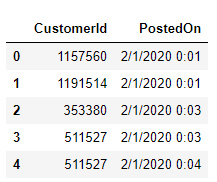
**Comparison of different forecasting models**

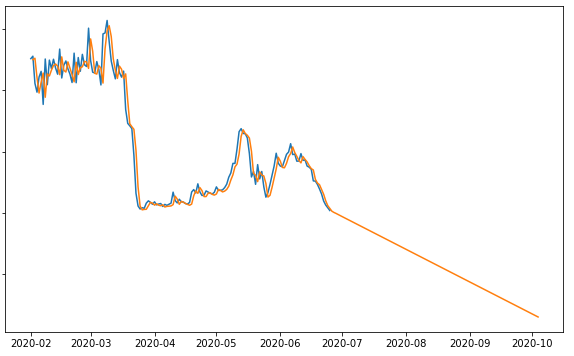
**Dataset:** The dataset consists of 2 columns, CustomerId and PostedOn.

CustomerId is the unique ID of the customer and PostedOn is the date on which the customer applied for the coupon. Based on this dataset we have tried on different forecasting models.



**Figure: Coupon Dataset**

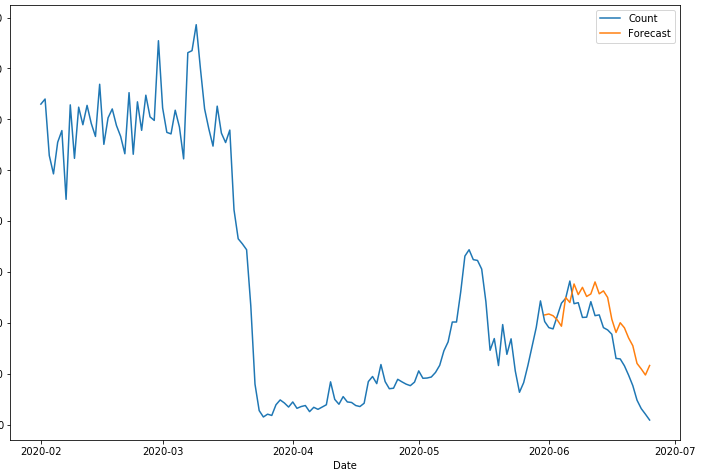
**ARIMA:** The Autoregressive Integrated Moving Average (ARIMA) method models the next step in the sequence as a linear function of the differenced observations and residual errors at prior time steps. It combines both Autoregression (AR) and Moving Average (MA) models as well as a differencing pre-processing step of the sequence to make the sequence stationary, called integration (I). The notation for the model involves specifying the order for the AR(p), I(d), and MA(q) models as parameters to an ARIMA function, e.g. ARIMA(p, d, q). An ARIMA model can also be used to develop AR, MA, and ARMA models.



**Figure: ARIMA Prediction Graph**

The main problem of this model is it goes for a negative prediction value when the data set is too much downwarded. The method is suitable for univariate time series with trend and without seasonal components.

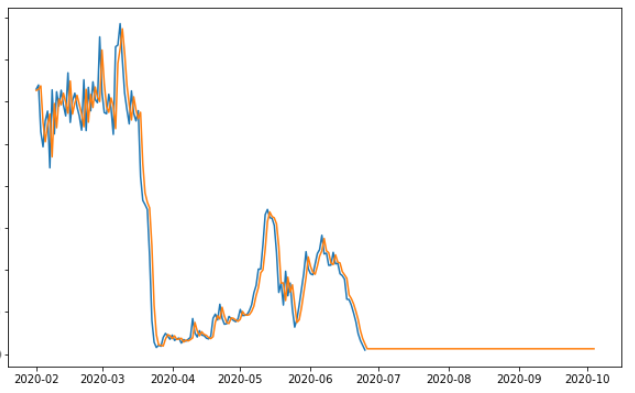
**SARIMAX:** The Seasonal Autoregressive Integrated Moving-Average with Exogenous Regressors (SARIMAX) is an extension of the SARIMA model that also includes the modeling of exogenous variables. Exogenous variables are also called covariates and can be thought of as parallel input sequences that have observations at the same time steps as the original series. The primary series may be referred to as endogenous data to contrast it from the exogenous sequence(s). The observations for exogenous variables are included in the model directly at each time step and are not modeled in the same way as the primary endogenous sequence (e.g. as an AR, MA, etc. process). The SARIMAX method can also be used to model the subsumed models with exogenous variables, such as ARX, MAX, ARMAX, and ARIMAX.



**Figure: SARIMAX Prediction Graph**

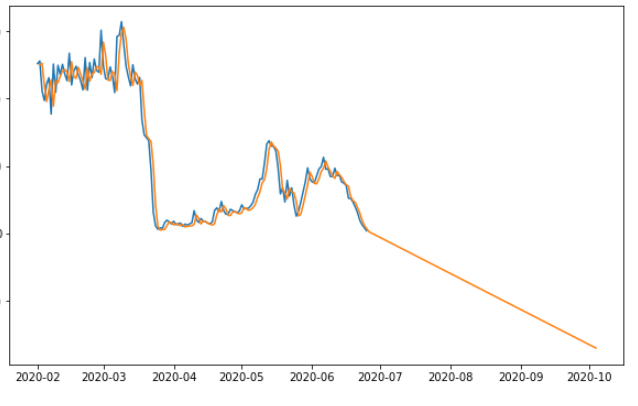
This model works impressively when you understand your data very well and if your data set has some kind of seasonality. If your data set does not have any seasonality, then there’s no need for trying this. The method is suitable for univariate time series with trend and/or seasonal components and exogenous variables.

**Simple Exponential Smoothing:** A simple exponential smoothing is one of the simplest ways to forecast a time series. The basic idea of this model is to assume that the future will be more or less the same as the (recent) past.



**Figure: Simple Exponential Smoothing Prediction Graph**

**Holt Winter’s Exponential Smoothing:** The Holt Winter’s Exponential Smoothing (HWES) also called the Triple Exponential Smoothing method models the next time step as an exponentially weighted linear function of observations at prior time steps, taking trends and seasonality into account. The method is suitable for univariate time series with trend and/or seasonal components.



**Figure: Holt Winter’s Exponential Smoothing Prediction Graph**

**Comparison:** As our data set didn’t have any seasonality or trend that’s why the SARIMAX model didn’t work well. The ARIMA model predicted well but for this data set Simple Exponential and Holt Winter’s Exponential model predicted better. Both of them did exactly the same prediction as Holt Winter’s Exponential is just a weighted Simple Exponential Method.

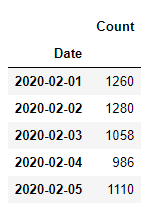
**Best Model:** Holt Winter’s Exponential model worked best in this dataset scenario.

* The basic idea of this model is to assume that the future will be more or less the same as the (recent) past.
* The only pattern that this model will be able to learn from demand history is its level.
* The level is the average value around which the demand varies over time.
* The exponential smoothing model will then forecast the future demand as its last estimation of the level.
* It is essential to understand that there is no definitive mathematical definition of the level. Instead, it is up to our model to **estimate** it.

*The exponential smoothing model will have some advantages compared to a simpler forecast model (such as a naive or a moving average):*

* The weight that is put on each observation decreases exponentially (we’ll discuss this in more detail later) over time (the most recent observation has the highest weight). This is often better than moving average models where the same weight is given to all the relevant historical months.
* Outliers and noise have less impact than with the naive method.

**Code Explanation:** After doing some preprocessing of the dataset we have transformed the dataset into this:



**Figure: Preprocessed Dataset**

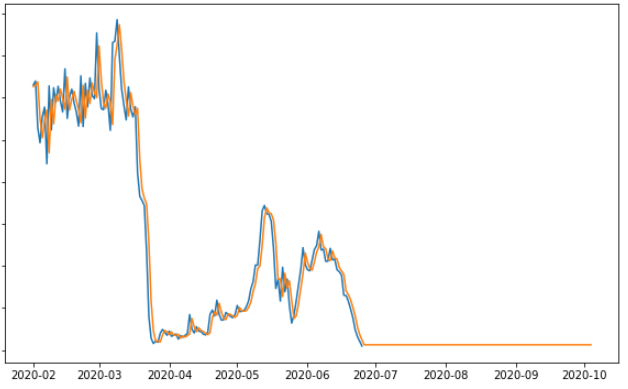
It is done in this way so that the model can learn easily. The Count column is the number of customers applied for coupons on that particular date.

|  |
| --- |
| from statsmodels.tsa.holtwinters import SimpleExpSmoothing model\_1 = SimpleExpSmoothing(df['Count']) |

This code section is showing how we imported the Simple Exponential Smoothing Method in python, model\_1 variable is an instance of the SimpleExpSmoothing Class that has a parameter of pandas Series. Here we have passed the ‘Count’ Column of the dataset.

|  |
| --- |
| model\_fit\_1 = model\_1.fit() # make prediction yhat = model\_fit\_1.predict(0,len(df)+100) pred = yhat.tolist() # data['pred'] = pred plt.plot(df['Count']) plt.plot(yhat) |

model\_fit\_1 is the model that learned from our given data. yhat variable is used for predicting the data based on the learning of our model. We have predicted 100 future days of coupon counts. The rest of the codes are showing the comparison between origin vs predicted as you can see below which shows that it predicted well.



**Figure: Origin vs Predicted values of Holt Winter’s Exponential Smoothing Model**