Luca Sinanaj
Mat. 844540

Streaming Data Management and Time Series Analysis Project

Abstract

The project aims to study, perform an exploratory analysis and finally forecast a daily time series representing the average number of days needed to close the requests that were closed that day. Methods from three families: ARIMA, UCM, and Machine Learning, were employed in the project's implementation and evaluated using the MAE metric.

Project's goals:

1. Provide forecasts for all the days from 2015-04-01 through 2015-11-07.
2. Use three different methods: an ARIMA model, a UCM model, and one from the Machine Learning family.
3. Select, through validation processes, the most promising algorithms, narrowing down the choice to the "best" three, one from each family. The goodness or badness of a model was assessed using the MAE (Mean Absolute Error) metric, which mathematically represents the distance between the predicted and actual values.

SUMMARY

1. Dataset and pre-processing

The dataset consists of three columns:

- 'date': represents the date in the format yyyy-mm-dd
- 'weekday': a string containing the day of the week corresponding to the date
- 'ave_days': average number of days needed to close the requests that were closed on that day.

The first step was to verify the temporal continuity of the series by checking for the absence of null values within the data. Regarding the 'ave_days' column, we noticed there are 202 null values which will be replaced with the value 1 (initially zero, but since a logarithmic transformation will be applied later, 1 was preferred). There are no null values for the other two columns. Duplicates within the historical series were also checked for and found to be absent.

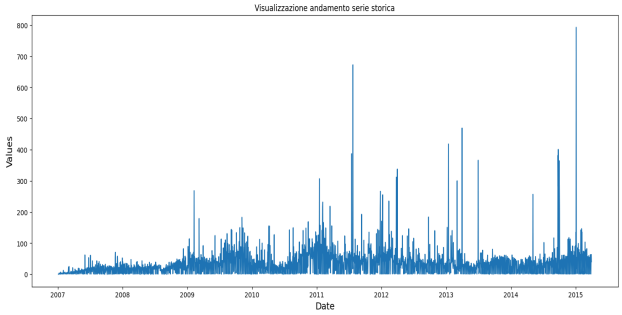
Additionally, several columns were added such as *Year, Month, Day, Month_Name, Num_DayofYear, Num_DayofWeek, Num_WeekofYear and Quarter,* in order to facilitate and continue the exploratory analysis.

2. Exploratory analysis

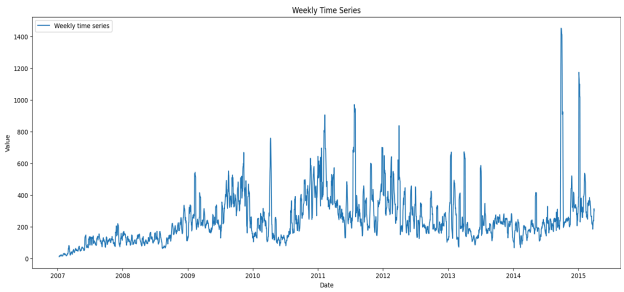
Before starting to analyze the time series, it is necessary to add a note regarding the missing values, which are distributed as follows:

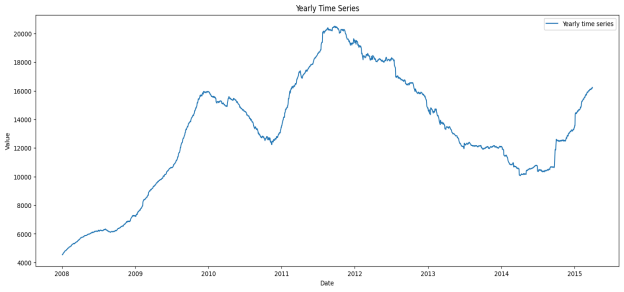
- Friday 4
- Monday 20
- Saturday 6
- **Sunday 167**
- Thursday 4
- Tuesday 1

We can notice that 167 out of the missing data points occur on Sundays. This observation can be logically justified by the fact that most companies do not operate on Sundays. This observation will be important during the model construction phase.

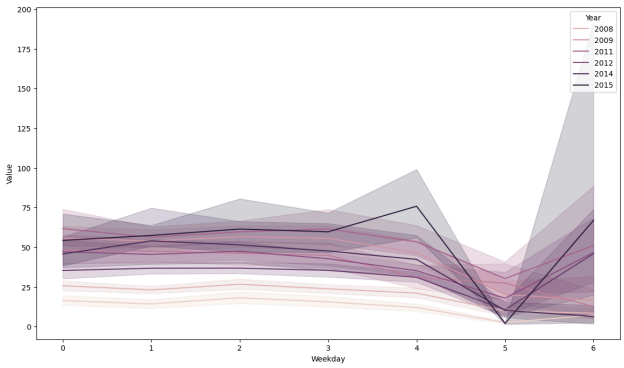


This first graph, the overall trend of the series is shown in its entirety. In order to get more details we decide to plot the same graph having weekly and yearly aggregated data.





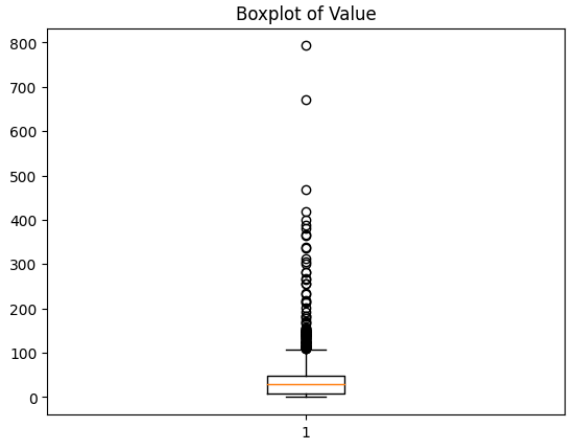
From the above graphs, we cannot determine a particular seasonal trend. However, we can observe a generally increasing trend between 2008 and 2012, followed by a decreasing trend between 2012 and 2014, but with a recovery in 2015. One striking observation from these initial graphs is the presence of many high outliers that deviate from the 'mean'.



This graph shows the trend divided by year of the ave_days value during the days of the week. We can observe an initial decreasing trend towards the weekend. It also shows the high variability of the data, particularly for the year 2015.

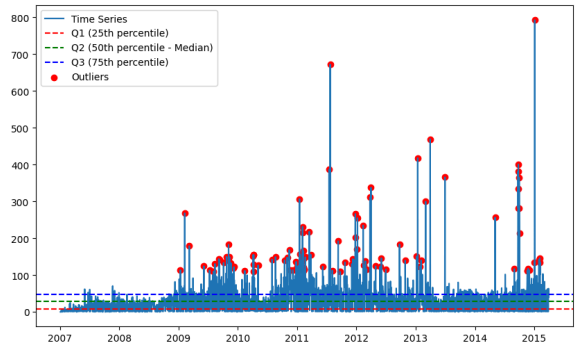
3. Outlier

In order to further analyze the outliers, Box Plots have been generated.



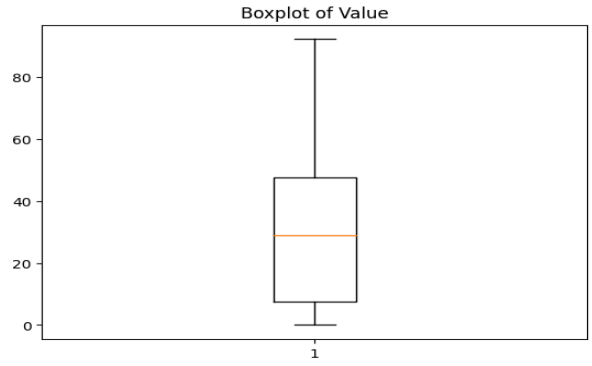
We can notice, as suspected, the presence of numerous outliers. The boxplot appears very compressed, with Q1 being equal to 1 and Q4 around 100. However, the values of ave_days extend up up to almost 800.

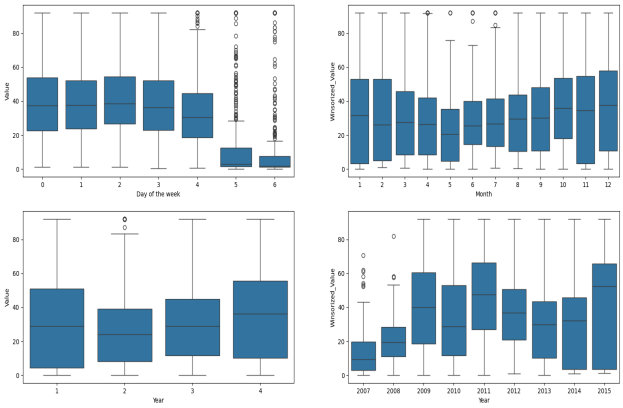
So, we identify the outlier values.



They represent 3.39% of the entire dataset. In order to handle these values, the **scipy.stats.mstats.winsorize** library was used, which sets *'The (limits[0])th lowest values to the (limits[0])th percentile, and the (limits[1])th highest values to the (1 - limits[1])th percentile'.*

It is then possible to observe how the distribution of the data changes.





4. Stationarity Analysis

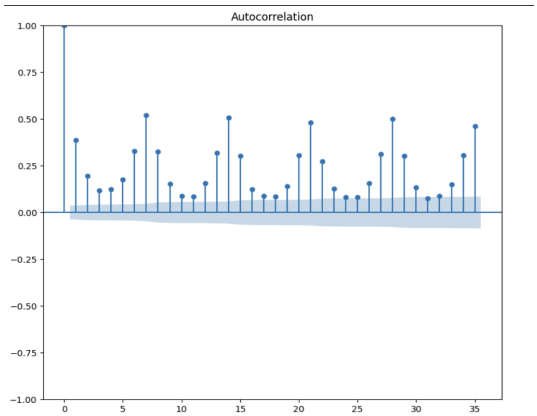
After the preprocessing steps, you conducted both the Augmented Dickey-Fuller (ADF) test and the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test to assess the stationarity of the time series. Stationarity essentially means that the statistical properties of the time series, such as mean and variance, remain constant over time. This step is crucial because, in order to perform accurate forecasts, it's important to work with stationary data, as non-stationarity can lead to unreliable predictions.

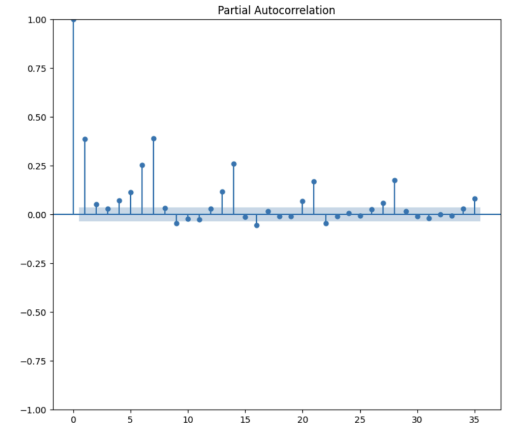
The ADF test checks for the presence of a unit root in the time series, which indicates non-stationarity. On the other hand, the KPSS test evaluates whether the time series is level or trend stationary.

The Augmented Dickey-Fuller (ADF) test yielded a p-value of 0.003312, leading to the rejection of the null hypothesis of non-stationarity. This suggests that the time series is indeed stationary.

On the contrary, the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test resulted in a p-value of 0.01, prompting the rejection of the null hypothesis of stationarity.

Given these conflicting results, further analyses were conducted on the autocorrelation and partial autocorrelation plots (ACF and PACF).



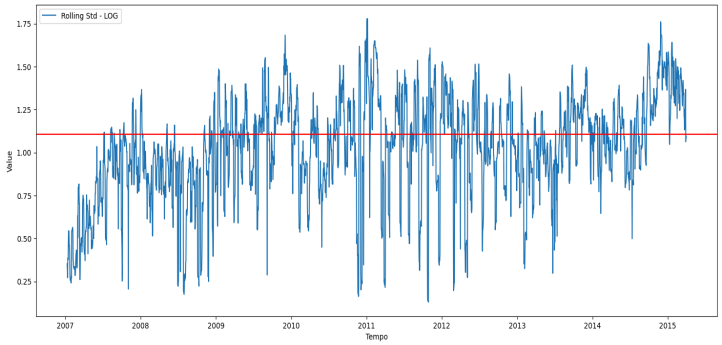


The Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots reveal notable correlations at lags of 7, 14, and 21 in the ACF plot. Additionally, the PACF plot highlights correlations specifically at a lag of 7. These findings strongly imply the existence of a weekly seasonal pattern within the data.

5. Transformations

In order to make the series stationary, logarithmic transformation, Box Cox transformation, and daily differencing were applied. Regarding the differencing technique, it was discarded as it returned negative values, which would not have been useful for subsequent analyses. Logarithmic and Box Cox transformations rendered the series stationary on average but not in variance.

Therefore, the logarithmic transformation was chosen based on the results of the ADF and KPSS tests.



ADF Test:

- ADF Statistic: -4.457063522740238
- p-value: 0.0002349168849814916

The ADF statistic is much more negative than the critical values, indicating that we can reject the null hypothesis of non-stationarity. Additionally, the p-value is very low (0.0002349168849814916), confirming evidence in favor of the stationarity of the series.

KPSS Test:

- KPSS Statistic: 5.067869
- p-value: 0.010000

The KPSS statistic exceeds the critical value at the 10% level, suggesting that we can reject the null hypothesis of stationarity at the 10% significance level. This suggests that the series may be non-stationary. However, the p-value is 0.010000, which is less than

the 5% significance level, suggesting that we may not completely reject the null hypothesis at the 5% significance level.

6. Dummy variables and train - test split

Before starting the construction of the series forecasting models, the dataset was modified in order to model holidays (days when it is presumed that companies do not work or work less). Dummy columns 'Is_Sunday', 'Is_Christmas', 'Is_Easter', and 'Is_Last_of_Year' were added accordingly. As anticipated in the exploratory analysis, Sundays play an important role in our series, which is why different weights were assigned to these variables, increasing the weight of the 'Is_Sunday' variable.

The dataset was then divided as follows:

- Train: 2007-01-04 to 2014-08-23

- Validation: 2014-08-24 to 2015-03-31

This was done to maintain the same forecast horizon as the one between the complete time series and 2015-11-07, which is the project's objective.

7. ARIMA

ARIMA (AutoRegressive Integrated Moving Average) is a statistical model commonly used in time series forecasting because it combines the ability to capture relationships among past data and to handle the presence of seasonality and trends in time series.

ARIMA utilizes an autoregressive component (AR) to capture the relationship between a value in the series and its past values, a moving average component (MA) to handle the presence of random forecasting errors, and an integrated differencing component (I) to address the presence of non-stationary trends in the series.

In particular, it was used SARIMAX, is an extension of the ARIMA model that includes components to handle both seasonal variations and exogenous variables.

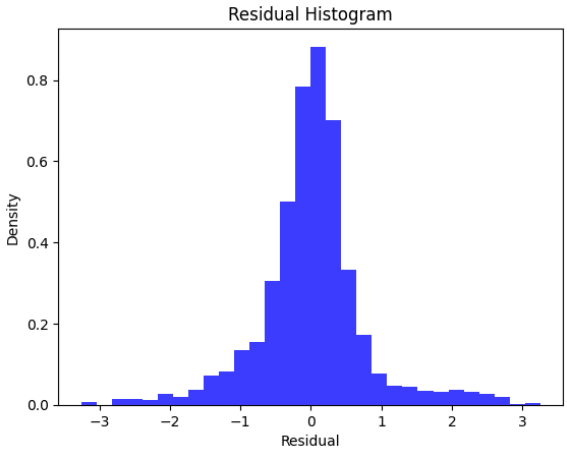
Different models were determined, and their parameters were extracted using the auto_arima model. Starting from the best one, they were then modified and evaluated based on the MAE.

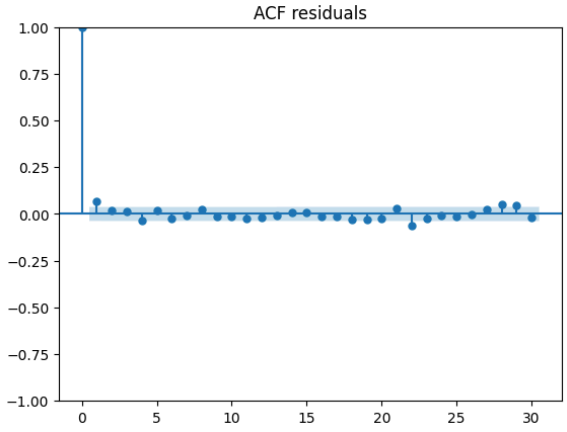
The general idea was to increase the AR and MA orders since the number of days needed to close the requests that were closed that day could depend closely on the closures of previous days.

Model	MAE
(1, 1, 1) (1, 0, 2)[7]	0.594
(1, 1, 1) (2, 0, 1)[7]	0.616
(2, 1, 3) (2, 2, 2)[7]	0.573
(2, 1, 3) (2, 2, 3)[7]	0.572

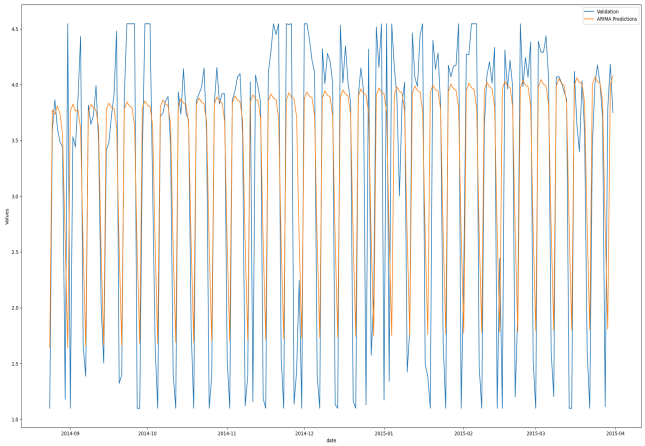
Based on the MAE values, Model 4 has the lowest error (MAE = 0.572), followed closely by Model 3 (MAE = 0.573), then Model 1 (MAE = 0.594), and finally Model 2 with the highest error (MAE = 0.616).

Therefore, Model 4 appears to be the best-performing SARIMAX model both for MAE and residual distribution which follow a normal distribution.

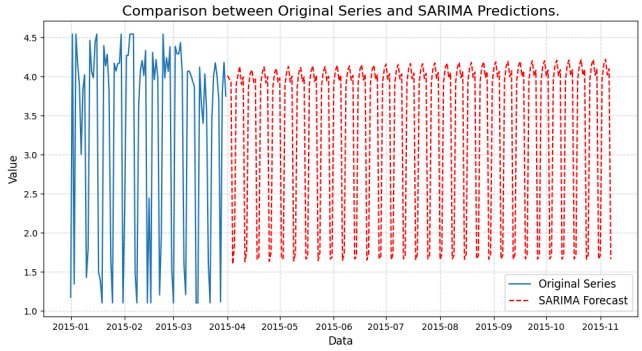




In the next figure we can see the difference between the prediction of the best ARIMA model and the original value on the test set.



After selecting the best Arima model, forecasts were made for the period from 2015-04-01 to 2015-11-07, representing the next seven months of the time series.



We can notice a slightly increasing trend.

8. UCM

The second family of models used is that of UCM (Unobserved Components Model).

UCM is another statistical model commonly used in time series forecasting because it can capture both the trend component and the seasonal component present in the data.

UCM uses a decomposition-based approach to identify and treat separately the various components present in the data, such as

trend, seasonality, cyclicality, and random noise. This model is highly flexible and adaptable, and it can be used in many different contexts. It is capable of providing accurate and reliable forecasts even in the presence of data with complex structures.

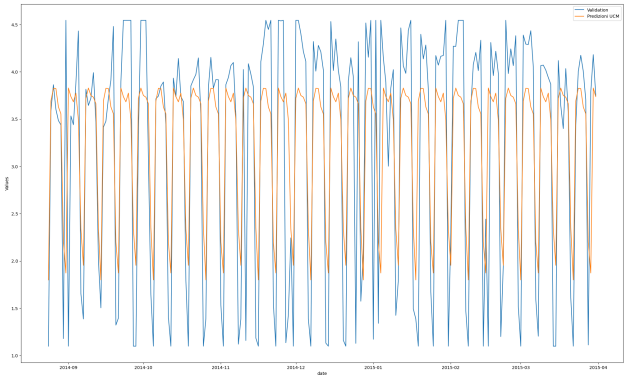
For the implementation of these models, it was decided to uphold the same partitioning of the time series into training and test sets as was done for the ARIMA models. Throughout the refinement process, multiple models were trained with diverse level components, including:

- 'llevel' (local level)
- 'rwalk' (random walk)
- 'rwdrift' (random walk with drift)
- 'strend' (smooth trend)

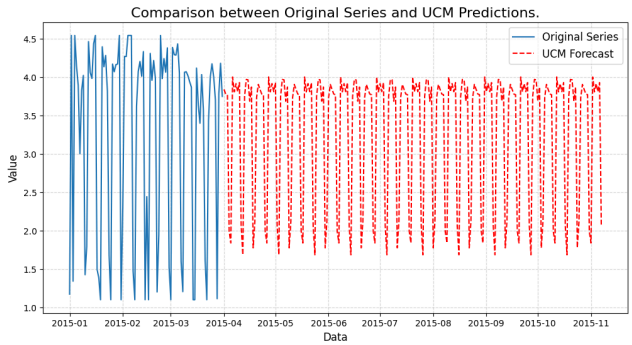
In the subsequent report, the top model for each component are showcased.

Level component	MAE
llevel	0.607
rwalk	0.833
rwdrift	0.883
strend	0.637

The local level component allows the baseline to vary over time, capturing short-term fluctuations or changes in the underlying level of the data. It is a flexible component that can adapt to changes in the data without assuming a constant or linear trend.



In this figure we can see the difference between the prediction of the best UCM model and the original value on the test set.



After selecting the best UCM model, as for ARIMA model forecasts were made for the period from 2015-04-01 to 2015-11-07, representing the next seven months of the time series. A more stable trend can be noticed.

9. Machine Learning

Initially, an approach with few regressors was tried, but it did not lead to good results. Therefore, through a trial and error process, the regressors to be included in the models were chosen to achieve satisfactory MAE.

Firstly, features derived from the date in the preprocessing phase were included, such as *'Num_DayofYear', 'Month', 'Num_DayofWeek', 'Num_WeekofYear'*. Additionally, the dummy variables presented earlier in Chapter 6 were added, with different weights assigned to each of these features, focusing particularly on the value of Sunday.

Infine sono stati sviluppati tre diversi modelli di Machine Learning e per ognuno di essi sono stati individuati i migliori iperparametri tramite **Grid Search.**

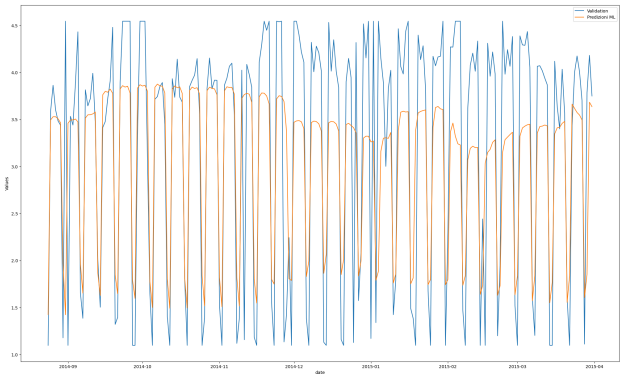
XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.It sequentially builds multiple weak learners and combines them to create a strong learner.It uses a gradient boosting framework which makes it highly efficient and scalable.

LightGBM is a gradient boosting framework developed by Microsoft that focuses on leaf-wise tree growth rather than level-wise tree growth. It is designed for distributed and efficient training of large-scale datasets.Uses a novel technique called Gradient-based One-Side Sampling (GOSS) to filter out the data instances for finding a split value while retaining the same information gain. It also employs Exclusive Feature Bundling (EFB) to reduce memory usage and speed up training.

Random Forest uses an ensemble of decision trees and evaluates the majority of predictions from the trees to obtain the final prediction. This model is very flexible and can handle multiple relationships between variables.

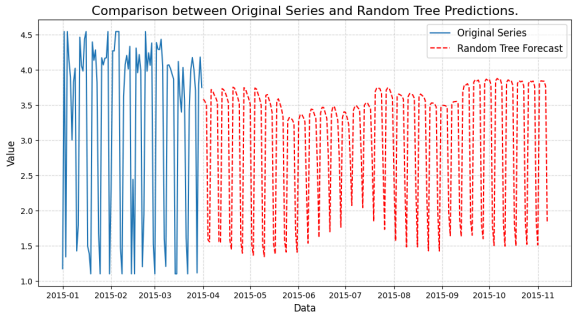
Model	MAE
XGBoost	0.650
LightGBM	0.595
Random Forest	0.583

As we can see Random Forest results to be the best ML model. In this figure we can see the difference between the prediction of its model and the original value on the test set.



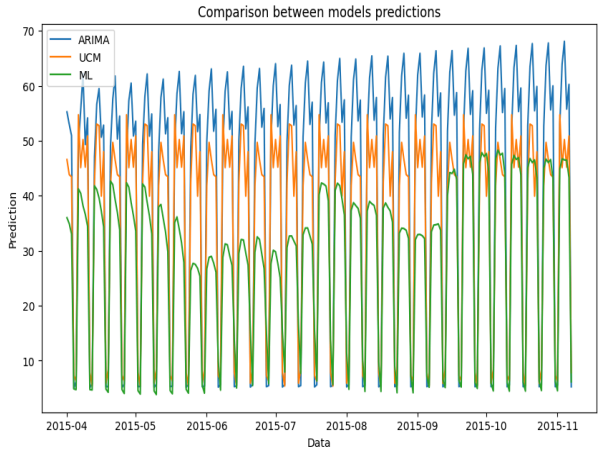
Thanks to the application of weights we can notice how this model is more accurate of

values equal to 0.



As before, model forecasts were made for the period from 2015-04-01 to 2015-11-07, representing the next seven months of the time series.

10. Forecast



The results of the best models from each prediction family were finally combined in order to compare them using a test set.
The best models result to be:

ARIMA (2, 1, 3) (2, 2, 3)[7], which after training from 2007-01-04 to 2014-08-23, resulted in a MAE of 0.572 on the validation set.

UCM with a level component local level, which after training from 2007-01-04 to 2014-08-23, resulted in a MAE of 0.607 on the validation set.

Random Forest with GridSearch to find hyperparameters, and feature weights which after training from 2007-01-04 to 2014-08-23, resulted in a MAE of 0.583 on the validation set.