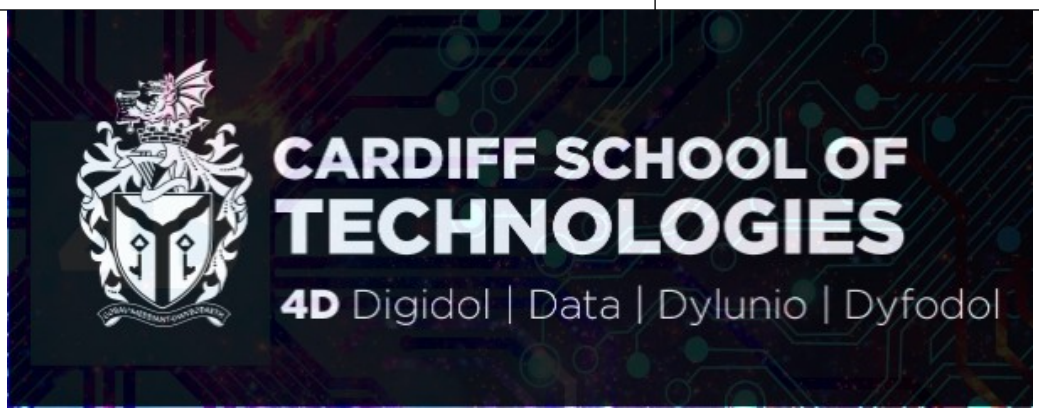
Cardiff Metropolitan University	
Cardiff School of Technologies	
Academic Year: 2021/2022	
Term: 1	
Module Name: Programming for data analysis	
Module Code: CIS7031	
MSc Programme: Data Science	
Assignment Title: Data analysis on Cervical cancer	
Student Name: Mohammad Aziz	**Student ID: 20192233**
Data Submitted: 03/12/2021	**Mark:**
Feedback:	
Signature:	**Date:**



Contents

1. Introduction....3
2. Overview of the data....3
3. List of the important columns....3
4. Library used....4
5. Imorting Libraries....5
6. Loading datasets....5
7. Data preparation....5
7.1 Raw data analysis....6
7.2 Data Cleaning....7
8. Data Visualization....10
8.1 Count plot graph....10
8.2 Density plot graph....11
9. Feature selection....18
9.1 Benefit of using feature selection....18
10. Feature Importance....19
11. Splitting the data....20
12. Modelling Creation....21
12.1 Support vector machine....22
12.1.1 Model Evaluation for SVM....23
12.2 K-Nearest neighbors....24
12.2.1 Model Evaluation for KNN....25
13. Comparison of SVM and KNN model....26
14. Critical reviews and techniques used....27
15. References....28

1. INTRODUCTION

Cervical cancer is a deadly disease that occurs in the open area from the neck of the womb/uterus to the vegina. It is also known as cervix cancer. The symptoms of this cancer is bleeding of vegina abnormally during periods and after sex. It can be cured using chemotherapy and radiotherapy surgery at the early stage of cancer but being honest cancer cells never die even after surgery, cancer cells replicate at a very fast rate and come up in your body parts at any time or we can say cancer cells are unpredictable.
Science and technologies are growing very fast day by day and It has been 50 years since cancer came into notice to the world but we have not found any medicine/pills or any treatment or scientific method yet that can cure cancer 100%. The treatment that we have right now can't be cured 100%, moreover they used to reduce the lifetime of a person.

Looking into this growing cancer, it is analysed on "cervical cancer" data available on the internet (Kaggle) by using tools and technologies and visualising beautifully by making various graphs, charts etc. AI and machine learning made this possible for the world to know and predict if someone has cancer or not by using required, maintained or organised data. Therefore, in this report I have tried to predict whether a patient has cervical cancer cells in their body or not with the help of two different machine learning models.

2. Overview of the Dataset

The dataset is all about a specific kind of cancer known as "Cervical cancer". There are a total 36 columns and 858 rows and this dataset is fetched from an open source big data library called Kaggle. Currently, cervical cancer datasets are raw datasets that means it has many unwanted or unnecessary things that need to be cleaned or sometimes need to modify in order to make it usable for the final machine learning or predictive models.

3. List of the Important Columns

1. Age: This column contains the age of the people that are infected with cervical cancer.

2. Number of Sexual partners: It carries number of partners(male and female) that stays together in order to have sex.
3. First Sexual Intercourse: It basically shows that at what age a female person had her first sex with her partner.
4. Num of pregnancies: This column is about how many times a female gets pregnant.
5. Smokes: Partners that take tobacco as smoke inside their body. This column has either 0 or 1. 0 means partner does not take tobacco however 1 indicates that partner does take tobacco.
6. Smokes(year): This column represents the total number of years that an individual took tobacco.
7. Smokes(packs/year): It is about how many tobacco packs per year have been taken by an individual.
8. Hormonal Contraceptives: Sexual partners uses some methods in order to keep birth control. This column contains 0 and 1. 0 represents no birth control and 1 means they use birth control methods.

4. Libraries Used

There are various python libraries that have been utilized for this assignment in order to analyse, visualise and predict accurately on the cervical cancer datasets. Jupyter notebook, an open source web application is used for implementing all the codes in python and all the necessary python and ML libraries.

Pandas: Pandas is an open source library which is mainly written in python programming language. Pandas library is mainly used for handling massive data for analysis and manipulation. There are various data structures in pandas that are very easy to use like it uses single word data structure and the data is automatically organised.

Matplotlib: It is also an open source python library that is mainly used for visualising the data easily by plotting various kinds of interactive graphs, charts, box plots, histogram, scatter plot, etc. And this library is mainly based on NumPy.

Numpy: Numpy is the python library which is based on high level mathematics. two dimensional, three dimensional and multi dimensional arrays are created easily with less time and memory.

5. Import libraries

Importing libraries is something that is very very important to begin analysis of data. Here it is imported a list of libraries shown in the figure[1].

```
import pandas as pnd
import numpy as nmp
import matplotlib.pyplot as mplt
import seaborn as sbrn
import sklearn.feature_selection as fs
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.calibration import CalibratedClassifierCV
from sklearn.svm import LinearSVC
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import roc_curve
import warnings
```

Figure[1]

6. Loading Datasets

It is compulsory to load the dataset in the pandas data frame using read_csv() if the file has comma separated values given as an example in the figure[2]. Read_csv is a pandas function that reads csv files from file location and puts whole csv files into the pandas data frame that looks beautiful for data analysis.

```
pnd_data_frame=pnd.read_csv("kag_risk_factors_cervical_cancer.csv")
```

figure[2]

Here the name of the csv file is “kag_risk_factors_cervical_cancer.csv” and it is written inside a function called read_csv. I used dot after “pnd” because any pandas function is used to access only by using dot and then name of the function.

7. Data Preparation

Data preparation is one of the most important parts in data analysis. It is the process of cleaning data, modifying, deleting useless cells and rows, removing duplicates, etc and making perfect datasets for visualisation and use for machine learning models.

7.1. Raw Data Analysis: It is the process of checking the original datasets if there are empty cells, useless cells, null values, cells that have symbols, type of data or columns, etc.

First step is to have an overview of the data by using pandas function info(). info() is the function that is used to show the list of the columns, counts number of rows for each column, data types for each column, counts number of data types, memory usages and also if the columns are Null or not.

```
pnd_data_frame.info();
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 858 entries, 0 to 857
Data columns (total 36 columns):
 #   Column                              Non-Null Count  Dtype 
---  ------                              --------------  ----- 
 0   Age                                 858 non-null    int64 
 1   Number of sexual partners           858 non-null    object
 2   First sexual intercourse            858 non-null    object
 3   Num of pregnancies                  858 non-null    object
 4   Smokes                              858 non-null    object
 5   Smokes (years)                      858 non-null    object
 6   Smokes (packs/year)                 858 non-null    object
 7   Hormonal Contraceptives             858 non-null    object
 8   Hormonal Contraceptives (years)     858 non-null    object
 9   IUD                                 858 non-null    object
 10  IUD (years)                         858 non-null    object
 11  STDs                                858 non-null    object
 12  STDs (number)                       858 non-null    object
 13  STDs:condylomatosis                 858 non-null    object
 14  STDs:cervical condylomatosis        858 non-null    object
 15  STDs:vaginal condylomatosis         858 non-null    object
 16  STDs:vulvo-perineal condylomatosis  858 non-null    object
 17  STDs:syphilis                       858 non-null    object
 18  STDs:pelvic inflammatory disease    858 non-null    object
 19  STDs:genital herpes                 858 non-null    object
 20  STDs:molluscum contagiosum          858 non-null    object
 21  STDs:AIDS                           858 non-null    object
 22  STDs:HIV                            858 non-null    object
 23  STDs:Hepatitis B                    858 non-null    object
 24  STDs:HPV                            858 non-null    object
 25  STDs: Number of diagnosis           858 non-null    int64 
 26  STDs: Time since first diagnosis    858 non-null    object
 27  STDs: Time since last diagnosis     858 non-null    object
 28  Dx:Cancer                           858 non-null    int64 
 29  Dx:CIN                              858 non-null    int64 
 30  Dx:HPV                              858 non-null    int64 
 31  Dx                                  858 non-null    int64 
 32  Hinselmann                          858 non-null    int64 
 33  Schiller                            858 non-null    int64 
 34  Citology                            858 non-null    int64 
 35  Biopsy                              858 non-null    int64 
dtypes: int64(10), object(26)
memory usage: 241.4+ KB
```

Figure[3]

From the figure[3], it can be seen that there are 36 columns and 858 rows in the cervical cancer datasets however, 10 columns and the rest 26 are integer data types and object data types respectively. The size of the whole datasets are approximately 241.4KB.

Before I begin with the cleaning, I need to have a look over the data in the pandas data frame using head(), tail() and sample() functions. Function head() is used to see 5 numbers of rows from top of the data. Function tail() is used to see 5 numbers of rows

from the bottom of the dataframe. However, we can put any positive integer in the brackets to see the required number of rows. sample() is also a pandas function that is used to extract 5 numbers of rows randomly from the datasets. And it would be a great idea to see dirt in the dataframe.

```
pnd_data_frame.head(7)
```

	Age	Number of sexual partners	First sexual intercourse	Num of pregnancies	Smokes	Smokes (years)	Smokes (packs/year)	Hormonal Contraceptives	Hormonal Contraceptives (years)	IUD	...	STDs: Time since first diagnosis	STDs: Time since last diagnosis	Dx:Cancer	Dx:CIN
0	18	4.0	15.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	?	?	0	0
1	15	1.0	14.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	?	?	0	0
2	34	1.0	?	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	?	?	0	0
3	52	5.0	16.0	4.0	1.0	37.0	37.0	1.0	3.0	0.0	...	?	?	1	0
4	46	3.0	21.0	4.0	0.0	0.0	0.0	1.0	15.0	0.0	...	?	?	0	0
5	42	3.0	23.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	...	?	?	0	0
6	51	3.0	17.0	6.0	1.0	34.0	3.4	0.0	0.0	1.0	...	?	?	0	0

7 rows × 36 columns

```
pnd_data_frame.tail(7)
```

	Age	Number of sexual partners	First sexual intercourse	Num of pregnancies	Smokes	Smokes (years)	Smokes (packs/year)	Hormonal Contraceptives	Hormonal Contraceptives (years)	IUD	...	STDs: Time since first diagnosis	STDs: Time since last diagnosis	Dx:Cancer	Dx:CIN
851	23	2.0	15.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	...	?	?	0	(
852	43	3.0	17.0	3.0	0.0	0.0	0.0	1.0	5.0	0.0	...	?	?	0	(
853	34	3.0	18.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	?	?	0	(
854	32	2.0	19.0	1.0	0.0	0.0	0.0	1.0	8.0	0.0	...	?	?	0	(
855	25	2.0	17.0	0.0	0.0	0.0	0.0	1.0	0.08	0.0	...	?	?	0	(
856	33	2.0	24.0	2.0	0.0	0.0	0.0	1.0	0.08	0.0	...	?	?	0	(
857	29	2.0	20.0	1.0	0.0	0.0	0.0	1.0	0.5	0.0	...	?	?	0	(

7 rows × 36 columns

```
pnd_data_frame.sample(7)
```

	Age	Number of sexual partners	First sexual intercourse	Num of pregnancies	Smokes	Smokes (years)	Smokes (packs/year)	Hormonal Contraceptives	Hormonal Contraceptives (years)	IUD	...	STDs: Time since first diagnosis	STDs: Time since last diagnosis	Dx:Cancer	Dx:CI
259	23	8.0	15.0	1.0	1.0	10.0	7.5	1.0	8.0	0.0	...	?	?	0	
766	21	1.0	14.0	2.0	0.0	0.0	0.0	1.0	4.0	0.0	...	?	?	0	
588	45	5.0	15.0	7.0	0.0	0.0	0.0	1.0	0.66	0.0	...	?	?	0	
597	19	2.0	15.0	2.0	0.0	0.0	0.0	1.0	1.0	0.0	...	5.0	2.0	0	
251	23	1.0	17.0	2.0	0.0	0.0	0.0	1.0	0.5	0.0	...	?	?	0	
782	29	3.0	15.0	2.0	1.0	10.0	0.5132021277	1.0	0.25	0.0	...	?	?	0	
620	24	3.0	18.0	2.0	1.0	5.0	0.5132021277	1.0	1.0	0.0	...	?	?	0	

7 rows × 36 columns

7.2. Cleaning the data: Cleaning is one of the necessary parts of the data analysis. As we have seen in the above 3 images, there is some dirt present in the datasets in the form of “?” and it is required to clean them.

```
pnd_data_frame=pnd_data_frame.replace("?", nmp.NaN)
```

Figure[4]

The easiest way to delete all the unwanted cells is to replace them with null values in all those cells that have symbols and it would be easy to handle all of the null cells in a single line of code.

```
print("Total number of null values in the datasets are {}". format(pnd_data_frame.isnull().sum().sum()))
```

```
Total number of null values in the datasets are 3622
```

Figure[5]

Now it can be clearly seen from the code mentioned in the figure[5] that there are a total 3622 number of null values in the whole dataset. Function isnull() is used to find null values in all columns as boolean values(True or False) but if sum() function is used with isnull() then it would come up with a total number of null values column wise.

```
pnd_data_frame.isnull().sum()
```

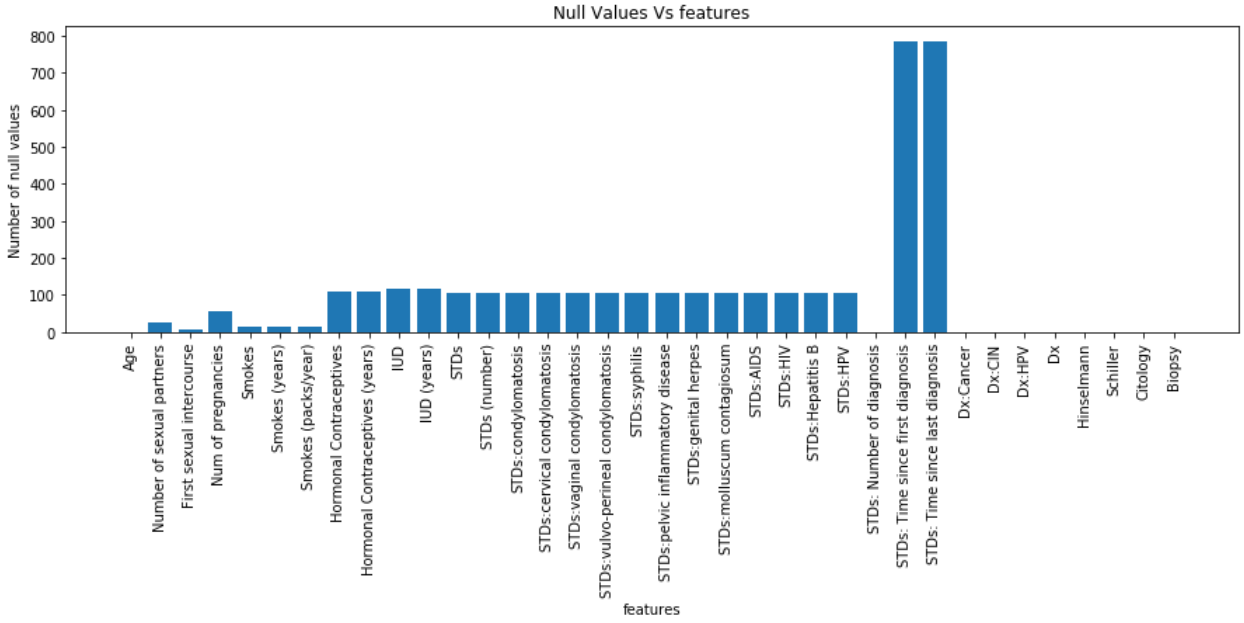
```
Age                                   0
Number of sexual partners            26
First sexual intercourse              7
Num of pregnancies                   56
Smokes                               13
Smokes (years)                       13
Smokes (packs/year)                  13
Hormonal Contraceptives             108
Hormonal Contraceptives (years)     108
IUD                                 117
IUD (years)                         117
STDs                                105
STDs (number)                       105
STDs:condylomatosis                 105
STDs:cervical condylomatosis        105
STDs:vaginal condylomatosis         105
STDs:vulvo-perineal condylomatosis  105
STDs:syphilis                       105
STDs:pelvic inflammatory disease    105
STDs:genital herpes                 105
STDs:molluscum contagiosum          105
STDs:AIDS                           105
STDs:HIV                            105
STDs:Hepatitis B                    105
STDs:HPV                            105
STDs: Number of diagnosis             0
STDs: Time since first diagnosis    787
STDs: Time since last diagnosis     787
Dx:Cancer                             0
Dx:CIN                                0
Dx:HPV                                0
Dx                                    0
Hinselmann                            0
Schiller                              0
Citology                              0
Biopsy                                0
dtype: int64
```

Figure[6]

In Figure[6] and visually can be seen Figure[8], there are two specific columns that have 787 numbers of null values each out of 858 rows. It would be perfectly fine if these two columns are removed from the dataset.

```
mplt.figure(figsize=(15, 5))
mplt.xticks(rotation=90)
mplt.bar(pnd_data_frame.columns, pnd_data_frame.isna().sum());
mplt.xlabel("features")
mplt.ylabel("Number of null values")
mplt.title("Null Values Vs features");
```

Figure[7]



Figure[8]

Now, it is a very easy task to delete two columns named “STDs: Time since first diagnosis” and “STDs: Time since last diagnosis” by using a predefined drop() function.

As it is seen in Figure[3] that dataframe have objects as well as integer data types. Need to convert object data types into numeric data types as machine learning models do not work with object data types. Code is given in the following figure[9].

```
pnd_data_frame=pnd_data_frame.apply(pnd.to_numeric)
```

Figure[9]

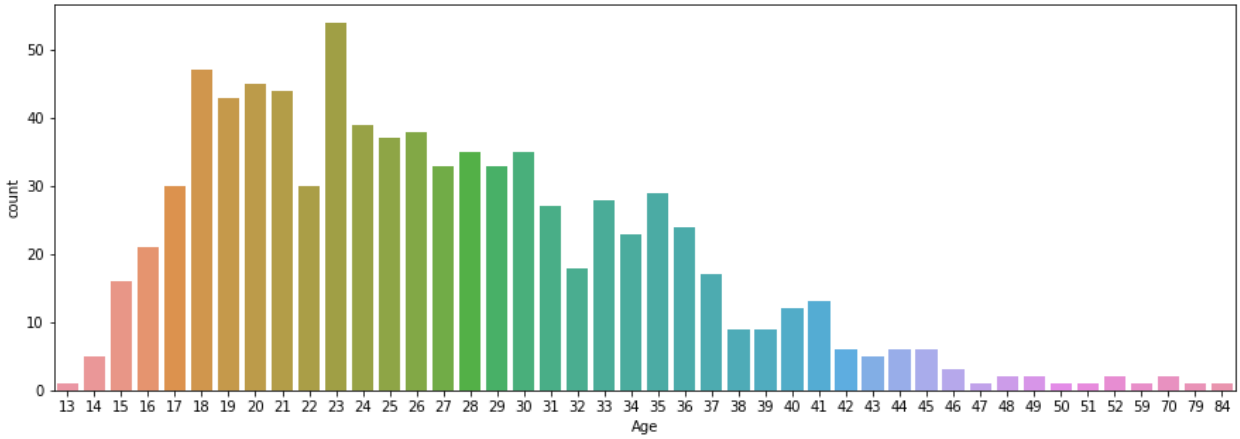
It is now the time to replace null values with some numeric values. Here the mean value has been calculated from each and every column and it has been replaced with null values with mean values in each corresponding column.

There were also some duplicated rows and those duplicate rows were deleted by using drop_duplicates() function.

8. Data Visualization

It is a graphical representation of data or information. Lots of graphs and charts are used in this report to represent data more meaningfully.

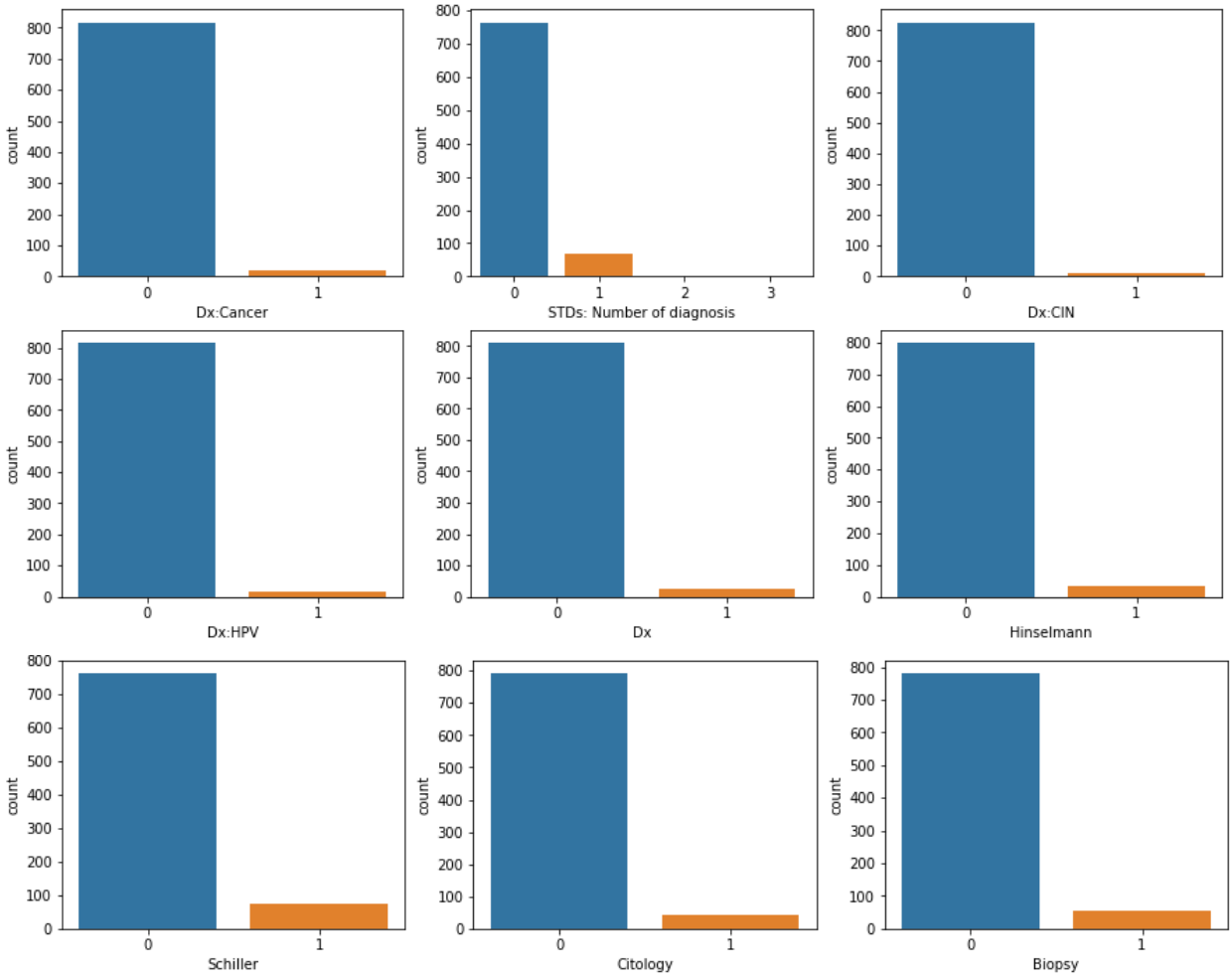
8.1. Count plot graph: This is the method to count each and every item in the particular column graphically by creating a bar graph. Countplot() is a builtin function in seaborn library and it is used to plot bar charts based on a particular column and counts each and every specific item. Cervical cancer dataset has only 10 columns that have integer data types and those columns can be seen through the countplot() function in the following figures.



Figure[10]

Looking at the “age vs count” graph shown in the Figure[10], numbers are distributed over the x-axis and y-axis; however, the x-axis represents the “age” of the people and the y-axis represents count. X-axis contains ages numbered from 13 to 84 while numbers from 0 to 60 on y axis. let ‘s see, for example, there are 30 people infected with cervical cancer whose age is 17. Around 58 people out of 585 whose age is 23 have positive cancer while Very small number of old age group people(40 to 84) have positive cervical cancer. Similarly it is observed for other aged group people.

For Dx: Cancer column, there are only 0 and 1 that denote negative Dx-Cancer and positive Dx-cancer respectively.



Figure[11]

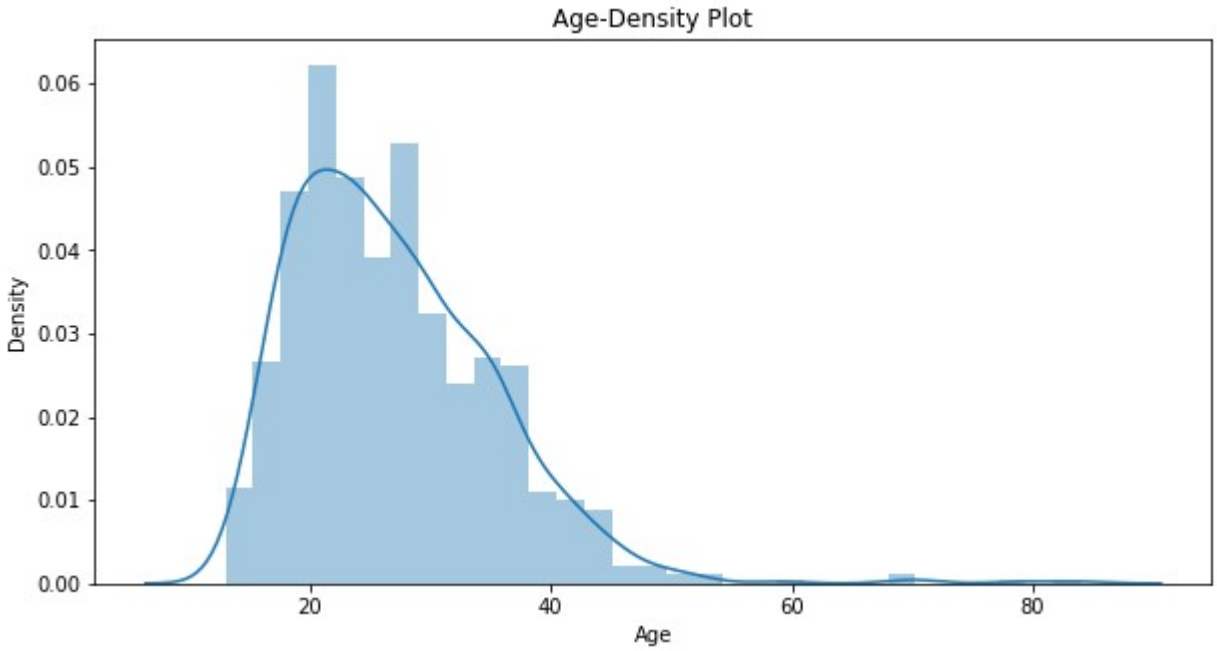
8.2. Density Plot graph: Density plot graph is a technique to find a distribution of data in continuous intervals. It distributes numeric variables. Density plots actually allow us to see a clear picture of the data distribution over the smooth curve graph.

Seaborn has a graphical function that plots density graph beautifully and the function is distplot(). distplot() basically a combination of histplot() (a matplotlib builtin function that use to create a histogram) and kdeplot(Kernel Density Plot, A seaborn function which shown distribution of data by drawing curve over the data).

- Age Vs Density: Density over the column names age is shown in the following graph and it can be observed by having a look over the graph of how the age is distributed. There is only one big curve that shows the peak point of the graph. In the following plot, the x-axis represents Age and y-axis represents density.

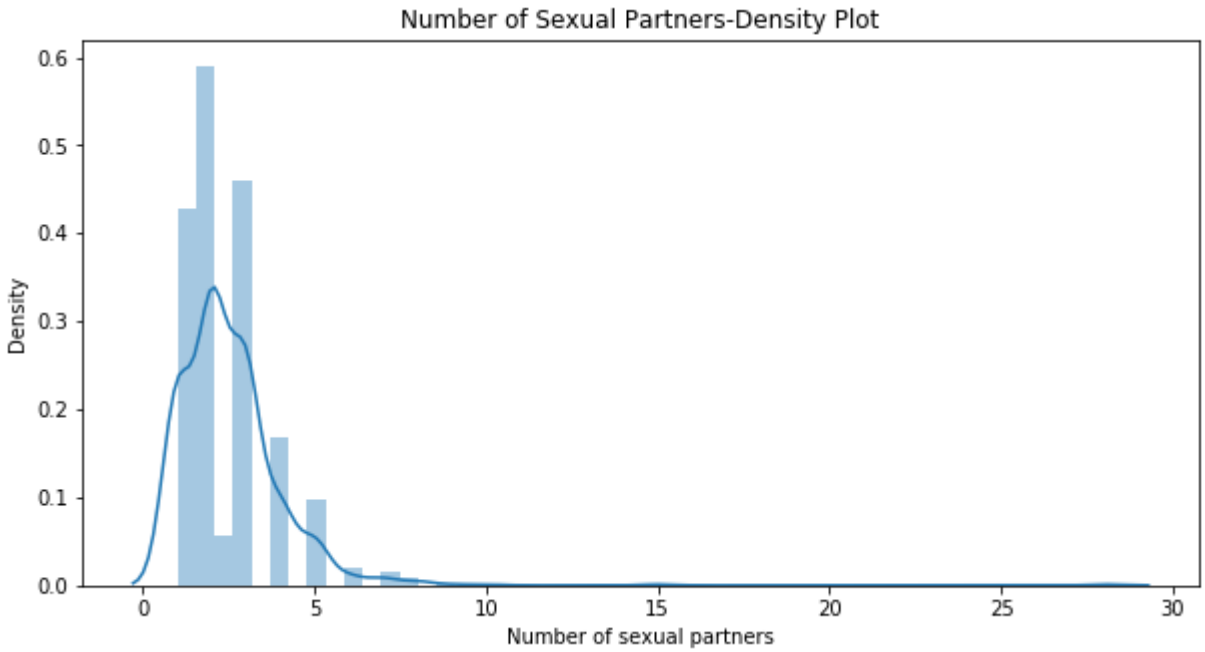
  By observing the figure[12], ages that lie between 18 and 43 are so densile while the old age group, tenagers and childs have seen very less positive tests

towards cervical cancer. It means adults (18 to 43) age groups have more positive cervical cancer than other age groups patients.



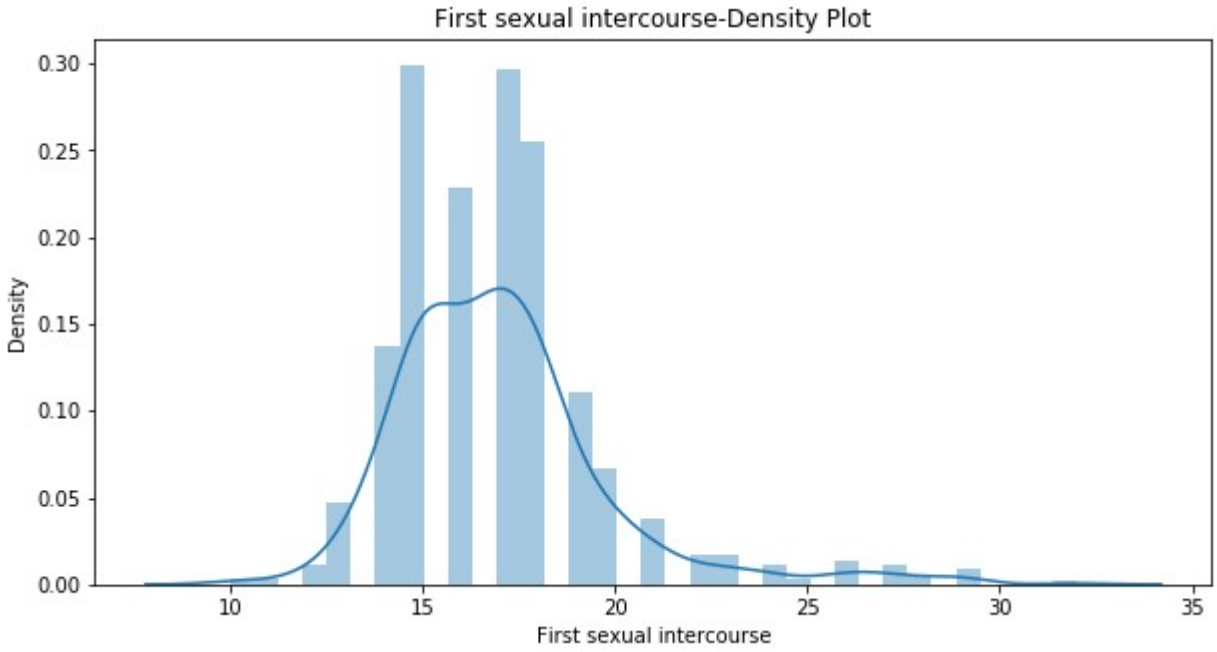
Figure[12]

- Number of Sexual partners Vs Density: Density plot on number of sexual partners can be seen in the below graph and the graph shows that those patients have more than 2 (range from 2 to 5) number of sexual partners are more fall in the category of cancer.



Figure[13]

- First Sexual intercourse Vs Density: For those patients who had their first sexual intercourse from age range 13 to 18 got more infected with this cancer because in the below graph density are more in that age range than other.



Figure[14]

- Number of Pregnancies Vs Density: Those female patients got pregnant between 2 to 5 times, more infected with cervical cancer.

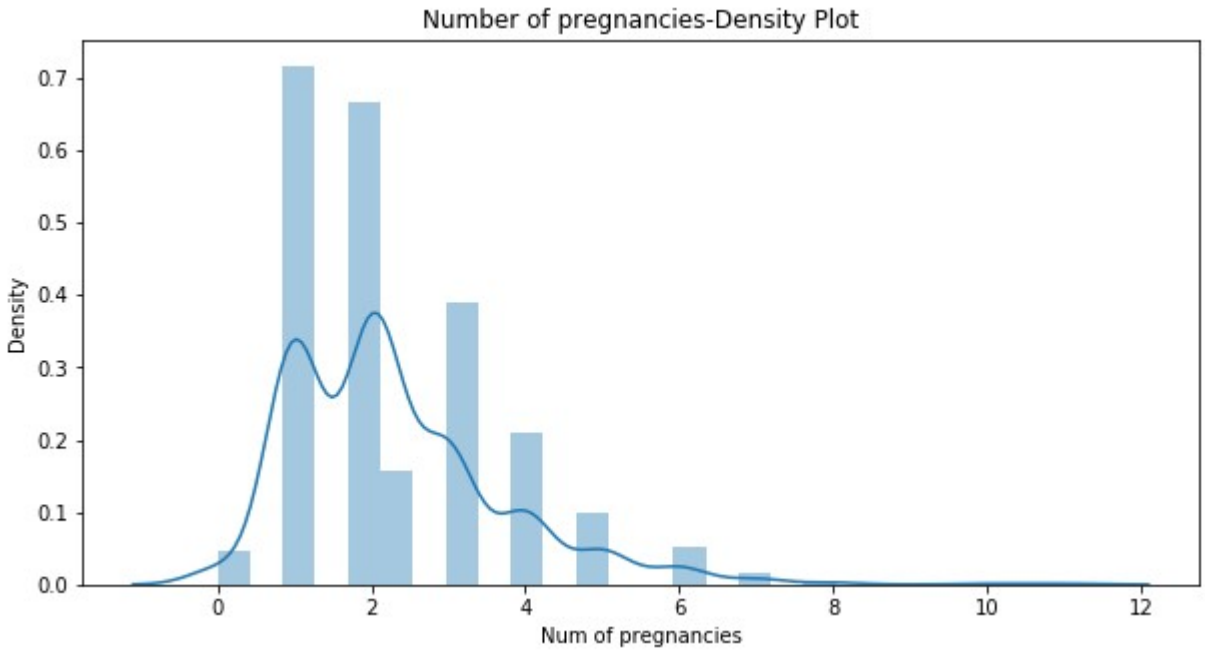
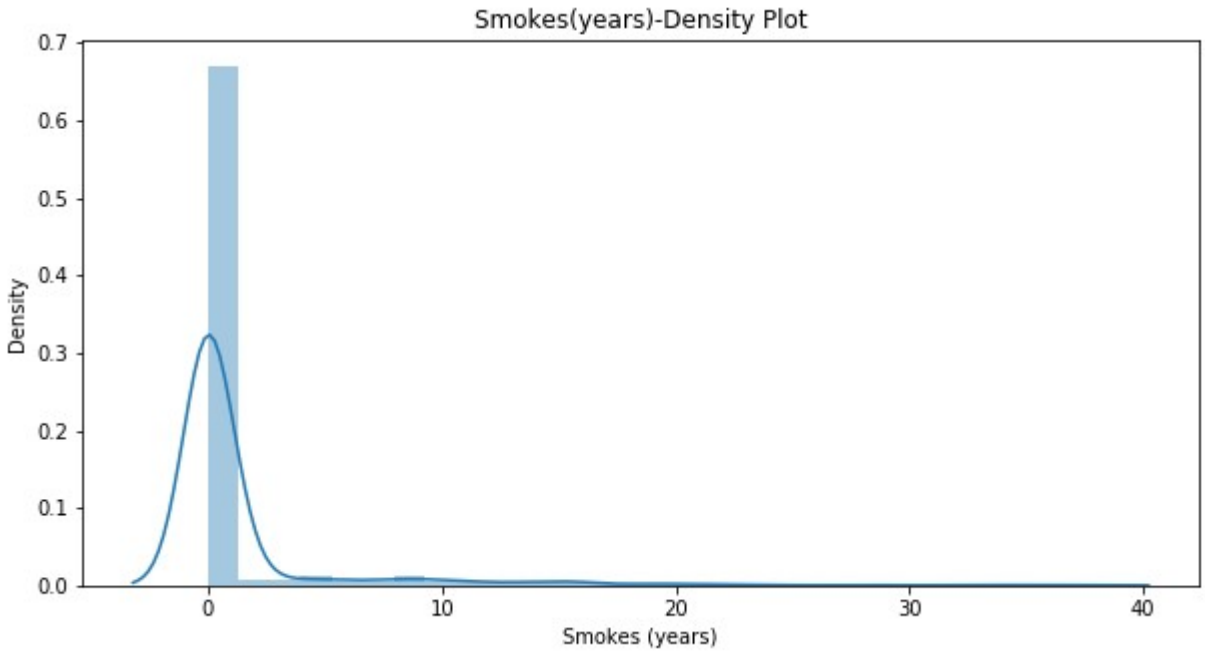


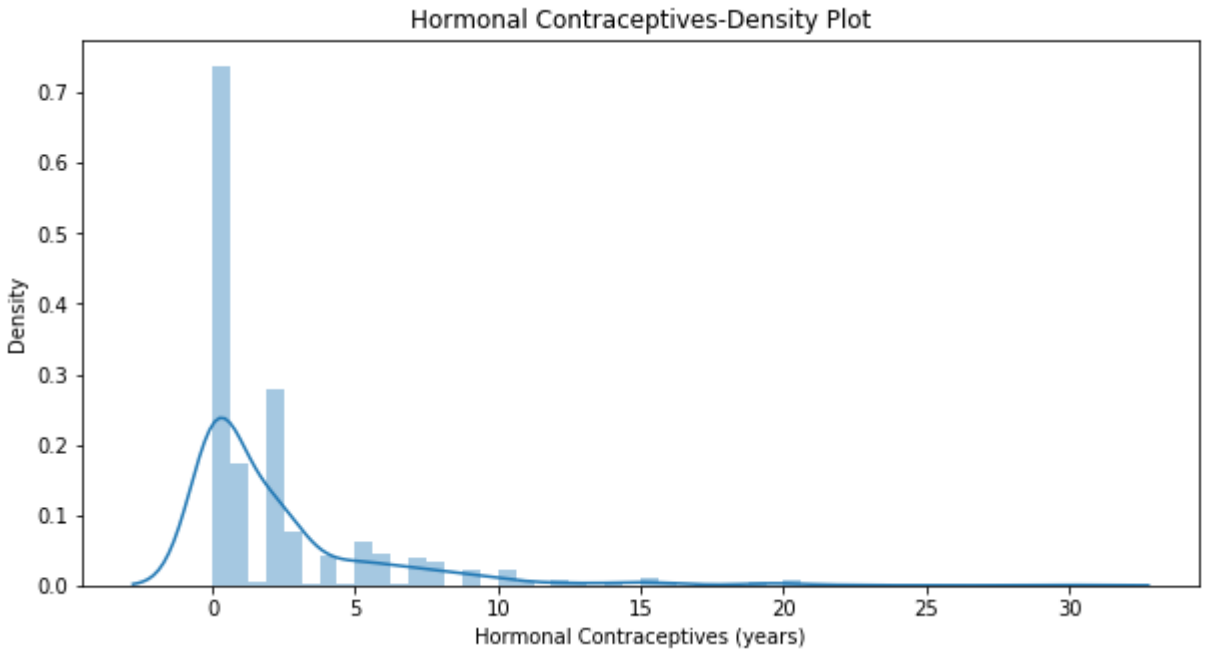
Figure 15

- Smoking(year) Vs Density: The density graph shows that those patients taking smokes from 0 to 2 years get cervical cancer but it is a very rare chance for the smokers to get cervical cancer. Cervical cancer does not strongly depend on smoking.



Figure[16]

- Hormonal Contraceptives(year) Vs Density: Hormonal contraceptives generally occurred in females. These contraceptives protect women from getting pregnant because these kinds of rings are placed in between vegina and uterus and do not allow females to get their egg fertilization. By the visualization from the following figure, Majorly females who use hormonal contraceptives less than 2 years are more prone to cancer.



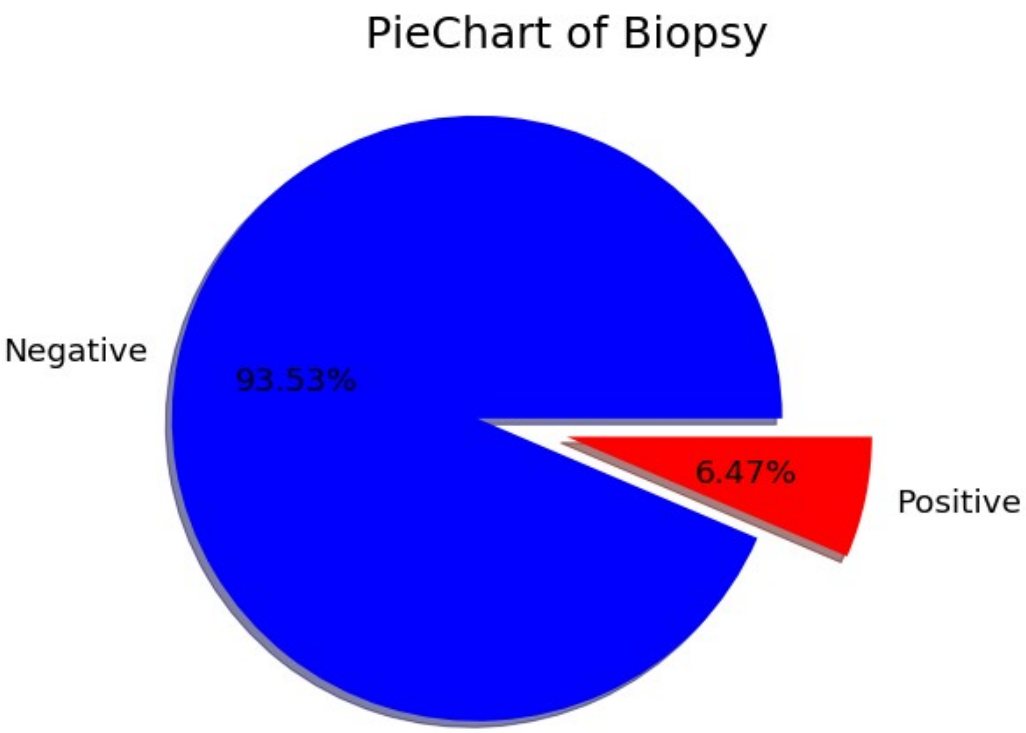
Figure[17]

- Biopsy: It is the method of testing or examining by taking samples of body tissues or cells in order to test the result whether an individual has been infected with disease or not, cancer in this case. The datasets have a column name biopsy and have number 0 and 1. 1 denotes that an individual has gotten a positive test towards cervical cancer while zero shows a negative test.

  Code has been prepared for creating a beautiful pie chart using numerical biopsy data. An important python library is used known as matplotlib for visualising numerical data in the form of pie charts. First line of the code shows the size of the pie chart where figsize is 8x8. Tags are given for categorizing labels 0 as negative and 1 as positive. To look beautiful, pie charts need to be sliced into two parts by using predefined keywords. Color has been given to both sliced parts in order to differentiate easily and then the pie chart is drawn by using the pie() function in matplotlib.

```
mplt.figure(figsize=(8,8))
tag=["Negative", "Positive"]
slice1=[0.3,0]
pie_color=["blue", "red"]
mplt.pie(total, labels=tag, explode=slice1,autopct="%1.2f%%",
         shadow=True, textprops={"fontsize": 18.5},
         colors=pie_color);
mplt.title("PieChart of Biopsy", fontsize=25);
```

Figure[18] Code for biopsy pie chart



Figure[18]

```
total=pnd_data_frame["Biopsy"].value_counts()
print(total)
```

```
0    781
1     54
Name: Biopsy, dtype: int64
```

Above pie chart is prepared on the basis of given biopsy data. There are a total of 835 numerical data(0 and 1) in which 781 cells of the data show 0 and the rest of them are 1. By using the data, it can be seen visually through a given pie chart that the red colored sliced part says 6.47% of the patients tested positive and the rest of the pie chart (blue colored) shows a

negative test (93.53% are negative). It means only 6.47 percent of people got cancer positive, however 93.53 percent did not get infected with cervical cancer.

9. Feature Selection

It is sometimes also called variable selection. Feature selection is a technique to minimise the number of independent variables or features. So basically what it does is it shows the best features with their score among all the features. The one who scores high would be preferred best and others accordingly. This usually applied on a huge number of features where it became hard to check each and every feature and it will also be a high cost and time taking process. Furthermore, it increases the performance of the model. It makes it easy to select features for any machine learning or deep learning model.

9.1 Benefits of using feature selection:

1. The first and most important benefit is that it reduces overfitting rather than having noisy data or redundant data that can create hard for models to make accurate decisions.
2. It increases the accuracy of the model because if the data is less noisy then the model will automatically work accurately.
3. There would be taking very less time to train the data for the best features rather than training all the features.

Below code is used to select the most important features. SelectKBest is a scikit-learn class which selects the best number of features among independent variables according to k value with best chi-squared score. K value is nothing but the highest score value. If k=15 it means SelectKBest will choose 15 numbers of best features. As the target data(0 or 1) is categorical, it needs to use the Chi-Squared test(chi2) because Chi-Squared only works on categorical types of data. Here the target is “v” and “U” is independent features. To apply this class on the independent variables, it has to use a fit function in order to select the best features. The final results of the best 15 features with best chi squared score or low p-value can be seen in the below figure.

```
top_features=fs.SelectKBest(score_func=fs.chi2, k=15)
v=pnd_data_frame["Biopsy"]
U=pnd_data_frame.drop(["Biopsy"], axis=1)
mod1=top_features.fit(U,v)
```

	Best_Features	Chi2-score
1	Schiller	404.814045
2	Hinselmann	244.179030
3	Citology	77.733404
4	Smokes (years)	42.399880
5	Hormonal Contraceptives (years)	29.440956
6	Dx:HPV	21.479063
7	Dx:Cancer	21.479063
8	Dx	20.444469
9	STDs:genital herpes	13.022859
10	STDs (number)	12.345046
11	STDs:HIV	11.267231
12	Dx:CIN	10.739531
13	STDs: Number of diagnosis	8.336329
14	STDs	7.618027
15	Age	6.571329

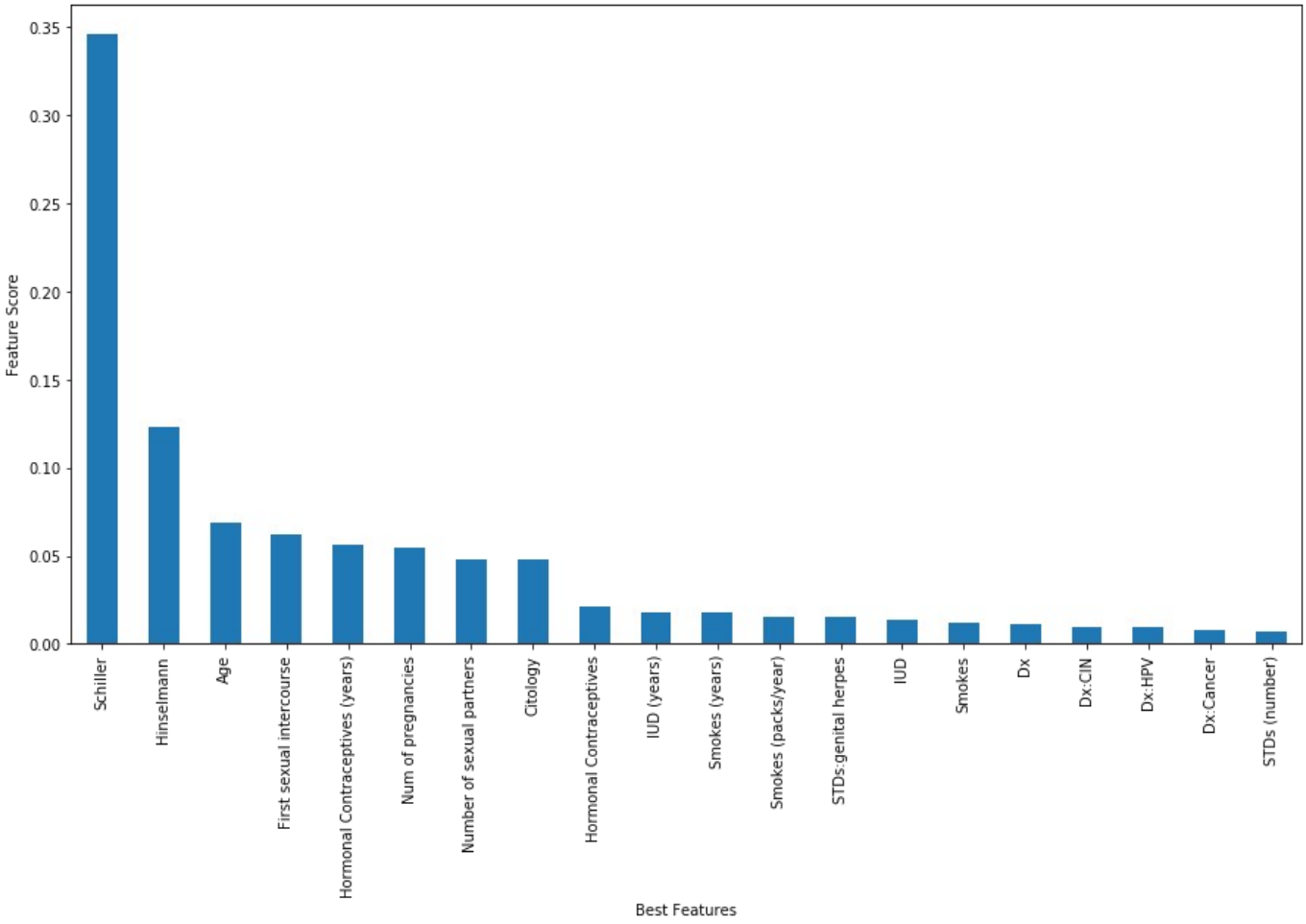
10. Feature Importance

It is used to give the importance to the features according to feature score. So basically feature importance is used to calculate score and gives importance to the features for model accuracy because features with large scores put a large impact on mathematical models and that would help in getting good accuracy of the model.

It helps better in understanding feature relationships with target variables. Those features that have high score need to be kept and less score features have to be deleted because less scored features does not impact or does not help in boosting model accuracy.

```
p_mod = ExtraTreesClassifier()
p_mod.fit(U,v)
p_mod.feature_importances_
mplt.figure(figsize=(18,8))
imp_feature = pnd.Series(p_mod.feature_importances_, index=U.columns)
mplt.xticks(rotation=50)
imp_feature.nlargest(20).plot.bar();
mplt.xlabel("Best Features")
mplt.ylabel("Feature Score");
```

In the above code mentioned in figure, ExtraTreeClassifier() is used to minimise the overfitting and over learning. It ensures that models do not overfit the data. The result is in the figure below as a form of graph.



Figure[19]

In this above figure, Schiller is given highly importance with a score 34.8. It means Schiller is a very important feature among all independent variables. Other features such as Hinselmann, Age, first sexual intercourse, hormonal Contraceptives(year), Num of pregnancies, number of sexual partners, Citology with scores 12.8, 7.3, 7.1, 7.01, 7.01, 6.82, 6.82 repectively. These features will impact the model with accuracy. If any of them is removed then there will be some accuracy loss for the model.

11. Splitting the data

This technique is one of the most important steps for any machine learning model. In this method, the datasets split into two parts generally for training and testing purposes for the model. The reason behind splitting the data into two parts is because the model needs to train with the training dataset and then test the model using test data or unseen data in order to get the accuracy of the model. Unseen data means the model is not aware of the data; it is totally new for the model.

There is a library in python that takes data and splits it automatically into training dataset and testing dataset however it is possible to split manually by coding simple python programming.

```
U_train, U_test, v_train, v_test = train_test_split(U, v, test_size = 0.2, random_state=37)
print(U_test.shape)
print(U_train.shape)
```

```
(167, 33)
(668, 33)
```

By the above code in the figure, it can be observed that train_tes_split() is a function that is used to split the data according to given parameters. The parameters are feature dataset(U), v(target), test_size/train_size and random_state. Test_size is equal to 0.2 means the data will split into two however, the test dataset will be 20% and the train dataset will be 80%. It is recommended to take at least more than 70% data for training purposes because more data for training will make the model more accurate. Once a model is trained then it can be tested using testing datasets.

Random state as random_state is used as a parameter in train_test_split() because if it is not set to any number then train and test data will get different data every time execute the code and it will be very hard to debug. Random_state can be set to any positive number, in the above code it is set to 37.

12. Modelling Creation

Model creation is the process of data analysis where data is in the form of a train dataset to fit or train the model and then later the same model is tested using unseen datasets or test dataset.

In this report, 2 models have been made in order to evaluate and compare the results between two different models. First model that is used for the train dataset is SVM also known as Support Vector machine and the second model is Knn or K-nearest neighbors.

12.1 Support vector machine: It is one of the most popular machine learning algorithms that works on classification and regression problems. It works as supervised learning where the model has to teach by providing a labelled training dataset and once the model is learned every pattern in the dataset then the model can be tested on the test dataset and expect to get the accurate results.

The main reason to select the svm model is that it is good on classification problems as the given cervical cancer dataset(target) is categorical(0 or 1). It is well organised in terms of memory i.e memory efficient.

```
model_svm=LinearSVC(max_iter=1000)
model_svm=CalibratedClassifierCV(model_svm)
clf=model_svm.fit(U_train,v_train)
```

LinearSVC is the class in the scikit learn and it takes one of the parameters max_iter that is used to set the number of iterations for the classifier to run. If max_iter=1000 then the classifier will run a maximum 1000 iteration but if there is no max_iter by default it is set to -1 it means it allows the classifier to have no limit on iteration and this can cause it to run longer than expected. model _svm is passed as a parameter to the CalibratedClassifierCV() class in order to train a calibrated classifier. And at the end models have to be trained by using fit() function as mentioned in the above code figure. U_trained and v_train are the training features however, v_train is the training target data.

```
y_predict=clf.predict(U_test)
print([y_predict])
```

```
[array([0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0])]
```

In the above code clf represents svm classifier or svm model. Now the svm model is ready to work on unseen datasets. The predict() function is used to take unseen features(U_test) to predict target(y_predict). And the results can be seen in the form of an array. Let’s see the accuracy of the svm model on test datasets.

```
print("SVM model Accuracy: {}%".format(round(accuracy_score(y_test, y_predict)*100, 2)))
```

```
SVM model Accuracy: 95.21%
```

The term accuracy_score does calculate the accuracy of the model by using test features with predicted targets). SVM model accuracy in the case of cervical cancer data is 95.21% and it is quite good accuracy.

12.1.1 Model Evaluation for SVM

Model evaluation means to examine and analyse the performance of a model.

```
Comparision_df=Comparision_df.rename(columns={"Biopsy": "Actual Values"})
Comparison_Column=nmp.where(Comparision_df["Actual Values"]==Comparision_df["Predicted Values"], True, False)
Comparision_df["Matching"]= Comparison_Column
Comparision_df.sample(10)
```

The above code is used to compare between predicted values that were obtained from the model and Actual Values(Biopsy) or the test dataset(20% unseen data) and the results can be seen in the following comparison table.

	Actual Values	Predicted Values	Matching
149	0	0	True
129	0	0	True
64	0	1	False
26	1	1	True
88	0	0	True
52	0	0	True
134	0	0	True
120	0	0	True
90	0	0	True
16	0	0	True

Comparison table between actual and predicted
For SVM model

```
Comparision_df["Matching"].value_counts()
```

```
True     159
False      8
Name: Matching, dtype: int64
```

It is clearly observed that only 8 values from both the columns(actual and predicted) are unmatching; the rest of them are matching true. As it has already been seen that accuracy of the model is 95.21% while 4.79% are inaccurate. 8 Unmatching values come in 4.79%.

Confusion Matrix for SVM: It is the matrix table that shows the performance measurement of the model.

```
confusion_matrix(v_test, y_predict)
```

```
array([[157,   6],
       [  2,   2]])
```

Confusion matrix table

In case of the SVM model, confusion matrix is shown in the above image where

True Positive = 157

False Positive = 2

True Negative = 2

False Negative= 6

True positive: when prediction actually becomes true like for example if one predicted that “Mia” went to Birmingham and she actually went then it would be true positive.

False Positive: When positive prediction becomes false.

True negative: When negative prediction becomes true.

False Negative: When prediction is negative and it is actually negative then it will come under the category of False negative.

Explanation:

By the above performance given in the confusion matrix, it is observed that 157 numbers of predictions done by svm model are actually correct, 2 number of correct predictions got actually wrong furthermore, 2 number of wrong predictions became true however, 6 number of wrong predictions was actually wrong.

12.2 K-Nearest Neighbor(KNN):

This algorithm is a very simple machine learning algorithm that also works for classification as well as regression problems. It is also supervised learning(works on labelled datasets), the same as the SVM model.

KNN model learns training dataset at the time of prediction and this makes KNN more faster than other machine learning models like SVM.

```
model_knn=KNeighborsClassifier()
knnn=model_knn.fit(U_train,v_train)
y_predict2=model_knn.predict(U_test)
```

KNeighborsClassifier() is the class that makes a KNN model to classify the best way to be able to get more accuracy. Then trained the model on a training dataset using fit() function. And at the end model_knn.predict() is used to predict the result.

```
print("KNN model Accuracy: {}%".format(round(accuracy_score(v_test, y_predict2)*100, 2)))
```

```
KNN model Accuracy: 96.41%
```

The accuracy obtained from the KNN model is 96.41%.

12.2.1 Model Evaluation for KNN

```
Comparision_df2=Comparision_df.rename(columns={"Biopsy": "Actual Values"})
Comparison_Column2=nmp.where(Comparision_df2["Actual Values"]==Comparision_df2["Predicted Values"], True, False)
Comparision_df2["Matching"]= Comparison_Column2
Comparision_df2.sample(10)
```

The above code is similar to svm but this is a comparison for KNN model.

	Actual Values	Predicted Values	Matching
159	1	1	True
147	0	0	True
47	0	0	True
74	0	0	True
137	0	0	True
79	0	0	True
154	0	0	True
125	0	0	True
43	0	1	False
18	0	0	True

Comparison table between actual and predicted
For KNN model

```
Comparision_df2["Matching"].value_counts()
```

```
True     161
False      6
Name: Matching, dtype: int64
```

By looking at the above image, it can be seen that there are 161 true values which means 161 predicted values are exactly matching with the actual target(v_test). Only 6 values are predicted wrong.

- **Confusion matrix for KNN**

  This part will show the overall performance of the model by calculating true positive, true negative, false positive and false negative.

```
confusion_matrix(v_test, y_predict2)
```

```
array([[163,   0],
       [  4,   0]])
```

Confusion matrix

Output of the above code represents an array of matrices where there are 4 numbers.

True Positive = 163

False Positive = 4

True Negative = 0

False Negative= 0

163 number of predictions became true or it matches exactly with the desired output, 4 number of positive predictions became false when compared to real output, and the rest of them is 0.

13. Comparison of SVM and KNN model

Both are statistical models that work on labelled data and both solve classification as well as regression type of problems. But let's see both models on the basis of performance calculated above.

Accuracy comparison of SVM and KNN:

	SVM	KNN
Accuracy	96.41%	97.6%

Both models have been compared to each other on the basis of accuracy and these are in such a way that the accuracy of the SVM model on cervical cancer datasets is 96.41% however 97.6% accuracy calculated for KNN model.

Classification Report of SVM and KNN:

	precision	recall	f1-score	support
0	0.99	0.97	0.98	163.00
1	0.38	0.75	0.50	4.00
accuracy	0.96	0.96	0.96	0.96
macro avg	0.68	0.86	0.74	167.00
weighted avg	0.98	0.96	0.97	167.00

(SVM classification report) Vs

	precision	recall	f1-score	support
0	0.98	1.00	0.99	163.00
1	0.00	0.00	0.00	4.00
accuracy	0.98	0.98	0.98	0.98
macro avg	0.49	0.50	0.49	167.00
weighted avg	0.95	0.98	0.96	167.00

(KNN classification report)

Classification report is the evaluation of the model completely where precision tells how much the model was predicted correctly. Recall is about how much percentage of positive cases the model was correct. F1-score represents percentage of positive correct prediction by model. 1.0 is considered the best f1-score however 0.0 is the worst f1-score. And support is the number of occurrences.

It became very clear by observing the classification report that overall accuracy of KNN model is more than SVM on the cancer dataset.

14. Critical reviews & Techniques used

It is analysed that 1 out of 10 female cancer patients are more likely to be infected with cervical cancer. The major reason behind this cancer is sexual intercourse without hormonal contraceptives with more than 1 partners involve during teenage. It becomes very interesting if cervical cancer can be predicted by looking at symptoms and this is what is done in this report. There are lots of advanced tools and techniques that have been used in order to predict accurately. Python and its libraries such as pandas, matplotlib, seaborn, scikit learn, etc have been used hugly for the analysis and predictions. Raw data is cleaned and made appropriate for the visualisation using bar graphs, pie charts, etc. Number of independent features has been reduced and removed unnecessary features by using feature selection techniques and gave them proper importance using feature importance techniques. Two models i.e Support vector machine and K-nearest neighbors are prepared, giving them proper training using training data in order to see the evaluation of the model using unseen data. Both the models are compared and it has been seen that KNN got a more accurate(97.6%) model than

SVM(96.41%) on cervical cancer data. The strength of this analysis is that a strong and good predictive model is prepared which has accuracy around 98% however the weakness is that KNN model accuracy depends on the quality of data and may be slow sometimes on big data. And both the models SVM as well as KNN will work only on labelled data.
By looking overall, It is recommended to use big data for SVM and KNN models because they work very smoothly and accurately.

15. References

[1] NHS (2021), *Cervical cancer* [Online], Available at: https://www.nhs.uk/conditions/cervical-cancer/ (Accessed: 18 November 2021).

[2] Galarnyk, M. (2018), *Understanding boxplots* [Online], Available at: https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd5 (Accessed: 18 November 2021).

[3] Brownlee, J. (2019), *How to Choose a Feature Selection Method For Machine Learning* [Online], Available at: https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/ (Accessed: 19 November 2021).

[4] Shaikh, R. (2018), *Feature Selection Techniques in Machine Learning with Python* [Online], Available at: https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e (Accessed: 20 November 2021).

[5] StackExchange (2019), *How does sklearn.SelectKBest uses chi2 test on continous data?* [Online], Available at: https://stats.stackexchange.com/questions/425368/how-does-sklearn-selectkbest-uses-chi2-test-on-continous-data (Accessed: 20 November 2021).

[6] SciKitLearn, (NA), *sklearn.feature_selection.SelectKBest* [Online], Available at: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html (Accessed: 21 November 2021).

[7] Shin, T. (2021), *Understanding Feature Importance and How to Implement it in Python* [Online], Available at: https://towardsdatascience.com/understanding-feature-importance-and-how-to-implement-it-in-python-ff0287b20285 (Accessed: 22 November 2021).

[8] Github, (2018), *SVC's max_iter default setting of -1 can cause very long running times* [Online], Available at: https://github.com/scikit-learn/scikit-learn/issues/11020 (Accessed: 22 November 2021).

[9] ScikitLearn, (NA), *sklearn.metrix.plot_confusion_matrix* [Online], Available at: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.plot_confusion_matrix.html (Accessed: 23 November 2021)

[10] Kohli, S. (2019), *Understanding a Classification Report For Your Machine Learning Model* [Online], Available at: https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397 (Accessed: 25 November 2021).