Assignment 3

Task 1 – Bayes' theorem

P(A|B) = (P(B|A) * P(A)) / P(B)

A.60% of the kids play football, and 36% of the kids play ice hockey. 40% of the kids who play football also play ice hockey.What percent of those that play ice hockey also play football?

In [7]:
```
 IceAndFootball = (0.4 * 0.6)/0.36
## The percent of kids who play football that also play ice hockey is about 66%:
print(IceAndFootball)
```

```
0.6666666666666666
```

40% of the kids like music, and 24% of the kids like to dance. Given that 30% of those that like music also likes to dance, what percent of those that like to dance also likes music?

In [8]:
```
DanceToMusic = (0.3 * 0.24)/0.4
## The percent of kids who like to dance that also like music is about 18%:
print(DanceToMusic)
```

```
0.17999999999999997
```

In a factory, machine X produces 60% of the daily output and machine Y produces 40% of the daily output. 2% of machine X's output is defective, and 1.5% of machine Y's output is defective.One day, an item is inspected at random, and found to be defective. What is the probability that it was produced by machine X?

In [21]:
```
xMachine = 0.6 * 0.02
yMachine = 0.4 * 0.015
```

The probability of a combined error for the y and x machine can be calculated as seen above. To find how big a percentile x is responsible for, we need to find the individual percentile that the machines are responcible for.

1 % of 0.018 is 0.00018, therefore we can calculate the individual percentiles.

In [19]:
```
0.006/0.00018
```

Out[19]:
```
33.33333333333333
```

In [20]:
```
0.012/0.00018
```

Out[20]:
```
66.66666666666666
```

This means that the probability of the defect output comming from machine X is about 66,7%

Task 2

Make a KNN classifier onthe IRIS datasetusing Python. Make sure to split the dataset into training and testing sets.

In [25]:
```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn import datasets
from sklearn.decomposition import PCA

import numpy as np
import pandas as pd
```

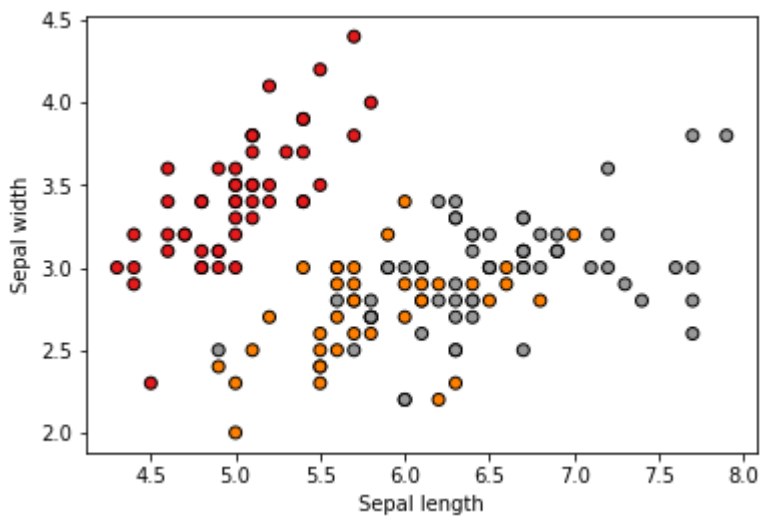
In [26]:
```
iris = datasets.load_iris()
```

In [54]:
```
type(iris.data)
y = iris.target
```

In [145]:
```
## Taking the first 2 collumns to make it easier. (length and width)
X = iris.data[:,:2]
```

In [57]:
```
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1,
            edgecolor='k')
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
```

Out[57]:
```
Text(0, 0.5, 'Sepal width')
```



In [139]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.2, random_stat
e = 2)
```

In [140]:
```
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train, y_train)
```

Out[140]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

In [141]:
```
y_pred = classifier.predict(X_test)
y_pred
```

Out[141]:
```
array([0, 0, 2, 0, 0, 2, 0, 2, 2, 0, 0, 0, 0, 0, 1, 1, 0, 1, 2, 1, 1, 1,
       2, 1, 1, 0, 0, 2, 0, 2])
```

In [142]:
```
y_test
```

Out[142]:
```
array([0, 0, 2, 0, 0, 2, 0, 2, 2, 0, 0, 0, 0, 0, 1, 1, 0, 1, 2, 1, 1, 1,
       2, 1, 1, 0, 0, 2, 0, 2])
```

In [143]:
```
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[14  0  0]
 [ 0  8  0]
 [ 0  0  8]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       1.00      1.00      1.00         8
           2       1.00      1.00      1.00         8

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

In [ ]:
```
```