Forecasting with Time Series Analysis and Classification using Logistic Regression

Pinaki Pani
Student ID: 23112573
Master of Science in Data Analytics
National College of Ireland
x23112573@student.ncirl.ie

***Abstract*—This project presents a dual-phase analysis leveraging Python for two disparate datasets. In the initial segment, an exhaustive time series analysis is conducted on historical weather data sourced from Dublin Airport. Recognizing the pivotal role of weather in daily life, the study focuses on accurate forecasting to enhance convenience. The second phase employs machine learning models for classification on a cardiac dataset, featuring essential attributes of 100 participants—age, weight, gender, and fitness score—to precisely determine the presence or absence of cardiac conditions. By merging these analyses, the research aims to contribute insights into both meteorological forecasting and cardiac health assessment, demonstrating the versatility of Python in diverse analytical domains.**

I. Introduction

A. Background

The project describes the process of analysing weather dataset and cardiac dataset available in comma seperated value(csv) format to check the factors influencing the weather and caridiac conditions using techniques like time series analysis and logistic regression for respective dataset. The weather dataset consists of multiple features influencing weather however the choice of feature depends on the last digit of the student id. The aim of this project is to accurately predict the considered feature for the year 2023 and test its prediction as well. Along with this the cardiac condition is accurately categorised using logistic regression as per the given independent features.

B. Objective

The primary objectives of the project revolves around two datasets. In the first dataset 'weather.csv' one of the given features is extracted using pandas package in python as per the last digit of the student ID. The extracted feature is then analysed using different ways of time series analysis concepts where the modeling of the data is done for the period of 2019 to 2022 and the forecasting is done for 2023. The forecasted data is then evaluated for each and every models applied, from which the best model is chosen. As for the second datatset, 'cardiac.csv' the independent features are analysed to categorize the data using pandas. Using logistic regression machine learning algorithm from python package sklearn, the dependent variable 'cardiac condition' is categorised in a binary way, deciding whether there is any presence of cardiac issue or not.

PART A. Time Series Analysis

II. Exploratory Data Analysis

The datasets are first explored using the python packages like matplotlib, pandas and plotly to analyze the trends and patterns exhibited by them for the purpose of providing crucial insights on the data. Later the insights and analysis are used to prepare the dataset, to run time series analysis models and machine learning models upon respective data.

A. Time Series Analysis - Weather Dataset

The dataset consists of 9 feature columns and 29889 instances of data that ranges over a period from 1st January 1942 to 31st October 2023. As per the brief the time series analysis is to be done for this project is according to the last digit of the author's student ID. Last digit being 3, decides that the feature to be fetched is Minimum Air Temperature given in degrees Celsius. With the help of pandas only this feature along with the date column is fetched into a new dataframe.

1) Dataset Summary: The new dataset upon which the analysis is to be done provides crucial information regarding the minimum temperature that is observed over time since 1942 to October 2023. The date column however is in the format of 'dd-mm-yy' and is of object type. First the date column is converted into datetime type with the date format of 'dd-mm-yyyy'. However upon conversion the dates when observed has an anomaly. The anomaly states that the dates from 1942 to 1972 are all converted to the 21st century, instead of being in 20th century. So using a offset of 100 years for the following anomalous dates using pandas package in python one century is dropped for the following instances.

2) Missing Data: The converted data is then scanned and analysed for null values. Upon verification using appropriate pandas function the feature does not contain any null values and maintains values for all the provided dates.

3) Exploratory Data Analysis: The Exploratory Analysis of the weather data involves mainly looking at the trends and the patterns for the purpose of identifying them and using them for further forecasting.

i. At first the data is plotted, showing the basic nature of the data by the help of line plot functionality with the help of matplotlib package in python. This representation is shown in the figure Fig1. The dataset's basic trend is depticted over the

given time period. The same trend is further supported and is zoomed in in the next figure Fig 2 where further details can be observed in the data's nature over time.

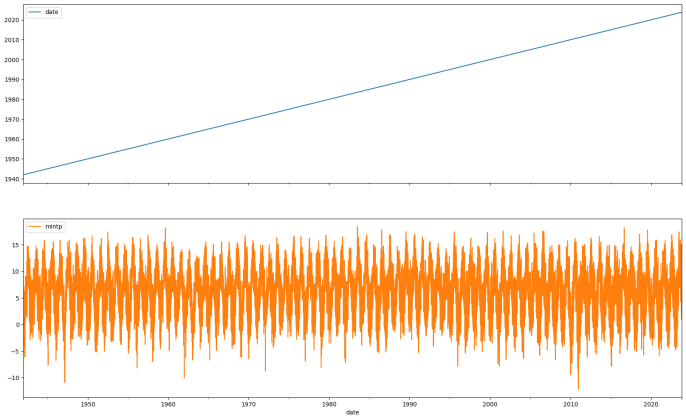

Fig. 1: Line plot of Data

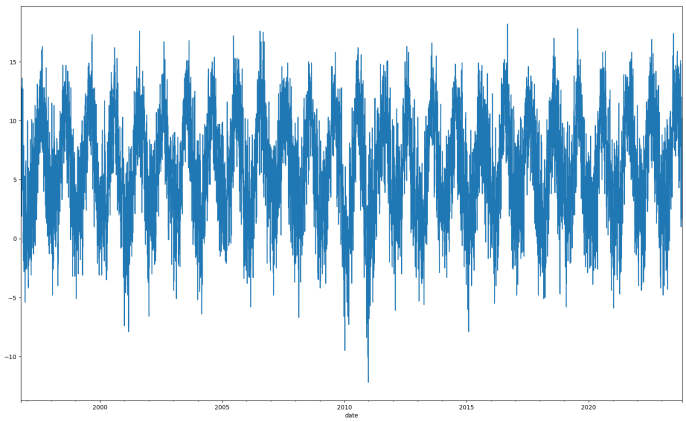

Fig. 2: Trend Plot

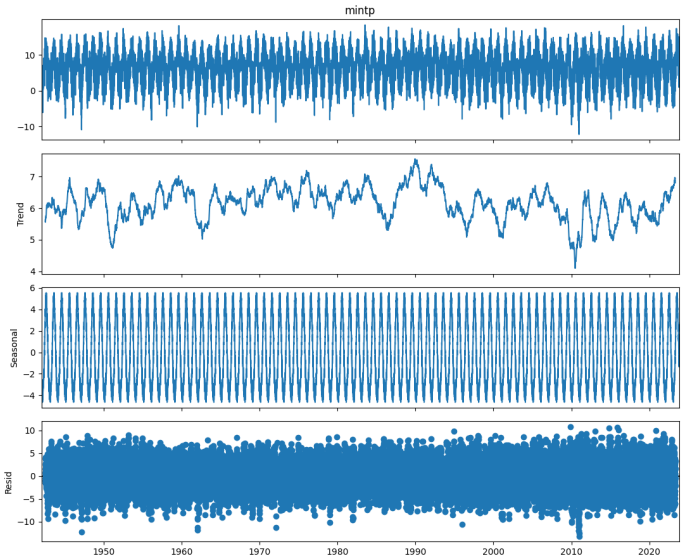

Fig. 3: Decomposed Plot

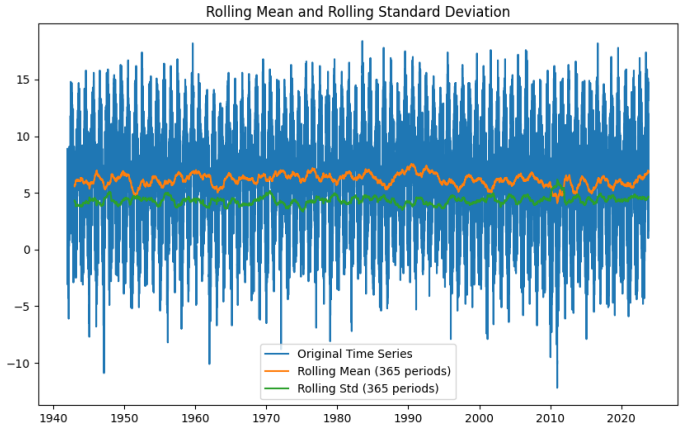

Fig. 4: Zoomed Trend Plot

ii. Using the help of stasmodel package in python the decomposition of the dataset is retrieved and is plotted across 4 rows showing basic flow of the data in the first figure, following which comes the trend of the data. Seasonality of the dataset is then plotted in the figure in third row after which at last are the residual plots. The seaonal decomposition plots are depicted in the figure Fig3.

iii. Next the Rolling mean and the standard deviation is used to smooth out the short term fluctuations in the data to identify key trends and patterns. These metrics are calculated with basic pandas 'rolling' functionality of the dataframes over a moving window of 30 days(monthly), providing a more stable representation of the underlying behavior. The trend of the overall data with rolling mean and rolling standard deviation is shown in the figure Fig 4. The mean and the variance throughout the dataset seems to be stable which is a sign of the dataset being stationary. A closer look to the rolling mean over the years along with the upper and lower bound of the rolling mean is shown in the figure Fig 5.

iv. Next the Null Hypothesis Testing for Augmented Dickey-Fuller Test is performed for the overall dataset to check for stationarity within the dataset. The ADF test is a statistical test used to determine whether a unit root is present in a time series dataset. The presence of a unit root indicates that a time series is non-stationary.Upon running a null hypothesis check to verify if a unit root is present in the time series against the other hypothesis that if the time series is stationary. The adfuller test is done with the help of the statsmodel's stattools package in python. Upon running ADF test the result statistic turns out to be -12.98365232195235, with the critical values as: '1%': -3.4305691789885526, '5%': -2.8616368711691322, '10%': -2.5668215615705376. As the ADF statistic seems to be smaller than all other critical values the null hypothesis of non-stationarity can be easily rejected.

III. Data Preparation

The dataset as discussed above is converted from object type to datetime format for the date column and the feature

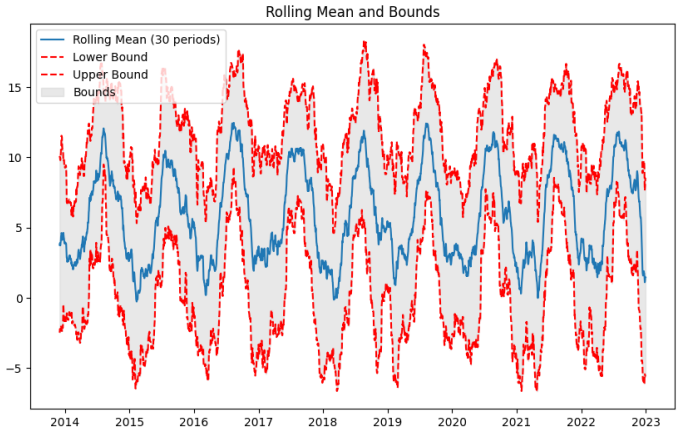

Fig. 5: Zoomed Trend Plot

as per the student ID is extracted for individual exploration and time series analysis. For the simple time series modelling just the rolling mean and variance is used for which using the pandas dataframe's rolling functionality is used to prekpare the data for forecasting. Once the data is ready the data is then checked upon for a particular time frame. The subset of data within the period of January 1st 2019 to 31st December of 2022 i.e, 2019 to 2022 is used for the training purpose of the data and the rest of the dataset of 2023 is used for the forecasting purposes.

A. *Seasonal Differentiation*

To begin with the data preparation for the modeling of time series analysis of the weather dataset, the dataset is decomposed to check for the seasonality of the data. The decomposition with the help of statsmodel's 'seasonal_decompose' functionality. The data represents annual seasonality where the trend repeats over the years. Now that the seasonality is identified, it is handled with the help of pandas dataframe's differentiation functionality. The 'diff' function helps to calculate the difference between the consecutive elements within the dataframe. As we are aware that the data has annual seasonality the period of differentiation is set as 365. Upon careful inspection of the differentiated data with the help of ACF(Autocorrelationo) and PACF(partiall Autocorrelation) plots it was evident that the data still conatined evidence of seasonality. Due to still existing seasonality the data was differentiated again with the same methodology and finally was gotten rid of it. Now the significant spikes within the dataset in the autocorrelation and partial autocorrelation plots were visible within the first few hundreds lags and was easy to identify the values of the ARIMA and SARIMA order(namely p, q, d for ARIMA & P, Q and D for SARIMA). The significant spikes after which the spikes became insignificant were quite evident after the second differentiation. The ACF and PACF of the dataset before differentiation is depicted in figure Fig 6. After the first and second differentiation the plots are represented in figure Fig 7 and Fig 8 respectively, showing the final stable non seasonal dataset.

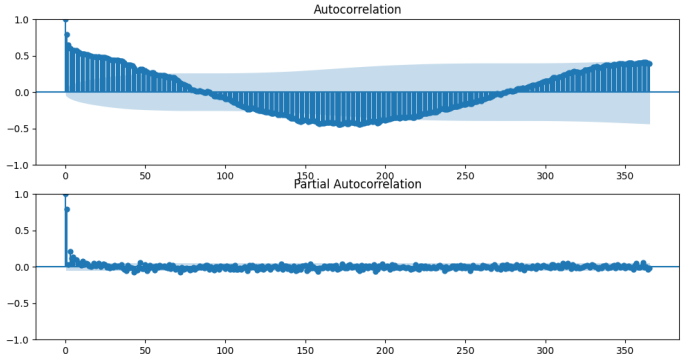

Fig. 6: Initial Seasonality ACF and PACF Plot

Fig. 7: Seasonality after first Differentiation

IV. Modelling

The modeling for the forecasting in the time series dataset is begun once the data become non-stationary and non-seasonal. The ADF test checked and double differentiated data is now ready for the modeling for this project as well.

A. *Data Splitting*

The dataset is split into 2 subsets where the first subset is the dataset within the period of begining of 2019 to end of 2022 and the second subset consists of the data for 2023 which will be later used for forecasting and model evaluation.

B. *Model Selection*

The weather dataset undergoes through multiple time series analysis models for proper evaluation.

1) Simple Time Series Analysis - Rolling Mean: The rolling mean model has been applied over the first subset of dataset(2019-2022) that is saved as training set. The rolling mean over the period of 30 days(monthly) is calculated and is set for the given time period. The forecast for the test time period is represented in the figure Fig 9.

2) Simple Exponential Smoothing: The next model considered for further time series analysis is Simple Exponential Smoothing. Exponential smoothing is a powerful method that takes into account the weighted average of past observations to make predictions for future values. Initially Single Exponential

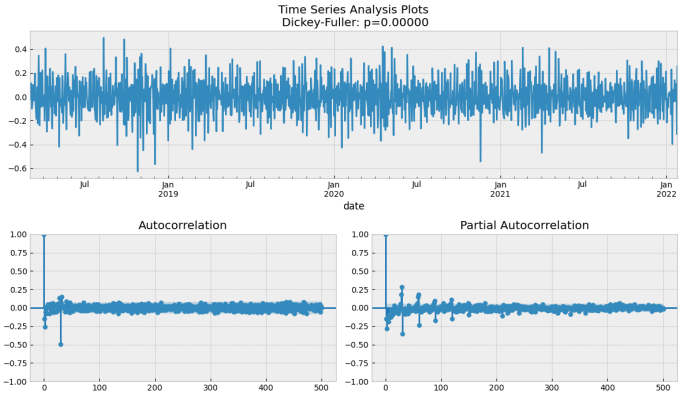

Fig. 8: Final Non Seasonal ACF/PACF Plot

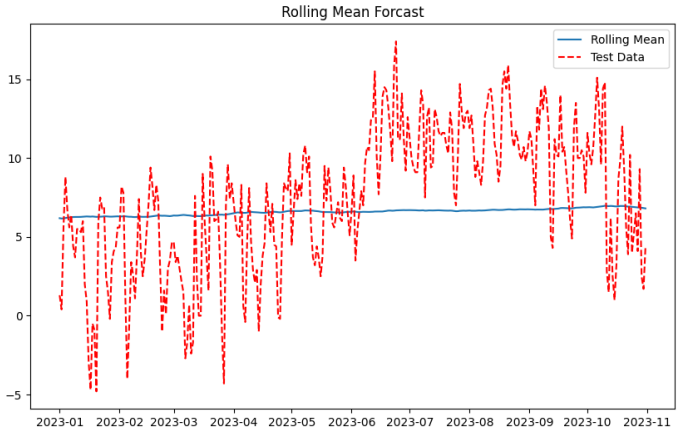

Fig. 9: Simple Time Series Forecast Plot

Smoothing is used to capture the underlying level of the time series. The smoothing parameter α was set at 0.6343376, determining the weight given to the most recent observation. The Simple exponential Smoothing done with the model is shown in the fig 10.

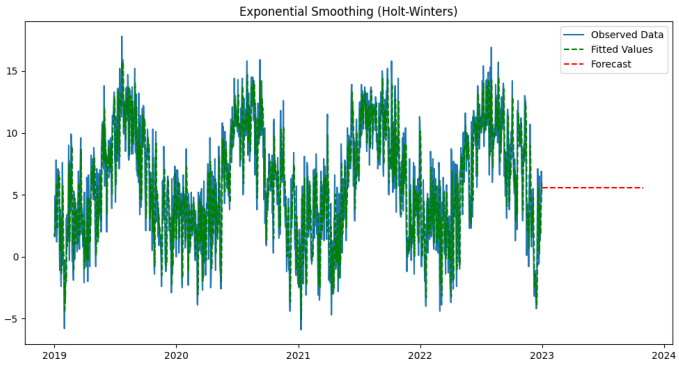

Fig. 10: Simple Exponential Smoothing Plot

3) Triple Exponential Smoothing: However as the weather dataset has seasonality, for time series with seasonality, the Holt-Winters Method was implemented. Smoothing parameters (α, β, and γ) were set at 0.6559994,0.0 and $7.8162e-09$, allowing for the inclusion of level, trend, and seasonality. The trained dataset and the fitted plot is shown along with the forecast in red line within the plot is represented in figure Fig 11.

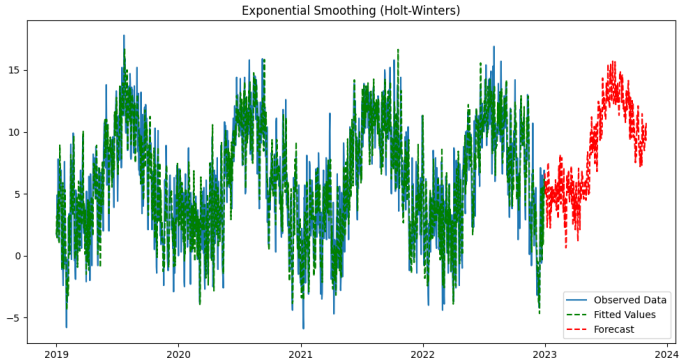

Fig. 11: Triple Exponential Smoothing Plot

4) Auto-Regressive Integrated Moving Average(ARIMA): The ARIMA model is chosen for its ability to effectively capture trends, seasonality, and autocorrelation in time series data.The ARIMA model is specified as ARIMA(p, d, q), where: p: Autoregressive (AR) order,d: Integration (I) order (number of differences) and q: Moving Average (MA) order.

5) Seasonal Auto-Regressive Integrated Moving Average(SARIMA): The SARIMAX model is an extension of the ARIMA model that includes exogenous variables and is chosen for its ability to capture seasonality, trends, and external factors influencing the time series. The SARIMAX model is specified as SARIMAX(p, d, q)(P, D, Q),s, where p is Autoregressive or aka AR order, d is the Integration order (number of differences), q is the moving average or MA order, P is the seasonal AutoRegressive order, D is seasonal integration order, Q is seasonal Moving Average order and finally s is the seasonal period. The values for the ARIMA and the Seasonal ARIMA orders are identified by the help of a helper function written in python and identified by the aic values obtained from the models ran over some given sets of possible values of mentioned orders. The best aic values are recorded by sorting them in ascending order in the function and the lowest values are returned.

V. Interpretation

The time series models are run one after another and the resepective p values and coefficients are recorded and displayed along with their residuals.

A. Simple Time Series Model - Rolling Mean

The rolling mean of the whole dataset is calculated as discussed above and the forecast for the test subset of the data is calculate resulting in a Mean Squared Error of 19.737856308332756.

B. Simple Exponential Smoothing

The Simple Exponential smoothing upon running shows a straight line forecast and the associated statistics for the model has some developments however aren't better than rolling mean model. The model result are summarised by the 'model.summary' functionality in python. The AIC recorded for the model has a values of 3041.072 and as is a Simple Exponential Smoothing so the associated alpha value is recorded at 0.6343376 which is also denoted to be optimized. The results are depicted in the figure Fig 12 and the residual distribution for the model is depicted in the figure Fig 13 where it can be seen that the residuals shows signs of slight right skewedness. The residuals for all the models are calculated by the help of python's 'model.resid' instances of models which is available in the statsmodel package, once they are fit over the data .

SimpleExpSmoothing Model Results

Dep. Variable:	mintp	**No. Observations:**	1461
Model:	SimpleExpSmoothing	**SSE**	11680.067
Optimized:	True	**AIC**	3041.072
Trend:	None	**BIC**	3051.646
Seasonal:	None	**AICC**	3041.100
Seasonal Periods:	None	**Date:**	Sat, 30 Dec 2023
Box-Cox:	False	**Time:**	16:03:41
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.6343376	alpha	True
initial_level	1.8000000	l.0	False

Fig. 12: Simple Exponential Smoothing Result Summary

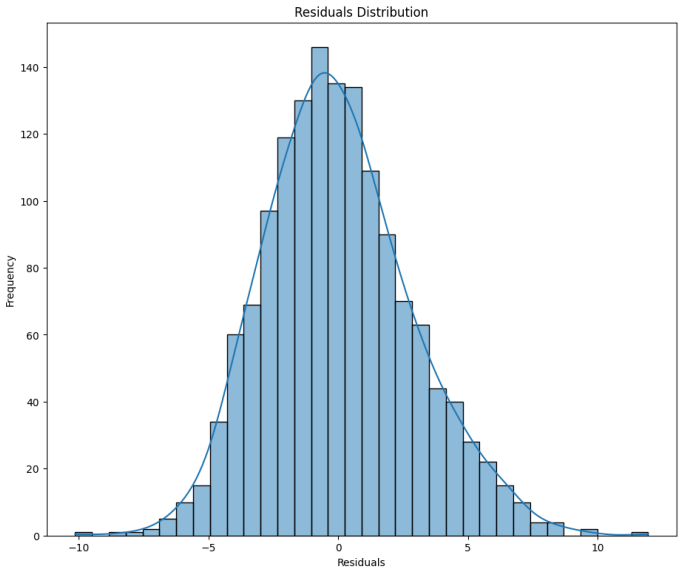

Fig. 13: Simple Exponential Smoothing Residual Distribution

C. Tripple Exponential Smoothing

The Triple Exponential Smoothing however has a better result and optimizes the result quite a bit. It also identifies the seasonality and the trend within the data. The AIC recorded for this model is 3370.692. The seasonality is identified to be 'Additive', and the recorded alha and gamma values are 0.6559994 and $7.8162e-09$ respectively. The Jarque-Bera testing results with a statistic of 13.796025983400762 and the p value is evaluated to be 0.0010097898978488327. The model's results are shown in the figure Fig 14 and the residual distribution is plotted in figure Fig 15 where the residuals still show signs of slight right skewedness.

ExponentialSmoothing Model Results

Dep. Variable:	mintp	**No. Observations:**	1461
Model:	ExponentialSmoothing	**SSE**	8880.330
Optimized:	True	**AIC**	3370.692
Trend:	None	**BIC**	5310.976
Seasonal:	Additive	**AICC**	3620.977
Seasonal Periods:	365	**Date:**	Sat, 30 Dec 2023
Box-Cox:	False	**Time:**	03:39:00
Box-Cox Coeff.:	None		

	coeff	code	optimized
smoothing_level	0.6559994	alpha	True
smoothing_seasonal	7.8162e-09	gamma	True
initial_level	4.5436588	l.0	True
initial_seasons.0	-2.8259870	s.0	True
initial_seasons.1	-1.3828258	s.1	True
initial_seasons.2	-2.0891898	s.2	True

Fig. 14: Tripple Exponential Smoothing Result Summary

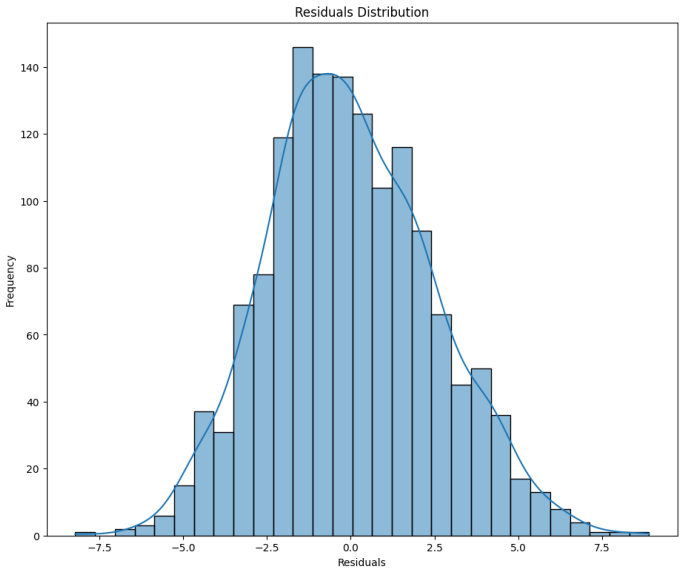

Fig. 15: Tripple Exponential Smoothing Residual Distribution

D. SARIMAX

The SARIMAX model is finally used for the purpose of handling the seasonality perfectly and helps to forecast the values with way better results. Upon checking the statistics all coefficients have p-values less than 0.05, suggesting statistical significance. The metric at 0.05 for p-value of the Ljung-Box test It suggests some autocorrelation at lag 1, but is borderline significant. However, 0.14 asthe p-value for the Jarque-Bera test shows that the the residuals may not be perfectly normal but has barely any skewedness and is symmetrical as skew measure is 0.03. The p-value for the test of heteroskedasticity records to be a higher p-value with a measure of 0.90, suggests that there is no strong evidence of heteroskedasticity as well. The Residual distribution and Result are shown in the fig 16 and 17 respectively.

SARIMAX Results

Dep. Variable:	mintp	**No. Observations:**	1461
Model:	SARIMAX(3, 2, 3)x(2, 2, [1, 2], 12)	**Log Likelihood**	739.885
Date:	Sat, 30 Dec 2023	**AIC**	-1457.770
Time:	03:28:46	**BIC**	-1399.812
Sample:	01-01-2019	**HQIC**	-1436.131
	- 12-31-2022		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-1.1431	0.027	-42.197	0.000	-1.196	-1.090
ar.L2	-0.4374	0.040	-10.962	0.000	-0.516	-0.359
ar.L3	-0.2565	0.027	-9.420	0.000	-0.310	-0.203
ma.L1	-0.9917	0.106	-9.384	0.000	-1.199	-0.785
ma.L2	-0.9994	0.210	-4.758	0.000	-1.411	-0.588
ma.L3	0.9917	0.106	9.364	0.000	0.784	1.199
ar.S.L12	-0.0651	0.033	-1.975	0.048	-0.130	-0.000
ar.S.L24	-0.0415	0.030	-1.379	0.168	-0.101	0.018
ma.S.L12	-1.8612	0.022	-85.412	0.000	-1.904	-1.818
ma.S.L24	0.8659	0.022	39.950	0.000	0.823	0.908
sigma2	0.0189	0.002	9.121	0.000	0.015	0.023

Ljung-Box (L1) (Q):	3.81	**Jarque-Bera (JB):**	3.99
Prob(Q):	0.05	**Prob(JB):**	0.14
Heteroskedasticity (H):	0.90	**Skew:**	0.03
Prob(H) (two-sided):	0.25	**Kurtosis:**	3.25

Fig. 16: SARIMA Result Summary

VI. Diagnostics

The Simple Time series model has close predictions given the result is a straight line as shown above in the figure Fig 9.

Similar Straight line forecasting is noted from the Simple Exponential Smoothing model as well and is depicted in the figure Fig 18.

The triple Exponential Smoothing however does identify the seasonality in the dataset and forecasts are pretty close to

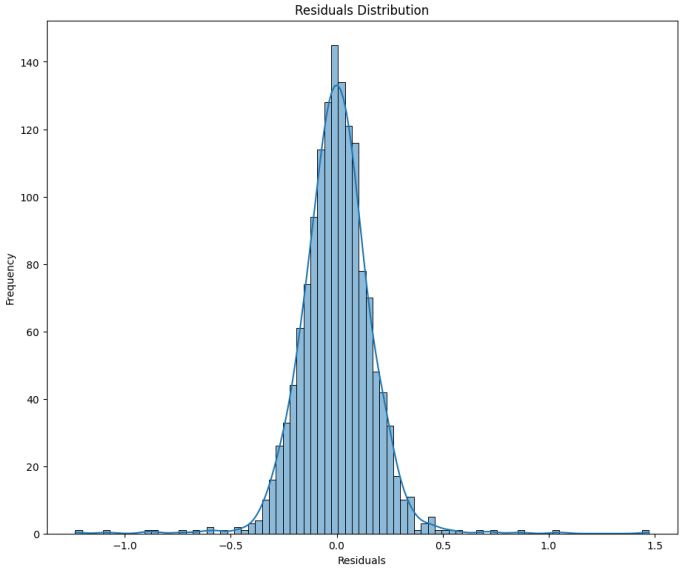

Fig. 17: SARIMA Residual Distribution

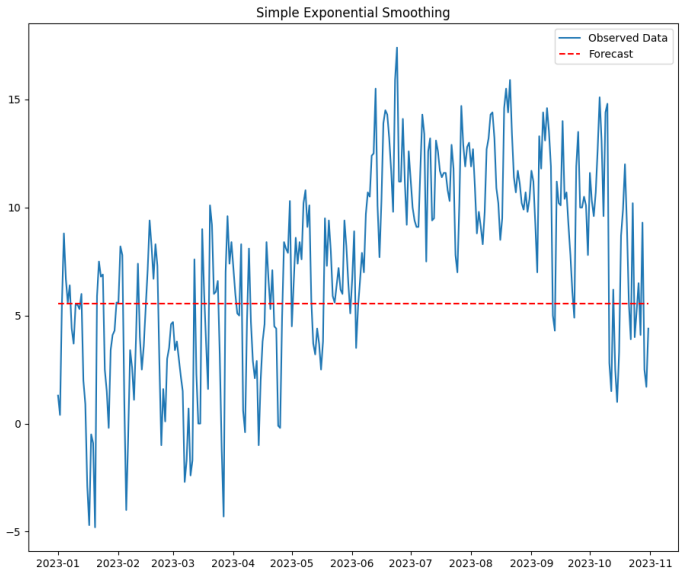

Fig. 18: Simple Exponential Forecasting

the actual test dataset. The forecast can be viewed in the figure Fig 19. Even though the model has better results than others still the residuals seems to be skewed and the Jarque-Bera Test Statistic is quite low at 13.796025983400762 The Arima is shown in the project however it is understandable that ARIMA would result in a straight line as there is no scope for seasonality in it but the dataset is seasonal. Hence SARIMAX is considered.

The SARIMAX model here is used to handle the seasonality of the data. The implementation of SARIMA is done with the statsmodel package in python where the 'tsa.statspace' package comes in handy to provide the SARIMAX model. With the checked statistics above in the interpretation section,

the SARIMAX models has the best statistics and optimizes to best results. The forecast for SARIMA is represented in the figure Fig 20. One of the main reason why SARIMAX is working best here is also due to the implementation of the rolling mean along with it. The SARIMA model is used over the rolling average dataset that was calculated with the window size of 30 (i.e, monthly). This allows to create the SARIMA model over the dataset and use the seasonal period as 12 instead of 365 as 365 runs infinitely in a basic Operating system. The amount of operations and space requirement becomes quite lower and works well. .

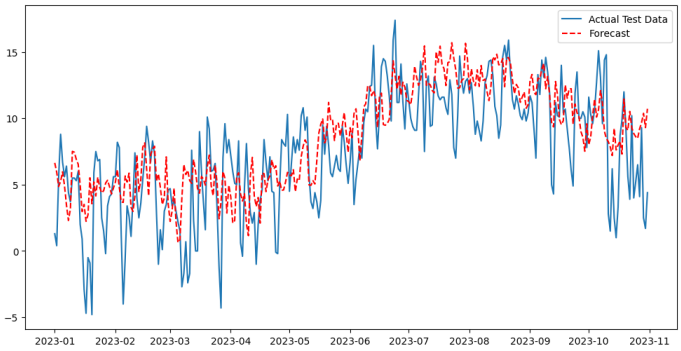

Fig. 19: Triple Exponential Forecasting

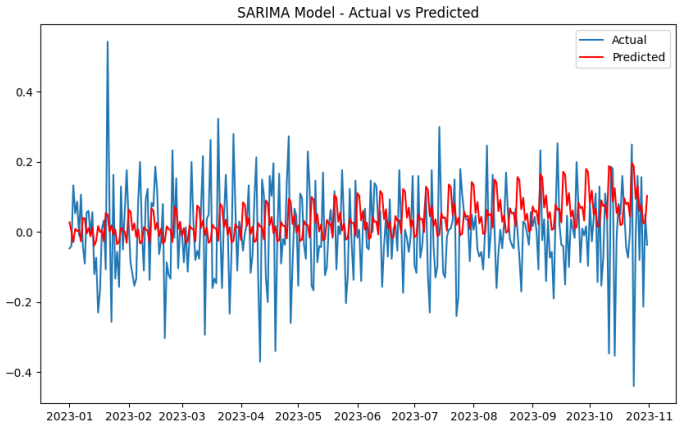

Fig. 20: SRIMAX Forecast Result

VII. Evaluation

The model evaluation for each and every model is done by the help of Mean Squared Error and Mean Absolute Error functionality available from the python package sklearn's metric module. This helps to calculate the mean of the squared errors of the prediction when compared to the actual values.
The Simple time series model where the rolling mean is considered to do forecasting over the test dataset results an MSE value of 19.737856308332756. The next model where the Simple Exponential Smoothing is used was expected to work better however the resulting MSE is larger than rolling mean model with a value of 23.494219181576863. The resulting forecast is a straight line.
As the Simple exponential works worse than the rolling mean due to not identifying the seasonality, triple exponential smoothing with (α, β, and γ) parameters was used now to identify the trend and the seasonality. This works way better than both the models with a resulting MSE value of 11.058754236758848.
Even though the result was better still the MSE seems to be a bit higher and hence Seasonal ARIMA was used as a result. The SARIMA results with a way better MSE and MAE values where the metric records to be 0.02 and 0.11 respectively. The final SARIMA model has the best result and the best fitting plot to the actual values so far.

PART B: Logistic Regression

II. Exploratory Data Analysis

The cardiac dataset is first explored using the python packages like matplotlib, pandas and plotly to analyze the trends and patterns exhibited by them for the purpose of providing crucial insights on the data. Later the insights and analysis are used to prepare the dataset, to run time series analysis models and machine learning models upon respective data.

1) Logistic Regression - Cardiac Dataset Summary: The data set includes 100 rows and 6 features, where the columns 'gender' and 'cardiac_condition' are of object type. In the 'gender' column, textual data denotes the gender of individuals, categorized as male or female. Likewise, the 'cardiac_condition' column uses textual format to represent binary data indicating the presence or absence of a cardiac condition in patients. Encoding these object-type columns is crucial for improving data analysis

2) Missing Data: The cardiac dataset is free of any missing values. There are no instances of Null or blank values within the dataset, ensuring smooth operations, and the dataset is entirely devoid of any cleanliness issues. The missing values are checked with the help of isna functionality present in pandas dataframes in python.

3) Exploratory Data Analysis: The Exploratory Analysis of the weather data involves mainly looking at the trends and the patterns for the purpose of identifying them and using them for further categorizing using machine learning models.
i. The data is initially converted into numerical columns from categorical using the Label Encoder functionality available in the sklearn package. Now to begin with the target column in checked for the existing categories and their count. The countplot allows to explore the class imbalance checkpoint in case there is any. Upon checking the histplot using the count of the categories it is evident that No cardiac condition cases are 65 observations and there are 35 observations of existing cardiac condition. The countplot with histograms are plotted and shown in the figure Fig 21.

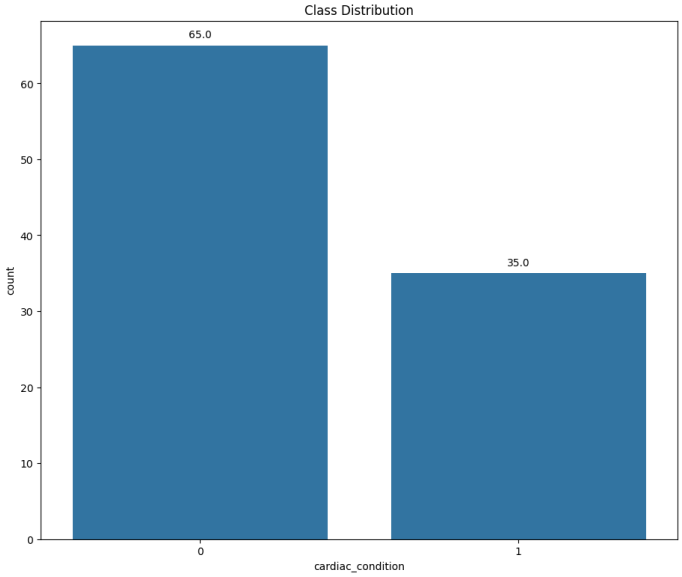

Fig. 21: Countplot of Target variable in Data

ii. Next the Correlation matrix for the numerical values are checked by the help of pandas 'corr' function for the dataframe and is plotted using seaborn package's 'heatmap' functionality. Upon careful consideration of the correlation matrix it is evident that there are not significantly high correlation between the features. The correlation matrix is represented in the figure Fig 22.

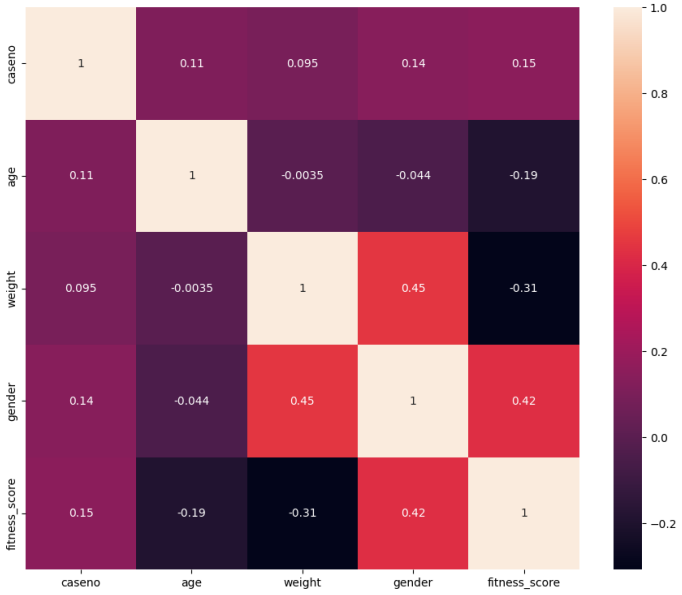

Fig. 22: Correlation Matrix Heatmap

iii. Next the histograms are plotted to show the distribution of the continuous numerical plots. The distribution for Age, Weight and Fitness Scores are shown in the below distribution plot. The plots are shown below in figures Fig 23, 24 and 25 respectively. Upon checking the distributions we can identify that the Age column is the column that is quite right skewed.

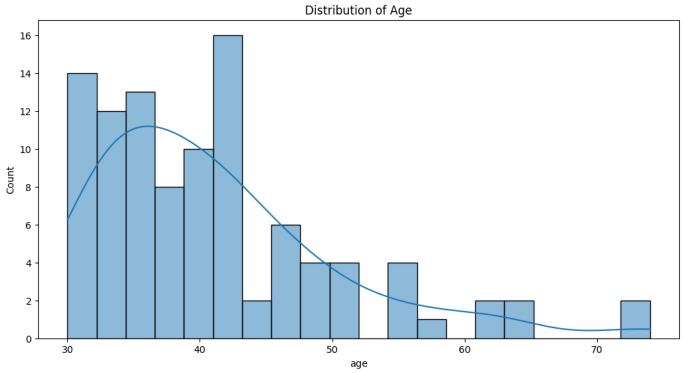

Fig. 23: Age Distribution Plot

iii.

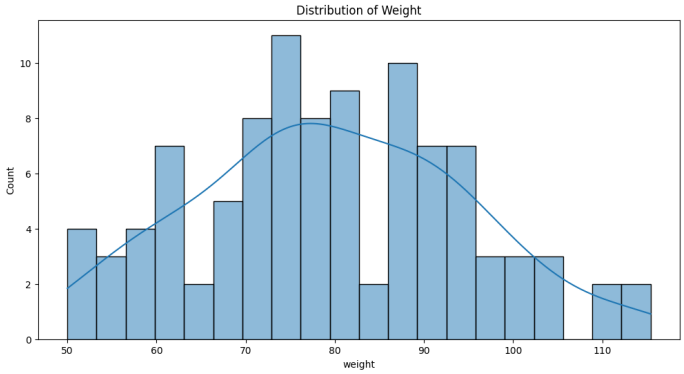

Fig. 24: Weight Distribution Plot

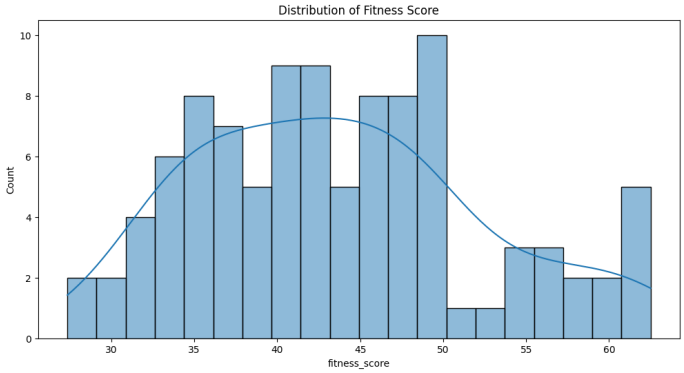

Fig. 25: Fitness Score Distribution Plot

iv. The distribution of the Fitness Score divided as per the gender is shown by the help of the violin plot in the figure Fig 26.

III. Data Preparation

The dataset as discussed above is converted from categorical non numerical to numerical using the Label Encoder package

Fig. 26: Fitness Score Violin Plot as per Gender

as mentioned above in the Exploratory Data Analysis section. Once the data is encoded and converted into numerical format, the imbalance of the target variable is checked using the value counts for the cardiac condition column. Upon careful consideration the caseno column has no contribution to the classification model and is hence dropped.

A. Data Splitting

The dataset is then divided into two subsets where the features except the target is stored in the variable called 'X' and the target column data is stored in a variable called 'y'.

B. Data Standardization

Once the dataset is divided into two subsets the dataset is scaled using the sklearn package's 'StandardScaler' functionality. The whole dataset is now standardized. Now that the features are all standardized the entire feature set and the target subset is divided into four further variable using the 'train_test_split' functionality available in the sklearn's preprocessing package. The splitting is done with the help of random seed same as the student id. There is a initial modeling done at this point that is discussed in the Modeling section of the report.

C. Boxplot and Outlier Handling

Next as the initial model's result is obtained the outliers are checked for the existing numerical columns. The existing boxplots are shown in the figure Fig 27. Once the 1.5 times of the inter quartile range is calculated and the data within that is contained for all the columns then the boxplots of the columns are plotted again. The boxplots after the outlier handling is shown in the figure Fig 28.

IV. Modelling

The modeling for the classification is started once the dataset is deemed ready after cleaning and preparation phase.

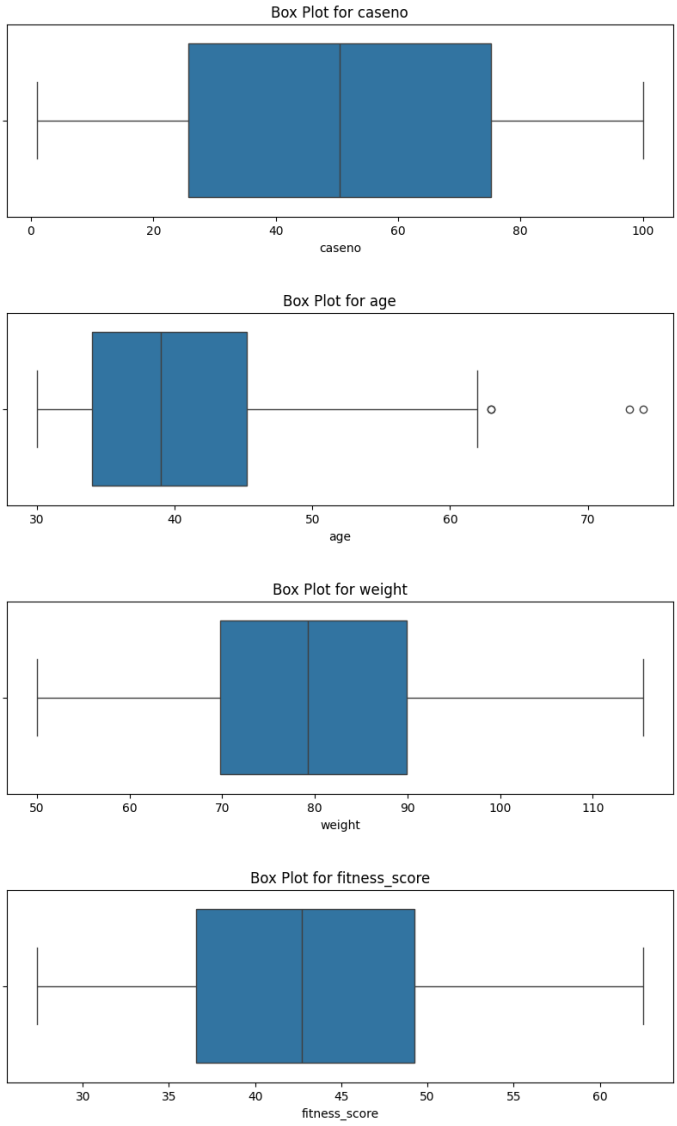

Fig. 27: Initial Boxplots for the Numeric Columns

D. Data Splitting

The dataset is split into training and testing set using sklearn's preprocessing package's 'train_test_split' functionality, where the seed used is same as the student ID.

E. Model Selection

For the Cardiac dataset the model selected is Logistic Regression.
Logistic Regression is a powerful tool for modeling binary outcomes, making it crucial to carefully select the appropriate model for accurate predictions. In this report, we delve into the model selection process for Logistic Regression, emphasizing key considerations and techniques.

V. Interpretation & Diagnostics

The initial model evaluates the performance of a Logistic Regression model on a binary classification task using various

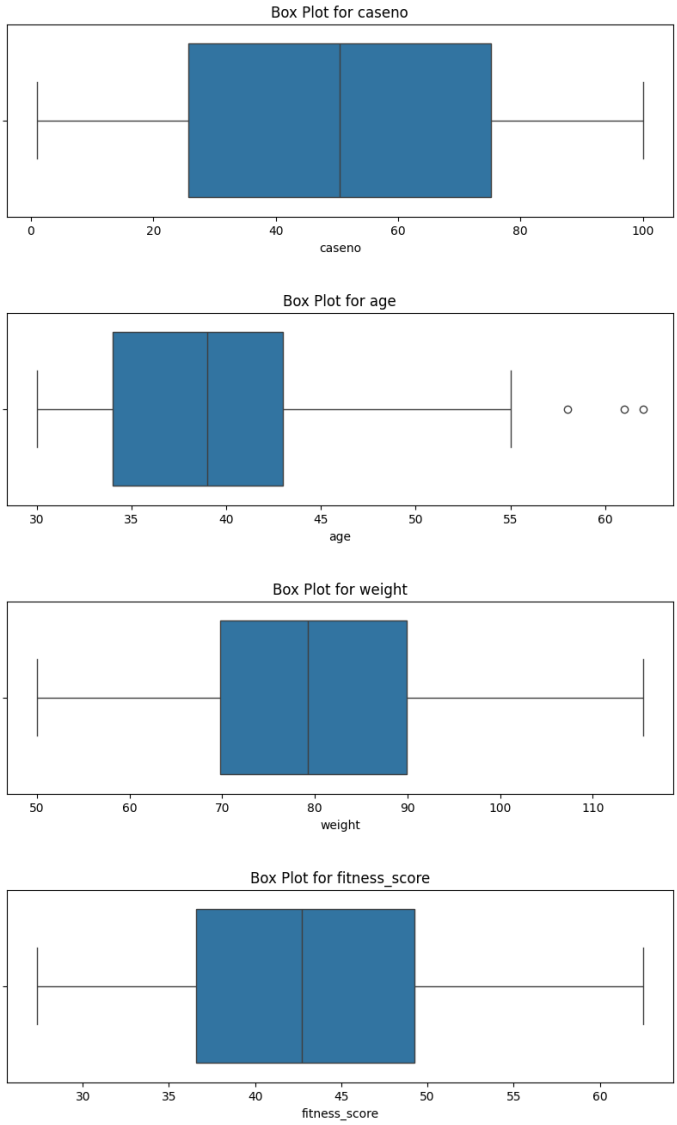

Fig. 28: Final Boxplot after Handling outliers

strategies to address class imbalance. The dataset consists of two classes, 0 and 1, with a relatively small number of instances in the minority class (class 1). The initial evaluation on unbalanced data showcases an accuracy of 80%, with precision, recall, and F1-score metrics illustrating a reasonable performance across both classes. The model appears to be slightly more effective in predicting instances of class 0 compared to class 1.

To mitigate the effects of class imbalance, three different techniques were employed: random oversampling, random undersampling, and the use of class weights. Random oversampling involves augmenting the minority class by duplicating instances, while random undersampling reduces the majority class by removing instances randomly. The results demonstrate that random oversampling has a positive impact on the model's ability to identify instances of the minority class (class 1), as evidenced by improvements in recall and F1-score for this class. However, it is essential to note that random oversampling does not completely eliminate the imbalance, and there is a trade-off with an increase in the false positive rate. On the other hand, random undersampling, while improving the recall and F1-score for class 1, results in a decrease in precision. This trade-off is expected as random undersampling reduces the number of instances used for training, potentially leading to a less accurate prediction for the majority class (class 0).

The intermediate Logistic Regression model has been tuned and optimized using a pipeline that incorporates a Random OverSampler for handling class imbalance. The hyperparameters were systematically explored through a grid search, and the best configuration was identified based on accuracy. The optimal hyperparameters for the Logistic Regression model are a regularization parameter (C) of 1 and a penalty term of 'l2'.The selected hyperparameters indicate that a moderate level of regularization is preferred, and the 'l2' penalty is chosen, suggesting that the model is utilizing Ridge regularization. Ridge regularization (L2 penalty) adds a term to the cost function that penalizes large coefficients, helping to prevent overfitting.

For the final model the analysis involves the creation of a pipeline integrating Principal Component Analysis (PCA) and Logistic Regression, aimed at enhancing the predictive performance of the model through dimensionality reduction. The process includes hyperparameter tuning using a grid search, and the optimal configuration is determined based on accuracy. The pipeline consists of two main components: PCA, which reduces the dimensionality of the feature space, and Logistic Regression, a classic classification algorithm. The hyperparameters selected for tuning include the number of components retained by PCA (n_components), the regularization parameter for Logistic Regression (C), and the penalty term (penalty), which can be either 'l1' or 'l2'(lasso or ridge).

.

VII. Evaluation

The model evaluation for each and every model is done by the help of accuracy check along with some other measure like precision, recall and F1 score. On evaluating the tuned model on the test set, it yields an accuracy of 85%, showcasing the effectiveness of the hyperparameter tuning process in enhancing the model's performance on it predictive capability. It's important to note that accuracy alone might not fully capture the performance, especially in the context of imbalanced datasets. Therefore, it is recommended to complement accuracy with other metrics such as precision test, recall test, and F1-score evaluation, especially when dealing with class-imbalanced scenarios.

However the evaluation of the final model with the best hyperparameters on the test set yields an accuracy of 90%. This high accuracy suggests that the combination of PCA for dimensionality reduction and Logistic Regression with optimized hyperparameters has significantly improved the model's ability to correctly classify instances.

Accuracy: The PCA and Logistic Regression model maintained its high accuracy at 90%.
Precision: The final model achieved a high precision for both classes, with 88% for Class 0 and 100% for Class 1.
Recall: Class 0 achieved perfect recall, while Class 1 demonstrated a recall of 67
F1 Score: The model showed high F1 scores for both classes, with a balanced F1 score of 0.93 for Class 0 and 0.80 for Class 1. The PCA and Logistic Regression model continues to exhibit impressive performance, offering a balance between precision and recall, making it a strong candidate for the classification task. The interpretation of coefficients and the classification report provides additional insights into the model's behavior and its ability to distinguish between the two classes. The choice of the most suitable model should align with the specific goals and considerations of the classification task.

Overall Conclusion

In the comprehensive analysis conducted for both time series and logistic regression, it was aimed to address distinct challenges and gain insights into different aspects of the data. The synergy of time series analysis and logistic regression offers a holistic understanding of the data. The time series analysis provided a foundational understanding of temporal patterns, while logistic regression helped to equip predictive capabilities for classification tasks. This comprehensive approach not only uncovers historical trends but also prepares researchers for making informed decisions based on future outcomes. For time series analysis Exponential smoothing models were employed to forecast future values based on historical data. The optimized models, evaluated through metrics like AIC and BIC, allowed us to fine-tune parameters for enhanced forecasting accuracy. Ljung-Box test, Jarque-Bera test, and other diagnostic checks ensured the models' reliability, shedding light on potential issues such as autocorrelation and normality of residuals. Finally the best results were obtained using Seasonal ARIMA(SARIMA) over the time series data with an outstanding MSE result.

For the cardiac dataset the PCA and Logistic Regression model emerged as a standout performer, boasting a remarkable 90% accuracy. The precision-recall trade-off was carefully examined, and the coefficients' interpretation provided insights into feature importance. In conclusion, the combined analysis equips stakeholders with a robust toolkit to harness the power of historical data for predictive modeling and informed decision-making. The iterative nature of these analyses allows for continuous refinement and optimization, ensuring adaptability to evolving data landscapes and dynamic business environments.

Acknowledgments

The completion of this project has been made possible through the support of Professor John Kelly, for his guidance, advice and critical feedback.

References

[1] Statology website https://www.statology.org/tutorials/
[2] *G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, "Time Series Analysis: Forecasting and Control," Wiley, 2015.*
[3] *R. H. Shumway and D. S. Stoffer, "Time Series Analysis and Its Applications: With R Examples," Springer, 2016.*
[4] *P. J. Brockwell and R. A. Davis, "Introduction to Time Series and Forecasting," Springer, 2016.*
[5] *J. Durbin and S. J. Koopman, "Time Series Analysis by State Space Methods," Oxford University Press, 2012.*
[6] *T. C. Mills, "Applied Time Series Analysis," Academic Press, 2011.*
[7] *R. J. Hyndman and G. Athanasopoulos, "Forecasting: principles and practice," OTexts, 2018. [Online]. Available: https://otexts.com/fpp3/.*
[8] *P. J. Brockwell and R. A. Davis, "Time Series Analysis: Theory and Methods," Springer, 2009.*
[9] *T. S. Ruey, "Analysis of Financial Time Series," Wiley, 2010.*
[10] *D. W. Hosmer Jr., S. Lemeshow, and R. X. Sturdivant, "Applied Logistic Regression," Wiley, 2013.*
[11] *S. Chatterjee and A. S. Hadi, "Regression Analysis by Example," Wiley, 2006.*
[12] *A. Agresti, "Categorical Data Analysis," Wiley, 2002.*