DSA8030: Individual Industry Based Project

ML/AI for Prediction of Physician and Laboratory Diagnostic Testing

Presented by :
Anjum Banu Ismail
MSc. Data Analytics
Student Id: 40313865

Academic Supervisor:
Dr. Laura Boyle
Queen's University Belfast

Company Supervisor:
Scott Morrison
Diaceutics PLC

Diaceutics
Better Testing, Better Treatment

QUEEN'S UNIVERSITY BELFAST

Table of Contents

1. Abstract

Using commercial data and historic Medicare data, this report aims to show my analysis of the Physicians and Laboratory patients data, and to predict the next volume of Medicare patients. The predictions are for each individual Lab and Physician ID.

2. Problem Statement

Health insurance in the United States is any program that aids in paying medical expenses, whether it comes from privately purchased insurance, government-funded welfare, or social services.

A federally administered insurance program, Medicare is a program into which Americans contribute throughout their working lives and enrol in after retiring or if they become disabled.

Commercial insurance must protect its business interests by excluding the most likely recipients of medical care by providing coverage to people under 65.

From 2017 to 2021, we have numbers of sufferers with malignant lung cancer that were referred by physicians (ordered testing) and laboratories (performed testing). Commercial data is always more up-to-date than Medicare data as seen in Table 1.

The purpose of the project is to create a generalized prediction model. The tasks involved in this project are as follows:

- Prepare the lab and physician data by downloading and pre-processing them.
- Decide on the appropriate imputation technique and fill in the missing quarter volumes.
- Training data that can predict volumes with less error is important.
- Select the right model for each lab and physician ID
- Create a simple web application that takes an ID as an input and predicts the right volumes for missing Medicare data

Year	Lab		Physician	
	Medicare	Commercial	Medicare	Commercial
2017	Q1	Q1	Q1	Q1
	Q2	Q2	Q2	Q2
	Q3	Q3	Q3	Q3
	Q4	Q4	Q4	Q4
2017	Q1	Q1	Q1	Q1
	Q2	Q2	Q2	Q2
	Q3	Q3	Q3	Q3
	Q4	Q4	Q4	Q4
2018	Q1	Q1	Q1	Q1
	Q2	Q2	Q2	Q2
	Q3	Q3	Q3	Q3
	Q4	Q4	Q4	Q4
2019	Q1	Q1	Q1	Q1
	Q2	Q2	Q2	Q2
	Q3	Q3	Q3	Q3
	Q4	Q4	Q4	Q4
2020	Q1	Q1	Q1	Q1
	Q2	Q2	Q2	Q2
	Q3	Q3	Q3	Q3
	Q4	Q4	Q4	Q4
2021	Q1	Q1	Q1	Q1
	Missing	Q2	Missing	Q2
	Missing	Q3	Missing	Q3

Table 1 Raw Dataset Structure

3. Data Exploration

Lab and Physician datasets have thousands of ID and each identifier has its own time series with either Medicare data or Commercial data or both.

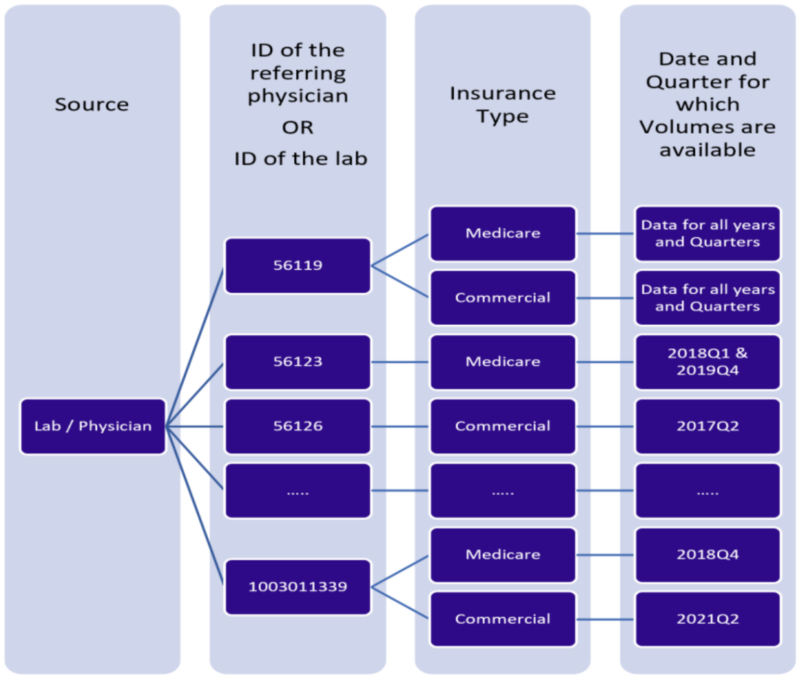


Fig 1 Quick look into the Data

Fig 1 illustrates that the data structure is not the same for every Lab and Physician ID. Below are some of the common possibilities.

- *Medicare and commercial data can be found on some IDs*
- *Some IDs may only contain Medicare information*
- *ID's with only commercial data may exist*
- *Volumes for some IDs will be available for all quarters from 2017 to 2021*
- *Volumes for some IDs may only be available for a specific period*
- *Commercial Data is current, unlike Medicare Data i.e.(2 Quarter Lag)*

A look at the dataset's challenges

- Insufficient training data
- There is no interdependence among time series
- There is not a single time series that is the same
- For most IDs, quarter volumes are missing
- No Trend or Seasonality observed
- No correlation between Medicare and Commercial Volumes
- During the COVID Pandemic, there was a lot of noise in the patient's volume.

4. Feature Engineering

4.1 Data Count

Once the data is downloaded for both Lab and Physician, the statistics dimensions and there feature names are established and the data is scanned to see if any of the capabilities is having NA.

Source	Rows	Features	Missing Values
Lab	18466	['id', 'type', 'year', 'quarter', 'volume']	id True type False year False quarter False volume False

Source	Rows	Features	Missing Values
Physician	18466	['id', 'type', 'year', 'quarter', 'volume']	id True type False year False quarter False volume False

Table 2 Dataset Information

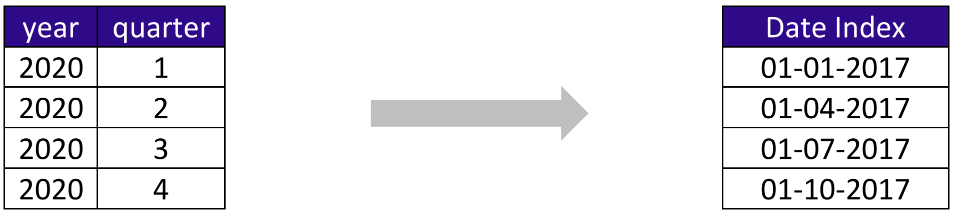
As in line with the observation seen in Table 2 each of the resources has missing entries for ID Feature

4.2 Dropping NA

Since the missing values are present only in id's that's the number one key for our evaluation and prediction, and hence the missing entries may be eliminated from the datasets. The percentage of missing entries are very small with **0.10%** for Lab and **0.014%** for Physician.

4.3 Translating the year and quarter to the Date format

Within the given dataset, indices include integers. The columns 'year' and 'quarter' is combined together to create time Date format with 1st date for every Quarter and this new Date format is indexed and hence we can easily make time series evaluation.



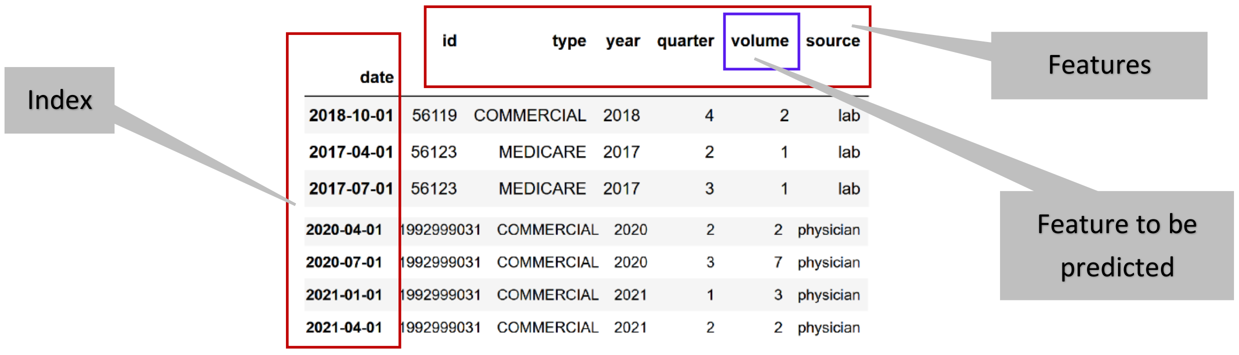
year	quarter
2020	1
2020	2
2020	3
2020	4

Date Index
01-01-2017
01-04-2017
01-07-2017
01-10-2017

Fig 2 Date format and Indexing

4.4 Combining Data

The given two separate laboratory and medical datasets are combined along to one Dataframe. A new column 'source' is created to represent the supply of the information from either Lab or Physician. Below is the final look on the Dataframe.



date	id	type	year	quarter	volume	source
2018-10-01	56119	COMMERCIAL	2018	4	2	lab
2017-04-01	56123	MEDICARE	2017	2	1	lab
2017-07-01	56123	MEDICARE	2017	3	1	lab
2020-04-01	1992999031	COMMERCIAL	2020	2	2	physician
2020-07-01	1992999031	COMMERCIAL	2020	3	7	physician
2021-01-01	1992999031	COMMERCIAL	2021	1	3	physician
2021-04-01	1992999031	COMMERCIAL	2021	2	2	physician

Fig 3 Final Data Frame Snippet

4.5 Time series Length and there Dates

Any Individual time series within Lab and Physician dataset has commercial dataset with 18 quarters of databases on the contrary Medicare dataset is forever lagged with 1 or 2 Quarters of data. There Begin (Up to now) and End (Up to now) for the given dataset is mentioned in beneath table.

Type	Length	Start Date	End Date
Medicare	17	01-01-2017	01-01-2021
Commercial	18	01-01-2017	01-04-2021

Table 3 Time Series Length and Dates

4.1 ID Count

With ID being one of the foremost critical feature to create numerous time series required for investigation. Underneath table conveys a check on the number of unique IDs found in Lab and Physician information set which are moreover assembled from “Medicare” and “Commercial”

Sourced	Type	Unique ID Count
Lab	Medicare	1550
Lab	Commercial	1234
Physician	Medicare	42022
Physician	Commercial	33291

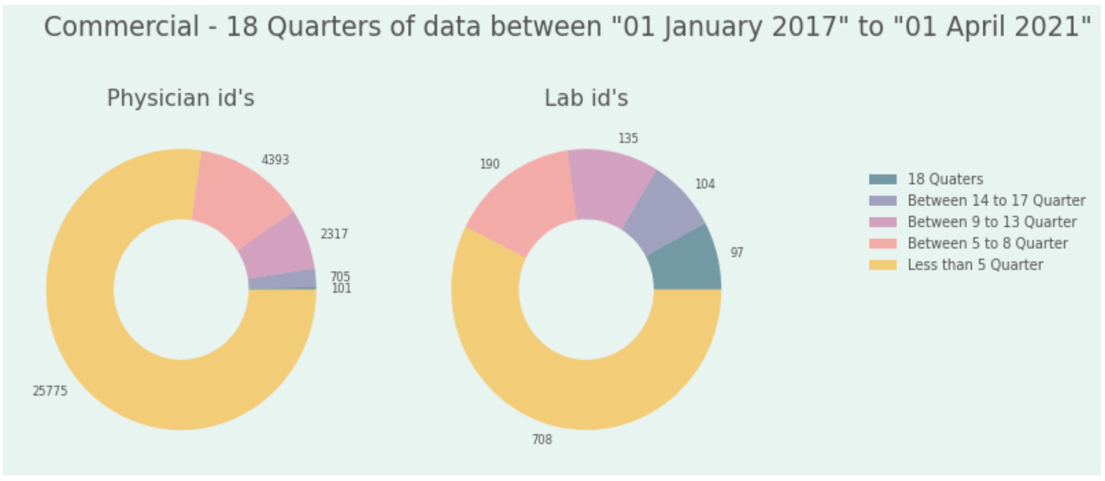
Table 4 Unique ID count

5. Data Visualization

5.1 Understanding ID’s with missing volumes

For the given Dataset the ID's are binned together depending on the volumes accessible for each quarter. The underneath plot appears us that there are exceptionally few ID's where the volumes are detailed for all the quarters for a given period.

The other set of IDs with up to 50% of information lost can be certainly be treated by applying different imputation techniques. Here the imputation will advantage us instead of overlooking them.



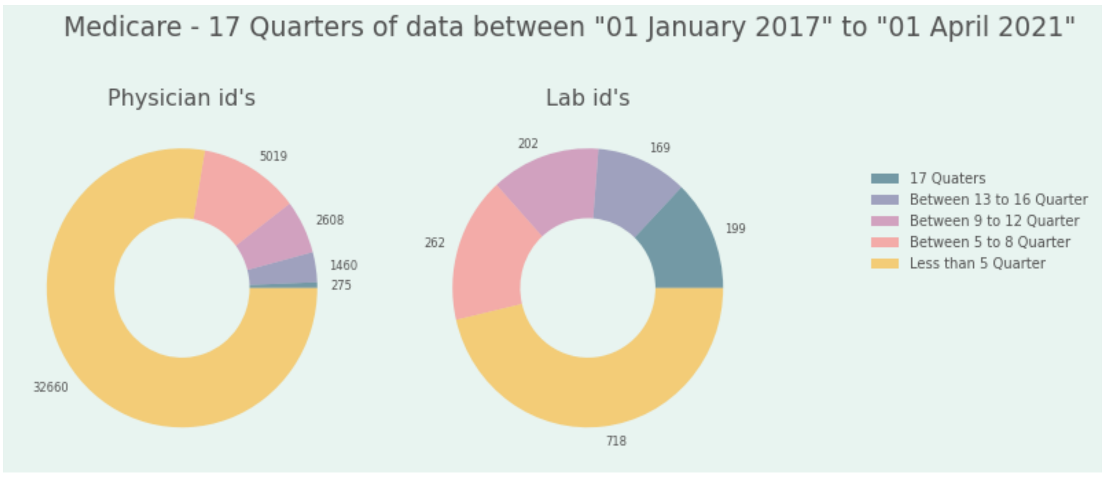


Fig 4 Unique ID tally based on lost volumes

We see the Amber colour is generally ruling in all 4 plots in Fig 4, appearing us that larger part of IDs has exceptionally less data. If it is imperative to examine these IDs hence we have to be request business to provide new data set.

In my examination, I have chosen the 50% as a threshold for lost values, anything over 50% lost values will basically be disregarded and not be utilized for forecast. Statistical guidance articles have stated that bias is likely in analyses with more than 50% missingness and that if more than 50% then results should only be considered as hypothesis generating.

5.2 Reports and Volumes

The first plot Fig 5(a) appears us the tally of report by Lab and Doctors, whereas the second plot Fig 5 (b) Count Plot gives a number on the overall volumes of patients.

While comparing both the plots, it is shocking to see that indeed in spite of the fact that the report from Lab are way less compare to Doctor, but the number of threatening cancer understanding tried in Lab is surpassing the patient count alluded by doctor.

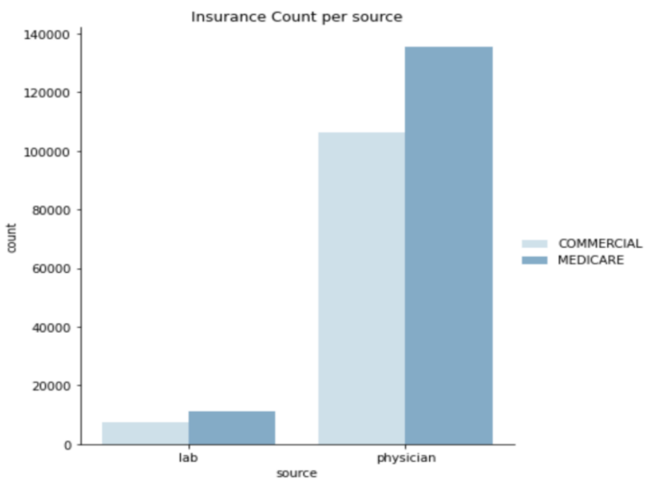


Fig 5 (a) Count Plot grouped by source

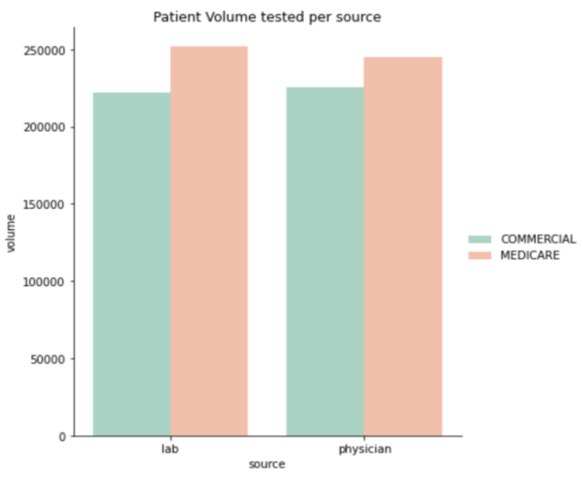


Fig 5 (b) Volume Plot grouped by source

To confirm the volumes over the year, the information is grouped within source and there types and from the below plot we see that the volumes were less in year 2017 compare to other year in both Medicare and commercial Protections types. Year 2018, 2019 and 2020 have inexact same number of volumes and obviously current year 2021 being is the least because of only 2 Quarter data available till date.

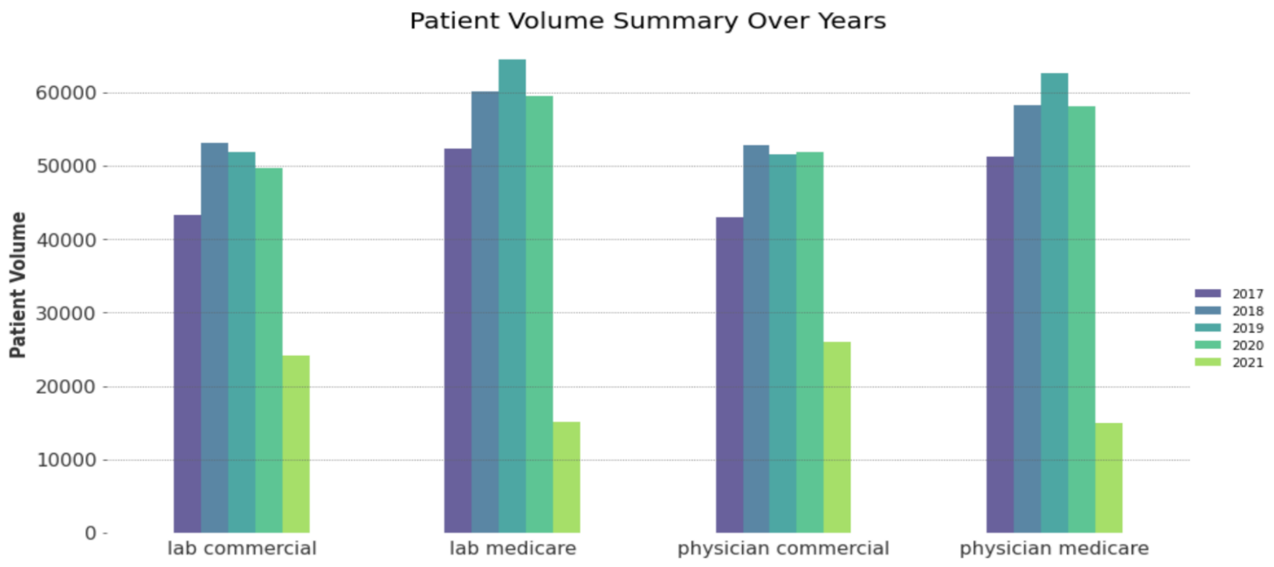


Fig 6 Patient volume over the years 2017 - 2021

From both the plots Fig 5(a)(b) and Fig 6 it is clear that that Commercial volumes were continuously less compare to Medicare

5.3 Patient Volume count

A below plot has years, usually divided into 4 quarters on the x-axis, which is the vertical axis, and volume of patient on the y-axis, which is the horizontal access.

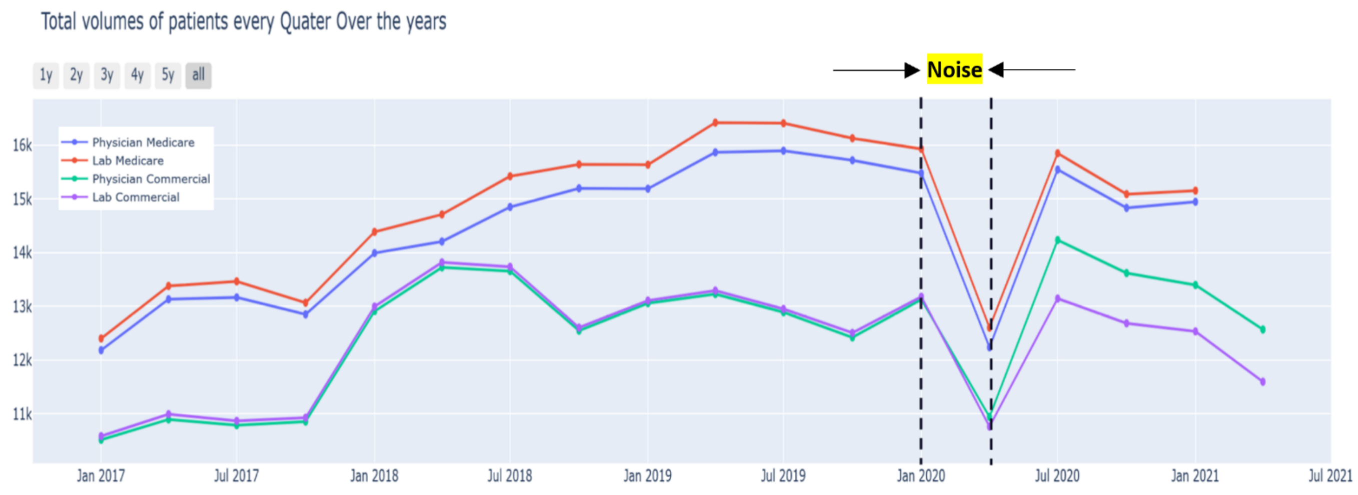


Fig 7 Total Patient Volumes and Noise observed

Some interesting observation seen in the below plot are Medicare and Commercial patient volume are correlated to each other over the years. There is no any seasonality observed in a year. Interestingly there is a upward trend seen in Medicare volume but the same is not with Commercial volume.

The COVID-19 pandemic seems to be the most important phenomenon observed from March 2020 in virtually all countries of the world which caused a historically large and rapid declines in volumes across all categories in 2nd quarter of 2020 resulting in a noisy data.

6. Imputation for Missing Data

6.1 Imputation Framework

There are 2 Imputation stages implemented in this project. At any point a given time series will have to experience at least one imputation check. Missing value Imputation is Machine learning-based procedure whereas Noise Imputation is simple mean substitution technique.

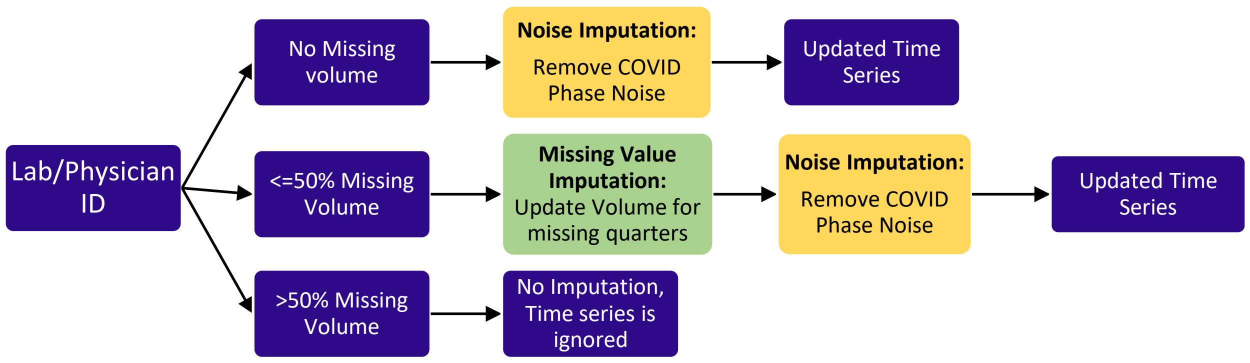

Lab/Physician ID
No Missing volume
Noise Imputation: Remove COVID Phase Noise
Updated Time Series
<=50% Missing Volume
Missing Value Imputation: Update Volume for missing quarters
Noise Imputation: Remove COVID Phase Noise
Updated Time Series
>50% Missing Volume
No Imputation, Time series is ignored

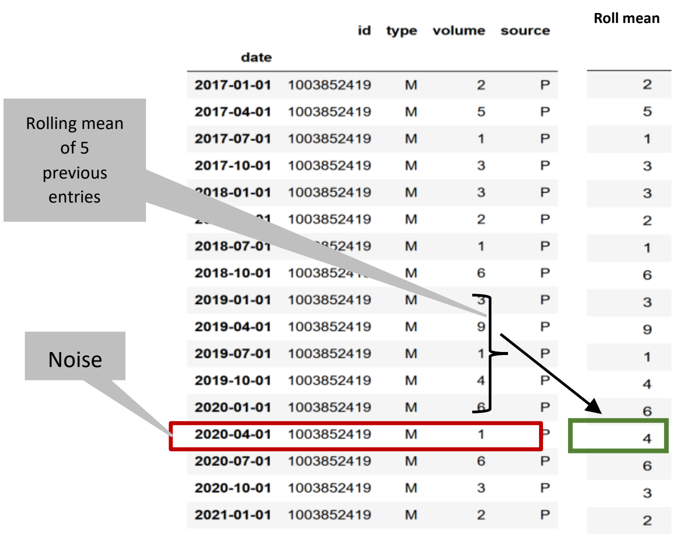
Fig 8 Flow chart representing Imputation stages

6.2 Noise Imputation

This is a simple way to impute the missing data. It replaces the Noise values (2nd Quarter of 2020) by the Rolling mean with window length of 5 of the same feature.

Once the Rolling mean is calculated the value is replaced in the Volume column and time series is ready for further ML operations.

This Imputation logic is applied only for time series without any missing data or applied to time series whose missing values are already imputed.


Rolling mean of 5 previous entries
Noise

	id	type	volume	source	Roll mean
date					
2017-01-01	1003852419	M	2	P	2
2017-04-01	1003852419	M	5	P	5
2017-07-01	1003852419	M	1	P	1
2017-10-01	1003852419	M	3	P	3
2018-01-01	1003852419	M	3	P	3
2018-04-01	1003852419	M	2	P	2
2018-07-01	1003852419	M	1	P	1
2018-10-01	1003852419	M	6	P	6
2019-01-01	1003852419	M	3	P	3
2019-04-01	1003852419	M	9	P	9
2019-07-01	1003852419	M	1	P	1
2019-10-01	1003852419	M	4	P	4
2020-01-01	1003852419	M	6	P	6
2020-04-01	1003852419	M	1	P	4
2020-07-01	1003852419	M	6	P	6
2020-10-01	1003852419	M	3	P	3
2021-01-01	1003852419	M	2	P	2

Fig 9 Noise Data Imputation with rolling mean

6.3 Missing Value Imputation

For the time series given we assume the probability of being missing is the same for all cases and data is **Missing completely at random (MCAR)**. The missing values are imputed using combination of different strategy and **Iterative Imputer** strategy and the best resulting in low "Mean Square Error" will be used to impute missing values for the remaining time series.

Techniques used are:

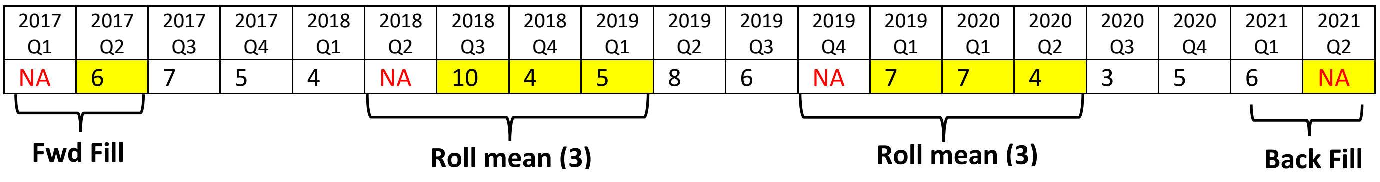
a. Simple Mean:
Simply calculate the mean of the observed values for that variable for all individuals who are non-missing. It has the advantage of keeping the same mean and the same sample size, but many, many disadvantages.

b. DecisionTreeRegressor
This is a non-linear regression Imputation technique. This technique use decision trees and forests to identify horizontal segments of a data set where the records belonging to a segment have higher similarity and attribute correlations. Using the similarity and correlations, missing values are then imputed.

c. ExtraTreesRegressor
To Mimic the behaviour of **missForest** we have decided to choose "ExtraTreesRegressor". This is chosen for its increased speed.

d. KNeighborsRegressor (n_neighbors=6)
This technique learns from samples with missing values by using a distance metric that accounts for missing values, rather than imputing them.

e. RollMean + Fwd Fill + Back Fill
Here a combination of 3 simple techniques is used to fill the missing value. Using roll mean with window size four takes care of the data in between, while forward and backward fill takes care of the data at extremes



2017 Q1	2017 Q2	2017 Q3	2017 Q4	2018 Q1	2018 Q2	2018 Q3	2018 Q4	2019 Q1	2019 Q2	2019 Q3	2019 Q4	2019 Q1	2020 Q1	2020 Q2	2020 Q3	2020 Q4	2021 Q1	2021 Q2
NA	6	7	5	4	NA	10	4	5	8	6	NA	7	7	4	3	5	6	NA

The goal is to compare different estimators to see which one is best. This is achieved by selecting 30 different random Time Series with no missing data and adding NA's randomly and applying these imputation techniques and validating the error between the actual and imputed values.

An overall mean square error MSE is calculate for 30 random Train and test time series and the metrics are mentioned as below. Even though the other Imputation gave less error for training data, but for the test data **RollMean + Fwd Fill + Back Fill** gave reduced error for all categories of data.

	Training				Testing			
	Lab Commercial	Lab Medicare	Physician Commercial	Physician Medicare	Lab Commercial	Lab Medicare	Physician Commercial	Physician Medicare
DecisionTreeRegressor	30.811193	4.81664	2.339278	1.89943	517.337584	103.53	3.764601	3.41823
ExtraTreesRegressor	22.455965	3.97097	2.063349	1.51593	509.853697	103.42	3.587014	3.03832
KNeighborsRegressor	28.390172	3.69579	1.880701	1.52109	519.169326	103.513	3.825596	2.81523
Simple Mean	33.088294	4.47718	2.366041	1.74221	510.911718	103.399	3.224616	2.73611
RollMean+FwdFill+BckFill	32.205187	4.23246	2.189622	1.5088	33.691987	19.9437	1.706198	1.52303

Table 5 Imputation techniques and there MSE

A basic plot is made beneath to show the actual and Imputed values, and it is clear that RollMean + Fwd Fill + Back Fill is practically same as actual information. Remembering the MSE error values from above table and forecast esteems from underneath plot we settle to utilize RollMean + Fwd Fill + Back Fill as our final Imputation method to fill in missing information.

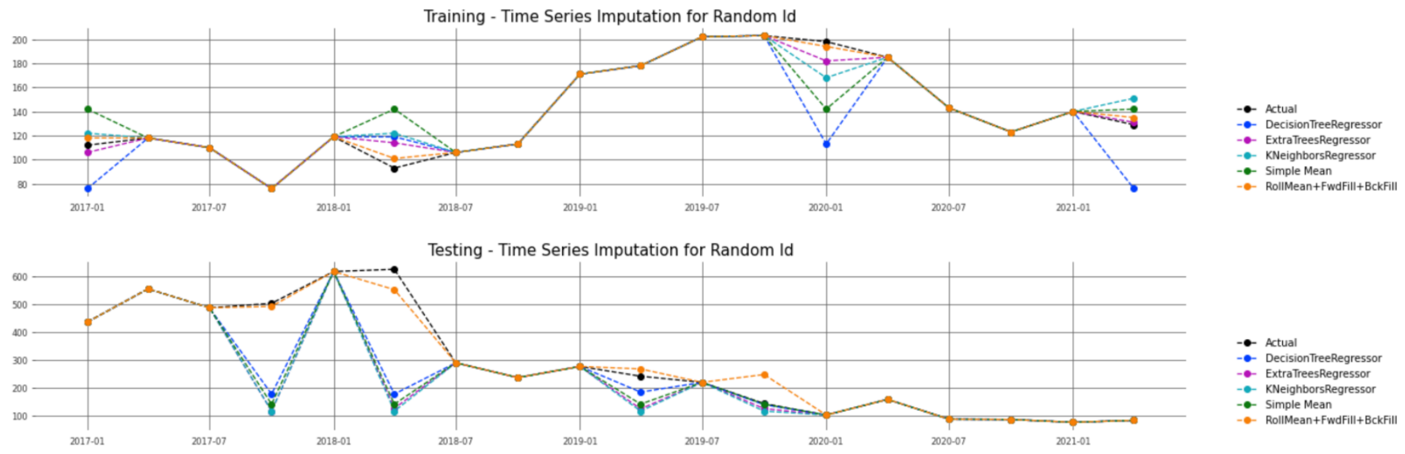


Fig 9 Actual vs Imputed values for a time series

6.4 Patient Volume count after Imputation

The drop seen in patient volume in Fig 7 during COVID pandemic stage (2020 second Quarter) should now be similar.

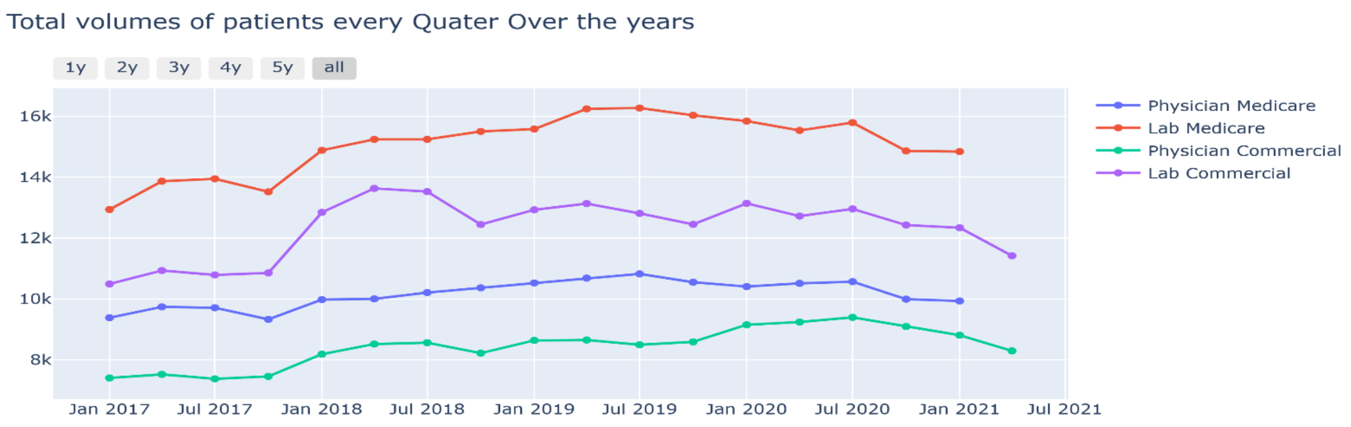


Fig 10 Total Patient Volumes without Noise

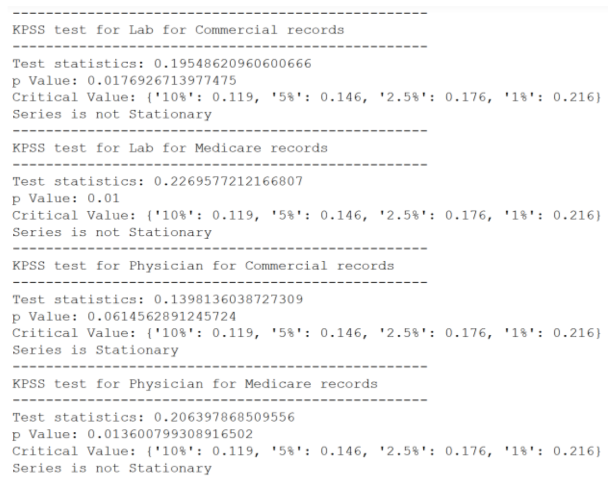
7. Stationary, Lag and Decomposition of Time series

7.1 Detecting Stationary in Time Series

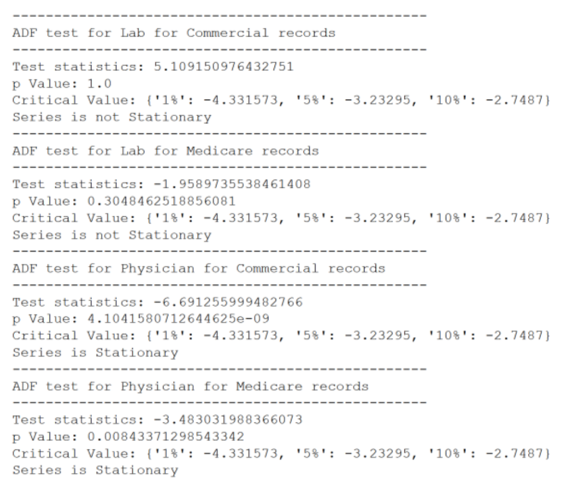
The most important point to make is that most of our useful time series theory applies only to stationary variables and not to non-stationary variables. Stationarity means that the statistical properties of a time series (or rather the process generating it) do not change over time. Stationarity is important because many useful analytical tools and statistical tests and models rely on it. The most basic methods for stationarity detection rely on plotting the data, or functions of it, and determining visually whether they present some known property of stationary (or non-stationary) data.

But some time it is really hard to simply visualise the plot and make a decision on time series stationary check, hence here we have used 2 different techniques to accommodate the complex data, **Augmented Dickey-Fuller (ADF) (using AR of any order *p* and supporting modelling of time trends)** and **KPSS test (based on linear regression, breaking up the series into three parts: a deterministic trend, a random walk (*rt*), and a stationary error)**

Both the test were applied for 4 categories of information and both of them came about showing series is not stationary.



```
--------------------------------------------------
KPSS test for Lab for Commercial records
--------------------------------------------------
Test statistics: 0.19548620960600666
p Value: 0.0176926713977475
Critical Value: {'10%': 0.119, '5%': 0.146, '2.5%': 0.176, '1%': 0.216}
Series is not Stationary
--------------------------------------------------
KPSS test for Lab for Medicare records
--------------------------------------------------
Test statistics: 0.2269577212166807
p Value: 0.01
Critical Value: {'10%': 0.119, '5%': 0.146, '2.5%': 0.176, '1%': 0.216}
Series is not Stationary
--------------------------------------------------
KPSS test for Physician for Commercial records
--------------------------------------------------
Test statistics: 0.1398136038727309
p Value: 0.0614562891245724
Critical Value: {'10%': 0.119, '5%': 0.146, '2.5%': 0.176, '1%': 0.216}
Series is Stationary
--------------------------------------------------
KPSS test for Physician for Medicare records
--------------------------------------------------
Test statistics: 0.206397868509556
p Value: 0.013600799308916502
Critical Value: {'10%': 0.119, '5%': 0.146, '2.5%': 0.176, '1%': 0.216}
Series is not Stationary
```



```
--------------------------------------------------
ADF test for Lab for Commercial records
--------------------------------------------------
Test statistics: 5.109150976432751
p Value: 1.0
Critical Value: {'1%': -4.331573, '5%': -3.23295, '10%': -2.7487}
Series is not Stationary
--------------------------------------------------
ADF test for Lab for Medicare records
--------------------------------------------------
Test statistics: -1.9589735538461408
p Value: 0.3048462518856081
Critical Value: {'1%': -4.331573, '5%': -3.23295, '10%': -2.7487}
Series is not Stationary
--------------------------------------------------
ADF test for Physician for Commercial records
--------------------------------------------------
Test statistics: -6.691255999482766
p Value: 4.104158071264 4625e-09
Critical Value: {'1%': -4.331573, '5%': -3.23295, '10%': -2.7487}
Series is Stationary
--------------------------------------------------
ADF test for Physician for Medicare records
--------------------------------------------------
Test statistics: -3.483031988366073
p Value: 0.00843371298543342
Critical Value: {'1%': -4.331573, '5%': -3.23295, '10%': -2.7487}
Series is Stationary
```

Fig 11 Stationary Check

7.2 Data Decomposition

Time series decomposition involves thinking of a series as a combination of level, trend, seasonality, and noise components. Decomposition provides a useful abstract model for thinking about time series generally and for better understanding problems during time series analysis and forecasting.

The components are defined as follows:

- **Level**: The average value in the series.
- **Trend**: The increasing or decreasing value in the series.
- **Seasonality**: The repeating short-term cycle in the series.
- **Residual**: The random variation in the series.

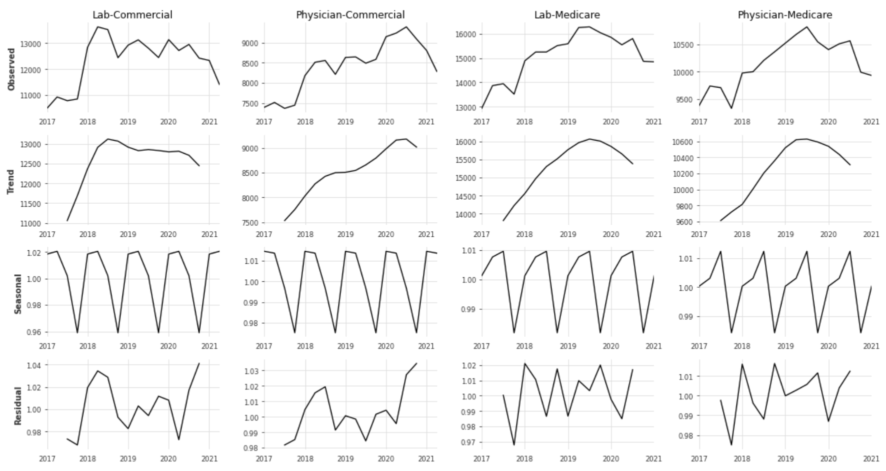


Fig 11 Time Series Decomposition

We have created a time series comprised for all 4 categories and decompose it as an additive model. In all 4 categories we see there is a upward trend. In this time series the components are available in uneven amounts hence can cause the model to predict wrong values while using ARIMA models. To improve the quality of the prediction we can perform detrending, deseasonalizing and perform some smoothing methods on it.

7.3 Lag and Auto-Correlation

Lags and Auto correlation are very useful in understanding seasonality and form the basis for autoregressive forecast models such as AR, ARMA, ARIMA, SARIMA. A simple lag operation simply shifts (offsets) a time series such that the “lagged” values are aligned with the actual time series. The **Fig 12** below illustrates the lag operation for lag 1. The lag plot shows exhibits a linear pattern for Medicare data. This shows that the data are strongly non-random and further suggests that an autoregressive model might be appropriate.

Lag plots can provide answers to the following questions:

- Are the data random?
- Is there serial correlation in the data?
- What is a suitable model for the data?
- Are there outliers in the data?

Fig 13 is an Auto correlation plot to determine the proper lag to use with the differencing method of eliminating seasonality. The flat lines in the plot correspond to 95% and 99% confidence bands. It is obvious that the maximum corelation of about 0.75 is achieved with Lag 1 only for Physician Medicare information alone, for other categories the correlation dip under 0.70. As the lags increase the Autocorrelation is dramatically diminishing.

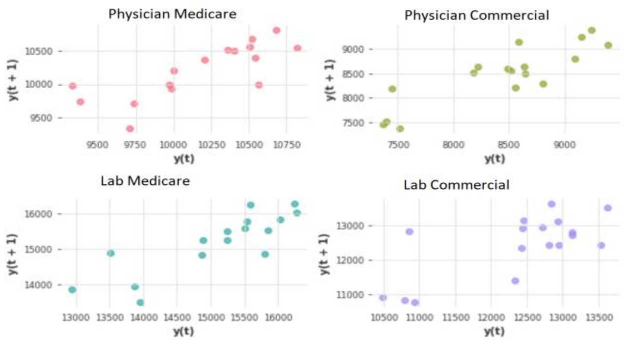


Fig 12 Actual vs Lag 1

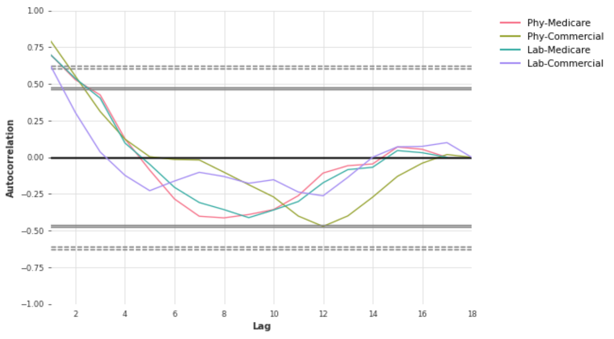


Fig 13 Lags and there Correlation

From section 7.1 we have as of now presumed that the time series are non-stationary, these can be changed or fixed by eliminating the patterns, or detrending. We have used 3 basic procedures (fig 14) 1 log difference, Rolling mean and Rolling Standard deviation. These techniques were applied, KPSS and ADF test were again conducted and the outcomes came out to show that the time series in now stationary

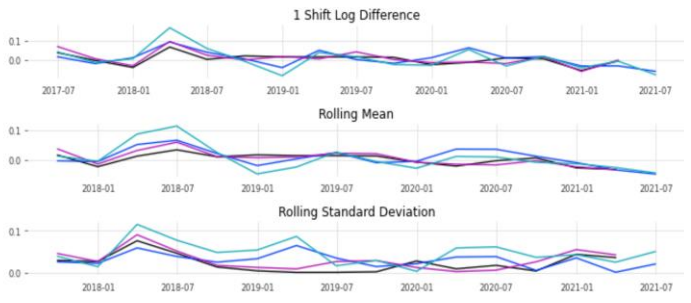


Fig 13 Stationary Time series

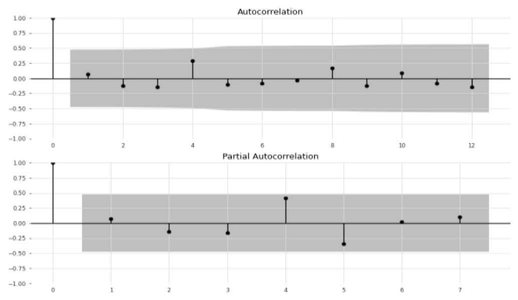


Fig 14 ACF and PACF plots

From **Fig 14** ACF and PACF plots allow you to determine the AR and MA components of an ARIMA model. Both the Seasonal and the non-Seasonal AR and MA components can be determined from the ACF and PACF plots. Looking at the plot we don’t see any significant lag or geometric decay.

8. Machine Learning Algorithms and Techniques

8.1 Train-Test Split and Metrics

After all taking a subset of valid IDs which are having over half of the patient volumes and applying various Imputation techniques, at last we have an aggregate of 6031 ID's for which ML Models ought to be applied for.

There are not many ID's which has both Medicare and Commercials info and some are having just Medicare. The Data having both Medicare and Commercial should be predicted based on its past volumes and also past and future dependent feature Commercial volumes. whereas the time series with only Medicare data are subjected to depend on past Medicare volumes.

	Count	Prediction required?
Lab ID's Medicare + Commercial	213	Yes
Lab ID's Medicare	357	Yes
Lab ID's Commercial	123	No
Physician ID's Medicare + Commercial	2128	Yes
Physician ID's Medicare	2215	Yes
Physician ID's Commercial	995	No
Total	6031	

Table 6 ID counts

Individual ID time series will be split chronologically based on years and Quarters, the first 70% of data will be used for Training and the rest 30% will be used for Testing and Validating the model

ID's with both Commercial and Medicare volumes

Date	ID	Medicare volumes	Commercial volumes
01/01/2017	56227	97	4
01/04/2017	56227	151	5
01/07/2017	56227	129	9
01/10/2017	56227	137	10
01/01/2018	56227	144	47
01/04/2018	56227	131	39
01/07/2018	56227	128	59
01/10/2018	56227	152	56
01/01/2019	56227	127	61
01/04/2019	56227	103	55
01/07/2019	56227	109	56
01/10/2019	56227	110	56
01/01/2020	56227	87	43
01/04/2020	56227	92	49
01/07/2020	56227	81	63
01/10/2020	56227	53	53
01/01/2021	56227	61	53
01/04/2021	56227	-	54

ID's with only Medicare volumes

Date	ID	Medicare volumes
01/01/2017	56209	4
01/04/2017	56209	4
01/07/2017	56209	1
01/10/2017	56209	3
01/01/2018	56209	11
01/04/2018	56209	14
01/07/2018	56209	9
01/10/2018	56209	12
01/01/2019	56209	12
01/04/2019	56209	15
01/07/2019	56209	18
01/10/2019	56209	13
01/01/2020	56209	9
01/04/2020	56209	12
01/07/2020	56209	8
01/10/2020	56209	6
01/01/2021	56209	7
01/04/2021	56209	-

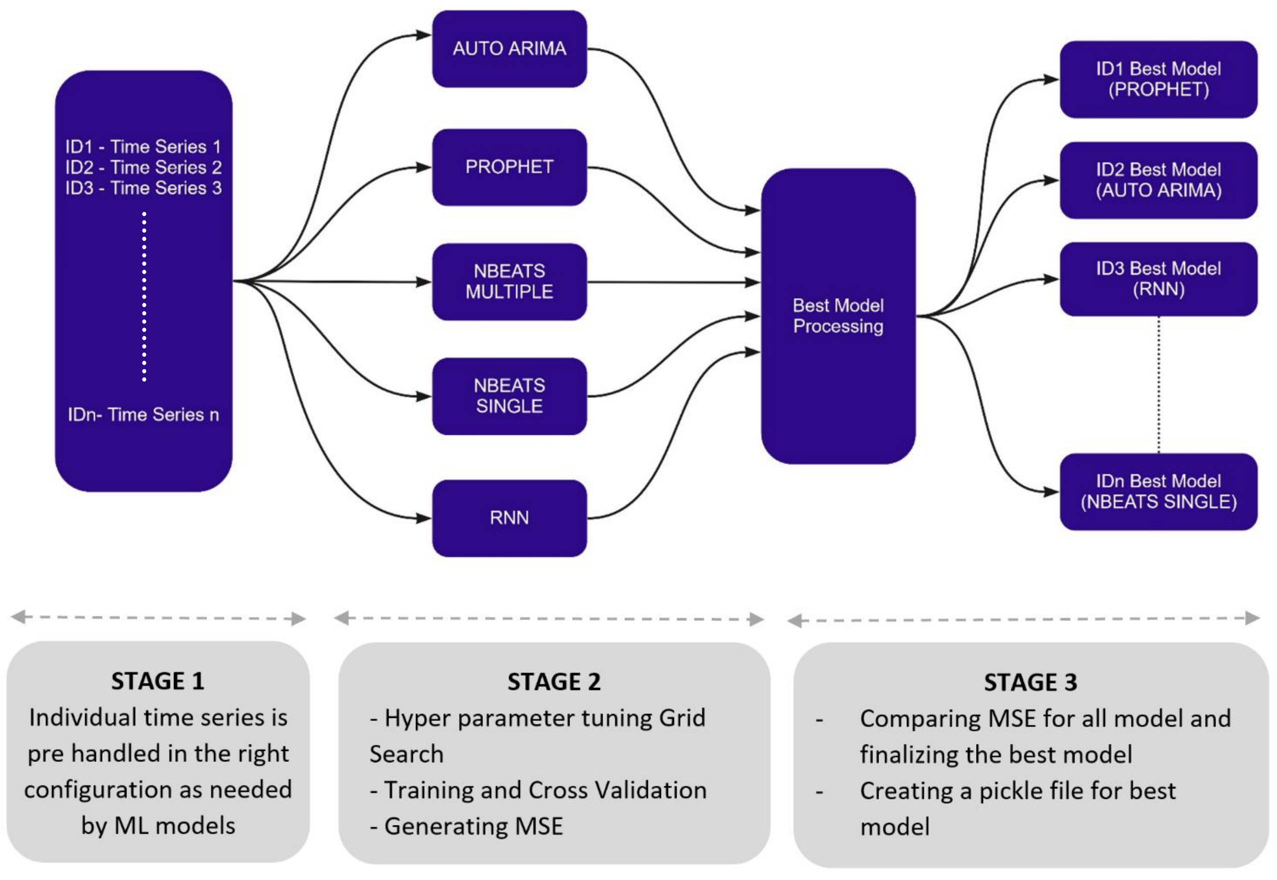
🟧	Training
🟩	Test / Validation
🟦	Prediction

Hitting at the right machine learning algorithm is the ideal approach to achieve higher accuracy. But, it is easier said than done. Some algorithms are better suited to a particular type of Time series than others. Hence, we should apply all relevant models and check the performance. The Metric used to evaluate the best model is “MSE” (Mean Square error).

Enhancing a model performance can be challenging at times, especially when we have multiple time series and every time series is unique, with some having other dependent feature added as an benefit for forecast, where as others became relying upon their own historic data. To get the best accuracy for each time series forecast we have implemented some ways like “**Multiple Algorithm**” and “**Algorithm Tuning**” approaches.

8.2 ML Architecture

The entire ML process is broken down into 3 steps. The IDs are separated in stage 1 and fed to the machine learning model in their required format. In stage 2 we follow the Hyper Tuning process within every ML model and choose the best hyperparameters to train on the Training Time Series and cross-validate them with the Test Time Series, capturing the best MSE. In stage 3, each MSE generated by the five ML models for an ID is compared, and the best one is chosen to be the final model, which is saved for future forecasting of Medicare volumes.



STAGE 1
Individual time series is pre handled in the right configuration as needed by ML models

STAGE 2
- Hyper parameter tuning Grid Search
- Training and Cross Validation
- Generating MSE

STAGE 3
- Comparing MSE for all model and finalizing the best model
- Creating a pickle file for best model

Fig 15 ML Flowchart

8.3 AUTO-ARIMA

ARIMA is an acronym that stands for Autoregressive Integrated Moving Average. Autoregression is a time series model that uses observations from previous time steps as input to the regression equation to predict the value at the next time step. A statistical library “**Pmdarima**” is used to here.

Some of the advantages of using this library are as below:

- A collection of statistical tests of stationarity and seasonality
- Time series utilities, such as differencing and inverse differencing
- Exogenous features can be used, which in our scenario uses commercial volumes if there are any.
- Seasonal time series decompositions
- Fast Processing time

Unlike ARIMA, AUTO ARIMA makes the task really simple by determining the values of Auto regression and moving average parameters. It simply bypasses the determine the values of p,q,d,P,Q. Auto ARIMA takes into account the AIC and BIC values generated to determine the best combination of parameters.

```
(y=train_med,X=train_com,
start_p=1,max_p=4,
start_q=1,max_q=4,
start_P=0,max_P=1,
start_Q=0,max_Q=1,
d=None,stationary=False,m=4,seasonal=False,
n_jobs=-1,trace=False,method = method,test='kpss',
stepwise=True,with_intercept=False,suppress_warnings=True,
error_action='ignore')
```

Fig 16 AUTO ARIMA Hyper tuning parameters

AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion) values are estimators to compare models. The lower these values, the better is the model.

Auto Arima does not require any additional grid search, the optimal parameter are chosen based on the values while fitting the data. Through the "**KPSS**" method, it also identifies non-stationary time series and converts them to stationary before computing the right parameters.

8.4 PROPHET

Facebook Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

The patterns of the each time series are complicated and change dynamically over time, but Prophet follows such changes only with the trend changing. Here we are using main component like changepoint and seasonality’s to hyper tune for every time series and get the right optimal model. Below are the different values provided to get the right parameter.

```
{'changepoint_prior_scale': [0.001, 0.01, 0.1,0.5],
 'seasonality_prior_scale': [0.01, 0.1, 1.0],
 'seasonality_mode':('multiplicative','additive')}
```

Fig 15 Prophet Hyper tuning parameters

The time series has data only for every quarter, hence the seasonality for yearly and weekly are set to False. For IDs having Commercial data there is an model created with additional regressor, The Regressor value must be known in the past and in the future, this is how it helps Prophet to adjust the forecast. **Additional regressors** feature **(Commercial Volumes)** is very important for accurate forecast calculation in Prophet. It helps to tune how the forecast is constructed and make prediction process more transparent.

8.5 NBEATS – Multiple and Single Model

Darts is a Python library for easy manipulation and forecasting of time series. It contains a variety of models, from classics such as ARIMA to deep neural networks. The models can all be used in the same way, using fit() and predict() functions, similar to scikit-learn. The library also makes it easy to back test models, and combine the predictions of several models and external regressors. Darts supports both univariate and multivariate time series and models. The neural networks can be trained on multiple time series, and some of the models offer probabilistic forecasts.

Some conclusions to derived are:

- Libraries like darts can provide us with a new way to work over time-series, allowing for flexibility and efficiency.
- As evident, we can load, process, and even train multiple datasets using a single library and model.
- With all the charm, one can successfully throne it as a scikit-learn for time series data.

N-BEATS (Neural basis expansion analysis for interpretable time series forecasting) is used for solving the univariate times series point forecasting problem using deep learning. It is a deep neural architecture based on backward and forward residual links and a very deep stack of fully-connected layers. Here the proposed architecture is augmented to provide outputs that are interpretable without considerable loss in accuracy. It uses a Past covariates models architecture as below.

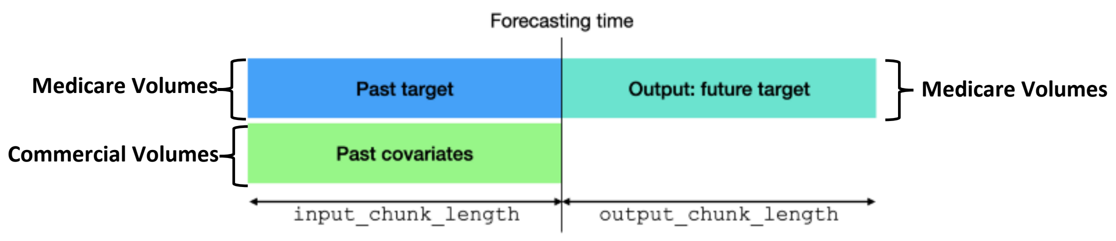


Fig 16 Depiction of the inputs/outputs for "Past Covariates models" at prediction time

The fit() and predict() methods of these models accept only a past_covariates argument (specifying one or a sequence of Time Series). These models will look only at past values of the covariate series when making a prediction. Grid search is used to identify the best "**input_chunk_length**", "**output_chunk_length**" and "**n_epochs**".

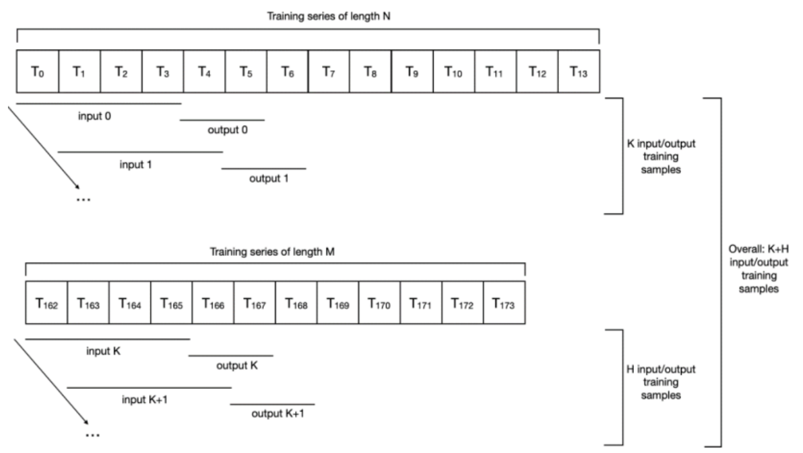

Training series of length N
T0 T1 T2 T3 T4 T5 T6 T7 T8 T9 T10 T11 T12 T13
input 0
output 0
input 1
output 1
K input/output training samples
Overall: K+H input/output training samples
Training series of length M
T162 T163 T164 T165 T166 T167 T168 T169 T170 T171 T172 T173
input K
output K
input K+1
output K+1
H input/output training samples

Fig 17 The slicing of two target time series to produce some input/output training samples, in the case of a Sequential Dataset

Two Nbeats models are created, where the multiple Nbeats model is fed with individual ID data together with covariates and targets. The model is trained separately for each individual ID. Alternatively, Nbeats single takes a list of target time series and covariant time series and feeds it to the model. It is not necessary that the series used for training be the same length (in fact, they do not even have to be the same frequency) as shown in Figure 17.

8.6 RNN

Darts library also includes RNN Model. RNNModel is fully recurrent in the sense that, at prediction time, an output is computed using these inputs:

- The previous target value, which will be set to the last known target value for the first prediction, and for all other predictions it will be set to the previous prediction
- The previous hidden state
- The current covariates (if the model was trained with covariates)

This model is suited for forecasting problems where the target series is highly dependent on covariates that are known in advance.

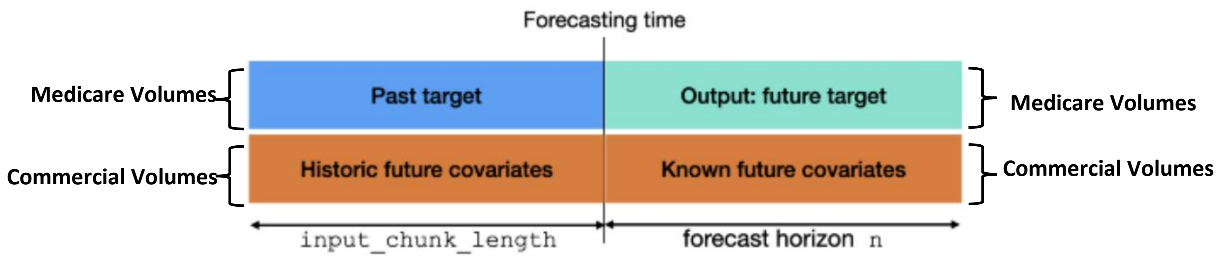
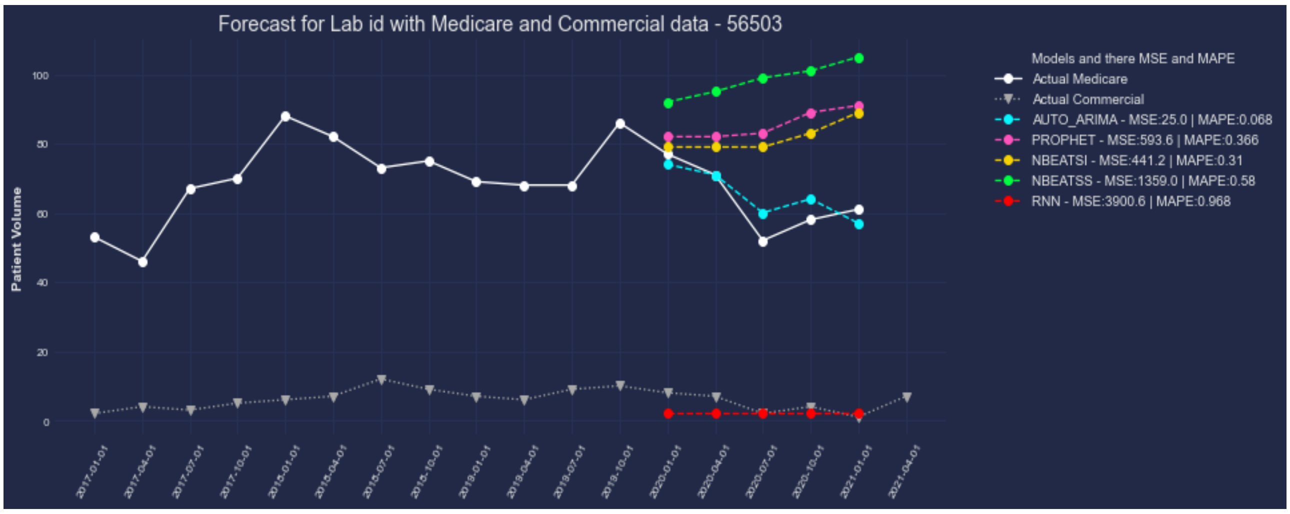


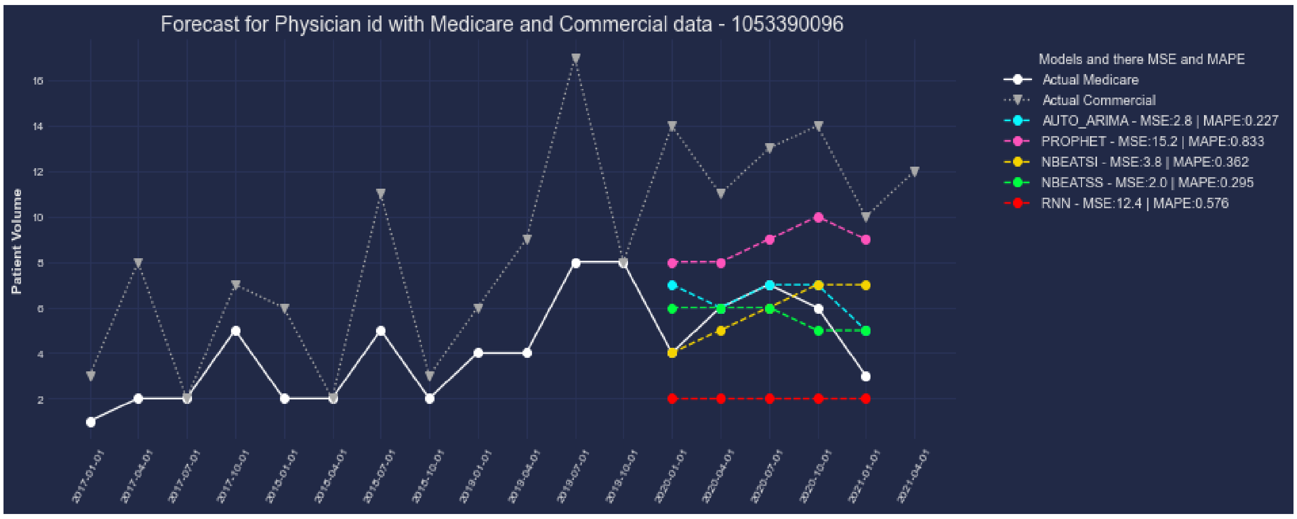
Fig 17 Depiction of the inputs/outputs for "Future Covariates models" at prediction time

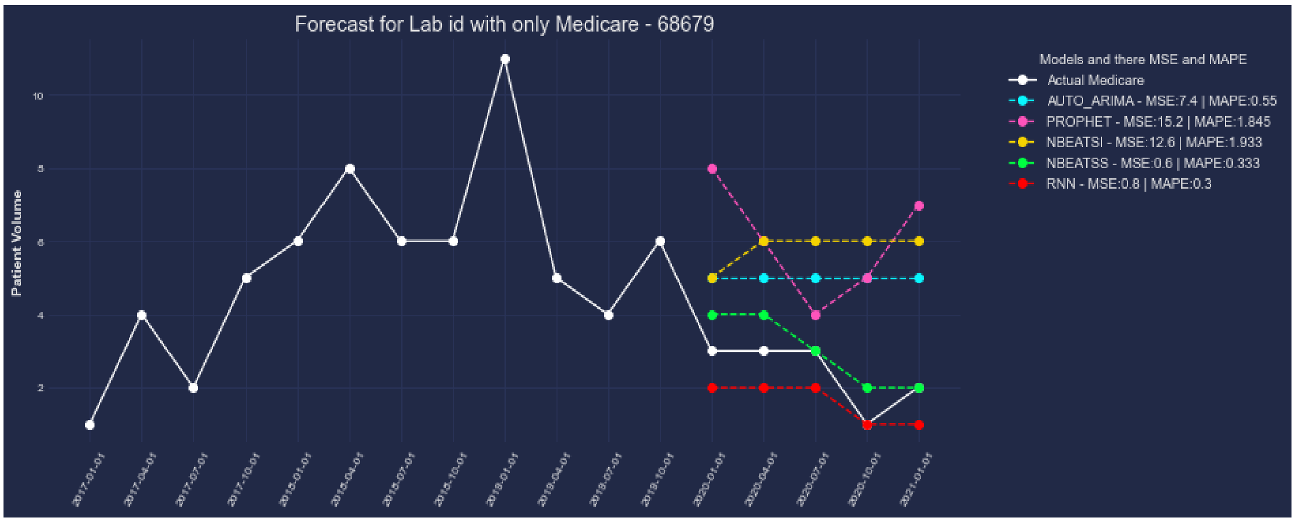
The fit() and predict() methods of these models accept only a future_covariates argument (specifying one or a sequence of Time Series). These models will look only at past values of the covariate series when making a prediction. Grid search is used to identify the best "**input_chunk_length**", "**forecast_horizon**", "**hidden state dimension**", "**model (LSTM / GRU)**" and "**n_epochs**".

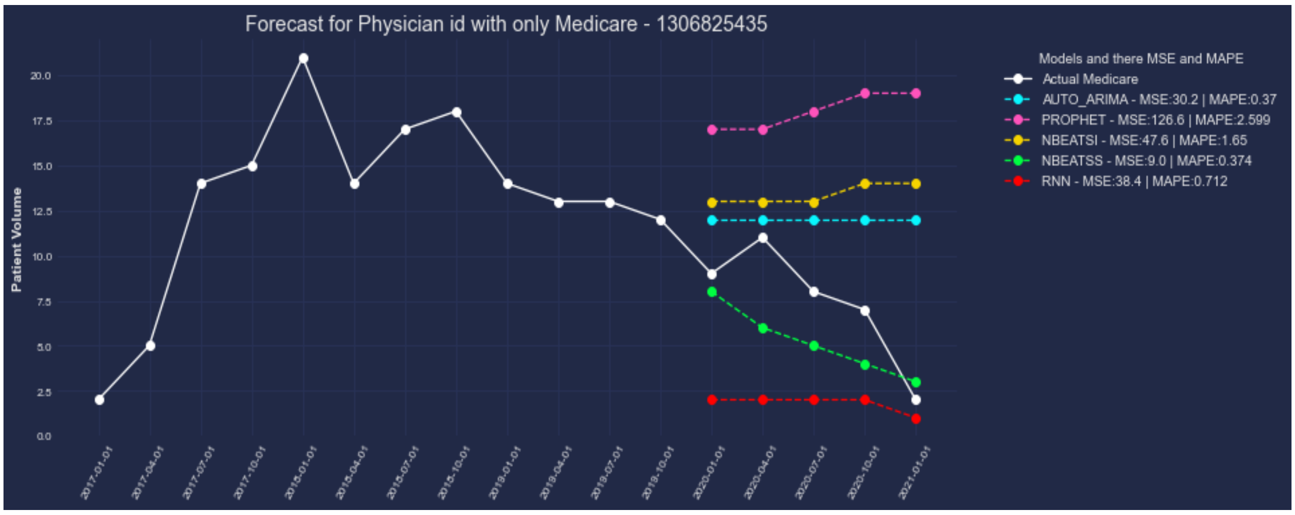
8.7 Visualizing The Test Forecast

Following the training time series using multiple algorithms, the test time series is then used to predict the results. Each model's new forecast is plotted alongside its actual time series, as well as its MSE and MAPE scores. Below are the plots for the random Lab and Physician ID's. Some ID's are having nest MSE with AUTO ARIMA and some with other models likes Nbeats with no common trend or pattern observed.









Using MSE generated for ID below are final best models count.

AUTO_ARIMA → 1808 IDs	PROPHET → 835 IDs	RNN → 698 IDs
NBEATS Multiple →639 IDs	NBEATS Single → 933 IDs	

9. Web Application

A simple Web application is created by integrating Machine Learning into Web Applications with Flask. The final model for each ID is save as a pickle file which will be used to predict the output when new input data is provided from our web-app.

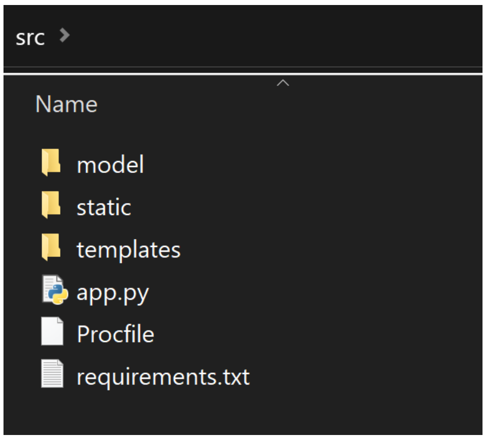
Now that we have our models, we can start developing our web application with Flask and below are the steps followed.

1) Install Flask in IDE. As part of this project Sypder IDE was used to develop flask applications.
2) Import necessary libraries, initialize the flask app, and load our ML models.
3) Define the app route for the default page of the web-app. @app.route("/") is a Python decorator that Flask provides to assign URLs in our app to functions easily. Whenever we hit the app domain *localhost:5000* at the given .route() it execute the home() function takes to 'home.html'. Flask uses the Jinja template library to render templates. In our application, we will use templates to render HTML which will display in the browser.
4) We create a new app route ('/predict') that reads the input from our 'home.html' form and on clicking the predict button, outputs the result using render_template

Project Structure:

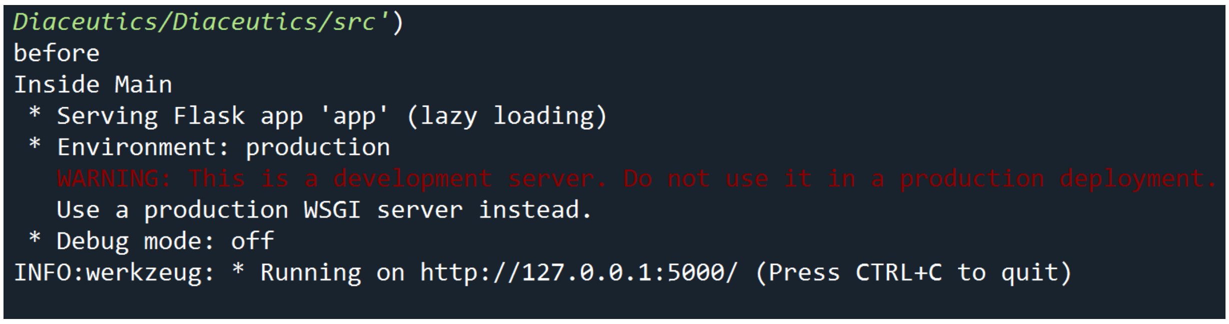
The project is saved in a folder called "src". On running the 'app.py', our application is hosted on the local server at port 5000.

- Model/pickle/best_mse_dict.pkl — This is the pickle file with final ML model for each ID
- app.py — This is the Flask application created
- templates — This folder contains our 'home.html' and 'predict.html' file. This is mandatory in Flask while rendering templates. All HTML files are placed under this folder.
- static — The static folder in Flask application is meant to hold the CSS and JavaScript files and images
- requirement.txt — The requirements.txt file will contain all of the dependencies for the flask app.



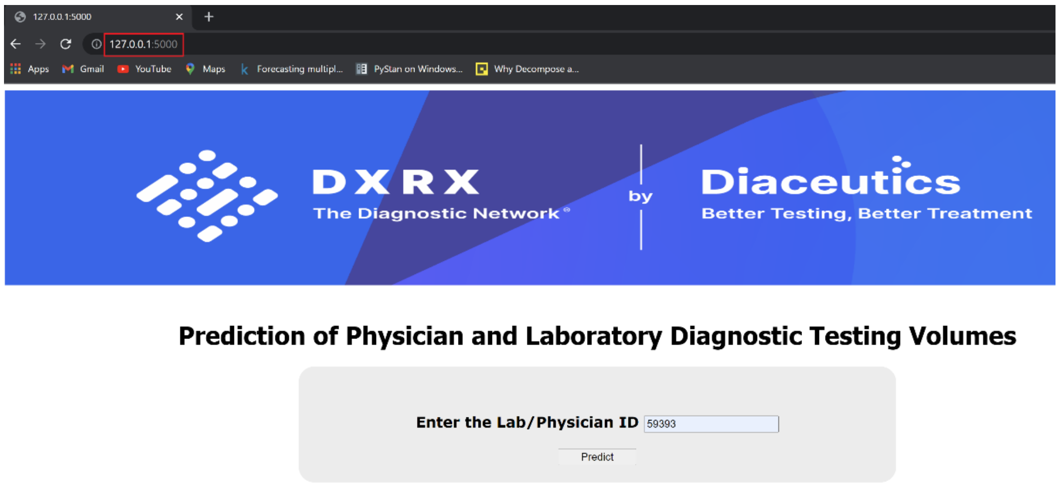
Running Web app:

- When **app.py** is executed



```
Diaceutics/Diaceutics/src')
before
Inside Main
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

- On clicking on the provided URL we get our website:

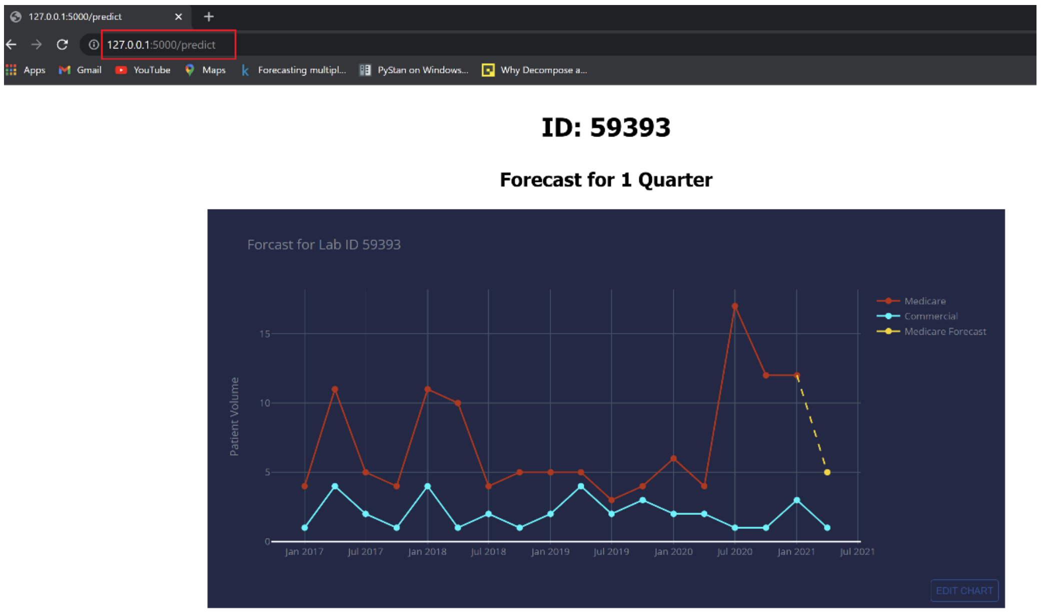


Prediction of Physician and Laboratory Diagnostic Testing Volumes

Enter the Lab/Physician ID 59393

Predict

- On entering the ID and clicking on predict button we are taken to the predict page where the predict function renders the 'predict.html' page with the output of our application.



ID: 59393

Forecast for 1 Quarter

10. References

10.1 Git Hub (Code)

https://github.com/AnjuBanu/Diaceutics_Medicare_Forecast.git

10.2 Research Papers

Application of facebook's prophet algorithm for successful sales forecasting based on real-world data (2020) University of Sarajevo, Bosnia and Herzegovinahttps://arxiv.org/ftp/arxiv/papers/2005/2005.07575.pdf

N-beats: neural basis expansion analysis for interpretable time series forecasting (2020) Element AI https://arxiv.org/pdf/1905.10437.pdf

Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network (2020) https://arxiv.org/pdf/1808.03314v9.pdf

Distributed and parallel time series feature extraction for industrial big data applications (2017) University of Freiburg, Freiburg, Germany https://arxiv.org/pdf/1610.07717v3.pdf

Deep Learning For Time Series Classification (2020) https://arxiv.org/pdf/2010.00567v1.pdf

10.3 Others

Forecasting: Principles and Practice" by Rob J. Hyndman and George Athanasopoulos https://otexts.com/fpp2/arima-r.html

"**Time Series Analysis**" by James Douglas Hamilton

Time Series Made Easy in Python https://github.com/unit8co/darts

Darts Nbeats Documentation https://unit8co.github.io/darts/examples/08-NBEATS-examples.html

Pmdarima https://pypi.org/project/pmdarima/

Time Series Forecasting Methods in Python (Cheat Sheet) https://machinelearningmastery.com/time-series-forecasting-methods-in-python-cheat-sheet/?unapproved=627419&moderation-

Prophet – Forecasting at scale by Facebook https://facebook.github.io/prophet/

Hierarchical Time Series Forecast for Apparel Industry — Doppler Effect https://medium.com/sfu-cspmp/doppler-effect-hierarchical-time-series-forecast-for-apparel-industry-45a55bb23be6

Flash Web application Framework https://palletsprojects.com/p/flask/

Forecasting multiple time-series using Prophet in parallel https://medium.com/spikelab/forecasting-multiples-time-series-using-prophet-in-parallel-2515abd1a245