www.bit.ly/mstp-view

KNN Algorithm

In pattern recognition, the k-nearest neighbors algorithm is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space

Lets consider the data

Height (in cms)	Weight (in kgs)	T Shirt Size
158	58	M
158	59	M
158	63	M
160	59	M
160	60	M
163	60	M
163	61	M
160	64	L
163	64	L
165	61	L
165	62	L
165	65	L
168	62	L
168	63	L
168	66	L
170	63	L
170	64	L
170	68	L

In [1]:

```
m=0
l=0

H=[158,158,158,160,163,163,160,163,163,165,165,168,168,170,170]
W=[58,59,63,59,60,60,61,64,64,61,62,65,62,63,66]
T=['m','m','m','m','m','m','m','m','m','l','l','l','l','l','l','l']
H1=161
W1=61
```

In [2]:

```
li = []
for i in range(0,len(H)):
    d = (((H[i]-H1) ** 2) + ((W[i]-W1) ** 2)) ** 0.5
    li.append(d)
li
```

Out[2]:

```
[4.242640687119285,
 3.605551275463989,
 3.605551275463989,
 2.23606797749979,
 2.23606797749979,
 2.23606797749979,
 1.0,
 3.605551275463989,
 3.605551275463989,
 4.0,
 4.123105625617661,
 8.06225774829855,
 7.0710678118654755,
 9.219544457292887,
 10.295630140987]
```

In [3]:

```
import numpy as np

h = np.array(H)
w = np.array(W)
np.sqrt(((h-H1) ** 2) + ((w-W1)**2))
```

Out[3]:

```
array([ 4.24264069,  3.60555128,  3.60555128,  2.23606798,  2.23606798,
        2.23606798,  1.        ,  3.60555128,  3.60555128,  4.        ,
        4.12310563,  8.06225775,  7.07106781,  9.21954446, 10.29563014])
```

In [4]:

```
import pandas as pd
shirt = pd.DataFrame({'Height':H,'Weight':W,'Size':T})
shirt
```

Out[4]:

	Height	Weight	Size
0	158	58	m
1	158	59	m
2	158	63	m
3	160	59	m
4	163	60	m
5	163	60	m
6	160	61	m
7	163	64	m
8	163	64	l
9	165	61	l
10	165	62	l
11	168	65	l
12	168	62	l
13	170	63	l
14	170	66	l

In [5]:

```
((shirt['Height'] - H1) ** 2 + (shirt['Weight'] - W1) ** 2) ** 0.5
```

Out[5]:

```
0      4.242641
1      3.605551
2      3.605551
3      2.236068
4      2.236068
5      2.236068
6      1.000000
7      3.605551
8      3.605551
9      4.000000
10     4.123106
11     8.062258
12     7.071068
13     9.219544
14    10.295630
dtype: float64
```

In [6]:

```
shirt.shape
```

Out[6]:

```
(15, 3)
```

In [7]:

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.scatterplot(x = shirt['Height'],y = shirt['Weight'],hue = shirt['Size'],label = 'He
ight Vs Weight',legend=False)
plt.scatter(161,61,color='r',marker='.',label = 'New Value')
plt.legend()
```

Out[7]:

```
<matplotlib.legend.Legend at 0x1b6cd360240>
```

In [8]:

```
X = shirt[['Height','Weight']]
Y = shirt['Size']
```

In [9]:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split


X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size = 0.3,random_state = 42)

knn = KNeighborsClassifier(n_neighbors = 3)

knn.fit(X_train,Y_train)
```

Out[9]:

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=None, n_neighbors=3, p=2,
           weights='uniform')
```

In [10]:

```
knn.predict([[161,61]])
```

Out[10]:

```
array(['m'], dtype=object)
```

In [11]:

```
from sklearn.metrics import accuracy_score

Y_pred = knn.predict(X_test)
accuracy_score(Y_test,Y_pred)
```

Out[11]:

```
1.0
```

In [12]:

```
heart = pd.read_csv('datasets/Heart_disease.csv',index_col = 0)
heart.head()
```

Out[12]:

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope
1	63	1	typical	145	233	1	2	150	0	2.3	3
2	67	1	asymptomatic	160	286	0	2	108	1	1.5	2
3	67	1	asymptomatic	120	229	0	2	129	1	2.6	2
4	37	1	nonanginal	130	250	0	0	187	0	3.5	3
5	41	0	nontypical	130	204	0	2	172	0	1.4	1

Attribute	Information
1	age
2	sex
3	chestpain type (4 values)
4	resting blood pressure
5	serum cholestoral in mg/dl
6	fasting blood sugar > 120 mg/dl
7	resting electrocardiographic results (values 0,1,2)
8	maximum heart rate achieved
9	exercise induced angina
10	oldpeak = ST depression induced by exercise relative to rest
11	the slope of the peak exercise ST segment
12	number of major vessels (0-3) colored by flourosopy
13	**thal:** 3 = normal; 6 = fixed defect; 7 = reversable defect
14	AHD alveolar hydatid disease

In [13]:

```
heart['ChestPain'].value_counts()
```

Out[13]:

```
asymptomatic    144
nonanginal       86
nontypical       50
typical          23
Name: ChestPain, dtype: int64
```

In [14]:

```
heart.shape
```

Out[14]:

```
(303, 14)
```

In [15]:

```
heart.describe()
```

Out[15]:

	Age	Sex	RestBP	Chol	Fbs	RestECG	MaxHR
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.438944	0.679868	131.689769	246.693069	0.148515	0.990099	149.607261
std	9.038662	0.467299	17.599748	51.776918	0.356198	0.994971	22.875003
min	29.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000
25%	48.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000
50%	56.000000	1.000000	130.000000	241.000000	0.000000	1.000000	153.000000
75%	61.000000	1.000000	140.000000	275.000000	0.000000	2.000000	166.000000
max	77.000000	1.000000	200.000000	564.000000	1.000000	2.000000	202.000000

In [16]:

```
heart.isnull().sum()
```

Out[16]:

```
Age          0
Sex          0
ChestPain    0
RestBP       0
Chol         0
Fbs          0
RestECG      0
MaxHR        0
ExAng        0
Oldpeak      0
Slope        0
Ca           4
Thal         2
AHD          0
dtype: int64
```

In [17]:

```
heart.dropna(inplace=True)
heart.isnull().sum()
```

Out[17]:

```
Age          0
Sex          0
ChestPain    0
RestBP       0
Chol         0
Fbs          0
RestECG      0
MaxHR        0
ExAng        0
Oldpeak      0
Slope        0
Ca           0
Thal         0
AHD          0
dtype: int64
```

In [18]:

```
heart.shape
```

Out[18]:

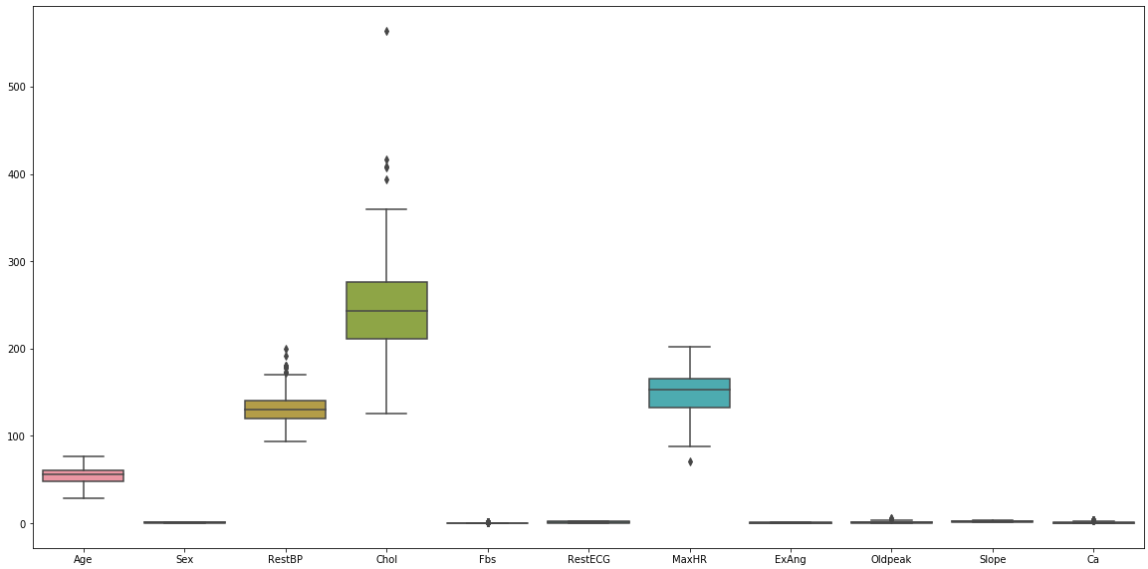
```
(297, 14)
```

In [19]:

```
import seaborn as sns
plt.figure(figsize=(20,10))
sns.boxplot(data = heart)
```

Out[19]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1b6cd3d9a20>
```

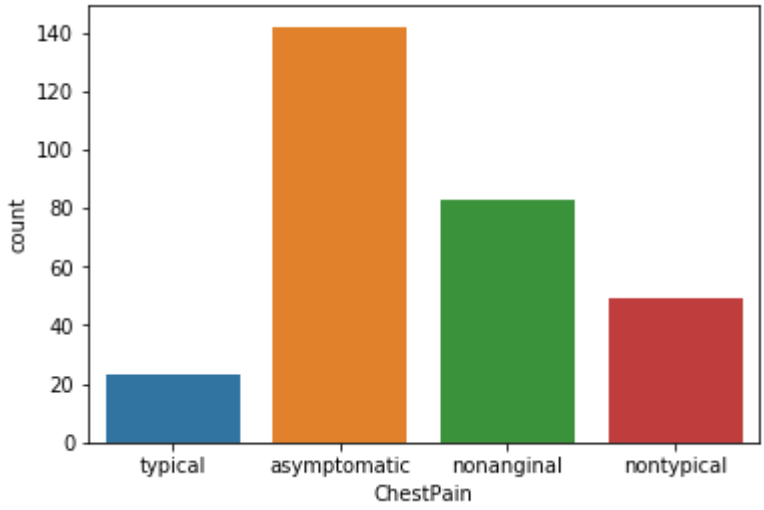


In [20]:

```
sns.countplot(heart['ChestPain'])
```

Out[20]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1b6cf3a39e8>
```

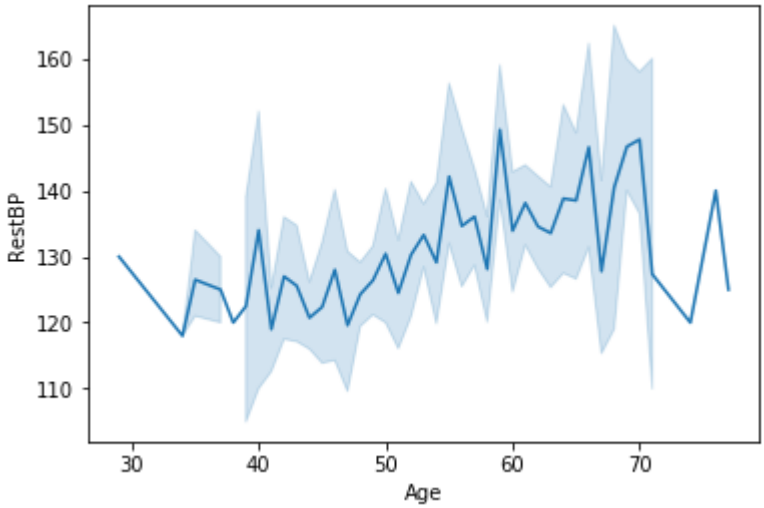


In [21]:

```
sns.lineplot(heart.Age,heart.RestBP)
```

Out[21]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1b6cefad748>
```

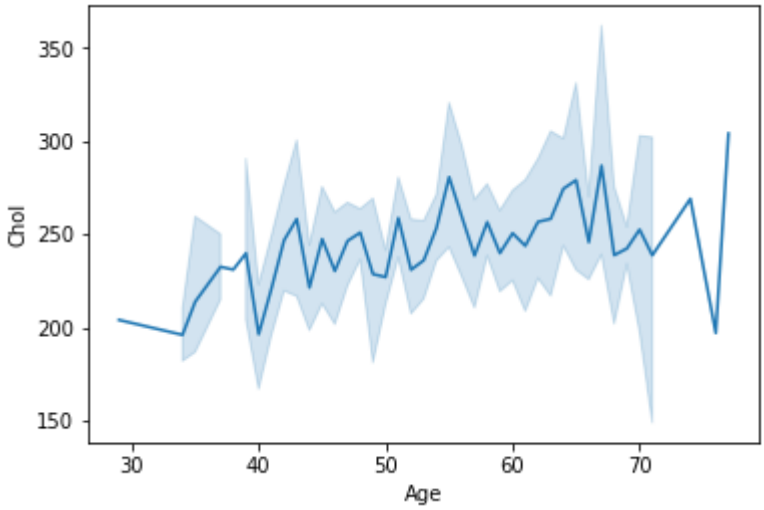


In [22]:

```
sns.lineplot(heart.Age,heart.Chol)
```

Out[22]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1b6cf025080>
```



In [23]:

```
from sklearn.preprocessing import LabelEncoder

lbc = LabelEncoder()

heart['ChestPain'] = lbc.fit_transform(heart['ChestPain'])
heart['Thal'] = lbc.fit_transform(heart['Thal'])
heart['AHD'] = lbc.fit_transform(heart['AHD'])

heart.head()
```

Out[23]:

	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca
1	63	1	3	145	233	1	2	150	0	2.3	3	0.0
2	67	1	0	160	286	0	2	108	1	1.5	2	3.0
3	67	1	0	120	229	0	2	129	1	2.6	2	2.0
4	37	1	1	130	250	0	0	187	0	3.5	3	0.0
5	41	0	2	130	204	0	2	172	0	1.4	1	0.0

In [24]:

```
X = heart.drop('AHD',axis = 1)
Y = heart['AHD']
```

In [25]:

```
from sklearn.model_selection import train_test_split

X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size = 0.3, random_state = 42
)
```

KNN algorithm

In [26]:

```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()

knn.fit(X_train,Y_train)

Y_pred = knn.predict(X_test)
```

In [27]:

```
from sklearn.metrics import accuracy_score,confusion_matrix

confusion_matrix(Y_test,Y_pred)
```

Out[27]:

```
array([[37, 12],
       [16, 25]], dtype=int64)
```

In [28]:

```
accuracy_score(Y_test,Y_pred)
```

Out[28]:

```
0.6888888888888889
```