Implementation of Naive Bayes Classifiers

Naive Bayes Classifiers: Idea behind Naive Bayes Classifier is Bayes theorem.Bayesian reasoning is applied to decision making and inferential statistics that deals with probability inference. It uses the knowledge of prior events to predict future events.

The Bayes Theorem:

$$P(h/D) = \frac{P(D/h)\,P(h)}{P(D)}$$

Where,
P(h) : Prior probability of hypothesis h
P(D) : Prior probability of training data D
P(h/D) : Probability of h given D
P(D/h) : Probability of D given h

Applications of Naive Bayes Classifier:

1. Spam Classification
   -Given an email, predict whether it is spam or not
2. Medical Diagnosis
   -Given a list of symptoms, predict whether a patient has disease X or not
3. Weather
   -Based on temperature, humidity, etc... predict if it will rain tomorrow

Features of Naïve Bayes Classifier:

1. Robust to isolated noise points
2. Handle missing values by ignoring the instance during probability estimate calculations
3. Robust to irrelevant attributes
4. Independence assumption may not hold for some attributes
5. Really easy to implement and often works well

Example: Predicting the heart diseases from given features.

```
#import required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix
#loading Data
dataset=pd.read_csv('heart.csv')
dataset.head()
```

Output:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
#splitting data in Feature(x) and Target(y)
X=dataset.iloc[:,:-1]
y=dataset.iloc[:,-1]
#splitting data to train and test the classifier
X_train, X_test, y_train, y_test=
train_test_split(X,y,test_size=0.2,random_state=0)
#applying GaussianNB
classifier=GaussianNB()
classifier.fit(X_train,y_train)
y_predict=classifier.predict(X_test)
#Checking accuracy
accuracy_score(y_predict, y_test)*100
```

Output:

```
85.24590163934425
```

```
#attach predicted output with x-test
```

```
X_test["predicted output"]=y_predict
# Main features are selected for plotting purpose as follows
df_for_plot=X_test[["trestbps","chol","thalach","oldpeak","pred
icted output"]]
df_for_plot.head()
```

Output:

	trestbps	chol	thalach	oldpeak	predicted output
225	145	174	125	2.6	0
152	170	227	155	0.6	1
228	170	288	159	0.2	1
201	125	258	141	2.8	0
52	130	231	146	1.8	0

```
#plot the Predicted output as follows
sns.pairplot(df_for_plot, hue="predicted output")
```

Output: