In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:

```
mydata =sns.load_dataset('iris')
mydata.head()
```

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [5]:

```
mydata.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [7]:

```
mydata.isnull().sum()
```

Out[7]:

```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

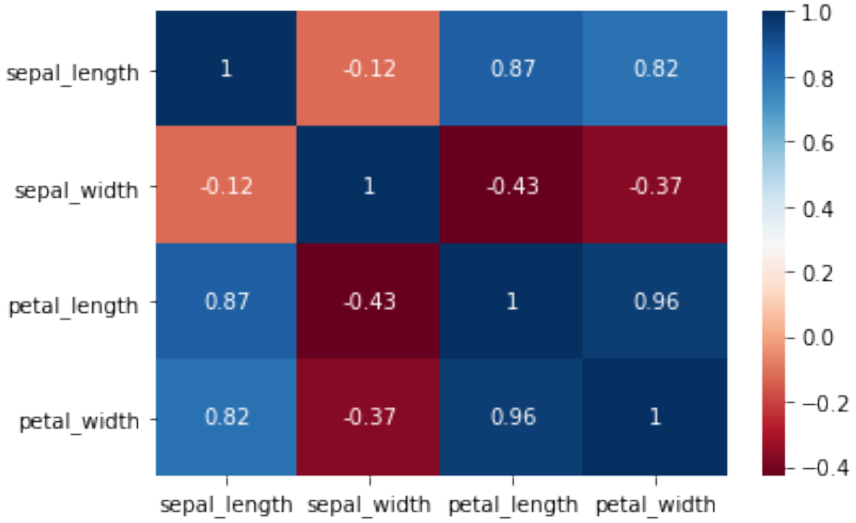
In [9]:
```
mydata.corr()
```

Out[9]:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.117570	0.871754	0.817941
sepal_width	-0.117570	1.000000	-0.428440	-0.366126
petal_length	0.871754	-0.428440	1.000000	0.962865
petal_width	0.817941	-0.366126	0.962865	1.000000

In [11]:
```
sns.heatmap(mydata.corr(),annot = True, cmap = 'RdBu')
```

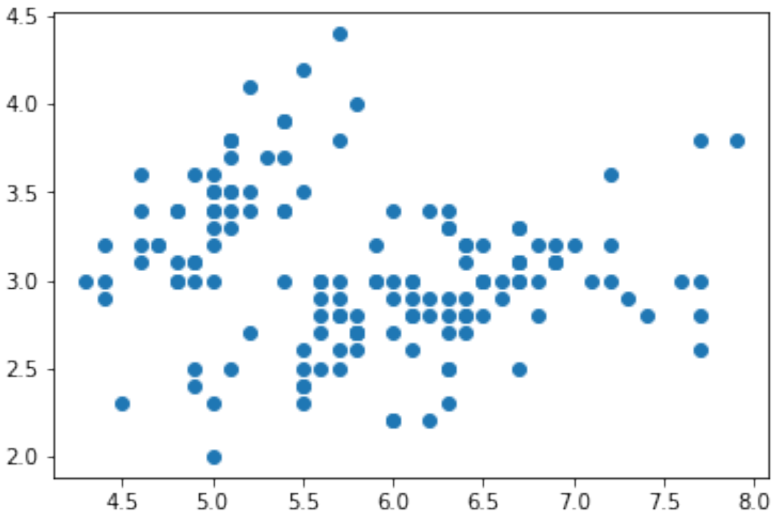
Out[11]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f96e2d82520>
```



In [4]:
```
mydata_col = mydata[['sepal_length','sepal_width']]
```

```
In [13]: plt.scatter(mydata['sepal_length'],mydata['sepal_width'])

Out[13]: <matplotlib.collections.PathCollection at 0x7f96e2eead60>
```



```
In [15]: x = mydata_col.values
         x
```

```
        [6.2, 2.8],
        [6.1, 3. ],
        [6.4, 2.8],
        [7.2, 3. ],
        [7.4, 2.8],
        [7.9, 3.8],
        [6.4, 2.8],
        [6.3, 2.8],
        [6.1, 2.6],
        [7.7, 3. ],
        [6.3, 3.4],
        [6.4, 3.1],
        [6. , 3. ],
        [6.9, 3.1],
        [6.7, 3.1],
        [6.9, 3.1],
        [5.8, 2.7],
        [6.8, 3.2],
        [6.7, 3.3],
        [6.7, 3. ],
```

```
In [16]: from sklearn.neighbors import NearestNeighbors
```

In [21]:

```
model = NearestNeighbors(n_neighbors = 3)
model.fit(x)
```

Out[21]:

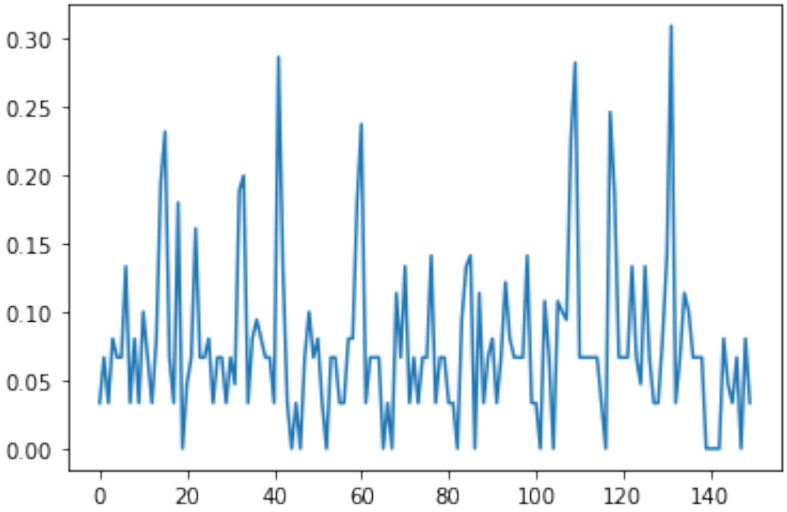
```
NearestNeighbors(n_neighbors=3)
```

In [23]:

```
distances,indexes = model.kneighbors(x)
plt.plot(distances.mean(axis = 1))
```

Out[23]:

```
[<matplotlib.lines.Line2D at 0x7f96e346c490>]
```



In [25]:

```
outlier = np.where(distances.mean(axis = 1) > 0.15)
outlier
```

Out[25]:

```
(array([ 14,  15,  18,  22,  32,  33,  41,  59,  60, 108, 109, 117, 1
18,
        131]),)
```

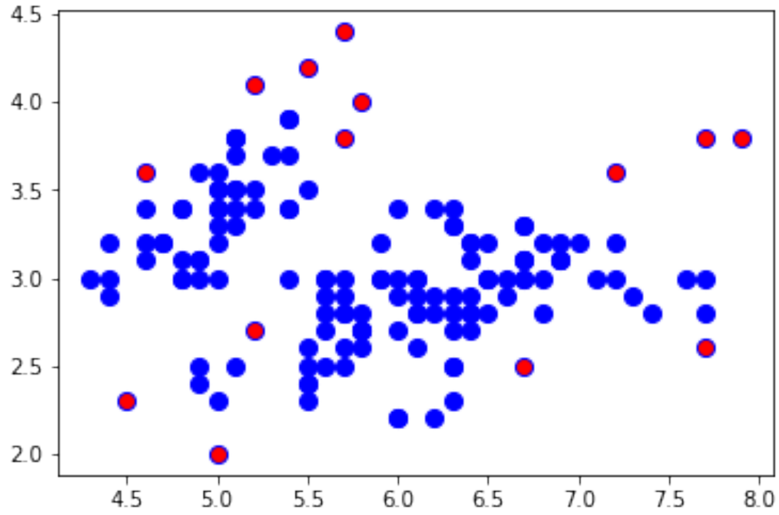
In [29]:
```
mydata_filtered = mydata.iloc[outlier]
mydata_filtered
```

Out[29]:

	sepal_length	sepal_width	petal_length	petal_width	species
14	5.8	4.0	1.2	0.2	setosa
15	5.7	4.4	1.5	0.4	setosa
18	5.7	3.8	1.7	0.3	setosa
22	4.6	3.6	1.0	0.2	setosa
32	5.2	4.1	1.5	0.1	setosa
33	5.5	4.2	1.4	0.2	setosa
41	4.5	2.3	1.3	0.3	setosa
59	5.2	2.7	3.9	1.4	versicolor
60	5.0	2.0	3.5	1.0	versicolor
108	6.7	2.5	5.8	1.8	virginica
109	7.2	3.6	6.1	2.5	virginica
117	7.7	3.8	6.7	2.2	virginica
118	7.7	2.6	6.9	2.3	virginica
131	7.9	3.8	6.4	2.0	virginica

In [31]:
```
plt.scatter(mydata['sepal_length'],mydata['sepal_width'],color = 'b',s
plt.scatter(mydata_filtered['sepal_length'],mydata_filtered['sepal_wic
```

Out[31]:
```
<matplotlib.collections.PathCollection at 0x7f96e3590940>
```



In [33]:
```
from sklearn.preprocessing import LabelEncoder
LE = LabelEncoder()
```

In [34]:
```
mydata_filtered["species"]=LE.fit_transform(mydata_filtered.species)
```

```
<ipython-input-34-3cc25ca3e265>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/panda
s-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
(https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)
  mydata_filtered["species"]=LE.fit_transform(mydata_filtered.species
)
```

In [42]:
```
y_dep = mydata_filtered["species"]
y_dep
len(y_dep)
```

Out[42]:
```
14
```

In [37]:

```
x_ind = mydata_filtered.drop("species",axis = 1)
x_ind
```

Out[37]:

	sepal_length	sepal_width	petal_length	petal_width
14	5.8	4.0	1.2	0.2
15	5.7	4.4	1.5	0.4
18	5.7	3.8	1.7	0.3
22	4.6	3.6	1.0	0.2
32	5.2	4.1	1.5	0.1
33	5.5	4.2	1.4	0.2
41	4.5	2.3	1.3	0.3
59	5.2	2.7	3.9	1.4
60	5.0	2.0	3.5	1.0
108	6.7	2.5	5.8	1.8
109	7.2	3.6	6.1	2.5
117	7.7	3.8	6.7	2.2
118	7.7	2.6	6.9	2.3
131	7.9	3.8	6.4	2.0

In [39]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test= train_test_split(x_ind,y_dep,train_size
from sklearn.neighbors import KNeighborsClassifier
```

In [40]:

```
x_ind.shape
```

Out[40]:

```
(14, 4)
```

In [41]:

```
y_dep.shape
```

Out[41]:

```
(14,)
```

In [43]:

```
np.sqrt(14)
```

Out[43]:

```
3.7416573867739413
```

In [45]:

```
knn =KNeighborsClassifier(n_neighbors = 5, p=3,metric ='euclidean')
```

In [46]:
```
knn.fit(x_train,y_train)
```

Out[46]:
```
KNeighborsClassifier(metric='euclidean', p=3)
```

In [47]:
```
from sklearn.metrics import confusion_matrix,classification_report,acc
```

In [49]:
```
y_pred=knn.predict(x_test)
y_pred
```

Out[49]:
```
array([2, 0, 0])
```

In [50]:
```
accuracy_score(y_test,y_pred)
```

Out[50]:
```
1.0
```

In [ ]: