In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [ ]:
```
df = pd.read_csv('mushrooms.csv')
df.head(n=10)
```

Out[ ]:

	type	cap_shape	cap_surface	cap_color	bruises	odor	gill_attachment	gill_spacing	gill_size	gill_co
0	p	x	s	n	t	p	f	c	n	
1	e	x	s	y	t	a	f	c	b	
2	e	b	s	w	t	l	f	c	b	
3	p	x	y	w	t	p	f	c	n	
4	e	x	s	g	f	n	f	w	b	
5	e	x	y	y	t	a	f	c	b	
6	e	b	s	w	t	a	f	c	b	
7	e	b	y	w	t	l	f	c	b	
8	p	x	y	w	t	p	f	c	n	
9	e	b	s	y	t	a	f	c	b	

10 rows × 23 columns

In [ ]:
```
df.shape
```

Out[ ]:
```
(8124, 23)
```

In [ ]:
```
df.describe()
```

Out[ ]:

	type	cap_shape	cap_surface	cap_color	bruises	odor	gill_attachment	gill_spacing	gill_size
count	8124	8124	8124	8124	8124	8124	8124	8124	8124
unique	2	6	4	10	2	9	2	2	2
top	e	x	y	n	f	n	f	c	b
freq	4208	3656	3244	2284	4748	3528	7914	6812	5612

4 rows × 23 columns

In [ ]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
```

In [ ]:
```
le = LabelEncoder()
```

In [ ]:
```
ds = df.apply(le.fit_transform) # Applies transform on each collumn
```

In [ ]:
```
ds
```

Out[ ]:

	type	cap_shape	cap_surface	cap_color	bruises	odor	gill_attachment	gill_spacing	gill_size	gi
0	1	5	2	4	1	6	1	0	1	
1	0	5	2	9	1	0	1	0	0	
2	0	0	2	8	1	3	1	0	0	
3	1	5	3	8	1	6	1	0	1	
4	0	5	2	3	0	5	1	1	0	
...	...	...	...	...	...	...	...	...	...	...
8119	0	3	2	4	0	5	0	0	0	
8120	0	5	2	4	0	5	0	0	0	
8121	0	2	2	4	0	5	0	0	0	
8122	1	3	3	4	0	8	1	0	1	
8123	0	5	2	4	0	5	0	0	0	

8124 rows × 23 columns

In [ ]:
```
data = ds.values
print(data.shape)
```

```
(8124, 23)
```

In [ ]:
```
print(data[:5,:])
data_y = data[:,0]
data_x = data[:,1:]
```

```
[[1 5 2 4 1 6 1 0 1 4 0 3 2 2 7 7 0 2 1 4 2 3 5]
 [0 5 2 9 1 0 1 0 0 4 0 2 2 2 7 7 0 2 1 4 3 2 1]
 [0 0 2 8 1 3 1 0 0 5 0 2 2 2 7 7 0 2 1 4 3 2 3]
 [1 5 3 8 1 6 1 0 1 5 0 3 2 2 7 7 0 2 1 4 2 3 5]
 [0 5 2 3 0 5 1 1 0 4 1 3 2 2 7 7 0 2 1 0 3 0 1]]
```

In [ ]:
```
x_train,x_test,y_train,y_test = train_test_split(data_x,data_y, test_size = 0.2)
```

In [ ]:
```
x_train.shape,y_train.shape
```

Out[ ]:
```
((6499, 22), (6499,))
```

In [ ]:
```
x_test.shape,y_test.shape
```

Out[ ]:
```
((1625, 22), (1625,))
```

In [ ]:
```
np.unique(y_train)
```

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

In [ ]:
```
a = np.array([0,0,0,1,1,0])
np.sum(a==1)
```

Out[ ]:
```
2
```

In [ ]:
```
def prior_prob(y_train,label):
    total_ex = y_train.shape[0]
    class_ex = np.sum(y_train==label)
    return (class_ex/float(total_ex))
```

In [ ]:
```
y = np.array([0,0,5,5,1,1,1,1,0,0])
```

In [ ]:
```
prior_prob(y,5)
```

Out[ ]:
```
0.2
```

In [ ]:
```
def cond_prob(x_train,y_train,feature_col,feature_val,label):
    x_filtered = x_train[y_train==label]
    numerator = np.sum(x_filtered[:,feature_col] == feature_val)
    denom = np.sum(y_train == label)
    return numerator/float(denom)
```

In [ ]:
```
def predict(x_train,y_train,xtest):
    classes = np.unique(y_train)
    n_features = x_train.shape[1]
    post_probs = []
    for label in classes:
        likelihood = 1.0
        for f in range(n_features):
            cond = cond_prob(x_train,y_train,f,xtest[f],label)
            likelihood *= cond
        prior = prior_prob(y_train,label)
        post = prior*likelihood
        post_probs.append(post)
    pred = np.argmax(post_probs)
    return pred
```

In [ ]:
```
output = predict(x_train,y_train,x_test[1])
print(output)
```

```
1
```

In [ ]:
```
print(y_test[1])
```

```
1
```

In [ ]:
```
def score(x_train,y_train,x_test,y_test):
    pred = []

    for i in range(x_test.shape[0]):
        pred_label = predict(x_train,y_train,x_test[i])
        pred.append(pred_label)
    pred = np.array(pred)
    accuracy = np.sum(pred==y_test)/y_test.shape[0]
    return accuracy
```

In [ ]:
```
print(score(x_train,y_train,x_test,y_test))
```

```
0.9987692307692307
```

In [ ]:
```

```