# Model

First, we'll go through the RNN and its LSTM structure. Following that, the LSTM will be completely discussed, including its bi-directional properties. Then we'll go into XGBoost and its capabilities in tackling this problem in greater depth. The Facebook Prophet algorithm will be explored, followed by an examination of the LightGBM.

## RNN

Recurrent Neural Networks are robust and effective classes of neural networks ranking among the top algorithms due to them being one with internal memory.

They are an old deep learning algorithm like many others developed in the 80s, but the true potential was discovered only recently. RNNs have been propelled to the forefront through the increased processing power and a large amount of data available today also with the advent of LSTM in the 90s.

Because of the internal memory, they retain vital information about the inputs received, making it very precise in predictions. Therefore, it is the trusted algorithm for time series data, speech, text, weather, video, and much more sequential data. Unlike, other algorithms RNNs have a more comprehensive grasp of the sequence and its context.

How do they work? The data is passed via a loop. After a choice has been made, it accounts for both the current and past inputs. This is shown in the figure below.

Diagram

Description automatically generated

Unlike a feed-forward neural network, RNNs add the past to the present. This is explained when we feed to the word “apple” as an input; the feed-forward would forget the previous information as it progresses, making it impossible to predict what letter comes next. RNNs remember information by making duplicates of the output and feeding it to the network.

The recurrence is a feedback link that has been created to allow past information to be sent back and received again. Figure 2 illustrates this point. RNNs can be layered and enlarged to better issue modeling while maintaining the previously improved time-series forecasts. Figure 3 depicts a more expanded form of RNN.

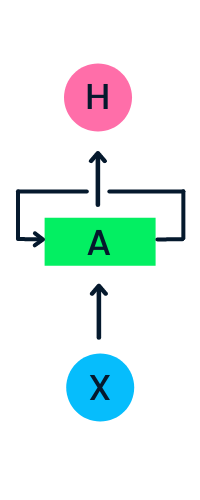


Figure 2: Simple RNN structure with a hidden layer

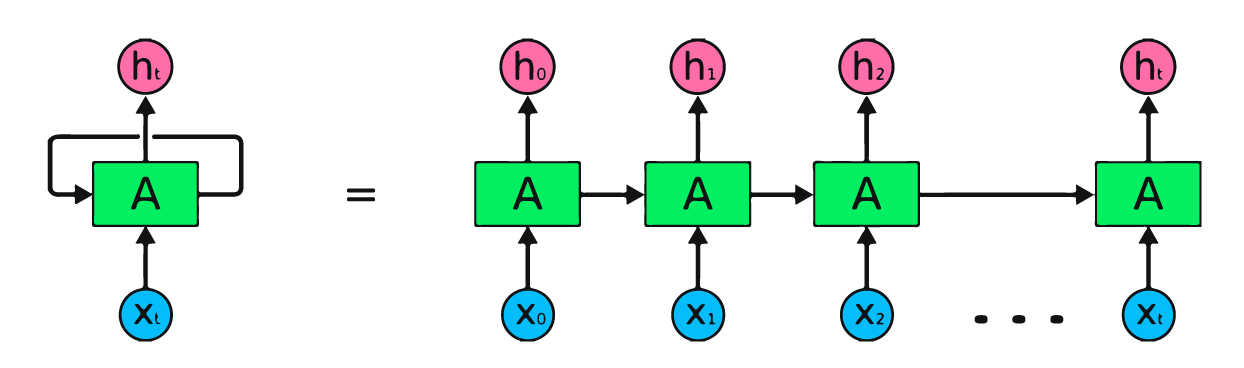


Figure 3: RNN with one hidden layer expanded over time.

### LSTM

Long short-term memory networks (LSTM) are a type of Deep Learning network. It is a class of RNN that can learn long-term connections, which is useful for solving sequence prediction issues. Apart from single data points like photos, LSTM has backpropagation, which means it can analyze the complete sequence of data. This is useful in natural language processing, machine learning, and other areas. The LSTM is a type of RNN that performs exceptionally well on a wide range of issues.

An LSTM model's central role is played by a memory cell called a 'cell state,' which maintains its state across time. The horizontal line which passes through the top of the diagram below represents the cell state. It can be compared to a conveyor system on which data just passes, unmodified.

In an LSTM, information can be provided to or withdrawn from the cell state, which is controlled by gates. These gates allow information to move in and out of the cell if desired. The method is aided by a pointwise multiplication operation as well as a sigmoid neural - network layer.

The sigmoid layer outputs integers between 0 and 1, with 0 indicating that "nothing should be let through" and 1 indicating that "everything should be let through". This is represented in Figure 4.

Diagram

Description automatically generated

Figure 4: This image visualizes a single LSTM neuron.

LSTM can address a variety of issues that prior learning algorithms, such as RNNs, couldn't. Long-term temporal connections can be efficiently handled by LSTM without the need for a lot of optimization work. This is used to deal with high-end issues.

### Bidirectional LSTM

Bidirectional-LSTM allows any neural network to store sequence information both in forwarding and backward directions.

The data runs in two channels in a bi-lstm, which distinguishes it from a conventional LSTM. We can have input move in one way, either forwards or backwards, with a normal LSTM. However, with bi-directional input, we can have the flow of information in both channels, preserving both futures as well as the past. Let's consider the sentence for a deeper understanding.

The vacant area in the line "guys visit..." cannot be filled. However, if we have an upcoming sentence like "the guys came out of the pub," we can readily anticipate that previous vacant space and then have the model do the same thing, and bi-LSTM enables the network to do so.

Diagram

Description automatically generated

The information flow from forward and backward levels can be seen in the figure. When jobs requiring sequence to sequence are required, BI-LSTM is typically used. Text categorization, natural language processing, and predictive models can all benefit from this type of network.

## XGBoost

This is a gradient boosting application that utilizes a decision-tree-based group ML model. Artificial NNs surpass most other algorithms or methodologies in prediction issues involving complex data (pictures, text, and so on). Decision tree-based algorithms, on the other hand, are now rated best-in-class for small-to-medium structured/tabular data.

Diagram

Description automatically generated

Evolution of XGBoost from Decision Trees

The XGBoost algorithm was created as part of a University of Washington research study. In 2016, Tianqi Chen and Carlos Guestrin submitted their article at the SIGKDD Conference, which ignited the ML industry. Since its inception, this algorithm has been acknowledged for not only topping a slew of Kaggle contests but also serving as the brains behind several cutting-edge industrial applications. As an outcome, the open-source projects have a robust ecosystem of data scientists collaborating with them, with 350 members and 3,600 contributions on GitHub. It sets itself apart through this means:

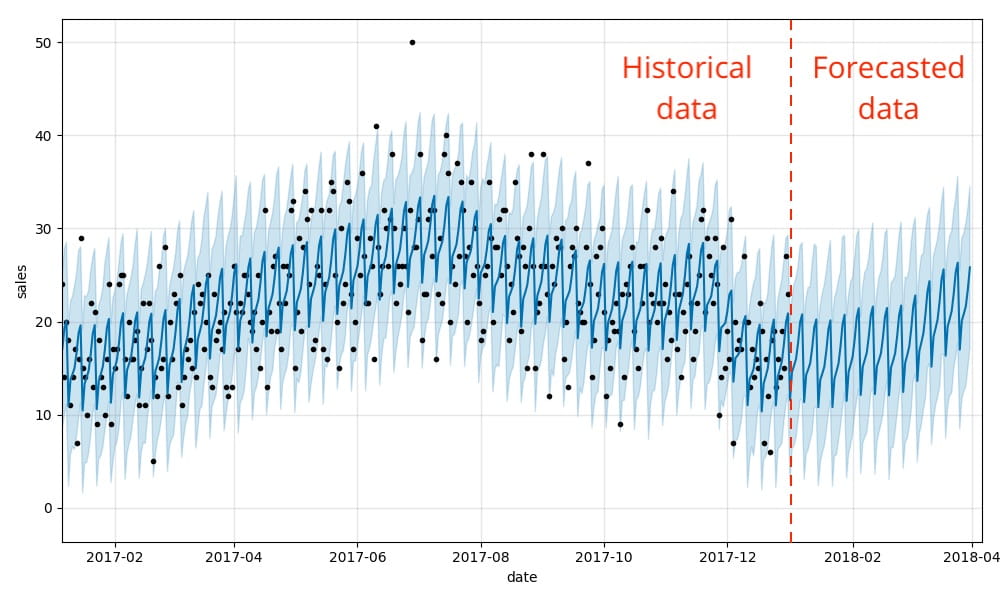
* Regression, classification, ranking, and user-defined prediction issues can all be solved using this tool.
* Very portable i.e., runs on popular Operating Systems (Windows, Linux, etc).
* Supports all major programming languages
* Supports cloud integration i.e., Azure, AWS, and other ecosystems.

With these facts, it does not mean the XGBoost algorithm should be used for all ML problems. A good rule of thumb is to explore both new and old frameworks to tackle the desired problems. It will only be a matter of time before a new model framework emerges that outperforms XGBoost in respect of predictive accuracy, adaptability, explainability, and practicality. XGBoost, on the other hand, will continue to rule the Machine Learning field till a serious opponent emerges.

## Prophet

Prophet is a time-series predicting procedure based on an additive model that fits non-linear trends with annual, weekly, and daily seasonality and holiday impacts. It performs effectively with time series with substantial seasonal influences and past data from multiple seasons. Prophet is forgiving of missing information and trend changes, and it usually manages anomalies efficiently.

Facebook's Core Data Science team published Prophet as open-source software. It may be downloaded from [CRAN](https://cran.r-project.org/web/packages/prophet/index.html) and [PyPI](https://pypi.org/project/prophet/).



Example of the Prophet algorithm output

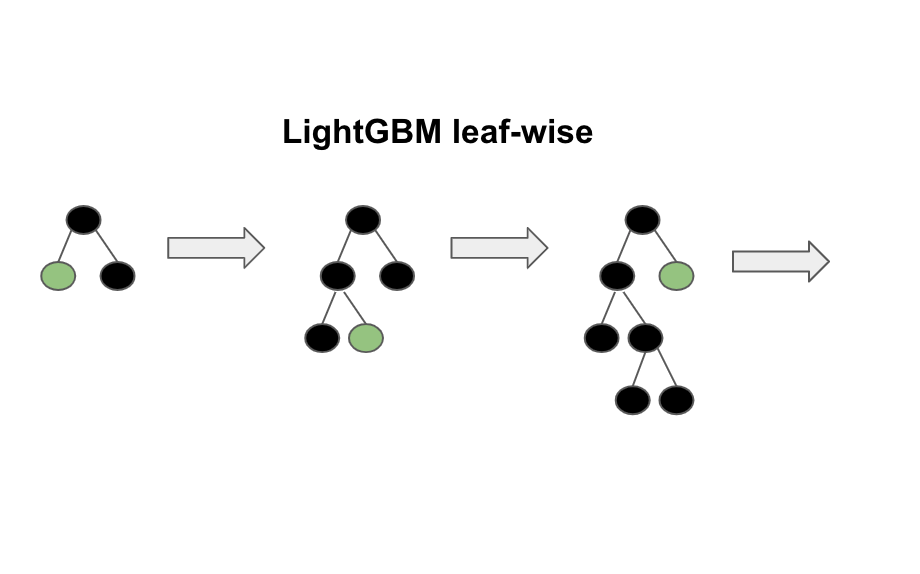
Prophet is utilized in a variety of Facebook apps to generate accurate predictions for strategy and goal setting. In most circumstances, we've found it to outperform any other strategy. We use Stan to fit models so that you may get forecasts in a matter of seconds. With no effort, you can get a fair prediction on sloppy data. Anomalies, missing information, and significant improvements in your time-series data are not a problem for Prophet. The Prophet method gives customers a lot of options for tweaking and adjusting predictions. By combining your subject expertise with human-interpretable variables, you can enhance your predictions. The Prophet process has been coded in R and Python, however, the fundamental Stan code for fitting is the same across both. To get predictions, use any language you're most confident with.

## LightGBM

This is a gradient boosting concept that includes tree-based learning methods, which are regarded to be quite a strong processing algorithm. This is thought to be a quick-processing algorithm.

Whilst trees of other algorithms develop horizontally, the LightGBM algorithm develops vertically, which means it expands leaf-wise while many other algorithms develop level-wise. To expand, LightGBM selects the leaf with the greatest loss. When expanding the same leaf, this can reduce loss more than a level-wise method.

Existing algorithms are finding it increasingly difficult to provide quick results as the number of data grows exponentially. For its computational power and ability to deliver findings quickly, LightGBM is dubbed "Light." It uses less bandwidth to operate and can handle massive volumes of data. Since the goal of the method is to achieve high accuracy while also bracing GPU leaning, this makes it the most often utilized technique in Hackathons.



Diagrammatic representation of Leaf-Wise Tree Growth

Unlike many other boosting techniques that develop trees level-by-level, LightGBM breaks the tree leaf-by-leaf. It grows the leaf with the greatest delta loss. The leaf-wise approach has a smaller loss than that of the level-wise algorithm because the leaf is stationary.

LightGBM isn't suited for small samples. Because of its sensitivity, it can quickly overfit little data. It's suitable for data with more than 10,000 entries. There are no predetermined criteria for considering whether to use LightGBM. This can be utilized for enormous amounts of data, particularly when great accuracy is required.

# Loss Functions

MAE, MSE, RMSE, and R2 are some of the evaluation metrics employed. The MAE indicates how well the model performs on most data by describing the mean error of the model's prediction. Because RMSE is derived from MSE, which gives significant weights to outliers, it is a good measure for predicting outliers. R2 is a metric that describes how effectively a model can represent and fit data. Naturally, the MAE, MSE, and RMSE should be as near to zero as feasible, and R2 should be as close to one as possible.

## Mean Squared Error

The degree of inaccuracy in statistical models is measured by the mean squared error (MSE). The average squared difference between observed and expected values is calculated. The MSE equals zero when a model has no errors. Its value rises as the model inaccuracy rises. The mean squared deviation (MSD) is another name for the mean squared error.

Where:

= number of data points

= observed values

= predicted values

## Mean Absolute Error

Mean Absolute Error (MAE) is the most basic measure of forecast accuracy. It is just the average of the absolute mistakes, as the name implies. The absolute error is defined as the absolute value of the difference between the predicted and actual value. It is a metric that quantifies the precision of continuous variables. This shows us how large of an error we may expect on average from the forecast. It is a statistic that assesses the average magnitude of errors in a set of predictions without taking their direction into account. The Mean Absolute Error is the average of the absolute differences between prediction and actual observation over the test sample, assuming that all individual deviations are equally weighted. The average model prediction error is expressed in units of the variable of interest in both Mean Absolute Error and Root Mean Square Error. The Mean Absolute Error and the Root Mean Square Error both have a range of 0 to 1 and are unaffected by error direction.

Where:

= prediction

= true value

= total number of data points

## Root Mean Squared Error

The root mean square error (RMSE) is the residuals' standard deviation (prediction errors). Residuals are a metric of just how far measured values are from the regression line; RMSE is a measure of how to spread out these residuals. In other words, it indicates how tightly the data is clustered around the best fit line. In climatology, forecasting, and regression analysis, root mean square error is widely used to check experimental results. When normalized observations and forecasts are utilized as RMSE inputs, the correlation coefficient has a direct link. If the correlation coefficient is 1, for example, the RMSE is 0 since all the points are on the regression line (and therefore there are no errors).

Where:

= variable i

= number of non-missing data points

= actual observations

= predicted observations

## R-squared

The coefficient of determination, or , is a statistic that indicates how well a model fits the data. It is a statistical tool about how well the regression line resembles the actual findings in the context of regression. When a statistical approach is used to forecast future outcomes or test hypotheses, it is critical to remember this. The overall sum of squares is the total of the distance the data is away from the mean all squared, and the sum squared regression is the sum of the residuals squared. It will take the value of 0 or 1 because it is a percentage.