Demand Forecast & Inventory Control of Pharmaceutical Drugs

Business Case Final Project

By

Alejando Córdoba, Roberto Corona, Emmanuel Hernández,

Esteban Marín, Victor Martínez & Gerardo Ruíz

Under the supervision of

Theresa Gebert

CENTRO MÉDICO DALINDE & CORRELATION ONE

May 2020

Contents

Abstract

The present business case attempts to reduce the cost of shortages of Dalinde's Hospital pharmacy on patent pharmaceutical drugs by analyzing time series data from January 2018 to December 2019 and predict the levels of consumption for the first three months of 2020 taking the nine most consumed drugs during that period. The methodology used was: 1) using an Autoregressive Integrated Moving Average [ARIMA] model and a Recurrent Neural Network [RNN] to forecast the demand; 2) proposing a Proportional-Integral-Derivative controller to manage the inventory, and; 3) a web dashboard to visualize historical data and with purchasing suggestions based on the forecasts and the PID control system. The impact of this case is an estimated saving of 4.48% of the total costs on the nine most consumed drugs.

1 Introduction

Located in the heart of Mexico City, in the Roma neighbourhood, Dalinde Medical center is a tertiary referral hospital with more than 70 years of experience supplying healthcare services. One of the main problems identified by the Head of Pharmacy, Julio Pérez, is the inventory stock of patented medicines that consistently run out in their storage. Although Dalinde buys two drug families: patented and generic drugs, Mr. Pérez, points out that the hospital does not have a problem with the latter. This issue is very recurrent, and directly affects the treatment of patients, for example, it impacts scheduled surgeries and postpones them. When a medication is not in the pharmacy and it is urgently required, a resource from the Pharmacy department is appointed to its work post to find the medicine from the retail provider directly. This generates unforeseen expenses, and an excessive price of medicines. The solution they looked for was to be able to predict scarcity, find what factors it correlates with and anticipate it to avoid monetary losses.

Figure 1: Dalinde Hospital

To solve the shortage problem that Mr. Pérez pointed out we got historical data of 27 months, from January 2018 to March 2020. The data included information about the consumption of the patent drugs (demand), the requests of the pharmacy, the amount bought, and the amount of medicines that were requested but not fulfilled. In order for us to avoid scarcity we decided to focus our efforts on predicting the demand. If the pharmacy personnel knows on advanced the amount of drugs that will be consumed in the future they, can plan their budget and the amount of specific drugs required for the next month(s), thus shortages issues will be anticipated and minimized. Because we got time series data, **we built 2 prediction models (ARIMA, and a Recurrent Neural Network) to forecast the demand of the top 9 patent drugs consumed that add up to $1.7 Millon Pesos ($91 thousand USD), and 1 Inventory Controller (PID). We also created a website where the pharmacy personnel can visualize historical information of the drugs, and make requests based on our predictions**.

2 Data Engineering

2.1 Infrastructure

We used an EC2 Ubuntu 18.04 image in AWS, and we created a Network Security Group to expose the ports needed for our teamwork in the cloud with Jupyter Notebook containers. We recreated the environment with portainer.io, and built different containers in a Docker system: **2 Development machines with Jupyter, 1 Database with PostgreSQL, and 1 Internet Site with Django.**

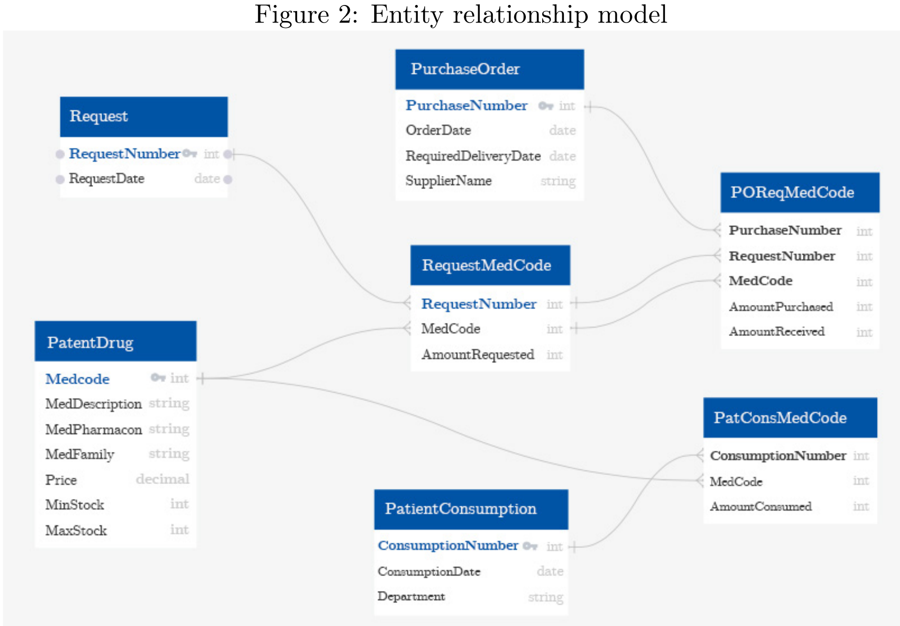
We decided to make our work in docker to ensure a quick infrastructure setup, most of the internal communication between containers is done by the internal docker VLAN, but thinking about further work with Dalinde hospital system, or ping from their own server, some endpoints were opened:

Table 1: Endpoints

Endpoint	**Usage**	**Endpoint**	**Usage**
9000	Portainer Admnistrator	2222	SSH | Django Container
8000	Django	8888	Jupyter Notebook 1
5432	Postgres	8889	Jupyter Notebook 2

Bulding the Website

Figure 2: Entity relationship model

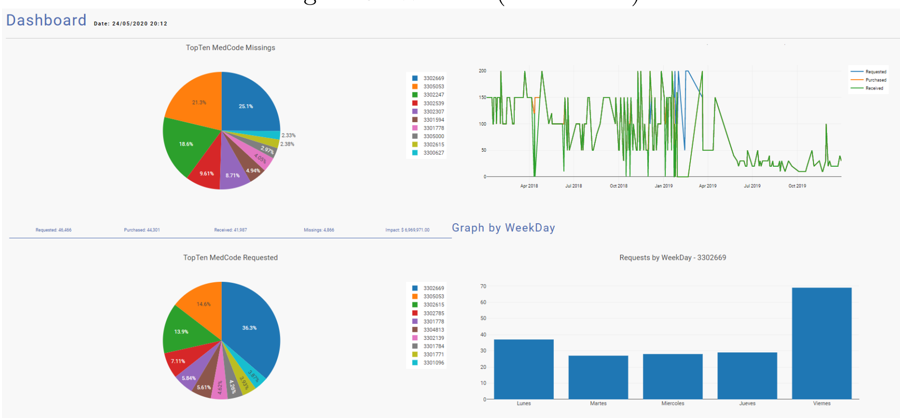


We needed a Framework that would allow us to quickly integrate the Database and be able to analyze the information and obtain reports in a simple way, for them we considered to use Django as the Web Aplication Framework. Figure 2 describes the database architecture.

2.2 Application

Dash allowed us to integrate with Django environment an Application Framework to create custom Dashboards with themes called layouts, in a simple and effective way. The graphs that were integrated in the Web Site are based with Plotly structure code. A preview of the Web Site is shown in figure 3, which can be accessed through this link.

Figure 3: Website (Dashboard)



3 Data Analysis and Computation

3.1 Data Cleaning

We got the data from the head of the pharmacy, Mr. Julio Pérez , and the hospital comptroller, Iván Ramos. This data comes from specific queries from Dalinde's system. This queries are reports in XLSX format with titles and not in proper database format. We got 3 different kinds of reports: 1) **Purchasing Orders Report (2 annual reports with daily frequency)**. 2) **Consumption Reports (27 monthly reports with monthly frequency)**. 3) **Incomplete Orders Reports (2 annual reports with daily frequency)**.

We had the ID Code for each patent drug to link all the reports. Besides of the information provided by the pharmacy, we obtained the market prices for each patent drug by coding a web scraper that looked for the price in different websites of drugs stores in Mexico City. We parsed the XLXS reports by getting rid of the rows and columns that were not informative, such as the header of the files and other subheaders from certain subsections. Certain reports had extra information about other types of drugs that were not patent drugs, so we decided to discard those observations. The output of this parsing and data cleaning procedure were 4 CSV files:

- 1 CSV file that contained the monthly consumption levels for patent drugs (January, 2018 - March, 2020).
- 1 CSV file that contained daily purchasing orders for patent drugs (January, 2018 - December 2019).
- 1 CSV file that contained daily incomplete purchasing orders for patent drugs (January, 2018 - December 2019).
- 1 CSV file that had the directory of names, prices, usages, and ID code for every patent drug.

The list of variables used for this business case is the following:

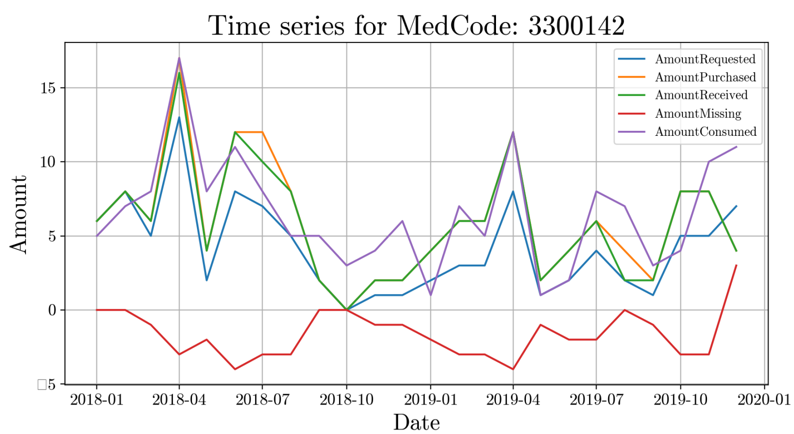
Table 2: List of Variables

Variable	**Description**	**Type**
RequestNumber	ID of the Requisition made by the pharmacy	String
RequestDate	Date of the Requisition made by the pharmacy	Date
PurchaseNumber	ID of the Purchased made by the Purchasing Department	Integer
PurchasingDate	Date the purchase was made	Date
MedCode	ID of the Patent Drug	Integer
AmountRequested	Amount of the Patent Drug Requested	Integer
AmountPurchased	Amount of the Patent Drug Purchased	Integer
ConsumptionDate	Date the Patent Drug was consumed	Date
AmountConsumed	Amount of the Patent Drug that was consumed	Integer
UnitaryCost	Unitary Cost of the Patent Drug	Float
TotalCost	Unitary Cost of the Patent Drug times the Amount Consumed	Float
ReceivedDate	Date the purchase was delivered	Date
AmountReceived	Amount of Patent Drug received after purchase	Integer
OrderPercentageFulfillment	Fraction of the amount purchased that was received	Float

3.2 Exploratory Data Analysis

From the purchasing data, we took the pharmacy requests that were not bought or not fulfilled and create an *Amount Missing* variable. Thus we took the difference of the amount requested by the pharmacy and the amount that was received. The positive difference indicates the shortage amount for a specific request order. In some cases this difference was negative, indicating that there was an excess supply. Since the variable of interest is the *Amount Consumed* of patent drugs, and this variable was reported with monthly frequency, we aggregated the other daily variables to a monthly frequency; this helped us to analyze all our time series in the same discrete frequency. The numeric time series of all our 1,681 patent drugs had the same monthly frequency. A visualization of the time series for one of these drugs is shown in figure 4.

Figure 4: Time series for MedCode: 3300142 aggregated by month



Our first approach was to focus on the top 10 drugs that where missing the most and represented a high monetary cost, but when we took this approach we realized that not all time series had continuous data, making our first approach not feasible. Having said that, we decided to narrow our analysis to the drugs that where uninterruptedly consumed during the 24 months (Jan, 2018 - Dec,2019). After this computation we got 87 time series, each for a patent drug. When computing the total cost of these 87 drugs, it added up $4.45 Million Pesos ($226 thousand USD).

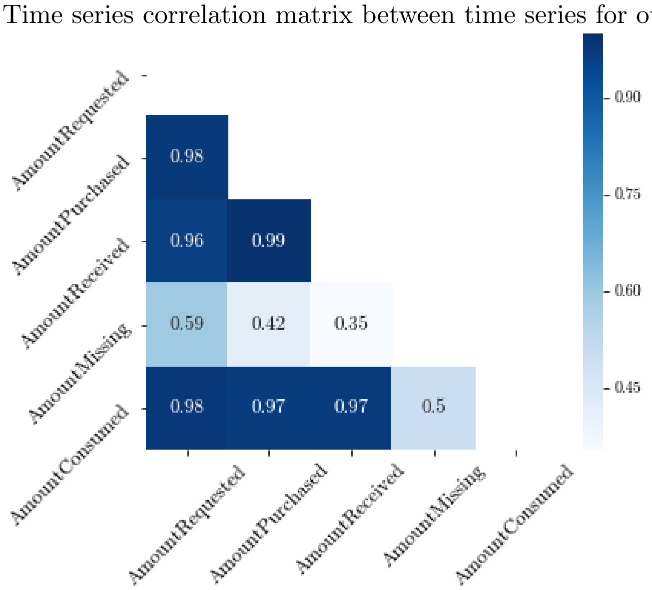
Afterwards, we took the drugs that presented at least one shortage throughout the 24 months of analysis and picked the top 9 with the highest demand and costs. These nine medicine codes (MedCodes) represent 40% of the total cost of drugs that were uninterruptedly consumed during 2018-2019, $1.79 Million Pesos ($91 thousand USD). These top 9 patent drugs, along with their descriptive statistics and costs, are listed in table 3.

Table 3: Descriptive statistics and total costs of the top 9 drugs

MedCode	**AmountConsumed**							**TotalCost [MXN]**
	Min	*Q25*	*Q50*	*Q75*	*Max*	*Mean*	*Std*	
3302669	83	263.00	663.5	1024.00	2116	708.21	554.29	517,482.09
3300663	3	11.75	20.0	49.00	97	30.71	27.59	370,028.77
3500322	20	37.00	43.0	90.50	146	61.79	35.72	210,346.97
3300142	1	4.00	6.5	8.00	17	6.58	3.79	142,854.83
3301771	20	52.75	77.5	93.75	160	77.17	34.13	130,868.65
3301778	23	47.75	78.0	146.75	401	118.46	94.29	129,145.96
3300115	2	4.00	8.5	43.00	78	25.04	25.39	127,295.20
3302934	1	4.00	6.0	8.25	11	6.13	2.69	85,727.93
3302307	1	26.25	44.5	63.00	112	46.54	29.34	80,901.40
							Total	1,794,651.80

Furthermore, when visualizing the time series for each of our drugs of interest like in figure 4, we realized that some variables had a strong correlation, the *Amount Consumed* being our independent variable and the rest of the amounts being our covariates. If we look at the correlation between these time series in figure 5 we realize that the highest correlation exists between the *Purchased* vs *Received*, followed by *Requested* vs *Purchased*, and *Consumed* vs *Requested*. This means that the pipeline of drug consumption, requisition, purchasing and reception is consistent for our time series.

Figure 5: Time series correlation matrix between time series for our top 9 drugs



3.3 Modeling

After establishing which patent drugs have had the most economic impact for the pharmacy, we decided to tackle the inventory shortages problem by anticipating the future consumption of these drugs based on data recorded throughout January 2018 - December 2019. This forecasts, complemented with an automated inventory control system, would be of great value for the project stakeholder to anticipate supply decisions.

3.3.1 The Mean Approach

Dalinde's pharmacy does not have a prediction model that can be used to plan their budget in advanced. To establish a baseline model that we could build on, we created a **simple benchmark** proposed by our Teaching Assistant, Theresa Gebert. This benchmark was made by calculating the month average of the *Amount Consumed* for our top 9 patent drugs. The month average is defined as the average of January (2018 and 2019) to compute the forecast for January 2020. The same methodology was used for February and March 2020. The forecasts calculated with this *Mean Approach* are presented in table 4.

Table 4: Baseline model forecast and Actual consumption

MedCode	**January 2020**		**February 2020**		**March 2020**	
	Mean consumption forecast	*Actual consumption*	*Mean consumption forecast*	*Actual consumption*	*Mean consumption forecast*	*Actual consumption*
3300115	34	4	36	5	42	11
3300142	3	4	7	5	6	2
3300663	44	35	18	5	51	12
3301771	112	72	91	16	63	99
3301778	340	22	210	6	154	3
3302307	18	5	22	0	15	0
3302669	1992	245	997	125	547	190
3302934	8	4	6	4	6	5
3500322	40	85	32	92	58	88
MAE	245		142		74	

The main objective of this *Mean Approach* is not to make a proper prediction, but to place a benchmark to see if any of our models performs better than the average.

3.3.2 ARIMA

ARIMA stands for an **Autoregressive Integrated Moving Average** time series statistical process. This approach is used to learn and understand the behavior of a statistical process across time, and it is very popular to perform time series forecasts. In the fields of economics, the ARIMA models are used to predict levels of supply, demand, GDP, etc.

Why ARIMA, ad how does it work?

Since in this business case we have time series data of the demand from patients for patent drugs, and we have continuous data for 87 patent drugs across 24 moths and at least 9 of those patent drugs also have reported some missing orders, the ARIMA methodology fits well to our problem. The

ARIMA is part of the ARIMAx family models, and a generalization of the Autoregressive Moving Average (ARMA) Models. The ARIMA models have 3 parameters that must be learnt:

- **(AR) Autoregressive parameter**: In an autoregression model, we forecast the variable of interest using a linear combination of past values of the variable. The term autoregression indicates that it is a regression of the variable against itself. We refer to this as an $AR(p)$ model, an autoregressive model of order p. [1]
- **(I) Integration parameter**: This parameter indicates the degrees of differentiation a time series needs order for it to be stationary. In order to check whether a Time Series needs to be differentiated we have to perform a Dicky-Fuller test for unitary root. [2]
- **(MA) Moving Average parameter**: Rather than using past values of the forecast variable in a regression, a moving average model uses past forecast errors in a regression-like. We refer to this as an $MA(q)$ model, a moving average model of order q. model.[3]

The general mathematical structure for an ARIMA (p,d,q) model is the following:

$$y'_t = c + \phi_1 y'_{t-1} + ... + \phi_p y'_{t-p} + \theta_1 \epsilon_{t-1} + ... + \theta_q \epsilon_{t-q} + \epsilon_t$$

where: y_t is the differenced series (it may have been differenced more than once).

Statistical Learning with ARIMA

The task was to train the 9 time series, each one for its respective *MedCode*, from January 2018 to December 2019 (24 months) and forecast the amount consumed for January, February, and March of the year 2020 (3 months). The challenge was to train 9 different ARIMA models and learn their (p, d, q) parameters for each one of the time series. Not all time series have the same structure, so the methodology for all of the 9 time series was the following:

1. Perform a Dicky-Fuller (DF) test of unit root to check if the time series was stationary.
2. Plot the Partial AutoCorrelation Function (PACF) to verify the $AR(p)$ term and suggest an order.
3. Plot the Autocorrelation Function (ACF) to verify the the $MA(q)$ term and suggest an order.
4. Estimate the ARIMA model with the parameters suggested from our observations of the DF test, the PACF, and the ACF.
5. Recalculate with different values for the (p, d, q) parameters and compare the ARIMAs with their Akaike Information Criteria (AIC).
6. Select the $ARIMA(p, d, q)$ with the lowest AIC for each of the 9 time series.

Forecasts with ARIMA

After performing the already mention methodology we can see a clear example for **MedCode 3300115**. The patent drug with MedCode: 3300115 had one of the best forecasts along with the patent drug with MedCode: 3300142. From figure 6 we can see the 24 months of training data (blue line), the first three months of 2020 with the actual amount consumed (black line), the forecast for those three months (dark green line), and the 95 percent confidence interval in grey. The suggested

[1] Forecasting: Principles and Practice, Rob J Hyndman and George Athanasopoulos. Retrieved from: https://otexts.com/fpp2/AR.html

[2] You can find a deeper explanation about this from Forecasting: Principles and Practice, Rob J Hyndman and George Athanasopoulos, https://otexts.com/fpp2/stationarity.html

[3] Forecasting: Principles and Practice, Rob J Hyndman and George Athanasopoulos. Retrieved from: https://otexts.com/fpp2/MA.html

ARIMA order for this drugs is a (3,0,0). Since we failed to reject the null hypothesis for unit root in the DF test, no differentiation was needed. The results for each of the time series and their respective forecasts for January, February, and March 2020 are in table 5:

Figure 6: ARIMA forecast for MedCode: 3300115

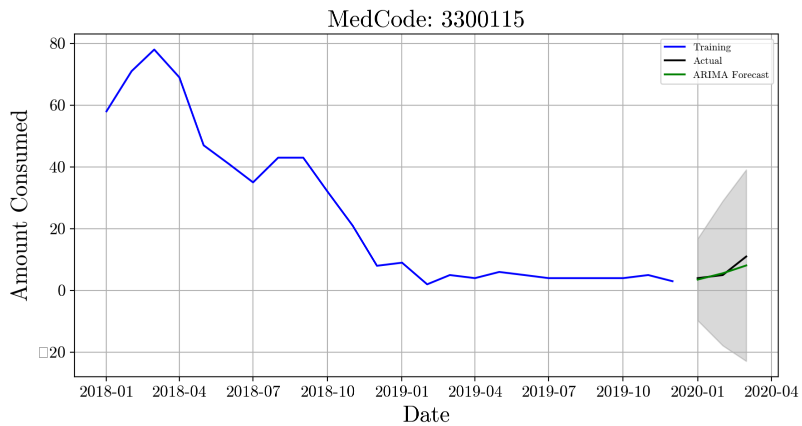


Table 5: ARIMA Forecasts Consumption and Real Consumption

MedCode	*ARIMA Order (p,d,q)*	Jan		Feb		Mar	
		ARIMA Consumption Forecast	*Real Consumption*	*ARIMA Consumption Forecast*	*Real Consumption*	*ARIMA Consumption Forecast*	*Real Consumption*
3300115	*(3,0,0)*	4	4	6	5	8	11
3300142	*(2,1,2)*	6	4	6	5	4	2
3300663	*(1,1,1)*	17	35	17	5	18	12
3301771	*(1,1,1)*	78	72	76	16	76	99
3301778	*(2,1,0)*	27	22	16	6	5	3
3302307	*(6,0,0)*	13	5				
3302669	*(5,2,1)*	259	245	280	125	266	190
3302934	*(4,2,1)*	6	4	8	4	9	5
3500322	*(1,2,1)*	78	85	79	92	81	88

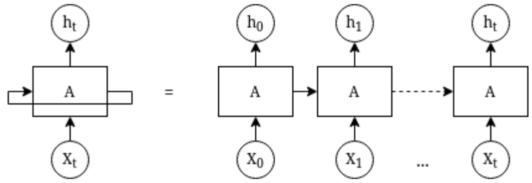
As we can see, each time series has different parameters that were learnt in order to make the best predictions using the ARIMA methodology, and the results are similar for some time series and significantly different to others. The next step that we took was try to create better forecast with a Recurrent Neural Network.

3.3.3 Recurrent Neural Network

Our third modelling approach was to implement a Recurrent Neural Network (RNN) for forecasting the monthly demand for each drug of interest. A RNN is a kind of Neural Network well suited for learning sequential data, such as time series, by implementing loops in its layer architecture (figure 7) that allow information to persist in subsequent iterations, helping the model *remember* data that it has seen before[4]. As shown in figure 7, the loop in a RNN unit can be understood as a series of neurons that pass data sequentially at different time steps. A LSTM unit, A, takes an input x_t from a previous layer and outputs a value h_t after iterating over the sequential data.

[4]Ming, Yao & Cao, Shaozu & Zhang, Ruixiang & Li, Zhen & Chen, Yuanzhe & Song, Yangqiu & Qu, Huamin. (2017). *Understanding Hidden Memories of Recurrent Neural Networks*. Retrieved from: https://www.researchgate.net/publication/320726805_Understanding_Hidden_Memories_of_Recurrent_Neural_Networks

Figure 7: Visualization of a Recurrent Neural Network unit



A traditional RNN will give more importance to immediately preceding data. However, when forecasting sequences, sometimes older data is as relevant as more recent data. Fortunately, there is a solution to this issue.

Long Short-Term Memory (LSTM) is a special kind of RNN that is good for *remembering* older relevant information when learning sequential data. It implements three interacting gates that help relevant information to persist while irrelevant information is discarded from earlier steps all the way to the latest steps. The *cell state* of a LSTM unit is defined by these gates:

- *Input gate*, which controls how much of the previous cell value flows into the new cell
- *Output gate*, which controls how its current state affects the activation of the unit
- *Forget gate*, which regulates what information is ruled out from the current cell state

Model implementation

A Neural Network with three layers, one LSTM layer of 16 units and two dense layers, was trained using only drugs with complete data, that is, those that presented observation points through all 24 months of 2018 and 2019. Their 24 attributes were differentiated to remove trends in the time series, and then normalized using this equation before being fed to the model:

$$x_{norm} = \frac{x-\bar{x}}{S}$$

Where x are the attributes, $\bar{x}$ and S are the mean and standard deviation of the samples. Normalization ensures a better fit for the Neural Network [5]. The effect of these transformations is shown in figure 8.

Forecasting results

A validation split was used when training the model to monitor its performance and prevent overfitting. Finally, the performance of the model was measured for our top 9 drugs which comprise the test-data split. The forecasted values are presented in table 6.

We found that on average, the model is able to predict the demand one-month ahead with an error of 18 units.

3.3.4 MAE, MAPE, and Performace

Now that we have the results and forecasts for the three used models, we can see an example of Med-Code: 3302934 and respective results of the models. From figure 9 we can see that the ARIMA makes

[5] Stöttner, T. (2016). *Why Data should be Normalized before Training a Neural Network.* Retrieved from: https://towardsdatascience.com/why-data-should-be-normalized-before-training-a-neural-network-c626b7f66c7d

Figure 8: Data differentiation and normalization for MedCode: 3301752

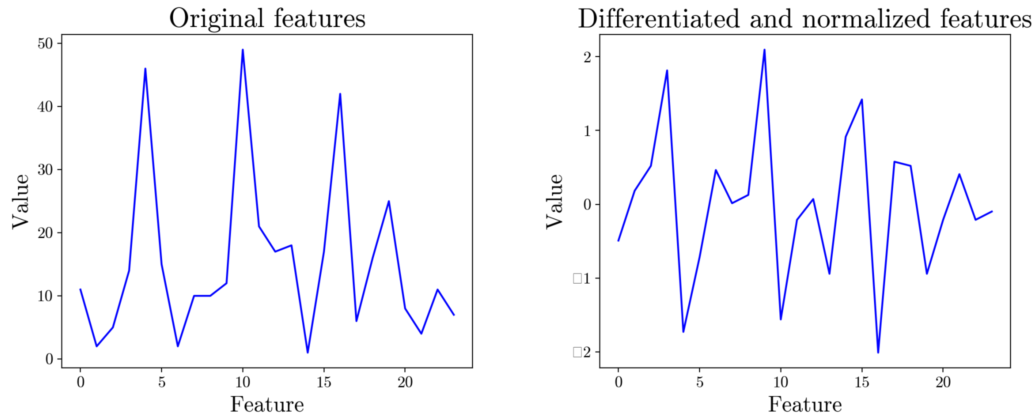
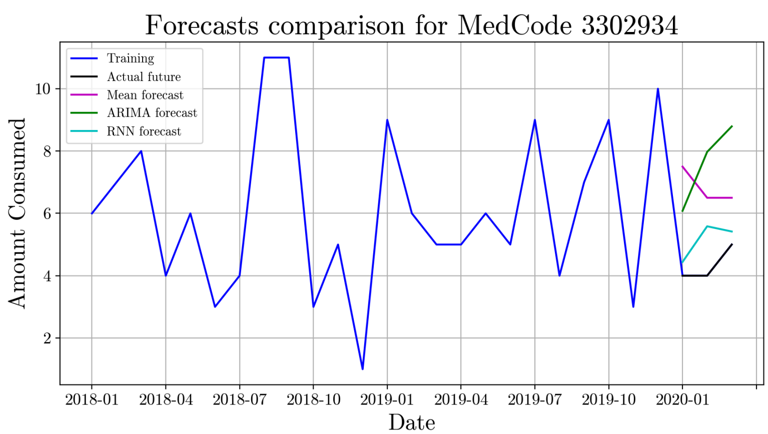


Table 6: RNN forecast and Actual consumption

MedCode	January 2020		February 2020		March 2020	
	RNN consumption forecast	*Actual consumption*	*RNN consumption forecast*	*Actual consumption*	*RNN consumption forecast*	*Actual consumption*
3300115	7	4	0	5	0	11
3300142	3	4	4	5	4	2
3300663	45	35	18	5	21	12
3301771	73	72	59	16	64	99
3301778	39	22	17	6	6	3
3302307	17	5	19	0	17	0
3302669	365	245	108	125	86	190
3302934	4	4	6	4	5	5
3500322	87	85	89	92	92	88
MAE	18		13		21	

the worst prediction, even below the Mean benchmark, but the RNN is the model that outperforms overall in this case.

Figure 9: Forecasts comparison for MedCode: 3302934



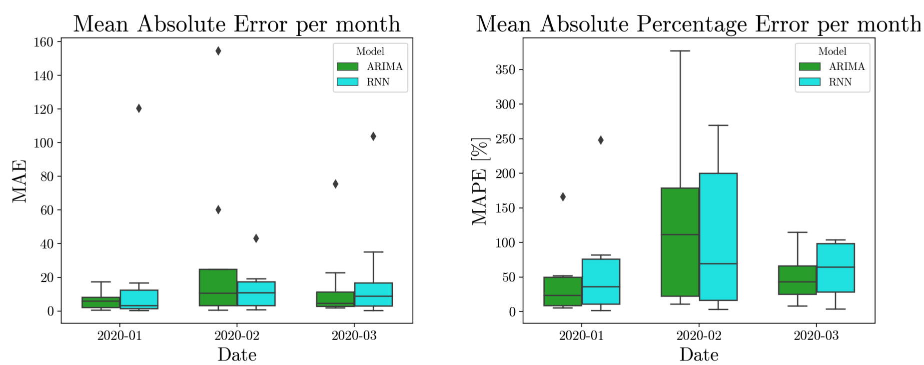
In order to check which of our models predicted the best forecasts, we must compare the prediction errors for each time series across our models. The Errors that we decided to use in this business case are the Mean Absolute Error (MAE) and the Mean Absolute Percentage Error (MAPE). The reason for using this errors and not other metrics is simple, the number of patent drug boxes consumed is an integer and an absolute number. Table 3 shows each time series with their respective errors for the three models : Mean baseline, ARIMA, and RNN.

Table 7: MEA and MAPE comparisons

MedCode	**Mean Absolute Error**			**Mean Absolute Percentage Error**		
	Mean	*ARIMA*	*RNN*	*Mean*	*ARIMA*	*RNN*
3300115	30.5	1.3	6.4	548.3	16.5	94.1
3300142	2.5	1.7	1.5	96.7	59.8	52.6
3300663	20.5	11.7	10.6	204.0	112.6	120.3
3301771	50.2	29.8	26.6	186.7	136.2	102.5
3301778	224.2	5.6	10.2	3289.4	82.0	118.6
3302307	16.7	8.3	16.1	250.0	166.4	248.4
3302669	992.2	81.2	80.6	532.9	56.3	39.3
3302934	2.5	3.3	0.8	60.0	75.7	19.7
3500322	45.0	9.3	3.0	50.7	10.4	3.4
Average	153.8	16.9	17.3	579.9	79.5	88.8

From table 7 the models that reported the least MAE and MAPE for each time series are marked in light blue. Firstly, we can observe that the base line model (Mean Approach) does not report the least error, neither for the MAE, nor for the MAPE. Secondly, the ARIMA model beats the RNN three out of nine times according to the MAE, and four out of nine when comparing to the MAPE; in all other cases the RNN performs the best. Lastly, we can see that the lowest average of all the MEA and MAPE for the nine time series is the ARIMA. But this metric could be missing important information about the steps ahead forecasts.

Figure 10: Errors of ARIMA and RNN forecasts per month



When we plot the MAE and the MAPE as time series boxplots (figure 10), we can observe that both the ARIMA and the RNN make very good forecasts for January, one step ahead forecast. In the case of February and March the both MAE and MAPE increase for our models. This analysis indicates that attempting to forecast beyond one step ahead significantly decreases the accuracy of our predictions.

3.3.5 The Monetary & Business Value of the Forecasts

Under the assumption that the *Amount Purchased* should be the same as the *Amount Consumed* in order to maintain a constant inventory level[6]. We exemplify the goodness of our predictions with the most demanded and with the highest monetary cost ($517,482 MXN) MedCode: 3302669 with an one step forecast.

The amount consumed for the MedCode 3302669 on January 2020 was 245 units. Knowing that the ARIMA had the best performance on forecasting January 2020 (by looking at the MAE and the MAPE in figure 10) we use its estimate to quantify the absolute difference with the real amount consumed and compare it with the real amount purchased (220).

$$AbsoulteDifference_{real} = |AmountConsumed_{January} - AmountPurchased_{January}| = 25$$

$$AbsoulteDifference_{forecast} = |AmountConsumed_{January} - ARIMAForcast_{January}| = 14$$

The unitary price for MedCode: 3302669 reported in January 2020 is $152.48 MXN. Multiplying the $AbsoulteDifference_{real}$ times the unitary cost we get $3,812 MXN; and multiplying the $AbsoluteDiffernce_{forecast}$ times the unitary cost we get $2,134.72 MXN. This means the forecast could have saved $1,677.28 MXN for the Dalinde Hospital in one month for this MedCode: 3302669.

If we take the unitary cost and multiply it times *Amount Consumed* we get $ 37,357 MXN of total cost. By taking the ratio of the absolute saving and the total cost for January 2020 we get:

$$SavingPercentage = \frac{1{,}677.28}{37{,}357} = 4.48\%$$

Extrapolating this results for our 9 MedCodes, and assuming that this will be an average behavior we could save 4.48% of 1.7 million MXN, giving a total amount of $89,732.59 MXN. Which is more that the expenditure of our 8th or 9th most relevant MedCodes (3302934, 2202307).

3.4 Inventory Control with a PID controller

The levels of inventory are parameters that bring up 3 immediate challenges:

1. If the inventory stock level is high, we have maintenance costs to attend, like: storage rent, staff for managing it, utilities (electricity), etc.
2. A high level of inventory represents a large investment that is not being recovered.
3. A low level of inventory represents scarcity or shortage, which could translate to loss of money when bringing care to patients.

Therefore, we see that an optimal inventory level could exist at minimal costs. To find this optimal level, we will use a control technique called PID, which stands for Proportional, Integral, and Differential control.

Inventory Management

To properly manage an inventory, the following policies are required:

[6] We ignore the current and optimal inventory level

1. Buy a defined amount (Q) when: AI (Actual Inventory Level) $< OL$, Optimum Level (of Required Inventory)
2. Buy the difference between $OL - AIL$ when: AIL, Actual Inventory Level $< OL$, Optimum Level (of Required Inventory)
3. If a defined time period passes, no matter what the inventory level is, one buys only when: $OL - AIL$ is positive.

With the given data, we can estimate the current inventory levels. This estimation cannot be reviewed with the Outflows column in the purchase orders report because when subtracting the *Received Amount* from the *Consumed Amount*, some results are negative and they shed light on unregistered shortages.To further review the average inventory stock levels, we perform a linear regression of the levels of inventory. The results are the following:

Table 8: Regression Estimates for Inventory Levels

MedCode	Slope
3302669	0.06749
3302307	-0.10765
3301778	0.04641
3301771	-0.00113
3300115	0.00162
3500322	0.00042
3300663	0.01900
3302934	0.00086
3300142	-0.00220

From the linear regressions, we conclude the following:

- There are 6 slopes very close to zero, and with little tendency to be over zero, which means that the current methodology for inventory control does not have the capacity to move the inventory from being on shortage.
- In the case of the 3 negative slopes, for MedCode: 3302307 there is not enough data, which makes it not valuable for our analysis. For MedCode: 3301771 the slope is valid for our analysis, but it is negative and so small that could be considered zero; this demonstrates the lack of capacity to move the inventory stock level from being on shortage. And lastly, MedCode: 3300142, shows a cumbersome behavior because it tells that the current methodology does not respond well to changes of demand levels, and on top of this, what makes it even worse is the demand that has a negative slope.

3.4.1 PID as an Opportunity

1. PID is designed for a system, in this case an inventory, to stay at a constant level despite of perturbations, which in our case, is the demand. PID suggests how much volume a purchase should have to keep constant inventory levels. **It is very clear that the inventory suffers from shortages given that its levels are always below zero, and an increase in volume is needed to reach an optimal level.**
2. When there is a perturbation, PID suggest purchases to be done most optimally for the stock inventory level to return as fast as possible to its required level without being affected by overpurchases or shortages. **It is clear that the demand is the cause of many perturbations of inventory stock levels, as we can observe in cases like MedCodes: 3301778, 3500322, and 3300663, being the most notorious ones.**

The following equation shows the integro-differential equation of a PID, where $e(t)$ is the error to correct and, respectively, K_P, K_I and K_D are the proportional, integral and differential coefficients of the controller:

$$PID(t) = K_P e(t) + K_I \int_0^t e(t)dt + K_D \frac{de(t)}{dt}$$

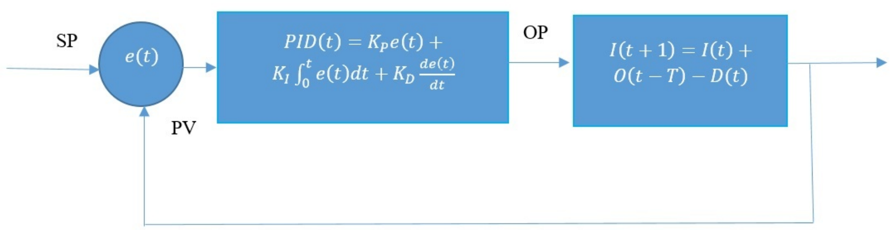
The best description of the process that we are interested is the following equation (B):

$$I(t+1) = I(t) + O(t-T) - D(t)$$

Where: $I(t+1)$ is the inventory at end of the day. $I(t)$ is the inventory at beginning of the day. $O(t-T)$ is the patent drugs that arrive today and were requested some time ago (T). $D(t)$ demand of patent drugs.

The model shown in Equation B is very similar to the model that simulates the integration of a plant with processes that represent delays in the responses of systems. To solve these models, we utilize a first-order Padé approximation in the Laplace space.

Figure 11: Closed-loop block diagram of the PID controller



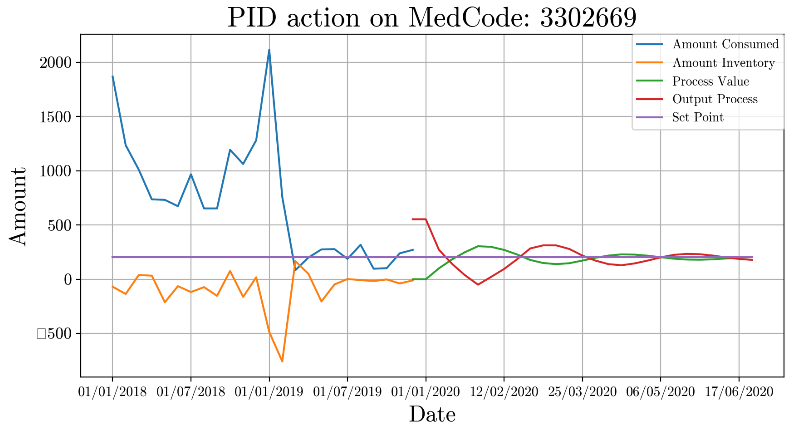
The process diagram above shows in which we can appreciate both equations in the Laplace space in a closed loop diagram. On this diagram we can observe how our required inventory level exists as *Set Point* (SP), and the current inventory level or *Process Value* (PV) is inserted in the PID model for processing, and the suggested volume output will be the *Output Process* (OP). This variable enters the demand process, and the output is the level of inventory left. This result is compared again with the required inventory level, and repeats the process. After some iterations, we get the desired inventory level. The speed to reach the desired inventory level will be determined by the constants in the PID that are shown in the Laplace space as K_P, K_I and K_D.

3.4.2 Implementation in our historical data

To implement the model: 1) We first assume that demand is at a quasi-stationary level, so we can set an inventory level adjusted to that demand. 2) To do this we take the data from the last year and perform a linear regression with which we obtain a line with an almost zero slope. 3) We evaluate the point-to-point line to obtain the mean, that is, the midpoint over which demand fluctuates. 4) This average will be our Set Point or optimal level. This Set Point is the basis for the controller to adjust to that demand.

Figure 12 shows the result when we apply the PID model to MedCode: 3302669. We see that: *Process Value* (*PV*) is the inventory we have when applying the model. As you can see it is swinging

Figure 12: Action of the PID controller over the inventory system for MedCode: 3302669



around the desired inventory or *Set Point* (SP). *Output Process* (OP) is the purchase suggestion, the suggestion is high at the beginning but later they behave in an oscillatory way, tending to stability. The final step is to implement the methodology for all the patent drugs. Adjusting each with its own Gains an optimal Set Points for the inventory.

4 Conclusions and Future Work

Working on this business case was an amazing challenge for our team. Firstly, we needed to build a proper business case and to solve it with data science and machine learning tools. Secondly, the members of our team come from different backgrounds, and taking advantage of our capacities was something that we had to discover along the way. Finally, forecasting is hard and trying to predict the future takes courage and humility to pay the cost of errors.

Regarding the technical issues, we can say that **time series analysis is a living thing**. The more historic information and the more diverse data that we could have, the better the forecasts will be. It's hard to predict more than one step ahead with the data that we have. Even though the predictions for more than one step ahead are not as accurate, **one month prediction is extremely valuable**. These results enhance the strategic planning for supply of patent drugs, due to the anticipation on patent drug demand one month before. The web dashboard tool is of great value for the pharmacy personnel, since it shows historical information of the patent drugs and makes a buying suggestions based on the ARIMA, RNN models and the PID controller designed.

As discussed in subsection 4.3.5 *The Monetary & Business Value of the Forecasts*, **the estimated savings of implementing our proposed solution represent a 4.48% of the total costs for our drugs of interest**. **This type of analysis can be reproduced to other kind of drugs, such as generic; and to other kind of hospital supplies**, such as bandages, syringes, and other consumable items, broadening the impact of implementing the tools of Data Science and Machine Learning. The joint benefits of forecasting the demand for hospital consumables and controlling its inventory will definitively increase the percentage of savings for the health care system, guiding their budget decisions and providing a more efficient health security for their patients.