**Time Series Forecasts**

**ICMS tax collection**

Guttenberg Ferreira Passos

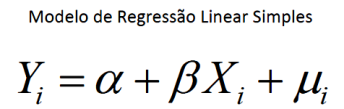
This article aims to make the ICMS tax collection forecast considering linear and non-linear models with time series.

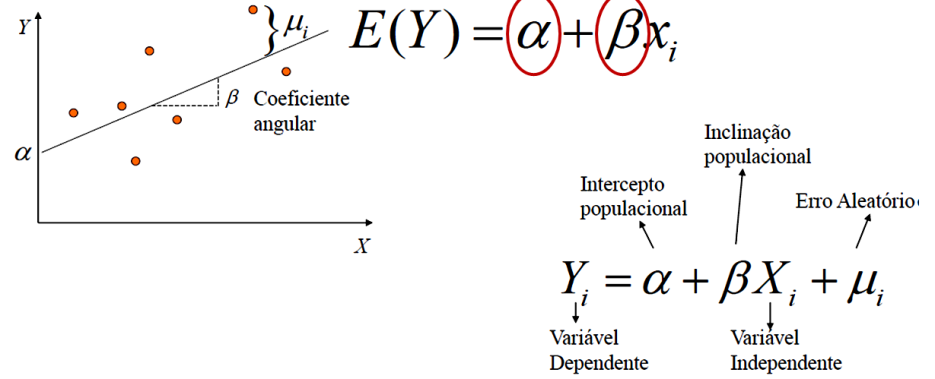
* Statistics with Trend Analysis were used for linear models;
* Statistical Methods, Artificial Intelligence, Machine Learning and Deep Learning were used for non-linear models, through the following algorithms:
* Naive;
* Exponential Smoothing;
* ARMA - Autoregressive Moving Average;
* ARIMA - Autoregressive Integrated Moving Average;
* SARIMAX – Seasonal Autoregressive Integrated Moving Average;
* DeepAR – GluonTS - Probabilistic Time Series Modeling;
* MLP - Multilayer Perceptron;
* LSTM - Long short-term memory.

Trend Analysis is an aspect of business analysis that attempts to predict the future movement of a product or service based on historical and statistical data. With this, it is possible to define action strategies, action plans and decision making.

One of these trends is Regression Analysis, which studies the relationship between a dependent variable and other independent variables. The relationship between them is represented by a mathematical model. This model is called the Simple Linear Regression Model (MRLS), which defines a linear relationship between the dependent variable and an independent variable.

MRLS Simple Linear Regression Model:





Biological Neuron

The Deep Learning Book Brazil is a [Data Science Academy](https://www.datascienceacademy.com.br/) initiative with the objective of helping to spread Deep Learning, one of the most revolutionary technologies of our time used in the construction of Artificial Intelligence applications.

According to the Deep Learning Book, the biological neuron is a cell, which can be divided into three sections: the cell body, the dendrites and the axon, each with specific but complementary functions.



Source: Data Science Academy - DSA

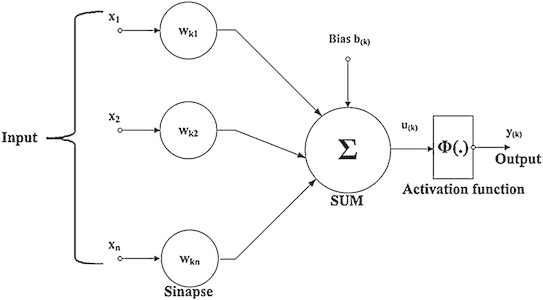
Artificial Neuron

An artificial neuron represents the basis of an Artificial Neural Network (ANN), a model of neuroinformatics oriented to biological neural networks.



Source: Data Science Academy - DSA

The knowledge of an ANN is encoded in the structure of the network, where the connections (synapses) between the units (neurons) that compose it are highlighted.



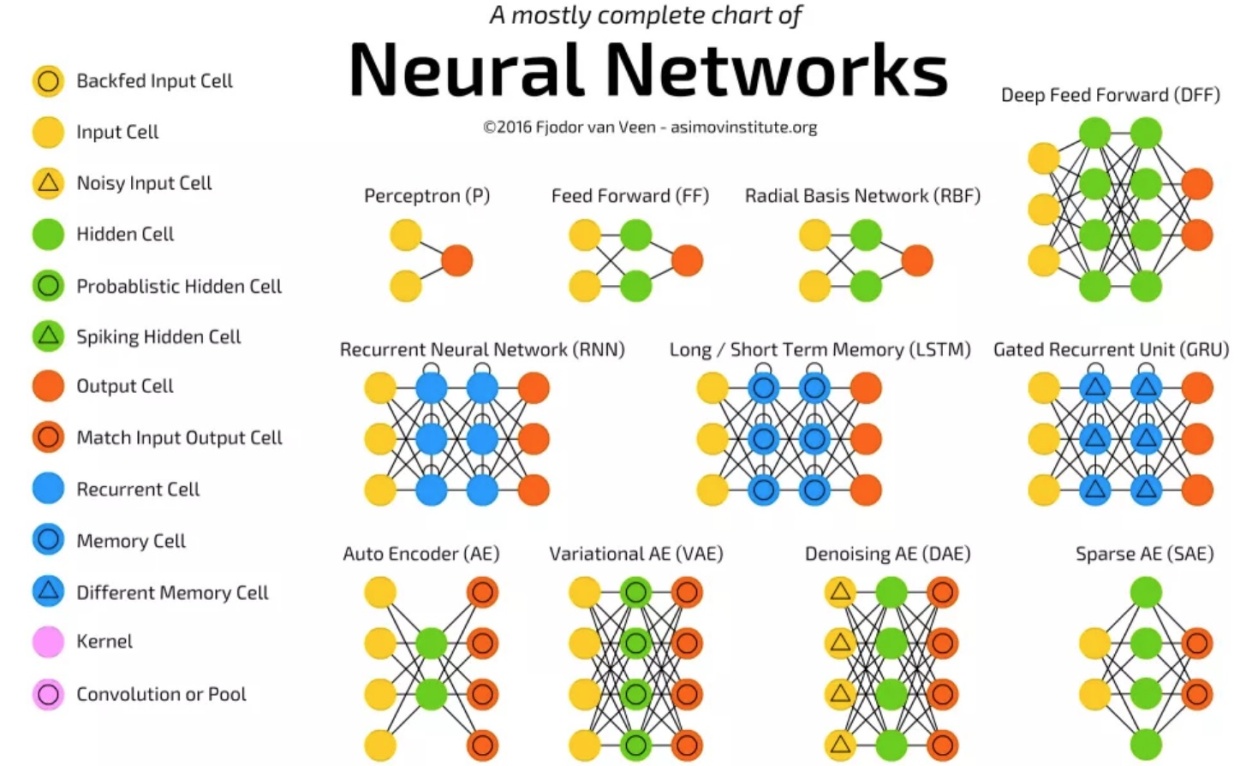
Source: Data Science Academy - DSA

In machine learning, Perceptron is a supervised learning algorithm of binary classifiers. A binary classifier is a function that can decide whether or not an input, represented by a vector of numbers, belongs to some specific class. It is a type of linear classifier, that is, a classification algorithm that makes its predictions based on a linear predictor function by combining a set of weights with the vector of features.

Neural Networks

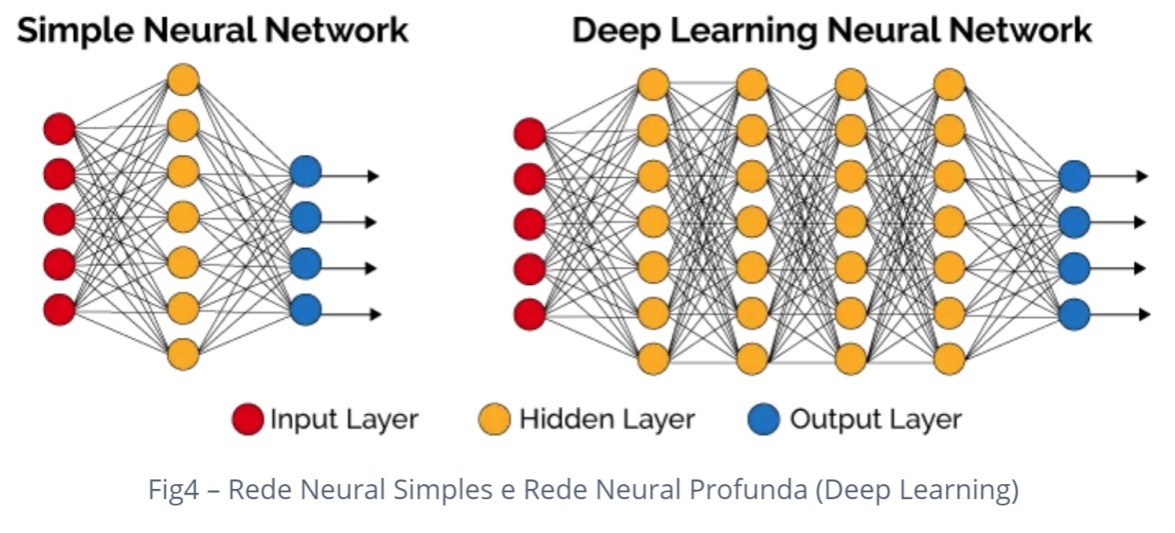
Neural networks are computing systems with interconnected nodes that function like neurons in the human brain. Using algorithms, they can recognize hidden patterns and correlations in raw data, group and classify them, and over time continually learn and improve.

Asimov Institute <https://www.asimovinstitute.org/neural-network-zoo/> has published a cheat sheet containing various neural network architectures, we will concentrate on the architectures below focusing on Perceptron(P), Feed Forward (FF), Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM):



Source: THE ASIMOV INSTITUTE

Deep learning is one of the foundations of Artificial Intelligence, a type of Machine Learning that trains computers to perform tasks like humans, which includes speech recognition, image identification and predictions, learning over time. We can say that it is a Neural Network with several hidden layers:



**Base Model**

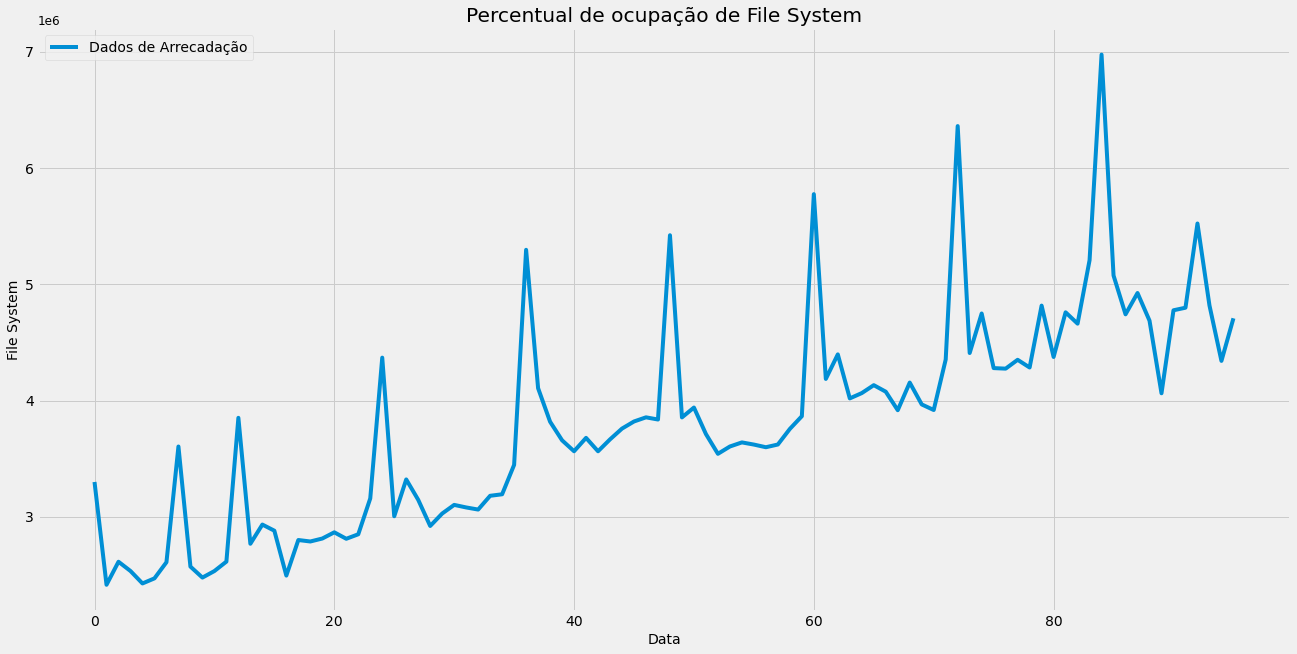
**Business Problem Definition**

ICMS tax collection forecast.

**Data set**

We use datasets that show ICMS collection. The data has records from the years 2010 to 2015.

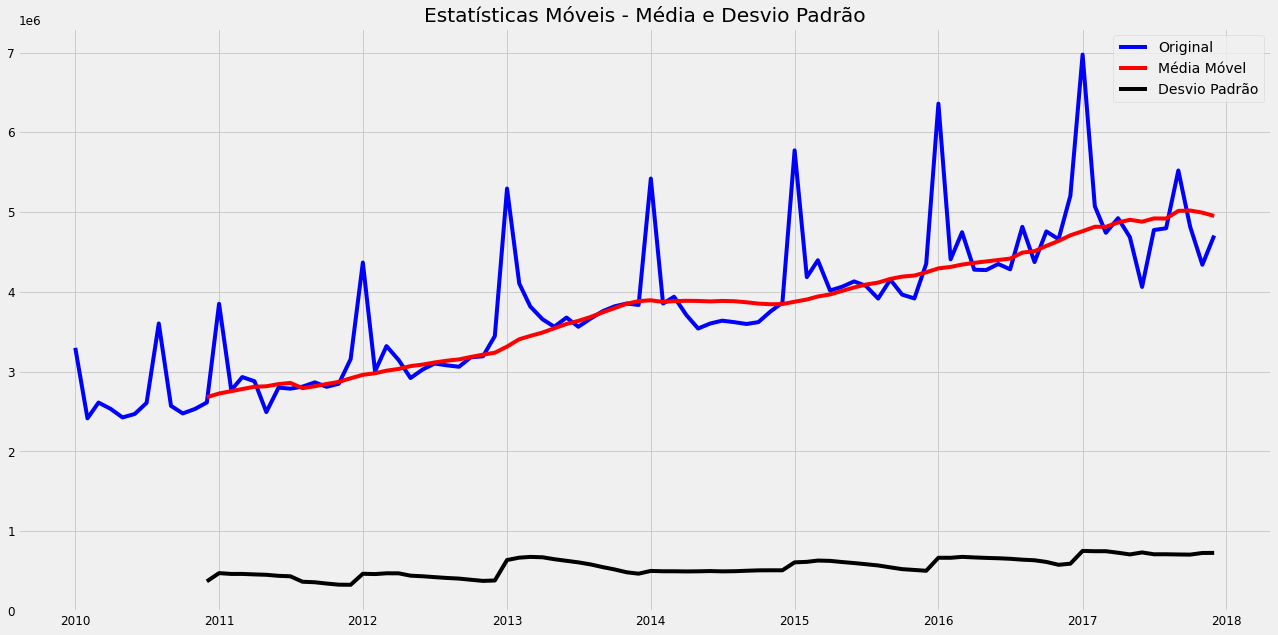
Database with 1 dataset with 2 columns, date and ICMS collection, with 96 records:



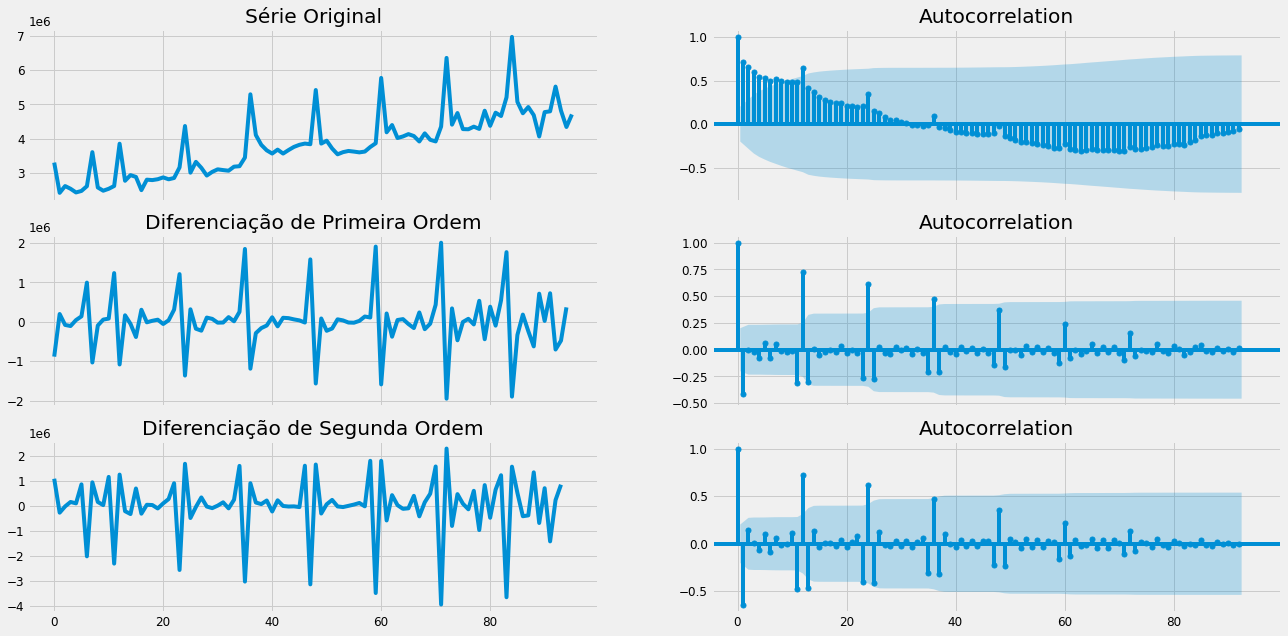
We noticed that there is a trend of increasing ICMS collection over time.

**Exploratory Data Analysis**

Let's test the stationarity of the series.



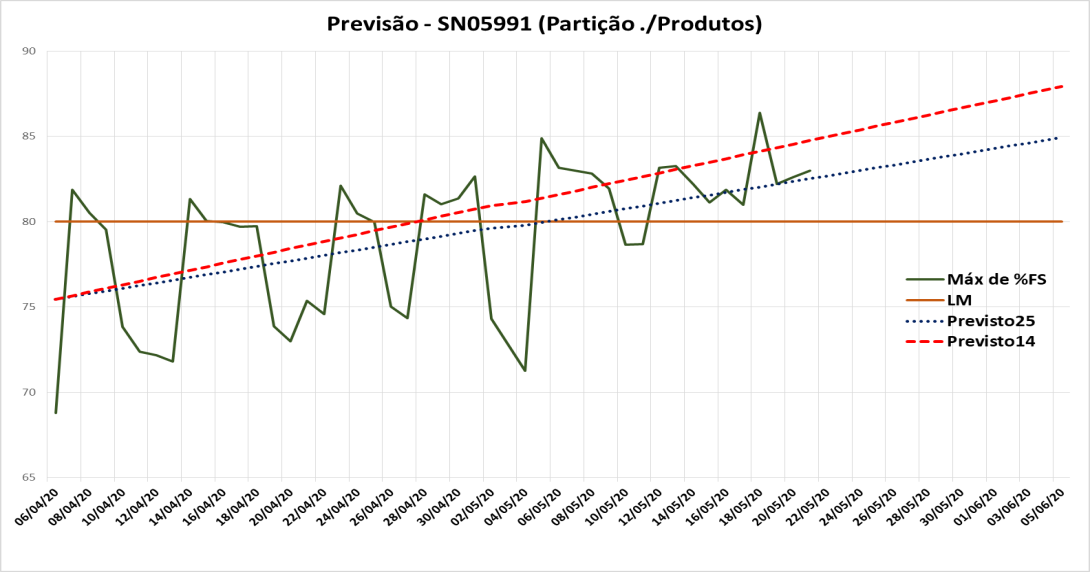
The ACF graph allows the evaluation of the minimum differentiation necessary to obtain a stationary series (Parameter d for the ARIMA Model):



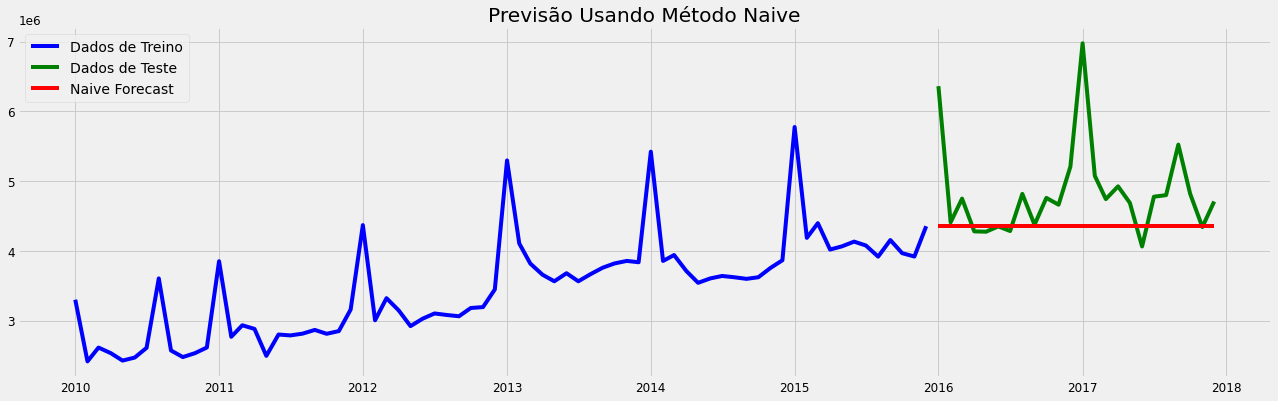
To run the models using Statistical Methods, the data were separated into:

* 72 training records and
* 24 validation records.

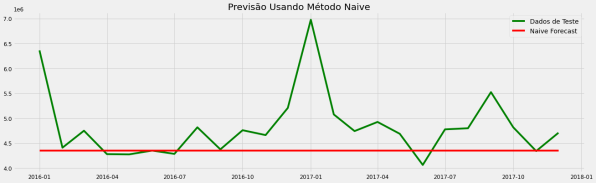
**Model 01 - Predictions Simple Linear Regression Method MRLS**

****

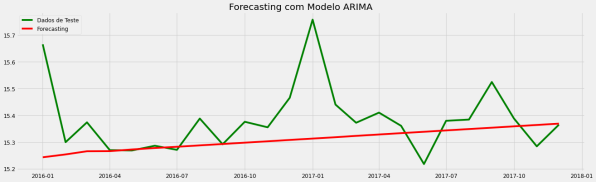
**Model 11 - Naive Method Forecasts**

****

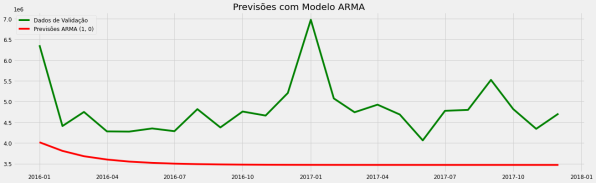
**11 – Naive Method**



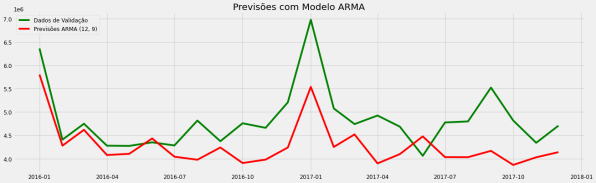
**13 - Forecasting – ARIMA LOG (2, 1, 0)**



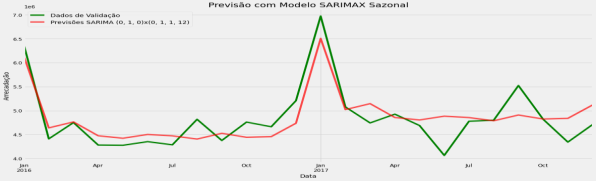
**15 – ARMA (1, 0)**

****

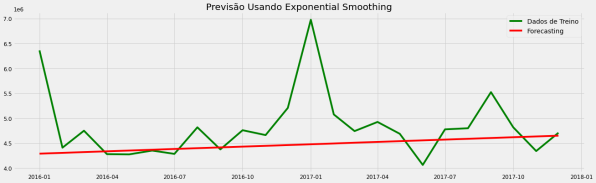
**15 – ARMA (12, 9)**

****

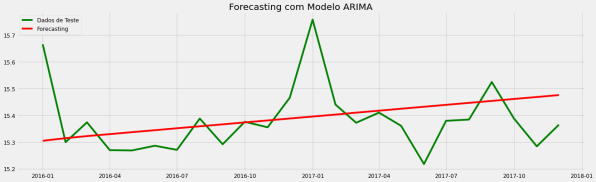
**17 – SARIMAX** **(0,1,0)(0,1,1,12)**

****

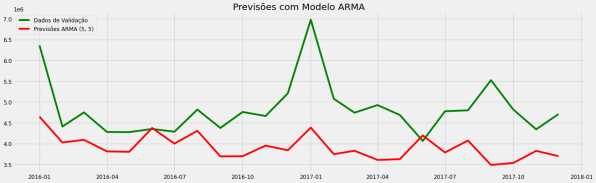
**12 – Exponential Smoothing**



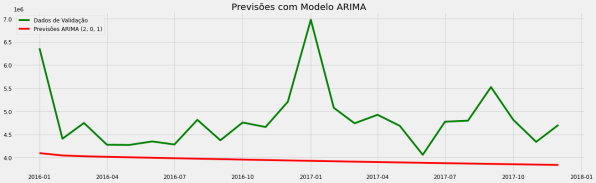
**14 - Forecasting – ARIMA LOG (1, 1, 1)**



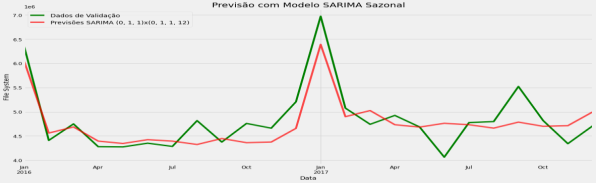
**15 – ARMA (5, 5)**



**16 - Predict - ARIMA (2, 0, 1)**



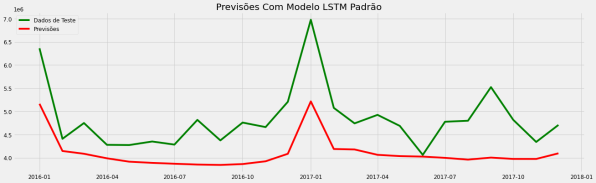
**18 – SARIMAX** **(0,1,1)(1,1,1,12)**

****

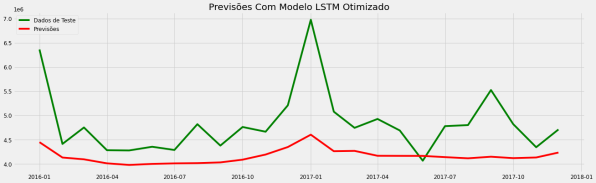
**Models using:**

**Inteligência Artificial - IA**

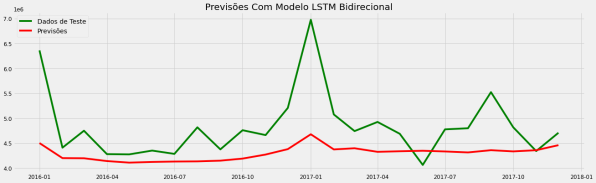
**22 – LSTM - IA (5 repetitions)**

****

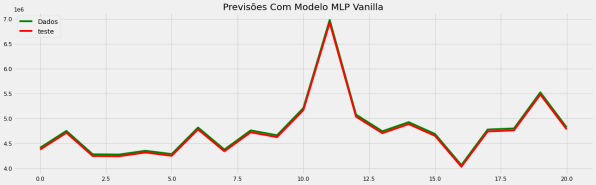
**23 – Optimized LSTM - IA**

****

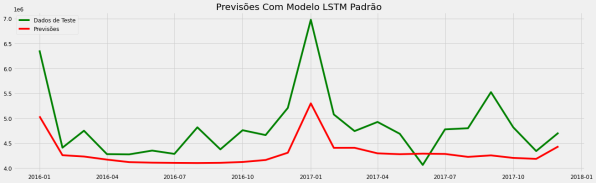
**25 – LSTM Bidirectional – IA**

****

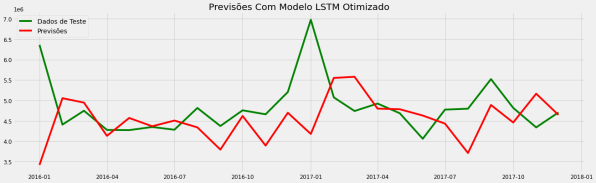
**RNN01 – MLP Vanilla – IA –**

****

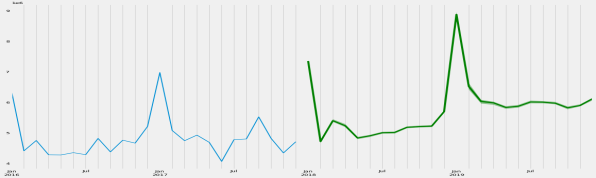
**22 – LSTM - IA (20 repetitions)**

****

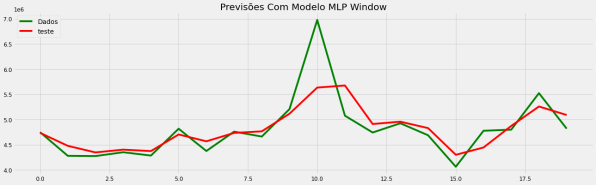
**24 – Stacked LSTM - IA**

****

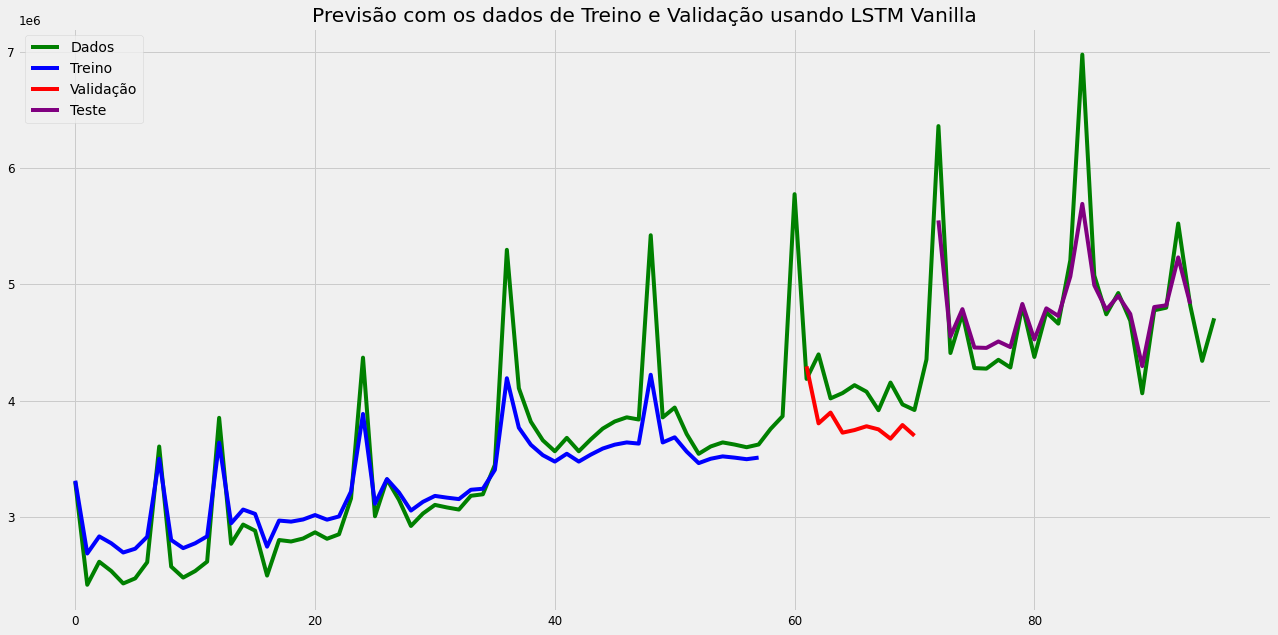
**25 – DeepAR – IA**

****

**RNN01 – MLP Window – IA –**

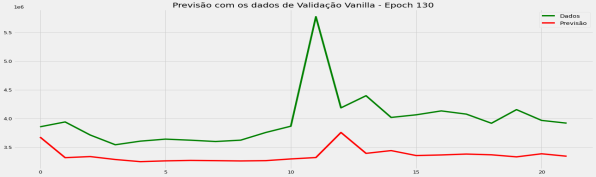
****

**Predictions using Artificial Intelligence - AI - Deep Learning**

****

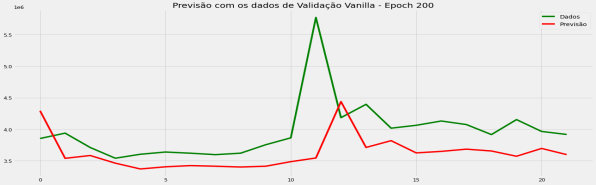
**RNN02 – LSTM Vanilla IA – Validation**

**Epoch = 130 - avoid the Overfitting**

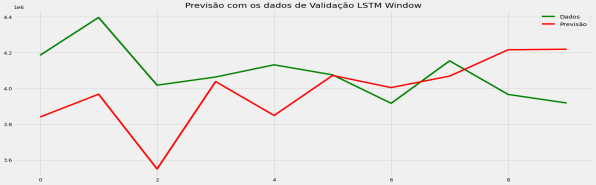
****

**RNN02 – LSTM Vanilla IA – Validation**

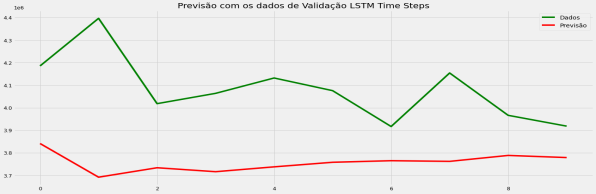
**Epoch = 200**

****

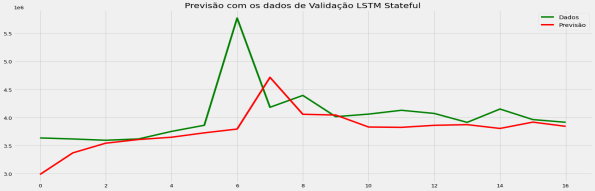
**RNN02 – LSTM Window IA– Validation**

****

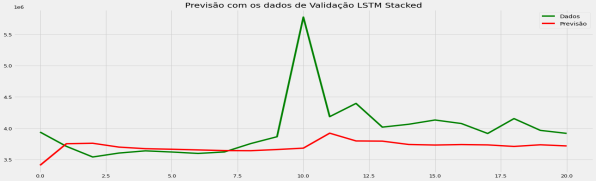
**RNN02–LSTM Time Steps IA Validation**

****

**RNN02 – LSTM Stateful IA – Validation**

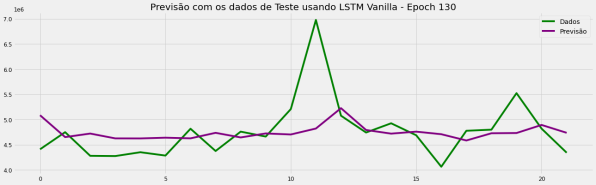
****

**RNN02 – LSTM Stacked IA – Validation**

****

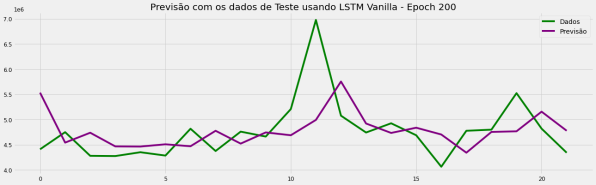
**RNN02 – LSTM Vanilla IA – Test**

**Epoch = 130 - avoid the Overfitting**

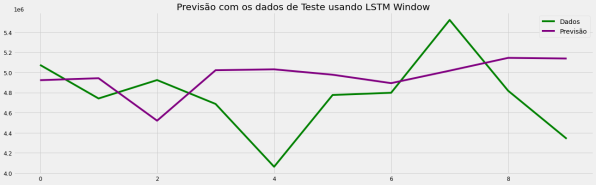
****

**RNN02 – LSTM Vanilla IA – Test**

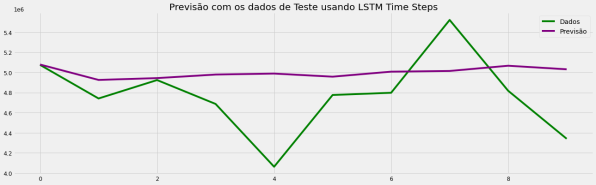
**Epoch = 200**

****

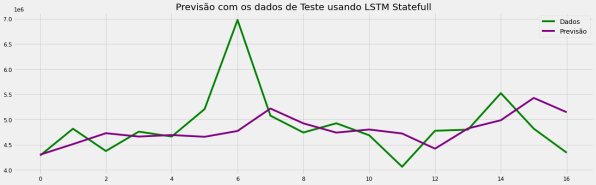
**RNN02 – LSTM Window IA – Test**

****

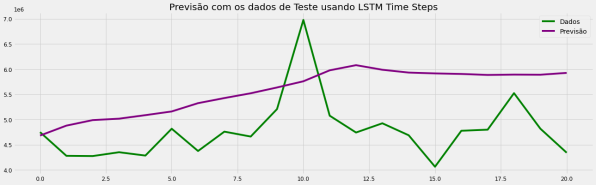
**RNN02 – LSTM Time Steps IA – Test**

****

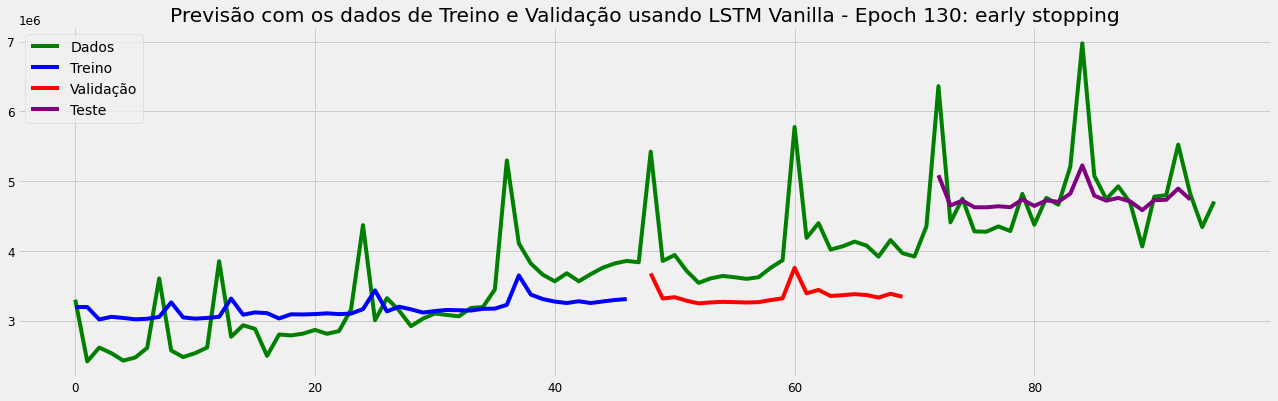
**RNN02 – LSTM Stateful IA – Test**

****

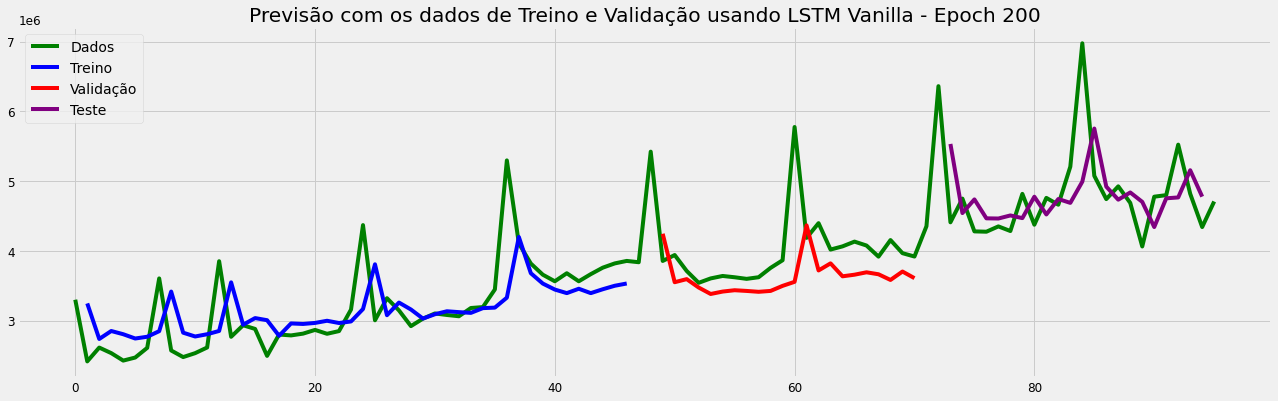
**RNN02 – LSTM Stacked IA – Test**

****

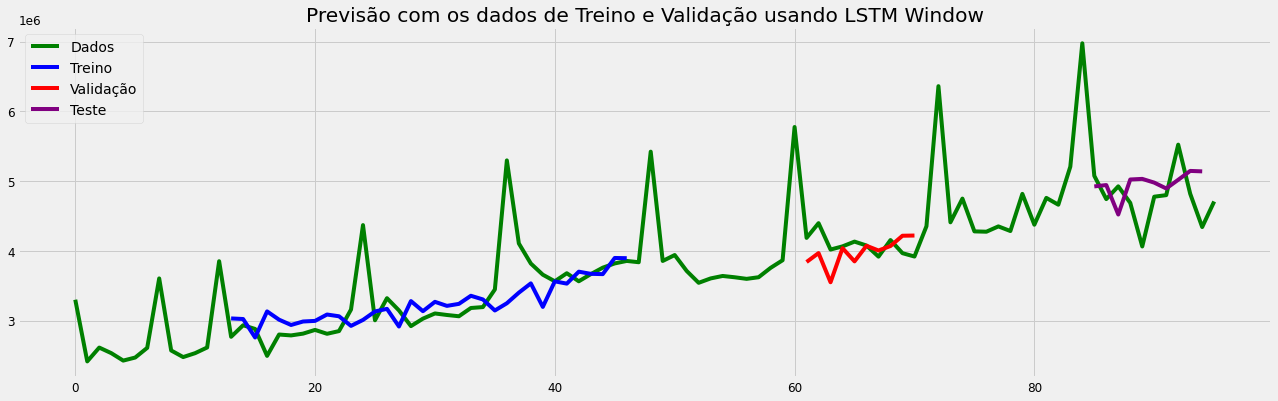
**Model RNN02 – LSTM Vanilla IA – Epoch = 130 – avoid the Overfitting**

****

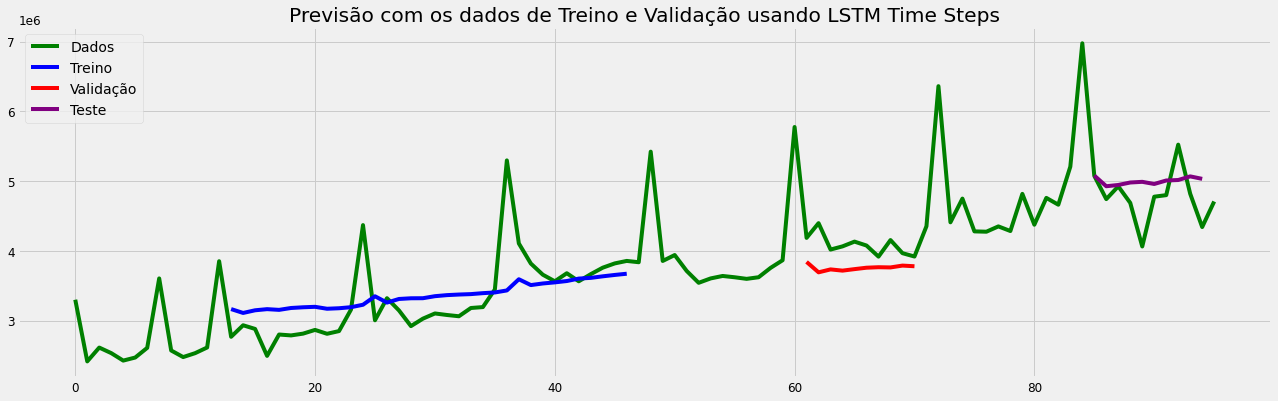
**Model RNN02 – LSTM Vanilla IA – Epoch = 200**

****

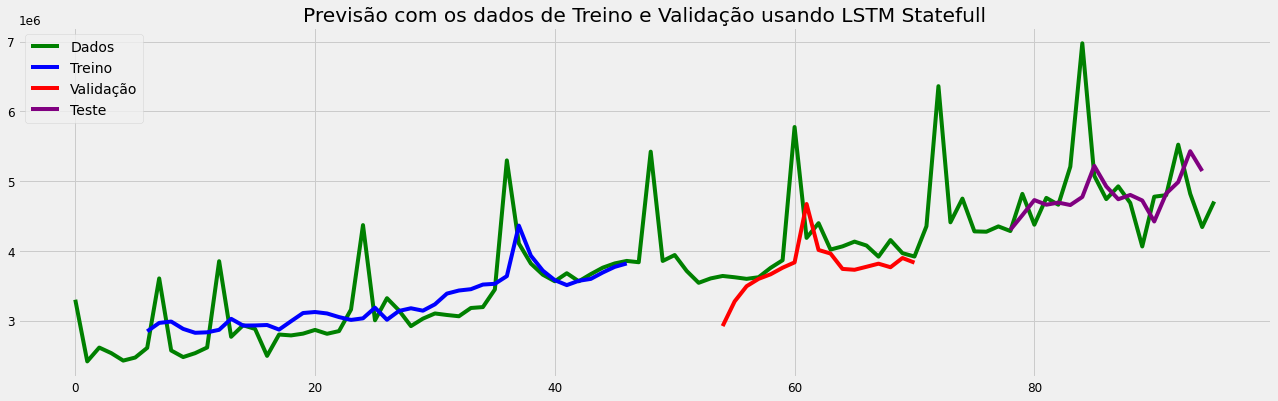
**Model RNN02 – LSTM Window IA**

****

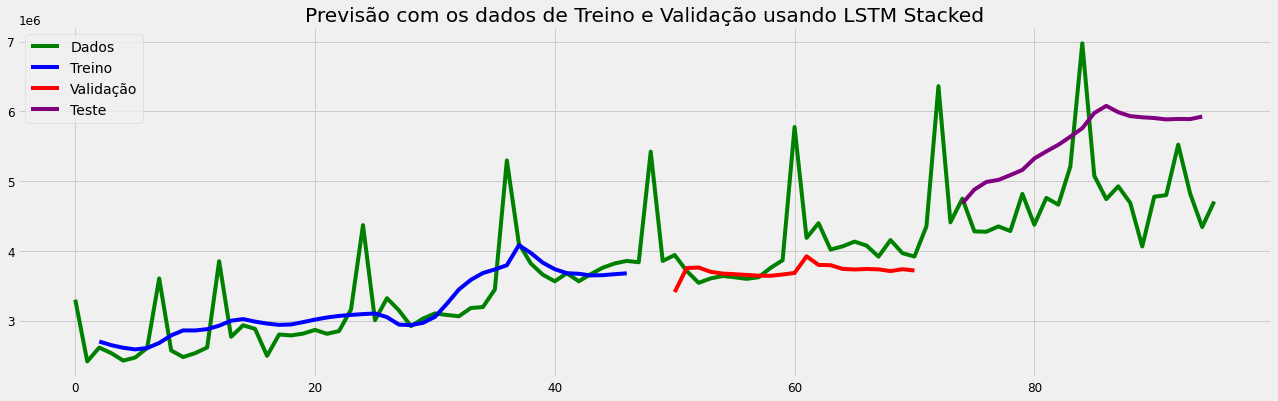
**Model RNN02– LSTM Time Steps**

****

**Model RNN02– LSTM Stateful**

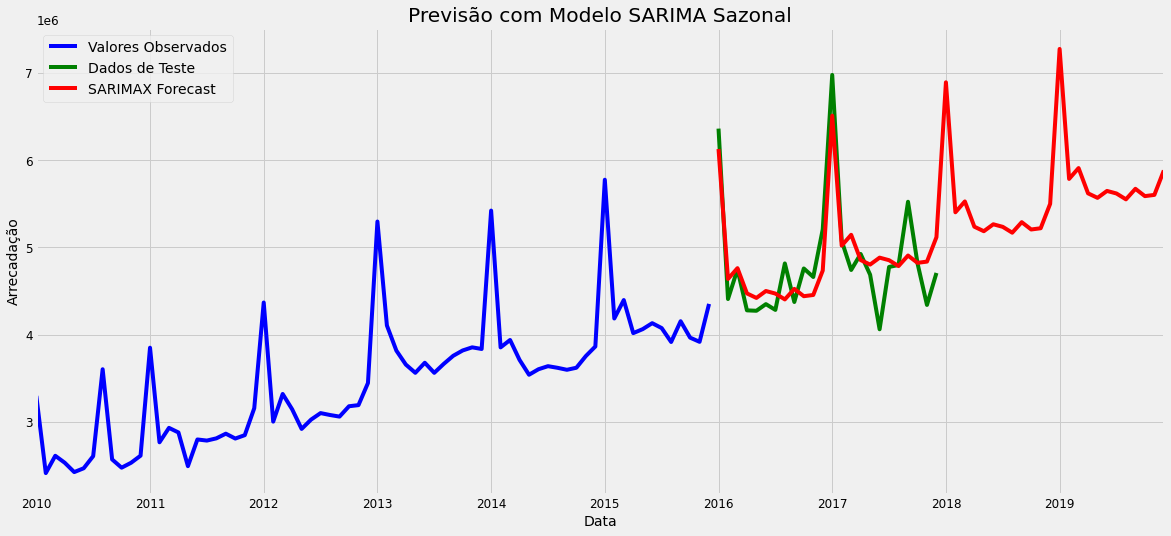
****

**Model RNN02– LSTM Stacked**

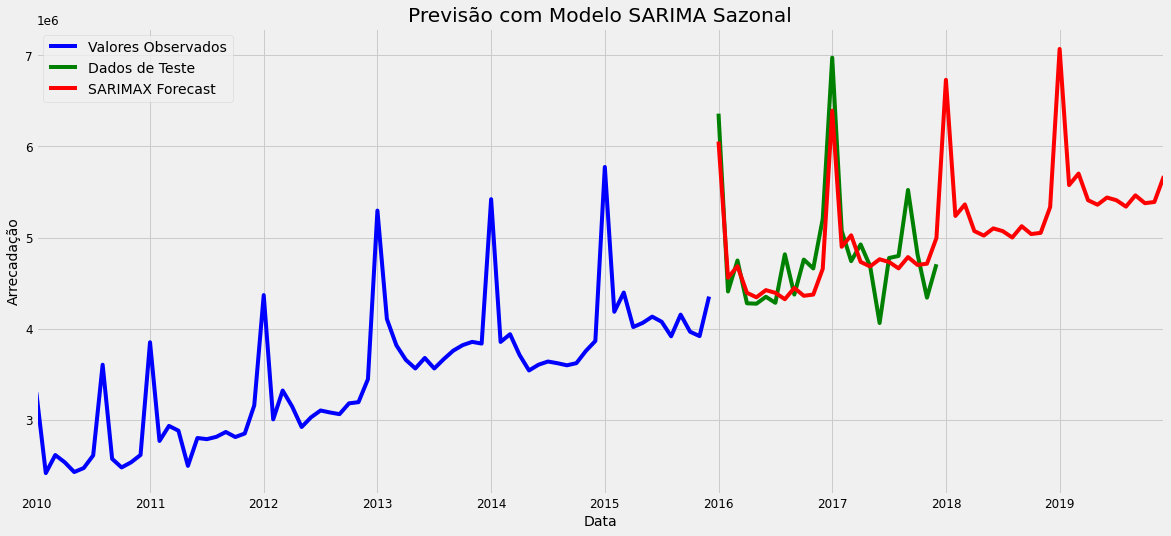
****

**Forecasting using Artificial Intelligence - AI**

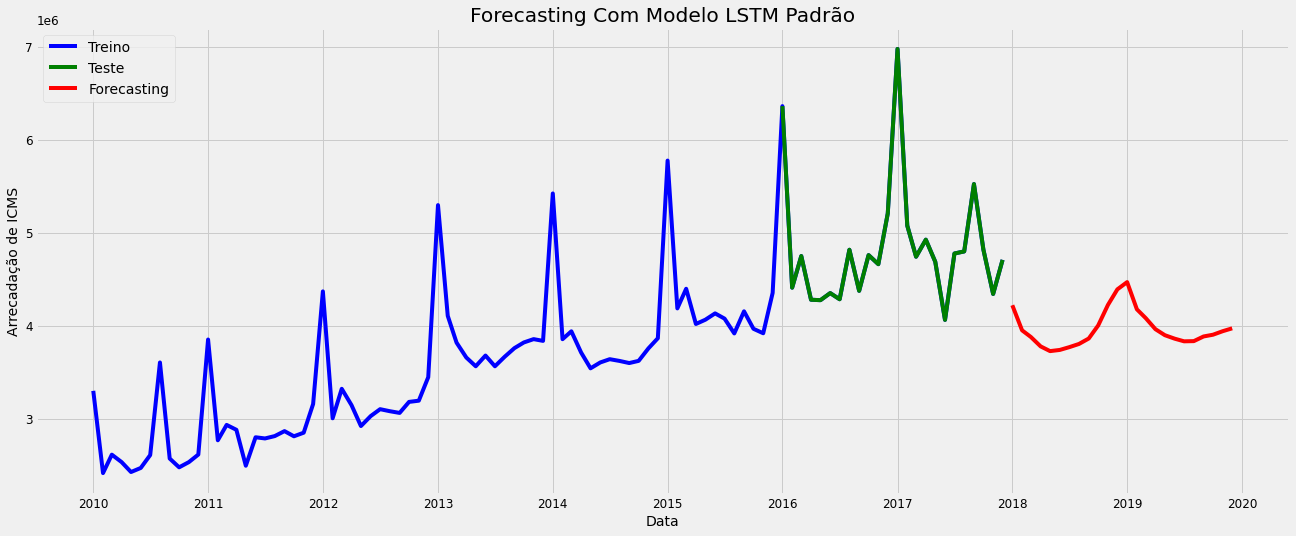
**Model 17 – SARIMAX** **(0,1,0)(0,1,1,12)**

****

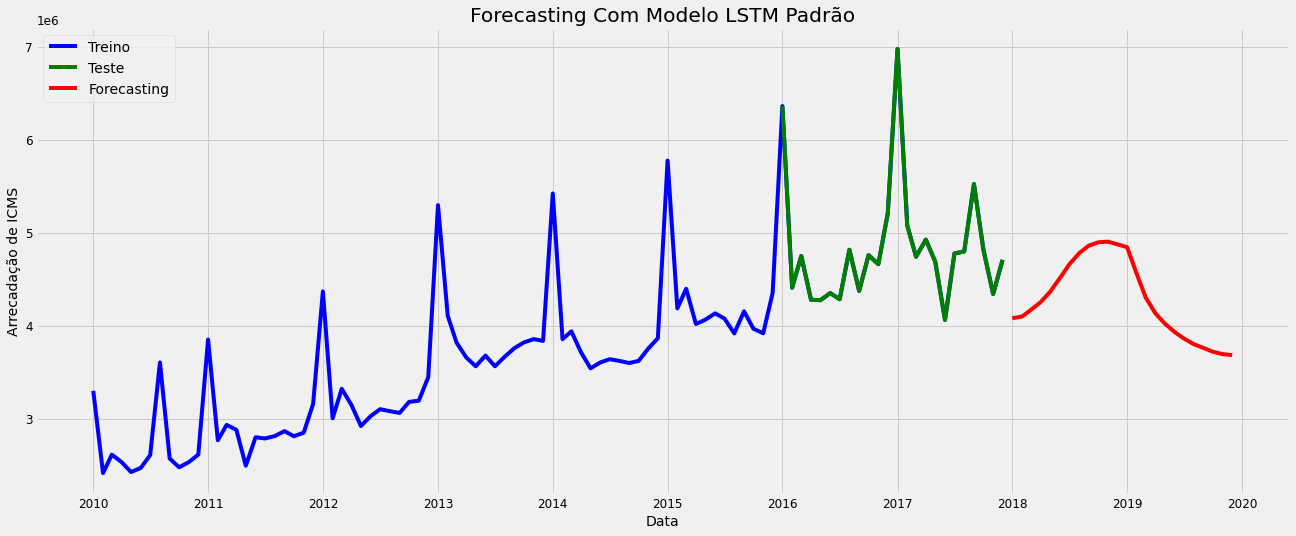
**Model 18 – SARIMAX** **(1,0,1)(1,1,1,12)**

****

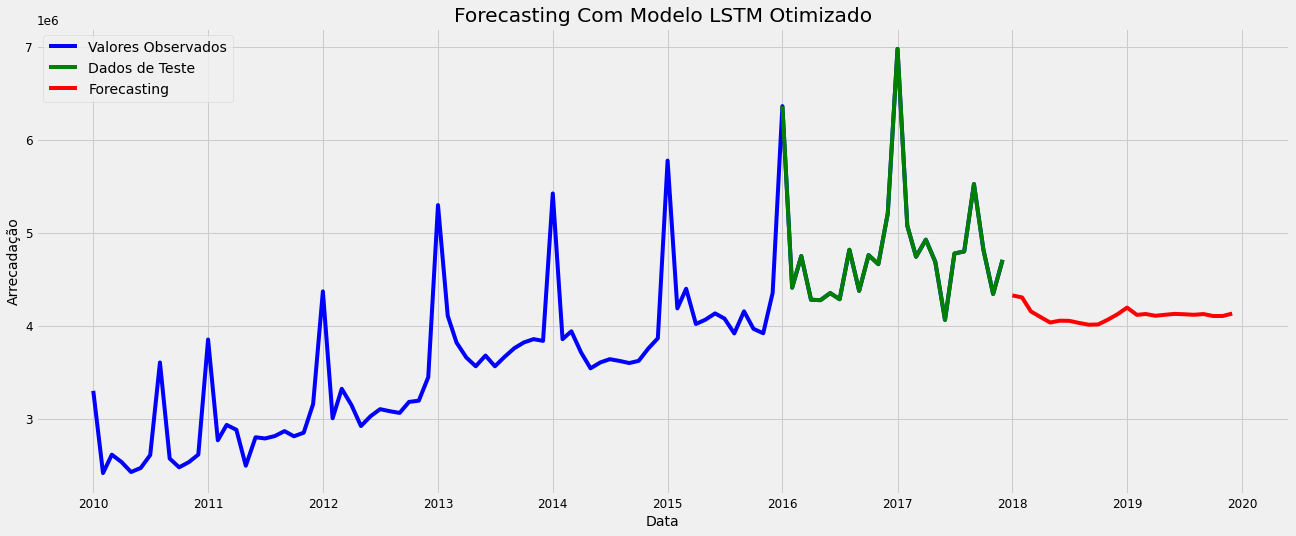
**Model 22 – LSTM - IA (5 repetitions)**

****

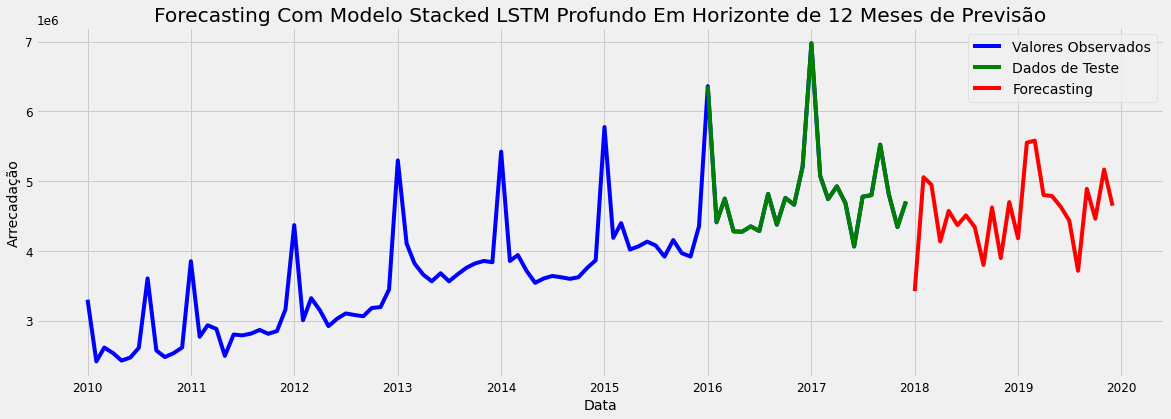
**Model 22 – LSTM - IA (20 repetitions)**

****

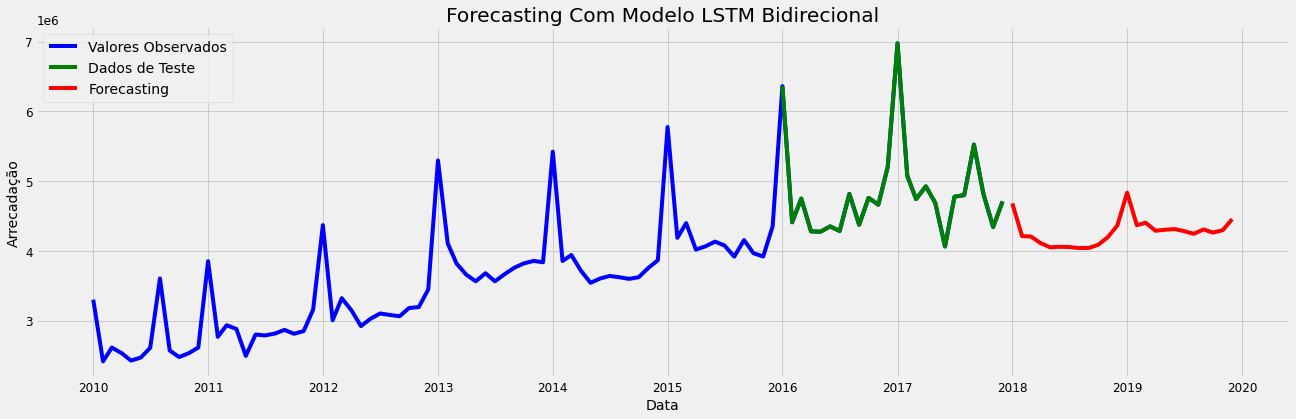
**Model 23 – Optimized LSTM - IA**

****

**Model 24 – Stacked LSTM - IA**

****

**Model 25 – LSTM Bidirectional – IA**

****

**Model Results:**



**Model Architectures using AI**

**Model 22 – LSTM - IA (5 repetitions)**

Número de repetições = 20

modelo\_lstm.add(LSTM(50, activation = 'relu', input\_shape = (n\_input, n\_features)))

modelo\_lstm.add(Dropout(0.10))

modelo\_lstm.add(Dense(100, activation = 'relu'))

modelo\_lstm.add(Dense(100, activation = 'relu'))

modelo\_lstm.add(Dense(1))

modelo\_lstm.compile(optimizer = 'adam', loss = 'mean\_squared\_error')

monitor = EarlyStopping(monitor='val\_loss', min\_delta=1e-3, patience=3, verbose=1, mode='auto')

modelo\_lstm.fit\_generator(generator, epochs = 200)

**Model 23 – Optimized LSTM - IA**

Número de repetições = 20

modelo\_lstm.add(LSTM(40, activation = 'tanh', return\_sequences = True, input\_shape = (n\_input, n\_features)))

modelo\_lstm.add(LSTM(40, activation = 'relu'))

modelo\_lstm.add(Dense(50, activation = 'relu'))

modelo\_lstm.add(Dense(50, activation = 'relu'))

modelo\_lstm.add(Dense(1))

adam = optimizers.Adam(lr = 0.001)

modelo\_lstm.compile(optimizer = adam, loss = 'mean\_squared\_error')

monitor = EarlyStopping(monitor='val\_loss', min\_delta=1e-3, patience=5, verbose=1, mode='auto')

modelo\_lstm.fit\_generator(generator, epochs = 200)

**Model 24 – Stacked LSTM - IA**

modelo\_lstm.add(LSTM(200, activation = 'relu', return\_sequences = True, input\_shape = (1, 1)))

modelo\_lstm.add(LSTM(100, activation = 'relu', return\_sequences = True))

modelo\_lstm.add(LSTM(50, activation = 'relu', return\_sequences = True))

modelo\_lstm.add(LSTM(25, activation = 'relu'))

modelo\_lstm.add(Dense(20, activation = 'relu'))

modelo\_lstm.add(Dense(10, activation = 'relu'))

modelo\_lstm.add(Dense(1))

modelo\_lstm.compile(optimizer = 'adam', loss = 'mean\_squared\_error')

modelo\_lstm.fit(X, y, epochs 5000, verbose = 1)

**Model 25 – LSTM Bidirectional - IA**

Número de repetições = 20

modelo\_lstm.add(Bidirectional(LSTM(41, activation = 'relu'), input\_shape = (41, 1)))

modelo\_lstm.add(Dense(1))

modelo\_lstm.compile(optimizer = 'adam', loss = 'mean\_squared\_error')

modelo\_lstm.fit\_generator(generator, epochs = 200)

lista\_hiperparametros():

n\_input = [24]

n\_nodes = [100]

n\_epochs = [200]

n\_batch = [5]

n\_diff = [12]

Número de repetições = 20

modelo\_lstm.add(Bidirectional(LSTM(100, activation = 'relu'), input\_shape = (41, 1)))

modelo\_lstm.add(Dense(1))

modelo\_lstm.compile(optimizer = 'adam', loss = 'mean\_squared\_error')

modelo\_lstm.fit\_generator(generator, epochs = 200)

**Model Architectures using AI Deep Learning**

**Model RNN01 – MLP Vanilla – IA**

model.add(Dense(8, input\_dim = look\_back, activation = 'relu'))

model.add(Dense(1))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam')

model.fit(trainX, trainY, epochs = 200, batch\_size = 2, verbose = 2)

**Model RNN01 – MLP Vanilla – IA**

model.add(LSTM(4, input\_shape = (1, look\_back)))

model.add(Dense(1))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam')

model.fit(trainX, trainY, epochs = 200, batch\_size = 1, verbose = 2)

**Model RNN02 – LSTM Vanilla – IA**

model.add(LSTM(4, input\_shape = (1, look\_back)))

model.add(Dense(1))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam')

model.fit(trainX, trainY, epochs = 200, batch\_size = 1, verbose = 2)

**Model RNN02 – LSTM Window – IA**

model.add(LSTM(4, input\_shape = (1, look\_back)))

model.add(Dense(1))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam')

model.fit(trainX, trainY, epochs = 200, batch\_size = 1, verbose = 2)

**Model RNN02 – LSTM Time Steps – IA**

model.add(LSTM(4, input\_shape = (None, 1)))

model.add(Dense(1))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam')

model.fit(trainX, trainY, epochs = 200, batch\_size = 1, verbose = 2)

**Model RNN02 – LSTM Stateful – IA**

model.add(LSTM(4, batch\_input\_shape = (batch\_size, look\_back, 1), stateful = True))

model.add(Dense(1))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam')

for i in range(200):

model.fit(trainX, trainY, epochs = 1, batch\_size = batch\_size, verbose = 2, shuffle = **False)**

model.reset\_states()

**Model RNN02 – LSTM Stacked – IA**

model.add(LSTM(4, batch\_input\_shape = (batch\_size, look\_back, 1), stateful = True, return\_sequences = True))

model.add(LSTM(4, batch\_input\_shape = (batch\_size, look\_back, 1), stateful = True))

model.add(Dense(1))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam')

for i in range(200):

model.fit(trainX, trainY, epochs = 1, batch\_size = batch\_size, verbose = 2, shuffle = False)

model.reset\_states()

The models were based on courses from the Data Science Academy DSA and the Community timeline on the portal: [www.datascienceacademy.com.br](http://www.datascienceacademy.com.br/)