Time series Analysis on Temperature data and Logistic Regression on Diabetes data

Pratheek Gogate
National College of Ireland
School of Computing
Dublin, Ireland
X22159789@student.ncirl.ie

PART-A: Time Series Analysis of Temperature Data ***Abstract*—Weather Forecasting is a very important phenomenon as it will help people to be prepared for weather that is coming their way, it can be either rain or extreme temperatures on both ends. In this project, we have two data sets that contain Temperature data. The first data set has a monthly temperature whereas the second data set has a yearly temperature. The data was created by the Climate Institute of the University of East Anglia and it had data from 1844 to 2004. Our objective was to understand the data and visualize it and choose the best models in Exponential Smoothing, ARIMA/SARIMA, and Simple time series models and then choose the best overall model for each data set, and then forecasted the data for 2004.**

I. OBJECTIVE

Weather forecasting is something very useful in day-to-day life as people can plan what to wear and also they can plan accordingly. If there is a chance of rain maybe they can plan for some indoor things or if it's a sunny day, they can plan for pick-nick and so on. So in our case, we have 2 data sets that have temperature data, based on months and years from 1844 to 2004 and we need to understand and visualize it and then build the models in Exponential Smoothing, ARIMA/SARIMA, and Simple time series models and choose the best in each type and then choose the best among those and forecast the value for 2004.

II. DESCRIPTION OF DATA SET

We have two data sets, one for the year and one for the month having temperature data and this had the data from 1844 to 2004. initially, we had only temperature and later we added month and year columns separately as we knew that this is the month and year-based data as it will help to plot with 2 columns. Then we plotted the graph for both to find the pattern after confirming that there are no null values. in figure 1 we could see the monthly data set after adding a month column, We have done the same for yearly data as well.

In Figures 2 and 3, we could see the ggplots for monthly data and yearly data. To be frank, it's very difficult to comment on monthly data as there are almost 1900 data points in it, and hence we can't really point out as points are densely located but pattern wise this looks like there is not much change as it looks constant. In yearly data, we could see a pattern as the temperature was on the lower side from the 1880s to 1900 and then there was not much difference in the trend.

	date	x
	<date>	<dbl>
1	1844-01-01	4.5
2	1844-02-01	2.4
3	1844-03-01	4.8
4	1844-04-01	9.1
5	1844-05-01	10.9
6	1844-06-01	12.9

Fig. 1. Monthly Data

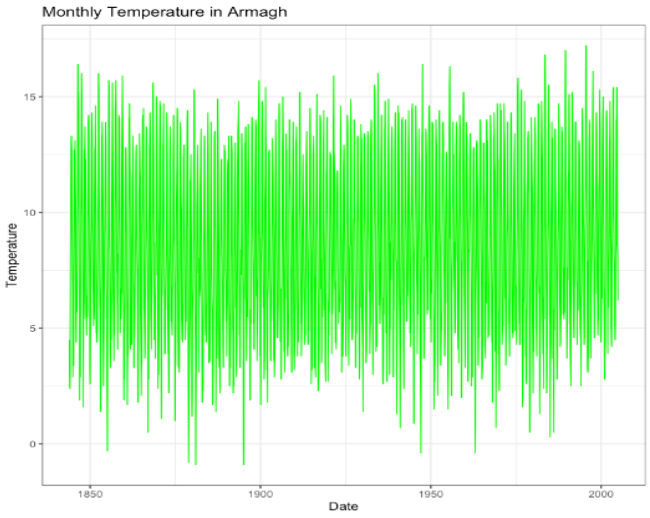

Fig. 2. Time Series Monthly Plot

DATA DECOMPOSITION:

Decomposition is done to explore more about the data. Mainly this contains 3 things. Trend, Seasonality, Residuals. The trend indicates there is a trend in one direction and it's mainly for the long term Seasonality is something that is limited to that season. For example, The temperature can be high in summer and low in winter season. The Residual means there is no fixed trend or pattern, it will just happen something like an earthquake or disaster. Decomposing can be done in two ways namely additive and multiplicative. Usually, when there is variability in the trend Multiplicative is used and when there is not much change in the trend, an additive method is used. If we look at Figure 4, we have plotted the monthly

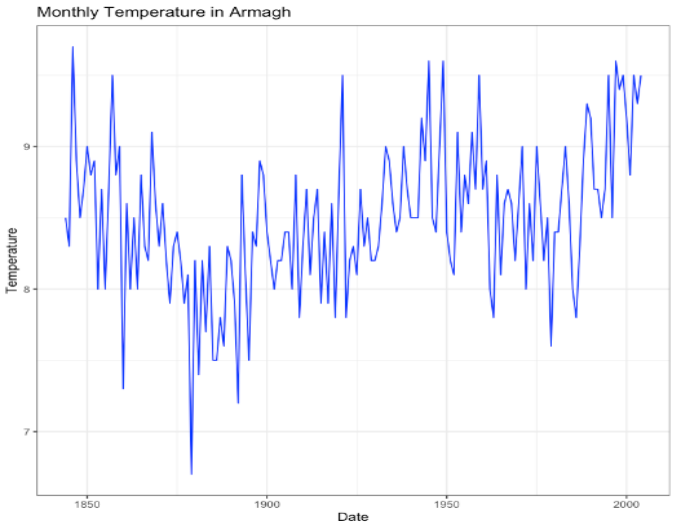

Fig. 3. Time Series Yearly Plot

decomposed data, there is not much variation in the trend except in the 1880s when the temperature was on the lower side and high near 1950 and 2000. Talking about seasonality there was none as we can see uniformity and there were residuals too.

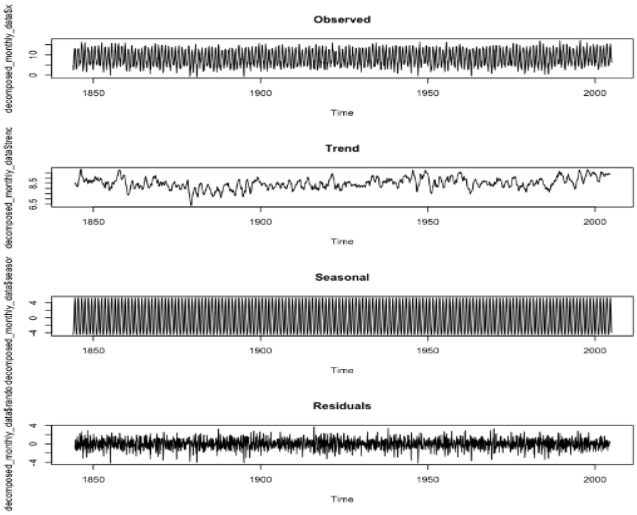

Fig. 4. Monthly Decomposed Data

III. MODEL BUILDING AND EVALUATION

Our objective in model building is to build models in Exponential Smoothing, ARIMA/SARIMA, and Simple time series models and choose the best-performing model both for monthly data and yearly data and then choose among those 3, which was the best or we can call it as an optimum model.

A. Exponential Smoothing

Exponential Smoothing is a powerful and simple technique to forecast the values based on historical data by giving more importance to recent data than older data. There are 3 types in it. Simple Exponential Smoothing, Holt's Exponential Smoothing, and Holt-Winters Exponential Smoothing.

1) Simple Exponential Smoothing: Simple Exponential Smoothing: This method is used when there is no trend or seasonality in the data. It works by assigning a weight to the most recent observation and using that weight to forecast future values. The formula for the same is:

F(t+1) = $\alpha * Y(t) + (1 - \alpha) * F(t)$

2) Holt's Exponential Smoothing: This method is used when there is a trend in the data, but no seasonality. It can be said as a continuation of simple exponential smoothing just that trend is added. it is also called as Double Exponential Smoothing. The formula for Holt's exponential smoothing is:

Level: L(t) = $\alpha * Y(t) + (1 - \alpha)(L(t-1) + T(t-1))$

Trend: T(t) = $\beta * (L(t) - L(t-1)) + (1 - \beta)T(t-1)$

Forecast: F(t+h) = L(t) + hT(t)

3) Holt-Winters Exponential Smoothing: It is also called as Triple Exponential Model, it can be said an extension of Double exponential smoothing as the seasonal component is also taken into consideration along with the trend.

Level: L(t) = $\alpha * (Y(t) - S(t-m)) + (1 - \alpha)(L(t-1) + T(t-1))$

Trend: T(t) = $\beta(L(t) - L(t-1)) + (1 - \beta)T(t-1)$

Seasonal: S(t) = (Y(t) - L(t)) + (1-)S(t-m)

Forecast: F(t+h) = L(t) + hT(t) + S(t+h-m)

So we have built all these 3 above models in our project each for month and year, but due to space issues in the report, we are only displaying the result which had the best output among these 3 for each month and year data. 1. Monthly Data:

```
ETS(A,A,A)

Call:
 ets(y = train_data)

  Smoothing parameters:
    alpha = 0.0267
    beta  = 1e-04
    gamma = 1e-04

  Initial states:
    l = 9.0553
    b = -0.0013
    s = -3.9238 -2.4576 0.5863 3.4298 5.2013 5.384
           4.0052 1.2916 -1.4396 -3.3028 -4.2866 -4.4879

  sigma:  1.2163

     AIC     AICc      BIC
15285.27 15285.59 15379.79

Training set error measures:
                     ME     RMSE       MAE  MPE MAPE      MASE      ACF1
Training set 0.01770418 1.211239 0.9465906 -Inf  Inf 0.7096361 0.1908923
```

Fig. 5. Result for Simple Exponential Smoothing Monthly data

For Monthly data, we have built all the above three models and we had an RMSE value of 1.211 and MAE value of 0.94 for simple exponential Smoothing, RMSE value of 2.51 for Double exponential smoothing and MAE value of 1.99, and RMSE value of 2.79 and MAE value of 1.99for Triple Exponential smoothing. So considering these values we felt

that Simple Exponential Smoothing was the best model among these and we already know that since it does not have any season and trend, it was always going to be simple exponential smoothing. 2. Yearly Data: For Yearly data, we have built all

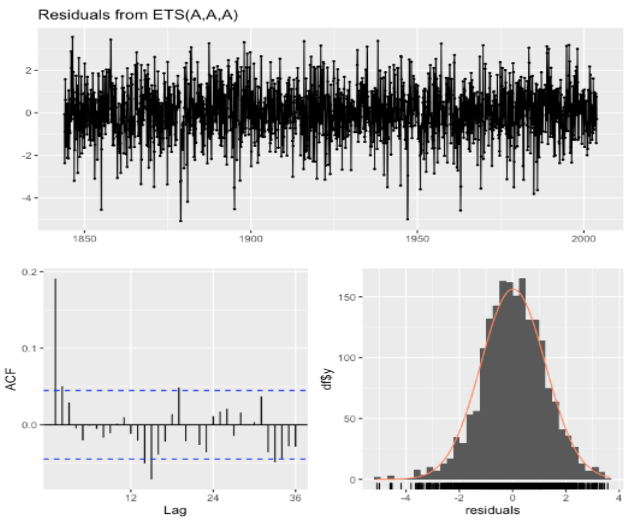

Fig. 6. Result Graph for Simple Exponential Smoothing Monthly data

```
#Exponential smoothing model
exponential_smoothing_model_yearly <- ets(train_data)
summary(exponential_smoothing_model_yearly)
```

```
ETS(A,N,N)

Call:
 ets(y = train_data)

  Smoothing parameters:
    alpha = 0.1682

  Initial states:
    l = 8.7104

  sigma:  0.4636

     AIC     AICc      BIC
570.0191 570.1729 579.2446

Training set error measures:
                     ME      RMSE       MAE        MPE     MAPE      MASE
Training set 0.01627885 0.4606912 0.3530613 -0.0935001 4.192907 0.7160299
                    ACF1
Training set -0.03098102
```

Fig. 7. Result for Simple Exponential Smoothing yearly data

the above three models and we had an RMSE value of 0.46, and MAE value of 0.35 for simple exponential Smoothing, an RMSE value of 0.51 for Double exponential smoothing, and an MAE value of 0.40. So considering these values we felt that Simple Exponential Smoothing was the best model among these and we already know that since it does not have any season and trend, it was always going to be simple exponential smoothing.

B. ARIMA/SARIMA

ARIMA (Autoregressive Integrated Moving Average) and SARIMA (Seasonal Autoregressive Integrated Moving Aver-

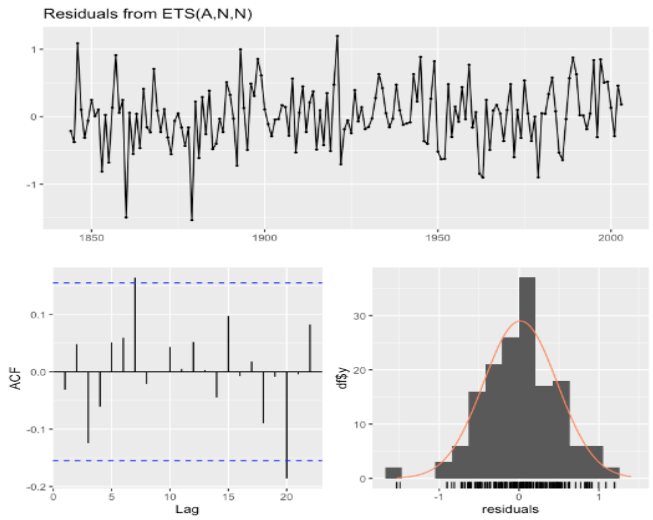

Fig. 8. Result Graph for Simple Exponential Smoothing Yearly data

```
#Fitting an ARIMA model to the training data for monthly data
library(forecast)
arima_model_monthly <- auto.arima(train_data,D=1)
summary(arima_model_monthly)
```

```
Series: train_data
ARIMA(1,0,2)(2,1,0)[12]

Coefficients:
         ar1      ma1      ma2     sar1     sar2
      0.4985  -0.2939  -0.0163  -0.6724  -0.3172
s.e.  0.1546   0.1558   0.0430   0.0218   0.0219

sigma^2 = 1.854:  log likelihood = -3296.89
AIC=6605.79   AICc=6605.83   BIC=6639.11

Training set error measures:
                     ME     RMSE      MAE  MPE MAPE      MASE          ACF1
Training set 0.007377913 1.355579 1.051322 -Inf  Inf 0.7881509 -0.0002581779
```

Fig. 9. Result of ARIMA Monthly data

age) are famous time series models that are used for forecasting. ARIMA is a model that is statistical in nature when a given time series data is a combination of three components: autoregression (AR), differencing (I), and moving average (MA). autoregression represents the relationship between the current value and previous values in the series, differencing represents the degree of change needed to make the series stationary, and MA represents the relationship between the current value and past error terms. SARIMA is an extension of ARIMA with seasonality which takes seasonal patterns into account in the data. It includes extra seasonal parameters that capture the seasonal component of the time series. The general equation of SARIMA is (p, d, q) x (P, D, Q)s, where p, d, q are the non-seasonal parameters, P, D, Q are the seasonal parameters, and s is the number of seasons in a year. So while coding [1], when we use auto.arima, automatically the

```
        Ljung-Box test

data:  Residuals from ETS(A,A,A)
Q* = 108.35, df = 24, p-value = 1.093e-12

Model df: 0.   Total lags used: 24
```

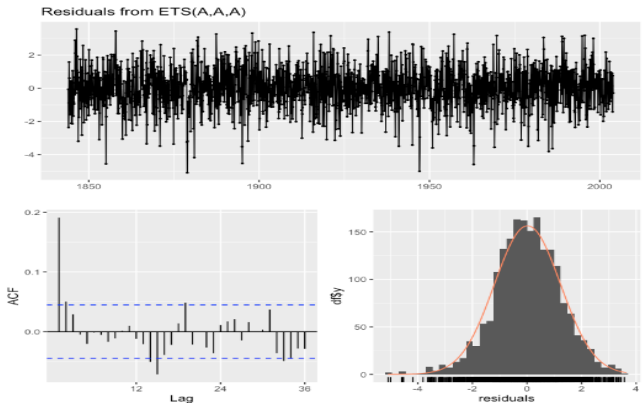

Fig. 10. Result graph of ARIMA Monthly data

best model which suits the data is taken. That is if there is a seasonality factor present then SARIMA gets picked else ARIMA gets picked. For monthly data, ARIMA got picked up and it had an RMSE value of 1.35 and MAE of 1.05. We

```
Series: train_data
ARIMA(0,1,1)

Coefficients:
          ma1
      -0.8274
s.e.   0.0412

sigma^2 = 0.2149:  log likelihood = -103.46
AIC=210.93   AICc=211   BIC=217.06

Training set error measures:
                     ME      RMSE       MAE         MPE     MAPE      MASE
Training set 0.01802921 0.4607088 0.3507608 -0.07201756 4.165348 0.7113643
                    ACF1
Training set -0.03212209
```

Fig. 11. Result of ARIMA Yearly data

had RMSE of 0.46 and MAE of 0.35 which is a good measure considering their low value for yearly data.

C. Simple time series models

Simple time series models help us to predict future values based on historical data, this is mainly used in forecasting, and not only that it's also used in finance, marketing, weather prediction, etc. We have many models in it, in this project we have used 3 models: 1. Naive Model: it is a basic simple time series model which assumes the future value equal to the last observed value and it does not consider the trend or seasonality. its mainly used to compare complex results. 2. Simple Average Method: As the name suggests in this model the future value will be calculated as the mean of the historical data. As it doesn't take trend, or seasonality into account, it is not suitable for complex time series problems. 3. Moving Average Model: it is similar to the simple average method but the difference is that only some historical value's average is taken, I mean there is a fixed number of terms. For example, to predict the next value, they take an average of the last 5 values and when then again when we need to predict the next value, the first value is left out and the last added value is taken for calculating the average since this number keeps moving, this is called moving average model.

```
                     ME      RMSE       MAE      MPE    MAPE      MASE
Training set 0.01770418 1.2112392 0.9465906     -Inf     Inf 0.7096361
Test set     0.17450272 0.7804156 0.6675100 1.792123 7.67026 0.5004161
                   ACF1 Theil's U
Training set  0.1908923        NA
Test set     -0.4164430 0.3343079

        Ljung-Box test

data:  Residuals
Q* = 17938, df = 24, p-value < 2.2e-16

Model df: 0.   Total lags used: 24
```

Fig. 12. Result of Moving average monthly data

1. Monthly data: We have built all the above three models to Monthly data and we had the following results: RMSE of 2.43, MAE of 2.01 for Naive Model, RMSE of 3.82, MAE of 3.34 for the Simple Average Method model, RMSE of 1.21, MAE of 0.94 for Moving average Model

Looking at the values "Moving average Model" suits best the data set.

2. Yearly data: We have built all the above three models to Yearly data and we had the following results: RMSE of 0.60, MAE of 0.49 for the Naive Model, RMSE of 0.59, MAE of 0.40 for the Simple Average Method model, RMSE of 0.46, MAE of 0.35 for Moving average Model

Looking at the values "Moving average Model" suits best the data set.

IV. Results

A. Montly Data

looking at all the results, I feel "simple exponential smoothing" is the best-fit model for monthly data as it had the RMSE value of 1.21 and MAE 0f 0.94 which was the lowest among other models. Forecasting the monthly values for 2004:

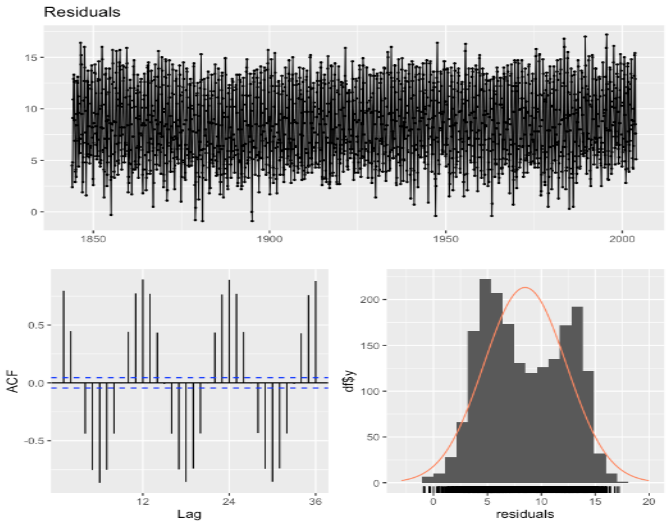

Fig. 13. Result graph of Moving average monthly data

```
                    ME      RMSE       MAE        MPE     MAPE      MASE
Training set 0.01627885 0.4606912 0.3530613 -0.0935001 4.192907 0.7160299
Test set     0.35143644 0.3514364 0.3514364  3.6993309 3.699331 0.7127346
                   ACF1
Training set -0.03098102
Test set             NA

        Ljung-Box test

data:  Residuals
Q* = 114.1, df = 10, p-value < 2.2e-16

Model df: 0.   Total lags used: 10
```

Fig. 14. Result of Moving average Yearly data

date	x
2004-01-01	5.1
2004-02-01	4.5
2004-03-01	6.2
2004-04-01	8.3
2004-05-01	10.7
2004-06-01	14.0
2004-07-01	13.6
2004-08-01	15.4
2004-09-01	13.2
2004-10-01	8.6
2004-11-01	8.2
2004-12-01	6.2

Fig. 15. Original 2004 monthly data

Jan 2004	4.826801
Feb 2004	5.029996
Mar 2004	6.014942
Apr 2004	7.882274
May 2004	10.613344
Jun 2004	13.328619
Jul 2004	14.709242
Aug 2004	14.530972
Sep 2004	12.760338
Oct 2004	9.918806
Nov 2004	6.877779
Dec 2004	5.412854

Fig. 16. Forcasted 2004 monthly data

B. Yearly Data

looking at all the results, I feel "simple exponential smoothing" is the best-fit model for Yearly data as well as it had the RMSE value of 0.46 and MAE of 0.35 which was the lowest among other models. Forecasting the Yearly value for 2004: If

date	x
2004-01-01	9.5

Fig. 17. Original 2004 Year data

	Point Forecast
2004	9.148564

Fig. 18. Forcasted 2004 Year data

we look at the original monthly values and forecasted values, there is not much difference as almost we have similar values with a difference of less than 1 unit which can clearly tell that our model is a good fit for this data set.

C. Yearly Data

So we have the original value as 9.5 and we have forecasted it as 9.14 which means our value is very close to the original which clearly indicates our model is a good fit for this data set.

V. CONCLUSION

On the basis of the Temperature data set for month and year, we have done the basic EDA and visualization and built models in different models of Exponential Smoothing, ARIMA/SARIMA, and Simple Time Series Models, and chose the best among those and finally for each month and year we had to choose the best model. After evaluating all the models, I feel Simple Exponential Smoothing is the best model for both the data sets as it had the lowest RMSE and MAE values, and also forecasted values were very near to actual values.

REFERENCES

[1] NCI Moodle: https://mymoodle.ncirl.ie/course/view.php?id=1593
[2] ExcelR Data Science course Bangalore
[3] W3 Schools: https://www.w3schools.com/r/
[4] Stack Overflow: https://stackoverflow.com/questions/31717850/error-package-or-namespace-load-failed-for-ggplot2-and-for-data-table
[5] Posit Item: https://community.rstudio.com/t/problems-with-loading-ggplot2/6250

PART-B: Logistic Regression on Diabetes data

***Abstract*—As people say, "Prevention is better than Cure". it would be very helpful for people if they have any disease and it could be found in starting stage itself as they can take proper medication to avoid worsening it. In this project, we took a Diabetes data set that contains details of blood samples of diabetic patients which had features like UREA, Creatinine Ratio, Average blood glucose, Cholesterol, Triglycerides, High-density lipoprotein, Low-density lipoprotein, Very-low-density lipoprotein cholesterol, Body-Mass-Index. We explored the data set and got more information about the data by performing Exploratory Data analysis using descriptive statistics and visualization. Then after performing the required transformation, we built the Logistic Regression model and evaluated the best model after rejecting intermediate models. Then using the final model we tested it with 'P' values and got the probability of 'prediabetic' cases being diagnosed as diabetic.**

***Index Terms*—Logistic Regression, Descriptive Statistics, Pre-diabetic, visualization**

VI. OBJECTIVE

Health Industry is a place where there is very less scope for error as it may cost a life. In situations where the patient is critical, the chances of survival become very minimal. So in order to avoid that we have Machine Learning algorithms that can predict the disease chances well in advance with the help of historical data related to different features of blood or body or anything which can be helpful and hence it will help in reducing the death rate. In our case, we are predicting Disabilities with the help of blood sample data and testing it with P values, and then verifying it with relevant assumptions and checking the performance.

VII. DEFINITION AND TERMINOLOGY

A. Descriptive Statistics

Descriptive Statistics is nothing but understanding the data set and understanding the business. So after importing the data set, we can try to understand it by looking at the summary, data-types and to know more about the distribution of the data, we can check to mean, median, mode and variance, standard deviance etc. So the main objective here is to explore the data and take insights from it. Also, we can plot the graphs like histograms, scattered plots, and other types of graphs which will help us to find the patterns in data and help us to understand better and we can transform the data as per our requirement for the model building purpose.

B. Logistic Regression

Logistic Regression is a type of model especially used in a situation where we need to predict YES/NO, 0/1, or binary values. when we need to find the dependent binary variable on the basis of one or more independent variables. So basically we will get the probability of binary output based on the independent variables. The Equation for Logistic Regression:

$\log(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k$

where:

p is the probability of a positive outcome

1-p is the probability of a negative outcome

$\beta\_0, \beta\_1, \beta\_2, ..., \beta\_k$

are the coefficients of independent variables

x1, x2, ..., xk are the values of independent variables

C. P value

P value is something used as an evaluation measure used in Hypothesis testing. A value less than 0.05 will be considered as acceptance of the Null hypothesis whereas a value more than 0.05 is considered as rejection of null hypothesis or acceptance of the alternative hypothesis. Usually in models, if the value of any of the variables is more than 0.05, that variable can be considered irrelevant for that model and can be removed.

D. AIC

AIC(Akaike Information Criterion) is a standard measure used to measure the model. It is a combination of how good the model fits and how complex the model is. So it provides a balance between complexity and fitting.

AIC = -2log(L) + 2k

where L is the likelihood of the data and k is the number of variables used in the model. If the AIC value is on the lower side then the model can be considered as good.

E. Classical assumptions

1. Linearity: The relationship between Log odds of the output variables should be linear with independent input variables.

2. Independence of observations: The observations must be independent of each other, there should not be any dependency between rows.

3. Absence of multicollinearity: There should not be high collinearity among input variables.

4. The absence of influential outliers: There should not be any outliers that will have an impact on the model results.

5. Large sample size: There should be enough rows so that the result should not be biased.

6. The presence of the dependent variable: there should be a proper dependent variable that is binary in nature.

VIII. DESCRIPTION OF DATA SET

We have a Diabetes patient dataset where we need to predict if a person has diabetes or not based on blood sample details. We have details like ID, No_Pation, Gender, AGE, Urea, Cr, HbA1c, Chol, TG, HDL, LDL, VLDL, BMI, and CLASS. In Figure 19 we can see these data. We have 1000 records for the same. Where CLASS is our target variable which has 'Y' and 'N 'values in it. We also have a 'P' value which means prediabetic, we need to check the probability of 'P' valued people having Diabetes using our final model.

IX. DESCRIPTIVE STATISTICS AND DATA VISUALIZATION

A. DATA UNDERSTANDING

In any project, the first and foremost thing is understanding the business and the data. So in my project after importing the

```
#understanding the structure
str(data)
```

```
tibble [947 x 14] (S3: tbl_df/tbl/data.frame)
 $ ID       : num [1:947] 502 420 680 634 721 759 636 788 82 132 ...
 $ No_Pation: num [1:947] 17975 47975 87656 34224 34225 ...
 $ Gender   : chr [1:947] "F" "F" "F" "F" ...
 $ AGE      : num [1:947] 50 50 50 45 50 32 31 33 30 45 ...
 $ Urea     : num [1:947] 4.7 4.7 4.7 2.3 2 3.6 4.4 3.3 3 4.6 ...
 $ Cr       : num [1:947] 46 46 46 24 50 28 55 53 42 54 ...
 $ HbA1c    : num [1:947] 4.9 4.9 4.9 4 4 4 4.2 4 4.1 5.1 ...
 $ Chol     : num [1:947] 4.2 4.2 4.2 2.9 3.6 3.8 3.6 4 4.9 4.2 ...
 $ TG       : num [1:947] 0.9 0.9 0.9 1 1.3 2 0.7 1.1 1.3 1.7 ...
 $ HDL      : num [1:947] 2.4 2.4 2.4 1 0.9 2.4 1.7 0.9 1.2 1.2 ...
 $ LDL      : num [1:947] 1.4 1.4 1.4 1.5 2.1 3.8 1.6 2.7 3.2 2.2 ...
 $ VLDL     : num [1:947] 0.5 0.5 0.5 0.4 0.6 1 0.3 1 0.5 0.8 ...
 $ BMI      : num [1:947] 24 24 24 21 24 24 23 21 22 23 ...
 $ CLASS    : chr [1:947] "N" "N" "N" "N" ...
```

Fig. 19. Structure of the Data set

data set, first I saw the structure of the data to understand what kind of data we have and what are types of the column, and then to get more clarity I printed the data using the head. Then I calculated the mean and standard deviation, max value, and min to know the variability in a data set, then also checked how many people have diabetes and how many are male, and how many are female., we can see this part in Figure 20.

```
In [8]: #Calculating minimum and maximum values for each variable
        print("Min values:")
        print(sapply(data, min))
        print("Max values:")
        print(sapply(data, max))
```

```
[1] "Min values:"
      ID No_Pation    Gender       AGE      Urea        Cr     HbA1c      Chol
     "1"     "123"      "F"      "20"     "0.5"       "6"     "0.9"       "0"
      TG       HDL       LDL      VLDL       BMI     CLASS
   "0.3"     "0.2"     "0.3"     "0.1"      "19"       "N"
[1] "Max values:"
      ID  No_Pation     Gender        AGE       Urea         Cr      HbA1c
   "800" "75435657"        "M"       "79"     "38.9"      "800"       "16"
    Chol         TG        HDL        LDL       VLDL        BMI      CLASS
  "10.3"     "13.8"      "9.9"      "9.9"       "35"    "47.75"        "Y"
```

```
In [9]: #Calculating the proportion of patients with diabetes and without diabetes
        table(data$CLASS)
```

```
  N   Y
103 844
```

```
In [10]: #Calculating the proportion of male and female patients
         table(data$Gender)
```

```
  F   M
418 529
```

Fig. 20. Exploring the data

Then also checked for missing and duplicate values and there were no such values found. Then I checked for a correlation between all the variables by correlation matrix and we could see that there is no collinearity between independent variables.

B. DATA VISUALIZATION

Visualization helps us to understand the data and get more information from it and we will also get many patterns in

```
#Correlation matrix of the numeric variables
cor(data[,c("AGE","Urea","Cr","HbA1c","Chol","TG","HDL","LDL","VLDL","BMI")])
```

A matrix: 10 × 10 of type dbl

	AGE	Urea	Cr	HbA1c	Chol	TG	HDL	LDL	VLDL	BMI
AGE	1.00000000	0.104074392	0.063636176	0.348310526	0.035583160	0.15112148	-0.02459755	0.023038081	-0.10975079	0.331684155
Urea	0.10407439	1.000000000	0.618130651	-0.035406493	-0.002574753	0.04343017	-0.04496733	-0.010278860	-0.01375536	0.038650550
Cr	0.06363618	0.618130651	1.000000000	-0.041327462	-0.007199211	0.05850913	-0.02809868	0.041526852	0.00927147	0.058706778
HbA1c	0.34831053	-0.035406493	-0.041327462	1.000000000	0.174766955	0.21996658	0.02389112	0.006721897	0.06321662	0.382033838
Chol	0.03558316	-0.002574753	-0.007199211	0.174766955	1.000000000	0.31999117	0.10340618	0.409993971	0.07310925	0.004481209
TG	0.15112148	0.043430170	0.058509125	0.219966581	0.319991166	1.00000000	-0.08183415	0.016560476	0.14009319	0.104836813
HDL	-0.02459755	-0.044967332	-0.028098685	0.023891122	0.103406179	-0.08183415	1.00000000	-0.148107076	-0.06097413	0.066828954
LDL	0.02303808	-0.010278860	0.041526852	0.006721897	0.409993971	0.01656048	-0.14810708	1.000000000	0.06217126	-0.078512985
VLDL	-0.10975079	-0.013755363	0.009271470	0.063216620	0.073109248	0.14009319	-0.06097413	0.062171258	1.00000000	0.191174403
BMI	0.33168415	0.038650550	0.058706778	0.382033838	0.004481209	0.10483681	0.06682895	-0.078512985	0.19117440	1.000000000

Fig. 21. Correlation Matrix

the data and also an easy technique to explain the data to someone. In Figure 22 we can see a boxplot which will help us to understand the data variability. We could observe that almost all variable's median is similar for both the patients and nonpatients but the range of values for patients with diabetes is large and which suggests that there is more variability. In HbA1c the median difference is comparatively large.

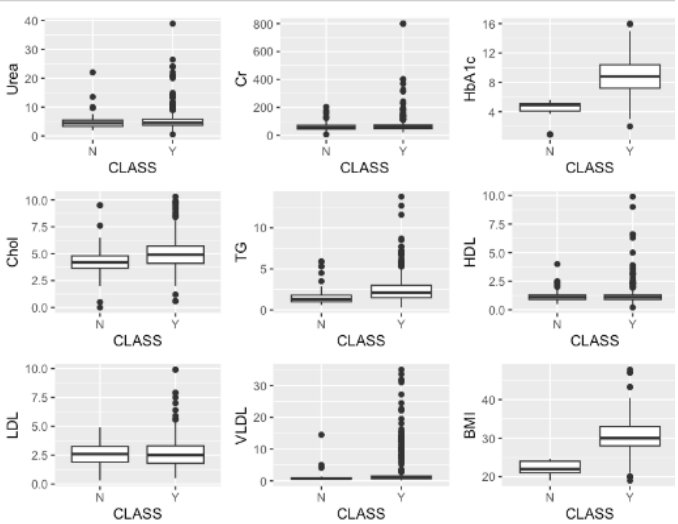

Fig. 22. Boxplot

In Figure 5, we could see a histogram plotted to check how the data is distributed, and from the figure, we could tell that Age, HbA1c, Chol, and BMI are relatively normally distributed and TG, CR, LDL, VLDL, HDL are left skewed.

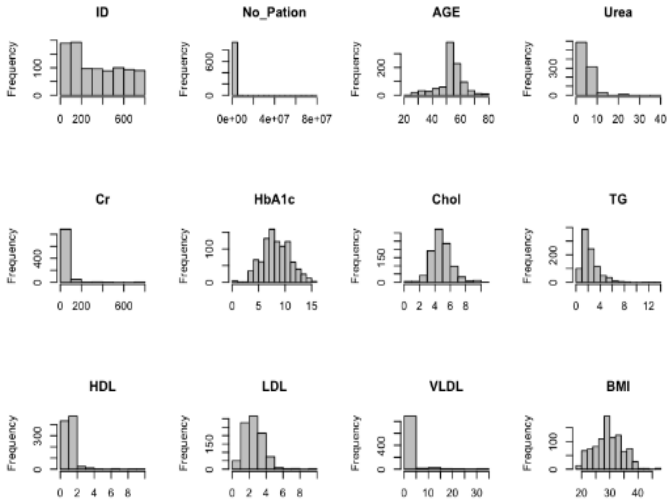

Fig. 23. Histogram

We could say that the scatter plot is a continuation of the correlation matrix or we can say it as a more clear correlation matrix. because in this we will get to see the graph along with values which makes things easier for us while understanding the data as we can also see both visually and we can see the values as well. By scatter plot, we can also check if there is any relationship between variables and collinearity between the variables.

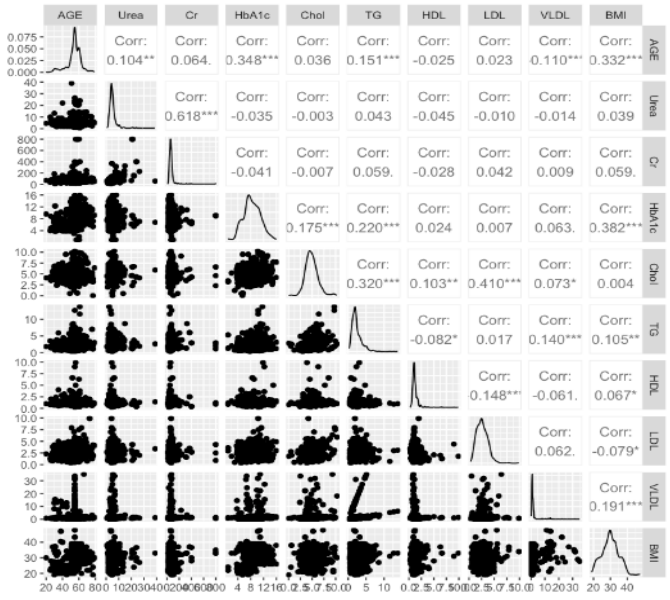

Fig. 24. Scatter plot

X. Model Building and Evaluation

A. Models

Model Building is the most important step in the analysis as this is the place where we build the model and based on the result of this we can predict the value, in this section we will explain how we came up with a final model.

1) Model 1: Model with all the input variables: So after doing all the exploratory data analysis, visualization, and transformation, we are ready for model building. Initially, we took all the input variables to build the model. So looking at

```
Call:
glm(formula = CLASS ~ ., family = binomial, data = train)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.17763   0.00000   0.00004   0.00111   2.38636

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -58.46881   12.89801  -4.533 5.81e-06 ***
Gender        0.91595    0.82718   1.107 0.268157
AGE          -0.02740    0.04632  -0.591 0.554252
Urea         -0.07331    0.17433  -0.421 0.674086
Cr            0.01815    0.01976   0.919 0.358233
HbA1c         1.97971    0.49932   3.965 7.34e-05 ***
Chol          0.95441    0.31582   3.022 0.002511 **
TG            2.46335    0.73033   3.373 0.000744 ***
HDL           0.96917    0.63600   1.524 0.127545
LDL           0.83390    0.46489   1.794 0.072852 .
VLDL         -0.07330    0.32053  -0.229 0.819113
BMI           1.52252    0.35516   4.287 1.81e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 425.112  on 661  degrees of freedom
Residual deviance:  51.908  on 650  degrees of freedom
AIC: 75.908
```

Fig. 25. Model1 Output

the result, we can easily eliminate many variables as the P value is more than 0.05 for Gender, Age, UREA, CR, VLDL, HDL, and LDL. So we remove these values as they don't contribute anything to the model and if you look at the stars which are located next to the variables indicate the significance level anyways these variables didn't have any stars which confirms again that they are of not much significance and hence we need to build the new model.

2) Model 2: Model with only significant variables: So after rejecting the first model, we built this model with only significant values. So looking at the output, we have all the variables with P values less than 5 which indicates all the variables are contributing to the model and if we look at the Null deviance and Residual error they are around 425 and 59 respectively and AIC is 69 which is a good measure to evaluate the model as it not only checks if the model fit but also check the number of variables. So with these things in mind, we can try to improve this model by transforming the variables.

3) Model 3: Model with Transformed Variables: So in order to improve the last model, we did some transformations on the variables. So we applied log for Chol and applied sqrt for BMI and we built the model. Looking at the result, the Residual deviance value has been improved to 57, and also AIC value is now improved to 69 which indicates that this is a better model than the previous model.

```r
#Building a logistic regression model on training data after eliminating unimportant variables
model2 <- glm(CLASS ~ HbA1c + Chol + TG + BMI, family = binomial, data = train)
summary(model2)
```

```
Warning message:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"
```

```
Call:
glm(formula = CLASS ~ HbA1c + Chol + TG + BMI, family = binomial,
    data = train)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.85935   0.00000   0.00016   0.00196   2.45512

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -46.5872     9.4030  -4.955 7.25e-07 ***
HbA1c         1.7600     0.3959   4.446 8.77e-06 ***
Chol          0.8847     0.2675   3.308 0.000941 ***
TG            2.0073     0.5669   3.541 0.000399 ***
BMI           1.2678     0.2902   4.369 1.25e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 425.112  on 661  degrees of freedom
Residual deviance:  59.066  on 657  degrees of freedom
AIC: 69.066

Number of Fisher Scoring iterations: 11
```

Fig. 26. Model2 Output

```
Call:
glm(formula = CLASS ~ HbA1c + log(Chol) + TG + sqrt(BMI), family = binomial,
    data = train)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.75932   0.00000   0.00008   0.00130   2.72891

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -89.0691    19.2293  -4.632 3.62e-06 ***
HbA1c         1.8694     0.4211   4.440 9.01e-06 ***
log(Chol)     4.8247     1.3940   3.461 0.000538 ***
TG            2.0905     0.6178   3.384 0.000715 ***
sqrt(BMI)    14.1416     3.1782   4.450 8.60e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 425.112  on 661  degrees of freedom
Residual deviance:  57.173  on 657  degrees of freedom
AIC: 67.173

Number of Fisher Scoring iterations: 11
```

Fig. 27. Model3 Output

B. Result Evaluation

Evaluation is a part where we will get to know if our built model is really the best one and how it fares against the test data. So We have used the below evaluation metrics to evaluate. 1. Confusion Matrix: it is a table that has True Positive, True Negative, False Positive, and False Negative values based on the model. 2. Precision: proportion of True Positive among all the positive results.

3. Accuracy: proportion of correct values among all the predicted values.

4. F1 Score: The harmonic mean of precision and recall.

5. AUC-ROC: it is a measure that tells how much area is under the Receiver Operating Characteristic (ROC) curve.

```r
conf_matrix <- table(test$CLASS, pred_class)
conf_matrix
```

```
   pred_class
      0   1
  0  32   6
  1   5 242
```

```r
#Accuracy
accuracy <- sum(diag(conf_matrix))/sum(conf_matrix)
accuracy
```

```
0.96140350877193
```

```r
#Precision
precision <- conf_matrix[2,2]/sum(conf_matrix[,2])
precision
```

```
0.975806451612903
```

```r
#Recall
recall <- conf_matrix[2,2]/sum(conf_matrix[2,])
recall
```

```
0.979757085020243
```

```r
#F1 Score
f1 <- 2 * precision * recall / (precision + recall)
f1
```

```
0.977777777777778
```

Fig. 28. Logistic Regression Results

So we are evaluating our final model using the above matrices, looking at the result, in the confusion matrix we could see values are more for True Positive, and True Negative values are more in number which indicates it's a good model. we have high Accuracy, precision, recall, F1 score, AUC-ROC with all the matrices are in the range of 96-98. Which definitely can say that our model is a good model.

XI. Result and Verifying the Assumptions

1) Result: Our objective was to build the model and check the probability of patients having Diabetes for the data which had 'P' Value.

```r
#Testing the final model on the 'Diabetes==P' cases
predicted_values_p <- predict(model3, newdata = diabetes_data_P, type = "response")
prob_diabetic <- mean(predicted_values_p > 0.5)
cat("Probability of diagnosing the 'P' class cases as diabetic: ", prob_diabetic, "\n")
```

```
Probability of diagnosing the 'P' class cases as diabetic:  0.7735849
```

Fig. 29. Final Result

Probability of diagnosing the 'P' class cases as diabetic: 0.7735849

2) Verifying the Assumptions: a. Linearity: in our case, the relationship between the input variables and the output variable is modeled as a straight line in the log-odds space and hence our first assumption is true.

b. Multicollinearity: Since all the values are below 5 we can

```
#Checking for multicollinearity
vif_values <- vif(model3)
vif_data <- data.frame(variable = names(vif_values), vif = vif_values)
print(vif_data)
```

```
              variable      vif
HbA1c             HbA1c 2.219582
log(Chol) log(Chol) 2.108695
TG                   TG 1.733639
sqrt(BMI) sqrt(BMI) 4.171760
```

Fig. 30. Multicolliarity assumption

safely say that there is no collinearity in our model and our second assumption is also met.

c. Influential Variables: Looking at the graph we can say

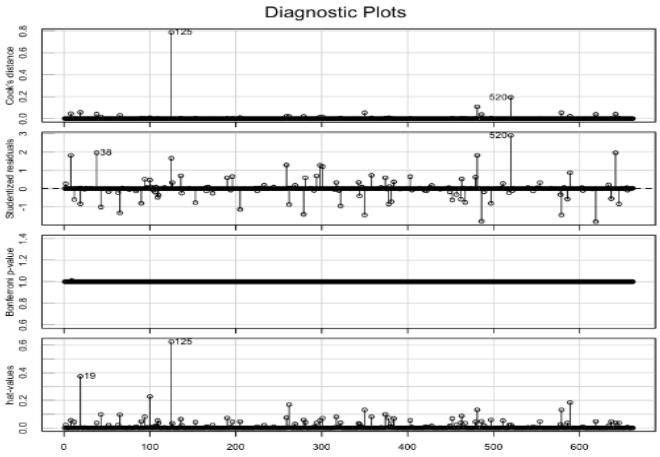

Fig. 31. Multicolliarity assumption

that we don't have any outliers or influential variables which are impacting the result and hence we can say our assumption is met.

d. Normality: In any model, data should be normal and that is the ideal case. in our case, we have performed the Shapiro-walk normality test and we have got p value less than 0.05 which indicates that the null hypothesis is true and which means the Normality assumption is met.

XII. Conclusion

Based on the given data after performing exploratory data analysis and visualizing the data, we have built the best fit for the Logistic Regression model rejecting intermediate models which met all the assumptions when tested we got good accuracy, precision, and F1 scores and also we calculated the Probability of diagnosing the 'P' class cases as diabetic and we got it as 0.7735849. Overall we have met our objective.

```
#Checking for normality of residuals
residuals <- residuals(model3, type = "pearson")
qqnorm(residuals)
qqline(residuals)
shapiro.test(residuals)
```

```
        Shapiro-Wilk normality test

data:  residuals
W = 0.26197, p-value < 2.2e-16
```

Fig. 32. Normality Test

References

[1] NCI Moodle https://mymoodle.ncirl.ie/course/view.php?id=1593
[2] ExcelR Data Science course Bangalore
[3] W3 Schools https://www.w3schools.com/r/