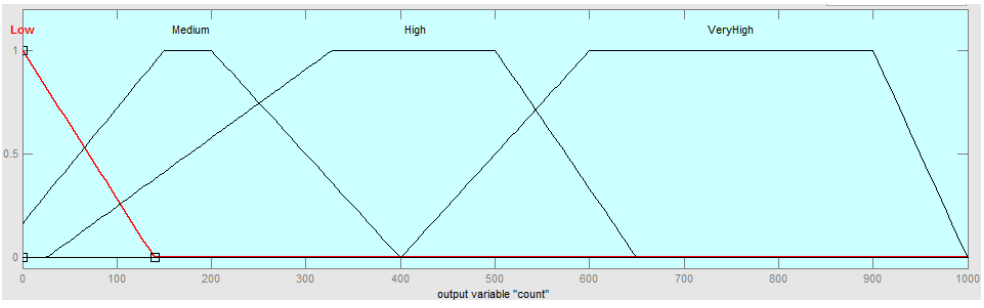


Picture 16 Temperature plot and related MFs

To do this, several choices were made:

- I used MFs that intersect each other either in cases where the feature assumes discrete values, this was made to avoid possible strange behavior due to different precision between the values and the system
- The original *atemp* feature contained some outliers that caused out of bounds error. These values were replaced with their precedent value to bring again the range [0 1]
- I thought were convenient for the month and weekday features, although they were discrete, to group them in few MFs (for example instead of one MFs for one months, three MFs for 12 months)
- The above plots, in which are shown the relations count/feature, were used to find relations that have been useful either to better define the MFs and to find the rules.

For the output I decided to consider the following MFs:



Picture 17 MFs plot of the Output FIS variable (daily)

For the rule assigning I've used a top down approach in which, first I've tried to fit the large scale trends considering inputs with a spread influence (i.e. year, seasons, month), and then I've added some rules that allow to fit small scale trends (i.e. weathersit, weekdays).

RULES:

1. If (3season is spring) and (4year is 2011) and (5month is ColdMonths1) and (9weathersit is BadWeather) and (11atemp is Cold) then (RentedBikes is VeryLow) (1)

2. If (3season is winter) and (5month is ColdMonths1) and (9weathersit is BadWeather) and (11atemp is Cold) then (RentedBikes is VeryLow) (1)

3. If (3season is fall) and (4year is 2012) and (5month is not ColdMonths2) and (9weathersit is not BadWeather) and (11atemp is Ideal) then (RentedBikes is High) (1)

4. If (4year is 2012) and (5month is HotMonths) and (9weathersit is not BadWeather) and (11atemp is Ideal) then (RentedBikes is High) (1)

5. If (5month is ColdMonths1) and (11atemp is Cold) then (RentedBikes is Low) (1)

6. If (5month is ColdMonths2) and (11atemp is Cold) then (RentedBikes is Low) (1)

7. If (4year is 2012) and (5month is ColdMonths2) then (RentedBikes is High) (1)

8. If (4year is 2011) and (5month is ColdMonths1) then (RentedBikes is VeryLow) (1)

9. If (4year is 2011) and (9weathersit is BadWeather) and (11atemp is Cold) then (RentedBikes is VeryLow) (1)

10. If (3season is spring) and (4year is 2011) then (RentedBikes is VeryLow) (1)

11. If (4year is 2011) and (5month is ColdMonths2) then (RentedBikes is Medium) (1)

12. If (4year is 2012) and (5month is ColdMonths2) then (RentedBikes is Medium) (1)

13. If (4year is 2012) and (5month is ColdMonths1) then (RentedBikes is Medium) (1)

14. If (3season is summer) and (4year is 2011) and (5month is ColdMonths1) then (RentedBikes is Medium) (1)

15. If (9weathersit is BadWeather) and (11atemp is Cold) then (RentedBikes is Low) (1)

16. If (7weekday is Weekend2) and (9weathersit is GoodWeather) and (11atemp is Ideal) then (RentedBikes is Medium) (1)

17. If (7weekday is Weekend1) and (9weathersit is GoodWeather) and (11atemp is Ideal) then (RentedBikes is Medium) (1)

PERFORMANCES

ME = -81.5094

MAE = 793.3932

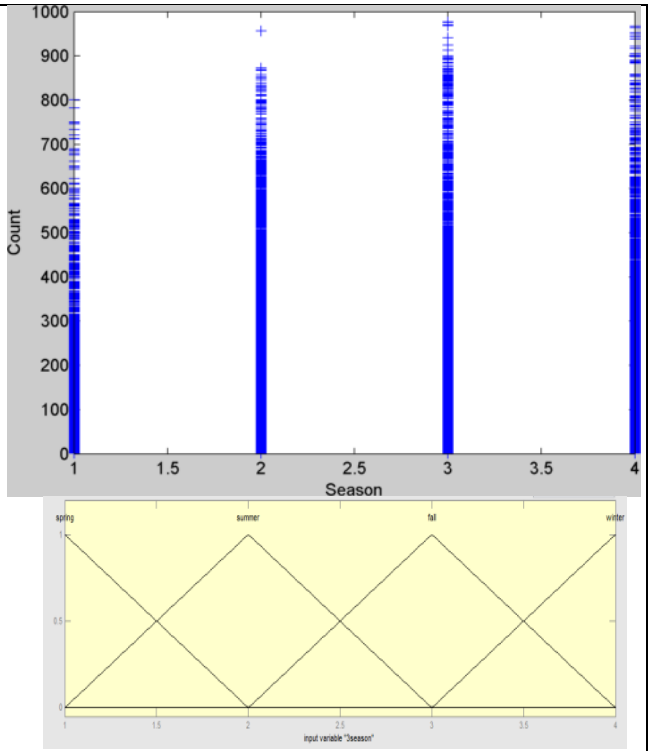
MSE = 2.4037e+007

MPE = -2.0250 %

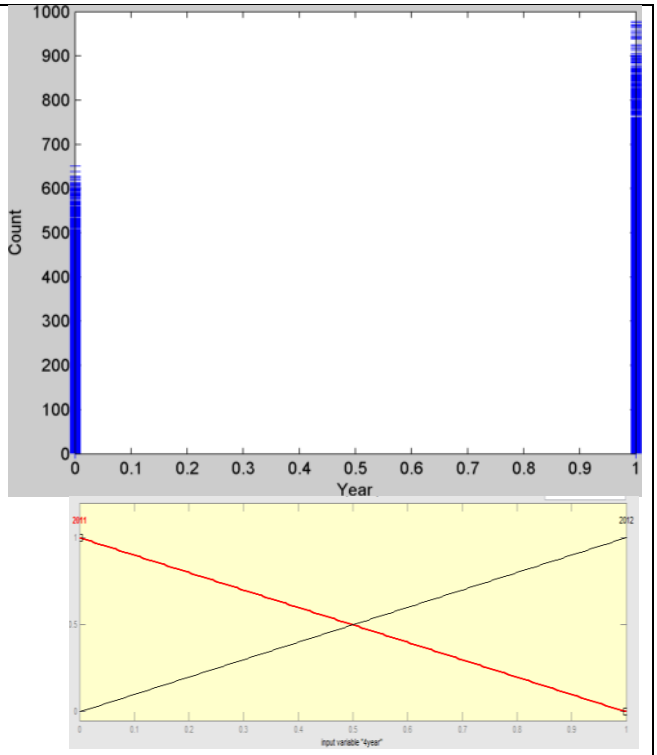
MAPE = 19.9763 %

2. MAMDANY FUZZY SYSTEM (HOURLY)

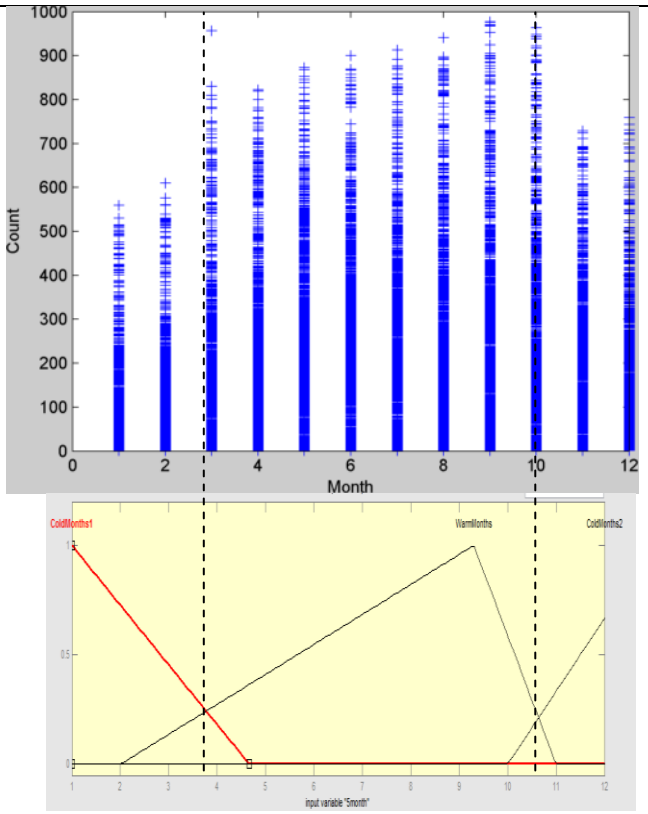
Either in this case I've plotted the target data (count or number of bike rentals) with respect to other features to understand how they affect it:



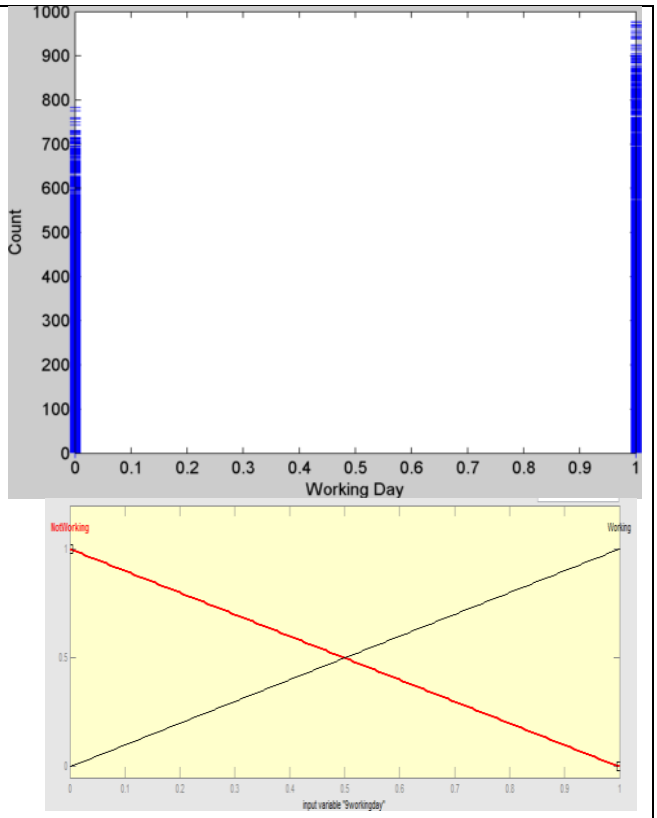
Picture 18 Seasons plot and related MFs



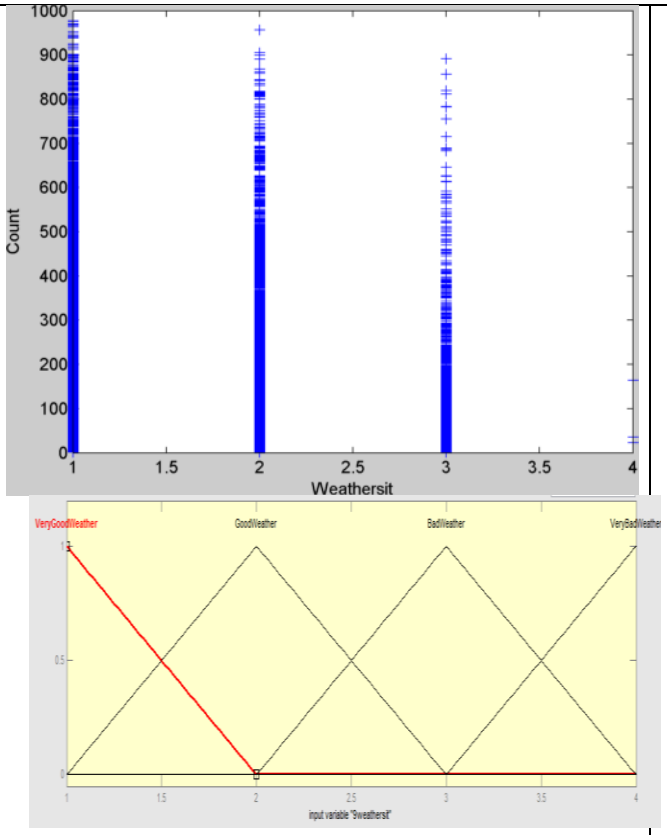
Picture 19 Years plot and related MFs



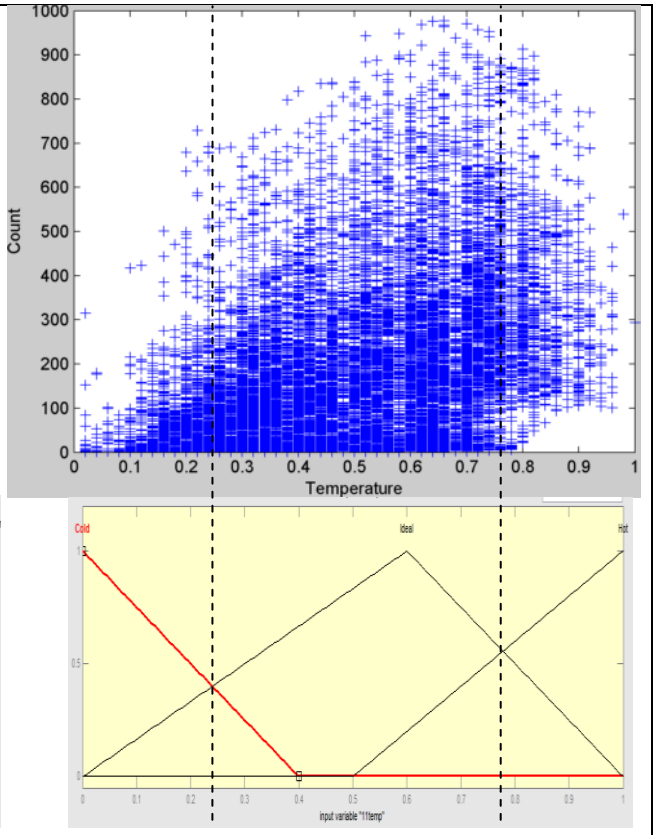
Picture 20 Months plot and related MFs



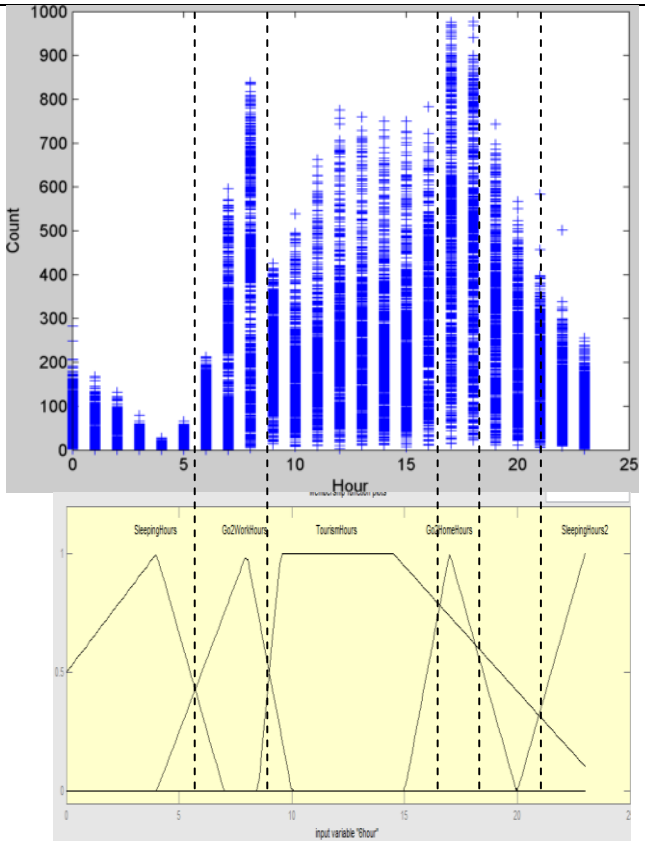
Picture 21 Working days plot and related MFs



Picture 22 Weather sits plot and related MFs

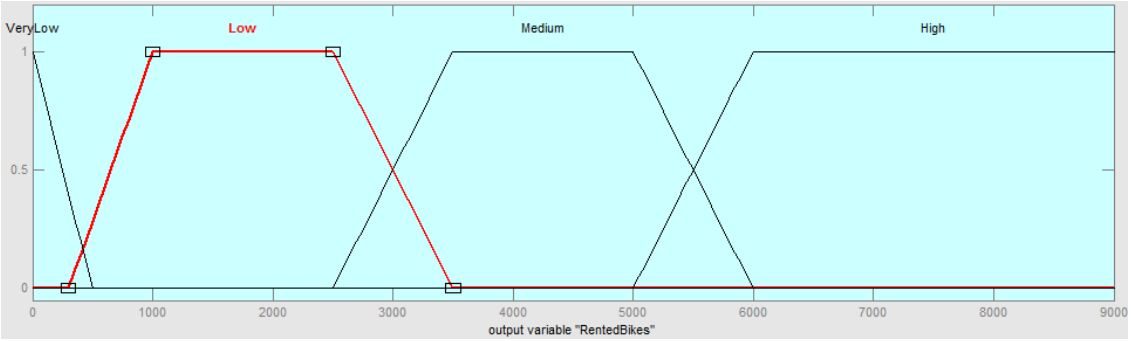


Picture 23 Temperatures plot and related MFs



Picture 24 Hours plot and related MFs

Either in this case I've tried to follow some intuitions to create groups that helped me in define less MFs, and consequently, less rules.



Picture 25 MFs plot of the Output FIS variable (daily)

For the rule assigning I've used the same approach as before.

RULES:

1. If (3season is Spring) then (count is Low) (1)

2. If (6hour is SleepingHours) then (count is Low) (1)

3. If (11temp is Cold) then (count is Low) (1)

4. If (10weathersit is BadWeather) then (count is Low) (1)

5. If (3season is Spring) and (5month is ColdMonths1) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is Medium) (1)

6. If (3season is Spring) and (5month is ColdMonths1) and (6hour is SleepingHours) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is Low) (1)

7. If (3season is Spring) and (5month is ColdMonths1) and (6hour is Go2WorkHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is Medium) (1)

8. If (3season is Spring) and (5month is ColdMonths1) and (6hour is SleepingHours2) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is Low) (1)

9. If (3season is Spring) and (5month is not ColdMonths2) and (6hour is TourismHours) and (10weathersit is not BadWeather) and (11temp is not Ideal) then (count is Low) (1)

10. If (6hour is SleepingHours2) then (count is Low) (1)

11. If (3season is not Spring) and (5month is not ColdMonths2) and (6hour is SleepingHours) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is Low) (1)

12. If (3season is not Spring) and (5month is not ColdMonths2) and (6hour is TourismHours) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is Medium) (1)

13. If (3season is not Spring) and (5month is not ColdMonths2) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not VeryBadWeather) and (11temp is Ideal) then (count is High) (1)

14. If (3season is Spring) and (5month is WarmMonths) and (6hour is TourismHours) and (9workingday is NotWorking) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is Medium) (1)

15. If (6hour is Go2WorkHours) and (9workingday is NotWorking) then (count is Low) (1)

16. If (3season is Summer) and (6hour is TourismHours) and (9workingday is NotWorking) then (count is High) (1)

17. If (5month is WarmMonths) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is not Cold) then (count is High) (1)

18. If (6hour is SleepingHours) and (11temp is Cold) then (count is Low) (1)

19. If (3season is Fall) and (4year is 2012) and (5month is ColdMonths2) and (6hour is not SleepingHours) and (11temp is Ideal) then (count is Medium) (1)

20. If (3season is Winter) and (4year is 2012) and (5month is not ColdMonths2) and (6hour is Go2WorkHours) and (9workingday is Working) and (10weathersit is not BadWeather) then (count is High) (1)

21. If (3season is Fall) and (4year is 2012) and (5month is not ColdMonths2) and (6hour is TourismHours) and (9workingday is NotWorking) and (10weathersit is not BadWeather) then (count is High) (1)

22. If (3season is Winter) and (4year is 2012) and (5month is not ColdMonths2) and (6hour is TourismHours) and (9workingday is NotWorking) and (10weathersit is not BadWeather) then (count is High) (1)

23. If (4year is 2012) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is not Cold) then (count is High) (1)

24. If (3season is Winter) and (4year is 2012) and (5month is not ColdMonths2) and (6hour is Go2WorkHours) and (9workingday is Working) then (count is High) (1)

25. If (3season is not Spring) and (5month is not ColdMonths2) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is VeryHigh) (1)

26. If (3season is not Spring) and (4year is 2012) and (5month is WarmMonths) and (6hour is Go2WorkHours) and (9workingday is Working) and (10weathersit is VeryGoodWeather) and (11temp is Ideal) then (count is VeryHigh) (1)

27. If (3season is Fall) and (4year is 2012) and (5month is not ColdMonths2) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is not Cold) then (count is VeryHigh) (1)

28. If (3season is Winter) and (4year is 2012) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is VeryGoodWeather) and (11temp is Ideal) then (count is VeryHigh) (1)

29. If (4year is 2012) and (6hour is TourismHours) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is High) (1)

30. If (4year is 2012) and (5month is WarmMonths) and (6hour is TourismHours) and (9workingday is NotWorking) and (10weathersit is VeryGoodWeather) and (11temp is Ideal) then (count is VeryHigh) (1)

31. If (6hour is Go2HomeHours) and (9workingday is Working) then (count is Medium) (1)

32. If (3season is Winter) and (4year is 2012) and (5month is ColdMonths2) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is VeryHigh) (1)

33. If (3season is Winter) and (4year is 2012) and (5month is ColdMonths2) and (6hour is Go2WorkHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is VeryHigh) (1)

34. If (3season is Winter) and (4year is 2012) and (5month is ColdMonths2) and (6hour is TourismHours) and (9workingday is NotWorking) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is High) (1)

35. If (3season is Fall) and (4year is 2012) and (5month is WarmMonths) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is Ideal) then (count is VeryHigh) (1)

36. If (4year is 2012) and (5month is WarmMonths) and (6hour is Go2HomeHours) and (9workingday is Working) then (count is VeryHigh) (1)

37. If (4year is 2012) and (5month is ColdMonths2) and (6hour is Go2HomeHours) and (9workingday is Working) and (10weathersit is not BadWeather) and (11temp is not Cold) then (count is VeryHigh) (1)

PERFORMANCES

ME = 19.3271

MAE = 84.5272

MSE = 6.8796e+004

MPE = 1.3288 %

MAPE = 59.0994 %

As it can be seen either for the fuzzy systems I've found more precise results in the hourly dataset w.r.t. the daily, this is a consequence of the larger amount of data.

3. ANFIS FUZZY SYSTEM (DAILY)

Another kind of approach is the Adaptive Network-based Fuzzy Inference System. In this case most of the work is done automatically by the *anfisedit* function.

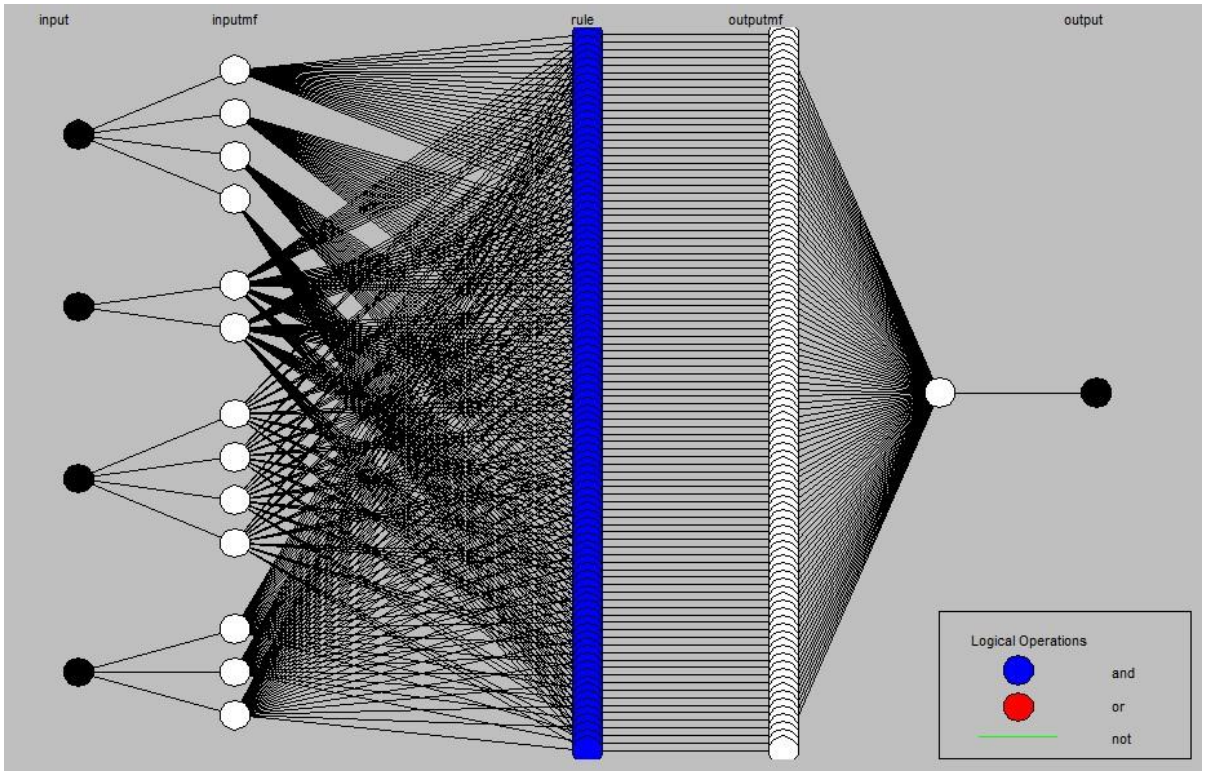
In the scripts ***day4_anfis.m*** and ***hour4_anfis.m*** I only normalized with a max-min normalization $\left(z = \frac{x - min}{max - min}\right)$ and divided data in the three subset used for training, testing and validation.

Than, through the ANFIS editor GUI, I gave the training, testing and checking set, selected the grid partition with Gaussian membership functions for the inputs and constant for the output.

Notice that for the high computational costs I used only the following feature as input

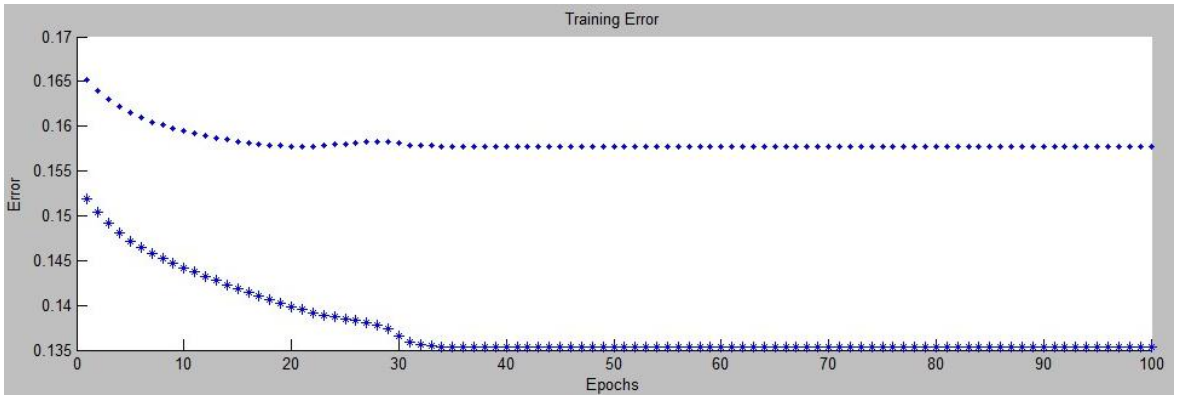
day.csv	3 – season (4 mfs) 7 – weekday (2 mfs) 9 – wheatersit (4 mfs) 11 – atemp (3 mfs)
hour.csv	3 – season (4 mfs) 6 – hr (4 mfs) 9 – workingday (4 mfs) 11 – temp (4 mfs)

The following pictures shows the results for the daily dataset:



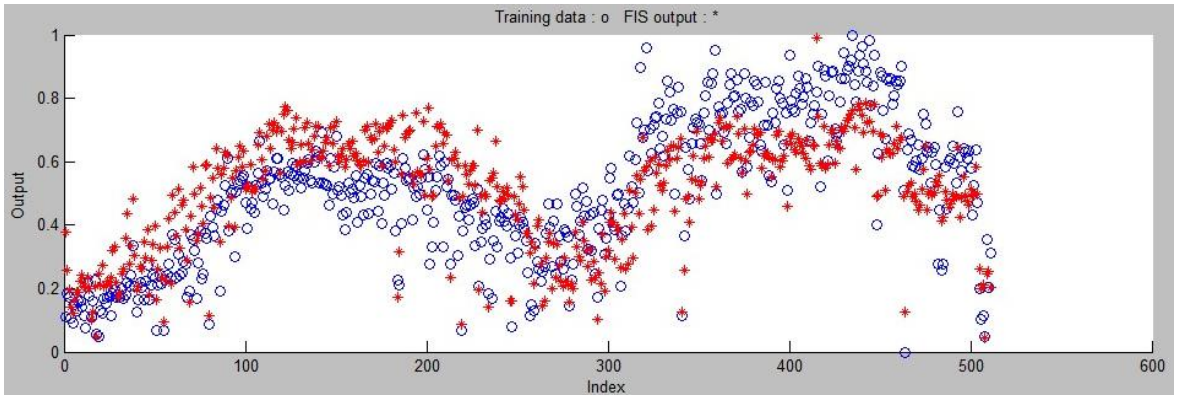
Picture 26 ANFIS Model Structure (daily dataset)

In this case I use a grid partitioning method that is the more computational expensive than subtractive clustering as it can be seen comparing the above with the ANFIS model for the hourly dataset present in the next section.

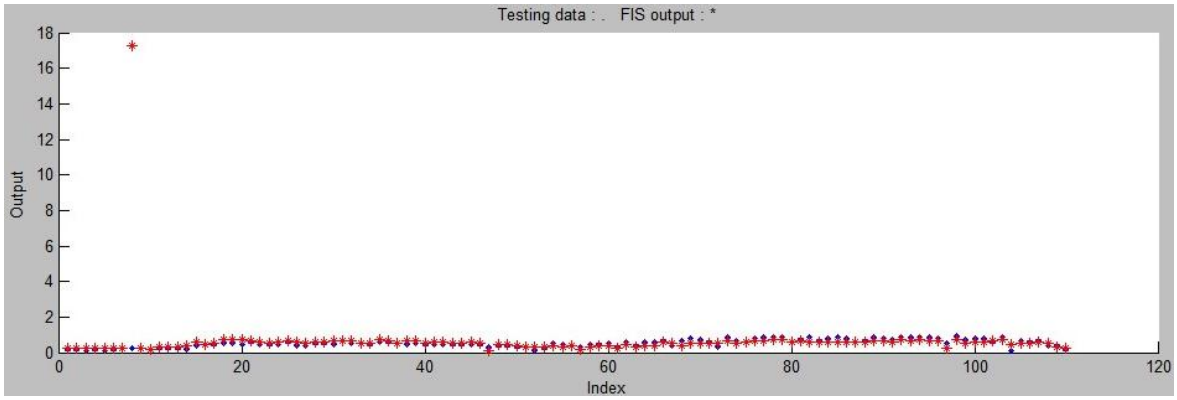


Picture 27 Training Error (daily dataset)

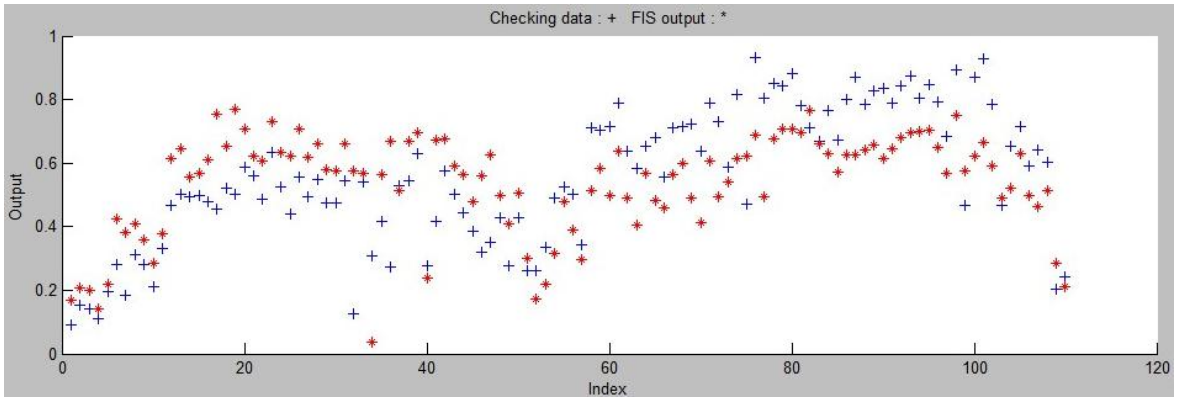
In the picture above it can be seen how Error goes down until about 33 epochs. Then it mantaints ca costant value either for the checking (the higher curve) and for the training.



Picture 28 Training set vs. FIS Output (daily dataset)



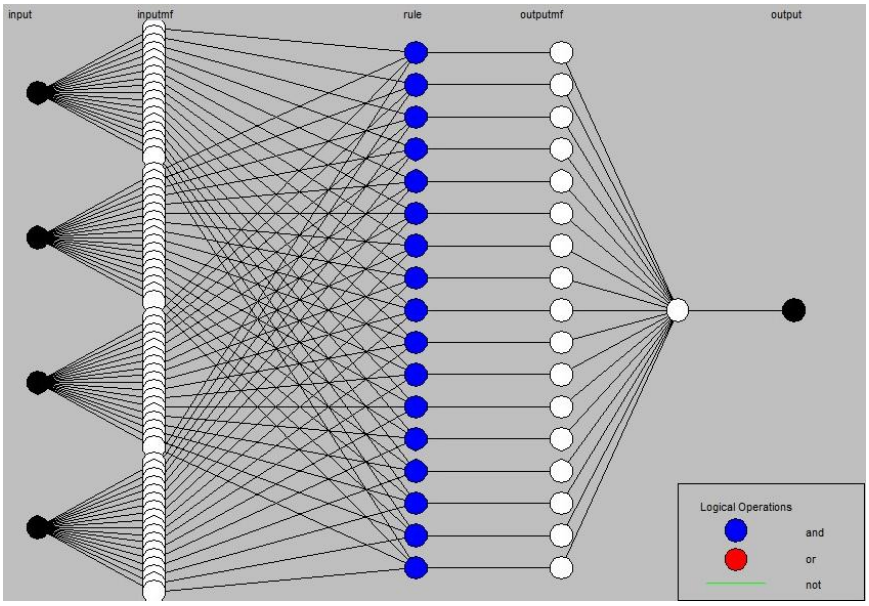
Picture 29 Testing set vs. FIS Output (daily dataset)



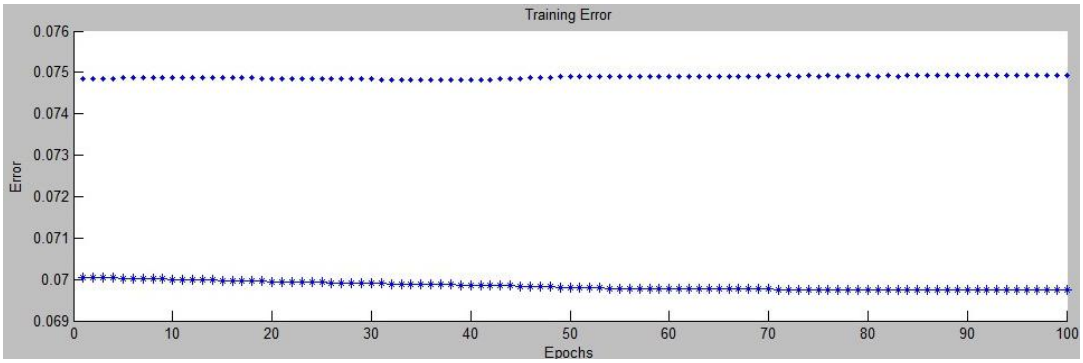
Picture 30 Checking set vs. FIS Output (daily dataset)

4. ANFIS FUZZY SYSTEM (HOURLY)

Whereas for the hour dataset I used a sub clustering partition method, that is a better solution with this great amount of data because it groups similar data points in a cluster. After some trials I preferred to use default values for the parameters because this seems to be a good compromise between computational costs and approximation. The following are the results:

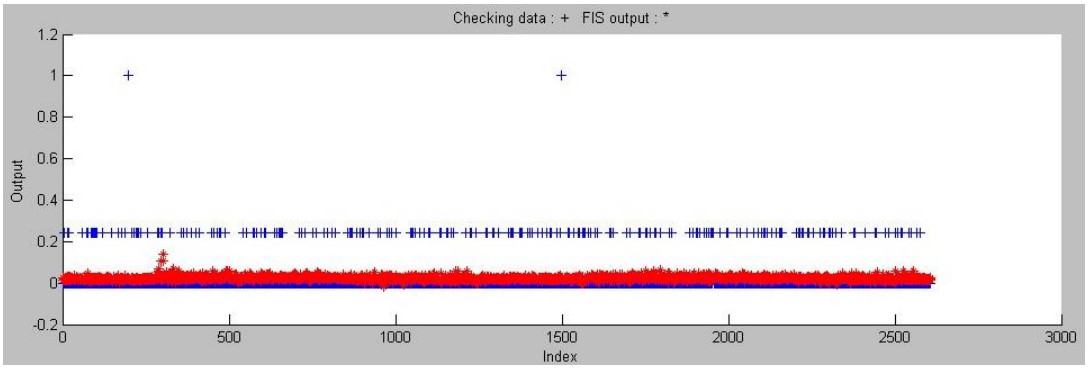


Picture 31 ANFIS Model Structure (hourly dataset)

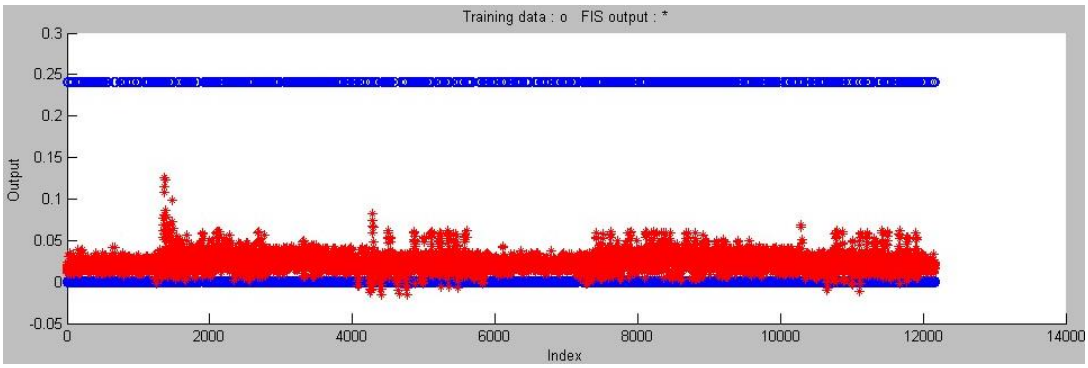


Picture 32 Training Error (hourly dataset)

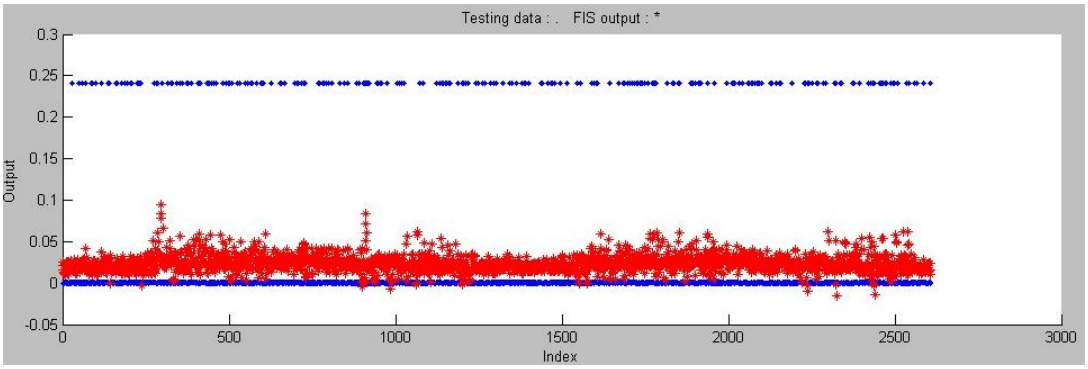
In the picture above the training error is pretty constant to 0.07, checking to 0.075.



Picture 33 Checking set vs. FIS Output (hourly dataset)



Picture 34 Training set vs. FIS Output (hourly dataset)



Picture 35 Testing set vs. FIS Output (hourly dataset)

As it can be seen, in both cases the training error is very low, especially in the hourly dataset (about 1 order of magnitude smaller)

5. Part II – Forecasting

The aim of this sections is to generate a Neural Network that, after the training, returns forecasts about the bike rentals. This work was done for the daily dataset only.

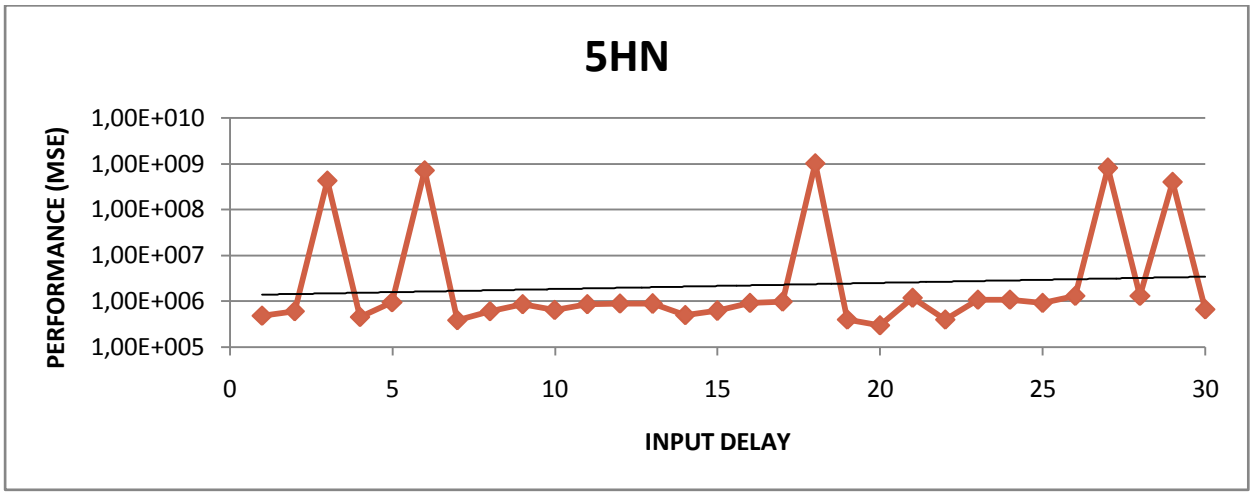
I. Open Loop strategy

In this case I have to consider the first year of the daily dataset and find the best couple of parameters (input delays, hidden layer size) to minimize the mean squared error of the forecast.

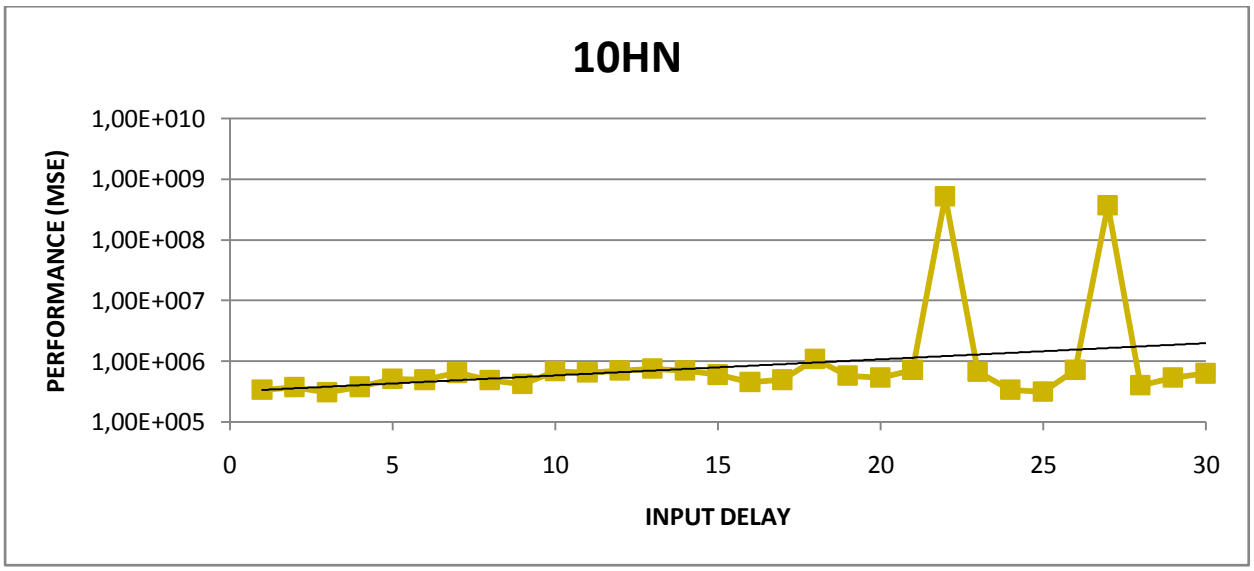
In the script ***day5_forecastOL.m***, after loading the data for the first year, I've pasted the script automatically generated by the NN Time Series Tool and I added a couple of for loops to help me in find some results varying on the input delays and hidden neurons.

I focused mainly on the performance (MSE) of the network, and using some plots I've tried to get insight from the MSEs.

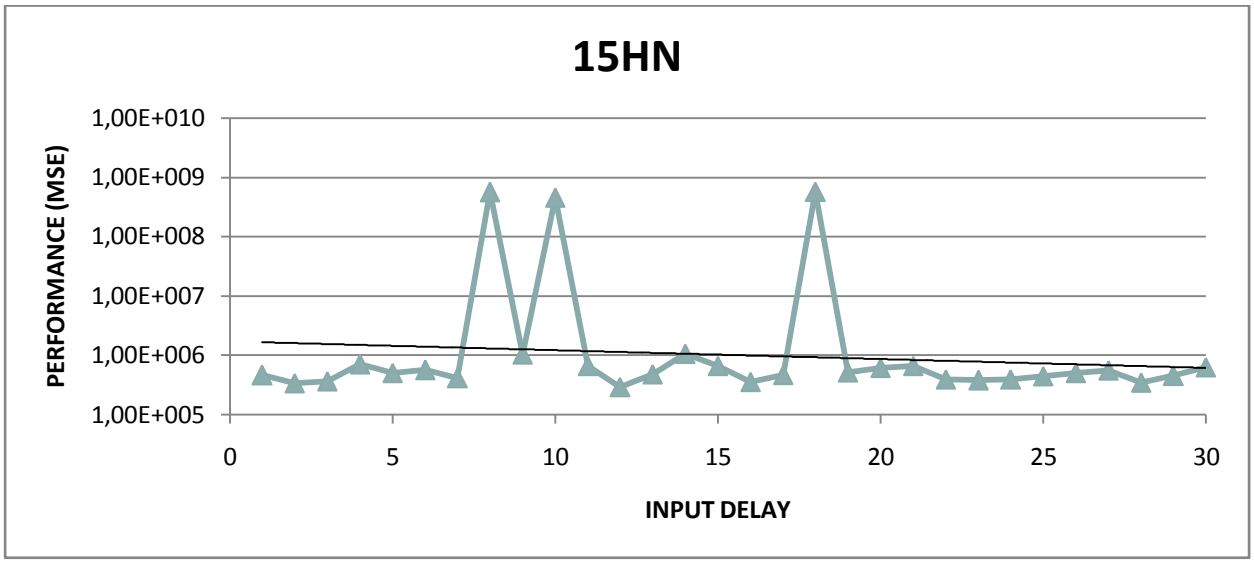
The followings are the MSE vs. delay plots with different number of hidden neurons (HN):



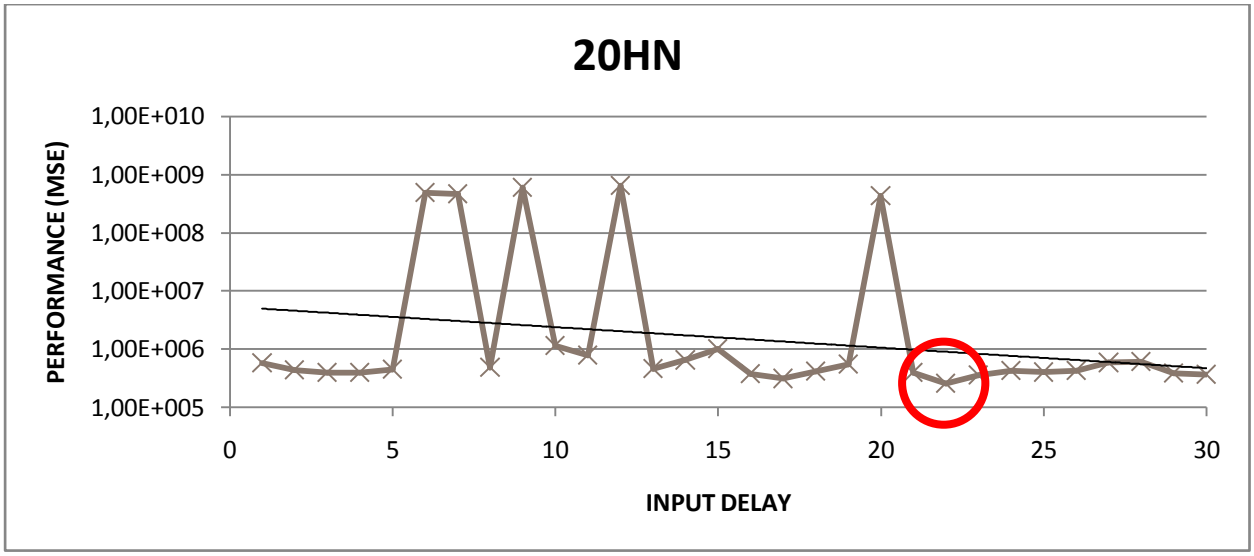
Picture 36 MSE with 5 Hidden Neurons



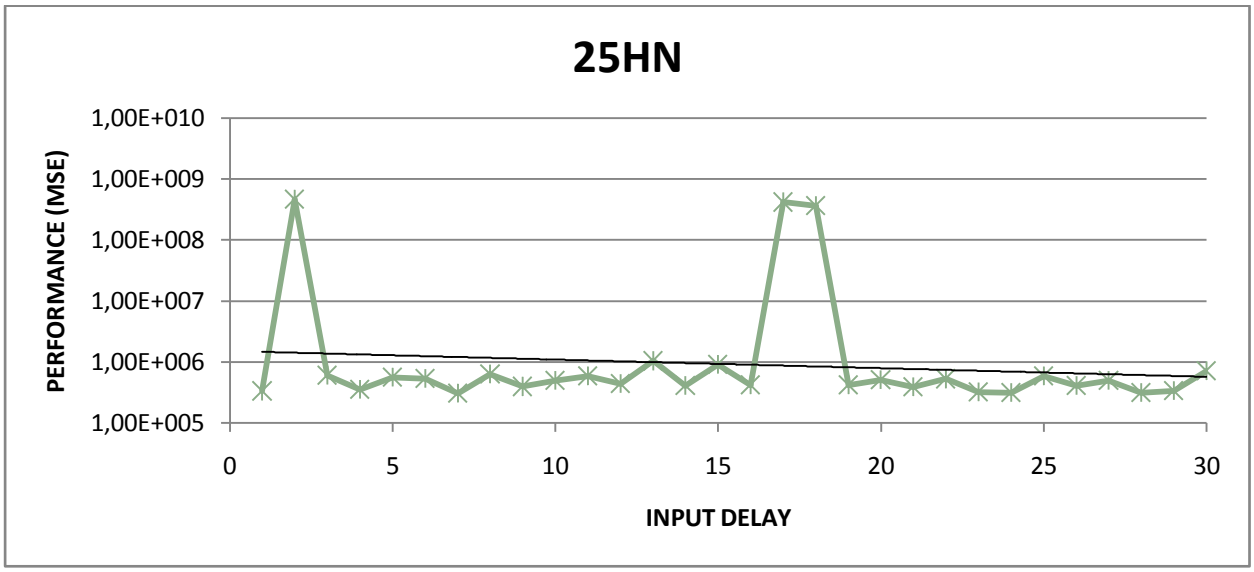
Picture 37 MSE with 10 Hidden Neurons



Picture 38 MSE with 15 Hidden Neurons



Picture 39 MSE with 20 Hidden Neurons

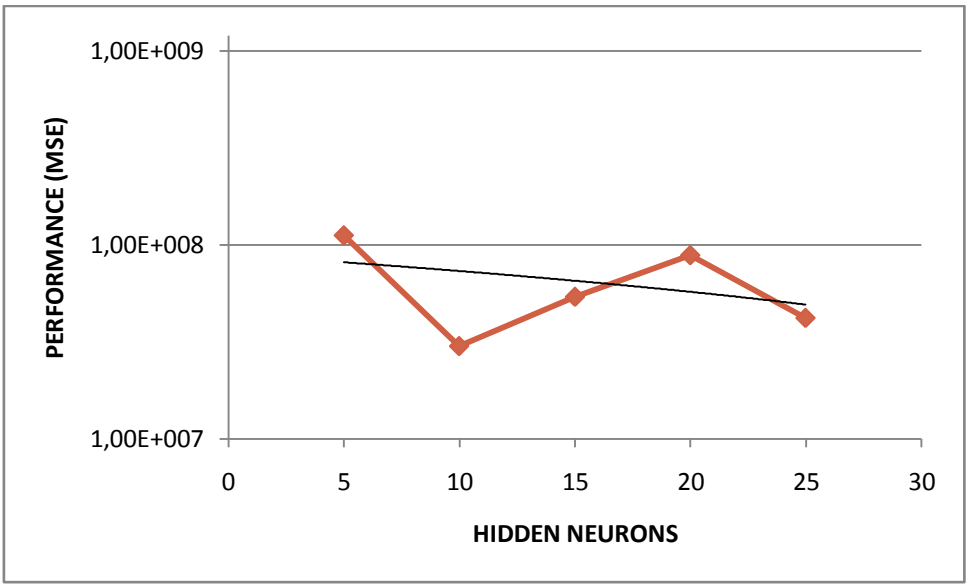


Picture 40 MSE with 25 Hidden Neurons

As it can be seen, until 10 hidden neurons, increasing the input delay increases the MSE, while, after 10, trend lines have negative slope.

HEURISTIC RULE 1:
if we consider more than 10 hidden neurons we can say that "*It's better to have a delay around 22*" (the minimum MSE for the simulation is $2.56 \cdot 10^5$ and corresponds to input delay=22 and hidden layer size=20).

Moreover, it looks like MSEs decreases growing the hidden layer size, as it can be seen in the following plot:



Picture 41 Performance vs Hidden Layer Size (mean values)

In the picture above every point is a MSE mean values calculated from delay 1 to 10. It can be seen that, growing the number of hidden neurons MSEs decrease.

The trend line is negatively sloped that means that MSE mainly decreases growing the number of neurons.

So, a heuristic rule could be "*Use more Hidden Neuron*", but this is known, as it is that more neurons require more computation, and they have a tendency to overfit the data when the number is set too high, but they allow the network to solve more complicated problems.

HEURISTIC RULE 2:
In this analysis we can say "It's better use more than 20 hidden neurons (but not much more!)".

Notice that this last rule is stronger than the first as it's also more computational expensive.

II. Closed Loop strategy

For the last part of the project was asked to evaluate the performance of a neural network that uses closed loop strategy to forecast the number of bike rentals for the year 2012, considering the forecast for 1,2,7,10 and 15 days ahead.

The script ***day6_forecastCL.m*** is pretty simple. After loading data, the main part consists in the automatic generated code needed to create and train the network. This is simply obtained from the GUI tool considering hidden layer size 15, input and feedback delay 20.

Then I modified the last part of the script which was dedicated to the early prediction.
In this part I reselected the same inputs used for the training (piece to piece), and then I gave them to the trained Neural Network in order to obtain the forecasts an study the performances.
To do this I consider 12 predictions (one every 30 days) for 1,2,7,10,15 days ahead.

I stored these results in the file ***day6_forecastCL_MSE.csv,*** then I rearranged in the file ***day6_forecastCL_MSE.xlsx***.

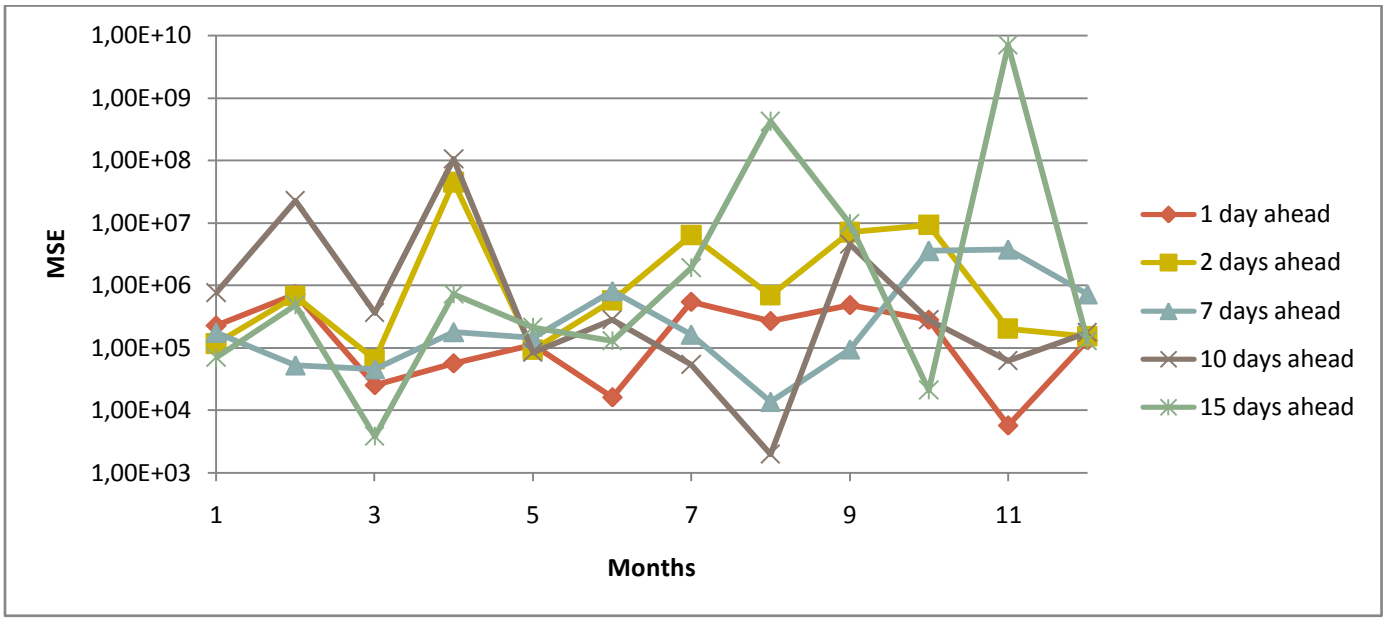
The final results are shown in the following table:

MSE		Months													
		1	2	3	4	5	6	7	8	9	10	11	12	Mean	Variance
Sa the pad	1	2,29E+05	7,28E+05	2,53E+04	5,71E+04	1,11E+05	1,60E+04	5,46E+05	2,68E+05	4,87E+05	2,81E+05	5,70E+03	1,33E+05	2,41E+05	5,53E+10
	2	1,16E+05	6,78E+05	6,53E+04	4,44E+07	9,27E+04	5,73E+05	6,39E+06	6,89E+05	7,05E+06	9,22E+06	2,02E+05	1,51E+05	5,80E+06	1,58E+14
	7	1,76E+05	5,28E+04	4,60E+04	1,79E+05	1,46E+05	8,06E+05	1,62E+05	1,38E+04	9,59E+04	3,55E+06	3,78E+06	7,23E+05	8,11E+05	1,85E+12
	10	7,56E+05	2,26E+07	3,61E+05	1,05E+08	8,47E+04	2,82E+05	5,44E+04	2,00E+03	4,52E+06	2,80E+05	6,23E+04	1,75E+05	1,12E+07	9,12E+14
	15	6,94E+04	4,88E+05	3,86E+03	7,16E+05	2,11E+05	1,28E+05	1,91E+06	4,18E+08	9,65E+06	2,11E+04	6,94E+09	1,30E+05	6,14E+08	3,98E+18

Picture 42 Table of the forecasting MSE

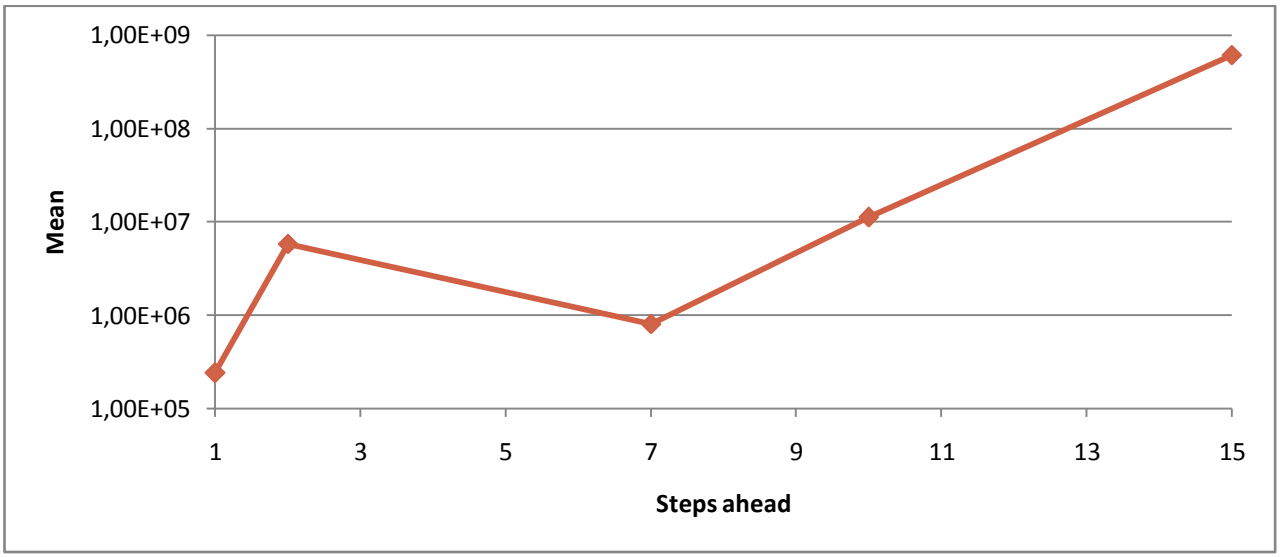
These results says several things about the forecasting.

The first is that there is a larger variability in the MSEs increasing the number of days ahead to whom refers the forecast. And this is quite obvious:



Picture 43 MSE vs. Months

As it can be seen In the following graph, either the MSE tends to increase with the days ahead:



Picture 44 Mean of MSE vs. days ahead

6. Tools

The followings are the software tools used:

- MATLAB 7.12.0 R2011a
- Microsoft Word 2007
- Microsoft Excel 2007
- Libre Office Calc 2014
- PhotoFiltre

7. Conclusions

The aim of this project was to evaluate different solution to develop a Fitting and a Forecasting system. To do this I've considered Neural Network and Fuzzy solutions that MATLAB equips.

The obtained results were then evaluated through the Mean Squared Error performance index, which gave similar results in the case of Neural Network.

For the Fuzzy systems different results were obtained in terms of MSE and other different indices were used to evaluate the performance.

It's needed also, to notice that different results were obtained for the two dataset. This is due to several factors, mainly because of the different size of the two dataset, this implies different precision, different approaches in programming, etc. In particular large size of the datasets implies: greater understanding of the values and less mechanical work, larger precision but longer computing time.
So different approaches were used to find the best tradeoffs between precision and computing time.

Another concept to underline is the statistical meaning of the results. Several of the shown results are the outcome of a set of trials. This was done to find the most meaningful insights.

To sum up the results.
I find that for the fitting MLP and RBF NNs obtain about the same results in terms of MSE: 10^3 for the daily dataset and 10^5 for the hourly dataset.

Instead with FISs I found that with a Mamdani MSE is strictly dependent to the rules that I've set: about 10^7 for the daily and 10^4 for the hourly dataset.

For the ANFIS I decided to normalize the values to obtain results that are less outliers dependent. In this case I obtained errors of about 0.135 for the daily and 0.07 for the hourly. Notice that, due to the normalization, here the maximum value of count is 1, so the error has to be evaluated in function of this.

The forecasting situation is different, what I can say is that with the open loop strategy I found that the MSE is sensible to the chosen input delay and number of neurons. Varying them may change MSE through different order of magnitude.

With the closed loop strategy I observed that the forecasting MSE is sensible to the number of steps ahead of the prediction. In particular increases either MSE order of magnitude and variability of the error.