2021

DETECTION & CLASSIFICATION OF PCOD/PCOS USING MACHINE LEARNING ALGORITHMS

ATHULYA SHANTY
2048030
2MDS

CHRIST UNIVERSITY | Bangalore

Detection and Classification of PCOD/ PCOS

INTRODUCTION

The chosen domain is Healthcare in women. Ovaries are an important part in the reproductive system of females. The reproductive system of women is controlled by the complex interplay of primarily five reproductive hormones namely estrogen, gonadotropin-releasing hormone, follicle stimulating hormone, progesterone and luteinizing hormone. An imbalance within these hormones leads to a hormonal disorder called the polycystic ovary syndrome (PCOS) or polycystic ovarian disease (PCOD) among women of reproductive age. The signs and symptoms of this disease include anovulation, menstrual dysfunction and signs of hyperandrogenism. Other signs and symptoms include hirsutism, infertility, obesity, metabolic syndrome, and diabetes .Women with PCOS/PCOD have high chances of hypertension. PCOS/PCOD is clinically diagnosed with an ultrasound abdomen scanning.

My aim is to build a model using machine learning algorithms which provides at most accuracy to predict the presence of PCOD/PCOS in women. This project aims to detect the polycystic ovaries Disease/Syndrome in the ovaries of a woman in order to recommend an immediate attention and need of diagnosis through a set of questionnaires. The diagnosis of PCOS/PCOD is important as this disease can cause problems with menstrual periods and make it difficult for females to conceive. If not treated it can cause insulin resistant diabetes, obesity and high cholesterol leading to heart disease.

This project aims to give an alert or warning to the females with PCOS/PCOD at its earliest stage possible. Even though there are no certain medicines for this syndrome/disease, lifestyle modifications such as diet, exercise and weight loss are considered to be the first-line treatment for women with PCOS/PCOD.

ABOUT THE DATA

The data was collected from a population of females living across the world. The source of data collection was primary and was carried out using Google forms which were distributed with the help of social media and messaging services. The google form contained 6 sections such as the main section, Personal details, Symptoms, Personal details & history, lifestyle and Medical history which contain sub questions that is curated to identify certain key criteria of PCOS/ PCOD which in turn help in developing a model to predict if the female has PCOD / PCOS or not. The collected data contains 882 rows and 27 observations.

IMPLEMENTATION DETAILS

The raw dataset contained 882 observations and 27 variables. The lengthy column names were renamed to shorter ones. The duplicate values in the dataset were removed. Even though the desired population is women, the survey was also attended by some of the males whom must be removed for further analysis. Therefore there comes a need to filter the dataset such that the required dataset contains only the data from females. The same was done by removing the observations with ‘other’ and ‘male’ as gender. Also, there were some unwanted variables for our study and they were removed. There were missing data in the dataset. Dropping the missing values was not a solution as it dropped the entire dataset and those were handled by replacing the null values with the string ‘unknown’. The column named ‘kids’ contained both numerical and string data which will result in errors in further processes. Therefore the numbers 0,1,2 were replaced to strings ‘zero’, ‘one’ and ’two’. After cleaning the data, the observations got reduced to 723 and the variables got reduced to 24.

The dataset had 511 observations which contained only independent variables whereas 212 observations contained both independent variables and dependent variable. These were separated into two different data frames for further analysis. There are no numerical variables found in the dataset whereas there were 24 categorical variables. These 24 categorical variables were converted to numerical variables by label encoding. The dataframe which contained both dependent and independent variables were divided into dependent variable and independent variables.

The dependent variable is ‘diagnosis’ and independent variables are 'age', 'body_type', 'scanning', 'before_period', 'irregular_period', 'painful_period', 'bleeding', 'period_cycle', 'period_duration', 'period_pain', 'clots', 'alcohol', 'smoking', 'stress', 'exercise', 'Hereditary', 'diabetes', 'hypothyroidism', 'hair_growth', 'acne', 'marital_status', 'kids' and 'work'. Different bar plots were plotted with their frequency for unique category of the categorical variables for a better understanding of the dataset.

The correlation matrix gave the correlation between the variables and the heatmap of the same gave a better way of visualization. The relevant features for the study were chosen based on the correlation between variables. The variables with correlation more than 0.2 were considered to be the important variables.

The variable to be predicted was categorical and hence different algorithms such as Naive Bayes classification, Logistic regression, KNN classification, Random Forest classifier and Decision tree classifier were used for modelling the data. The data frame which have both the independent and

dependent variables were splitted to X_train, y_train, X_test and y_test. Each model was trained by inputting X_train, y_train values of the training dataset. Model evaluation and the confusion matrix were drawn for each algorithm.

The data frame which had only dependent variables were inputted to the two models with the highest accuracy for predicting the diagnosis. The predicted diagnosis by the models with highest accuracies were also compared.

RESULTS AND DISCUSSION

- The lengthy column names was renamed to shorter ones.

```
#Reading the dataset
df=pd.read_excel(r'E:\files\Desktop\2MDS\MACHINE_LEARNING\projects\uncleaned_data.xlsx')
df.head()
```

Timestamp	What is your name?	Select your Gender	Pick your age limit	How would you describe your Body Physic ?	Have you done an ultrasound abdomen scanning and what does your report say?	Do you notice any of these right before your period begins?	Are you experiencing irregular or late periods?	Are you experiencing painful periods?	Are you experiencing Excessive bleeding?	...	Do you exercise regularly?	Is your mother diagnosed with PCOS/PCOD?	Do you suffe from diabetes'

```
df.rename(renamed, axis='columns', inplace=True)
df.head()
```

Timestamp	name	gender	age	body_type	scanning	before_period	irregular_period	painful_period	bleeding	...	exercise	Hereditary	diabetes	hypo

- Removing duplicates removed 113 duplicate records and resulted in 763 observations with 27 variables.

```
df.shape
```

```
(882, 27)
```

```
#Removing duplicates
df=df.drop_duplicates(subset=['name'])
df.shape
```

```
(763, 27)
```

- Dropping ‘male’ and ‘other’ from gender resulted in 723 observations with 27 variables.

```
#Removing males and other categories from gender variable.
df.drop(df[df['gender'] == 'Male'].index, inplace = True)
df.drop(df[df['gender'] == 'Other'].index, inplace = True)
```

```
df.shape
```

```
(723, 27)
```

- Removal of unwanted columns such as ‘Timestamp’, ’name’, etc resulted in 723 observations with 24 variables.

```
df=df.drop(['Timestamp','name','gender'],axis=1)
df.shape
```

```
(723, 24)
```

- Handling missing data was done by filling the null values with the variable ‘unknown’.

```
#Filling missing values with 'unknown'
df['marital_status']=df['marital_status'].fillna('unknown')
df['work']=df['work'].fillna('unknown')
df['kids']=df['kids'].fillna('unknown')
df['diagnosis']=df['diagnosis'].fillna('Didnt check')
```

- The dataset was splitted into 2 data frames. One for training and the other for testing. The observations which had both dependent and independent variables were added to 1st data frame for testing and the observations with only independent variables were added to 2nd data frame. The observations in which the response of ‘Didn’t check’ has been added to data frame 1 for prediction.

```
df2=df1=df
```

```
#Data for prediction
df1=df1[df1['diagnosis'] == 'Didnt check']
#Dropping diagnosis column of predictive dataset as it contains only 'Didnt check'
df1=df1.drop(['diagnosis'],axis=1)
df1.shape
```

```
(511, 23)
```

```
#Data for modeling
df2.drop(df2[df2['diagnosis'] == 'Didnt check'].index, inplace = True)
df2.shape
```

```
(212, 24)
```

- The dataset contained 24 categorical variables and no numerical variables.

```
# List of numerical variables
numerical = [feature for feature in df.columns if ((df[feature].dtypes != 'O') & (feature not in ['deposit']))]
print('Number of numerical variables: ', len(numerical))

print(numerical)
```

```
Number of numerical variables:  0
[]
```

```
#List of categorical variables
categorical=[]
for col in df.select_dtypes(include='object').columns:
    categorical.append(col)
print('Number of categorical variables: ', len(categorical))
print(categorical)
```

```
Number of categorical variables:  24
['age', 'body_type', 'scanning', 'before_period', 'irregular_period', 'painful_period', 'bleeding', 'period_cycle', 'period_dur
ation', 'period_pain', 'clots', 'alcohol', 'smoking', 'stress', 'exercise', 'Hereditary', 'diabetes', 'hypothyroidism', 'hair_g
rowth', 'acne', 'marital_status', 'kids', 'work', 'diagnosis']
```

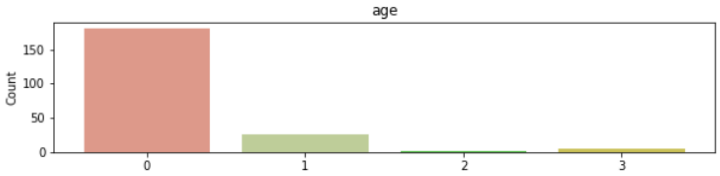
- The categorical variables of both the data frames have been converted to numerical by Label Encoding.

```
#Label Encoding for categorical variables in df1
from sklearn import preprocessing
for i in range(0,(len(categorical)-1)):
    le=preprocessing.LabelEncoder()
    le.fit(df1[categorical[i]])
    df1[categorical[i]]=le.transform(df1[categorical[i]])
```

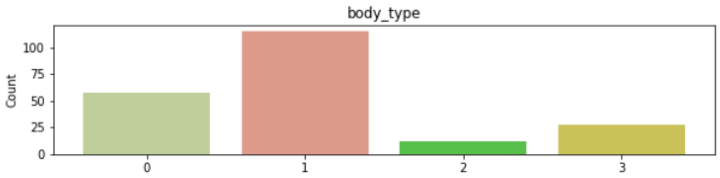
```
#Label Encoding for categorical variables in df2
from sklearn import preprocessing
for i in range(0,len(categorical)):
    le=preprocessing.LabelEncoder()
    le.fit(df2[categorical[i]])
    df2[categorical[i]]=le.transform(df2[categorical[i]])
```

- The count of each value of every variables were plotted in a bar graph.

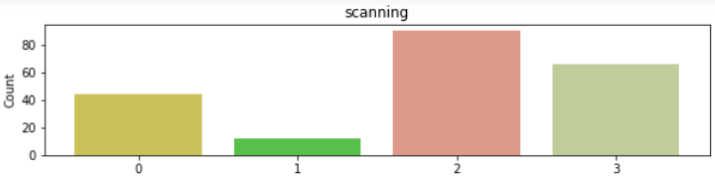
```
colors = ['#DD998A','#BECD99','#CAC158','#58BF4B']
import matplotlib.pyplot as plt
def bar_plot(variable):
    var = df[variable]
    varValue = var.value_counts()
    plt.figure(figsize = (10,2))
    plt.bar(varValue.index,varValue,color=colors)
    plt.xticks(varValue.index,varValue.index.values)
    plt.ylabel("Count")
    plt.title(variable)
    plt.show()
    print("{}: \n {}".format(variable,varValue))

for c in categorical:
    bar_plot(c)
```



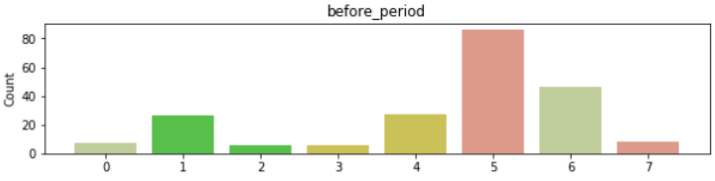

```
age:
 0    180
1     26
3      4
2      2
Name: age, dtype: int64
```




```
body_type:
 1    115
0     58
3     27
2     12
Name: body_type, dtype: int64
```

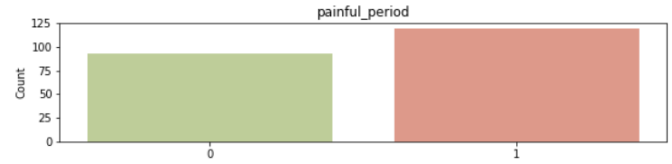



```
scanning:
 2    90
3    66
0    44
1    12
Name: scanning, dtype: int64
```

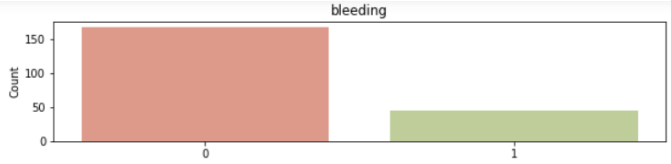



```
before_period:
 5    86
6    46
4    27
1    26
7     8
0     7
3     6
2     6
Name: before_period, dtype: int64
```

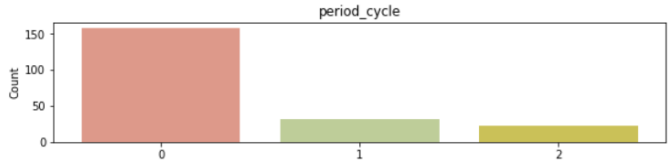

```
irregular_period:
 0    118
1     94
Name: irregular_period, dtype: int64
```



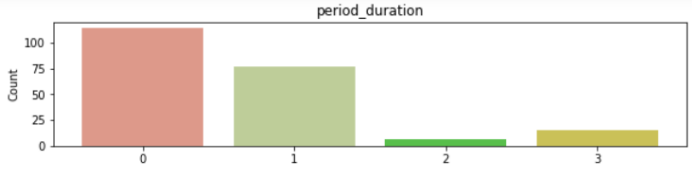

```
painful_period:
 1    119
0     93
Name: painful_period, dtype: int64
```



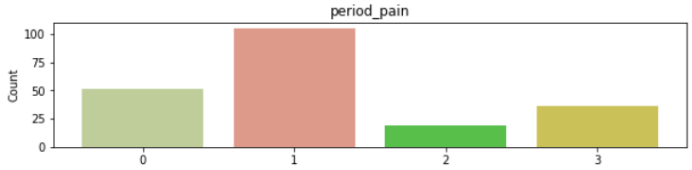

```
bleeding:
 0    167
1     45
Name: bleeding, dtype: int64
```



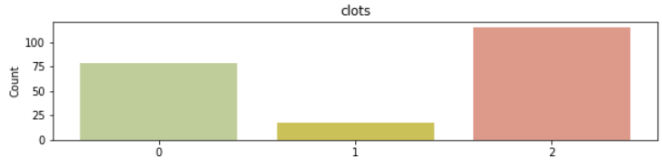

```
period_cycle:
 0    158
1     31
2     23
Name: period_cycle, dtype: int64
```



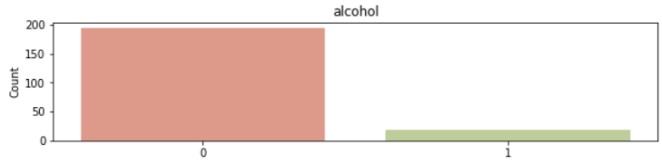

```
period_duration:
 0    114
1     77
3     15
2      6
Name: period_duration, dtype: int64
```



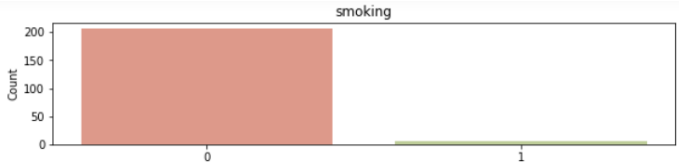

```
period_pain:
 1    105
0     52
3     36
2     19
Name: period_pain, dtype: int64
```



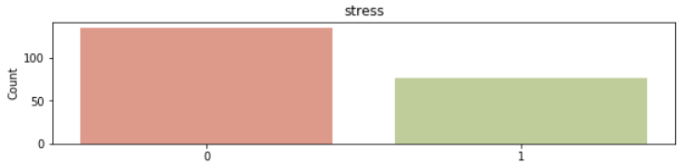

```
clots:
 2    115
0      79
1      18
Name: clots, dtype: int64
```



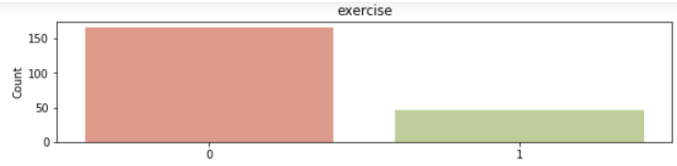

```
alcohol:
 0    194
1     18
Name: alcohol, dtype: int64
```



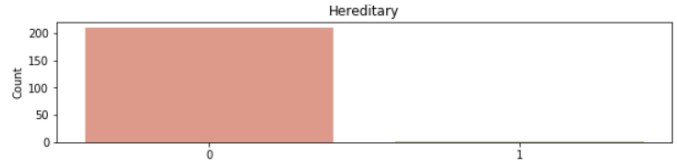

```
smoking:
 0    206
1      6
Name: smoking, dtype: int64
```



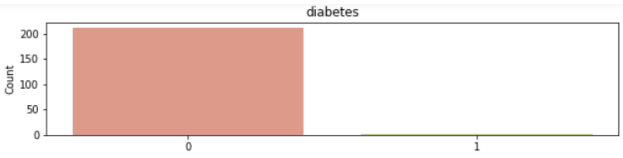

```
stress:
 0    135
1     77
Name: stress, dtype: int64
```



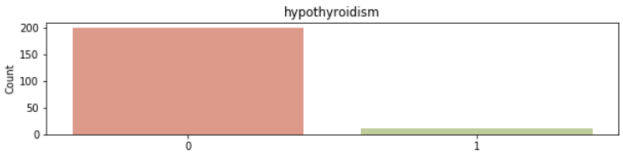

```
exercise:
 0    166
1     46
Name: exercise, dtype: int64
```



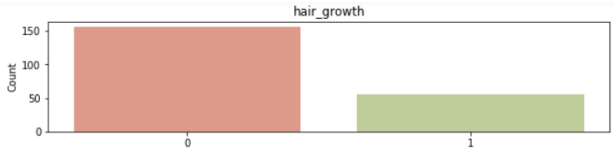

```
Hereditary:
 0    210
1      2
Name: Hereditary, dtype: int64
```



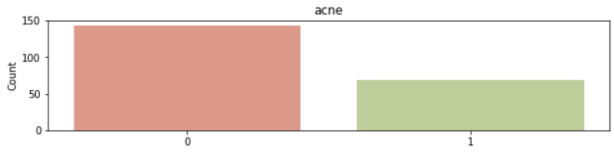

```
diabetes:
 0    211
1      1
Name: diabetes, dtype: int64
```



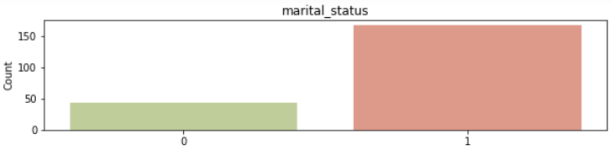

```
hypothyroidism:
 0    200
1     12
Name: hypothyroidism, dtype: int64
```



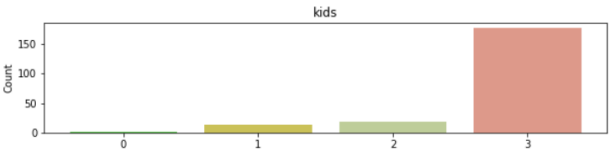

```
hair_growth:
 0    156
1     56
Name: hair_growth, dtype: int64
```



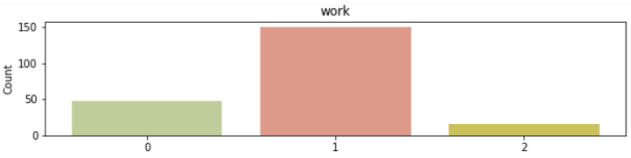

```
acne:
 0    143
1     69
Name: acne, dtype: int64
```



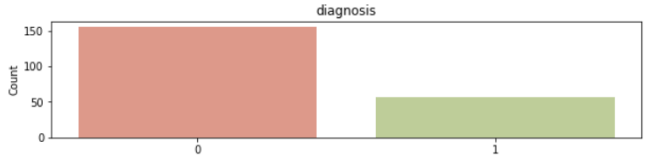

```
marital_status:
 1    168
0     44
Name: marital_status, dtype: int64
```




```
kids:
 3    177
2     19
1     14
0      2
Name: kids, dtype: int64
```




```
work:
 1    150
0     47
2     15
Name: work, dtype: int64
```

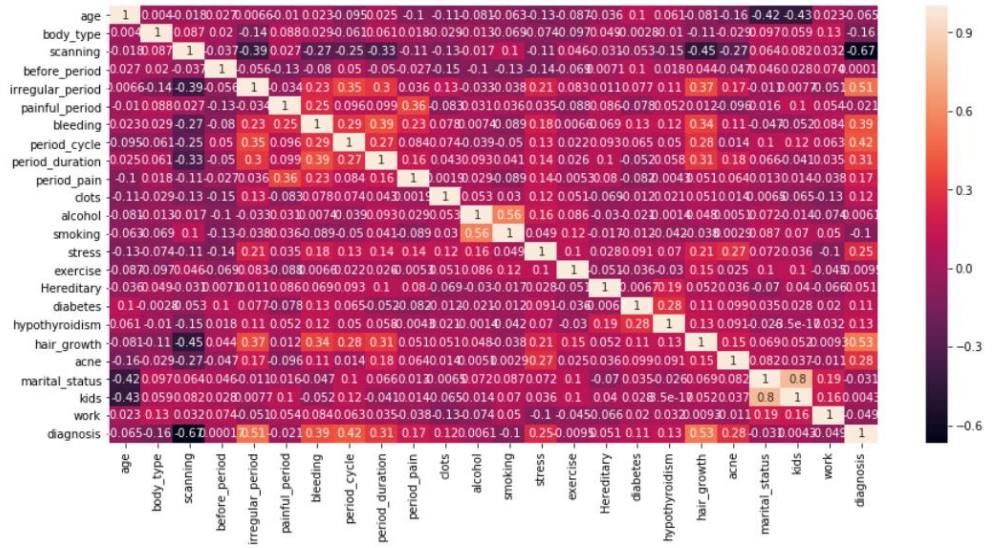



```
diagnosis:
 0    155
1     57
Name: diagnosis, dtype: int64
```

- Correlation heat map has been plotted.

In [32]:
```
## Checking for correlation
import matplotlib.pyplot as plt
import seaborn as sns
cor_mat=df2.corr()
fig = plt.figure(figsize=(15,7))
sns.heatmap(cor_mat,annot=True)
```

Out[32]: <matplotlib.axes._subplots.AxesSubplot at 0x28fb04da400>



- The relevant features was found to be 'scanning', 'irregular_period', 'bleeding', 'period_cycle', 'stress', 'hair_growth' and 'acne' through the method of correlation with cut-off as 0.2.

```
#Correlation with output variable
cor=df2.corr()
cor_target = abs(cor['diagnosis'])
#Selecting highly correlated features
relevant_features = cor_target[cor_target>0.2]
print('There are ',(len(relevant_features)-1),'relevant features out of which diagnosis is the target variable')
relevant_features[0:(len(relevant_features)-1)]

```

```
There are  8 relevant features out of which diagnosis is the target variable

scanning            0.666752
irregular_period    0.508112
bleeding            0.387688
period_cycle        0.417865
period_duration     0.314478
stress              0.249914
hair_growth         0.529517
acne                0.282646
Name: diagnosis, dtype: float64
```

```
relevant_features=['scanning','irregular_period','bleeding','period_cycle','stress','hair_growth','acne']
print(relevant_features)
```

```
['scanning', 'irregular_period', 'bleeding', 'period_cycle', 'stress', 'hair_growth', 'acne']
```

- The dataset which have both dependent and independent variable was splitted into Xtrain, ytrain, Xtest and ytest for modelling.

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

```
X_train.shape
```

```
(169, 23)
```

```
X_test.shape
```

```
(43, 23)
```

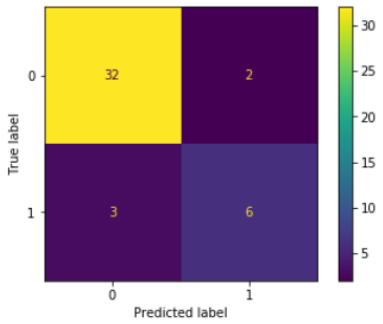
- There were a total of 5 models. They are Naïve Bayesian model, Logistic Regression model, KNN model, Random Forest Classification model, and Decision Tree model with an accuracy of 88.37%, 90.70%, 90.70%, 88.37% and 90.7% respectively. Confusion matrix for each matrix was plotted.

```
#Naive Bayesian Classifier
clf=GaussianNB()
model=clf.fit(X_train,y_train)
y_pred=model.predict(X_test)
ac=accuracy_score(y_test,y_pred,normalize=True)
a=np.round(ac*100,2)
print('Accuracy is ',a,'%')
```

```
Accuracy is  88.37 %
```

```
#Model Evaluation of Naive Bayesian Classifier
print(f"Score in Test Data : {model.score(X_test,y_test)}")

cm=confusion_matrix(y_test, y_pred)
p_right=cm[0][0]+cm[1][1]
p_wrong=cm[0][1]+cm[1][0]

print(f"Right classification : {p_right}")
print(f"Wrong classification : {p_wrong}")

plot_confusion_matrix(clf, X_test, y_test)
plt.show()
```

```
Score in Test Data : 0.8837209302325582
Right classification : 38
Wrong classification : 5
```

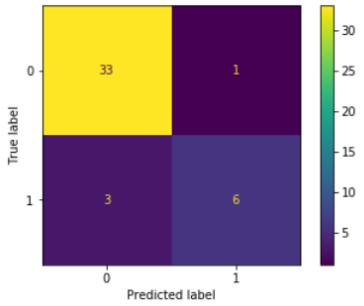



```
#Logistic Regression
clf=LogisticRegression()
model=clf.fit(X_train,y_train)
y_pred=model.predict(X_test)
ac=accuracy_score(y_test,y_pred,normalize=True)
b=np.round(ac*100,2)
print('Accuracy is ',b,'%')
```

```
Accuracy is  90.7 %
```

```
#Model Evaluation of logistic Regression
print(f"Score in Test Data : {model.score(X_test,y_test)}")

cm=confusion_matrix(y_test, y_pred)
p_right=cm[0][0]+cm[1][1]
p_wrong=cm[0][1]+cm[1][0]

print(f"Right classification : {p_right}")
print(f"Wrong classification : {p_wrong}")

plot_confusion_matrix(clf, X_test, y_test)
plt.show()
```

```
Score in Test Data : 0.9069767441860465
Right classification : 39
Wrong classification : 4
```

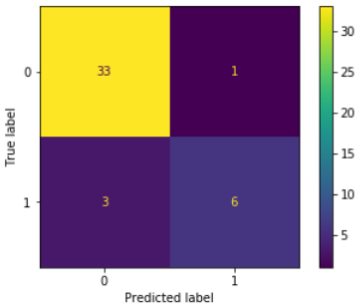



```
#KNN
clf=neighbors.KNeighborsClassifier(n_neighbors=3)
model=clf.fit(X_train,y_train)
y_pred=model.predict(X_test)
ac=accuracy_score(y_test,y_pred,normalize=True)
c=np.round(ac*100,2)
print('Accuracy is ',c,'%')
```

```
Accuracy is  90.7 %
```

```
#Model Evaluation of KNN
print(f"Score in Test Data : {model.score(X_test,y_test)}")

cm=confusion_matrix(y_test, y_pred)
p_right=cm[0][0]+cm[1][1]
p_wrong=cm[0][1]+cm[1][0]

print(f"Right classification : {p_right}")
print(f"Wrong classification : {p_wrong}")

plot_confusion_matrix(clf, X_test, y_test)
plt.show()
```

```
Score in Test Data : 0.9069767441860465
Right classification : 39
Wrong classification : 4
```

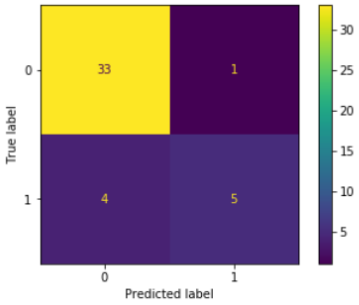



```
#Random Forest Classifier
clf = RandomForestClassifier(max_depth=2, random_state=0)
model=clf.fit(X_train, y_train)
y_pred=model.predict(X_test)
ac=accuracy_score(y_test,y_pred,normalize=True)
d=np.round(ac*100,2)
print('Accuracy is ',d,'%')
```

```
Accuracy is  88.37 %
```

```python
#Model Evaluation of Random Forest Classifier
print(f"Score in Test Data : {model.score(X_test,y_test)}")

cm=confusion_matrix(y_test, y_pred)
p_right=cm[0][0]+cm[1][1]
p_wrong=cm[0][1]+cm[1][0]

print(f"Right classification : {p_right}")
print(f"Wrong classification : {p_wrong}")

plot_confusion_matrix(clf, X_test, y_test)
plt.show()
```

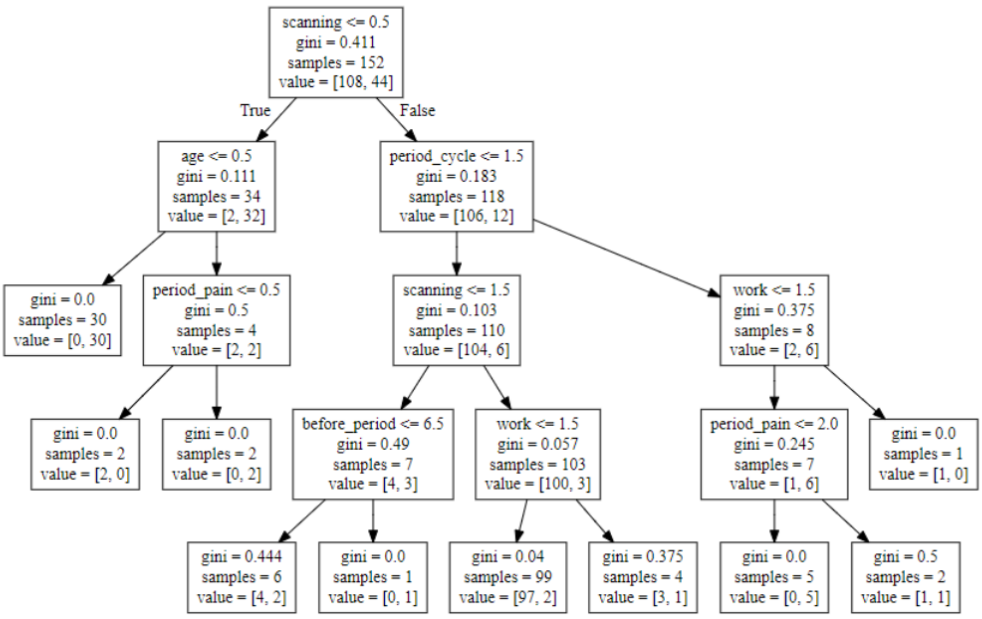
```
Score in Test Data : 0.8837209302325582
Right classification : 38
Wrong classification : 5
```




```python
#Decision Tree
clf = tree.DecisionTreeClassifier(max_depth=4)
model = clf.fit(X_train, y_train)
y_pred=model.predict(X_test)
ac=accuracy_score(y_test,y_pred,normalize=True)
e=np.round(ac*100,2)
print('Accuracy is ',e,'%')
```

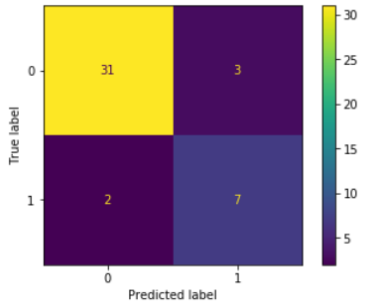
```
Accuracy is  88.37 %
```

```python
import graphviz
with open("DecisionTree.dot",'w') as f:
    f=tree.export_graphviz(model,feature_names=X.columns,out_file=f);
```



```
#Model Evaluation of Decision Tree
print(f"Score in Test Data : {model.score(X_test,y_test)}")

cm=confusion_matrix(y_test, y_pred)
p_right=cm[0][0]+cm[1][1]
p_wrong=cm[0][1]+cm[1][0]

print(f"Right classification : {p_right}")
print(f"Wrong classification : {p_wrong}")

plot_confusion_matrix(clf, X_test, y_test)
plt.show()
```

```
Score in Test Data : 0.8837209302325582
Right classification : 38
Wrong classification : 5
```



- The best models were found to be Logistic Regression and KNN with 90.70% of accuracy.

```
dictionary={'Algorithm':['Naive Bayesian Classifier','Logistic regression','KNN','Random Forest Classifier','Decision Tree Cl
          'Accuracy':[a,b,c,d,e]}
n=pd.DataFrame(dictionary)
n.sort_values(by=['Accuracy'],ascending=False)
```

	Accuracy	Algorithm
1	90.70	Logistic regression
2	90.70	KNN
0	88.37	Naive Bayesian Classifier
3	88.37	Random Forest Classifier
4	88.37	Decision Tree Classifier

- The diagnosis were predicted for the test data using the models with high accuracy.

In [56]:
```
#Prediction using Logistic Regression
clf=LogisticRegression()
model=clf.fit(X,y)
y_pred=model.predict(X2)
df3=df1
df4=pd.DataFrame(y_pred)
pred_diag={0:'Predicted Diagnosis'}
df4.rename(pred_diag, axis=1, inplace=True)
df4.head(10)
```

Out[56]:

	Predicted Diagnosis
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	1
9	1

```
#Prediction using KNN
clf=neighbors.KNeighborsClassifier(n_neighbors=3)
model=clf.fit(X, y)
y_pred=model.predict(X2)
df6=df1
df7=pd.DataFrame(y_pred)
pred_diag={0:'Predicted Diagnosis'}
df7.rename(pred_diag, axis=1, inplace=True)
df7.head(10)
```

	Predicted Diagnosis
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	1
9	0

- The diagnosis predicted by both the Logistic Regression model and KNN model were compared. 485 predictions were predicted the same by both the models and 26 predictions were predicted differently by both the models.

```
df9=df4==df7
df9['Predicted Diagnosis'].value_counts()
```

```
True     485
False     26
Name: Predicted Diagnosis, dtype: int64
```

LIMITATIONS

- A minimum of 20 cysts in the ovaries were considered to be PCOD/PCOS. But we were not able to collect the number of cysts in each ovary which would have made the model much more accurate.
- The training data set was comparatively very less to the testing dataset. Reaching out to the ones with the disease should have been increased for a better accuracy and prediction.

CONCLUSION

Hormone imbalances affect a woman's health in many different ways. PCOS/PCOD can increase the risk of infertility, metabolic syndrome, sleep apnea, endometrial cancer, and depression. A healthy lifestyle, eating a healthy diet and exercising regularly will help the women with PCOS/PCOD.A model with 90.70% of accuracy has been built for the prediction of PCOD/PCOS. The models Logistic Regression and KNN performed well than Naïve Bayesian, Random Forest, and Decision tree. This models with high accuracy helps in the diagnosis of the syndrome/disease at it earliest stage

possible and would be able to warn them beforehand. 485 predictions were same by both models whereas 26 predictions were not the same.

FUTURE WORK

- I am planning to further continue this project by collecting more data which would increase the accuracy rate and improve the prediction.
- I am planning to perform unsupervised learning algorithm on the data frame where dependent variable is absent. I also wanted to compare the predictions of both supervised and unsupervised learning algorithms.
- I am planning to develop a mobile application or web page which will take all the variables as inputs and give the predictions as output with a warning message. The app will also display the methods to be performed for recovering from PCOS/PCOD.

BIBLIOGRAPGHY & REFERENCES

https://www.healthline.com/health/polycystic-ovary-disease#health-effects

https://www.yashodahospitals.com/diseases-treatments/pcod-pcos-symptoms-causes-treatment/