In [1]:
```
import pandas as pd
import numpy as np
import math
import operator
```

In [2]:
```
data = pd.read_csv('Desktop\mdd.csv')
print(data.head())
```

```
                                Area Mode of shopping  \
0              Urban (Metropolitan )  Online shopping
1  Semi- Urban (Smaller City or Town)     Local stores
2              Urban (Metropolitan )    Super markets
3              Urban (Metropolitan )     Local stores
4              Urban (Metropolitan )    Super markets

                                        Want
0                   Got most items except a few
1                   Got most items except a few
2  Got only a few items as store was stocked out
3                              Got every easily
4                   Got most items except a few
```

In [3]:
```
X = data.iloc[:, :-1].values
y = data.iloc[:, 2].values
```

In [8]:
```
from sklearn.preprocessing import LabelEncoder
labelEncoder_area =  LabelEncoder()
X[:,1] = labelEncoder_area.fit_transform(X[:,1])
print(X)
```

```
[[3 1]
 [2 0]
 [3 2]
 ...
 [3 1]
 [2 2]
 [2 2]]
```

In [9]:
```
labelEncoder_want =  LabelEncoder()
y = labelEncoder_want.fit_transform(y)
print(y)
```

```
[1 1 2 0 1 0 0 2 1 0 3 1 1 1 1 1 2 1 1 0 0 0 1 2 3 1 2 1 1 0 0 2 2 1 0 1 0
 1 0 1 1 1 1 1 2 1 1 1 1 0 1 0 0 0 0 0 1 1 2 0 1 1 0 1 0 0 2 0 3 0 1 1 2 1
 1 0 0 0 3 1 1 1 3 0 1 0 1 0 0 0 0 1 2 1 0 1 1 1 0 0 1 1 1 2 0 1 1 1 1 3 2
 1 1 0 2 3 1 1 2 0 1 0 0 0 0 1 1 1 2 0 1 1 3 1 1 1 0 1 2 1 1 2 1 2 1 0 0 1
 1 0 1 0 1 1 0 0 1 1 0 1 1 2 0 0 1 2 3 1 0 1 1 2 0 2 2 1 1 1 0 2 1 1 3 1 0
 1 0 0 0 2 0 0 0 1 0 1 0 0 1 0 1 1 1 0 1 1 1 1 1 0 2 0 0 2 3 0 2 1 1 1 0 1
 1 0 1 0 0 1 1 1 1 0 2 1 1 1 1 0 0 3 0 1 1 1 1 1 1 2 0 0 1 0 1 1 1 3 0 1 1
 1 2 2 1 1 0 0 1 2 1 1 1 0 0 1 0 1 2 1 0 2 0 1 0 0 1 2 1 1 1 0 1 0 3 1 3 2
 2 2 2 2 0 2 1 0 0 1 1 1 1 1 2 0 0 3 0 0 0 1 1 0 0 1 1 1 0 0 1 0 1 3 2 1 2
 0 1 2 0 1 0 0 2 1 0 1 1 1 3 0 1 1 1 0 1 1 1 0 0 1 0 1 1 0 0 0 0 1 0 1 1 0
 1 2 1 1 3 2 1 1 1 1 2 1 3 1 1 1 0 0 0 0 2 3 1 1 0 1 3 3 1 1 0 1 2 1 1 0 2
 1 0 1 2 0 0 1 3 2 0 1 0 1 1 1 1 2 0 3 0 0 1 0 2 0 1 0 1 1 1 1 1 0 0 1 1 0
 1 0 1 1 2 0 1 1 1 3 1 1 0 0 3 0 2 0 1 2 1 1 1 2 2 1 0 0 0 1 1 1 2 0 2 1 0
 1 3 1 1 1 2 2 0 1 1 1 1 1 1 1 1 1 1 1 1]
```

In [10]:
```
import numpy as np
X = np.vstack(X[:, :]).astype(np.float)
```

In [11]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

In [12]:
```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

In [27]:
```
from sklearn.neighbors import KNeighborsClassifier
# metric = minkowski and p=2 is Euclidean Distance
# metric = minkowski and p=1 is Manhattan Distance
classifier = KNeighborsClassifier(n_neighbors=5, metric="minkowski",p=2)
classifier.fit(X_train, y_train)
```

Out[27]:
```
KNeighborsClassifier()
```

In [28]:
```

y_pred = classifier.predict(X_test)
```

In [ ]:
```

```

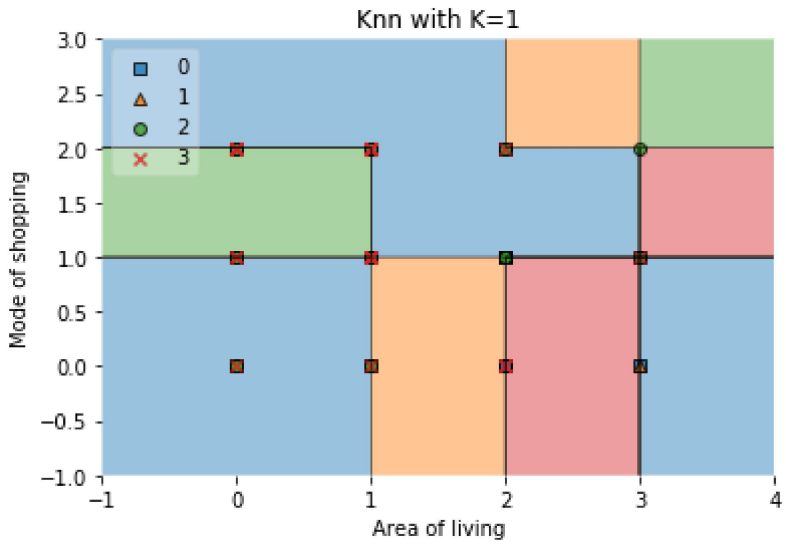
In [29]:
```
from sklearn import metrics
cm = metrics.confusion_matrix(y_test, y_pred)
print(cm)
rep = metrics.classification_report(y_test, y_pred)
print(rep)
```

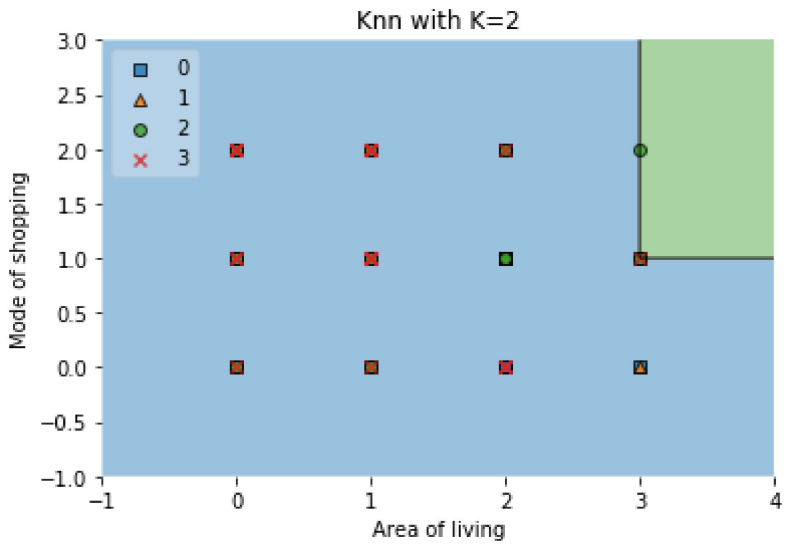
```
[[ 7 26  0  0]
 [15 49  0  0]
 [ 3 16  0  0]
 [ 4  6  0  0]]
              precision    recall  f1-score   support

           0       0.24      0.21      0.23        33
           1       0.51      0.77      0.61        64
           2       0.00      0.00      0.00        19
           3       0.00      0.00      0.00        10

    accuracy                           0.44       126
   macro avg       0.19      0.24      0.21       126
weighted avg       0.32      0.44      0.37       126



C:\Users\Sabarri Krishnan\AppData\Roaming\Python\Python37\site-packages\sklearn\metrics\_classification.py:122
1: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predic
ted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

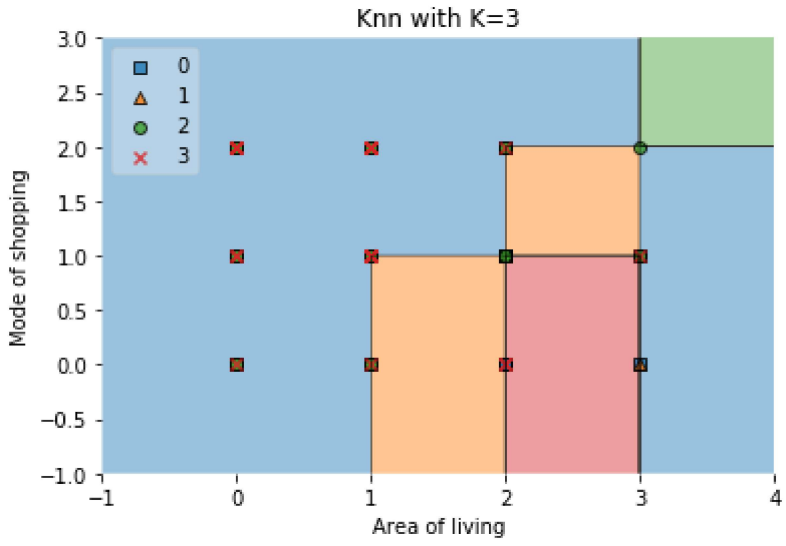
In [16]:
```
import matplotlib.pyplot as plt
from sklearn import datasets,neighbors
from mlxtend.plotting import plot_decision_regions
```

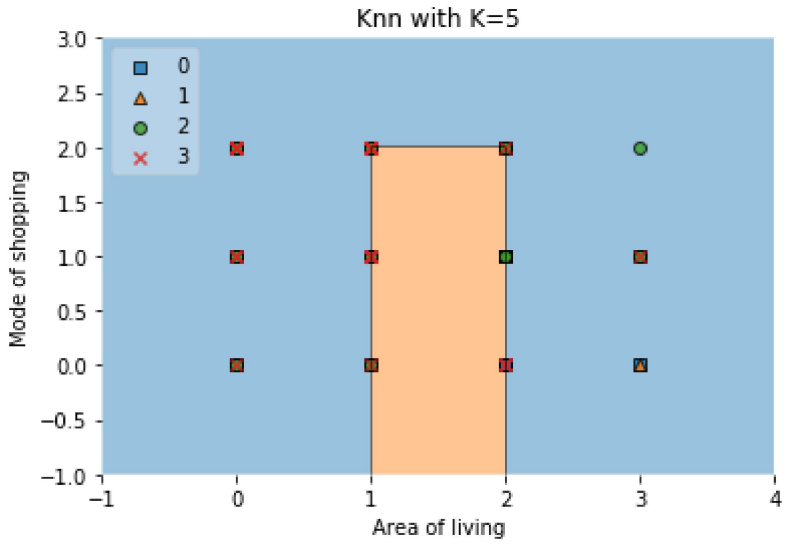
In [17]:
```
def knn_comparison(data, k):
    x = data[['Area of living','Mode of shopping']].values
    y = data['Availability'].astype(int).values
    clf = neighbors.KNeighborsClassifier(n_neighbors=k)
    clf.fit(x, y)
# Plotting decision region
    plot_decision_regions(x, y, clf=clf, legend=2)
# Adding axes annotations
    plt.xlabel('Area of living')
    plt.ylabel('Mode of shopping')
    plt.title('Knn with K='+ str(k))
    plt.show()
```

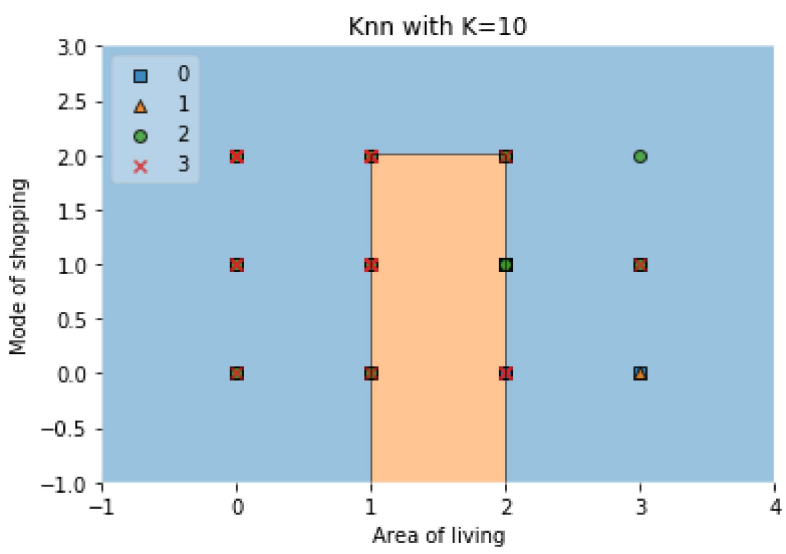
In [42]:
```
data1=pd.read_csv('Desktop\mdd-Copy1.csv')
for i in [1,2,3,5,10,80]:
    knn_comparison(data1, i)
```



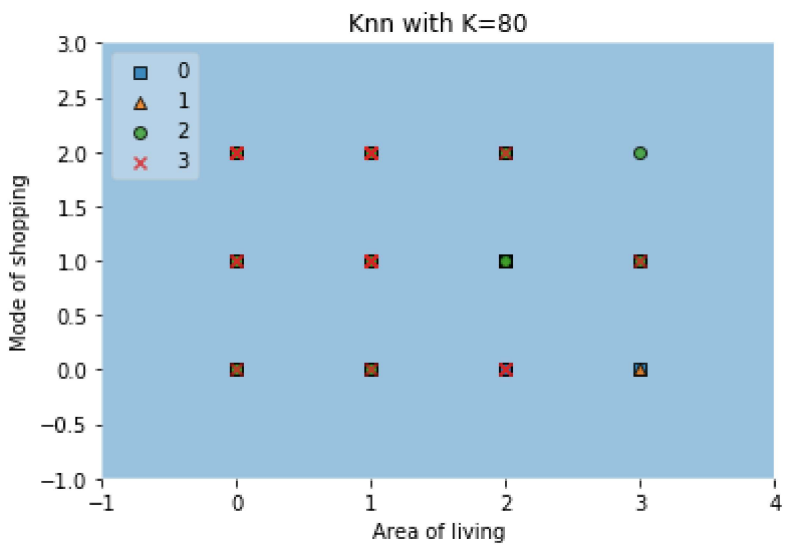








```
C:\Users\Sabarri Krishnan\Anaconda3\lib\site-packages\matplotlib\contour.py:1243: UserWarning: No contour leve
ls were found within the data range.
  warnings.warn("No contour levels were found"
```



In [ ]:

In [ ]: