Proceedings of the CMPUT 551. Student Final Mini-Project. Fall 2017.

Prof. Martha White

Predictions on the number of users at Capital Bike Sharing

Juan Tzintzún

University of Alberta, Department of Mechanical Engineering, Engineering Management.

Abstract

In this project we realized a performance analysis of three learning algorithms to predict the number of users per hour on a bike sharing system in the next 12-hour horizon. The choose of the algorithms was done to overcome specific trends arising in our data. We performed an analysis on the prediction of the algorithms to show the expected advantages and disadvantages of the model in the different seasons, trends and unexpected events. Our objective is to determine which of the algorithms have a better accuracy rate, and under which circumstances. We are also interested about the running time; about the hyperparameters we will delimit them when defining the methodology and the experiments. The project content is as follows, in section one we will introduce briefly about the data set and the previous work related with it, in section two we will introduce the dataset, in section three we will present the algorithms and the reason why we choose those algorithms; in section 4 we present the methodology and the experiments; finally, in section 5 we conclude which algorithm is better presenting the result of our ANOVA analysis.

1) Introduction

The data set under study is related to 2-year usage record of a bike sharing system namely Capital Bike Sharing (CBS) at Washington, D.C., USA. This data set was previously studied by [1] just with a different approach. In [1] the authors aimed to label the data according class events along the year to pattern the number of users in the bike sharing spot. In our Project we would like to study the performance of three different algorithms for the prediction of the number of users of bikes. The data is a historic record of the number of users per hour along a two-year time horizon. Our objective is to determine which of our three algorithms under study is more suitable for the prediction of the future number of users in the next 12 hours.

2) Data set

This data set was retrieved from a similar study about demand patterns on bike sharing systems [1] however while the authors in [1] aimed to predict event labels on the data. In our study we would like to predict the number of users in a new horizon time interval (the interval is of one hour). Additionally, the data set although similar it is not the same, for further details we will present the characteristics of our data set later in this section. As we mention earlier the data set is a record of a bicycle sharing system that present the number of users per hour in a specific spot. The features that will predict our target value (number of users per hour) are variables that we think will influence the target by some amount. Capital Bike Sharing keep track of the number of bike requested per hour at the spot of study, while there are automatic bike rentals at different location spots in the city for purposes of this project we will focus on data of one of these stations where we would like to predict the future number of users on this specific spot. To predict the

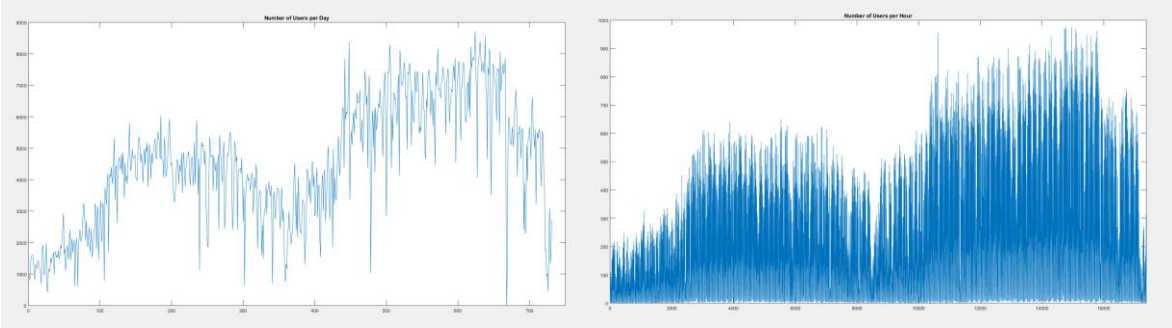
future number of users we need to consider several features that might affect the demand patters. According to previous knowledge about the production and seasonal demand patterns from experts in the area [1]. We conclude that the factors that are likely to influence the demand are:

- Calendar, the data set includes features that represented the year, month, day and hour.
- Type of day, it could be Holiday, weekend or workdays
- Season, the number of bike user while have a seasonal demand for this reason we need to consider the 4 seasons as a feature to predict future outputs.
- Weather State, another obvious consideration when trying to determine which features can capture the relation between the number of users is the weather conditions. This are summarized at table 1.
- Another consideration is also the temperature, the humidity and windspeed. A difference to the weather conditions. The attributes of temperature, humidity and windspeed try to capture the weather sensation on the users while the weather state is more categorical value that represent the initial perception by the user according to what can see.

Table 1 Features

Feature	Range	Type	Explanation
Year	(0:2011,1:2012)	Integer	
Month	(1 to 12)	Integer	
Day	(0 to 6)	Integer	0: Monday, 1: Tuesday,…
Hour	(0 to 23)	Integer	
Season	(1 to 4)	Integer	1: springer, 2: summer, 3: fall, 4: winter
Holiday	(0,1)	Binary	Whether is holyday or not
Working Day	(0,1)	Binary	Whether the day is neither weekend nor holyday
Weather State	(1,2,3,4)	Integer	1: Clear, Few clouds, Partly cloudy, Partly cloudy 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
Temperature	[8:39]	Float	Normalized temperature in Celsius. The values are derived via (t-t_min)/(t_max-t_min)
Humidity	(0:1)	Float	Normalized temperature in Celsius. The values are derived via (t-t_min)/(t_max-t_min)
Windspeed	(0:1)	Float	Normalized temperature in Celsius. The values are derived via (t-t_min)/(t_max-t_min)
Casual Count	-	Integer	Number of casual users by hour
Member Count	-	Integer	Number of registered users by hour
Total Count	-	Integer	Total count of users

We have 17,379 samples and 13 features. Our target is the total count. It represents the number of users per hour, in picture 1 (a) and 1 (b) you can see and time series representation of our data.



3) Machine Learning Algorithms

The selection of the algorithms is based on, we know some algorithms will perform better in certain scenarios, so we would like to choose algorithms that are good for specific cases on our input space.

K nearest neighbors Regression, the K-NN regression is a non-parametric model that predicts according to the k nearest neighbors in the dataset we are studying. We will use this algorithm to find the nearest pattern, in our historic data, to our current trend. This algorithm is useful in situations where we already know what happen in similar situations. However, one disadvantages of this approach is the inefficiency of the algorithm when dealing with large train sets. For example, in our data set we would have some holydays days in which we would expect an augmented number of users. Given that we have a two-year period of data for analysis we expect to have knowledge about what happened before in a similar scenario, generally K-NN regression can be interpreted to find the nearest pattern in our data history to the current pattern.

Neural Network, neural networks are a learning representation in which we will have a connection between neurons, and some weights associated with these connections. These neurons consist on a series of inputs, activation functions and outputs that will transform some input data from one initial domain into another domain that more accurately represent our target prediction. We choose Neural Networks to capture the seasonality on the demand. Authors in [1] mention that Neural Networks works well for these seasonal patterns since they are good at capturing regularities within a set or patters. It has been widely used before for forecasting and time-series prediction [1]. As you can visualize in figure 1(a) and 1(b) the number of users clearly present patterns in the data, obviously seasonal pattern exists in the bike series demand, both, in an hour basis and day basis. The idea behind choosing NN is to capture this seasonality.

Support Vector Regression, we choose to use support vector regression because it has some advantages. For example, since the SVR depended only in a subset of the training data we can expect it generalizes better than the two previous algorithms. The demand has a lot of noise that is caused by different events in the timeline caused by irregular events such as holydays, concerts, parades, accidents, etc. We can expect better predictions using SVR s when dealing which these irregular events. [1]

4) Methodology and experiments

<u>Objective</u>

The objective of the experiment is to evaluate each algorithm over the data set of bike users in the Capital Bike Sharing System. We choose some time ranges to evaluate how each model behaves under different situations. For the methodology we refer to picture 2. First, we will divide our dataset, since our problem

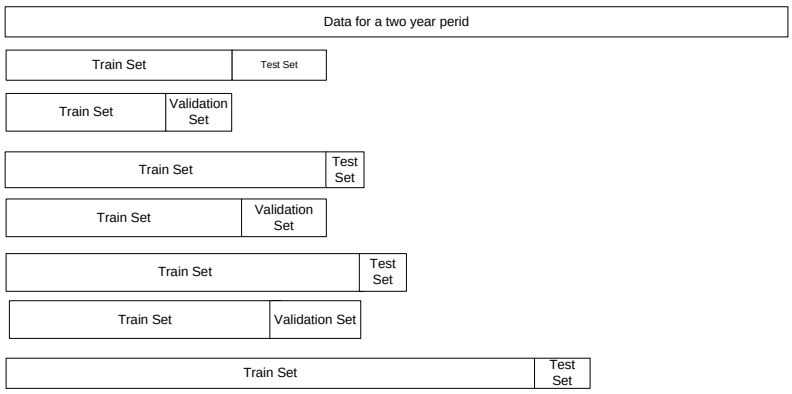
is a time series analysis we must have special considerations on this stage. Once we have split the data set, we will delimitate the hyperparameters of our algorithms, then we will optimize our model to obtain our regressor, by continuing the process in picture 2. We will make a prediction using our regressor and the test set. The we will evaluate our model under different performance measures. We will carry out 30 experiments and then we will conclude which algorithm is better using and ANOVA analysis.

Data Split

Normally when applying machine learning algorithms, we would split our data into a training / validation and test set, and then we would use cross validation to train over all the subsamples to test every model combination, when we use cross validation we are assuming there is not relationship between the observations and the time. Even that our algorithms assume stationarity in the data, the idea of randomly sampling to create train and test sets is not appropriate because we may have the case when we are trying to predict future observations using information about data even further in the future. Another complication arises for example, our data may show that after holydays the number of users is low, but if you sue random sampling we can loss this information. When we split our data, we must respect the temporal order of the data for time series forecasting there exist some approach call walk forward validation. Walk forward validation (WFV) is an approach suitable or our problem [3]. For WFV we need to define two parameters, the minimum number of observations, this is the minimum number of observations we need to train our model, the second parameter is the sliding or expanding window, that is, we need to decide whether we will train our model on all the data available or only using the most recent observations. In our case we will be training in the whole data set and updating over additional information when it is available.

The split of the data is described as follows.

1. First, we will split our data in two years and we will train over the first year. That means we will define our minimum number of observations to be one year.
2. The model makes a prediction for the next time step. Our sliding window will be 12 hours. We will be predicting for the next 12 hours.
3. The prediction is stored or evaluated against the known value.
4. The window is expanded to include the known value and the process is repeated (go to step 1.)



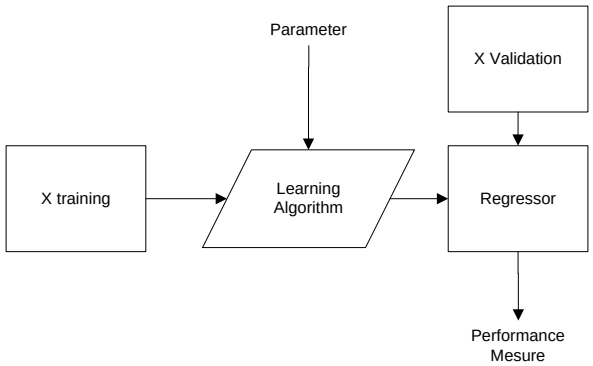
We will split the data into train set, and a test set over the first year. According to a 70/30 percent proportion respectively. Once we have our train set once again we will split using a 70/30 percent proportion to obtain the train set and the validation set. The validation set will be sued for the hyperparameters tuning.

Hyperparameter Selection

Every machine learning algorithm have different setting parameters, some of them can be learned when optimizing to minimize our cost and for some other parameters we need to be specified before running algorithm. The parameters we will test to measure the performance of our algorithm are summarized in the following table. For the metric distance we will test on the Manhattan distance, we choose this metric because is a good metric for real value data, however, since most of our data is categorical we also want to try Hamming distance, and Canberra distance since we know this metrics performs better on integer-valued Data. The center we choose to set our center to be equal to the number of features we have, that means our center number will fourteen. For hidden layers we will test 8, and 16, 32, and, we will leave the epoch number constant as 100 because we are not interested on add more training time to the algorithm and avoid possible overfitting. For the hidden layers we will use a sigmoid activation function and for the output layer a linear activation function.

ML algorithms	**Hyperparameters**
K Nearest Neighbors	**K**=14 **Metric distance**= Hamming distance, Canberra distance, Manhattan distance
Neural Network	**Hidden layer size**= 8, 16, 32 **Epochs**=100
Support Vector Regression	**Kernel**= Gaussian Kernel, Lineal kernel **K**=14

To determine the hyperparameter selection we will train our algorithm in a training set using a set of the parameters defined before, and then use a validation set to test our regressor. This will output a performance measure, (the performance measure will be defined in the following sections). We will select our hyperparameters base on the best performance measure we obtain, less prediction error. The parameter selection will be used along the next series of predictions and held as constant. In figure you can see a diagram of the approach we follow for parameter selection.



Learning

The objective of this experiment is to evaluate the performance of each model over our bike data set. For our experiments we would like to evaluate our algorithms according to the accuracy of the prediction and the running time. Our set consist on 17380 samples and we would like to split our data in the first year of samples that will be use as training basis. We will predict over the next 24 hours period and compare, to later update our data set to add the new information available and try to predict for the next horizon interval.

Performance measure

The selection of the performance measure is one the most important thigs when evaluating different learning algorithms since it will reflect how accurate our predictions are. For our experimental setting we will used three performance measures. Mean Absolute Error (MAE), Mean Absolute Percentage Error and Root Mean Squared Error (RMSE).

- MAE is defined as follows MAE=$\frac{1}{n}\sum_{i=1}^{n}\|\hat{y}-y\|$. This error measures the absolute deviation from our prediction and the real target. We would like to reduce this error as much as possible.
- MAPE is defined as follows MAPE=$\frac{1}{n}\sum_{i=1}^{n}\left\|\frac{\hat{y}-y}{y}\right\|$ as MAE this error measures the deviation forms the real value but as a percentage value. One drawback of this performance measures occurs if you have output that are equal to zero. However, in our case it does not occur, and we choose this performance measure because the error does not depend on the scale of our data set.
- RMSE is defined as follows RSDE =$\sqrt{\frac{\sum_{i=1}^{n}(\hat{y}-y)^2}{n}}$

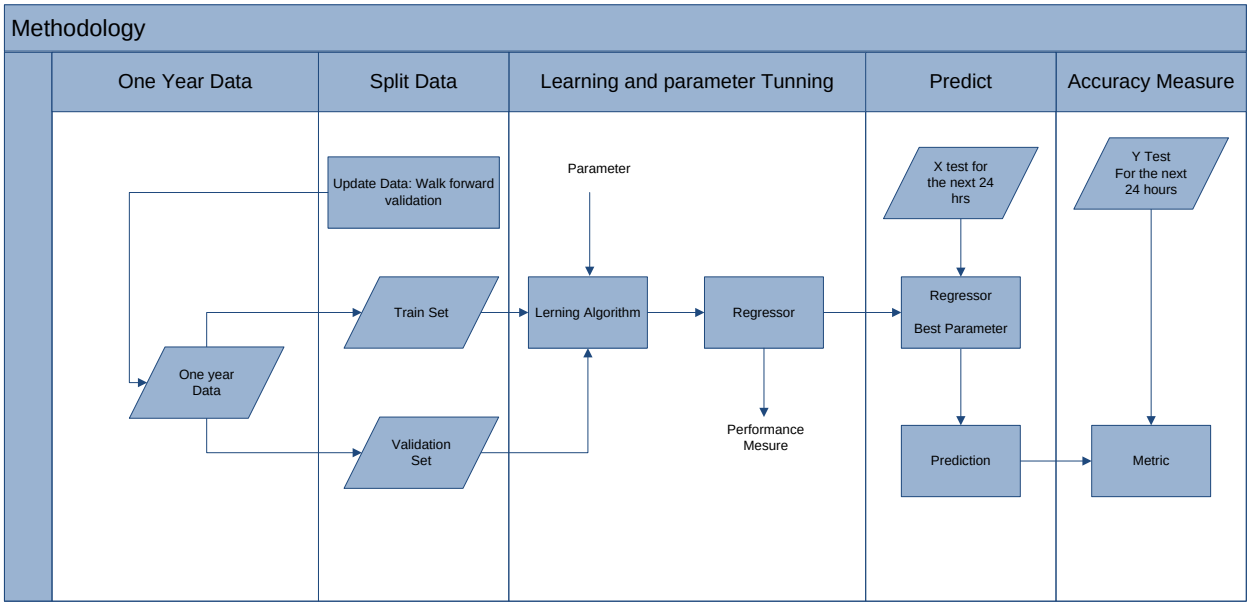
Experiment Settings

In this section we will describe the methodology we use to evaluate the algorithms. In the previous sections we presented several features relevant for this section, now we will put everything together to present the step by step procedure we followed when performing the analysis. We carried out our experiments using python 2.7 in a personal computer of 2.7 GHz and 8 GB of RAM with an Intel Core(TM) i5-6400 processor and utilizing scikit-learn library to program the learning algorithms and predict our data.

The procedure is described as follows. One can see in figure a diagram representation of the methodology.

1) First, we will train our learning algorithm on the first year of data.
2) The data split is done as we mention before in the data split section, we will use the first-year data as training set. Our training set is divided again to obtain a validation set.
3) We will test on the validation set with all the hyper parameters to obtain different prediction and errors.
4) According to the most accurate combination of regressor and parameter settings we will set those parameters as constant for the rest of the experimentation.
5) Using the regressor and parameter combination selected before we will predict on the test set, the test set will be the next twenty-four-hour data.
6) Once we got a prediction we will compute the error on our prediction using the performance measure we define previously in this work.

7) After we obtain the data we will perfume statistical significance test to conclude either the algorithms have the same expected error. One of our hypothesis presented before is that the algorithms will have different performance in different scenarios, we will test them in specific time intervals to show they prediction accuracy in different time intervals.



5) Conclusions

Data Analysis

In here we present the average error for the different parameters settings. We select the parameter setting with the lowest training error. This parameter setting is mentioned in their respective section, the settings were used when we perform our predictions.

	Canberra Distance	Draycurtis Distance	Manhattan Distance
MAE	18.9823092	16.13784736	16.54849315
MAPE	31.31496536	46.63150875	45.21525047
RMSE	30.77834761	26.13434154	26.86198316
Mean	27.02520739	29.63456588	29.54190893

	RBF Kernel	Linear Kernel
MAE	98.62227885	0.059807061
MAPE	98.62227885	0.059807061
RMSE	132.6035625	0.065124721
Mean	109.9493734	0.061579614

	8 hidden	16 hidden	32 hidden
MAE	0.266944444	698.2063793	0.4294079
MAPE	0.277238432	729.6278997	0.3590281
RMSE	0.22968399	723.3605405	0.3612518
Mean	0.257955622	717.0649399	0.3832293

Statistical significance Test

In this section we compare the average error for all the algorithms a perform a significance test to try to conclude if the errors are statistically different when we try to predict in the first time window.

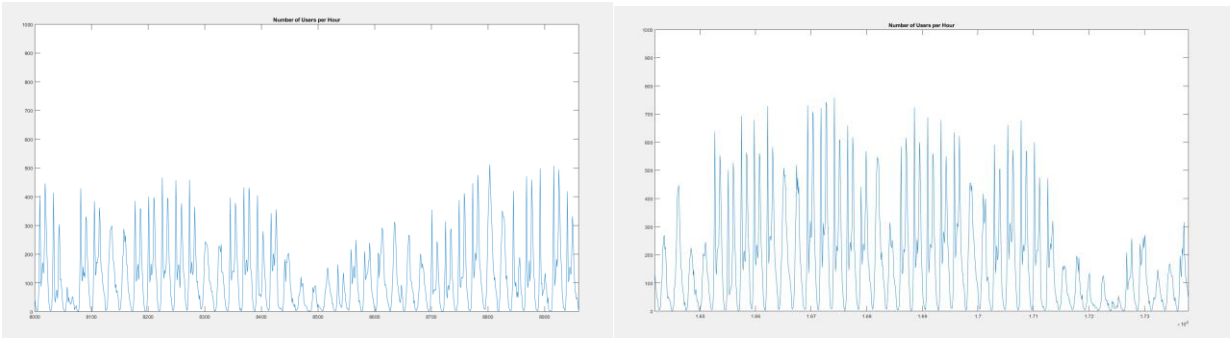
Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
X1	157.53	1	157.53	0.7	0.4491
X2	1455.3	1	1455.3	6.49	0.0634
X3	193.06	1	193.06	0.86	0.4059
Error	896.73	4	224.18		
Total	2702.63	7			

We can conclude that the algorithm expected error are not statistical different when we predict for the first next 24 hours interval.

Other Considerations to notice.

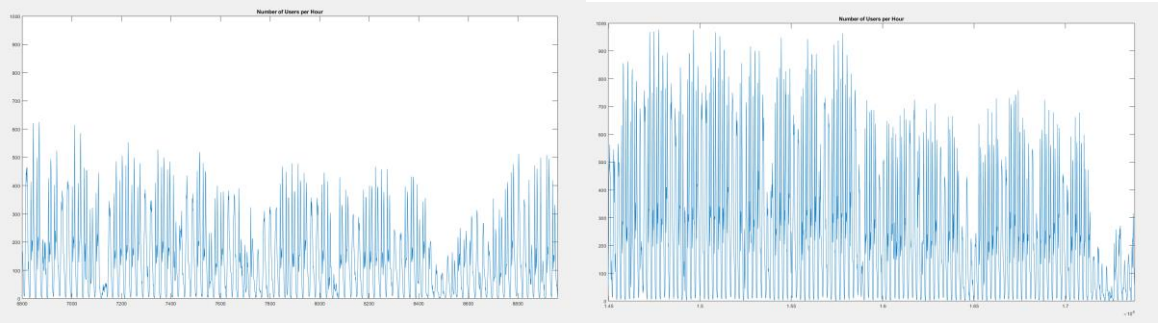
Case 1: SVM will generalize better that KNN and NN. We expect the error be lower when we have unexpected events in our time series.

In this case we will analyze over the error we obtain with the different parameters to try to answer our hypothesis about that support vector machine will outperform the other algorithms in situations where we found unexpected observations in our data. To have empirical evidences of this statement we need to find the error for the SVR to be lower than the other algorithms and we need to have that the expected error is statistical different to the error obtained for the KNN and NN. We will analyze the predictions in a time window that correspond to the same period for the year one and two. To be more specific the exact time is the month of December. As you can observe in picture one it corresponds to the number of users around January 12 th (the data point in the middle of every picture corresponds to January 12 th). As you can see, for the second year (picture number two) the number of users is significantly different around that dates, this might be because in the second-year holydays did not end until later in the month for the second year. An unexpected event in which our SVR vector should be performing better according to the hypothesis.

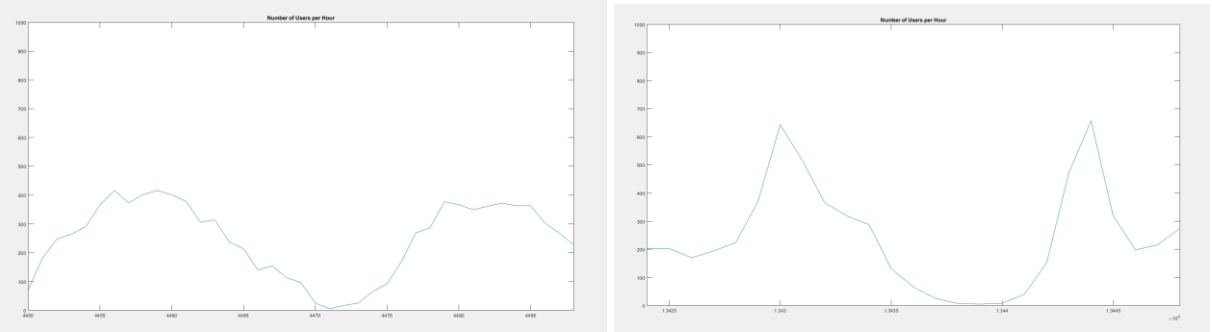


Case 2: NN will work better for seasonal patterns in the timeseries We expect the error be lower when we have unexpected events in our time series.

For this case we will analyze the error prediction when there is seasonal change in the number of users of bicycles, for example when the winter start. We will analyze the data from corresponding to fall and winter.



Case 3: KNN will perform better when we have already knowledge about what happen in similar situations. We expect the error be lower when we analyses the demand for the Independence Day.



References

[1] Fanaee-T, H., & Gama, J. (2013). Event labeling combining ensemble detectors and background knowledge. Progress in Artificial Intelligence, 2(2-3), 113-127. doi:10.1007/s13748-013-0040-3

[2] Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. International Journal of Forecasting, 22(4), 679-688. doi:10.1016/j.ijforecast.2006.03.001

[3] https://machinelearningmastery.com/backtest-machine-learning-models-time-series-forecasting/

[4] http://scikit-learn.org/stable/