PREDICTION OF PCOS BASED ON PHYSICAL AND PHYSIOLOGICAL FACTORS.

Radhika Gedela, Harshitha Kandi, Lakshmi Aparna Valiveti, Kavya Yalavarthi, and Corey May

Indiana University Purdue University Indianapolis, IN 46202, USA
rgedela@iu.edu, hkandi@iu.edu, lavaliv@iu.edu, kyalavar@iu.edu, coremay@iu.edu

Abstract. This project utilizes quantitative research and a retrospective observational study to analyze various physical and physiological attributes in order to develop a machine-learning model for predicting PCOS. The objective is to identify patterns and associations between these attributes and the presence or absence of PCOS to improve diagnostic accuracy. Through data analysis, several risk factors were identified that can help predict the likelihood of a woman having PCOS, including lifestyle and hormonal factors. These findings can be used to develop targeted interventions to reduce the incidence and impact of PCOS.
Employing statistical methodologies in quantitative research to analyze the PCOS dataset aims to identify actionable insights that can enhance the health and quality of life of women with PCOS. By understanding the risk factors and underlying causes of PCOS, healthcare professionals, and researchers can develop more targeted and effective interventions that can reduce the incidence and impact of PCOS. Ultimately, the machine learning model developed in this project can assist in predicting PCOS, improving the accuracy and efficiency of PCOS diagnosis.

Keywords: PCOS, · Machine learning, · methodology, · diagnostic accuracy.

1 Scope of the project:

1.1 Introduction:

Polycystic Ovary Syndrome (PCOS) is a hormonal disorder that affects women of reproductive age. It is characterized by the presence of multiple cysts on the ovaries, which can cause irregular menstrual periods, excessive hair growth, acne, and fertility problems. PCOS is also associated with insulin resistance, which can lead to type 2 diabetes and other health problems. The exact cause of PCOS is not known, but it is believed to be a combination of genetic and environmental factors. PCOS is a common condition, affecting approximately

1 in 10 women worldwide. Although there is no cure for PCOS, its symptoms can be managed with lifestyle changes, medication, and other therapies. Early diagnosis and treatment of PCOS can help prevent long-term complications and improve the quality of life for affected women.

1.2 Aim:

We aim "To analyze the PCOS (Polycystic Ovary Syndrome) dataset using statistical methodologies in quantitative research to identify connections with the patient's physical and physiological characteristics that would result in a higher or lower probability of determining PCOS".

Research questions:
a] Does any relationship exist between the attributes like AMH, PRG, TSH, FSH, LH, Vit D, Random Blood Sugar, Size of the endometrium, follicle size, hip: waist ratio, Menstrual cycle length, Weight gain, hair loss, skin darkening, acne, exercise to PCOS?
b] Do the factors like AMH, PRG, TSH, FSH, LH, Vit D, Random Blood Sugar, Size of the endometrium, follicle size, hip: waist ratio, Menstrual cycle length, Weight gain, hair loss, skin darkening, pimples, exercise help in predicting PCOS?

Hypothesis:

Null Hypothesis: There is no relationship between physical and physiological factors to PCOS which means that we cannot predict PCOS based on these attributes.
Alternative hypothesis: There is a relationship between physical and physiological factors to PCOS which means that We can predict PCOS based on these attributes.

1.3 Purpose:

The purpose of our study is to comprehend the connection between hormonal levels such as AMH, FSH, LH, TSH, and Vit D3. Physical changes such as endometrium thickness, follicle size, waist: hip, menstrual cycle, weight gain, hair loss, skin darkening, and pimples correlate to polycystic ovarian syndrome (PCOS).
We aim to uncover a correlation between these morphological and physiological characteristics so that we may design future predicting models to aid with preventive copies for future development.

2 METHODOLOGY:

Type of Study: a. We performed descriptive and inferential studies using quantitative methods to analyze the data.

b. We performed predictive analysis to develop models that can predict the occurrence of PCOS based on different attributes in the dataset.
c. Our study is a retrospective observational study. We investigated the relationship between PCOS and other attributes in the dataset.

2.1 Data Analysis and Management:

MySQL, Python, Pandas, Matplotlib, Scipy, numpy, seaborn, sklearn.

2.2 Steps involved:

1. Data Extraction
2. Data Cleaning
3. Data Analysis
4. Data Exploration and Visual
5. Model building and Evaluation

2.3 Team Members and their Responsibilities:

At the outset of the project, we had a team comprising individuals with diverse skills and backgrounds. We agreed upon the specific roles and responsibilities that each team member would undertake.

Table 1. Team members and responsibilities.

Team Member	Responsibilities	Background
Radhika Gedela	Quantitative analysis (Inferential Statistics) Machine learning Models Final Project Presentation	Doctor of Pharmacy
Harshitha Kandi	Descriptive Statistics Designing Basic schema plan Preparing Final Project Presentation	Bachelor of Dental Surgery
Lakshmi Aparna Valiveti	Planning factors to be compared and Performance Studies (Correlational and Regression analysis) Project Proposal Final Project Presentation	Doctor of Pharmacy
Kavya Yalavarthi	Performing Data extraction using SQL Performing Normality Tests	Bachelor of Dental Surgery
Corey May	Project Proposal Data Visualization using python Final Project Presentation	Management Analyst

2.4 Project Challenges:

Although our team worked well together, we faced some challenges during our project. Firstly, as most of us were from non-technical backgrounds, we initially struggled with assigning tasks and deciding who would handle which aspect of the programming. Another challenge was coordinating with our online team member who had different time constraints due to being a full-time employee with family responsibilities, causing some delays in completing tasks. Communication was also sometimes difficult due to different expectations between online and in-person team members.
We also faced technical challenges such as importing all our data as varchar data type, causing a slow-down in our workflow when we had to convert them to their respective data types. We had to decide whether to handle outliers or null values first during data cleaning, and after several trials, we concluded that handling null values first was more efficient.
In data cleaning, we had to decide whether to handle outliers or null values first. After performing trials in both cases, we concluded that handling null values first was more effective.
Lastly, we faced confusion about feature selection and engineering, as well as data normalization and standardization during the draft proposal stage, due to varying information from different sources. Despite these challenges, we learned valuable lessons and completed the project successfully.

3 Data Collection:

3.1 Data Source:

We have collected our dataset from Kaggle. KAGGLE PCOS-DATASET
https://www.kaggle.com/datasets/prasadbobby/pcosdata

4 Data extraction and storage:

4.1 Data Importation:

Our team was granted shared access to phpMyAdmin by the professor, and we utilized this database for importing the dataset. We began by generating a table called 'Patients_PCOS' and then altered the data types to integers and floats in accordance with the values.(Fig.1)

4.2 Data Extraction and storage:

It involved importing data, which was then loaded into a MySQL database, connected with Jupyter, and saved in a Pandas Data Frame

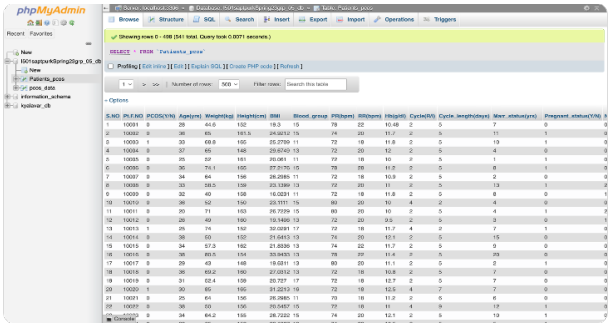


Fig. 1. structure of the table "Patients_PCOS"

5 Data description:

The dataset comprises patient data on Polycystic ovary syndrome with 42 attributes and 541 rows which includes binary and numerical data.

The 42 attributes were: PCOS(Y/N), Age, Pulse rate, Respiratory rate, Menstrual cycle length, marriage status, number of abortions, BP Systolic, BP diastolic, Follicle number left and right, Weight, height, BMI, Hemoglobin, PRL, Vitamin D3, AMH, FSH, LH, TSH, PRG, RBS, BP Systolic, BP diastolic, Average Follicle size right and left, Insulin levels, endometrium thickness, Waist, Hip, and waist: hip, Blood groups, Cycle(R/I), Pregnant status, Weight gain, Hair growth, Skin darkening, Hair loss, Pimples, Fast food, Regular exercise.

At the beginning of the research study, we decided to drop the attributes Serial number and Patient file number from our data frame, as they were not essential for our analysis. We named our data frame as "df" after removing these attributes.

To understand our data better, we checked the shape of the data frame, the data types of each attribute, and identified the unique values present in each column. We also checked for the value counts of our dependent variable.

After analyzing the data, we separated our attributes into two categories - categorical and numerical. We created two lists called "cat_cols" and "num_cols" to store our categorical and numerical columns, respectively.

6 Data Cleaning:

During the data cleaning process, we carefully examined the selected attributes as our dataset contained a small number of null and negative values. This step was crucial to ensure the accuracy and reliability of our analysis.

6.1 Identifying Null values:

During the data cleaning process, the "isnull()" function was used to identify null values in our dataset. It was identified that there were null values in the 'Marr_status(yrs)', 'Fast_food(Y/N)', and 'AMH' attributes(Fig.2).

```
# Check for null values in the dataset
print(df.isnull().sum())
```

```
PCOS(Y/N)                0
Age(yrs)                 0
Weight(kg)               0
Height(cm)               0
BMI                      0
Blood_group              0
PR(bpm)                  0
RR(bpm)                  0
Hb(g/dl)                 0
Cycle(R/I)               0
Cycle_length(days)       0
Marr_status(yrs)         1
Pregnant_status(Y/N)     0
No_of_abortions          0
FSH                      0
LH                       0
FSH/LH                   0
Hip(inch)                0
Waist(inch)              0
Waist:Hip                0
TSH                      0
AMH                      1
PRL                      0
Vit_D3                   0
PRG                      0
RBS                      0
Wt_gain(Y/N)             0
Hair_growth(Y/N)         0
Skin_darkening(Y/N)      0
Hair_loss(Y/N)           0
Pimples(Y/N)             0
Fast_food(Y/N)           1
Reg_exercise(Y/N)        0
BP_systolic              0
```

Fig. 2. Identifying Null values

6.2 Replacing the null values:

To address the null values, we replaced them with the median for numerical data which are 'AMH' and 'Marr_status(yrs)', and the mode for a categorical attribute which is 'Fast_food(Y/N)(Fig.3)'.

We also double-checked to ensure that there were no remaining null values present in the data.

REPLACING THE NULL VALUES WITH MODE FOR CATEGORRICAL VARIABLES

```
df['Fast_food(Y/N)'].fillna(df['Fast_food(Y/N)'].mode()[0], inplace=True)
```

```
mode_value = df['Fast_food(Y/N)'].mode()
print(mode_value)
```

```
0    1.
Name: Fast_food(Y/N), dtype: object
```

REPLACING THE NULL VALUES WITH MEDIAN FOR CONTINUOUS VARIABLES

```
df['Marr_status(yrs)'].fillna(df['Marr_status(yrs)'].mode()[0], inplace=True)
```

```
median_value = df['Marr_status(yrs)'].median()
print(median_value)
```

```
7.0
```

```
df['AMH'].fillna(df['AMH'].median(), inplace=True)
```

```
median_value = df['AMH'].median()
print(median_value)
```

```
3.7
```

Fig. 3. Replacing the null values

6.3 Changing the Data types of the attributes which had null values:

After replacing the null values, we changed the data types of the respective attributes into int and float for 'Fast_food(Y/N)', 'Marr_status(yrs)', and 'AMH' to ensure consistency in the dataset(Fig.4).

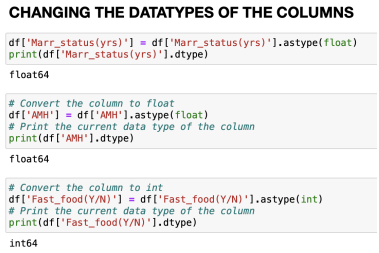


Fig. 4. Changing the Data types

6.4 Identifying and handling the outliers:

By using the Inter quartile range (IQR) method we identified outliers.

The .clip() method was used to limit the outliers in the data frame to their respective upper and lower bounds, which were determined using the Interquartile Range method. We later ensured that all the outliers were capped (Fig.5).

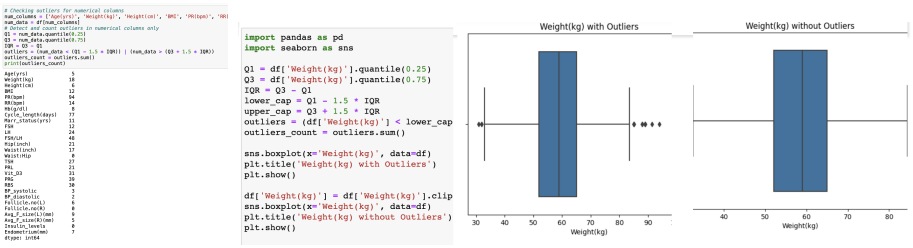


Fig. 5. Identifying, visualizing and handling the outliers

7 Data Analysis:

Python was used for data analysis in which linear regression, Mann-Whitney U test, Chi-Square independence test, spearman's correlation were employed to

analyze the data. We implemented a predictive model building using different machine learning models like logistic regression, decision tree, Naïve Bayes, and support vector Machine.

Below are some of the steps we took to effectively analyze the data:

7.1 Descriptive Statistics:

Descriptive statistics were done for the numerical columns, which helped to reveal important features of the data such as its count, mean, standard deviation, minimum and maximum values, and quartiles (Fig.6).

```
num_cols = ['Age(yrs)', 'Weight(kg)', 'Height(cm)', 'BMI', 'PR(bpm)', 'RR(bpm)', 'Hb(g/dl)', 'Cycle_length(days)', '
df[num_cols].describe()
```

	Age(yrs)	Weight(kg)	Height(cm)	BMI	PR(bpm)	RR(bpm)	Hb(g/dl)	Cycle_length(days)	Marr_status(yrs)	No_of_abortions	...	P
count	541.000000	541.000000	541.000000	541.000000	541.000000	541.000000	541.000000	541.000000	541.000000	541.000000	...	541.000
mean	31.416821	59.481331	156.466906	24.275733	73.146026	19.199630	11.153198	4.847505	7.572458	0.288355	...	0.375
std	5.372470	10.543763	5.945107	3.915366	2.079647	1.544668	0.842803	1.065163	4.474102	0.692575	...	0.158
min	20.000000	32.500000	140.000000	14.150750	69.000000	16.000000	8.700000	2.500000	0.000000	0.000000	...	0.047
25%	28.000000	52.000000	152.000000	21.641300	72.000000	18.000000	10.500000	4.000000	4.000000	0.000000	...	0.250
50%	31.000000	59.000000	156.000000	24.238200	72.000000	18.000000	11.000000	5.000000	7.000000	0.000000	...	0.320
75%	35.000000	65.000000	160.000000	26.635000	74.000000	20.000000	11.700000	5.000000	10.000000	0.000000	...	0.450
max	45.500000	84.500000	172.000000	34.125550	77.000000	23.000000	13.500000	6.500000	19.000000	5.000000	...	0.750

8 rows × 30 columns

Fig. 6. Descriptive Statistics

7.2 Normality test:

We performed the Shapiro-wilk test to know the distribution of the data whether it is normally distributed or not normally distributed before proceeding to parametric or non-parametric tests(Fig.7). We visualized the distribution of independent variables with the dependent variable through histograms and KDE plots.

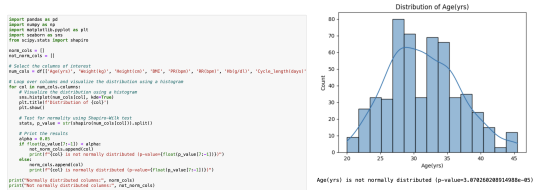


Fig. 7. Normality test

Similarly, we performed the normality test for other attributes also.

7.3 Hypothesis testing:

MannWhitney U test:
The Mann-Whitney U test was performed between numerical discrete independent variables and target variable (PCOS(Y/N) (Fig.8).

```
from scipy.stats import mannwhitneyu

pcos = df["PCOS(Y/N)"]
cols = [ 'Age(yrs)','PR(bpm)', 'RR(bpm)', 'Cycle_length(days)', 'Marr_status(yrs)', 'No_of_abortions', 'Hip(inch)',
         'Avg_F_size(L)(mm)', 'Avg_F_size(R)(mm)',  'Endometrium(mm)']

for col in cols:
    data = df[col]
    statistic, p_value = mannwhitneyu(data[pcos == 0], data[pcos == 1])

    if p_value < 0.05:
        print(f"Mann-Whitney U test comparing PCOS(Y/N) and {col}: \nStatistic = {statistic:.4f}, p-value = {p_value
    else:
        print(f"Mann-Whitney U test comparing PCOS(Y/N) and {col}: \nStatistic = {statistic:.4f}, p-value = {p_value

    if p_value < 0.05:
        print(f"There is a significant difference between PCOS and {col}\n")
    else:
        print(f"There is no significant difference between PCOS and {col}\n")
```

Fig. 8. Mann-Whitney U test

Linear regression:
Linear regression is performed between numerical continuous variables and predicting variable (PCOS Y/N) (Fig.9)

```
import pandas as pd
import statsmodels.api as sm

# Select numerical columns and target variable
cont_cols = [ 'Weight(kg)', 'Height(cm)', 'BMI', 'Hb(g/dl)', 'FSH', 'LH', 'FSH/LH', 'Waist:Hip', 'TSH', 'AMH', 'PRL'
          'RBS',  'Insulin_levels']
target_col = 'PCOS(Y/N)'

# Split data into features (X) and target (y)
X = df[cont_cols]
y = df[target_col]

# Add intercept to X
X = sm.add_constant(X)

# Fit linear regression model
model = sm.OLS(y, X).fit()

# Print regression results
print(model.summary())


# Loop through the coefficients and corresponding p-values
rejected_features = []
accepted_features = []
for i, p_value in enumerate(model.pvalues[1:]): # excluding intercept
    if p_value >= 0.05:
        rejected_features.append(cont_cols[i])
    else:
        accepted_features.append(cont_cols[i])

# Print the rejected and accepted features
print("Rejected features:", rejected_features)
print("Accepted features:", accepted_features)
```

```
                            OLS Regression Results
```

Fig. 9. Linear regression

Chi-Square Independence test:
The Chi-Square test is performed between independent categorical variables and predictive variable(Fig.10).

```
import pandas as pd
from scipy.stats import chi2_contingency, chi2

# Separate categorical and numerical columns
cat_cols = ['Blood_group', 'Cycle(R/I)', 'Pregnant_status(Y/N)', 'Wt_gain(Y/N)', 'Hair_growth(Y/N)', 'Skin_darkening
cat_data = df[cat_cols]

# Create a dictionary to store the p-values
p_values = {}

# Set the level of significance
alpha = 0.05

# Loop through the categorical columns and perform chi-square test
for col in cat_cols:
    contingency_table = pd.crosstab(cat_data[col], df['PCOS(Y/N)'])
    chi_square_statistic, p_value, ddof, expected = chi2_contingency(contingency_table)
    p_values[col] = p_value
    critical_value = chi2.ppf(q=1-alpha, df=ddof)
    print(f"\nChi-square test for {col}:")
    print(f"Chi-square statistic: {chi_square_statistic:.4f}")
    print(f"p-value: {p_value: }")
    print(f"Degree of freedom: {ddof}")
    print(f"Critical value: {critical_value:.4f}")

    if chi_square_statistic >= critical_value:
        print("Reject H0, there is a relationship between PCOS(Y/N) categorical variables",col)
    else:
        print("Fail to reject H0, there is no relationship between PCOS(Y/N) categorical variables", col)
    if p_value <= alpha:
        print("Reject H0 because there is a relationship between PCOS(Y/N) categorical variables", col)
    else:
        print("Fail to reject H0 because there is no relationship between PCOS(Y/N) categorical variables", col)
```

Fig. 10. Chi-Square Independence test

8 Correlation studies:

8.1 Correlation matrix for numerical attributes:

Spearman correlation for numerical columns (Fig.11)

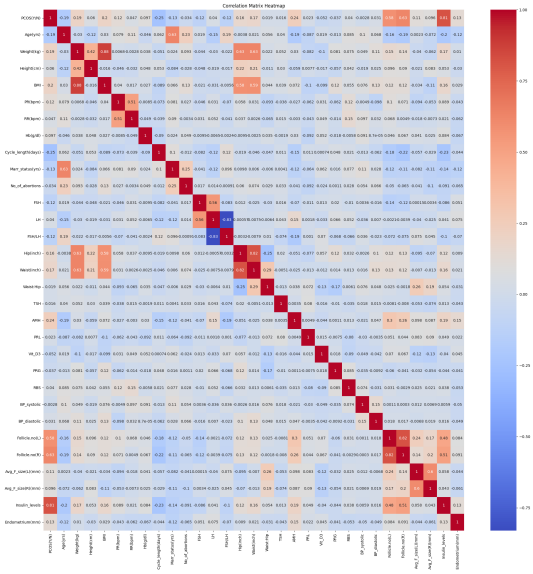


Fig. 11. Spearman correlation for numerical columns

8.2 Inference for numerical columns:

High positive correlation: 'Insulin_levels', 'Follicle.no(L)', 'Follicle.no(R)'
High negative correlation: 'Cycle_length(days)', 'Age(yrs)', 'Marr_status(yrs)','FSH','FSH/LH'

8.3 Correlation matrix for categorical attributes:

Spearman correlation for categorical variables (Fig.12)

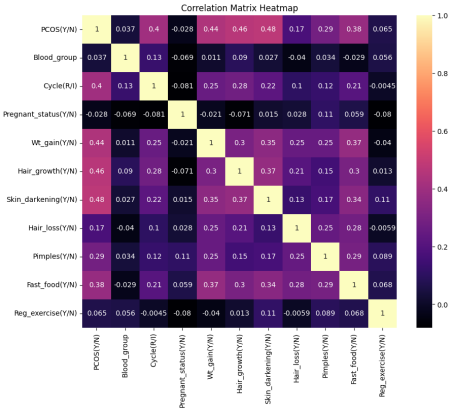


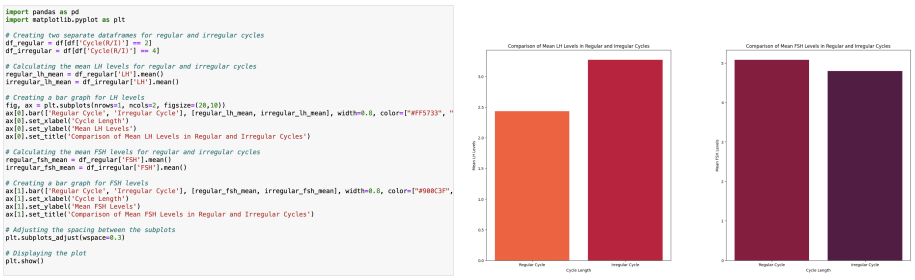
Fig. 12. Correlation matrix for categorical attributes

8.4 Inference for categorical variables:

High positive correlation: 'Skin_darkening(Y/N)', 'Hair_growth(Y/N)', 'Wt_gain(Y/N)', 'Cycle(R/I)'
High Negative correlation: 'Pregnant_status(Y/N)'

9 Data Visualization:

We chose bar graphs and stacked bar graphs to visualize the categorical data. To visualize numerical attributes histograms, KDE plots, and scatter plots were used.

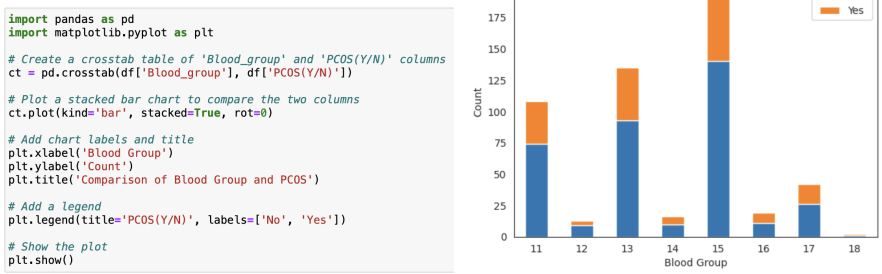


```
import pandas as pd
import matplotlib.pyplot as plt

# Creating two separate dataframes for regular and irregular cycles
df_regular = df[df['Cycle(R/I)'] == 2]
df_irregular = df[df['Cycle(R/I)'] == 4]

# Calculating the mean LH levels for regular and irregular cycles
regular_lh_mean = df_regular['LH'].mean()
irregular_lh_mean = df_irregular['LH'].mean()

# Creating a bar graph for LH levels
fig, ax = plt.subplots(nrows=1, ncols=2, figsize=(20,10))
ax[0].bar(['Regular Cycle', 'Irregular Cycle'], [regular_lh_mean, irregular_lh_mean], width=0.8, color=["#FF5733", "
ax[0].set_xlabel('Cycle Length')
ax[0].set_ylabel('Mean LH Levels')
ax[0].set_title('Comparison of Mean LH Levels in Regular and Irregular Cycles')

# Calculating the mean FSH levels for regular and irregular cycles
regular_fsh_mean = df_regular['FSH'].mean()
irregular_fsh_mean = df_irregular['FSH'].mean()

# Creating a bar graph for FSH levels
ax[1].bar(['Regular Cycle', 'Irregular Cycle'], [regular_fsh_mean, irregular_fsh_mean], width=0.8, color=["#900C3F",
ax[1].set_xlabel('Cycle Length')
ax[1].set_ylabel('Mean FSH Levels')
ax[1].set_title('Comparison of Mean FSH Levels in Regular and Irregular Cycles')

# Adjusting the spacing between the subplots
plt.subplots_adjust(wspace=0.3)

# Displaying the plot
plt.show()
```


Fig. 13. Bar graphs depicting a comparison of LH, FSH with Cycle(R/L)

```
import pandas as pd
import matplotlib.pyplot as plt

# Creating a list of columns to analyze
columns = ['Blood_group', 'Cycle(R/I)', 'Pregnant_status(Y/N)', 'Wt_gain(Y/N)',
           'Hair_growth(Y/N)', 'Skin_darkening(Y/N)', 'Hair_loss(Y/N)', 'Pimples(Y/N)',
           'Fast_food(Y/N)', 'Reg_exercise(Y/N)']

# Setting up the figure and different axes on the figure
fig, axs = plt.subplots(nrows=2, ncols=5, figsize=(20, 10))

# Define colors for each variable
color_dict = {
    'Blood_group': ['#FFC0CB', '#BEBEBE'],
    'Cycle(R/I)': ['#B0E0E6', '#FFA07A'],
    'Pregnant_status(Y/N)': ['#90EE90', '#FFA500'],
    'Wt_gain(Y/N)': ['#B7CEFA', '#FFD700'],
    'Hair_growth(Y/N)': ['#ADD8E6', '#FF69B4'],
    'Skin_darkening(Y/N)': ['pink', '#1E90FF'],
    'Hair_loss(Y/N)': ['#90EE90', '#FF6347'],
    'Pimples(Y/N)': ['#FAE68C', '#6A5ACD'],
    'Fast_food(Y/N)': ['#D3D3D3', '#BA55D3'],
    'Reg_exercise(Y/N)': ['#FFB6C1', '#7B68EE']
}

# Plotting each crosstab on a different axis
for i, col in enumerate(columns):
    ct = pd.crosstab(df[col], df['PCOS(Y/N)'])
    ct.plot(kind='bar', ax=axs[i//5][i%5], color=color_dict[col], rot=0, grid=False)
    axs[i//5][i%5].set_xticks([0,1])
    axs[i//5][i%5].set_xticklabels(["No","Yes"])
    axs[i//5][i%5].legend(labels=['NO PCOS', 'PCOS'])
# Add overall title and save figure
plt.suptitle('PCOS Analysis', fontsize=16, y=1.05)

plt.tight_layout()
```


Fig. 14. Bar Charts of PCOS(Y/N) vs Categorical columns



```
import pandas as pd
import matplotlib.pyplot as plt

# Create a crosstab table of 'Blood_group' and 'PCOS(Y/N)' columns
ct = pd.crosstab(df['Blood_group'], df['PCOS(Y/N)'])

# Plot a stacked bar chart to compare the two columns
ct.plot(kind='bar', stacked=True, rot=0)

# Add chart labels and title
plt.xlabel('Blood Group')
plt.ylabel('Count')
plt.title('Comparison of Blood Group and PCOS')

# Add a legend
plt.legend(title='PCOS(Y/N)', labels=['No', 'Yes'])

# Show the plot
plt.show()
```


Fig. 15. Stacked bar chart depicting the comparison of blood_groups with PCOS(Y/N)

```python
import numpy as np
import matplotlib.pyplot as plt

# generate some sample data
ages = np.arange(18, 50)
bmi = np.random.normal(loc=25, scale=4, size=len(ages))
pcos = np.random.randint(0, 2, size=len(ages))

# Ages and BMI for people who don't have PCOS
plt.scatter(ages[pcos==0],
            bmi[pcos==0],
            c="blue",
            alpha=0.5)

# Ages and BMI for people who have PCOS
plt.scatter(ages[pcos==1],
            bmi[pcos==1],
            c="red")

# Styling
plt.title("Age and BMI's correlation with having PCOS")
plt.xlabel("Age")
plt.ylabel("BMI")
plt.legend(["No PCOS", "PCOS"])

# set the x-tick labels to consecutive age numbers and rotate them by 45 degrees
plt.xticks(ages, rotation=90)

# set the figure size
plt.figure(figsize=(100, 6))

plt.show()
```

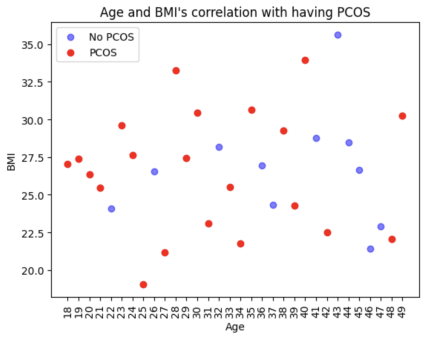


Fig. 16. Scatter plot depicting the correlation between Age and BMI with PCOS(Y/N)

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# create a new column for Age ranges
df['Age range'] = pd.cut(df['Age(yrs)'], bins=[0, 20, 30, 40, 50, 60, 70, 80, 90, 100])

# create cross-tabulation
ct = pd.crosstab(df['PCOS(Y/N)'], df['Age range'])

# create heatmap
fig, ax = plt.subplots(figsize=(10, 5))
sns.heatmap(ct, cmap='Oranges', annot=True, fmt='d', ax=ax, annot_kws={"fontsize":12})

# set plot labels
ax.set_xlabel('Age range', fontsize=20)
ax.set_ylabel('PCOS(Y/N)', fontsize=14)
ax.set_title('PCOS vs Age', fontsize=16)

# show plot
plt.show()
```

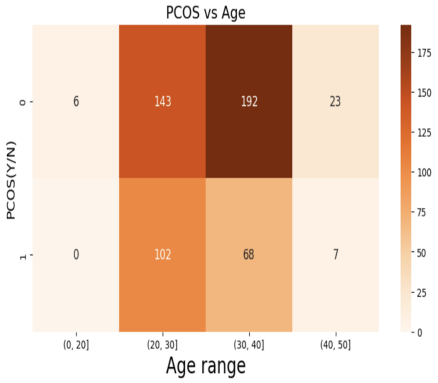


Fig. 17. Cross-tabulation Heatmap of PCOS (Y/N) vs Age Range

```python
import seaborn as sns
import matplotlib.pyplot as plt

# Subset the data for pregnant and non-pregnant individuals
pregnant = df[df['Pregnant_status(Y/N)']=='Y']
non_pregnant = df[df['Pregnant_status(Y/N)']=='N']

# Create density plots
fig, axs = plt.subplots(nrows=1, ncols=3, figsize=(15, 5))

sns.kdeplot(x='No_of_abortions', hue='Pregnant_status(Y/N)', data=df, ax=axs[0])
sns.kdeplot(x='AMH', hue='Pregnant_status(Y/N)', data=df, ax=axs[1])
sns.kdeplot(x='PRG', hue='Pregnant_status(Y/N)', data=df, ax=axs[2])

# Set axis labels and title
axs[0].set_xlabel('No of Abortions')
axs[0].set_ylabel('Density')
axs[0].set_title('Distribution of No of Abortions')

axs[1].set_xlabel('AMH')
axs[1].set_ylabel('Density')
axs[1].set_title('Distribution of AMH')

axs[2].set_xlabel('PRG')
axs[2].set_ylabel('Density')
axs[2].set_title('Distribution of PRG')

plt.tight_layout()
plt.show()
```

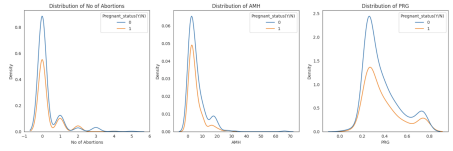


Fig. 18. Density Plots for No of Abortions, AMH, and PRG with Pregnancy Status(Y/N)

10 Model Building:

We opted for using classifiers in our modeling approach due to the binary nature of the predicting variable, which has two possible outcomes (Yes[1], No [0]).

10.1 Preparing data for model building:

Since the data has already been processed to remove any inconsistencies or errors, we proceeded with selecting the features essential for model building by considering the relationship between independent variables. We used statistical analysis, specifically examining the p-values, and correlation to determine which factors to be included in our model (Fig.19).

Once we identified the most important features, we divided the data into two parts: X and Y. After that, we applied two pre-processing techniques, SMOTE and scaling with the min-max scaler(Fig.19).
SMOTE helps to address the class imbalance in the data, while scaling ensures that all features are on a similar scale between 0 and 1. By doing this, we can improve the accuracy and performance of the machine learning models.
These are the features we chose for machine learning model building 'PCOS(Y/N)', 'Age(yrs)', 'Weight(kg)', 'FSH/LH', 'Waist:Hip', 'AMH', 'Wt_gain(Y/N)', 'Cycle(R/I)', 'RBS', 'Pregnant_status(Y/N)', 'Hair_growth(Y/N)', 'Skin_darkening(Y/N)', 'Hair_loss(Y/N)', 'Fast_food(Y/N)', 'Avg_F_size(R)(mm)', 'Endometrium(mm)'

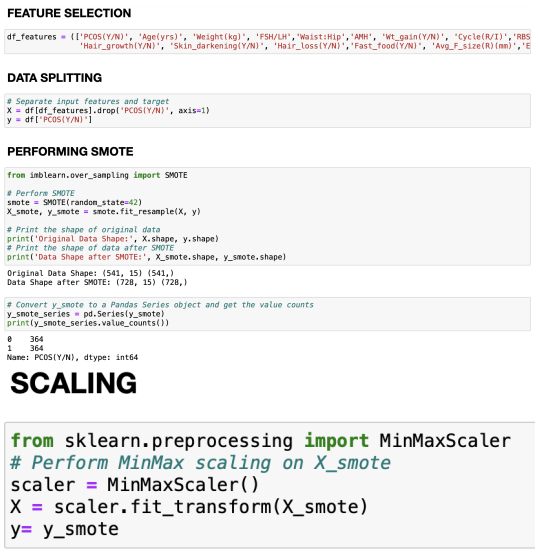


Fig. 19. Feature Selection, SMOTE and Scaling

We performed:
1. Logistic regression
2. Gaussian Naive Bayes
3. Decision tree classifier
4. Support vector machine

10.2 Logistic regression:

Logistic regression, which is also known as the sigmoid function, is a simple yet interpretable algorithm that assumes a relationship between predictor variables and the target variables.

Due to the limited and small size of our dataset, there is a high probability of overfitting, making it necessary to choose an appropriate algorithm. Therefore, I opted for logistic regression, as can handle small datasets more effectively. (Fig.20, Fig.21)

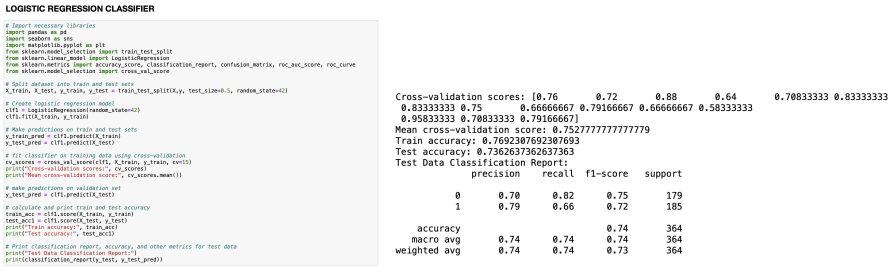


Fig. 20. Logistic regression

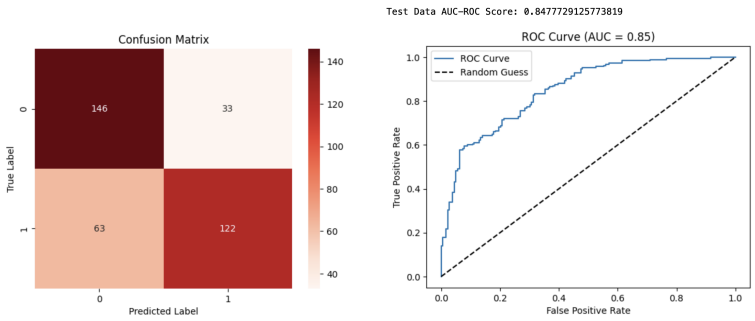


Fig. 21. Confusion Matrix and AUC-ROC Curve of Logistic regression

10.3 Gaussian Naïve bayes:

Gaussian Naive Bayes is a commonly used algorithm for classification tasks, appreciated for its simplicity, quick training time, and ability to handle high-dimensional data. Additionally, it can produce reliable results with even small datasets by modeling the distribution of each feature independently.(Fig.22, Fig.23)

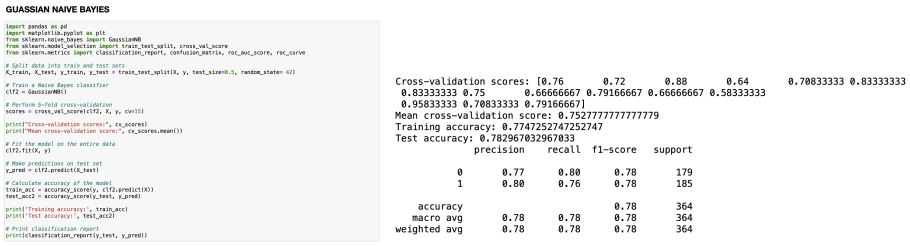


Fig. 22. Gaussian Naïve bayes

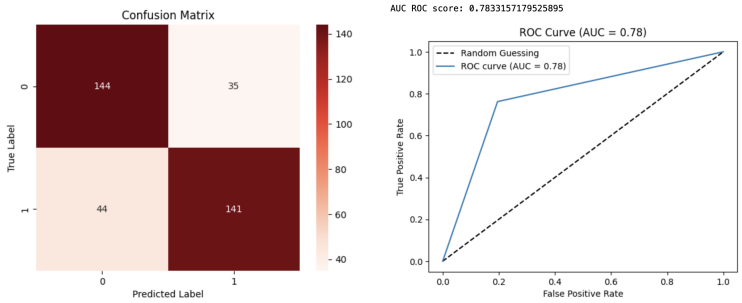


Fig. 23. Confusion Matrix and AUC-ROC Curve of Gaussian Naïve bayes

10.4 Decision tree classifier:

Decision Tree is also a simple and interpretable algorithm for building machine learning models. It creates a tree-like structure by repeatedly splitting the data based on the most informative feature, resulting in a set of rules that can be easily understood and explained.

This algorithm is also capable of handling noisy and complex data sets, making it suitable for a wide range of applications. (Fig.24, Fig.25)

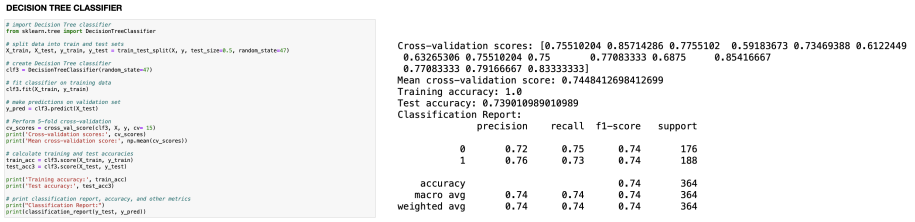


Fig. 24. Decision tree classifier

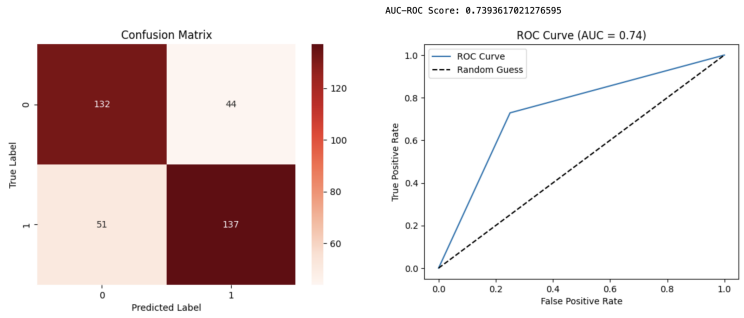


Fig. 25. Confusion Matrix and AUC-ROC Curve of Decision tree classifier

10.5 Support vector machine:

SVM is flexible and can handle a variety of data types, including numerical, categorical, and textual data.

SVM is capable of achieving high accuracy even on complex and non-linear datasets. It is also designed to be robust to outliers, making it a suitable choice for datasets with noisy or incomplete data. ((Fig.26, Fig.27.)

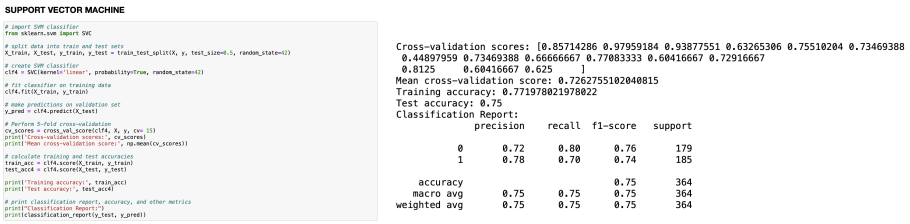


Fig. 26. Support vector machine

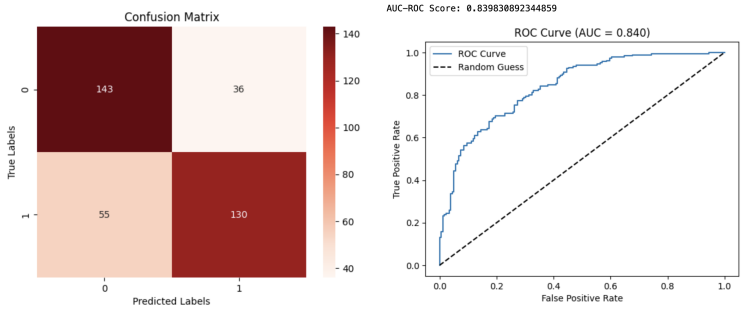


Fig. 27. Confusion Matrix and AUC-ROC Curve of Support vector machine

11 Model evaluation : (Fig.28, Fig.29,Fig.30)

To evaluate the models, we plotted AUC-ROC curves for each and represented their respective test accuracies using a bar graph.

MODEL EVALUATUION

```
clf1 = LogisticRegression(random_state=42)
clf2 = GaussianNB()
clf3 = DecisionTreeClassifier(random_state=47)
clf4 = SVC(kernel='linear', probability=True, random_state=42)

# Fit the model on the entire data
clf1.fit(X_train, y_train)
y_test_pred_lr = clf1.predict_proba(X_test)[:,1]
clf2.fit(X, y)
y_test_pred_nb = clf2.predict_proba(X_test)[:,1]
clf3.fit(X_train, y_train)
y_test_pred_dt = clf3.predict_proba(X_test)[:,1]
clf4.fit(X_train, y_train)
y_test_pred_svm = clf4.predict_proba(X_test)[:,1]

# Calculate AUC-ROC score for each model
lr_auc = auc_roc_lr
nb_auc = auc_roc_gnb
dt_auc = auc_roc_dt
svm_auc = auc_roc_svm

# Plot ROC curves for each model on a single plot
fpr_lr, tpr_lr, _ = roc_curve(y_test, y_test_pred_lr)
fpr_nb, tpr_nb, _ = roc_curve(y_test, y_test_pred_nb)
fpr_dt, tpr_dt, _ = roc_curve(y_test, y_test_pred_dt)
fpr_svm, tpr_svm, _ = roc_curve(y_test, y_test_pred_svm)

plt.plot([0, 1], [0, 1], 'k--')
plt.plot(fpr_lr, tpr_lr, label='Logistic Regression (AUC = %0.3f)' % lr_auc)
plt.plot(fpr_nb, tpr_nb, label='Naive Bayes (AUC = %0.3f)' % nb_auc)
plt.plot(fpr_dt, tpr_dt, label='Decision Tree (AUC = %0.3f)' % dt_auc)
plt.plot(fpr_svm, tpr_svm, label='SVM (AUC = %0.3f)' % svm_auc)

plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('ROC curve')
plt.legend(loc='best')
plt.show()
```

Fig. 28. code for ROC curves of all models

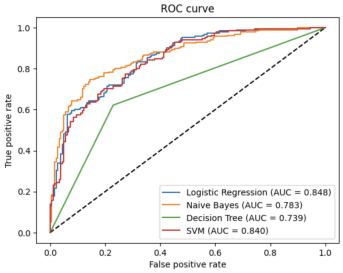


Fig. 29. ROC curves of all models

```
import matplotlib.pyplot as plt

# List of classifiers and their corresponding test accuracies
classifiers = ['Logistic Regression', 'Naive Bayes', 'Decision Tree', 'SVM']
test_accs = [test_acc1, test_acc2, test_acc3, test_acc4]

# Plot the bar plot
plt.figure(figsize=(8,6))
sns.set_style("white")
ax = sns.barplot(x=['Logistic Regression', 'Naive Bayes', 'Decision Tree', 'SVM'], y=test_accs, palette='viridis')
plt.ylim(0.5,1.0)
plt.title('Testing Accuracies of Different Models')
plt.xlabel('Model')
plt.ylabel('Testing Accuracy')

# Add scores above the bars
for i in ax.containers:
    ax.bar_label(i, label_type="edge", fontsize=10)
plt.show()
```

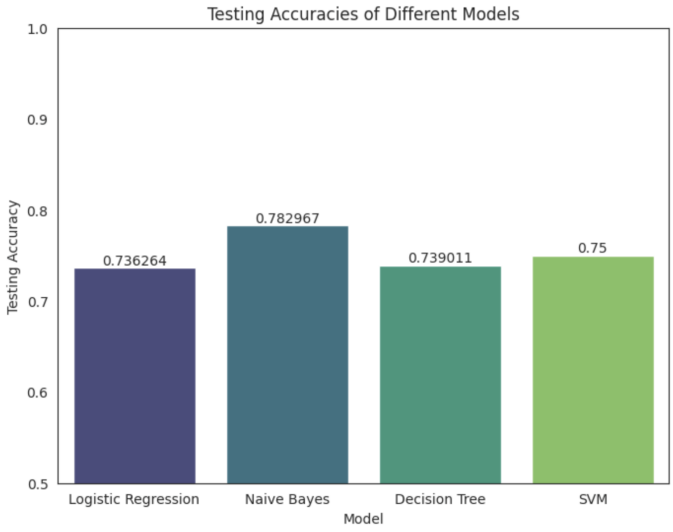


Fig. 30. Bar Graph depicting Test accuracies

12 Conclusion:

12.1 Summary of Findings:

Attributes like Age(yrs), Weight(kg), BMI, PR(bpm), Hb(g/dl), Cycle_length(days), Marr_status(yrs), FSH, FSH/LH, Hip(inch), Waist(inch), Avg_F_size(L)(mm), Avg_F_size(R)(mm), Insulin_levels,Endometrium(mm),Cycle(R/I), Wt_gain(Y/N), Hair_growth(Y/N), Skin_darkening(Y/N), Hair_loss(Y/N), Pimples(Y/N), Fast_food(Y/N) rejected the null hypothesis.

Table 2. Model evaluation based on test accuracies.

Model	Test Accuracy	AUC-ROC Score
Logistic regression	0.733	0.849
Gaussian Naive Bayes	0.785	0.786
Decision tree classifier	0.711	0.742
Support Vector Machine	0.741	0.842

Based on the test accuracies, the Gaussian Naive Bayes model performed the best with a test accuracy of 0.785, followed by the Support vector machine with a test accuracy of 0.741, and the Logistic regression with a test accuracy of 0.733 and the Decision tree with a test accuracy of 0.711. Hence, the best-fit model for our dataset is Gaussian Naive Bayes.

13 Limitations:

While we did our best to control limitations, as with any research study, there will always be limitations. The first limitation is that the data set was last updated in 2021 and it lacked data description, but fortunately, we communicated with the author through LinkedIn and he updated the dataset with a description.

The dataset we had was small with only 541 which made it easily susceptible to overfitting, so we had to implement scaling, smote analysis, and cross-validation to overcome the issue and to build a perfect model.

The major limitation was our lack of experience in the technical field, we just learned a few basics necessary to start programming, but during the initial stages of the project we were so scared to learn the way to approach the project and write codes but the classes were planned strategically to cover steps of the project sequentially, this made comfortable with the project.

14 Appendix:

Data collection
Data extraction and Data loading
The connection between My SQL and Jupyter
Data description
Data cleaning
Data analysis
Exploratory data analysis
Normality testing
Hypothesis testing
Correlation
Data visualization
Model building
Model evaluation

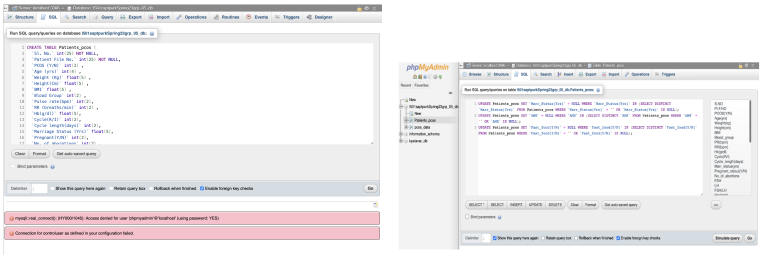


Fig. 31. SQL Codes

Fig. 32. Connecting SQL with Jupyter

```
df_old
```

	S.NO	Pt.F.NO	PCOS(Y/N)	Age(yrs)	Weight(kg)	Height(cm)	BMI	Blood_group	PR(bpm)	RR(bpm)	...	Fast_food(Y/N)	Reg_exercise(Y/N)	BP_systolic
0	1	10001	0	28	44.6	152.000	19.3000	15	78	22	...	1	0	11(
1	2	10002	0	36	65.0	161.500	24.9212	15	74	20	...	0	0	12(
2	3	10003	1	33	68.8	165.000	25.2709	11	72	18	...	1	0	12(
3	4	10004	0	37	65.0	148.000	29.6749	13	72	20	...	0	0	12(
4	5	10005	0	25	52.0	161.000	20.0610	11	72	18	...	0	0	12(
...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
536	537	10537	0	35	50.0	164.592	18.5000	17	72	16	...	0	0	11(
537	538	10538	0	30	63.2	158.000	25.3000	15	72	18	...	0	0	11(
538	539	10539	0	36	54.0	152.000	23.4000	13	74	20	...	0	0	11(
539	540	10540	0	27	50.0	150.000	22.2000	15	74	20	...	0	0	11(
540	541	10541	1	23	82.0	165.000	30.1000	13	80	20	...	1	0	12(

541 rows × 43 columns

Fig. 33. Data frame(df_Old)

DROPPING THE UNWANTED COLUMNS AND ASSIGNING A NEW DATAFRAME

```
# create a new DataFrame with dropped columns
df = df_old.drop(['S.NO', 'Pt.F.NO'], axis=1)
```

we removed Serial number, Patient file number attributes which are not essential for research study and named the data frame as df

```
# the dataframe after removing the unwanted columns
df
```

	PCOS(Y/N)	Age(yrs)	Weight(kg)	Height(cm)	BMI	Blood_group	PR(bpm)	RR(bpm)	Hb(g/dl)	Cycle(R/I)	...	Fast_food(Y/N)	Reg_exercise(Y/N)	BP_sy
0	0	28	44.6	152.000	19.3000	15	78	22	10.48	2	...	1	0	
1	0	36	65.0	161.500	24.9212	15	74	20	11.70	2	...	0	0	
2	1	33	68.8	165.000	25.2709	11	72	18	11.80	2	...	1	0	
3	0	37	65.0	148.000	29.6749	13	72	20	12.00	2	...	0	0	
4	0	25	52.0	161.000	20.0610	11	72	18	10.00	2	...	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
536	0	35	50.0	164.592	18.5000	17	72	16	11.00	2	...	0	0	
537	0	30	63.2	158.000	25.3000	15	72	18	10.80	2	...	0	0	
538	0	36	54.0	152.000	23.4000	13	74	20	10.80	2	...	0	0	
539	0	27	50.0	150.000	22.2000	15	74	20	12.00	4	...	0	0	
540	1	23	82.0	165.000	30.1000	13	80	20	10.20	4	...	1	0	

541 rows × 41 columns

Fig. 34. Dropping Unwanted columns

```
import pandas as pd
# Get the number of rows and columns
num_rows, num_cols = df.shape
print("Number of rows:", num_rows)
print("Number of columns:", num_cols)
```

```
Number of rows: 541
Number of columns: 41
```

```
df.shape
```

```
(541, 41)
```

Fig. 35. Shape and Number of rows

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541 entries, 0 to 540
Data columns (total 41 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   PCOS(Y/N)             541 non-null    int64
 1   Age(yrs)              541 non-null    int64
 2   Weight(kg)            541 non-null    float64
 3   Height(cm)            541 non-null    float64
 4   BMI                   541 non-null    float64
 5   Blood_group           541 non-null    int64
 6   PR(bpm)               541 non-null    int64
 7   RR(bpm)               541 non-null    int64
 8   Hb(g/dl)              541 non-null    float64
 9   Cycle(R/I)            541 non-null    int64
 10  Cycle_length(days)    541 non-null    int64
 11  Marr_status(yrs)      540 non-null    object
 12  Pregnant_status(Y/N)  541 non-null    int64
 13  No_of_abortions       541 non-null    int64
 14  FSH                   541 non-null    float64
 15  LH                    541 non-null    float64
 16  FSH/LH                541 non-null    float64
 17  Hip(inch)             541 non-null    int64
 18  Waist(inch)           541 non-null    int64
 19  Waist:Hip             541 non-null    float64
 20  TSH                   541 non-null    float64
 21  AMH                   540 non-null    object
 22  PRL                   541 non-null    float64
 23  Vit_D3                541 non-null    float64
 24  PRG                   541 non-null    float64
 25  RBS                   541 non-null    float64
 26  Wt_gain(Y/N)          541 non-null    int64
 27  Hair_growth(Y/N)      541 non-null    int64
 28  Skin_darkening(Y/N)   541 non-null    int64
 29  Hair_loss(Y/N)        541 non-null    int64
 30  Pimples(Y/N)          541 non-null    int64
 31  Fast_food(Y/N)        540 non-null    object
 32  Reg_exercise(Y/N)     541 non-null    int64
```

Fig. 36. Information of new data frame(df)

```
df.columns
```

```
Index(['PCOS(Y/N)', 'Age(yrs)', 'Weight(kg)', 'Height(cm)', 'BMI',
       'Blood_group', 'PR(bpm)', 'RR(bpm)', 'Hb(g/dl)', 'Cycle(R/I)',
       'Cycle_length(days)', 'Marr_status(yrs)', 'Pregnant_status(Y/N)',
       'No_of_abortions', 'FSH', 'LH', 'FSH/LH', 'Hip(inch)', 'Waist(inch)',
       'Waist:Hip', 'TSH', 'AMH', 'PRL', 'Vit_D3', 'PRG', 'RBS',
       'Wt_gain(Y/N)', 'Hair_growth(Y/N)', 'Skin_darkening(Y/N)',
       'Hair_loss(Y/N)', 'Pimples(Y/N)', 'Fast_food(Y/N)', 'Reg_exercise(Y/N)',
       'BP_systolic', 'BP_diastolic', 'Follicle.no(L)', 'Follicle.no(R)',
       'Avg_F_size(L)(mm)', 'Avg_F_size(R)(mm)', 'Insulin_levels',
       'Endometrium(mm)'],
      dtype='object')
```

Fig. 37. Column names

```
# checkin g for no of unique values in that columns
for i in df.columns:
     print(i,len(df[i].unique()))
```

```
PCOS(Y/N) 2
Age(yrs) 29
Weight(kg) 117
Height(cm) 50
BMI 355
Blood_group 8
PR(bpm) 11
RR(bpm) 8
Hb(g/dl) 46
Cycle(R/I) 3
Cycle_length(days) 12
Marr_status(yrs) 35
Pregnant_status(Y/N) 2
No_of_abortions 6
FSH 371
LH 342
FSH/LH 512
Hip(inch) 19
Waist(inch) 23
Waist:Hip 96
TSH 308
AMH 301
PRL 481
Vit_D3 331
PRG 89
RBS 55
Wt_gain(Y/N) 2
Hair_growth(Y/N) 2
Skin_darkening(Y/N) 2
Hair_loss(Y/N) 2
Pimples(Y/N) 2
Fast_food(Y/N) 3
Reg_exercise(Y/N) 2
BP_systolic 6
BP_diastolic 5
Follicle.no(L) 21
Follicle.no(R) 20
```

Fig. 38. Unique values

```
cat_cols = ['PCOS(Y/N)', 'Blood_group', 'Cycle(R/I)', 'Pregnant_status(Y/N)', 'Wt_gain(Y/N)', 'Hair_growth(Y/N)', 'S
num_cols = ['Age(yrs)', 'Weight(kg)', 'Height(cm)', 'BMI', 'PR(bpm)', 'RR(bpm)', 'Hb(g/dl)', 'Cycle_length(days)',

print(f'Categorical Variable: {cat_cols}')
print(f'Numerical Variable: {num_cols}')
```

```
Categorical Variable: ['PCOS(Y/N)', 'Blood_group', 'Cycle(R/I)', 'Pregnant_status(Y/N)', 'Wt_gain(Y/N)', 'Hair_grow
th(Y/N)', 'Skin_darkening(Y/N)', 'Hair_loss(Y/N)', 'Pimples(Y/N)', 'Fast_food(Y/N)', 'Reg_exercise(Y/N)']
Numerical Variable: ['Age(yrs)', 'Weight(kg)', 'Height(cm)', 'BMI', 'PR(bpm)', 'RR(bpm)', 'Hb(g/dl)', 'Cycle_lengt
h(days)', 'Marr_status(yrs)', 'No_of_abortions', 'FSH', 'LH', 'FSH/LH', 'Hip(inch)', 'Waist(inch)', 'Waist:Hip', 'T
SH', 'AMH', 'PRL', 'Vit_D3', 'PRG', 'RBS', 'BP_systolic', 'BP_diastolic', 'Follicle.no(L)', 'Follicle.no(R)', 'Avg_
F_size(L)(mm)', 'Avg_F_size(R)(mm)', 'Insulin_levels', 'Endometrium(mm)']
```

```
df["PCOS(Y/N)"].value_counts()
```

```
0    364
1    177
Name: PCOS(Y/N), dtype: int64
```

Fig. 39. Seperating categorical and numerical columns and Count of PCOS(Y/N)

```
print(df.isnull().any())
```

```
PCOS(Y/N)                False
Age(yrs)                 False
Weight(kg)               False
Height(cm)               False
BMI                      False
Blood_group              False
PR(bpm)                  False
RR(bpm)                  False
Hb(g/dl)                 False
Cycle(R/I)               False
Cycle_length(days)       False
Marr_status(yrs)          True
Pregnant_status(Y/N)     False
No_of_abortions          False
FSH                      False
LH                       False
FSH/LH                   False
Hip(inch)                False
Waist(inch)              False
Waist:Hip                False
TSH                      False
AMH                       True
PRL                      False
Vit_D3                   False
PRG                      False
RBS                      False
Wt_gain(Y/N)             False
Hair_growth(Y/N)         False
Skin_darkening(Y/N)      False
Hair_loss(Y/N)           False
Pimples(Y/N)             False
Fast_food(Y/N)            True
Reg_exercise(Y/N)        False
BP_systolic              False
BP_diastolic             False
Follicle.no(L)           False
Follicle.no(R)           False
Avg_F_size(L)(mm)        False
```

Fig. 40. Checking for null values

AGAIN CHECKING THE NULL VALUES IF STILL EXISTS

```
# Check if any null values still exist in the entire DataFrame
if df.isnull().any().any():
    print("Null values exist in the dataset")
else:
    print("No null values exist in the dataset")
```

```
No null values exist in the dataset
```

```
print(df.isnull().any())
```

```
PCOS(Y/N)                False
Age(yrs)                 False
Weight(kg)               False
Height(cm)               False
BMI                      False
Blood_group              False
PR(bpm)                  False
RR(bpm)                  False
Hb(g/dl)                 False
Cycle(R/I)               False
Cycle_length(days)       False
Marr_status(yrs)         False
Pregnant_status(Y/N)     False
No_of_abortions          False
FSH                      False
LH                       False
FSH/LH                   False
Hip(inch)                False
Waist(inch)              False
Waist:Hip                False
TSH                      False
AMH                      False
PRL                      False
Vit_D3                   False
PRG                      False
RBS                      False
Wt_gain(Y/N)             False
Hair_growth(Y/N)         False
```

Fig. 41. After replacing, again checking for null values

CHECKING IF THERE WERE ANY DUPLICATED ROWS EXISTS

```
df[df.duplicated()]
```

PCOS(Y/N)	Age(yrs)	Weight(kg)	Height(cm)	BMI	Blood_group	PR(bpm)	RR(bpm)	Hb(g/dl)	Cycle(R/I)	...	Fast_food(Y/N)	Reg_exercise(Y/N)	BP_systolic

0 rows × 41 columns

Fig. 42. Checking if any duplicates exist

VISUALISING THE DATA BEFORE REMOVING OUTLIERS

```
import matplotlib.pyplot as plt
# Separate categorical and numerical columns
cat_cols = ['PCOS(Y/N)', 'Blood_group', 'Cycle(R/I)', 'Pregnant_status(Y/N)', 'Wt_gain(Y/N)', 'Hair_growth(Y/N)', 'S

num_cols = ['Age(yrs)', 'Weight(kg)', 'Height(cm)', 'BMI', 'PR(bpm)', 'RR(bpm)', 'Hb(g/dl)', 'Cycle_length(days)', '

# Separate categorical and numerical data frames
cat_data = df[cat_cols]
num_data = df[num_cols]

# Set colors for numerical columns
num_colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k', 'orange', 'purple', 'brown', 'pink', 'gray', 'olive', 'navy','salmo

# Set colors for categorical columns
cat_colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k', 'orange', 'purple', 'brown', 'pink']

# Visualize categorical columns in bar chart
for i, col in enumerate(cat_cols):
    plt.figure()
    cat_data[col].value_counts().plot(kind='bar', color=cat_colors[i % len(cat_colors)])
    plt.title(col)
    plt.xlabel(col)
    plt.ylabel("Count")
    plt.show()

# Visualize numerical columns in histogram
for i, col in enumerate(num_cols):
    plt.figure()
    plt.hist(num_data[col], color=num_colors[i % len(num_colors)])
    plt.title(col)
    plt.xlabel(col)
    plt.ylabel("Count")
    plt.show()
```

Fig. 43. Visualizing data before removing outliers

Fig. 44. a. Minimum and Maximum values

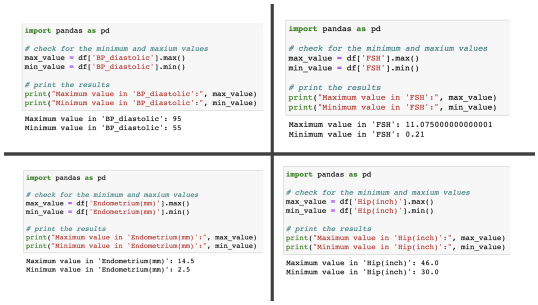


Fig. 45. b. Minimum and Maximum values

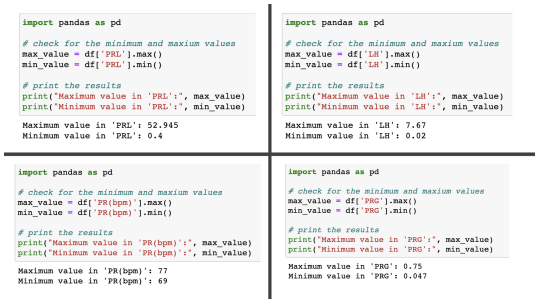


Fig. 46. c. Minimum and Maximum values

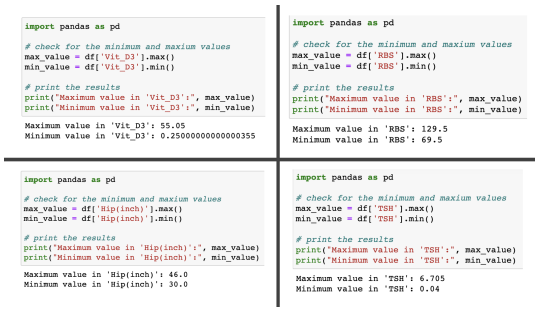


Fig. 47. d. Minimum and Maximum values

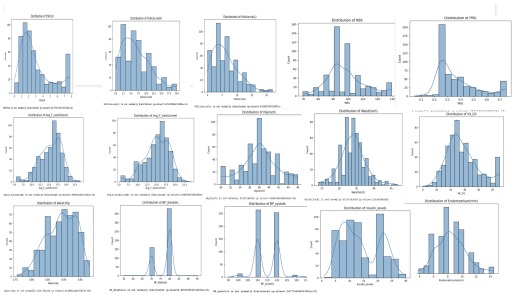


Fig. 48. a. Results of Normality testing

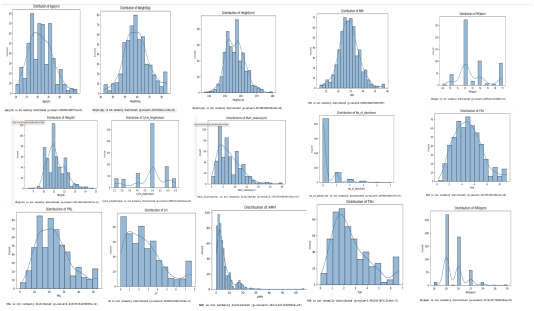


Fig. 49. b. Results of Normality testing

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              PCOS(Y/N)   R-squared:                       0.809
Model:                            OLS   Adj. R-squared:                  0.804
Method:                 Least Squares   F-statistic:                     148.6
Date:                Thu, 04 May 2023   Prob (F-statistic):          9.39e-178
Time:                        18:52:29   Log-Likelihood:                 90.038
No. Observations:                 541   AIC:                            -148.1
Df Residuals:                     525   BIC:                            -79.38
Df Model:                          15
Covariance Type:            nonrobust
=====================================================================================
                        coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------
const                -0.4046      1.268     -0.319      0.750      -2.896       2.087
Weight(kg)            0.0013      0.011      0.124      0.901      -0.019       0.022
Height(cm)            0.0001      0.008      0.014      0.989      -0.016       0.016
BMI                   0.0016      0.026      0.061      0.952      -0.049       0.052
Hb(g/dl)              0.0130      0.011      1.199      0.231      -0.008       0.034
FSH                  -0.0067      0.005     -1.374      0.170      -0.016       0.003
LH                    0.0039      0.008      0.508      0.611      -0.011       0.019
FSH/LH                0.0029      0.006      0.458      0.647      -0.009       0.015
Waist:Hip            -0.3943      0.200     -1.975      0.049      -0.787      -0.002
TSH                   0.0006      0.006      0.105      0.917      -0.010       0.012
AMH                   0.0043      0.002      2.666      0.008       0.001       0.008
PRL                  -0.0006      0.001     -0.815      0.415      -0.002       0.001
Vit_D3               -0.0004      0.001     -0.500      0.617      -0.002       0.001
PRG                  -0.0098      0.058     -0.169      0.866      -0.124       0.105
RBS                  -0.0003      0.001     -0.457      0.648      -0.002       0.001
Insulin_levels        0.0606      0.001     42.865      0.000       0.058       0.063
==============================================================================
Omnibus:                       53.160   Durbin-Watson:                   1.473
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               40.400
Skew:                          -0.569   Prob(JB):                     1.69e-09
Kurtosis:                       2.296   Cond. No.                     2.85e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.85e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
Rejected features: ['Weight(kg)', 'Height(cm)', 'BMI', 'Hb(g/dl)', 'FSH', 'LH', 'FSH/LH', 'TSH', 'PRL', 'Vit_D3', '
PRG', 'RBS']
Accepted features: ['Waist:Hip', 'AMH', 'Insulin_levels']
```

Fig. 50. Results of Linear Regression

```
Mann-Whitney U test comparing PCOS(Y/N) and Age(yrs):
Statistic = 39545.0000, p-value = 0.00001673530510128288, Reject null hypothesis
There is a significant difference between PCOS and Age(yrs)

Mann-Whitney U test comparing PCOS(Y/N) and PR(bpm):
Statistic = 27662.5000, p-value = 0.004005916414342451 83, Reject null hypothesis
There is a significant difference between PCOS and PR(bpm)

Mann-Whitney U test comparing PCOS(Y/N) and RR(bpm):
Statistic = 30524.0000, p-value = 0.27787314565049081594, Cannot reject null hypothesis
There is no significant difference between PCOS and RR(bpm)

Mann-Whitney U test comparing PCOS(Y/N) and Cycle_length(days):
Statistic = 41380.0000, p-value = 0.00000000688460551371, Reject null hypothesis
There is a significant difference between PCOS and Cycle_length(days)

Mann-Whitney U test comparing PCOS(Y/N) and Marr_status(yrs):
Statistic = 37291.5000, p-value = 0.0028477318260851 7644, Reject null hypothesis
There is a significant difference between PCOS and Marr_status(yrs)

Mann-Whitney U test comparing PCOS(Y/N) and No_of_abortions:
Statistic = 33149.5000, p-value = 0.42439801010822675309, Cannot reject null hypothesis
There is no significant difference between PCOS and No_of_abortions

Mann-Whitney U test comparing PCOS(Y/N) and Hip(inch):
Statistic = 25803.0000, p-value = 0.00015535550766592760, Reject null hypothesis
There is a significant difference between PCOS and Hip(inch)

Mann-Whitney U test comparing PCOS(Y/N) and Waist(inch):
Statistic = 25334.0000, p-value = 0.00004946013817245037, Reject null hypothesis
There is a significant difference between PCOS and Waist(inch)

Mann-Whitney U test comparing PCOS(Y/N) and BP_systolic:
Statistic = 32312.5000, p-value = 0.94818591399759366567, Cannot reject null hypothesis
There is no significant difference between PCOS and BP_systolic


Mann-Whitney U test comparing PCOS(Y/N) and BP_systolic:
Statistic = 32312.5000, p-value = 0.94818591399759366567, Cannot reject null hypothesis
There is no significant difference between PCOS and BP_systolic

Mann-Whitney U test comparing PCOS(Y/N) and BP_diastolic:
Statistic = 31235.0000, p-value = 0.47016012493728853716, Cannot reject null hypothesis
There is no significant difference between PCOS and BP_diastolic

Mann-Whitney U test comparing PCOS(Y/N) and Follicle.no(L):
Statistic = 9274.5000, p-value = 0.0000000000000000000, Reject null hypothesis
There is a significant difference between PCOS and Follicle.no(L)

Mann-Whitney U test comparing PCOS(Y/N) and Follicle.no(R):
Statistic = 7397.0000, p-value = 0.0000000000000000000, Reject null hypothesis
There is a significant difference between PCOS and Follicle.no(R)

Mann-Whitney U test comparing PCOS(Y/N) and Avg_F_size(L)(mm):
Statistic = 27853.5000, p-value = 0.01020516801087069554, Reject null hypothesis
There is a significant difference between PCOS and Avg_F_size(L)(mm)

Mann-Whitney U test comparing PCOS(Y/N) and Avg_F_size(R)(mm):
Statistic = 28414.5000, p-value = 0.02517546164693388114, Reject null hypothesis
There is a significant difference between PCOS and Avg_F_size(R)(mm)

Mann-Whitney U test comparing PCOS(Y/N) and Endometrium(mm):
Statistic = 27211.5000, p-value = 0.00299043643591219389, Reject null hypothesis
There is a significant difference between PCOS and Endometrium(mm)
```

Fig. 51. Results of Mann Whitney U test

```
Chi-square test for Blood_group:
chi-square statistic: 2.0488
p-value:  0.9570901598044939
degree of freedom: 7
critical value: 14.0671
Fail to reject H0, there is no relationship between PCOS(Y/N) categorical variables Blood_group
Fail to reject H0 because there is no relationship between PCOS(Y/N) categorical variables Blood_group

Chi-square test for Cycle(R/I):
chi-square statistic: 87.3948
p-value:  1.0530565295416136e-19
degree of freedom: 2
critical value: 5.9915
Reject H0, there is a relationship between PCOS(Y/N) categorical variables Cycle(R/I)
Reject H0 because there is a relationship between PCOS(Y/N) categorical variables Cycle(R/I)

Chi-square test for Pregnant_status(Y/N):
chi-square statistic: 0.2990
p-value:  0.5845296923733834
degree of freedom: 1
critical value: 3.8415
Fail to reject H0, there is no relationship between PCOS(Y/N) categorical variables Pregnant_status(Y/N)
Fail to reject H0 because there is no relationship between PCOS(Y/N) categorical variables Pregnant_status(Y/N)

Chi-square test for Wt_gain(Y/N):
chi-square statistic: 103.3061
p-value:  2.8716592631832225e-24
degree of freedom: 1
critical value: 3.8415
Reject H0, there is a relationship between PCOS(Y/N) categorical variables Wt_gain(Y/N)
Reject H0 because there is a relationship between PCOS(Y/N) categorical variables Wt_gain(Y/N)

Chi-square test for Hair_growth(Y/N):
chi-square statistic: 114.5990
p-value:  9.633522761918661e-27
degree of freedom: 1
critical value: 3.8415
Reject H0, there is a relationship between PCOS(Y/N) categorical variables Hair_growth(Y/N)
Reject H0 because there is a relationship between PCOS(Y/N) categorical variables Hair_growth(Y/N)

Chi-square test for Skin_darkening(Y/N):
chi-square statistic: 120.2514
p-value:  5.573087703802324e-28
degree of freedom: 1
critical value: 3.8415
Reject H0, there is a relationship between PCOS(Y/N) categorical variables Skin_darkening(Y/N)
Reject H0 because there is a relationship between PCOS(Y/N) categorical variables Skin_darkening(Y/N)

Chi-square test for Hair_loss(Y/N):
chi-square statistic: 15.4371
p-value:  8.529746219991574e-05
degree of freedom: 1
critical value: 3.8415
Reject H0, there is a relationship between PCOS(Y/N) categorical variables Hair_loss(Y/N)
Reject H0 because there is a relationship between PCOS(Y/N) categorical variables Hair_loss(Y/N)

Chi-square test for Pimples(Y/N):
chi-square statistic: 43.0640
p-value:  5.297702610086416 4e-11
degree of freedom: 1
critical value: 3.8415
Reject H0, there is a relationship between PCOS(Y/N) categorical variables Pimples(Y/N)
Reject H0 because there is a relationship between PCOS(Y/N) categorical variables Pimples(Y/N)

Chi-square test for Fast_food(Y/N):
chi-square statistic: 74.9628
p-value:  4.7966157605881 68e-18
degree of freedom: 1
critical value: 3.8415
Reject H0, there is a relationship between PCOS(Y/N) categorical variables Fast_food(Y/N)
Reject H0 because there is a relationship between PCOS(Y/N) categorical variables Fast_food(Y/N)

Chi-square test for Reg_exercise(Y/N):
chi-square statistic: 1.9982
p-value:  0.15748975012488226
degree of freedom: 1
critical value: 3.8415
Fail to reject H0, there is no relationship between PCOS(Y/N) categorical variables Reg_exercise(Y/N)
Fail to reject H0 because there is no relationship between PCOS(Y/N) categorical variables Reg_exercise(Y/N)
```

Fig. 52. Results of Chi-Square test

```
# Print confusion matrix
print("Confusion Matrix:")
cm = confusion_matrix(y_test, y_test_pred)
print(cm)

# Visualize the confusion matrix
sns.heatmap(cm, annot=True, fmt="d", cmap="Reds")
plt.title("Confusion Matrix")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()

# Calculate and print AUC-ROC score for test data
y_test_pred_proba = clf1.predict_proba(X_test)[:,1]
print("Test Data AUC-ROC Score:", roc_auc_score(y_test, y_test_pred_proba))

# Calculate AUC-ROC score for test data
auc_roc_lr = roc_auc_score(y_test, y_test_pred_proba)

# Plot ROC curve for test data
fpr, tpr, thresholds = roc_curve(y_test, y_test_pred_proba)
plt.plot(fpr, tpr, label="ROC Curve")
plt.plot([0, 1], [0, 1], "k--", label="Random Guess")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title('ROC Curve (AUC = %0.2f)' %auc_roc_lr)
plt.legend()
plt.show()
```

```
Confusion Matrix:
[[146  33]
 [ 63 122]]
```

Fig. 53. Code for Confusion matrix, AUC-ROC Curve of Logistic Regression

```python
# Calculate confusion matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)

# Plot confusion matrix
sns.heatmap(cm, annot=True, fmt="d", cmap="Reds")
plt.title("Confusion Matrix")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()

# Calculate AUC ROC score
auc_roc_gnb = roc_auc_score(y_test, y_pred)
print('AUC ROC score:', auc_roc_gnb)

# Plot ROC curve
fpr, tpr, thresholds = roc_curve(y_test, y_pred)
auc_roc = roc_auc_score(y_test, y_pred)
plt.plot([0, 1], [0, 1], 'k--', label='Random Guessing')
plt.plot(fpr, tpr, label='ROC curve (AUC = %0.2f)' % auc_roc)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve (AUC = %0.2f)' % auc_roc_gnb)
plt.legend()
plt.show()
```

```
[[144  35]
 [ 44 141]]
```

Fig. 54. Code for Confusion matrix, AUC-ROC Curve of Gaussian Naive Bayes

```python
# Print confusion matrix
print("Confusion Matrix:")
cm = confusion_matrix(y_test, y_pred)
print(cm)

# Visualize the confusion matrix
sns.heatmap(cm, annot=True, fmt="d", cmap="Reds")
plt.title("Confusion Matrix")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()

# calculate and print AUC-ROC score
y_pred_proba = clf3.predict_proba(X_test)[:,1]
auc_roc_dt = roc_auc_score(y_test, y_pred_proba)
print("AUC-ROC Score:", auc_roc_dt)

# plot ROC curve
fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
plt.plot(fpr, tpr, label="ROC Curve")
plt.plot([0, 1], [0, 1], "k--", label="Random Guess")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title('ROC Curve (AUC = %0.2f)' %auc_roc_dt)
plt.legend()
plt.show()
```

```
Confusion Matrix:
[[132  44]
 [ 51 137]]
```

Fig. 55. Code for Confusion matrix, AUC-ROC Curve of Decision tree classifier

```python
# print confusion matrix
y_pred = clf4.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(cm)

# plot confusion matrix
sns.heatmap(cm, annot=True, cmap="Reds", fmt="d")
plt.xlabel("Predicted Labels")
plt.ylabel("True Labels")
plt.title("Confusion Matrix")
plt.show()

# calculate and print AUC-ROC score
y_pred_proba = clf4.predict_proba(X_test)[:,1]
auc_roc_svm = roc_auc_score(y_test, y_pred_proba)
print("AUC-ROC Score:", roc_auc_score(y_test, y_pred_proba))

# plot ROC curve
fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
plt.plot(fpr, tpr, label="ROC Curve")
plt.plot([0, 1], [0, 1], "k--", label="Random Guess")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title('ROC Curve (AUC = %0.3f)' %auc_roc_svm)
plt.legend()
plt.show()
```

```
Confusion Matrix:
[[143  36]
 [ 55 130]]
```

Fig. 56. Code for Confusion matrix, AUC-ROC Curve of Support Vector Machine

PREDICTION TESTS

```python
prediction_set = {
                  'Age(yrs)': 28,
                  'Weight(kg)': 65,
                  'FSH/LH': 1.2,
                  'Waist:Hip': 0.8,
                  'AMH': 3.5,
                  'Wt_gain(Y/N)': 0,
                  'Cycle(R/I)': 2,
                  'RBS': 90,
                  'Pregnant_status(Y/N)': 0,
                  'Hair_growth(Y/N)': 1,
                  'Skin_darkening(Y/N)': 0,
                  'Hair_loss(Y/N)': 0,
                  'Fast_food(Y/N)': 1,
                  'Avg_F_size(R)(mm)': 18,
                  'Endometrium(mm)': 8}

# Make predictions on new data
prediction = clf4.predict(pd.DataFrame([prediction_set]))
print("Prediction:", prediction)
```

```
Prediction: [0]
```

Fig. 57. a. Prediction

```
prediction_set1 = {
                   'Age(yrs)': 35,
                   'Weight(kg)': 70,
                   'FSH/LH': 2.5,
                   'Waist:Hip': 1.2,
                   'AMH': 10.0,
                   'Wt_gain(Y/N)': 1,
                   'Cycle(R/I)': 4,
                   'RBS': 120,
                   'Pregnant_status(Y/N)': 1,
                   'Hair_growth(Y/N)': 0,
                   'Skin_darkening(Y/N)': 1,
                   'Hair_loss(Y/N)': 1,
                   'Fast_food(Y/N)': 1,
                   'Avg_F_size(R)(mm)': 30,
                   'Endometrium(mm)': 100}

# Make predictions on new data
prediction = clf4.predict(pd.DataFrame([prediction_set1]))
print("Prediction:", prediction)
```

```
Prediction: [0]
```

Fig. 58. b. Prediction

```
prediction_set = {
                   'Age(yrs)': 30,
                   'Weight(kg)': 150,
                   'FSH/LH': 10.5,
                   'Waist:Hip': 10.9,
                   'AMH': 26.2,
                   'Wt_gain(Y/N)': 1,
                   'Cycle(R/I)': 3,
                   'RBS': 180,
                   'Pregnant_status(Y/N)': 0,
                   'Hair_growth(Y/N)': 1,
                   'Skin_darkening(Y/N)': 1,
                   'Hair_loss(Y/N)': 1,
                   'Fast_food(Y/N)': 1,
                   'Avg_F_size(R)(mm)': 40,
                   'Endometrium(mm)': 20}
# Make predictions on new data
prediction = clf2.predict(pd.DataFrame([prediction_set]))
print("Prediction:", prediction)
```

```
Prediction: [1]
```

Fig. 59. c. Prediction

References

1. Sadeghi, H. M., Adeli, I., Calina, D., Docea, A. O., Mousavi, T., Daniali, M., Nikfar, S., Tsatsakis, A., Abdollahi, M. (2022). Polycystic Ovary Syndrome: A Comprehensive Review of Pathogenesis, Management, and Drug Repurposing. *International journal of molecular sciences*, 23(2), 583. https://doi.org/10.3390/ijms23020583
2. Ndefo, U. A., Eaton, A., Green, M. R. (2013). Polycystic ovary syndrome: a review of treatment options with a focus on pharmacological approaches. *P T : a peer-reviewed journal for formulary management*, 38(6), 336–355.
3. Deswal, R., Narwal, V., Dang, A., Pundir, C. S. (2020). The Prevalence of Polycystic Ovary Syndrome: A Brief Systematic Review. *Journal of human reproductive sciences*, 13(4), 261–271. https://doi.org/10.4103/jhrs.JHRS$_9$5$_1$8
4. Patel, S. (2018). Polycystic ovary syndrome (PCOS), an inflammatory, systemic, lifestyle endocrinopathy. *The Journal of steroid biochemistry and molecular biology*, 182, 27–36. https://doi.org/10.1016/j.jsbmb.2018.04.008
5. N V Durga Prasad K. (2022). *pcos-data* [Data set]. Kaggle. https://doi.org/10.34740/KAGGLE/DSV/3337880