

FINAL REPORT

LIQUORPREDICTOOL: VISUALIZING AND PROJECTING TAX AND TAX STAMPS ON LIQUOR FOR ANTIOQUIA'S DEPARTMENT

September 3, 2021



Camilo Antonio Albarracin

Juan Pablo Urrutia

Esteban Caicedo Graciano

Samuel David Cudriz

William Arcos Martinez

Jhoan Steven Delgado

Application deployed at: <https://liquorpredictool.net/>

CONTENT

INTRODUCTION AND BUSINESS CONTEXT	2
SOLUTION SCOPE	3
DATA OVERVIEW	4
Raw datasets	4
Exploratory analysis	5
MODEL	8
Data Preparation	8
Model Selection	9
Analysis & Interpretability	11
WEB APP	12
Application Infrastructure	12
Application overview	13
CONCLUSION	19
FUTURE WORK	20
REFERENCES	21

1. INTRODUCTION AND BUSINESS CONTEXT

The excise tax or consumption tax in this document refers to the excise tax on liquors, wines, aperitifs and similar products in the department of Antioquia. This tax is collected by the department, and the people responsible for paying the tax are the producers, importers and jointly the distributors of any liquor, wine, aperitifs and similar. Moreover, the transporters and retail vendors are also responsible when they cannot justify the origin of their products. There are several laws and decrees that regulate the collection of this tax, how it is calculated, how it is collected, how it is sanctioned in case of evasion and in general everything around the production, transportation and commercialization of liquor; however they are beyond the scope of this report and were only used as context for the resolution of the problem.

Having explained the consumption tax, the department of Antioquia is the entity that supplies the signage elements for the liquor excise taxpayers. These signage elements are mandatory for most liquors since they guarantee that they have a known and certified origin, that it's suitable for human consumption and that it has its tax obligations fulfilled (has not been smuggled). The simplified flow of events is like this: the taxpayers request a certain number of signage elements to the department providing documentation on the goods they're trying to legalize, this is made through some forms that must include the pricing of the elements and a calculation of the amount of taxes that is due to pay for that amount of signage elements; after the payment has been completed and the forms have been validated, the department provides the taxpayer with the signage elements so that the different products are labeled as legal and ready to sell.

Although there have been some improvements on the process and most of the data is being stored electronically in databases, the department does not have a tool that would allow it to rapidly see trends or statistics on important variables such as amount of tax collected, number of stamps delivered, the most common signed product, etc. The department would also like to have an estimate on how many signage elements would be requested in subsequent periods and with that estimate also the return in taxes that they would get. This is important since liquor stamps represent operating expenses for the entity. On the other hand, consumption tax represents an important income for Antioquia. Addressing this problem will help them identify the taxpayers and products that are generating the most tax revenues for the department and those that generate the least. At the same time, it will help them minimize the operating cost involved in the delivery of liquor stamps so different strategies can be implemented to increase tax collection for these revenues. Overall improving the public finances of the department.

2. SOLUTION SCOPE

For the solution of this project we have developed a self-sustainable online application that contains various sections, each section addresses a specific need and contains data visualizations (bar, lines, scatter, box plots, KPIs, etc.). Some of these visualizations are interactive, where different filters and selections can be made through web components so the user can select which data to visualize and how. Most of the requirements have been addressed in this section and we will explore them in detail later.

This application was created using the knowledge gain through the DS4A course in all of its lectures, using a set of conventional Python libraries to perform tasks of cleaning, transforming, graphing and modeling of the data, there were some other libraries and frameworks used to create the web application and some knowledge of cloud architecture involved in making the application online.

The solution takes into account all the data provided by the department of antioquia regarding the consumption tax, no additional data sources were needed to complete the project however new data can be added to the application if needed so the application updates all its KPI's and graphs.

A specific requirement for the solution was that the solution could be used throughout time, having a constant flow of input data and updating the projections when new data is added. This was one of the key points to work on, and it is explained further on.

3. DATA OVERVIEW

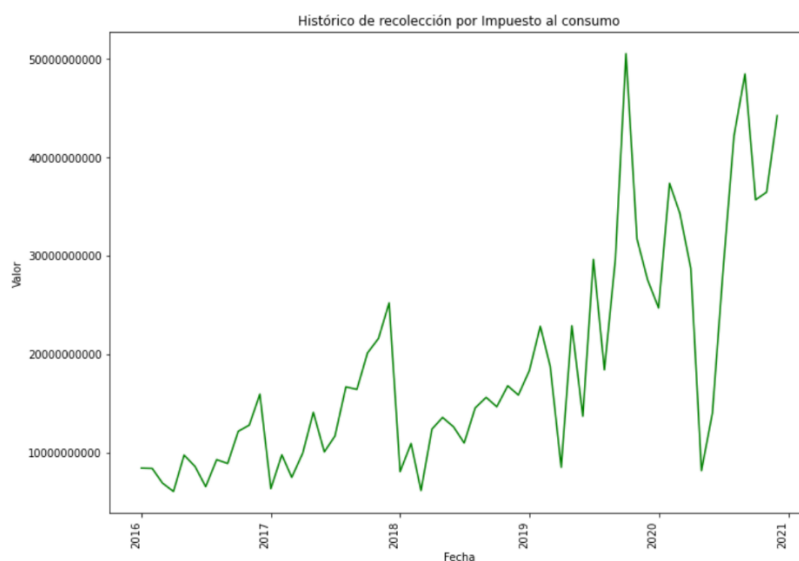
a. Raw datasets

Data sets	Source	Description	Dataframe features
Declaraciones Productos Importados	<ul style="list-style-type: none"> • Declaraciones de producto Importado 2016.csv, • Declaraciones de producto Importado 2017.csv, • Declaraciones de producto Importado 2018.csv, • Declaraciones de producto Importado 2019.csv, • Declaraciones de producto Importado 2020.csv 	<ul style="list-style-type: none"> • Files in csv format containing the tax information that companies contribute for each imported product • The information is divided by years in files where in each file there is information for each year from 2016 to 2020 	<ul style="list-style-type: none"> • Each file has 34 columns and between 18000 and 21000 rows • There are at least 15 measures and 19 dimensions with text, numeric and date data types
Declaraciones Productos Nacionales	<ul style="list-style-type: none"> • Declaraciones de Producto Nacional 2016.csv, • Declaraciones de Producto Nacional 2017.csv, • Declaraciones de Producto Nacional 2018.csv, • Declaraciones de Producto Nacional 2019.csv, • Declaraciones de Producto Nacional 2020.csv 	<ul style="list-style-type: none"> • Files in csv format containing the tax information that companies contribute for each national product • The information is divided by years in files where in each file there is information for each year from 2016 to 2020 	<ul style="list-style-type: none"> • Each file has 33 columns and between 6000 and 9000 rows • There are at least 15 measures and 18 dimensions with text, numeric and date data types
Productos Estampillados	<ul style="list-style-type: none"> • Productos Estampillados 2016.csv, • Productos Estampillados 2017.csv, • Productos Estampillados 2018.csv, 	<ul style="list-style-type: none"> • Files in csv format containing the the quantity and cost of stamps issued by each product and company 	<ul style="list-style-type: none"> • Each file has 14 columns and between 16000 and 18000 rows • There are at least 2 measures and 12 dimensions with text, numeric and date data types

	<ul style="list-style-type: none"> • Productos Estampillados 2019.csv • Productos Estampillados 2020.csv 	<ul style="list-style-type: none"> • The information is divided by years in files where in each file there is information for each year from 2016 to 2020 	
--	----------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

b. Exploratory analysis

Since there were three different datasets we performed a separate exploratory analysis in each one to discover behaviors and select the variables that were important to the project. Even though the datasets were relatively clean, still some formatting and transformation was needed to work with them. There was also the need to create new date columns to aid the ease in the creation of the model or the presentation of different graphs. In this section we leave some of the graphs constructed during the preliminary exploratory analysis:

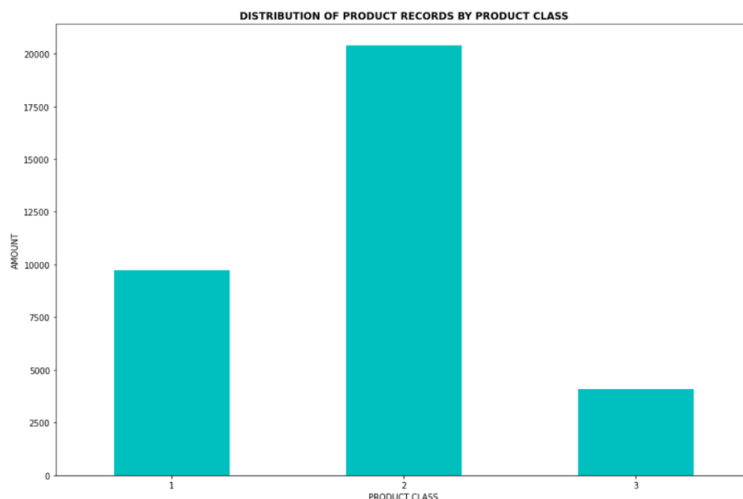


This first graph shows the timeline of collected excise tax. We do see a clear upwards trend through the years and some peaks in months that could imply seasonality, it is also worth mentioning the drop in the middle of 2020 possible due to the SARS-CoV-2 pandemic.

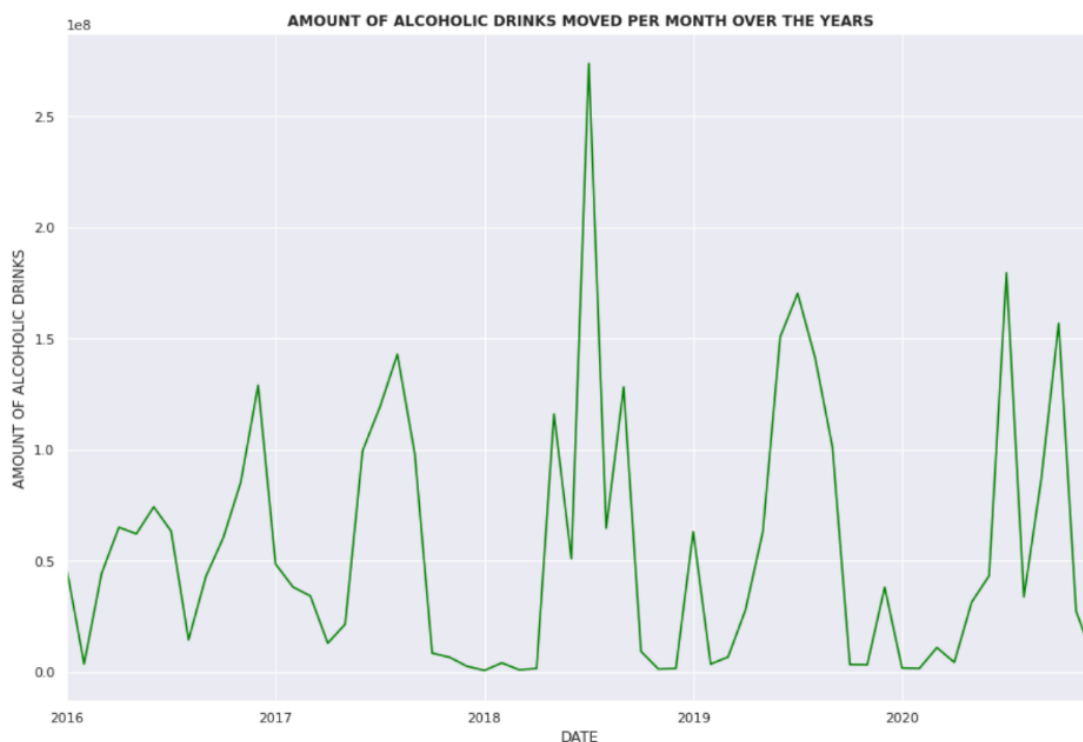
However, the patterns seen in the number of stamps (which bounces in July and at the end of the year) can be explained by the following:

The first week of August occurs the 'Feria de las flores' festival, which increases tourism in the department, and people usually ask for holidays at that time, which can increase

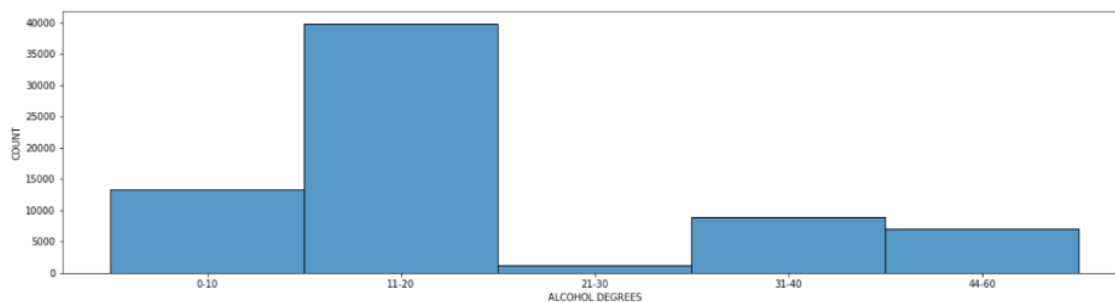
consumption. The month of December is the time of the Christmas festivities, which greatly increases the consumption of liquor. In addition, the large peak that occurred in June 2018 may be related to the sporting events that took place at that time (World Cup: Russia 2018).



The excise task is applied to three categories 1: Beers, 2: Liquors/wines/aperitifs and 3: Cigars/tobacco, however the stamps are only required on Liquors, wines and aperitifs, we can see their distribution on the chart and note that the category that requires the stamps is also the one with most records.



Even though the tax income had an upward trend, we do not see a clear tendency on the amount of alcoholic drinks that were registered over time. Note that there is a drop in 2020, and some kind of seasonality with some extreme values in the year of 2018.



Reviewing alcohol percentage present on the products we see that most of them are gathered around 11-20 degrees; hence soft drinks such as beers or heavy drinks such as whisky, vodka and the famous aguardiente don't appear as much in this dataset.

4. MODEL

In order to obtain the value of taxes to be collected in a period of time and the amount of stamps to be delivered by the government of Antioquia, two models were built to predict each of these variables. The machine learning regression model was chosen because it is a type of model suitable for predicting continuous values in time series. Also, the available dataset meets the characteristics to build this model.

A Python library called “Pycaret” was used to develop the model. PyCaret is an open-source, low-code machine learning library and end-to-end model management tool built in Python for automating machine learning workflows.

The main reason for using this library is the ease of use, the simplification of the processes, the speed with which it performs the tasks and the delivery of accurate results. In addition, due to the short time for the development of the solution, it is very useful to use this type of tools that help us to meet the delivery goals.

The steps involved in the development of the model were as follows:

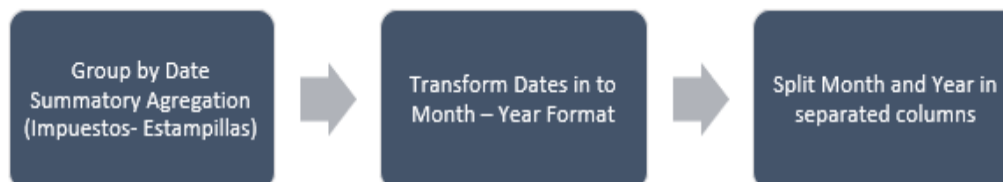


After model selection, a prediction was made on a future time series.

The following is a general description of what was done in each of these projects.

a. Data Preparation

Since there was no absolute continuity in the dates for all the characteristics, a process of grouping and ordering was carried out to train the model.



For this process the "setup" function of the Pycaret library is used where the following parameters are given:

- Train Dataset
- Test Dataset
- Target (Impuestos – Estampillas)
- Strategy (Timeseries)
- Numeric Features

b. Model Selection

Pycaret performs a comparison between all models to determine the one that yields the best metrics, in our case the model that best fit the solution was the Ridge Regression model.

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
ridge	Ridge Regression	255440.5000	100454741333.3333	305418.8646	0.3082	0.2432	0.1972	1.0367
knn	K Neighbors Regressor	266264.0908	134778858160.1200	364535.2185	0.1383	0.2517	0.1897	0.0200
rf	Random Forest Regressor	271669.0175	130421735046.1111	357477.0789	0.1529	0.2473	0.1888	0.1133
ada	AdaBoost Regressor	282888.3948	144575860016.0966	368956.8277	0.0409	0.2652	0.1988	0.0467
lar	Least Angle Regression	284032.2263	125999219466.1710	349175.9120	0.1588	0.2837	0.2108	0.0133
br	Bayesian Ridge	287499.4368	129374136935.3561	356607.1753	0.1765	0.2864	0.2180	0.0200
huber	Huber Regressor	290250.4012	130402075090.7115	348582.0491	0.0994	0.2906	0.2226	0.0300
lasso	Lasso Regression	295793.6562	126758346752.0000	353676.4688	0.2019	0.2551	0.2212	1.2867
en	Elastic Net	297257.6927	127387391317.3333	354203.6458	0.1923	0.2569	0.2227	1.0033
gbr	Gradient Boosting Regressor	298490.5448	165585905407.6006	395124.5092	-0.1180	0.2870	0.2049	0.0267
omp	Orthogonal Matching Pursuit	298965.2946	128589399230.8990	355405.7792	0.1783	0.2595	0.2244	0.0167
et	Extra Trees Regressor	315292.5647	172639183849.8826	413599.8996	-0.1088	0.2986	0.2242	0.0867
dt	Decision Tree Regressor	331008.6667	183830393181.3053	411778.6909	-0.2118	0.3080	0.2248	0.0167
llar	Lasso Least Angle Regression	383932.8109	192609321292.5194	436886.1722	-0.1670	0.3096	0.2932	0.0133
lightgbm	Light Gradient Boosting Machine	383932.8120	192609322295.2909	436886.1734	-0.1670	0.3096	0.2932	0.3133
par	Passive Aggressive Regressor	470653.3528	303791146819.0582	535943.8802	-0.9263	0.4088	0.3402	0.0167

Ridge regression is very similar to least squares, except that the coefficient ridge is estimated by minimizing a slightly different quantity. In particular, the regression

ridge regression coefficient estimates $\hat{\beta}^R$ are the values that minimize

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^p \beta_j^2 = \text{RSS} + \lambda \sum_{j=1}^p \beta_j^2,$$

where $\lambda \geq 0$ is a tuning parameter, to be determined separately and RSS is,

$$\text{RSS} = \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2.$$

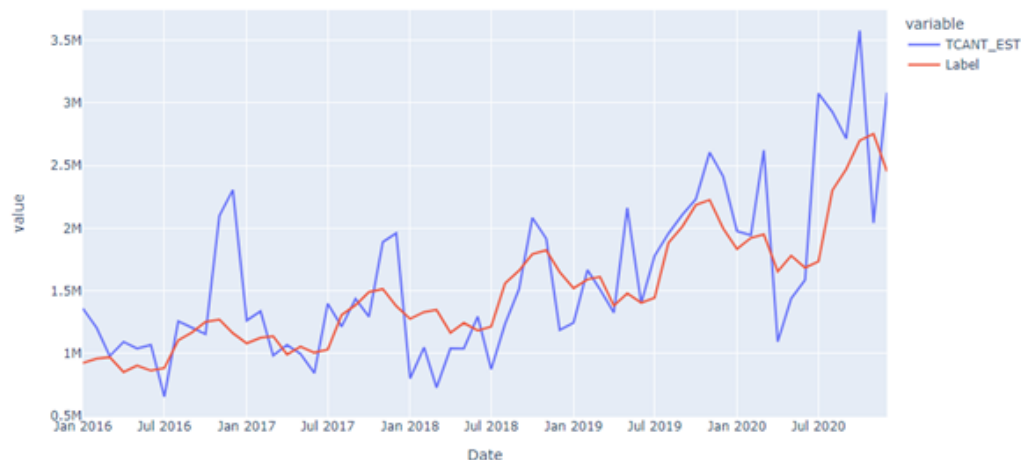
Ridge regression seeks coefficient estimates that fit the data well, by making RSS small. However, the second term, $\lambda \sum_j \beta_j^2$, called a shrinkage penalty, is small when β_1, \dots, β_p are close to zero, and so it has the effect of shrinking the estimates of β_j towards zero. The tuning parameter λ serves to control the relative impact of these two terms on the regression coefficient estimates. When $\lambda = 0$, the penalty term has no effect, and ridge regression will produce the least squares estimates. However, as $\lambda \rightarrow \infty$, the impact of the shrinkage penalty grows, and the ridge regression coefficient estimates will approach zero. Notice that, Ridge regression produces a different set of coefficient estimates, $\hat{\beta}_\lambda^R$, for each value of λ . (Selecting a good value for λ is critical).

The shrinkage penalty is applied to β_1, \dots, β_p , but not to the intercept β_0 , this is because the intercept is simply a measure of the mean value of the response when $x_{i1} = x_{i2} = \dots = x_{ip} = 0$. If one assumes that the variables have been centered to have mean zero before ridge regression is performed, then the estimated intercept will take the form $\hat{\beta}_0 = \bar{y} = \sum_{i=1}^n y_i / n$.

The project is then created and finalized with the functions "create_model" and "finalize model".

c. Analysis & Interpretability

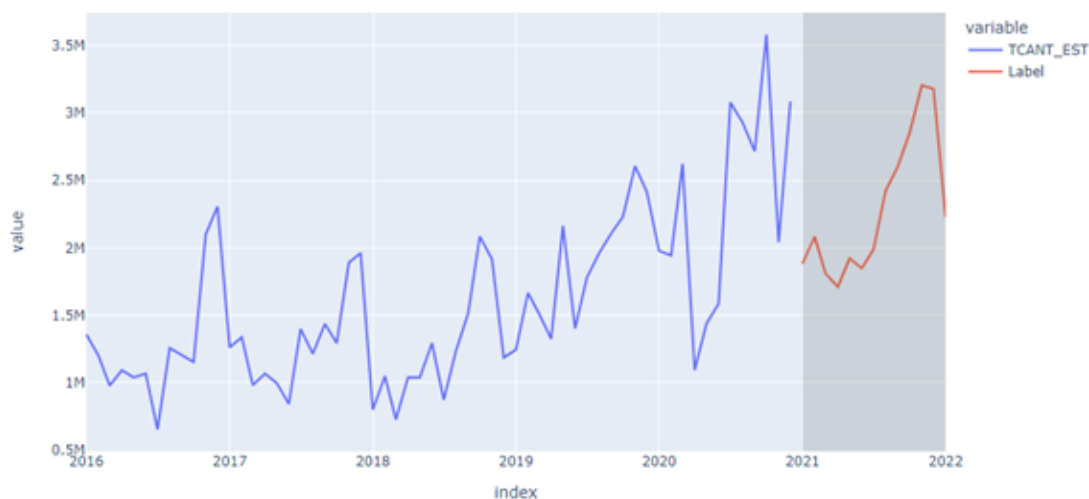
Validations are performed by constructing plots to review actual versus predicted data on the same time series:



Several tests were performed with various models, training data and time series to arrive at the choice of the optimal model for the solution.

For the prediction process, a future time series was created with a constant range between start and end dates. Subsequently, the prediction model was applied to this range with the function "predict_model", initially the range of these dates was the year following the last date reported in the dataset (2021).

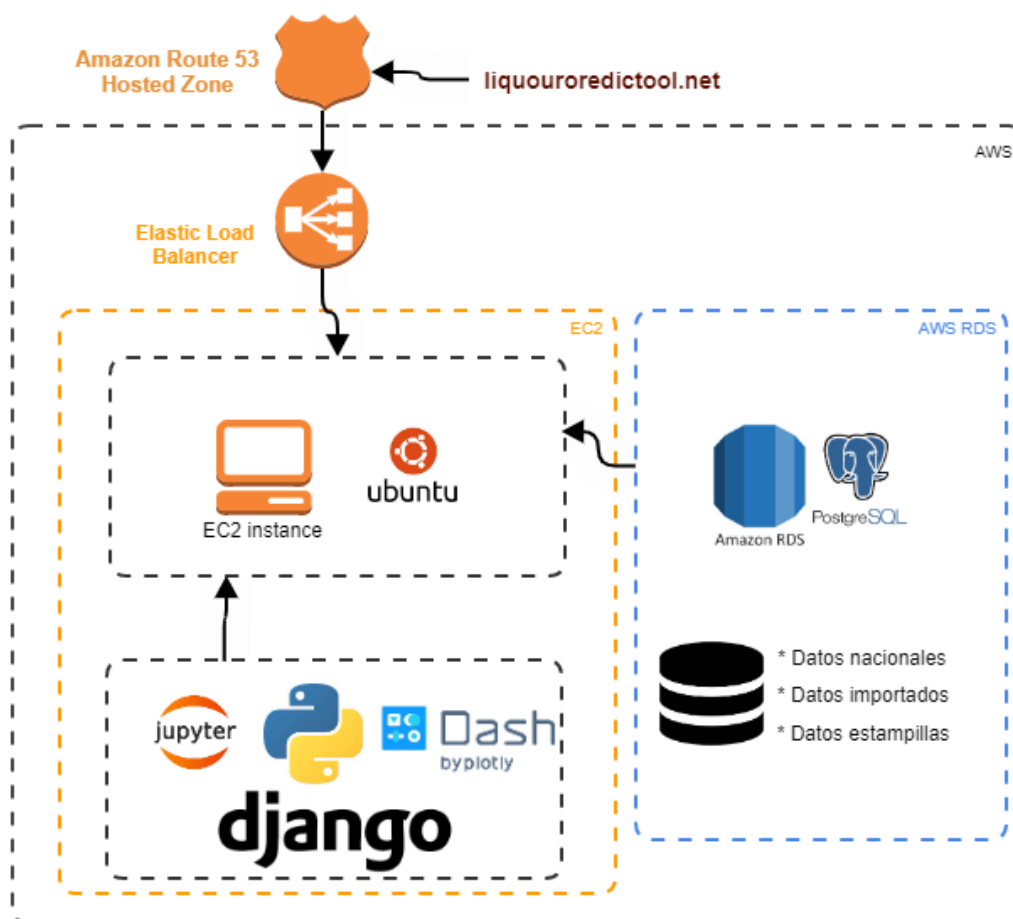
A new dataset was created where the real data and the prediction data were joined and then a line plot was created showing the real data on the blue line and then the prediction data on the red line. This graph is the final output of the model together with the prediction data.



5. WEB APP

a. Application Infrastructure

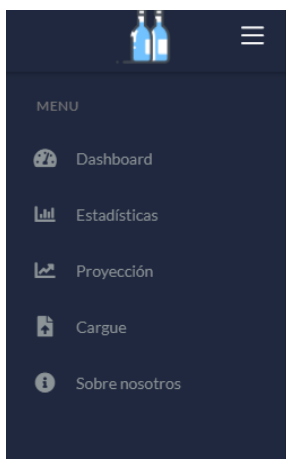
The solution's data was stored in a PostgreSQL database in AWS RDS. The domain was obtained through Route 53 (AWS), and the front-end is deployed in a EC2 instance, as shown below:



Since the department will upload new datasets at any time, various scripts had to be coded in order to transform and clean the raw datasets into something that can be used by the solution in order to create the predictions. Specifically, this was firstly designed in Jupyter notebooks, and accomplished with python files (.py). The final solution reads the new data in .csv format, performs the correspondent cleaning process, and uploads it to the database (hosted in RDS). In the same way, all the data used by the solution is called from the RDS service, through SQL sentences written in the code (and executed using SQLAlchemy).

b. Application overview

A web application was designed to show the user important information and analysis about the data and let them improve the model with new data. The application has 4 sections as shown in the image below:



- **Dashboard:** In this section important static information is displayed. The dashboard is divided in two sections: KPI section and Graph section.

- **KPI's Section:**



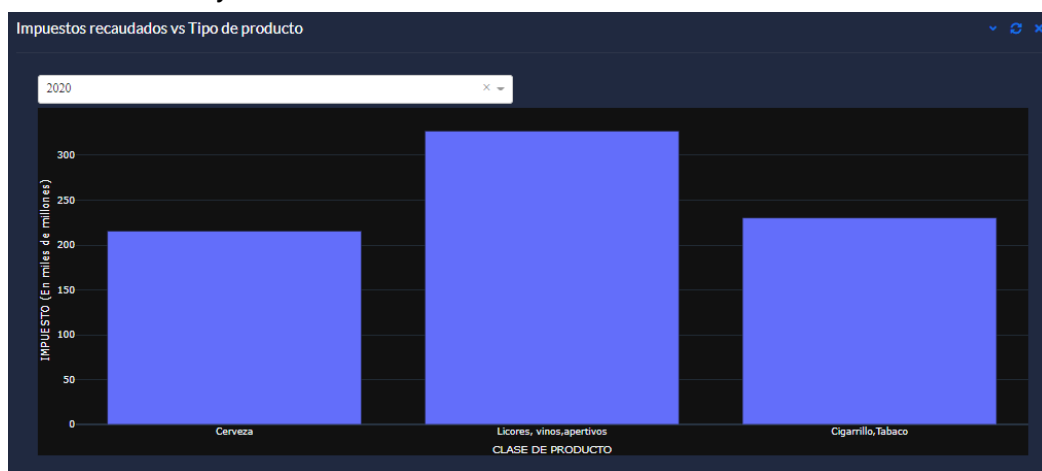
1. *Impuestos recaudados*: This KPI shows the total amount of taxes that were collected in a specific year. The user can select what year they want information for.
2. *Estampillas entregadas*: Here we show the total amount of stamps that were delivered in the year selected.
3. *Cantidad de empresas*: The total Company quantity that is in the system and that paid taxes.
4. *Variación impuesto vs periodo anterior*: This KPI shows a comparison between the amount of taxes collected in the previous year compared with the taxes in the present year. This information is shown as a percentage.
5. *Variación estampillas vs periodo anterior*: This KPI follows the same logic as the previous one, just that here we compare the total amount of stamps that were

delivered. So here we compare stamps delivered in the selected year compared with the previous one.

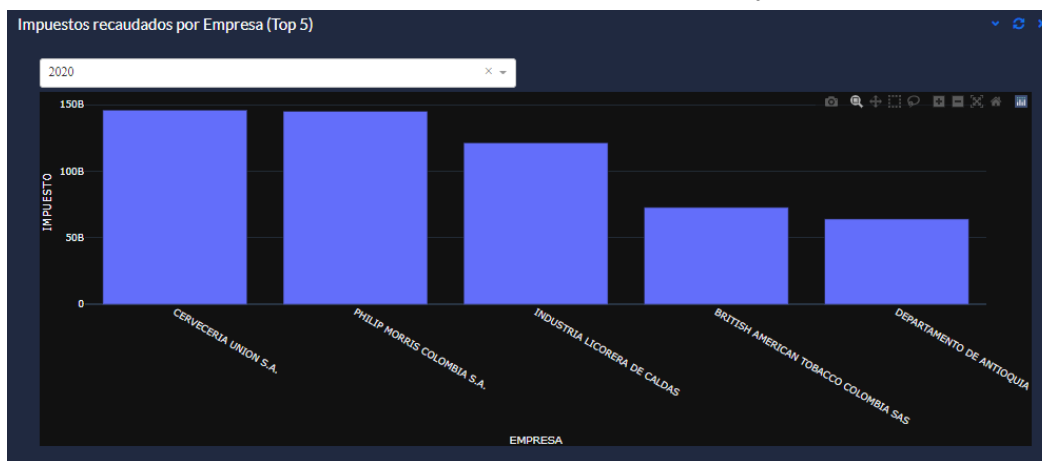
6. Cantidad de transacciones: information about the total amount of transactions made in the selected year.

- **Graphs Section:**

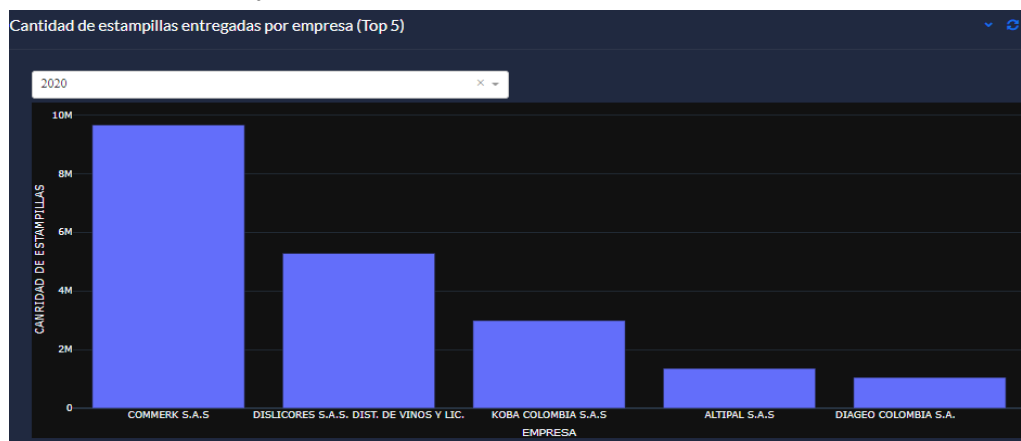
1. *Impuestos recaudados vs Tipo de producto:* In this section we show a bar graph showing the total amount of taxes collected in a specific year divided by brands.



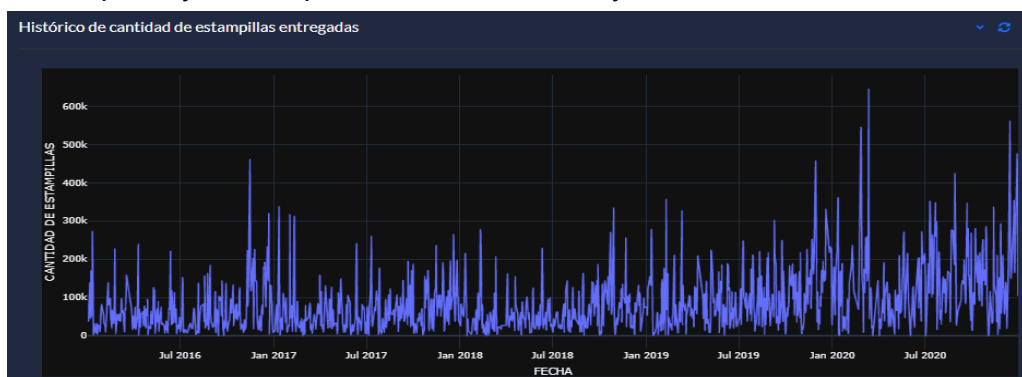
2. *Impuestos recaudados por empresa (Top 5):* Here we show a bar graph showing the total amount of taxes divided by company. These are the 5 companies that have paid more taxes in a specific year.



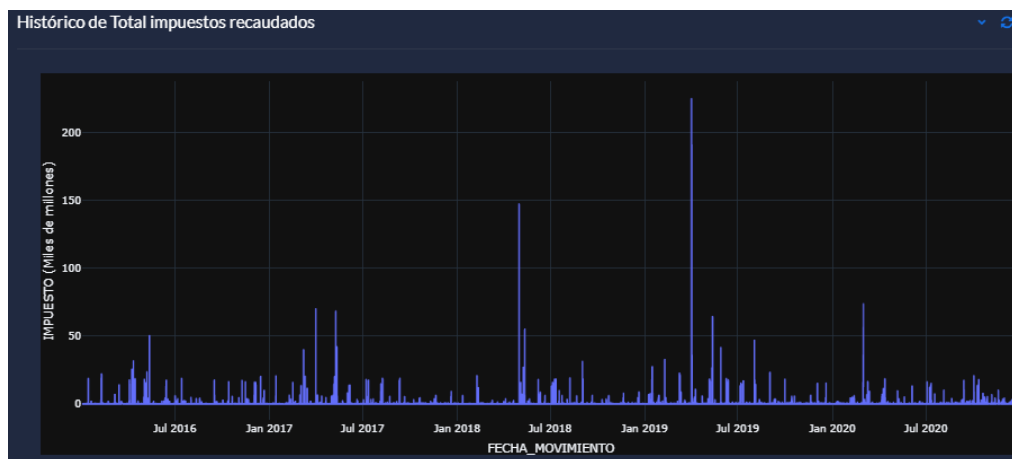
3. *Cantidad de estampillas entregadas por empresa:* We show the information about the 5 companies that have received the most stamps in the selected year.



4. *Histórico cantidad estampillas entregadas:* This is a graph showing the quantity of stamps that were delivered by date.

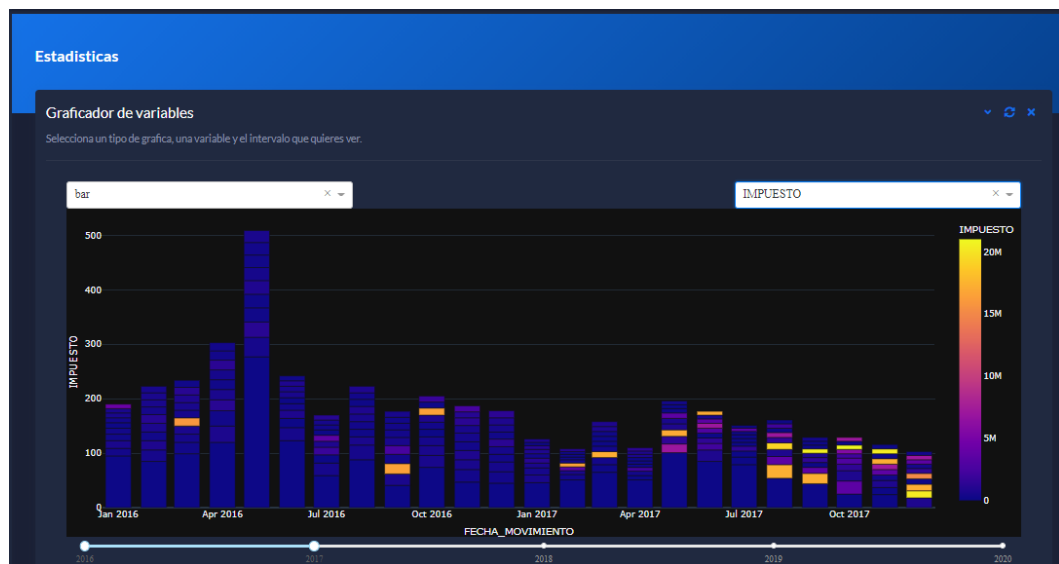


5. *Histórico de total impuestos recaudados:* On this graph we show the taxes that were collected on specific dates.

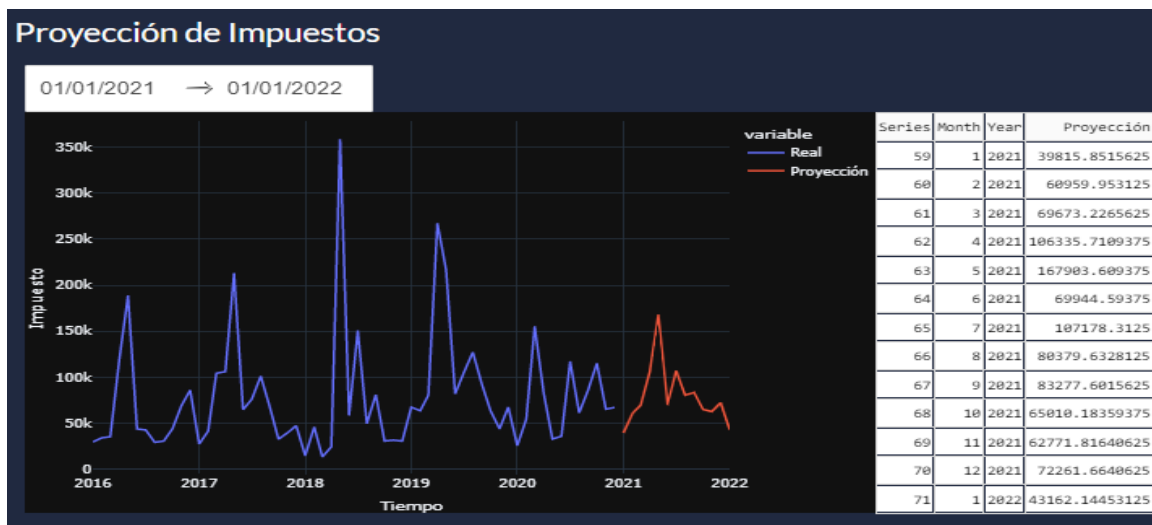
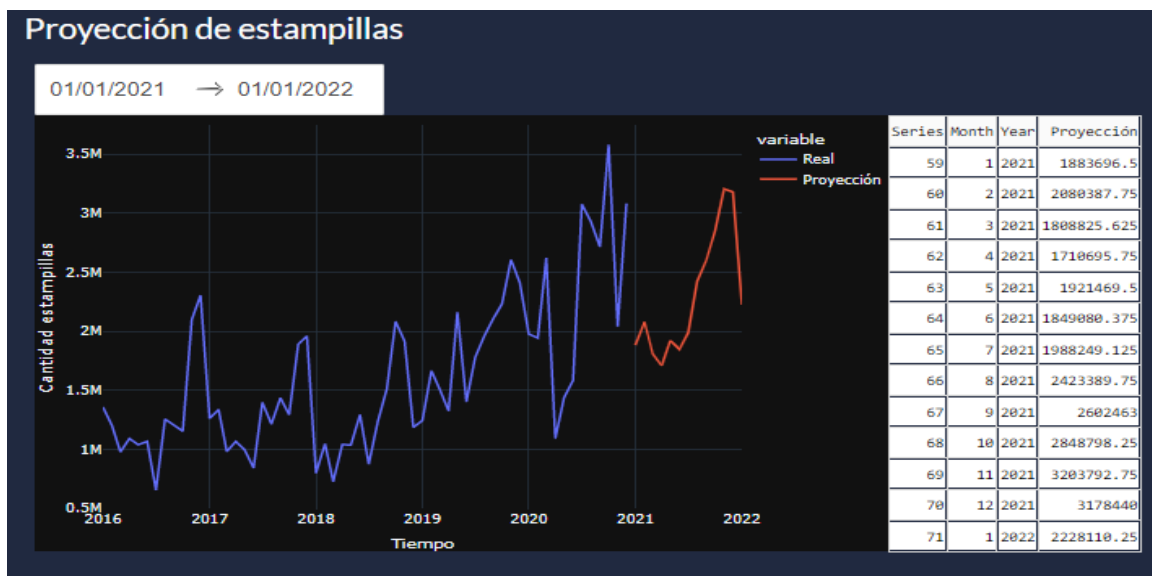


- **Estadísticas:** In this section The user can find a graph with some data statistics. This section allows you to change the variable of interest (Tipo movimiento, Nit, Empresa, etc) and also select different graph types (line, scatter, bar). It also allows you to change the year.

With this section the user can get a better sense of the data distribution for every field in the data and check how this changes in the time.



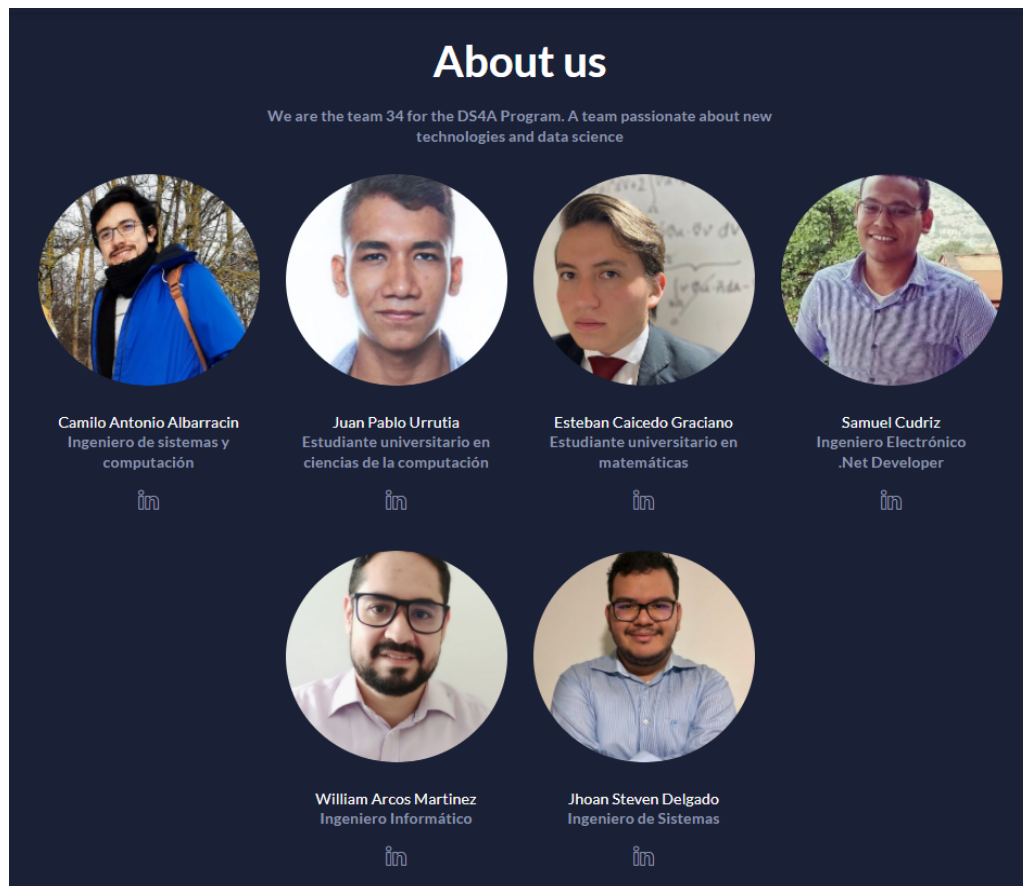
- **Proyección:** Here we can find a model that makes a prediction about the amount of taxes and amount of stamps to deliver over time. This graph can be configured to show the prediction up to 4 years.



- **Cargue:** This is the entry point for the data. Here the user can upload files for Importados, Nacionales and Estampillados. When the user presses the Cargar button, these files are processed and inserted into the postgres database. Through the reset button, the user can also reset the database if wrong data was uploaded.



- **Sobre Nosotros:** This page shows information about the team that developed the solution.



About us

We are the team 34 for the DS4A Program. A team passionate about new technologies and data science

Camilo Antonio Albarracín
Ingeniero de sistemas y computación

Juan Pablo Urrutia
Estudiante universitario en ciencias de la computación

Esteban Caicedo Graciano
Estudiante universitario en matemáticas

Samuel Cudriz
Ingeniero Electrónico
.Net Developer

William Arcos Martinez
Ingeniero Informático

Jhoan Steven Delgado
Ingeniero de Sistemas

6. CONCLUSIONS

With our proposed solution (liquorpredictool) we could help to generate some insights from data as well as forecasts for the prediction of stamps and tax revenue for Antioquia's department so they can take guided decisions to maximize the outcome, adding more value to their business.

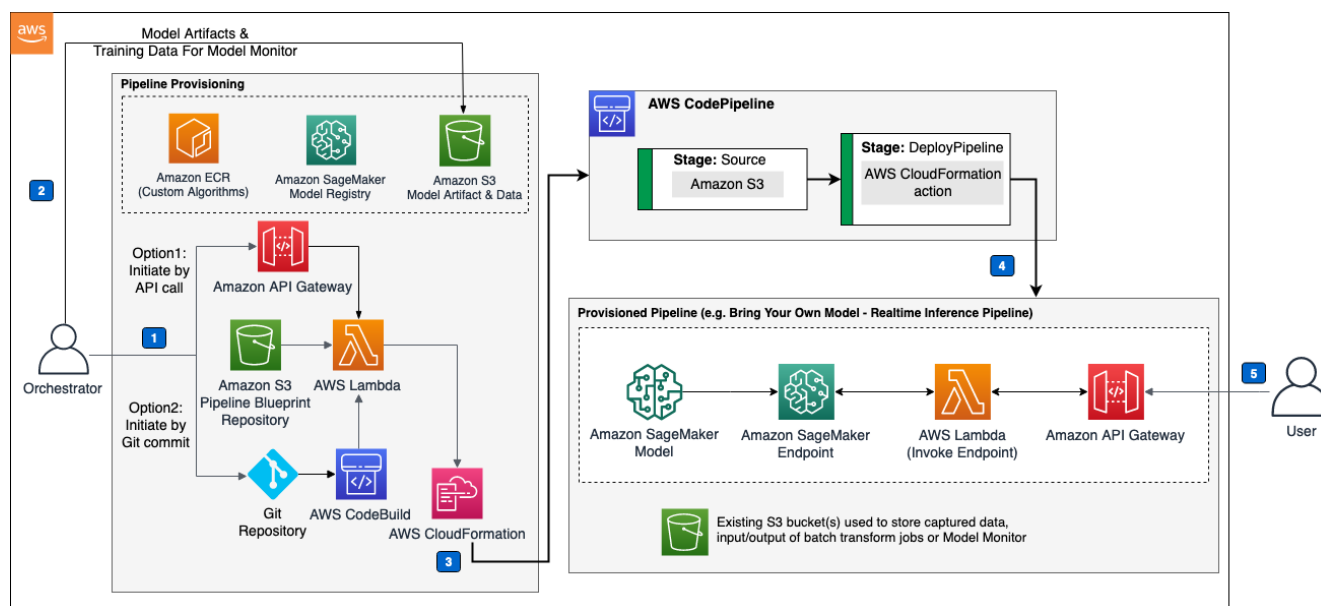
Our conclusions include the following:

- It is possible to train and make use of a regression model (In this case Ridge Regression) with the given data to forecast both the number of stamps and tax revenue for Antioquia's department.
- Pycaret can be used in a production environment simplifying the process of comparing and training several regression models to choose the best one.
- Our web app can manage new data points as time goes by for the visualization tools to get more insights.
- Our web app can be scalable to integrate MLOps as we discuss in the future work section

7. FUTURE WORK

With the current project we could give a solution to Antioquia's department problem concerning the forecast (model) of the tax income and the quantity of the stamps as well as data visualization dashboards and statistics for quick insights throughout the years. Since we live in a world that is constantly changing, the models have to be updated and retrained to not lose their ability to generalize the knowledge and make a good prediction (Forecasting) as years go by, additionally if several people are using the model and trying to make predictions at the same time, we can have the case of a bottleneck and lose the availability of the model, thus, as future work we propose an improvement in the life cycle of the machine learning model making an implementation of MLOps (Machine Learning Operations) so the model can be automatically retrained when it starts deviating from the ground truth.

Implementing MLOps into the system is not an easy task if we try to do it manually, but thankfully AWS has made a framework MLOps Framework that automates this process and help us scale the system architecture with the best practices for machine learning model productionization uploading our custom models, configuring the orchestration of the pipeline, and monitoring the pipeline's operations, making an easy integration with the current system all in one place.



<https://aws.amazon.com/solutions/implementations/aws-mlops-framework/>

We also propose exception handling when it comes to uploading new files to the RDS database through the Cargue menu to identify some bad data, as well as avoid duplicated data in the database. Finally, it'd be important to keep security in mind to avoid unauthorized people having access to the app and the data. We propose a login window or a restricted IP list so only authorized employees are able to access it within Antioquia's department.

8. REFERENCES

- Datasets provided by Antioquia's department
- Dash by Plotly. n.d. Dash Documentation & User Guide | Plotly. [online] Available at: [<https://dash.plotly.com/>](https://dash.plotly.com/).
- Django, 2021. Django documentation | Django. [online] Docs.djangoproject.com. Available at: [<https://docs.djangoproject.com/en/3.2/>](https://docs.djangoproject.com/en/3.2/).
- Bootstrap team, 2021. Bootstrap documentation. [online] Getbootstrap.com. Available at: [<https://getbootstrap.com/docs/5.1/getting-started/introduction/>](https://getbootstrap.com/docs/5.1/getting-started/introduction/).
- Ali, M., 2021. Time Series Forecasting with PyCaret Regression Module. [online] Medium. Available at: [e-237b703a0c63](https://towardsdatascience.com/time-series-forecasting-with-pycaret-regression-module-237b703a0c63).
- Ali, M., 2021. Pycaret main repository. [online] GitHub. Available at: <https://github.com/pycaret/pycaret>.
- Codewithharry.com. 2021. How to host Django Application using gunicorn & nginx in Production [online] Available at: <https://www.codewithharry.com/blogpost/django-deploy-nginx-gunicorn>.
- An Introduction to Statistical Learning - with Applications in R (Gareth James, et al 2013).