Q1

In [1]:
```python
import pandas as pd
data = pd.read_csv('iris_01.csv')
df = pd.DataFrame()
df['sl'] = data['sepal length (cm)']
df['sw'] = data['sepal width (cm)']
df['pl'] = data['petal length (cm)']
df['pw'] = data['petal width (cm)']
df['class'] = data['Class']
df['predicted'] = data['Predicted_class']
```

In [2]:
```python
import statistics
def setMean(feature):
    m = statistics.mean(feature)
    for i in range(len(feature)):
        if feature[i]==0:
            feature[i]=m
    return feature

def setMedian(feature):
    m = statistics.median(feature)
    for i in range(len(feature)):
        if feature[i]==0:
            feature[i]=m
    return feature

def setMode(feature):
    m = statistics.mode(feature)
    for i in range(len(feature)):
        if feature[i]==0:
            feature[i]=m
    return feature

def setZero(feature):
    for i in range(len(feature)):
        if feature[i]==0:
            feature[i]=0
    return feature

def setMax(feature):
    m = feature.max()
    for i in range(len(feature)):
        if feature[i]==0:
            feature[i]=m
    return feature

df['sl'] = setMean(df['sl'])
df['sw'] = setMean(df['sw'])
df['pl'] = setMean(df['pl'])
df['pw'] = setMean(df['pw'])
df.head()
```

```
<ipython-input-2-87c2e8985537>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexin
g.html#returning-a-view-versus-a-copy
  feature[i]=m
```

Out[2]:

	sl	sw	pl	pw	class	predicted
0	5.1	3.5	1.4	0.2	0	0
1	4.9	3.0	1.4	0.2	0	0
2	4.7	3.2	1.3	0.2	0	0
3	4.6	3.1	1.5	0.2	0	0
4	5.0	3.6	1.4	0.2	0	0

Q2

In [17]:
```python
import numpy as np
matrix = np.array([[0,0,0],[0,0,0],[0,0,0]])

for i in range(len(df['class'])):
    if df['class'][i] == df['predicted'][i]:
        matrix[df['class'][i],df['class'][i]]+=1
    else :
        matrix[df['class'][i],df['predicted'][i]]+=1

print(matrix)
fp = matrix.sum(axis=1) - np.diag(matrix)
tp = np.diag(matrix)
fn = matrix.sum(axis=0) - np.diag(matrix)
tn = matrix.sum() - (fp+tp+fn)
print(fp,tp,fn,tn)

tpr = tp/(tp+fn)
tnr = tn/(tn+fp)
fpr = fp/(fp+tn)
fnr = fn/(fn+tp)
accuracy = (tp+tn)/(tp+tn+fp+fn)
precision = tp/(fp+tp)
recall = tpr

B = 2

f1 = 2 * precision*recall/(precision+recall)
fB = (1+B**2) * precision*recall /(B**2 * (precision+recall))
print("TPR : ",tpr)
print("FPR : ",fpr)
print("TNR : ",tnr)
print("FNR : ",fnr)
print("Accuracy : ",accuracy)
print("F1 Score : ",f1)
print("fB score : ",fB)
```

```
[[46  2  2]
 [ 0 45  5]
 [ 2  4 44]]
[4 5 6] [46 45 44] [2 6 7] [98 94 93]
TPR :  [0.95833333 0.88235294 0.8627451 ]
FPR :  [0.03921569 0.05050505 0.06060606]
TNR :  [0.96078431 0.94949495 0.93939394]
FNR :  [0.04166667 0.11764706 0.1372549 ]
Accuracy :  [0.96       0.92666667 0.91333333]
F1 Score :  [0.93877551 0.89108911 0.87128713]
fB score :  [0.58673469 0.55693069 0.54455446]
```

Q4

In [26]:
```python
from sklearn.preprocessing import normalize
x = df[['sl','sw','pl','pw']]
y = df['class']
x_normalized = normalize(x)
print(x)
print(x_normalized)
df_temp = pd.DataFrame()
df_temp['normalized_x'] = x_normalized
df_temp.to_csv('normalized.csv',index=False)
```

```
        sl        sw   pl   pw
0    5.100  3.500000  1.4  0.2
1    4.900  3.000000  1.4  0.2
2    4.700  3.200000  1.3  0.2
3    4.600  3.100000  1.5  0.2
4    5.000  3.600000  1.4  0.2
..     ...       ...  ...  ...
145  5.354  3.000000  5.2  2.3
146  6.300  2.500000  5.0  1.9
147  6.500  3.000000  5.2  2.0
148  6.200  2.933333  5.4  2.3
149  5.900  3.000000  5.1  1.8

[150 rows x 4 columns]
[[0.80377277 0.55160877 0.22064351 0.0315205 ]
 [0.82813287 0.50702013 0.23660939 0.03380134]
 [0.80533308 0.54831188 0.2227517  0.03426949]
 [0.80003025 0.53915082 0.26087943 0.03478392]
 [0.790965   0.5694948  0.2214702  0.0316386 ]
 [0.78158239 0.56932598 0.24816774 0.05839241]
 [0.78010936 0.57660257 0.23742459 0.0508767 ]
 [0.80218492 0.54548574 0.24065548 0.0320874 ]
 [0.80642366 0.5315065  0.25658935 0.03665562]
 [0.81803119 0.51752994 0.25041771 0.01669451]
 [0.80373519 0.55070744 0.22325977 0.02976797]
 [0.81814085 0.51955153 0.24449484 0.03056185]
 [0.82307218 0.51442011 0.24006272 0.01714734]
 [0.8025126  0.55989251 0.20529392 0.01866308]
 [0.81120865 0.55945424 0.16783627 0.02797271]
 [0.77381111 0.59732787 0.2036345  0.05430253]
 [0.79177176 0.5767482  0.1922494  0.05915366]
 [0.80327412 0.55126656 0.22050662 0.04725142]
 [0.8068282  0.53788547 0.24063297 0.04246464]
 [0.77964883 0.58091482 0.22930848 0.0458617 ]
 [0.8173379  0.51462016 0.25731008 0.03027177]
 [0.78591858 0.57017622 0.23115252 0.06164067]
 [0.81967348 0.55114392 0.15309553 0.03061911]
 [0.80597792 0.52151512 0.26865931 0.07901744]
 [0.80795665 0.49375129 0.31981618 0.03366486]
 [0.82647451 0.4958847  0.26447184 0.03305898]
 [0.79778206 0.5424918  0.25529026 0.06382256]
 [0.80641965 0.54278246 0.23262105 0.03101614]
 [0.82393303 0.52322979 0.21544756 0.03077822]
 [0.79524064 0.54144043 0.27072022 0.03384003]
 [0.80846584 0.52213419 0.26948861 0.03368608]
 [0.82225028 0.51771314 0.22840286 0.06090743]
 [0.77492285 0.59342243 0.21710577 0.01447372]
 [0.77867447 0.59462414 0.19820805 0.02831544]
 [0.81768942 0.51731371 0.25031309 0.03337508]
 [0.82512295 0.52807869 0.19802951 0.03300492]
 [0.82699754 0.52627116 0.19547215 0.03007264]
 [0.78523221 0.5769053  0.22435206 0.01602515]
 [0.78576832 0.53575113 0.23215882 0.20406165]
 [0.8395196  0.48286095 0.24691753 0.03292234]
 [0.80033301 0.56023311 0.20808658 0.04801998]
 [0.86093857 0.44003527 0.24871559 0.0573959 ]
 [0.78609038 0.57170209 0.23225397 0.03573138]
 [0.8000764  0.52302342 0.23909642 0.17075469]
 [0.76693897 0.57144472 0.28572236 0.06015208]
 [0.82210585 0.51381615 0.23978087 0.05138162]
 [0.77729093 0.57915795 0.24385598 0.030482  ]
 [0.79594782 0.55370283 0.24224499 0.03460643]
 [0.79837025 0.55735281 0.22595384 0.03012718]
 [0.81228363 0.5361072  0.22743942 0.03249135]
 [0.76701103 0.35063361 0.51499312 0.15340221]
 [0.74549757 0.37274878 0.52417798 0.17472599]
 [0.75519285 0.33928954 0.53629637 0.16417236]
 [0.74501136 0.32004597 0.55660169 0.18089555]
 [0.7581754  0.32659863 0.5365549  0.17496355]
 [0.72232962 0.35482858 0.57026022 0.16474184]
 [0.72634846 0.38046824 0.54187901 0.18446945]
 [0.75916547 0.37183615 0.51127471 0.15493173]
 [0.76301853 0.33526572 0.53180079 0.15029153]
 [0.72460233 0.37623583 0.54345175 0.19508524]
 [0.73045692 0.42853473 0.51131984 0.14609138]
 [0.73923462 0.37588201 0.52623481 0.187941  ]
 [0.7533726  0.30956663 0.56284841 0.1407121 ]
 [0.73081412 0.34743622 0.56308629 0.16772783]
 [0.75911707 0.3931142  0.48800383 0.17622361]
 [0.76945444 0.35601624 0.50531337 0.16078153]
 [0.70631892 0.37838513 0.5675777  0.18919257]
 [0.75676497 0.35228714 0.53495455 0.13047672]
 [0.76444238 0.27125375 0.55483721 0.18494574]
 [0.76185188 0.34011245 0.53057542 0.14964948]
 [0.6985796  0.37889063 0.56833595 0.21312598]
 [0.76546929 0.36809452 0.50194708 0.1631328 ]
 [0.74143307 0.29421947 0.57667016 0.17653168]
 [0.73659895 0.33811099 0.56754345 0.14490471]
 [0.76954607 0.34870056 0.51703877 0.13739604]
 [0.76785726 0.34902603 0.51190484 0.16287881]
 [0.76467269 0.31486523 0.53976896 0.15743261]
 [0.74088576 0.33173989 0.55289982 0.18798594]
 [0.73350949 0.35452959 0.55013212 0.18337737]
 [0.78667474 0.35883409 0.48304589 0.13801311]
 [0.76521855 0.33391355 0.52869645 0.15304371]
 [0.77242925 0.33706004 0.51963422 0.14044168]
 [0.76434981 0.35581802 0.51395936 0.15814134]
 [0.78360611 0.35262275 0.46685511 0.20896163]
 [0.69333409 0.38518561 0.57777841 0.1925928 ]
 [0.71524936 0.40530797 0.53643702 0.19073316]
 [0.75457341 0.34913098 0.52932761 0.16893434]
 [0.77530021 0.28304611 0.54147951 0.15998258]
 [0.72992443 0.39103094 0.53440896 0.16944674]
 [0.74714194 0.33960997 0.54337595 0.17659719]
 [0.72337118 0.34195729 0.57869695 0.15782644]
 [0.73260391 0.36029701 0.55245541 0.1681386 ]
 [0.76262994 0.34186859 0.52595168 0.1577855 ]
 [0.76986879 0.35413965 0.5081134  0.15397376]
 [0.73544284 0.35458851 0.55158213 0.1707278 ]
 [0.76368653 0.40194028 0.47893417 0.16077611]
 [0.73446047 0.37367287 0.5411814  0.16750853]
 [0.75728103 0.3542121  0.52521104 0.15878473]
 [0.7970182  0.37216016 0.44659219 0.16375047]
 [0.7431482  0.36505526 0.5345452  0.16948994]
 [0.65387747 0.34250725 0.62274045 0.25947519]
 [0.69052512 0.32145135 0.60718588 0.22620651]
 [0.71491405 0.30207636 0.59408351 0.21145345]
 [0.69276796 0.31889319 0.61579374 0.1979337 ]
 [0.68619022 0.31670318 0.61229281 0.232249  ]
 [0.57856468 0.32418641 0.7132101  0.22693049]
 [0.67054118 0.34211284 0.61580312 0.23263673]
 [0.71366557 0.28351098 0.61590317 0.17597233]
 [0.71414125 0.26647062 0.61821183 0.19185884]
 [0.70634266 0.28776923 0.59842919 0.24525787]
 [0.71562645 0.3523084  0.56149152 0.22019275]
 [0.71576546 0.30196356 0.59274328 0.21249287]
 [0.71718148 0.31640359 0.58007326 0.22148252]
 [0.6925518  0.30375079 0.60750157 0.24300063]
 [0.67767924 0.32715549 0.59589036 0.28041899]
 [0.69589887 0.34794944 0.57629125 0.25008866]
 [0.70610474 0.3258945  0.59747324 0.1955367 ]
 [0.69299099 0.34199555 0.60299216 0.19799743]
 [0.70600618 0.2383917  0.63265489 0.21088496]
 [0.72712585 0.26661281 0.60593821 0.18178146]
 [0.70558934 0.32722984 0.58287815 0.23519645]
 [0.69719677 0.34859838 0.61004717 0.14226134]
 [0.71486543 0.25995106 0.62202576 0.18567933]
 [0.73122464 0.31338199 0.56873028 0.20892133]
 [0.69595601 0.3427843  0.59208198 0.21813547]
 [0.81472655 0.36210069 0.40449664 0.20368164]
 [0.72785195 0.32870733 0.56349829 0.21131186]
 [0.71171214 0.35002236 0.57170319 0.21001342]
 [0.70907571 0.31022063 0.62044125 0.12659956]
 [0.73089855 0.30454106 0.58877939 0.1624219 ]
 [0.83262701 0.31504806 0.40221135 0.21378261]
 [0.71578999 0.34430405 0.5798805  0.18121266]
 [0.69417747 0.30370264 0.60740528 0.2386235 ]
 [0.72366005 0.32162669 0.58582004 0.17230001]
 [0.69385414 0.29574111 0.63698085 0.15924521]
 [0.82858836 0.32282663 0.38466587 0.24750042]
 [0.67017484 0.36168166 0.59571097 0.2553047 ]
 [0.69804799 0.338117   0.59988499 0.196326  ]
 [0.71066905 0.35533453 0.56853524 0.21320072]
 [0.72415258 0.32534391 0.56672811 0.22039426]
 [0.69997037 0.32386689 0.58504986 0.25073566]
 [0.66108337 0.38277147 0.6297208  0.14109039]
 [0.69052512 0.32145135 0.60718588 0.22620651]
 [0.69193502 0.32561648 0.60035539 0.23403685]
 [0.68914871 0.33943145 0.58629069 0.25714504]
 [0.63994681 0.35858058 0.62153967 0.27491178]
 [0.72965359 0.28954508 0.57909015 0.22005426]
 [0.71653899 0.3307103  0.57323119 0.22047353]
 [0.68679589 0.32493569 0.59817707 0.25477912]
 [0.69025916 0.35097923 0.5966647  0.21058754]]

---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
<ipython-input-26-53bac83b9b94> in <module>
      6 print(x_normalized)
      7 df_temp = pd.DataFrame()
----> 8 df_temp['normalized_x'] = x_normalized
      9 df_temp.to_csv('normalized.csv',index=False)

D:\Anaconda\lib\site-packages\pandas\core\frame.py in __setitem__(self, key, value)
   2936         else:
   2937             # set column
-> 2938             self._set_item(key, value)
   2939
   2940     def _setitem_slice(self, key, value):

D:\Anaconda\lib\site-packages\pandas\core\frame.py in _set_item(self, key, value)
   2997         """
   2998
-> 2999         self._ensure_valid_index(value)
   3000         value = self._sanitize_column(key, value)
   3001         NDFrame._set_item(self, key, value)

D:\Anaconda\lib\site-packages\pandas\core\frame.py in _ensure_valid_index(self, value)
   3052         if not len(self.index) and is_list_like(value) and len(value):
   3053             try:
-> 3054                 value = Series(value)
   3055             except (ValueError, NotImplementedError, TypeError):
   3056                 raise ValueError(

D:\Anaconda\lib\site-packages\pandas\core\series.py in __init__(self, data, index, dtype, name, copy,
 fastpath)
    303                     data = data.copy()
    304             else:
--> 305                 data = sanitize_array(data, index, dtype, copy, raise_cast_failure=True)
    306
    307                 data = SingleBlockManager(data, index, fastpath=True)

D:\Anaconda\lib\site-packages\pandas\core\construction.py in sanitize_array(data, index, dtype, copy,
 raise_cast_failure)
    480     elif subarr.ndim > 1:
    481         if isinstance(data, np.ndarray):
--> 482             raise Exception("Data must be 1-dimensional")
    483         else:
    484             subarr = com.asarray_tuplesafe(data, dtype=dtype)

Exception: Data must be 1-dimensional
```

Q5

In [18]:
```python
from sklearn.datasets import load_iris
dataset = load_iris()
x = dataset['data']
y = dataset['target']
cm = np.cov(x.T)
values,vectors = np.linalg.eig(cm)
pc1 = x.dot(vectors.T[0])
pc2 = x.dot(vectors.T[1])
print(pc1,pc2)
```

```
[2.81823951 2.78822345 2.61337456 2.75702228 2.7736486  3.2215055
 2.68182738 2.87622016 2.6159824  2.82960933 2.99541804 2.8896099
 2.71625587 2.27856139 2.85761474 3.1163261  2.87883726 2.85406843
 3.30254481 2.91437873 3.19210892 2.9586599  2.28642572 3.19963195
 3.14661108 2.99569623 3.03354506 2.94004523 2.86283042 2.87037575
 2.91496666 3.09243264 2.8535028  2.90362838 2.86543825 2.63612348
 2.87712708 2.70168102 2.52186309 2.91235882 2.73226271 2.65299643
 2.50495859 3.09675065 3.29287589 2.78791371 2.96421687 2.66290296
 2.95927938 2.79900535 6.78719082 6.43485366 6.96666745 5.68568285
 6.59046839 6.14403422 6.5974258  4.75324246 6.54649696 5.49361973
 4.99452425 6.01406369 5.76734164 6.48729964 5.32843976 6.43022591
 6.16264889 5.73847013 6.44709886 5.54759211 6.61864831 5.86025355
 6.80054901 6.42409406 6.21721846 6.40253951 6.83438957 7.06016729
 6.31565578 5.19678135 5.43423864 5.31274266 5.63879384 6.88239157
 6.09037158 6.30922345 6.72305602 6.31746037 5.74832281 5.66877835
 5.96716542 6.39318033 5.73291316 4.79783337 5.85934663 5.83429961
 5.87858078 6.14494114 4.59589527 5.80136597 8.03355786 6.91760101
 8.11904115 7.47389619 7.85237105 8.89940387 6.02359738 8.4349522
 7.82359395 8.4191161  7.16413929 7.30576709 7.66795693 6.84852871
 7.08829336 7.40682151 7.45205419 8.9894205  9.29801055 6.80315685
 7.93018305 6.70136624 9.00228517 6.89113126 7.77779564 8.11645561
 6.76087329 6.79349719 7.62597386 7.89036815 8.34403791 8.73303879
 7.66180278 6.94652637 7.28365994 8.57886506 7.64660845 7.40746328
 6.67169147 7.60997628 7.81651984 7.42463293 6.91760101 8.06537851
 7.92111132 7.44647493 7.02953175 7.26671085 7.40330675 6.89255399] [-5.64634982 -5.14995135 -5.18200
315 -5.0086536  -5.65370709 -6.06828303
 -5.23749119 -5.49033754 -4.74864082 -5.21317833 -5.97202148 -5.34168252
 -5.09184058 -4.81555799 -6.50571721 -6.66501491 -6.13763209 -5.63880172
 -6.19979162 -5.84051289 -5.71829851 -5.75994864 -5.46042065 -5.42566143
 -5.28967072 -5.1809357  -5.45790407 -5.69467143 -5.63899256 -5.12999135
 -5.12263409 -5.73787684 -6.1403164  -6.42009834 -5.20563023 -5.39631705
 -5.9263226  -5.59559631 -4.83899423 -5.55599641 -5.59048011 -4.385992
 -4.98502652 -5.51582401 -5.76361572 -5.07674437 -5.83072372 -5.09900701
 -5.9063626  -5.43465866 -6.01211305 -5.64528622 -5.83121539 -4.49899357
 -5.40154325 -4.90870571 -5.61042085 -4.32206162 -5.55531448 -4.60387067
 -4.06098139 -5.22297134 -4.77691611 -5.20213472 -5.07209837 -5.79413207
 -4.97398291 -4.99334181 -4.78380703 -4.7431182  -5.24233572 -5.25802755
 -4.99916527 -5.14421478 -5.47600852 -5.65545705 -5.57139345 -5.59444802
 -5.16360228 -4.95869039 -4.62178045 -4.64666581 -5.01292014 -4.90599829
 -4.84266516 -5.52113489 -5.73457217 -4.95491552 -5.05842818 -4.64502585
 -4.65624103 -5.29248813 -4.92256673 -4.31470435 -4.82204248 -5.11429789
 -5.03373365 -5.34469077 -4.57085921 -4.97805477 -5.31710347 -4.75203623
 -5.67085573 -5.14722467 -5.28669163 -5.87778925 -4.13419385 -5.68245258
 -5.08312107 -6.10974453 -5.56918098 -5.11131496 -5.54322816 -4.55013423
 -4.78731186 -5.44620327 -5.36889584 -6.50269191 -5.58427555 -4.56580294
 -5.70514859 -4.72086105 -5.78762668 -5.12255325 -5.66194318 -5.88785393
 -5.14724778 -5.21028393 -5.1172231  -5.79159238 -5.70222174 -6.70111766
 -5.109675   -5.18353917 -4.8270509  -6.01503825 -5.46701678 -5.3762531
 -5.16196231 -5.69924045 -5.51060386 -5.73615604 -4.75203623 -5.60481518
 -5.63175077 -5.51448488 -4.95163559 -5.40581143 -5.44358054 -5.04429164]
```

Q6

In [20]:
```python
data2 = pd.read_csv('Boston_Housing_01.csv')
data2.head()
```

Out[20]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	Predicted
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0	27.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6	22.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7	32.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4	32.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2	34.2

In [23]:
```python
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score

def meanAbsoluteError(actual,predicted):
    sum = 0
    for i in range(len(actual)):
        sum+= abs(actual[i] - predicted[i])
    return sum/len(actual)

def meanSquaredError(actual,predicted):
    sum = 0
    for i in range(len(actual)):
        sum += ((actual[i] - predicted[i])**2)
    return sum/len(actual)

def rootMeanSquaredError(actual,predicted):
    return np.sqrt(meanSquaredError(actual,predicted))

def correlation(actual,predicted):
    xy = actual.dot(predicted)
    x2 = actual.dot(actual)
    y2 = predicted.dot(predicted)

    n = len(actual)*xy.sum() - (actual.sum()*predicted.sum())
    d = (len(actual)*x2.sum() - (actual.sum())**2) * (len(predicted)*y2.sum() - (predicted.sum())**2)

    return n/np.sqrt(d)

def r2Score(actual,predicted):
    return correlation(actual,predicted)**2


print("MAE : ",meanAbsoluteError(data2['MEDV'],data2['Predicted']))
print("MSE : ",meanSquaredError(data2['MEDV'],data2['Predicted']))
print("RMSE : ",rootMeanSquaredError(data2['MEDV'],data2['Predicted']))
print("Correlation Coeeficient : ",correlation(data2['MEDV'],data2['Predicted']))
print("R2 Score : ",r2Score(data2['MEDV'],data2['Predicted']))
```

```
MAE :  1.6956521739130435
MSE :  3.964426877470356
RMSE :  1.9910868583440442
Correlation Coeeficient :  0.9771112761435973
R2 Score :  0.9547464459669692
```

In [ ]: