EXPLORATORY PROJECT

WEATHER PREDICTION USING PYTHON (IDEAL PLAYING CONDITIONS)

Group members

NAME	ROLL NUMBER
ARITRA BANERJEE	21045175
ARYAN PATIDAR	21045026
HARSHRAJ JOSHI	21045055
HARSH YADAV	21045057
HARSHIT SHARMA	21045058

SKILLS INVOLVED

1.PYTHON PROGRAMMING
2.MACHINE LEARNING ALGORITHMS
3.STATISTICAL METHODS
4.BAYES THEOREM
5.GAUSSIAN NAIVE BAYES ALGORITHM
6.GAUSSIAN CURVE PLOTTING

TASKS PERFORMED BY EACH INDIVIDUAL

Harshraj Joshi and Harshit Sharma

Did the research work on how the project can be designed efficiently and decided the algorithm to be used in the project. Worked on the probabilistic aspects of the project and learned the Gaussian Naive Bayes algorithm. Designed a flow scheme for the model and prepared the psuedo code based on the algorithm.

Aritra Banerjee and Aryan Patidar

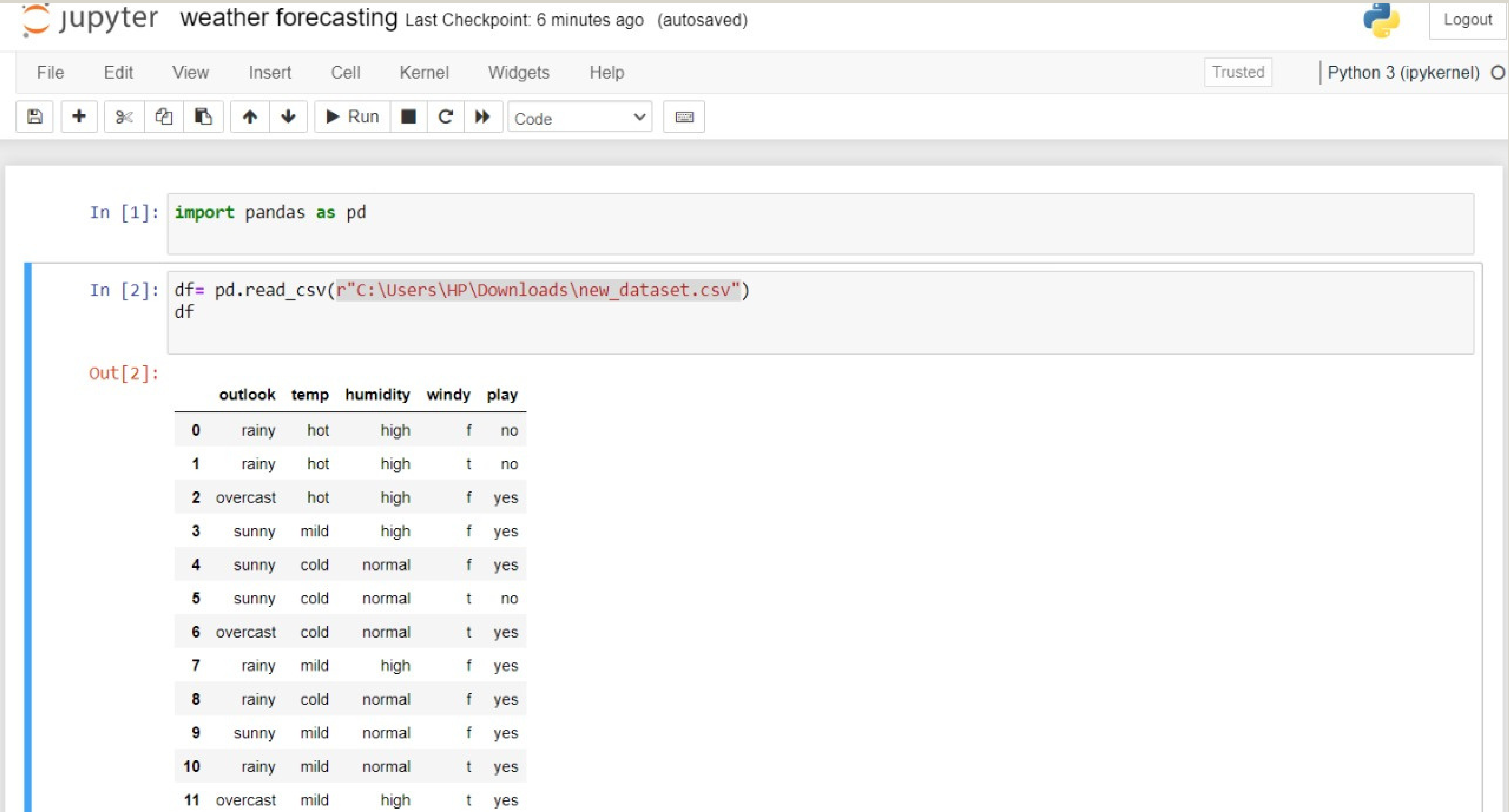
Aritra and Aryan learned Python programming language.

We learned python and then used jupyter notebook(ide) and used a library called pandas to read a csv file that we took from a wetaher site of a particular place and than a new data frame in which we converted all the string data base to numerical was done than we used gaussian naive bayes theoremn to predict the probability of the game using multiple provided factors on that day.

Harsh Yadav and Harshit Sharma

We did some research in probability deciding which algorithm would be best to predict whether a game could be played or not. We also did the necessary to understand Naive Gaussian method and perform calculations(Given at end) for various datasets at given points to establish the credibility of the method.

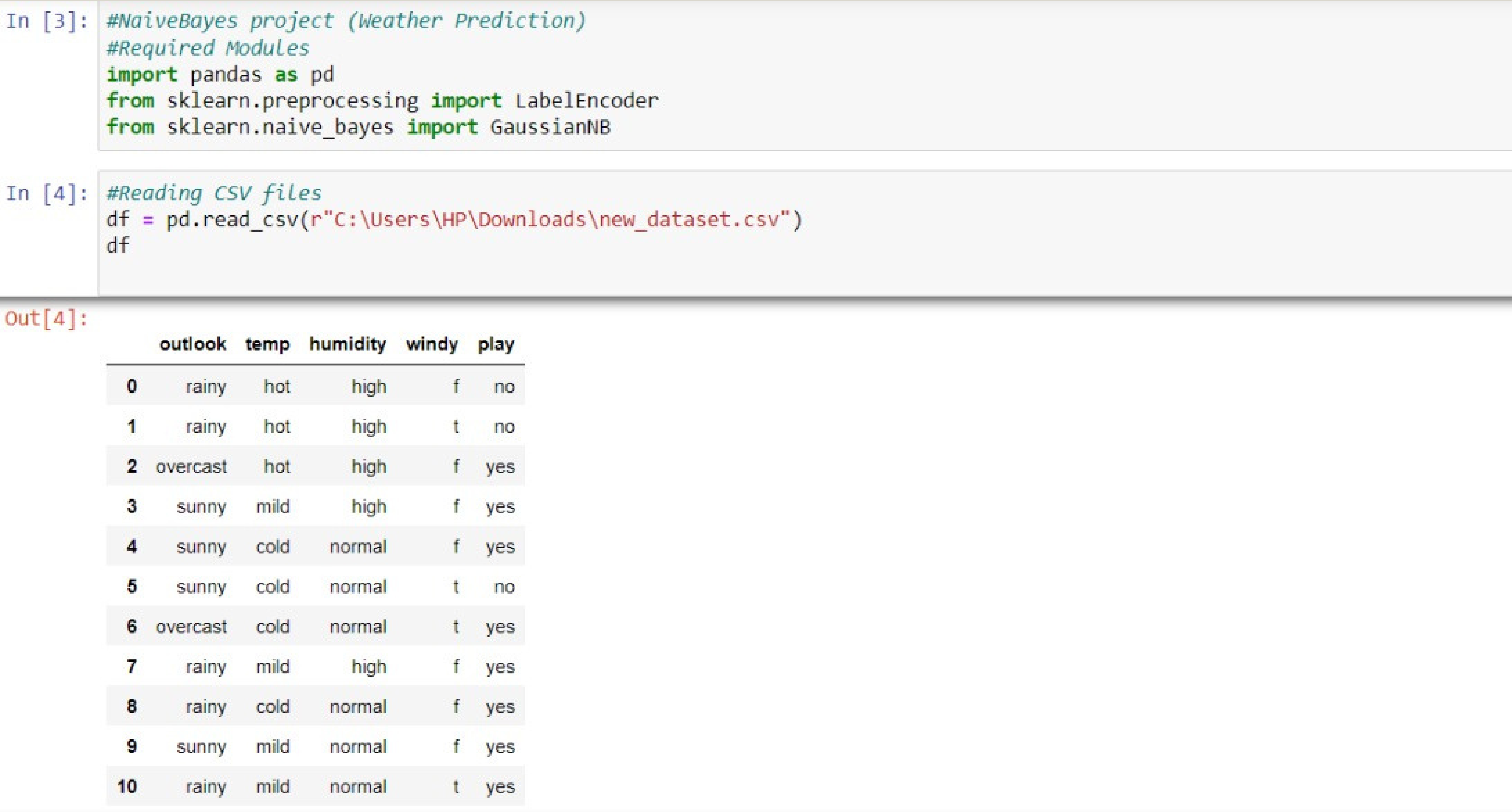
coding part



```
In [1]: import pandas as pd
```

```
In [2]: df= pd.read_csv(r"C:\Users\HP\Downloads\new_dataset.csv")
        df
```

```
Out[2]:
      outlook  temp  humidity  windy  play
   0    rainy   hot      high      f    no
   1    rainy   hot      high      t    no
   2 overcast   hot      high      f   yes
   3    sunny  mild      high      f   yes
   4    sunny  cold    normal      f   yes
   5    sunny  cold    normal      t    no
   6 overcast  cold    normal      t   yes
   7    rainy  mild      high      f   yes
   8    rainy  cold    normal      f   yes
   9    sunny  mild    normal      f   yes
  10    rainy  mild    normal      t   yes
  11 overcast  mild      high      t   yes
```



In [3]:

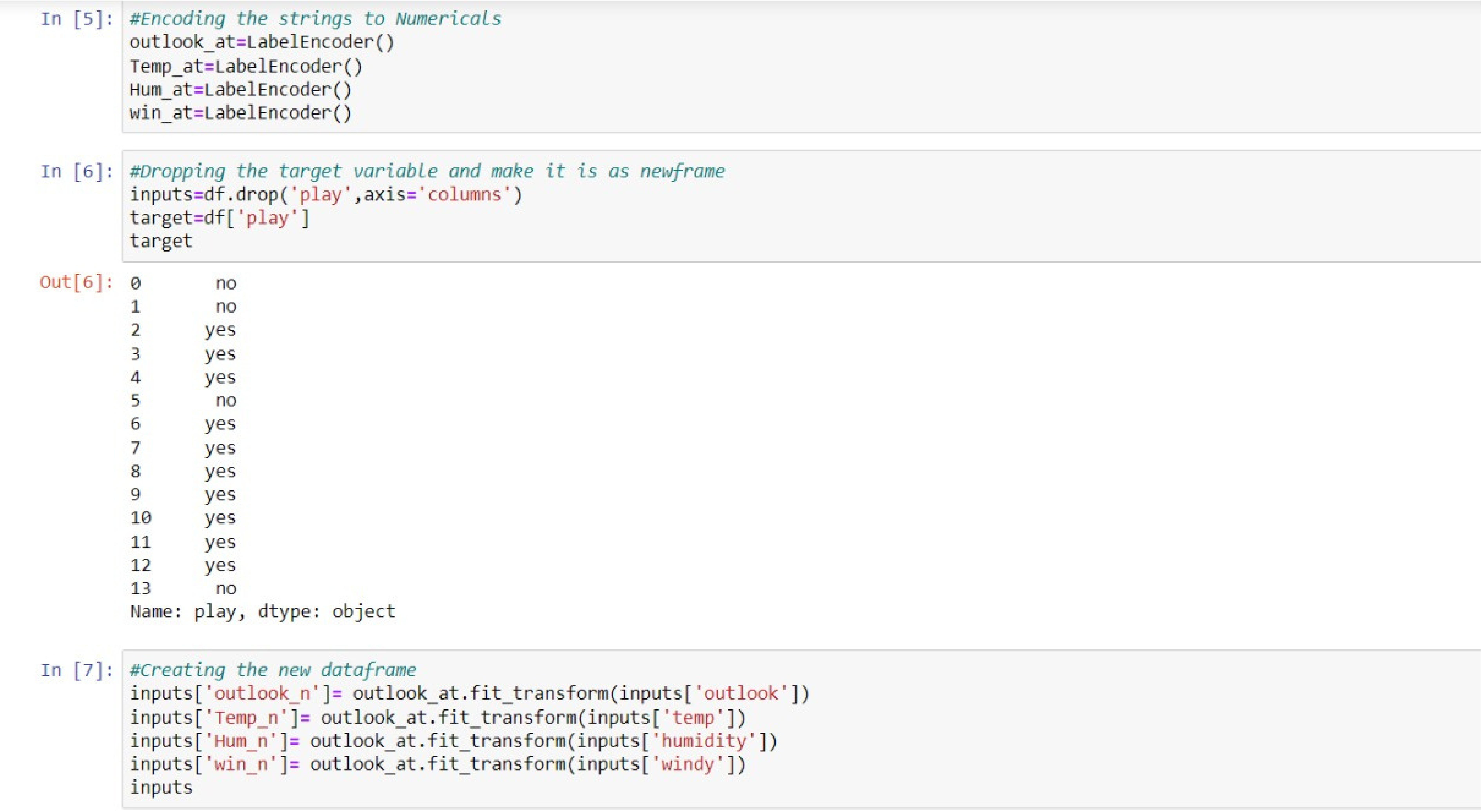
```python
#NaiveBayes project (Weather Prediction)
#Required Modules
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.naive_bayes import GaussianNB
```

In [4]:

```python
#Reading CSV files
df = pd.read_csv(r"C:\Users\HP\Downloads\new_dataset.csv")
df
```

Out[4]:

	outlook	temp	humidity	windy	play
0	rainy	hot	high	f	no
1	rainy	hot	high	t	no
2	overcast	hot	high	f	yes
3	sunny	mild	high	f	yes
4	sunny	cold	normal	f	yes
5	sunny	cold	normal	t	no
6	overcast	cold	normal	t	yes
7	rainy	mild	high	f	yes
8	rainy	cold	normal	f	yes
9	sunny	mild	normal	f	yes
10	rainy	mild	normal	t	yes



In [5]:

```
#Encoding the strings to Numericals
outlook_at=LabelEncoder()
Temp_at=LabelEncoder()
Hum_at=LabelEncoder()
win_at=LabelEncoder()
```

In [6]:

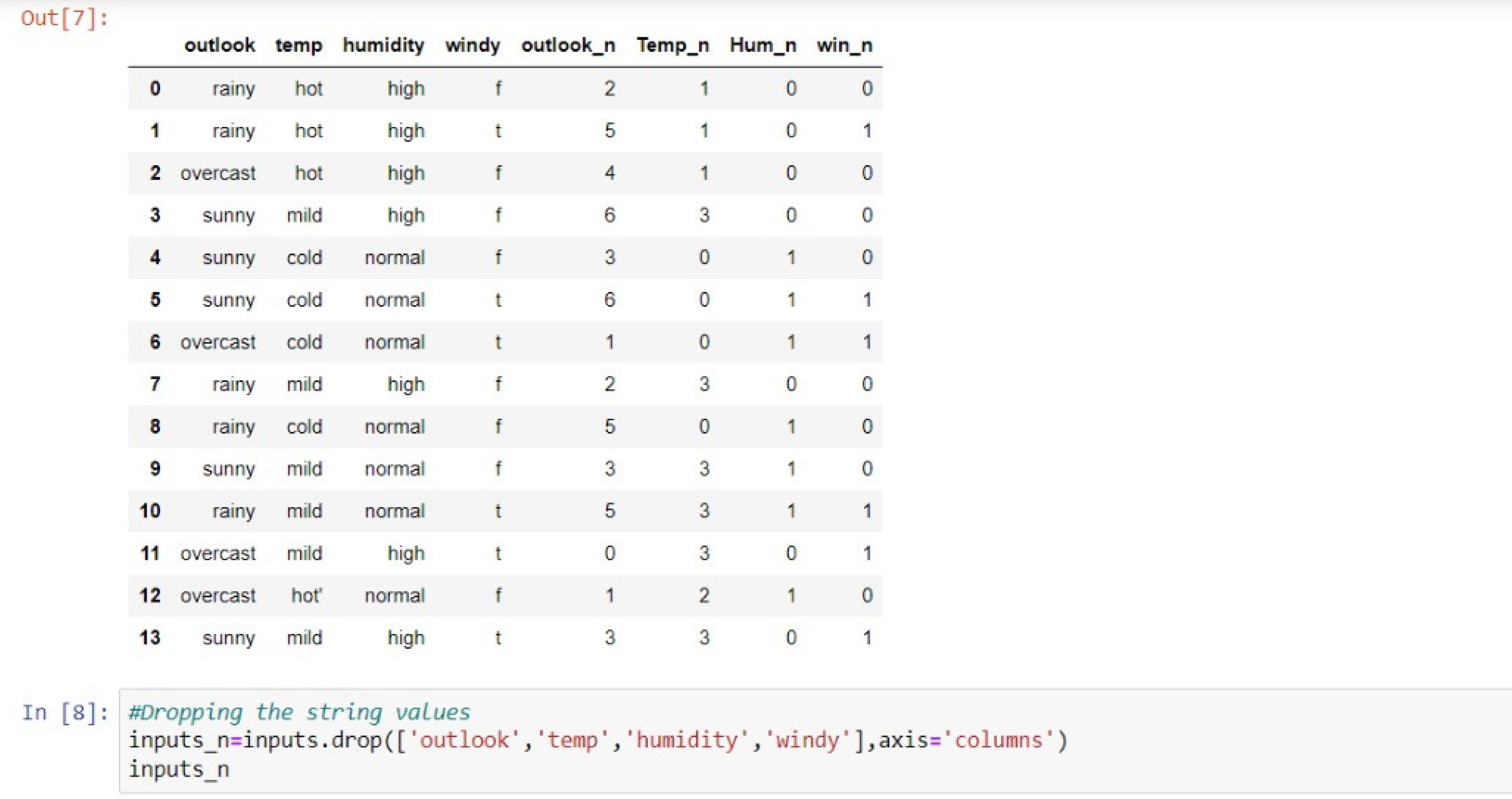
```
#Dropping the target variable and make it is as newframe
inputs=df.drop('play',axis='columns')
target=df['play']
target
```

Out[6]:

```
0      no
1      no
2     yes
3     yes
4     yes
5      no
6     yes
7     yes
8     yes
9     yes
10    yes
11    yes
12    yes
13     no
Name: play, dtype: object
```

In [7]:

```
#Creating the new dataframe
inputs['outlook_n']= outlook_at.fit_transform(inputs['outlook'])
inputs['Temp_n']= outlook_at.fit_transform(inputs['temp'])
inputs['Hum_n']= outlook_at.fit_transform(inputs['humidity'])
inputs['win_n']= outlook_at.fit_transform(inputs['windy'])
inputs
```

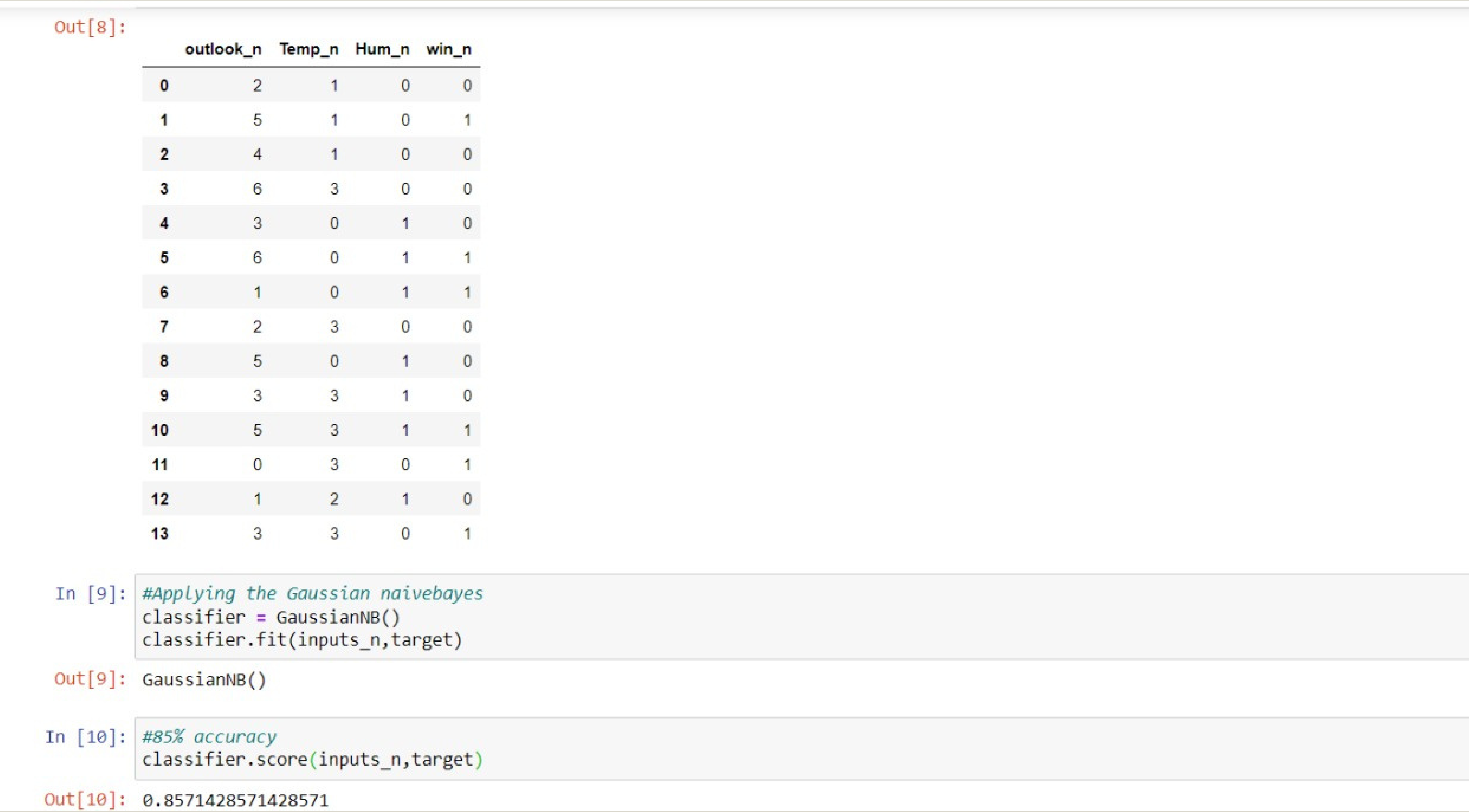


Out[7]:

	outlook	temp	humidity	windy	outlook_n	Temp_n	Hum_n	win_n
0	rainy	hot	high	f	2	1	0	0
1	rainy	hot	high	t	5	1	0	1
2	overcast	hot	high	f	4	1	0	0
3	sunny	mild	high	f	6	3	0	0
4	sunny	cold	normal	f	3	0	1	0
5	sunny	cold	normal	t	6	0	1	1
6	overcast	cold	normal	t	1	0	1	1
7	rainy	mild	high	f	2	3	0	0
8	rainy	cold	normal	f	5	0	1	0
9	sunny	mild	normal	f	3	3	1	0
10	rainy	mild	normal	t	5	3	1	1
11	overcast	mild	high	t	0	3	0	1
12	overcast	hot'	normal	f	1	2	1	0
13	sunny	mild	high	t	3	3	0	1

In [8]:

```
#Dropping the string values
inputs_n=inputs.drop(['outlook','temp','humidity','windy'],axis='columns')
inputs_n
```



Out[8]:

	outlook_n	Temp_n	Hum_n	win_n
0	2	1	0	0
1	5	1	0	1
2	4	1	0	0
3	6	3	0	0
4	3	0	1	0
5	6	0	1	1
6	1	0	1	1
7	2	3	0	0
8	5	0	1	0
9	3	3	1	0
10	5	3	1	1
11	0	3	0	1
12	1	2	1	0
13	3	3	0	1

In [9]:

```
#Applying the Gaussian naivebayes
classifier = GaussianNB()
classifier.fit(inputs_n,target)
```

Out[9]:

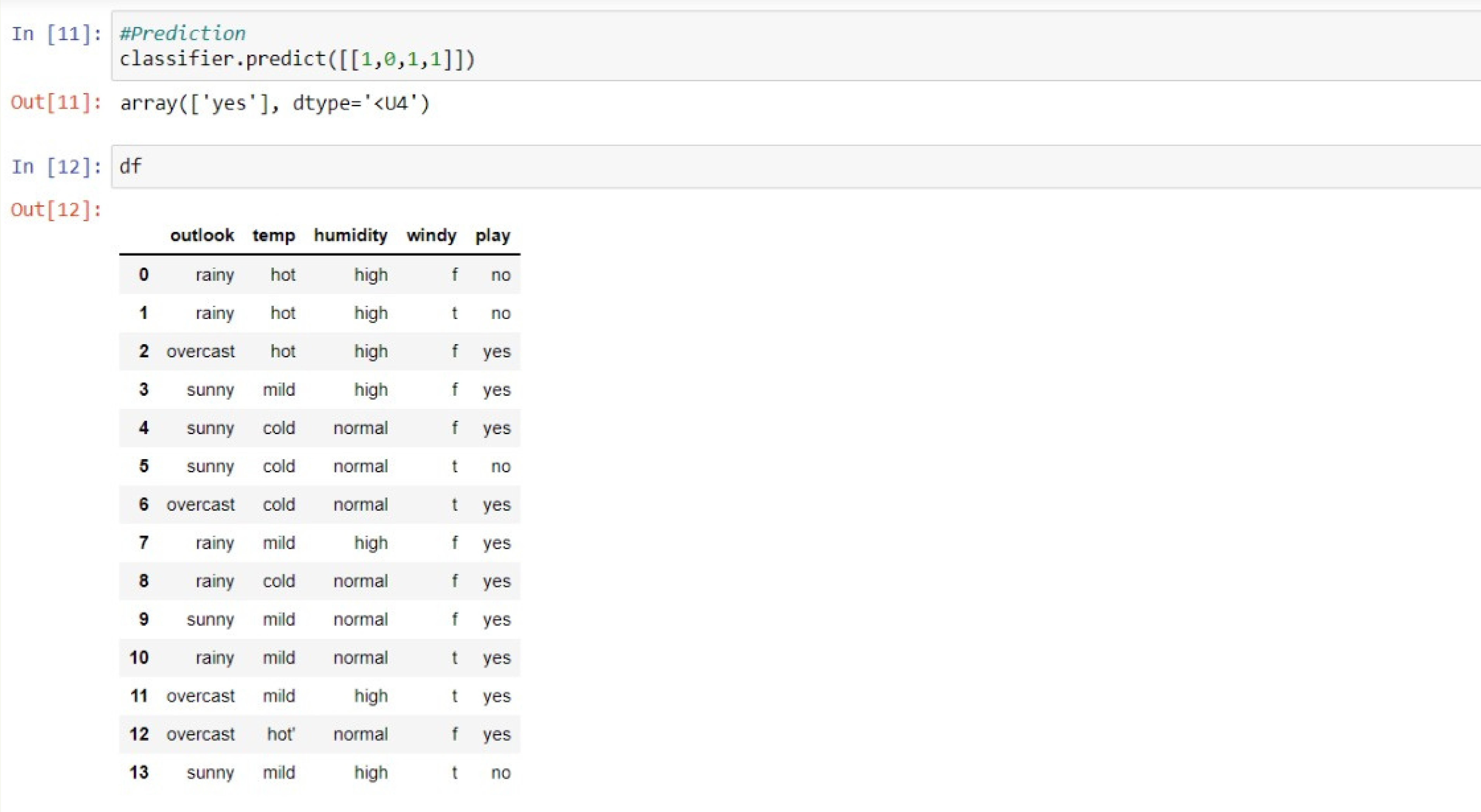
```
GaussianNB()
```

In [10]:

```
#85% accuracy
classifier.score(inputs_n,target)
```

Out[10]:

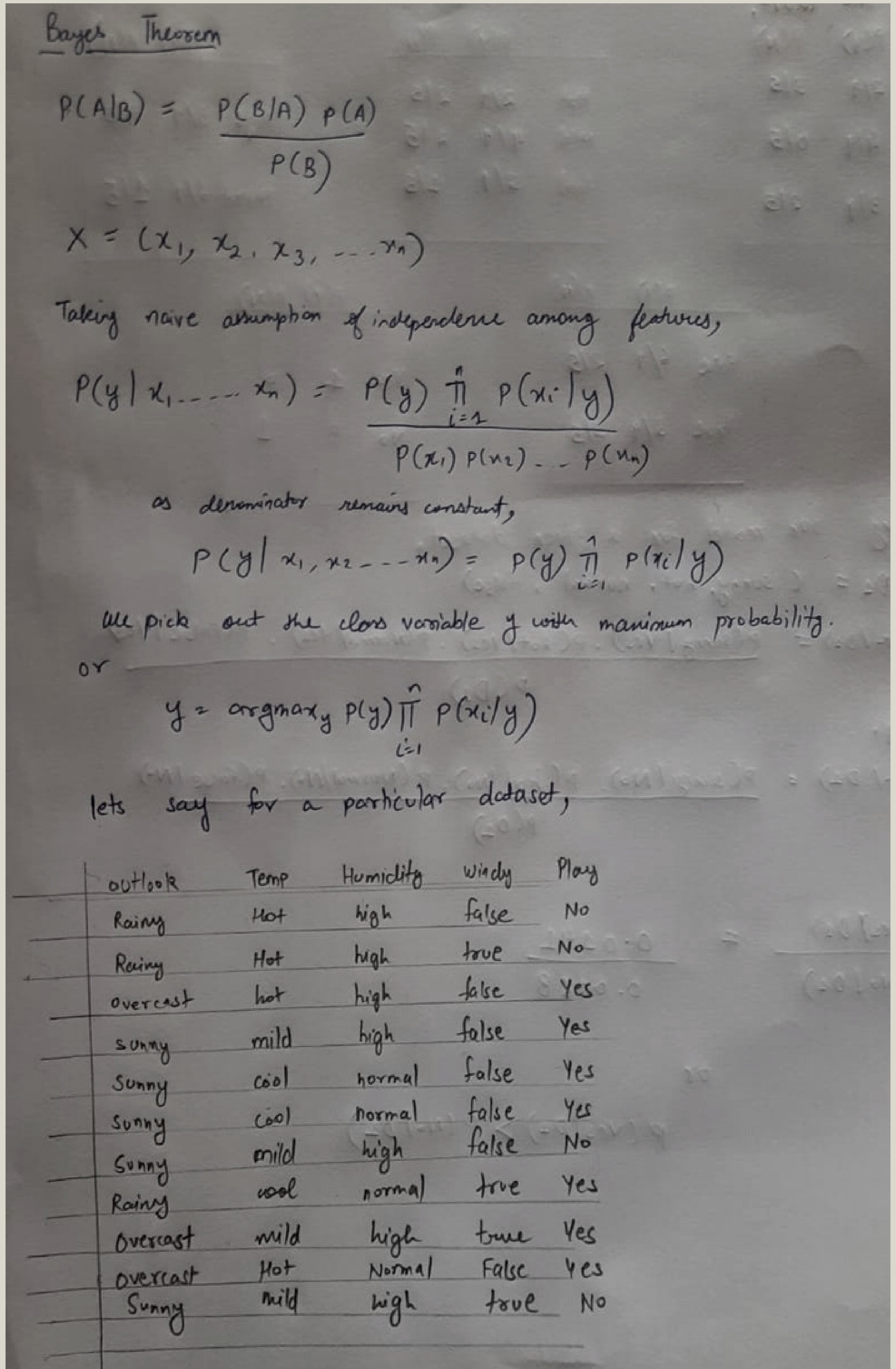
```
0.8571428571428571
```



```
In [11]: #Prediction
         classifier.predict([[1,0,1,1]])

Out[11]: array(['yes'], dtype='<U4')

In [12]: df

Out[12]:
```

	outlook	temp	humidity	windy	play
0	rainy	hot	high	f	no
1	rainy	hot	high	t	no
2	overcast	hot	high	f	yes
3	sunny	mild	high	f	yes
4	sunny	cold	normal	f	yes
5	sunny	cold	normal	t	no
6	overcast	cold	normal	t	yes
7	rainy	mild	high	f	yes
8	rainy	cold	normal	f	yes
9	sunny	mild	normal	f	yes
10	rainy	mild	normal	t	yes
11	overcast	mild	high	t	yes
12	overcast	hot'	normal	f	yes
13	sunny	mild	high	t	no

CALCULATIONS



Bayes Theorem

$$P(A|B) = \frac{P(B/A)\,p(A)}{P(B)}$$

$$X = (x_1, x_2, x_3, \ldots x_n)$$

Taking naive assumption of independence among features,

$$P(y|x_1 \ldots x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1)\,P(x_2)\ldots P(x_n)}$$

as denominator remains constant,

$$P(y|x_1, x_2 \ldots x_n) = P(y)\prod_{i=1}^{n} P(x_i/y)$$

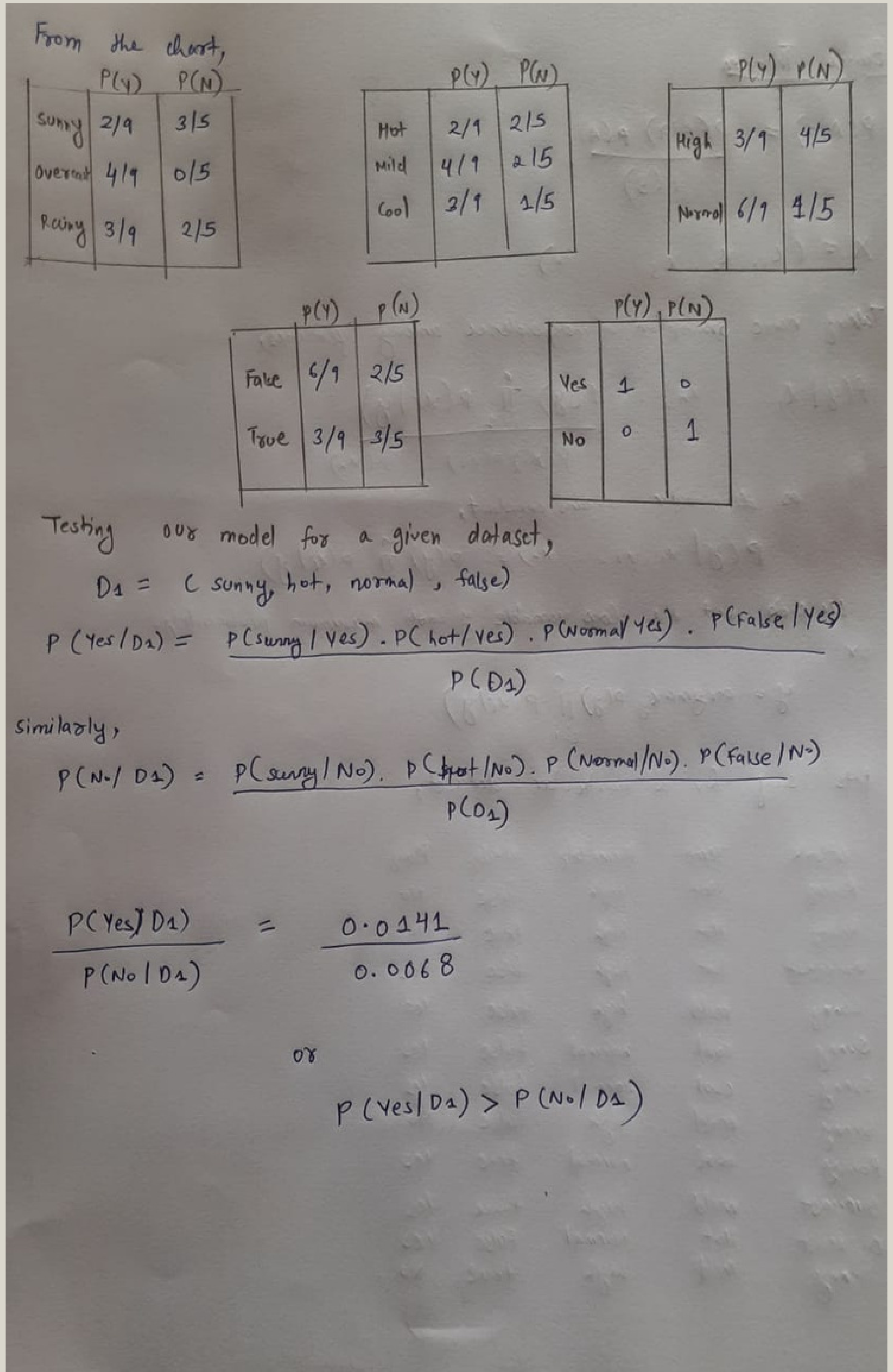
we pick out the class variable y with maximum probability.

or

$$y = \text{argmax}_y\, P(y)\prod_{i=1}^{n} P(x_i/y)$$

lets say for a particular dataset,

outlook	Temp	Humidity	windy	Play
Rainy	Hot	high	false	No
Rainy	Hot	high	true	No
overcast	hot	high	false	Yes
sunny	mild	high	false	Yes
Sunny	cool	normal	false	Yes
Sunny	cool	normal	false	Yes
Sunny	mild	high	false	No
Rainy	cool	normal	true	Yes
Overcast	mild	high	true	Yes
overcast	Hot	Normal	False	Yes
Sunny	mild	high	true	No



From the chart,

	P(Y)	P(N)
Sunny	2/9	3/5
Overcast	4/9	0/5
Rainy	3/9	2/5

	P(Y)	P(N)
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

	P(Y)	P(N)
High	3/9	4/5
Normal	6/9	1/5

	P(Y)	P(N)
False	6/9	2/5
True	3/9	3/5

	P(Y)	P(N)
Yes	1	0
No	0	1

Testing our model for a given dataset,

$$D_1 = (\text{sunny, hot, normal, false})$$

$$P(Yes/D_1) = \frac{P(sunny/Yes) \cdot P(hot/Yes) \cdot P(Normal/Yes) \cdot P(False/Yes)}{P(D_1)}$$

similarly,

$$P(No/D_1) = \frac{P(sunny/No) \cdot P(hot/No) \cdot P(Normal/No) \cdot P(False/No)}{P(D_1)}$$

$$\frac{P(Yes/D_1)}{P(No|D_1)} = \frac{0.0141}{0.0068}$$

or

$$P(Yes|D_1) > P(No/D_1)$$