In [7]:

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

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler,MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from mlxtend.plotting import plot_decision_regions

import warnings
warnings.filterwarnings("ignore")
```

In [ ]:

```

```

1. U shape

In [8]:

```
column_names=["a","b",'c']
df = pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\1.us
```

In [9]:

```
df
```

Out[9]:

```
            a          b    c
  0   0.031595   0.986988  0.0
  1   2.115098  -0.046244  1.0
  2   0.882490  -0.075756  0.0
  3  -0.055144  -0.037332  1.0
  4   0.829545  -0.539321  1.0
...        ...        ...  ...
 95   1.699453   0.587720  1.0
 96   0.218623  -0.652521  1.0
 97   0.952914  -0.419766  1.0
 98  -1.318500   0.423112  0.0
 99  -1.296818   0.184147  0.0

100 rows × 3 columns
```

In [10]:

```
df.head()
```

Out[10]:

	a	b	c
0	0.031595	0.986988	0.0
1	2.115098	-0.046244	1.0
2	0.882490	-0.075756	0.0
3	-0.055144	-0.037332	1.0
4	0.829545	-0.539321	1.0

In [11]:

```
df.describe()
```

Out[11]:

	a	b	c
count	100.000000	100.000000	100.000000
mean	0.500420	0.228701	0.500000
std	0.891044	0.592885	0.502519
min	-1.318500	-1.035702	0.000000
25%	-0.140330	-0.203260	0.000000
50%	0.470678	0.188660	0.500000
75%	1.112008	0.658448	1.000000
max	2.181372	1.571899	1.000000

In [12]:

```
df["c"]=df["c"].astype("int")
```

In [13]:

```
fv=df.iloc[:,:2]
cv=df.iloc[:,-1]
```

spliting into x_train,y_train and x_test,y_test

In [14]:

```
x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,stratify=cv,rand
```

splitinf into x_trainf,y_trainf and x_crossvalidation and y_crossvalidation

In [16]:

```
x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

Column normalization

In [17]:

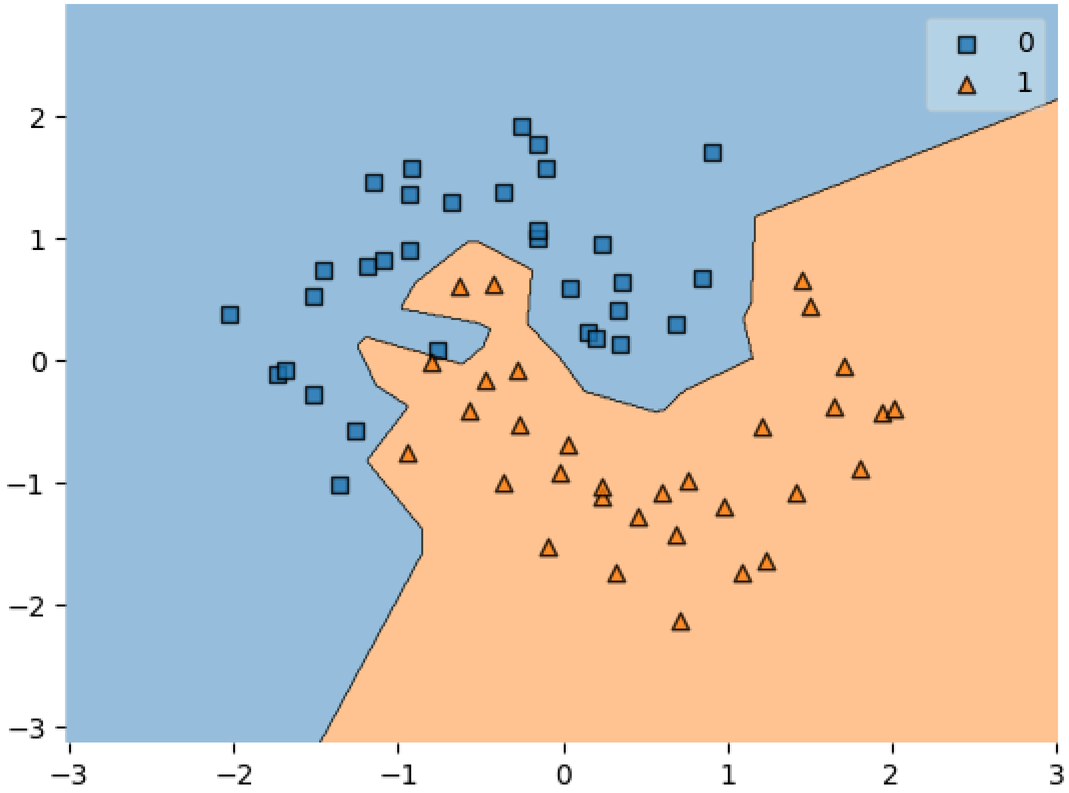
```
std=StandardScaler()
px_trainf=std.fit_transform(x_trainf)
px_test=std.transform(x_test)
px_cv=std.transform(x_cv)
```

multiple decision regions for the diffren k values for ushape

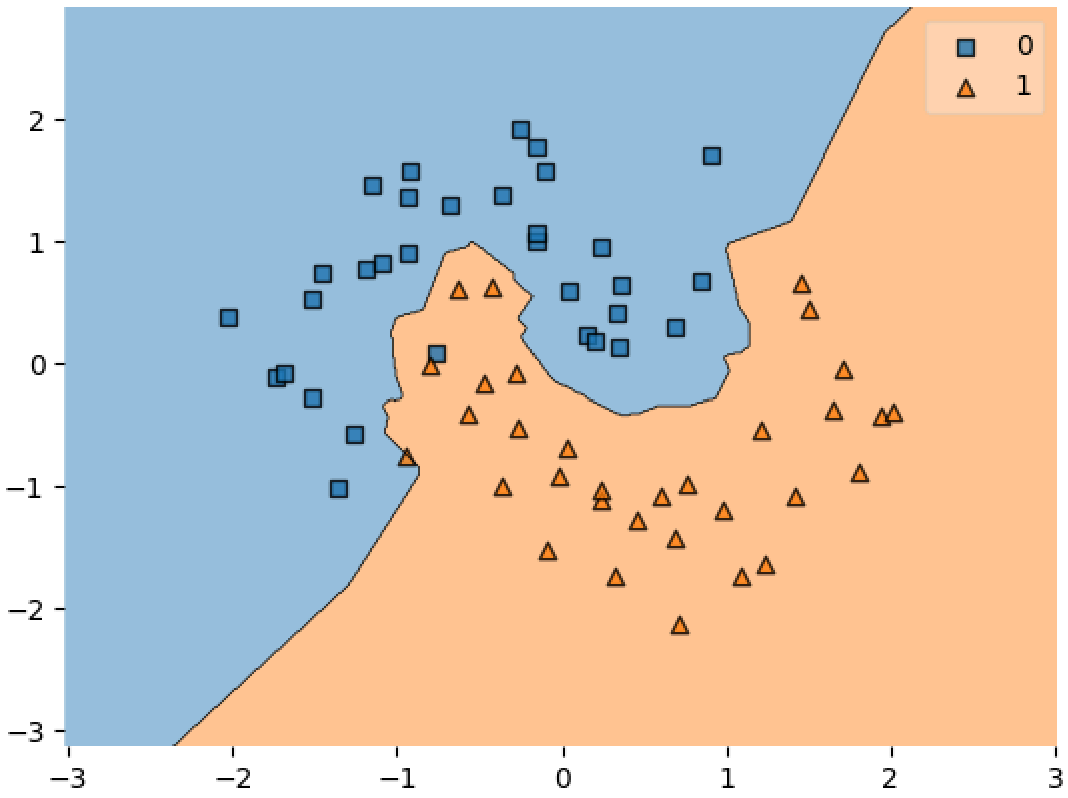
In [23]:

```
for i in range(1,10,2):
    knn=KNeighborsClassifier(n_neighbors=i)
    model=knn.fit(px_trainf,y_trainf)
    predicted=model.predict(px_cv)
    print(f"k is equal to = {i} , accuracy score",accuracy_score(y_cv,predicted))
    plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
    plt.show()
```

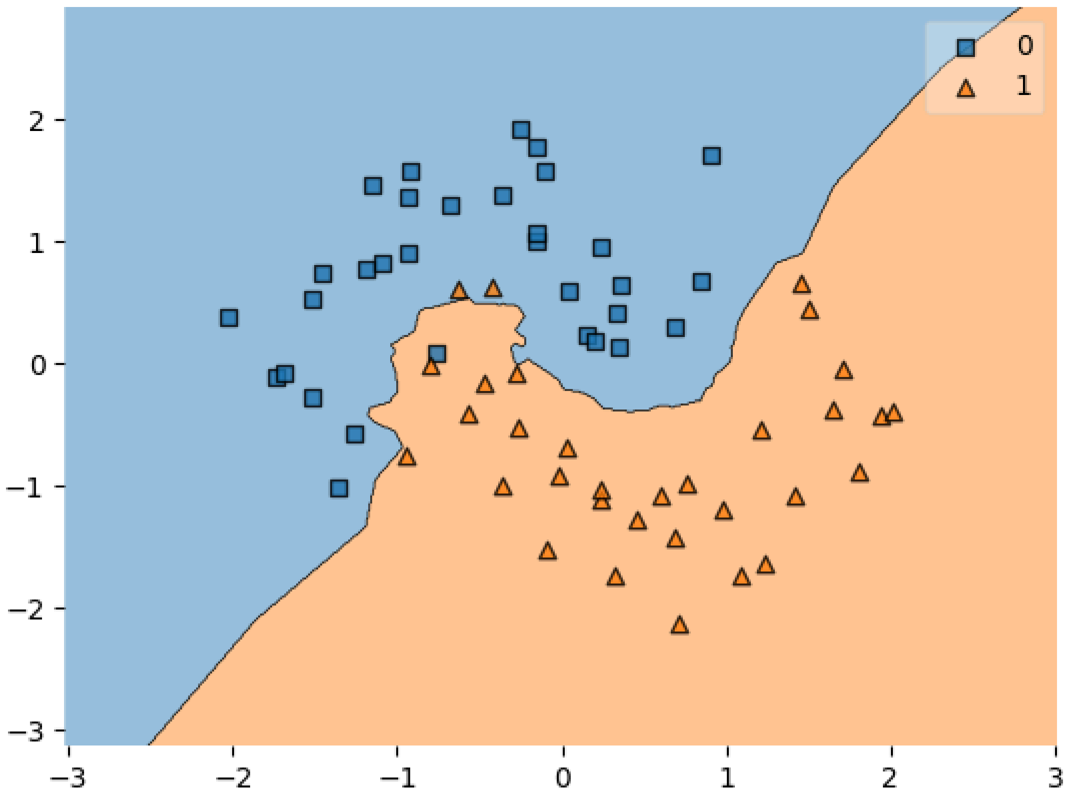
```
k is equal to = 1 , accuracy score 0.875
```



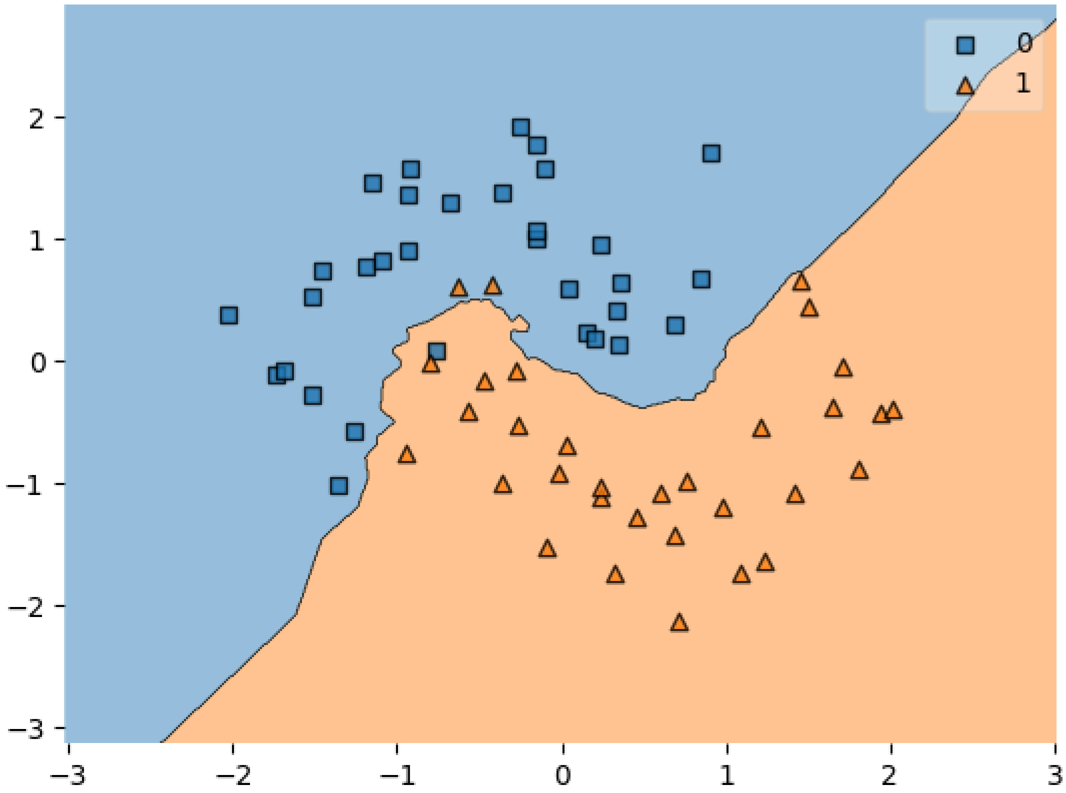
```
k is equal to = 3 , accuracy score 0.875
```



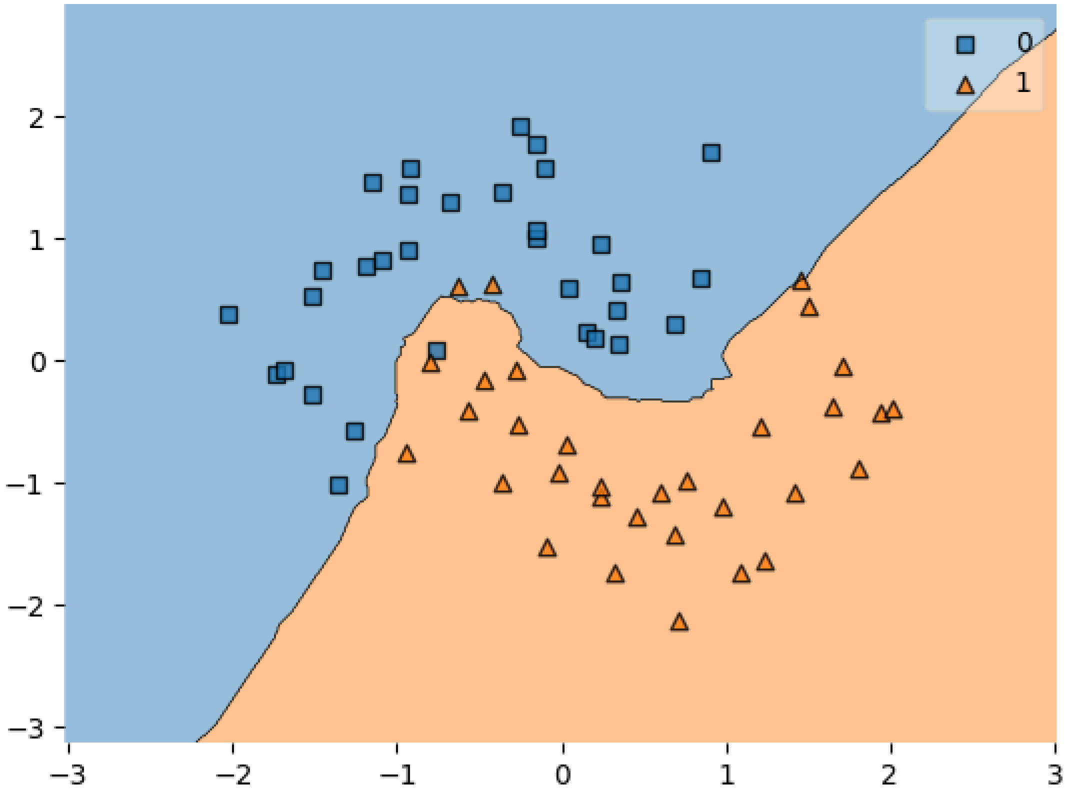
```
k is equal to = 5 , accuracy score 0.8125
```



```
k is equal to = 7 , accuracy score 0.875
```



```
k is equal to = 9 , accuracy score 0.875
```



from the above plot at where k=3 I'am able to make the correct decision

2. concentriccir1

In [32]:
```
columns=["a","b","c"]
df=pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\2.conc
```

In [33]:
```
df
```

Out[33]:

	a	b	c
0	-0.382891	-0.090840	1.0
1	-0.020962	-0.477874	1.0
2	-0.396116	-1.289427	0.0
3	-0.618130	-0.063837	1.0
4	0.703478	-0.187038	1.0
...	...	...	...
95	-0.474862	-0.224981	1.0
96	0.126272	0.869784	0.0
97	-0.647365	-0.363424	1.0
98	0.474405	1.011016	0.0
99	-0.385658	-0.810312	0.0

100 rows × 3 columns

In [34]:
```
fv=df.iloc[:,:2]
cv=df.iloc[:,-1]
```

In [35]:
```
cv=cv.astype(int)
```

In [36]:
```
x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,random_state=2)
```

In [37]:
```
x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

In [38]:
```
std=StandardScaler()
px_trainf=std.fit_transform(x_trainf)
px_test=std.transform(x_test)
px_cv=std.transform(x_cv)
```

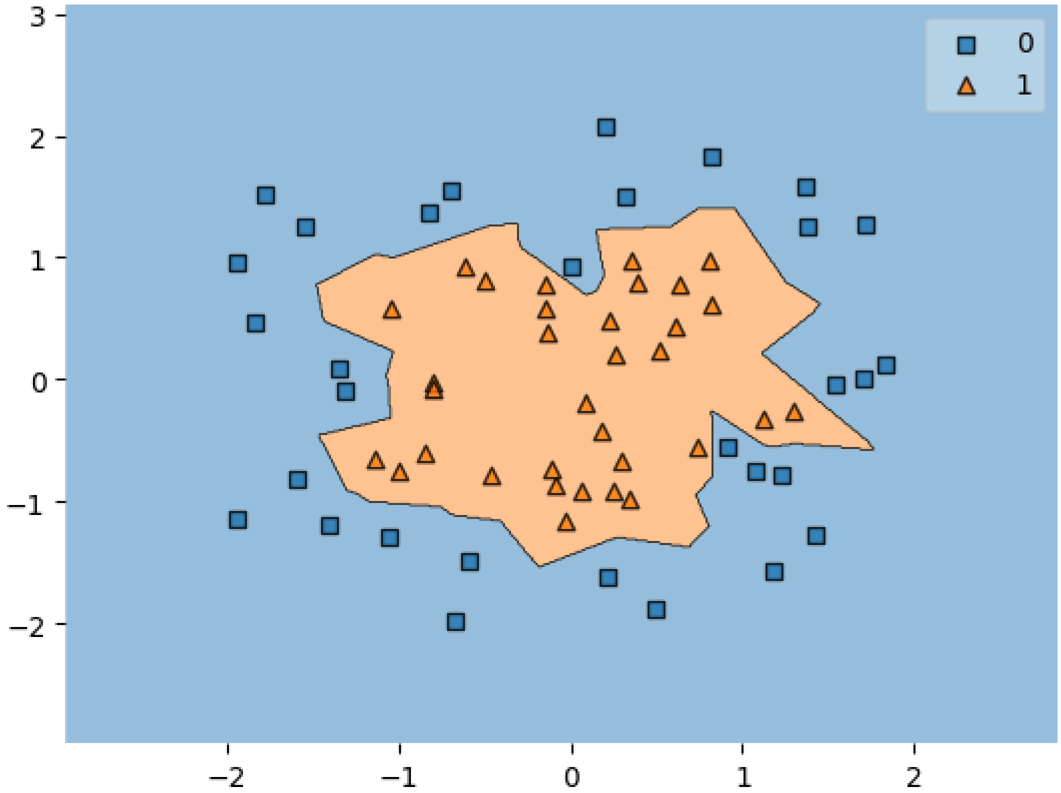
In [39]:
```
knn=KNeighborsClassifier(n_neighbors=1)
model=knn.fit(px_trainf,y_trainf)
predicted=model.predict(px_cv)
```

In [40]:
```
accuracy_score(y_cv,predicted)
```

Out[40]:
```
0.8125
```

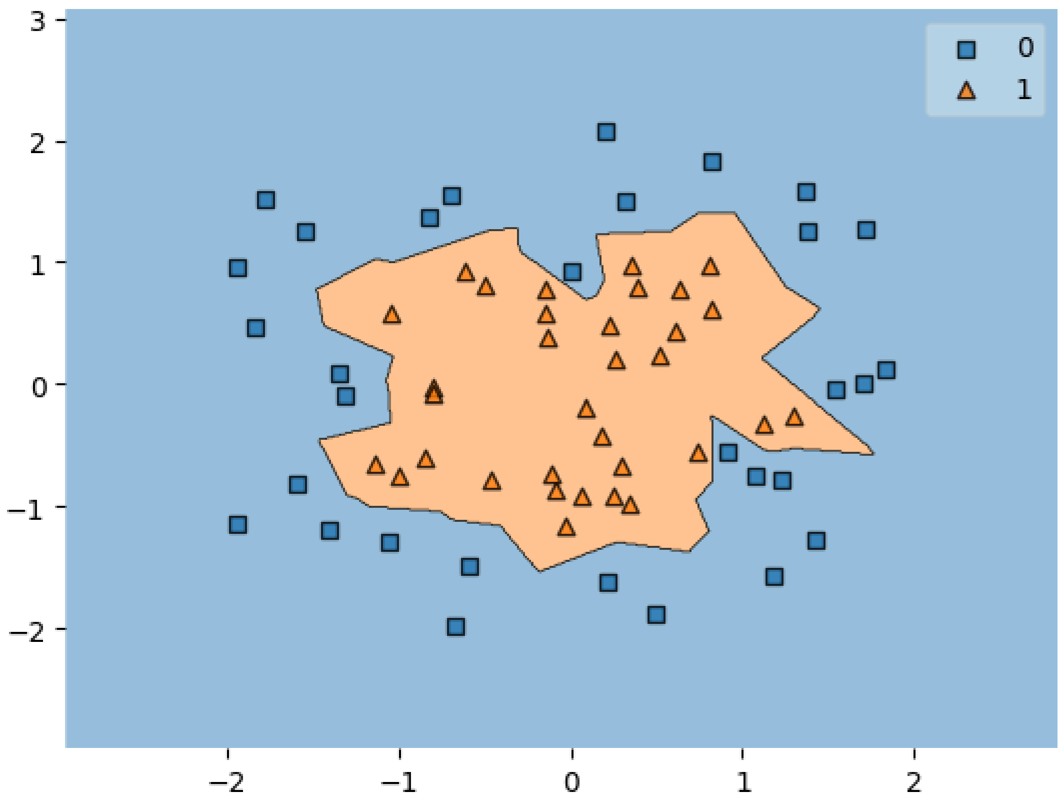
In [41]:
```
plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
```

Out[41]:
```
<Axes: >
```

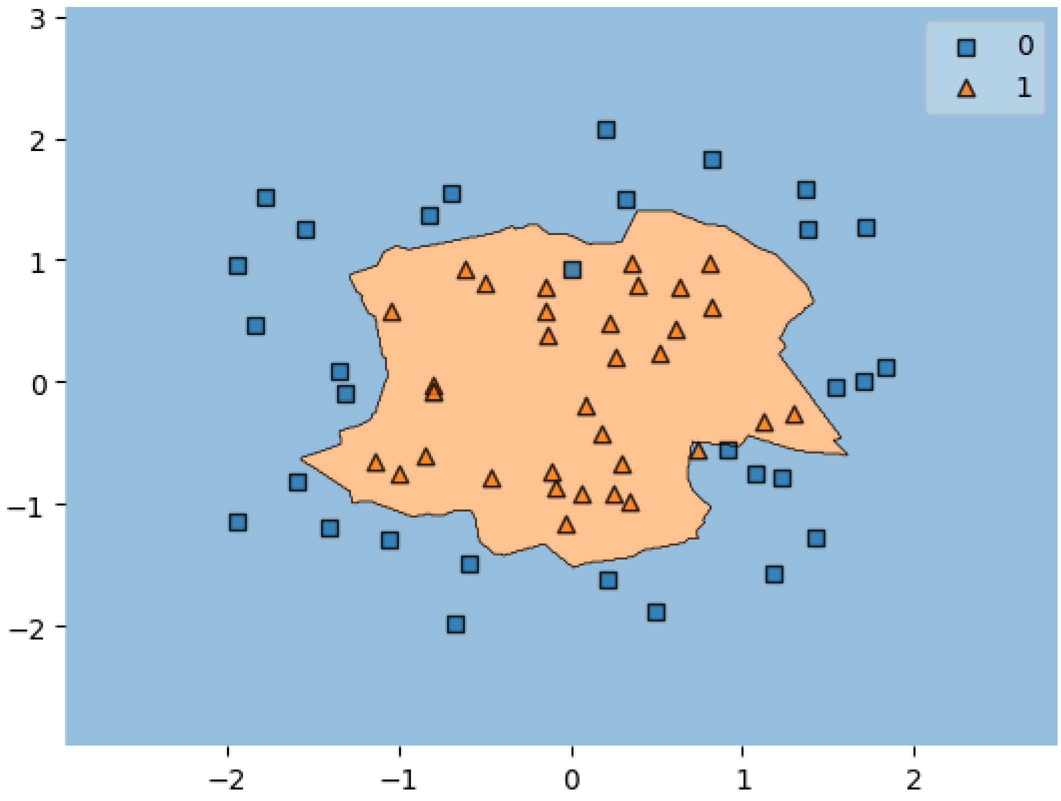


In [42]:
```
for i in range(1,8,2):
    knn=KNeighborsClassifier(n_neighbors=i)
    model=knn.fit(px_trainf,y_trainf)
    predicted=model.predict(px_cv)
    print(f"k is equal to = {i} , accuracy score",accuracy_score(y_cv,predicted))
    plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
    plt.show()
```

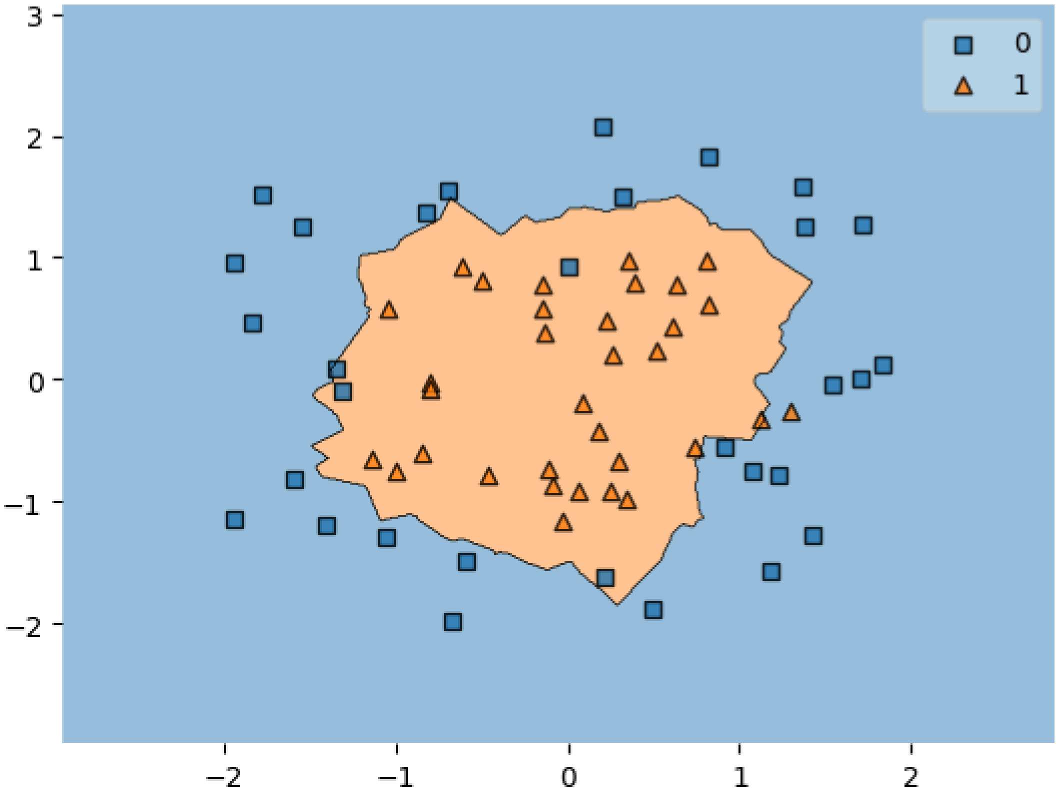
```
k is equal to = 1 , accuracy score 0.8125
```



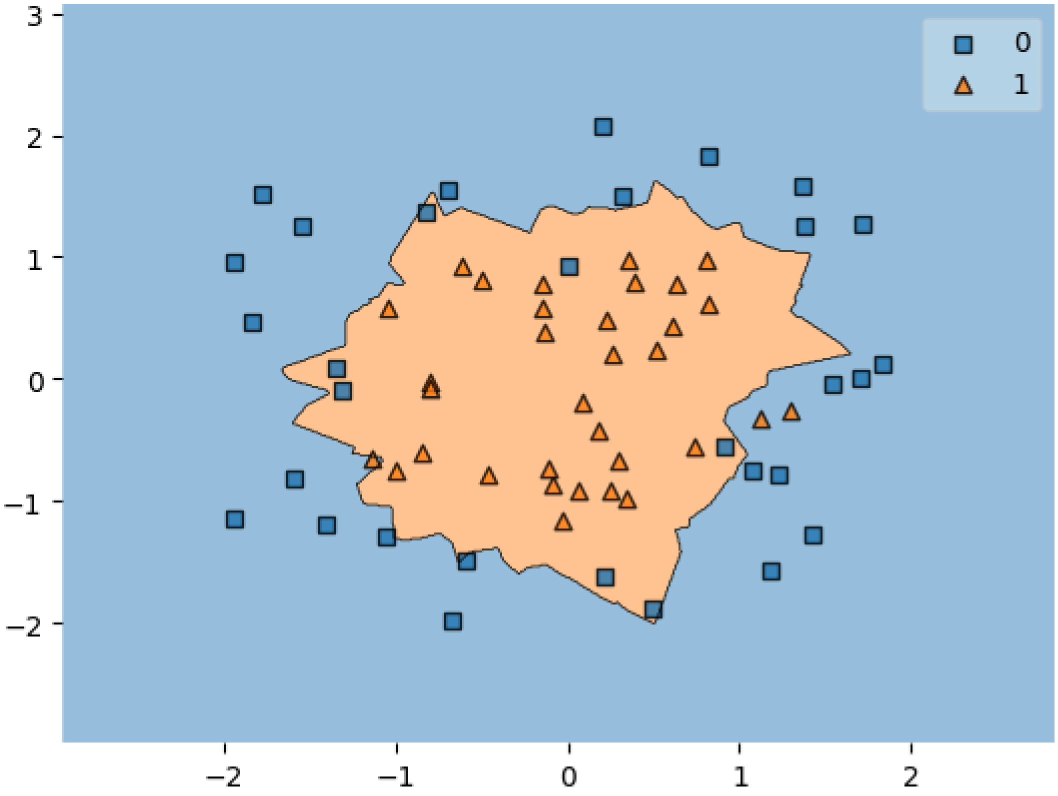
```
k is equal to = 3 , accuracy score 0.875
```



```
k is equal to = 5 , accuracy score 0.9375
```



```
k is equal to = 7 , accuracy score 0.875
```



from the above plot at where k=5 I'am able to make the correct decision

3. concentriccir2

In [48]:

```
df=pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\3.conc
df
```

Out[48]:

	a	b	c
0	0.700335	-0.247068	0.0
1	-3.950019	2.740080	1.0
2	0.150222	-2.157638	1.0
3	-1.672050	-0.941519	1.0
4	2.560483	-1.846577	1.0
...	...	...	...
495	2.177895	2.984489	1.0
496	1.778905	2.869205	1.0
497	0.894180	3.069959	0.0
498	0.849439	3.875435	0.0
499	5.217443	1.400818	0.0

500 rows × 3 columns

In [49]:

```
fv=df.iloc[:,:2]
cv=df.iloc[:,-1]
```

In [50]:

```
cv=cv.astype(int)
```

In [51]:

```
x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,random_state=2)
```

In [52]:

```
x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

In [53]:

```
std=StandardScaler()
px_trainf=std.fit_transform(x_trainf)
px_test=std.transform(x_test)
px_cv=std.transform(x_cv)
```

In [54]:

```
knn=KNeighborsClassifier(n_neighbors=1)
model=knn.fit(px_trainf,y_trainf)
predicted=model.predict(px_cv)
```

In [55]:

```
accuracy_score(y_cv,predicted)
```

Out[55]:

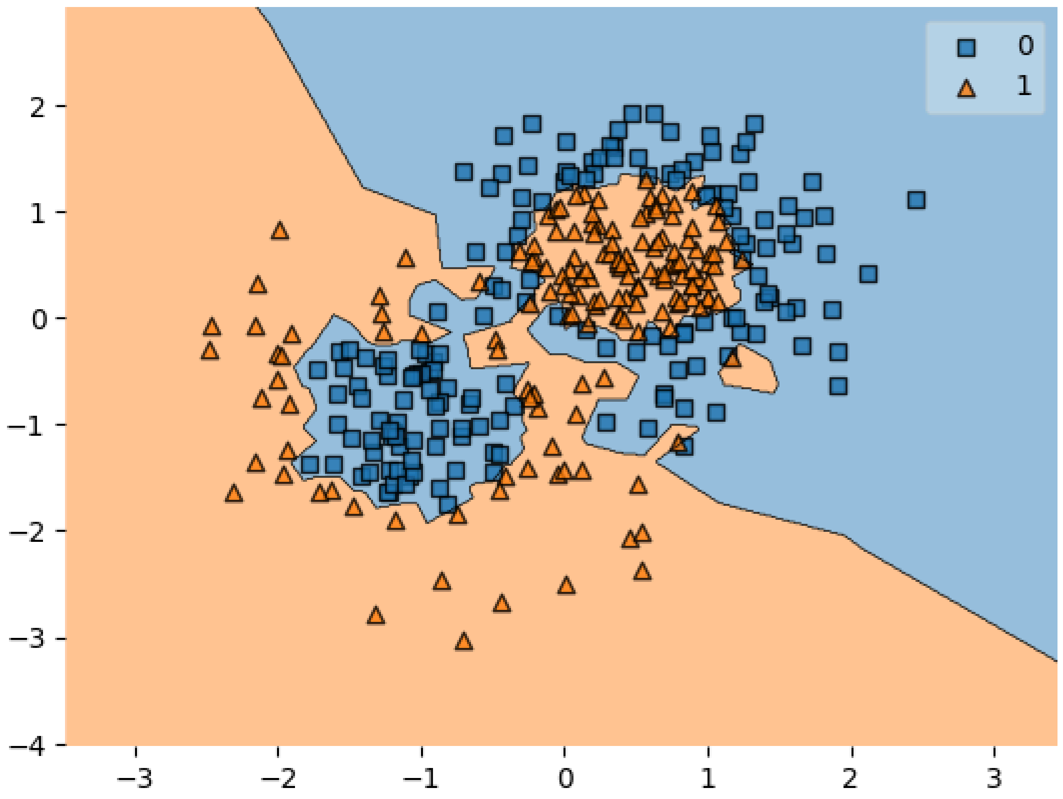
```
0.9125
```

In [56]:

```
plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
```

Out[56]:

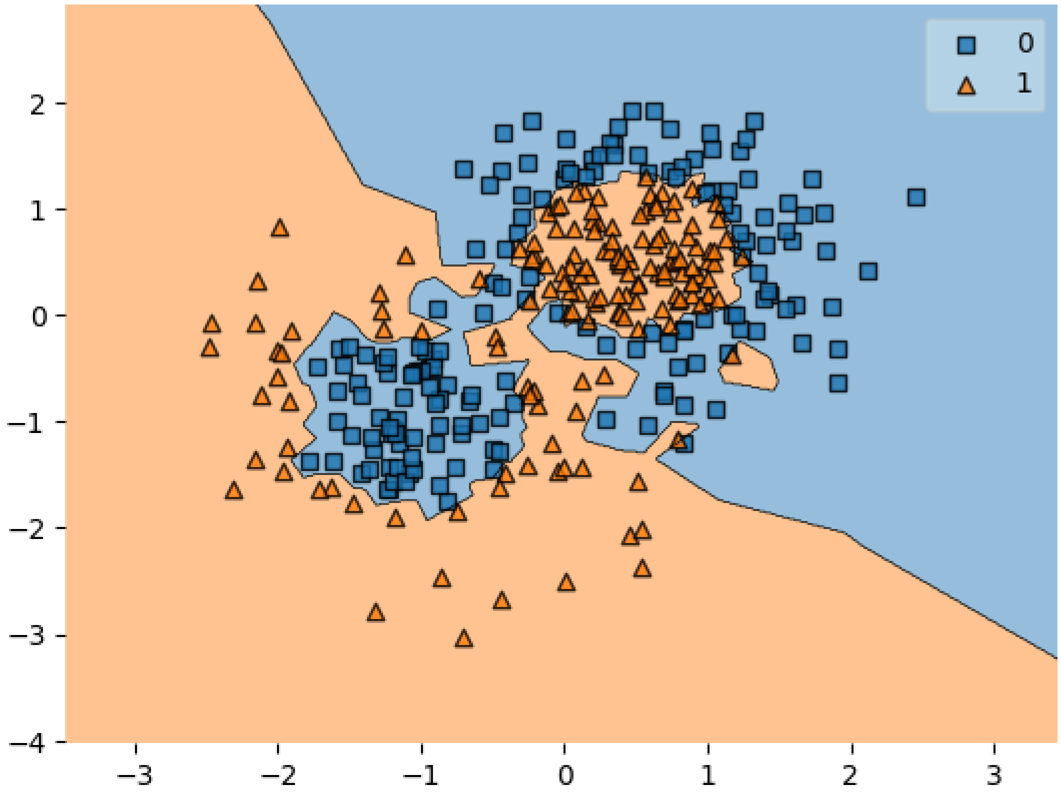
```
<Axes: >
```



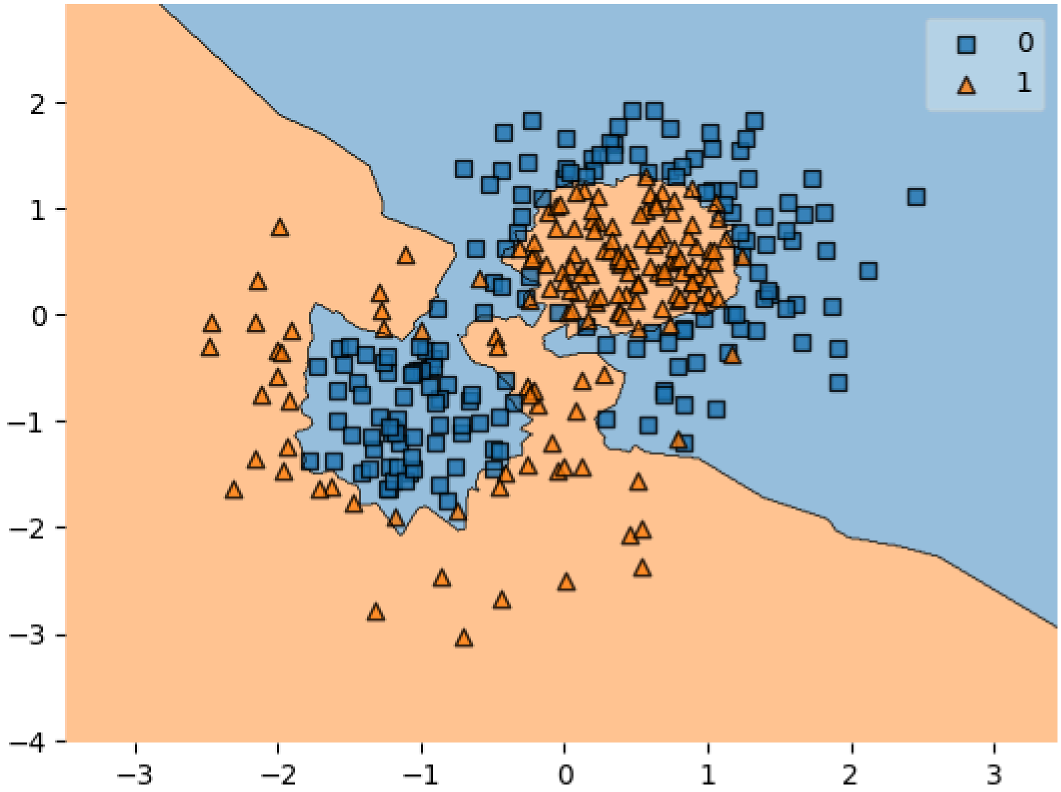
In [57]:

```
for i in range(1,30,2):
    knn=KNeighborsClassifier(n_neighbors=i)
    model=knn.fit(px_trainf,y_trainf)
    predicted=model.predict(px_cv)
    print(f"k is equal to = {i} , accuracy score",accuracy_score(y_cv,predicted))
    plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
    plt.show()
```

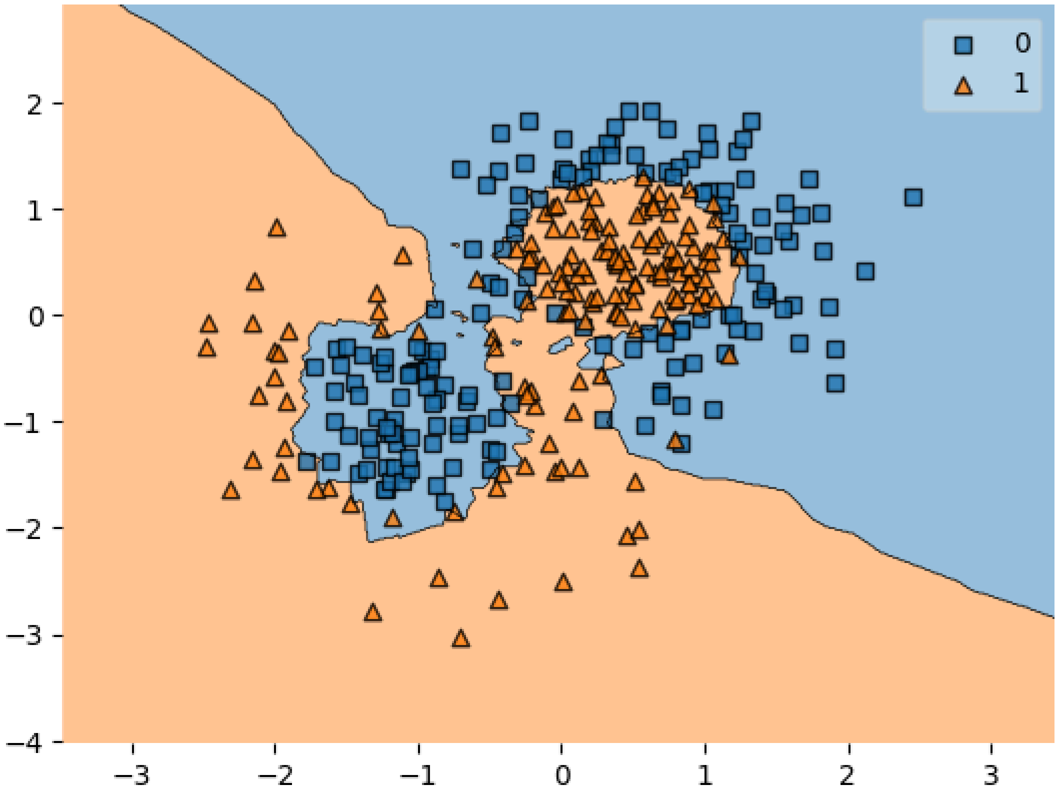
```
k is equal to = 1 , accuracy score 0.9125
```



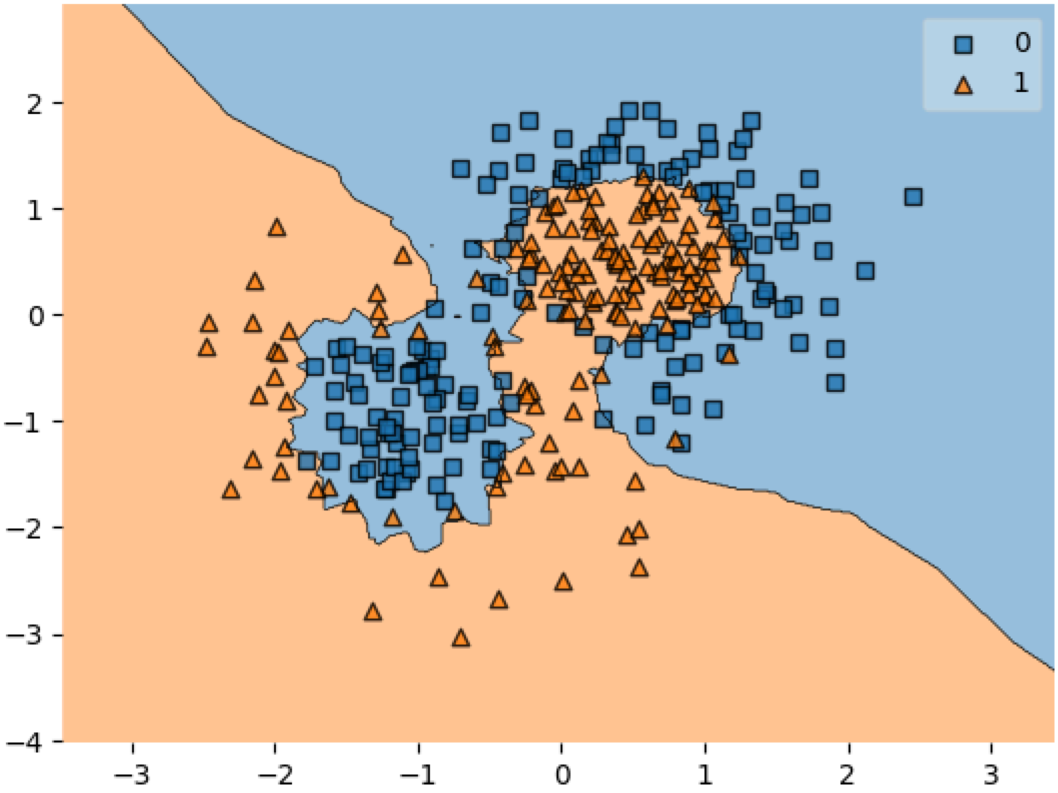
```
k is equal to = 3 , accuracy score 0.875
```



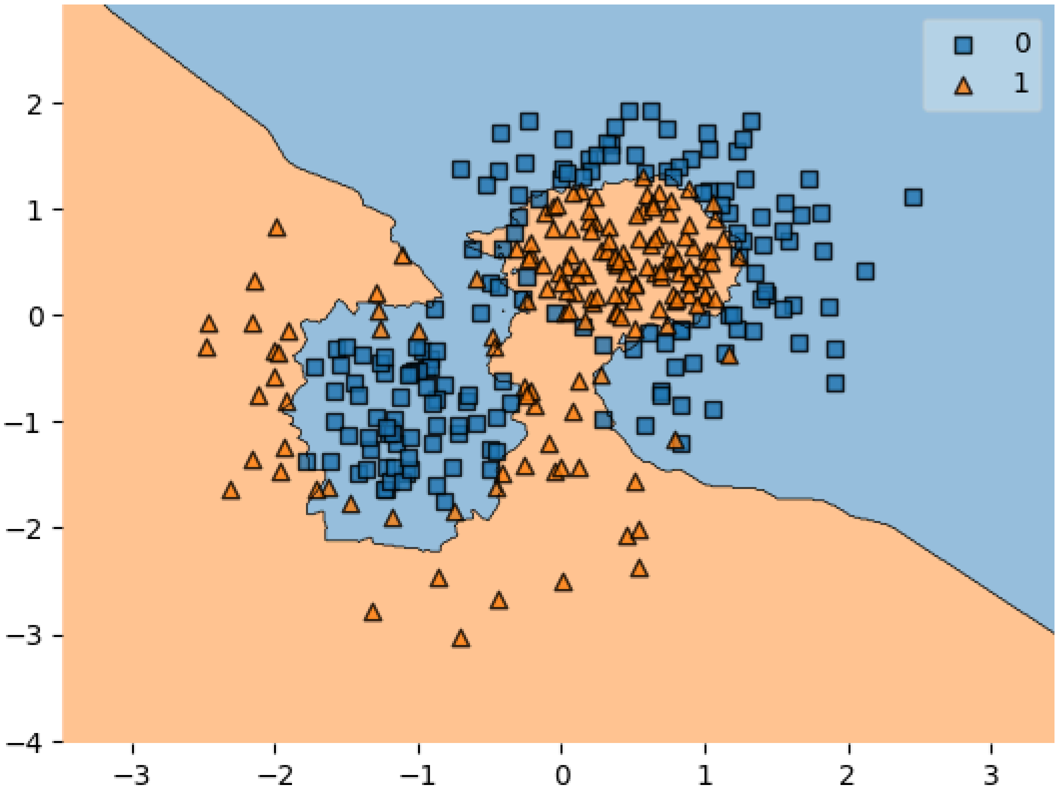
```
k is equal to = 5 , accuracy score 0.85
```



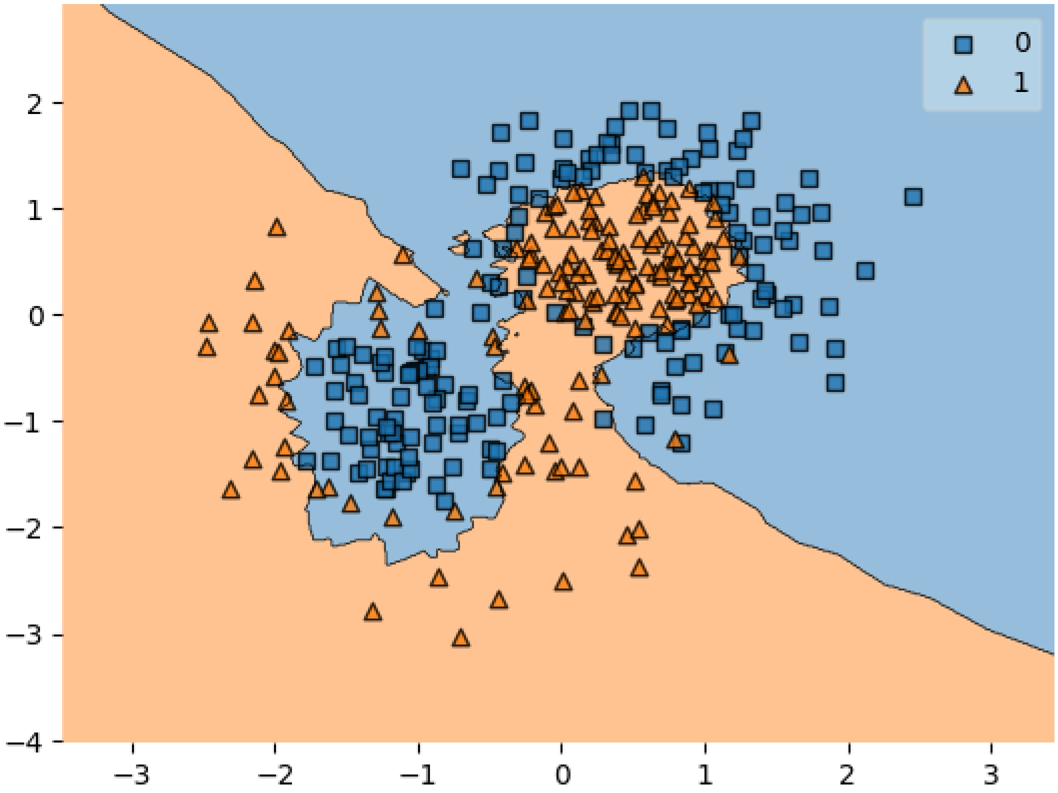
```
k is equal to = 7 , accuracy score 0.8125
```



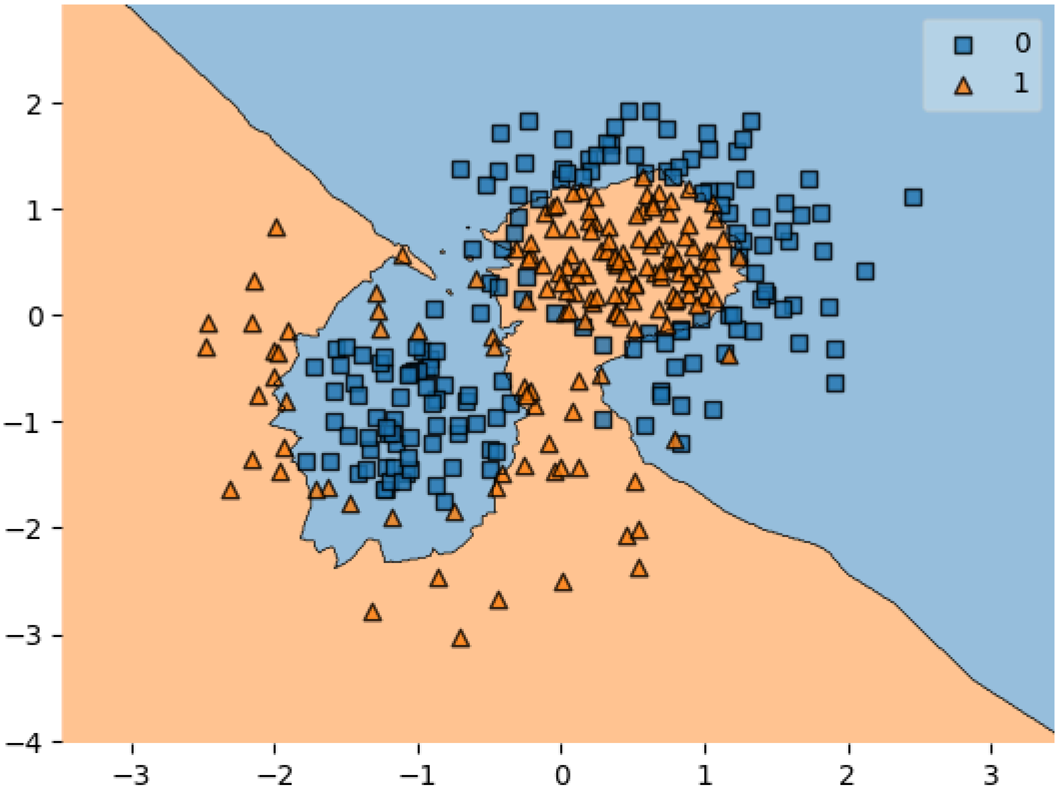
```
k is equal to = 9 , accuracy score 0.825
```



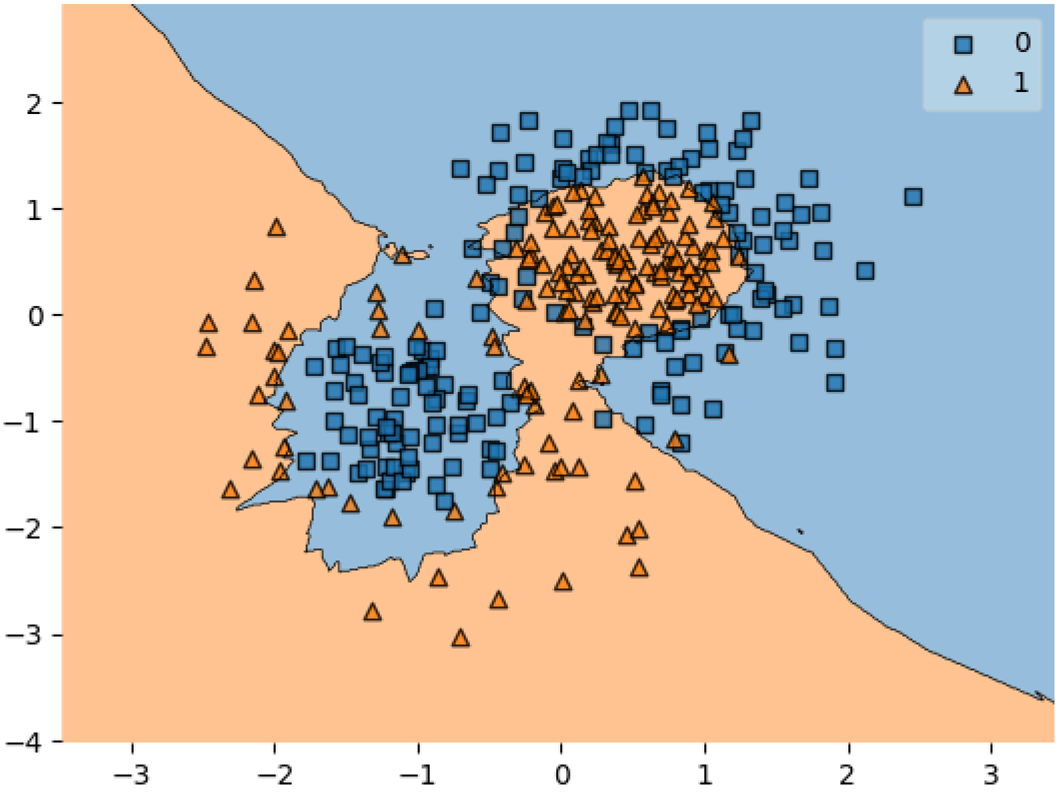
```
k is equal to = 11 , accuracy score 0.85
```



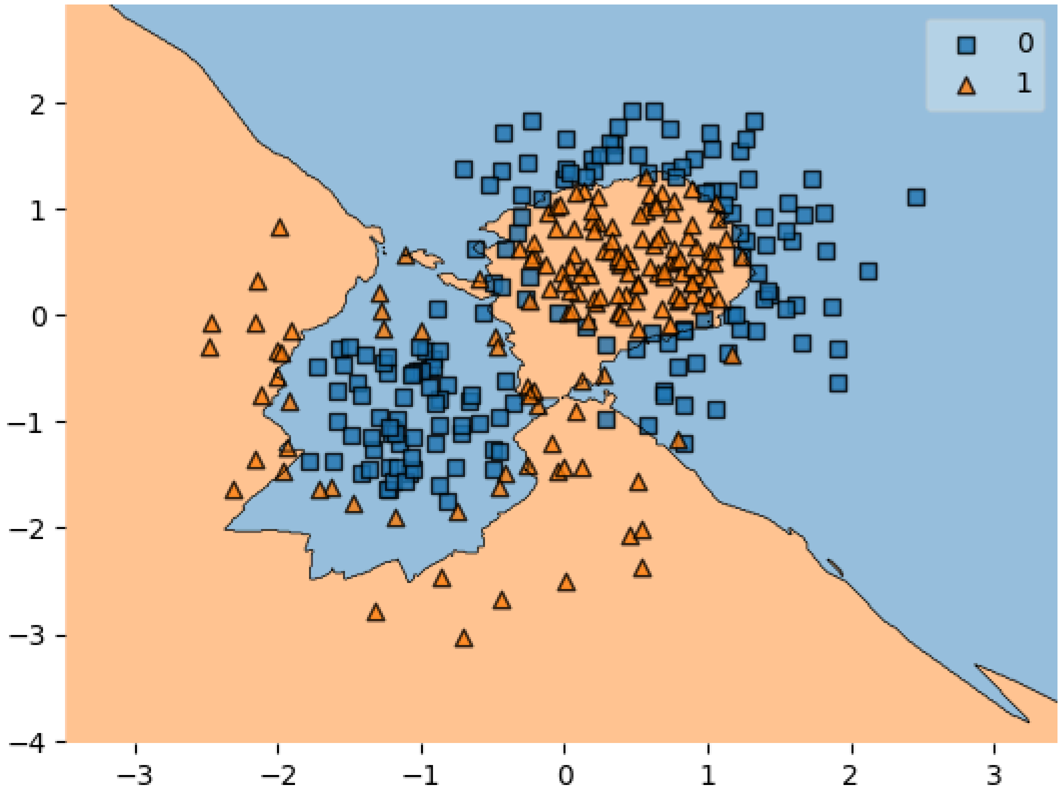
```
k is equal to = 13 , accuracy score 0.8125
```



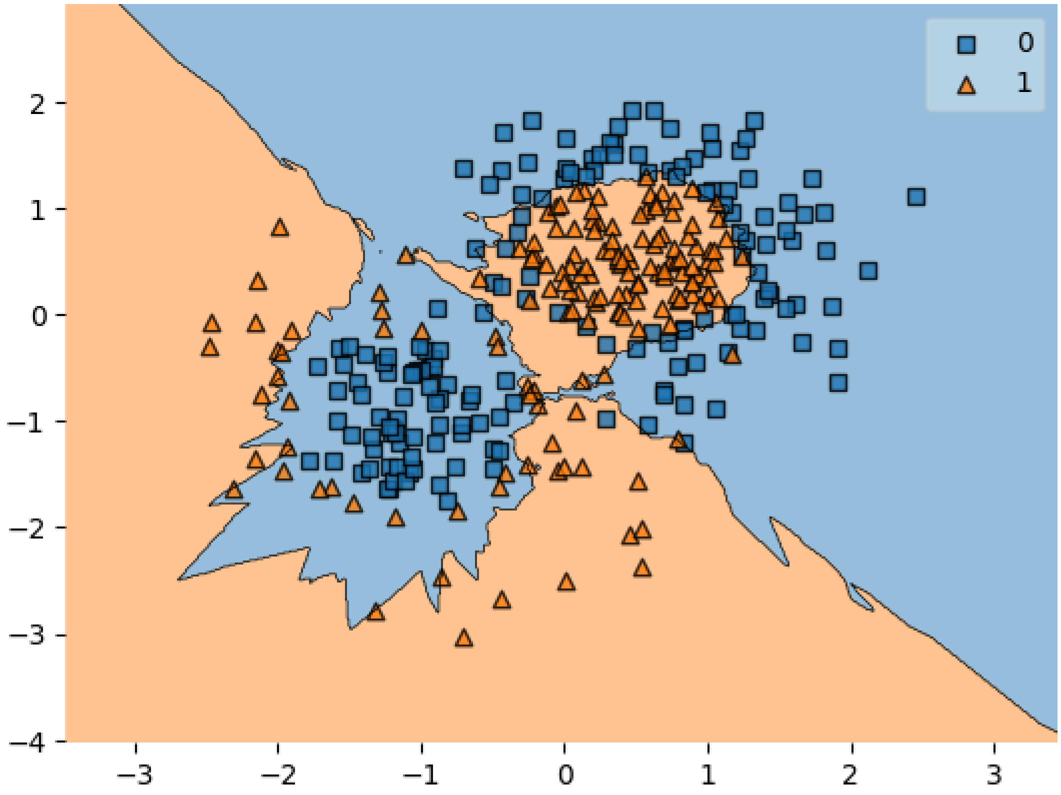
```
k is equal to = 15 , accuracy score 0.8125
```



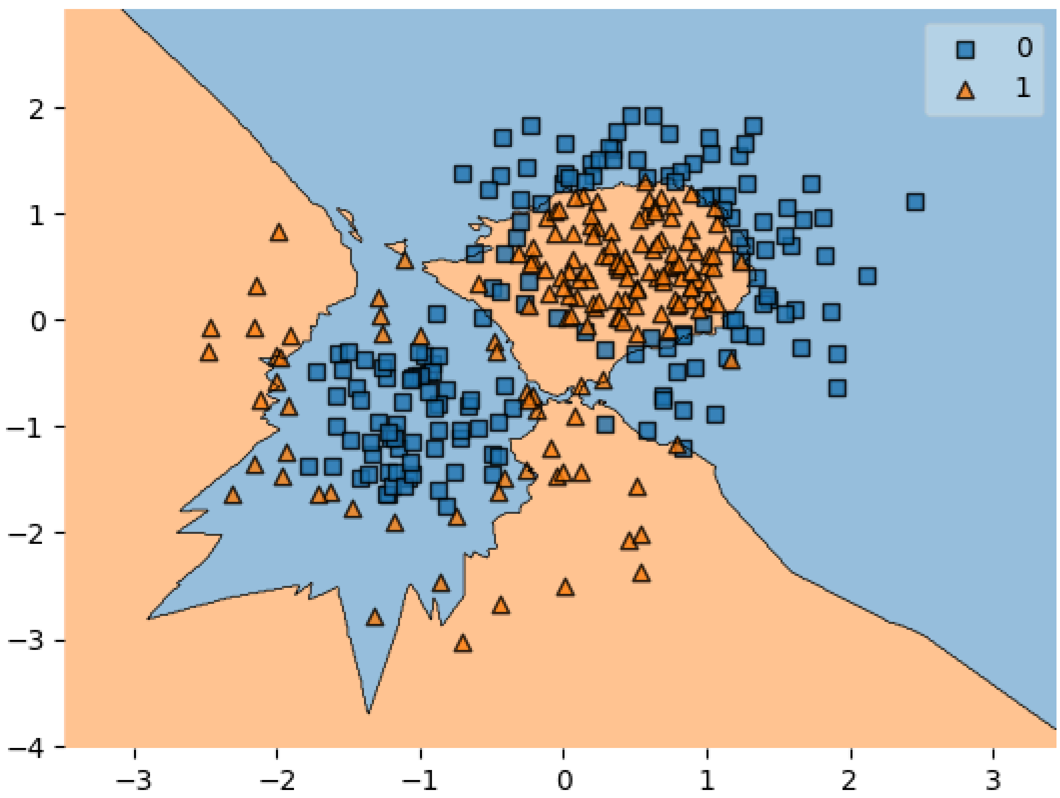
```
k is equal to = 17 , accuracy score 0.7875
```



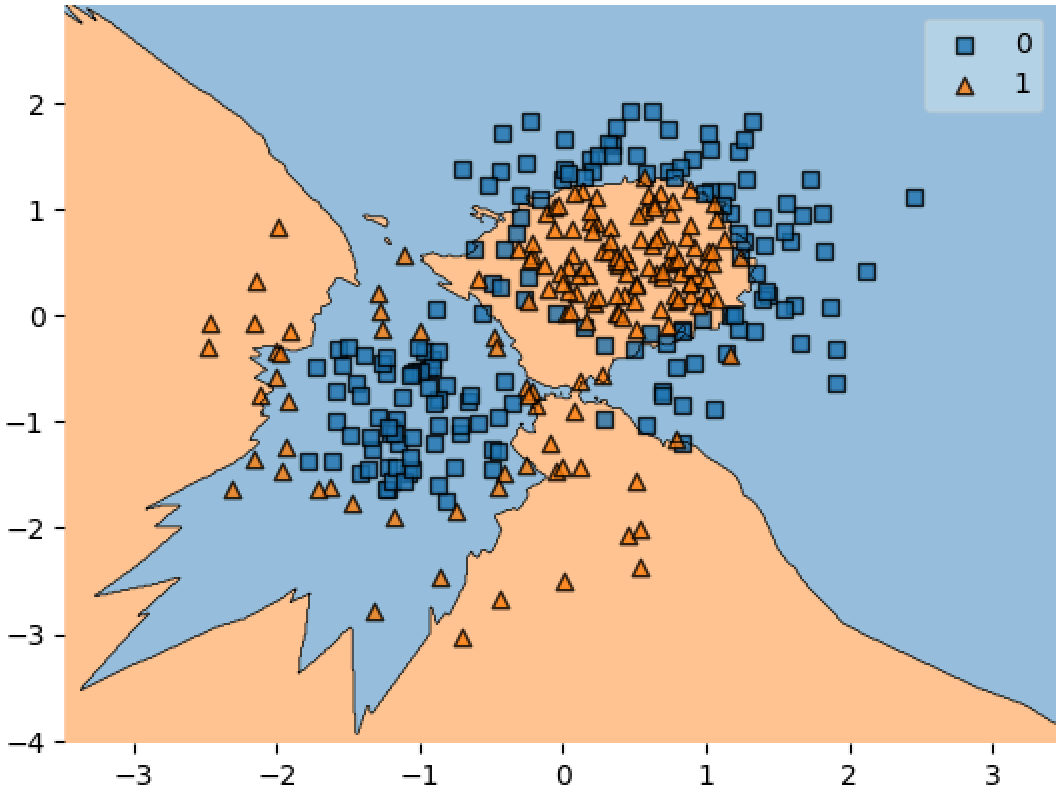
```
k is equal to = 19 , accuracy score 0.775
```



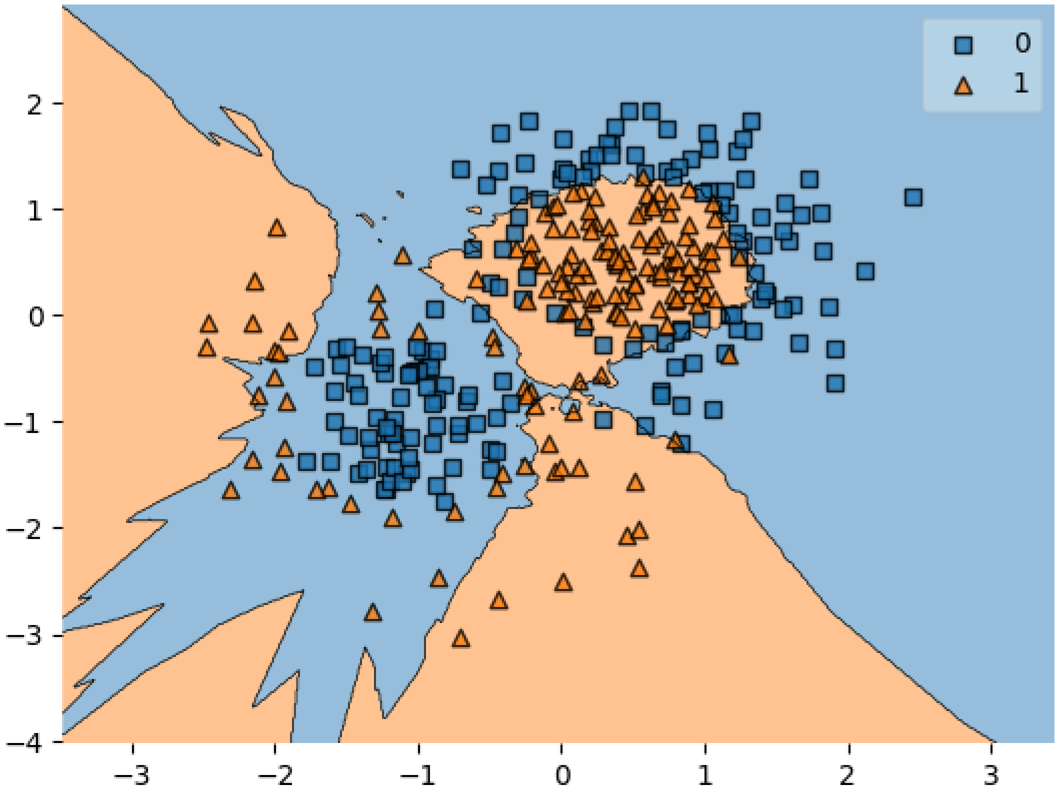
```
k is equal to = 21 , accuracy score 0.7875
```



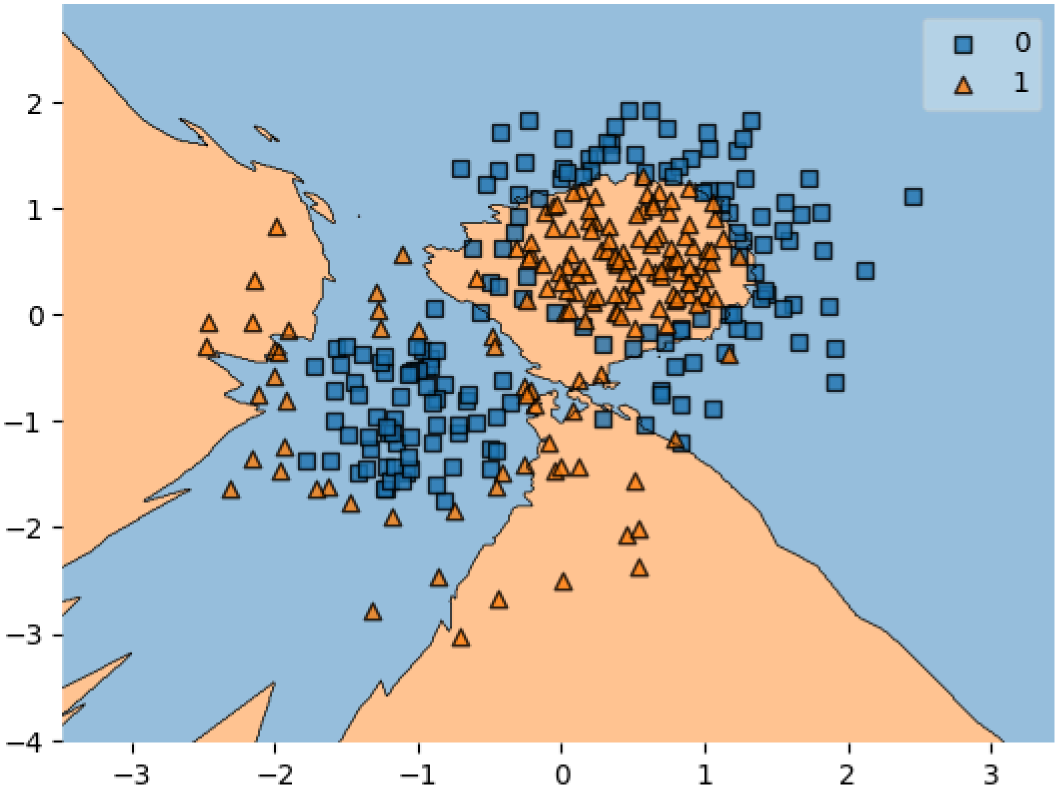
```
k is equal to = 23 , accuracy score 0.7625
```



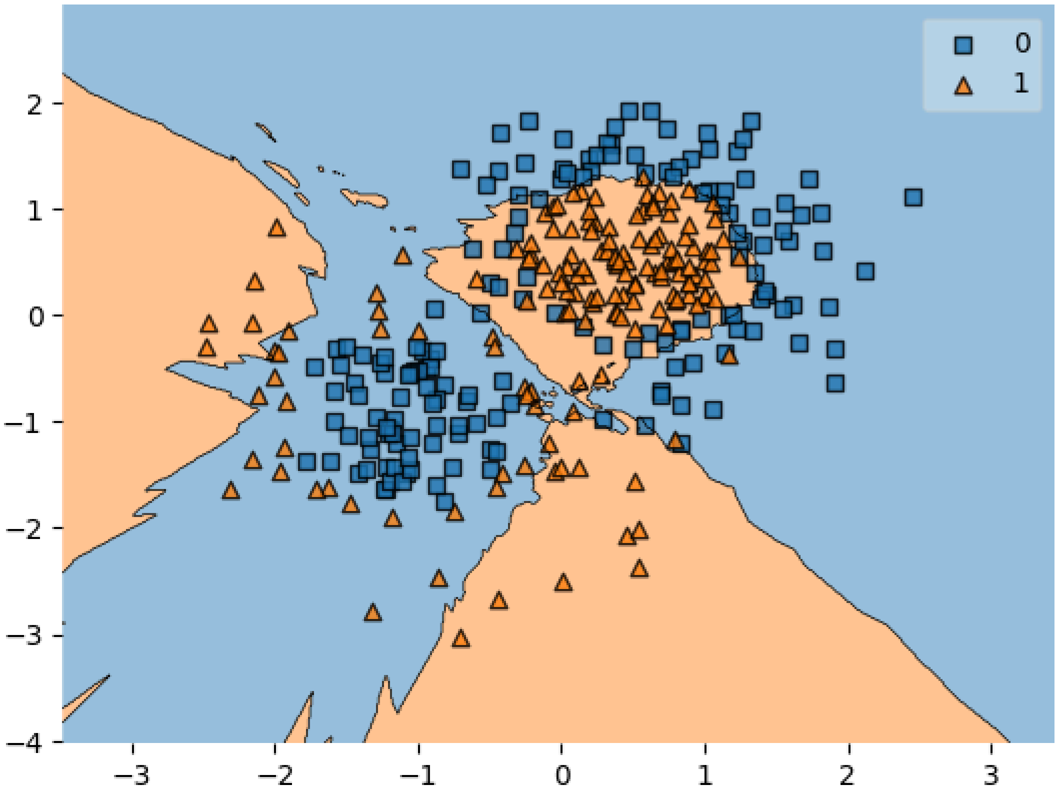
```
k is equal to = 25 , accuracy score 0.7625
```



```
k is equal to = 27 , accuracy score 0.75
```



```
k is equal to = 29 , accuracy score 0.7625
```



from the above plot at where k=9 I'am able to make the correct decision

4.linearsep

In [61]:
```
df=pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\4.line
```

In [62]:
```
df
```

Out[62]:

	a	b	c
0	-0.177497	0.930496	1.0
1	1.977424	1.766155	0.0
2	1.800024	1.700343	0.0
3	-0.770837	2.359163	1.0
4	-0.308009	1.594063	1.0
...	...	...	...
95	2.632382	1.271305	0.0
96	-0.040256	1.782708	1.0
97	-0.787453	1.400357	1.0
98	2.702441	1.587444	0.0
99	1.290969	2.751937	1.0

100 rows × 3 columns

In [63]:
```
fv=df.iloc[:,:2]
cv=df.iloc[:,-1]
```

In [64]:
```
cv=cv.astype(int)
```

In [65]:
```
x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,random_state=2)
```

In [66]:
```
x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

In [67]:
```
std=StandardScaler()
px_trainf=std.fit_transform(x_trainf)
px_test=std.transform(x_test)
px_cv=std.transform(x_cv)
```

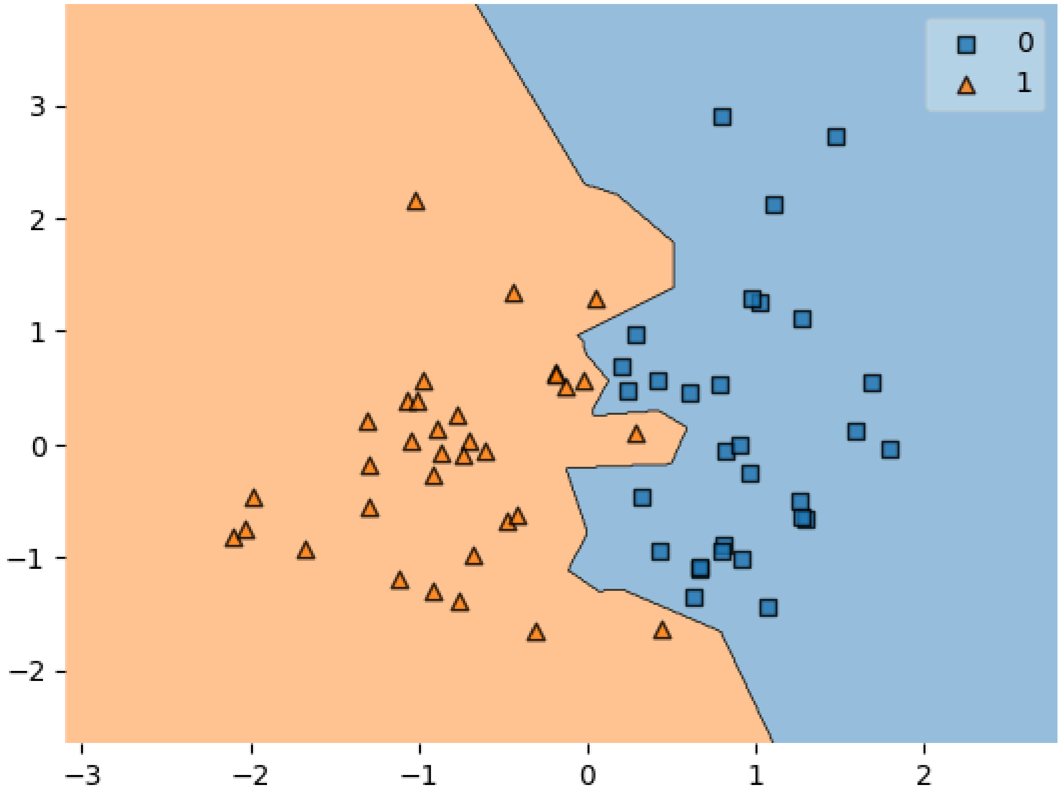
In [68]:
```
knn=KNeighborsClassifier(n_neighbors=1)
model=knn.fit(px_trainf,y_trainf)
predicted=model.predict(px_cv)
```

In [69]:
```
accuracy_score(y_cv,predicted)
```

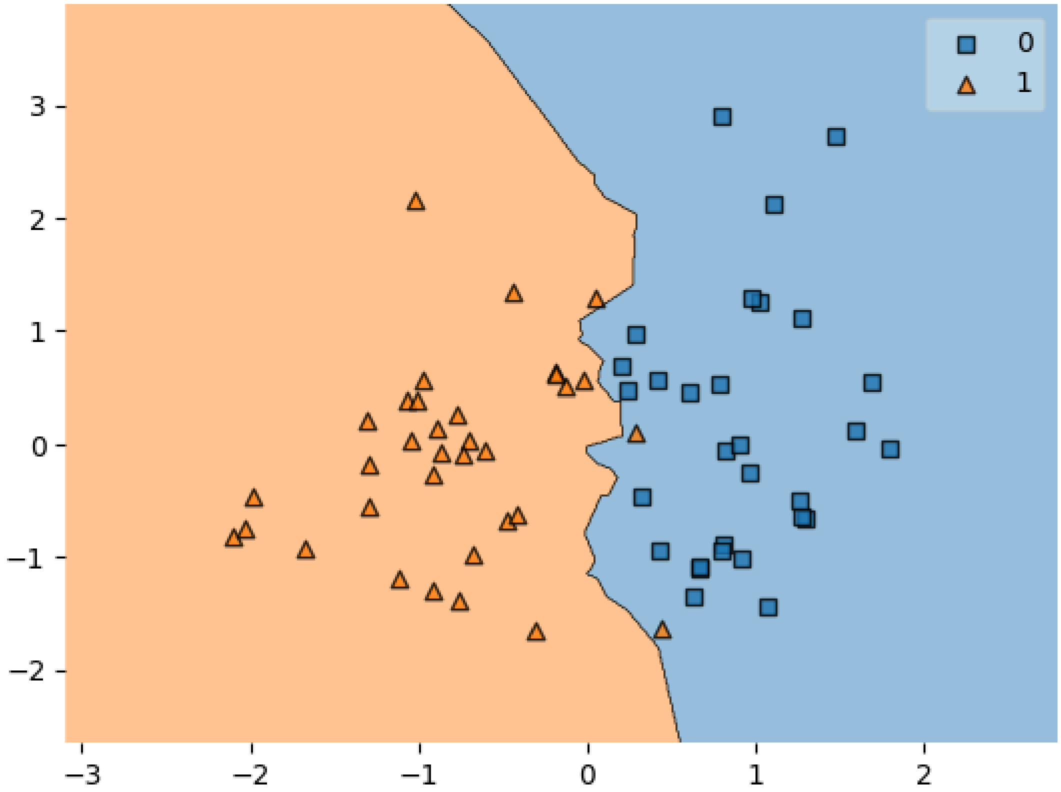
Out[69]:
```
0.9375
```

In [70]:
```
for i in range(1,20,2):
    knn=KNeighborsClassifier(n_neighbors=i)
    model=knn.fit(px_trainf,y_trainf)
    predicted=model.predict(px_cv)
    print(f"k is equal to = {i} , accuracy score",accuracy_score(y_cv,predicted))
    plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
    plt.show()
```

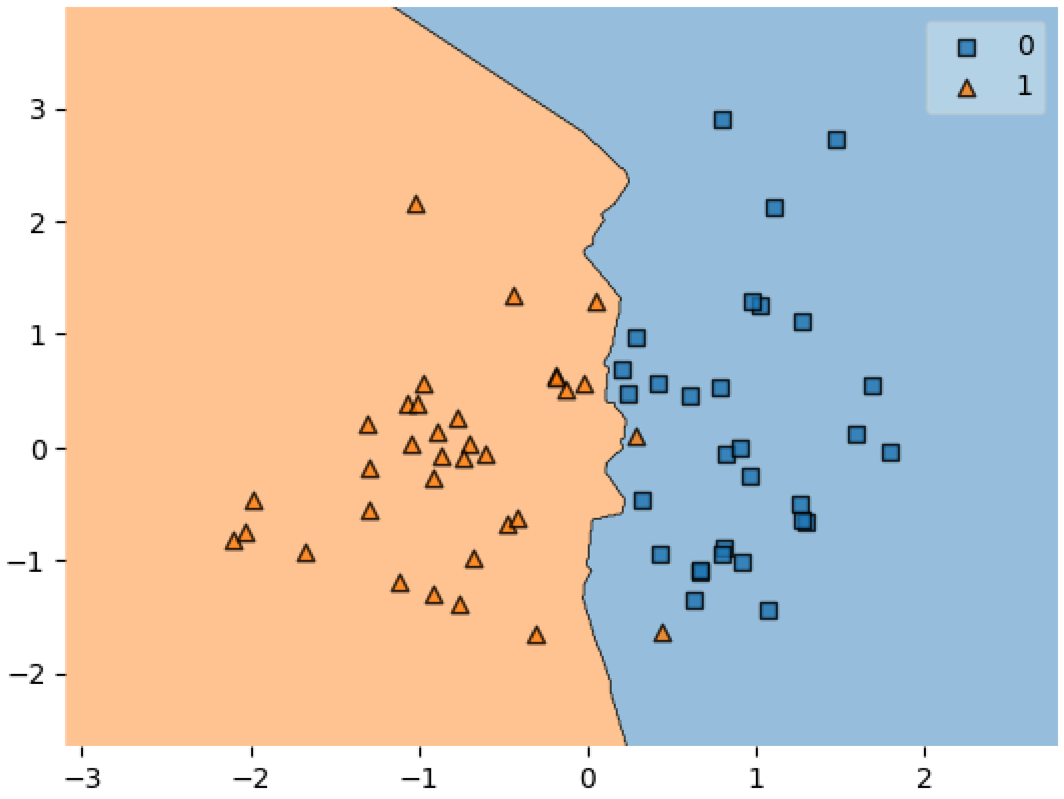
```
k is equal to = 1 , accuracy score 0.9375
```



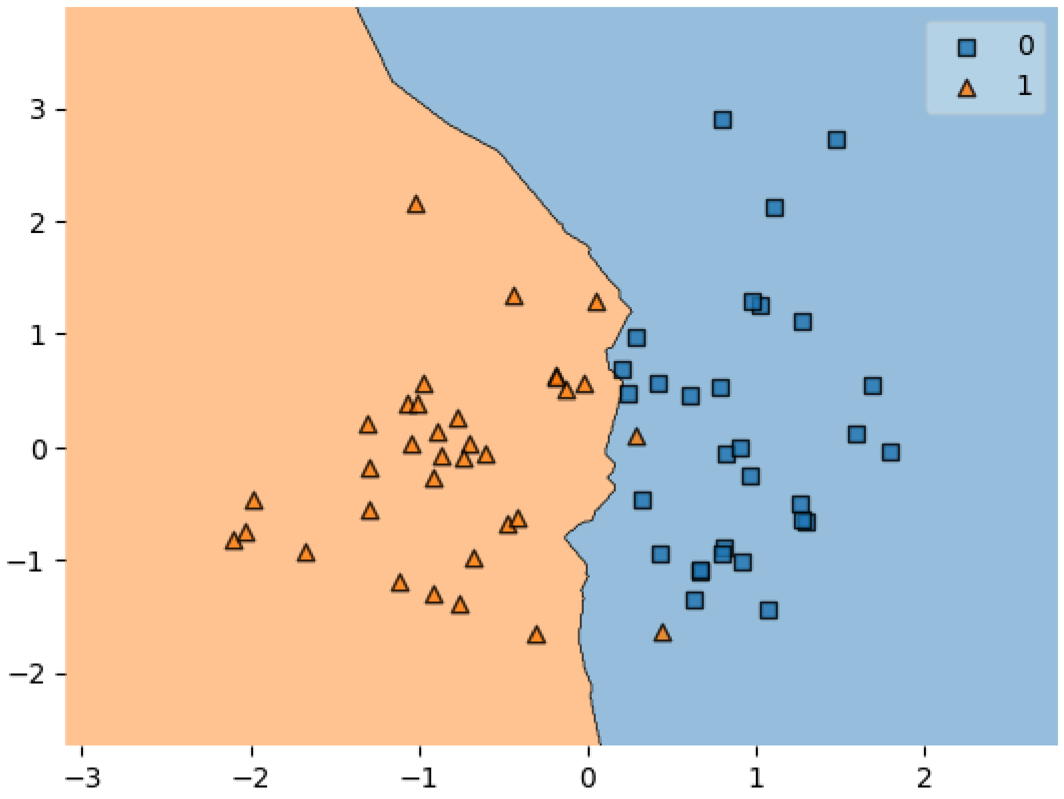
```
k is equal to = 3 , accuracy score 0.9375
```



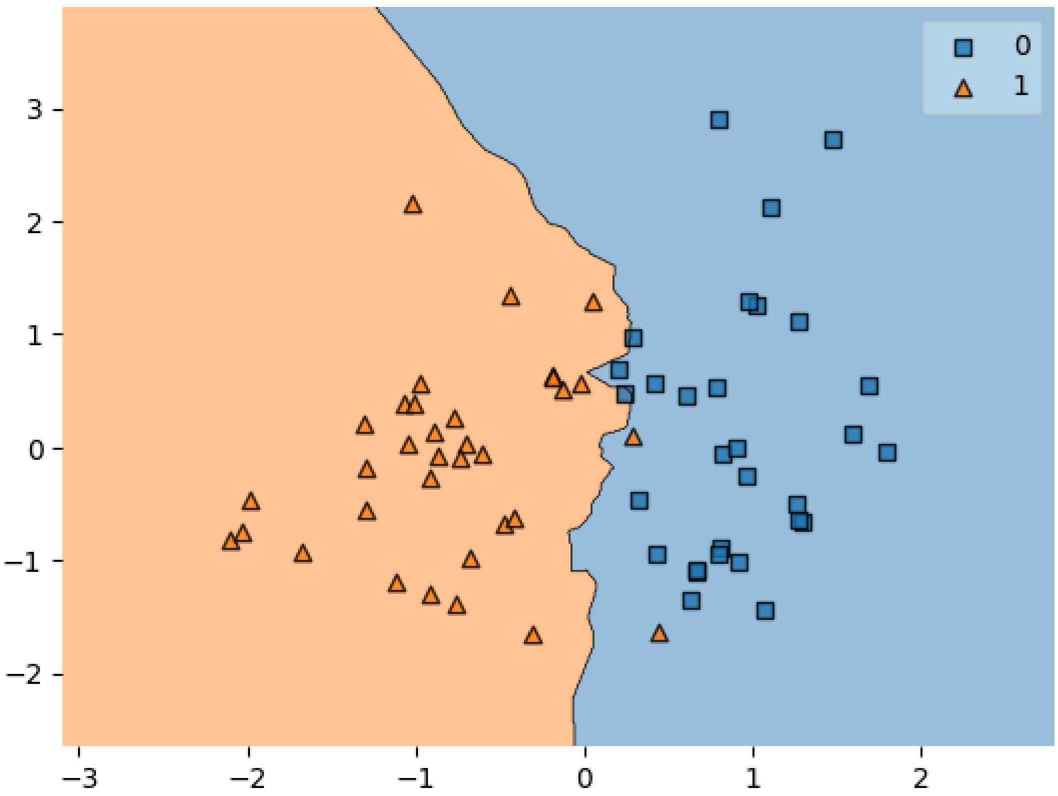
```
k is equal to = 5 , accuracy score 0.9375
```



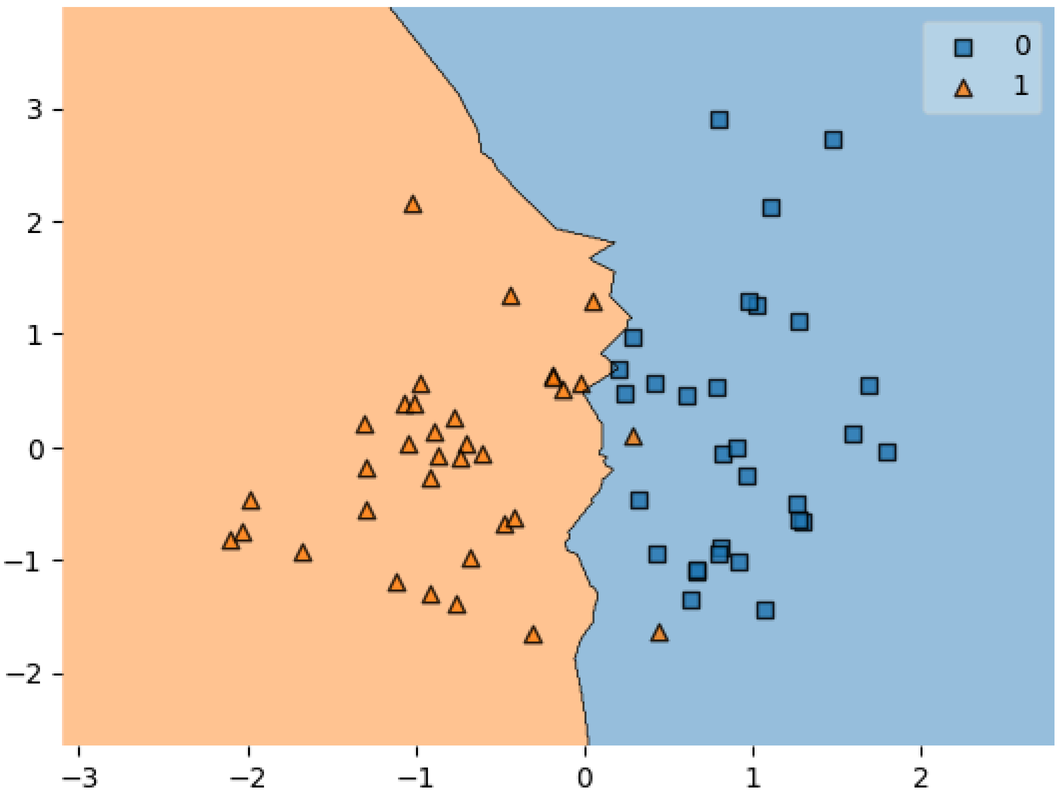
```
k is equal to = 7 , accuracy score 0.9375
```



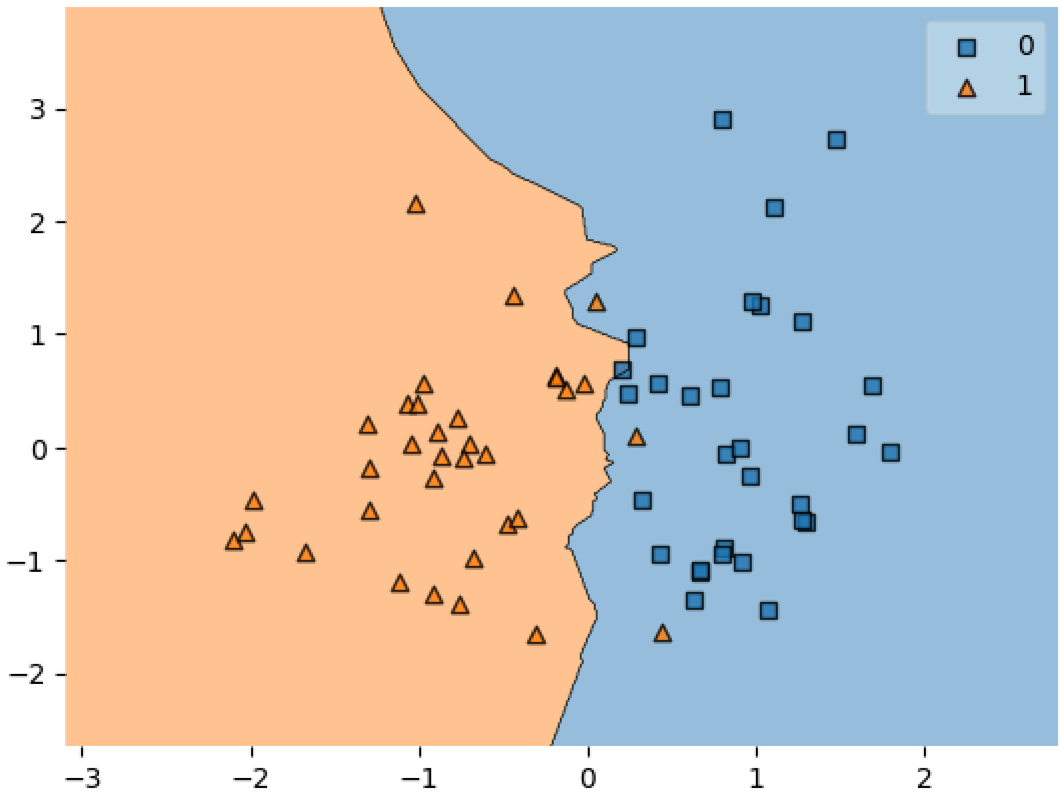
```
k is equal to = 9 , accuracy score 0.9375
```



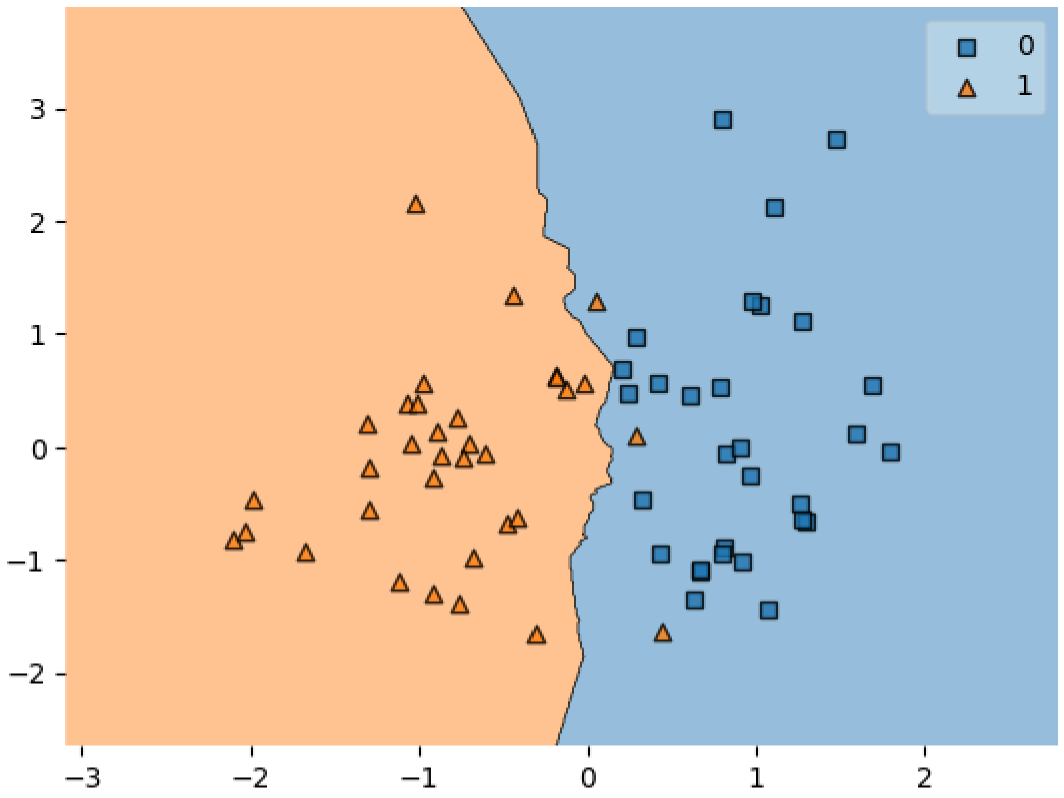
```
k is equal to = 11 , accuracy score 0.9375
```



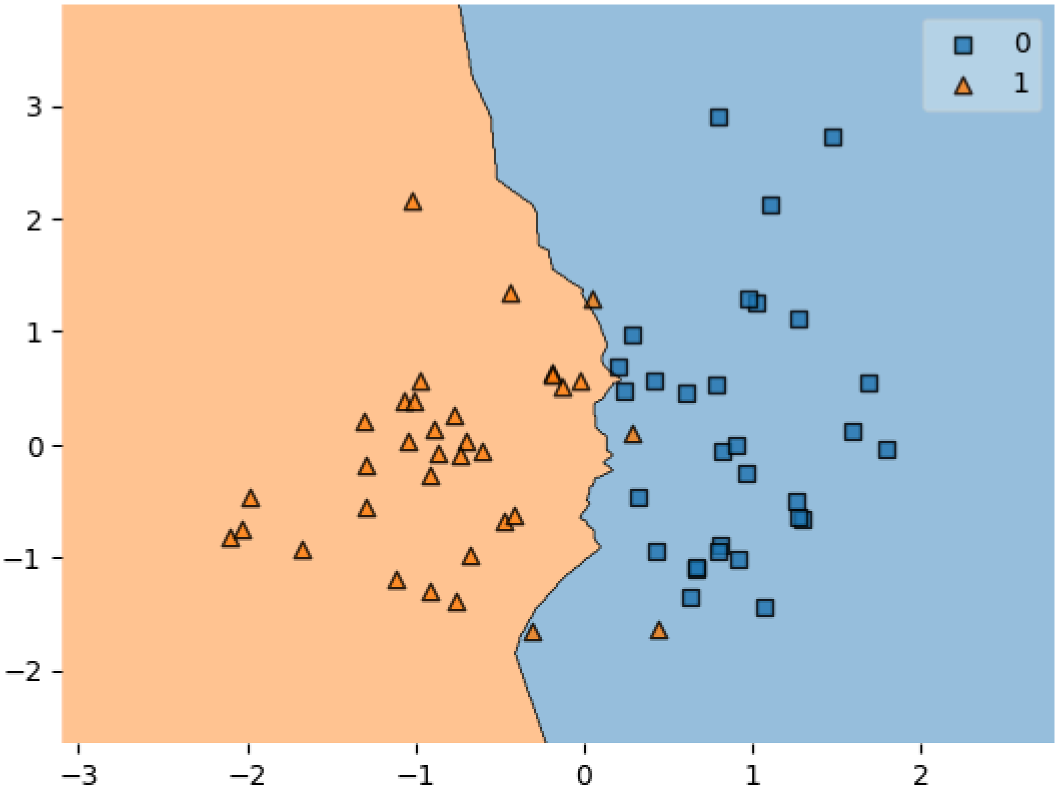
```
k is equal to = 13 , accuracy score 0.9375
```



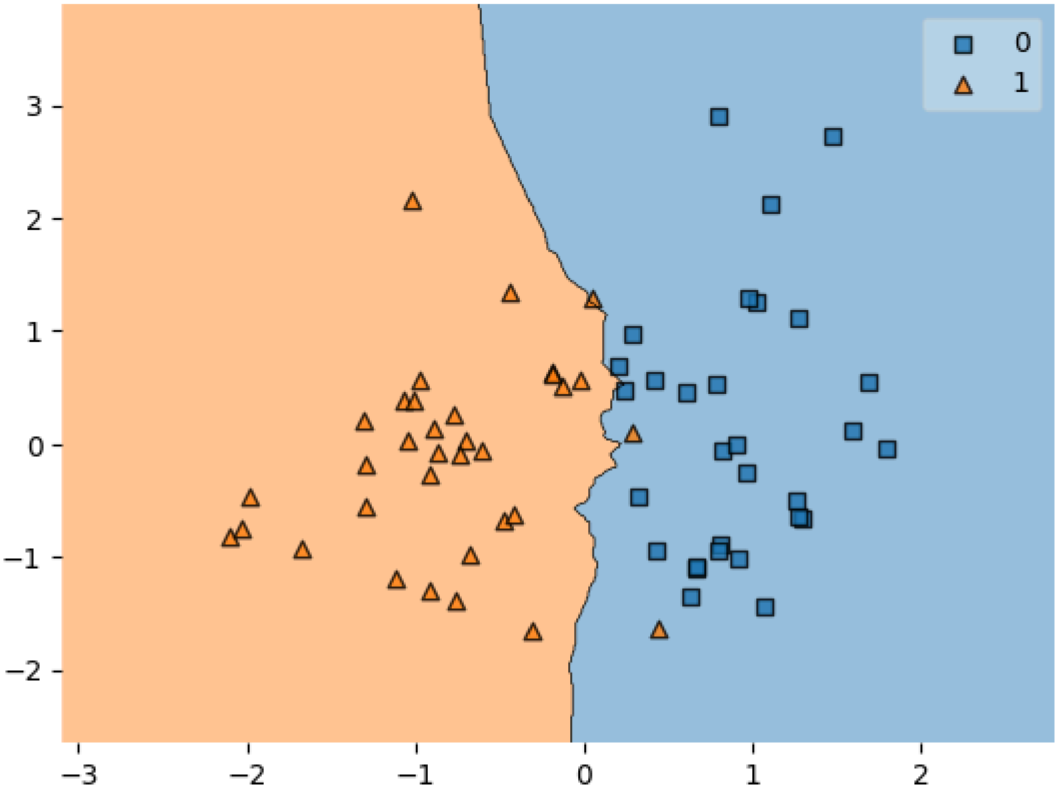
```
k is equal to = 15 , accuracy score 0.9375
```



```
k is equal to = 17 , accuracy score 0.9375
```



```
k is equal to = 19 , accuracy score 0.9375
```



from the above plot at where k=7 I'am able to make the correct decision

5.outlier

```
In [75]: df=pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\5.outl
```

```
In [76]: df
```

Out[76]:

	a	b	c
0	-17.897000	7.662423	0
1	-26.343161	-3.055257	0
2	-19.059771	-8.531838	0
3	-16.383898	-2.352667	0
4	-12.926541	9.074994	0
...	...	...	...
595	4.782462	-29.002590	0
596	3.990671	-27.664533	0
597	1.968937	-27.666538	0
598	0.397395	-28.864856	0
599	2.778266	-29.555160	0

600 rows × 3 columns

```
In [77]: fv=df.iloc[:,:2]
         cv=df.iloc[:,-1]
```

```
In [78]: df["c"].unique()

Out[78]: array([0, 1], dtype=int64)
```

```
In [79]: x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,random_state=2)
```

```
In [80]: x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

```
In [81]: std=StandardScaler()
         px_trainf=std.fit_transform(x_trainf)
         px_test=std.transform(x_test)
         px_cv=std.transform(x_cv)
```

```
In [82]: knn=KNeighborsClassifier(n_neighbors=1)
         model=knn.fit(px_trainf,y_trainf)
         predicted=model.predict(px_cv)
```

```
In [83]: accuracy_score(y_cv,predicted)

Out[83]: 1.0
```

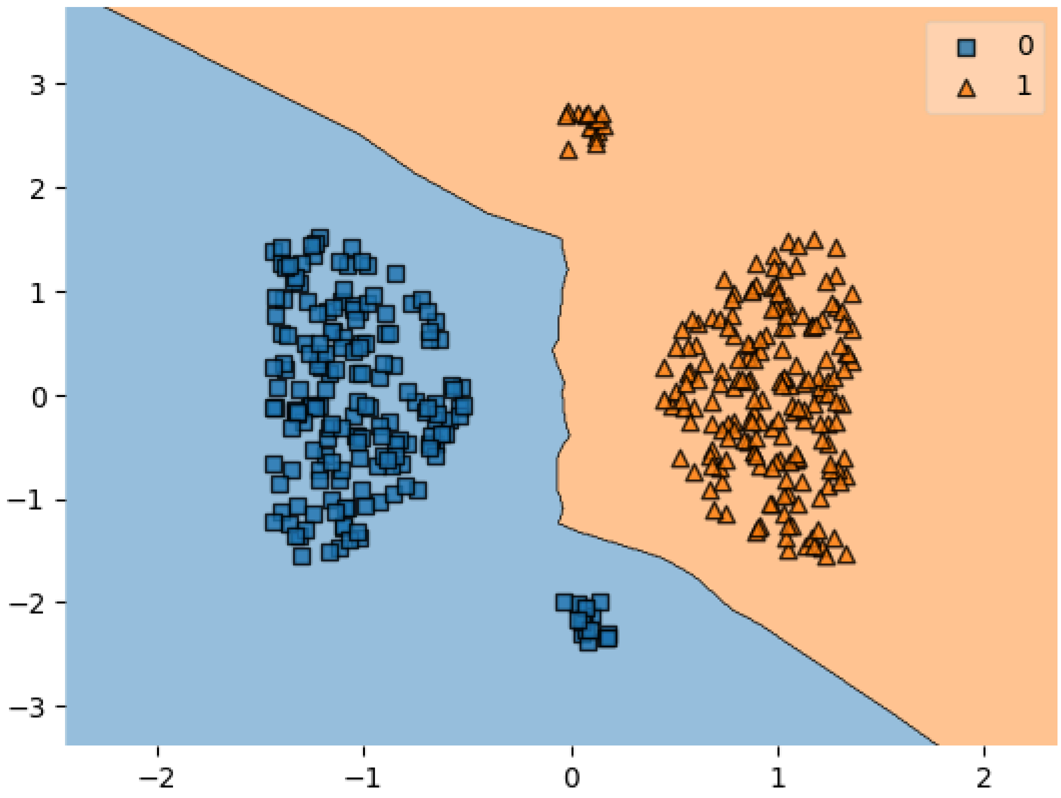
In [84]:

```python
from sklearn.metrics import accuracy_score
```

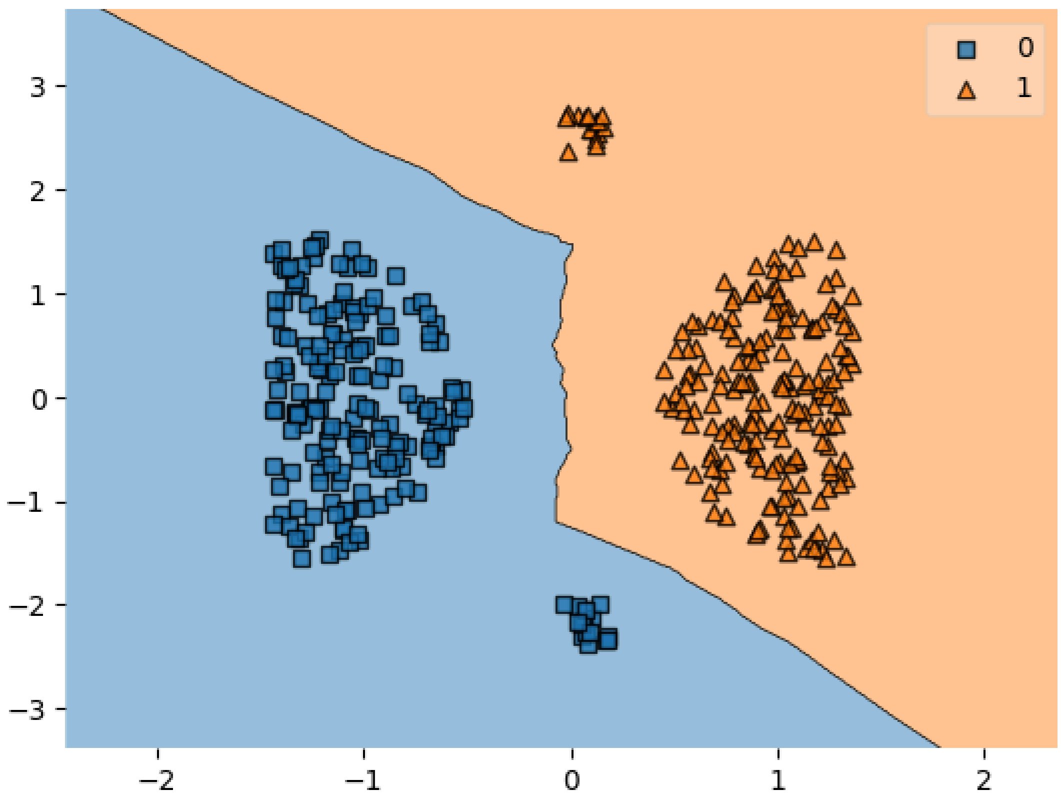
In [85]:

```python
for i in range(1,20,2):
    knn=KNeighborsClassifier(n_neighbors=i)
    model=knn.fit(px_trainf,y_trainf)
    predicted=model.predict(px_cv)
    print(accuracy_score(y_cv,predicted))
    plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
    plt.show()
```

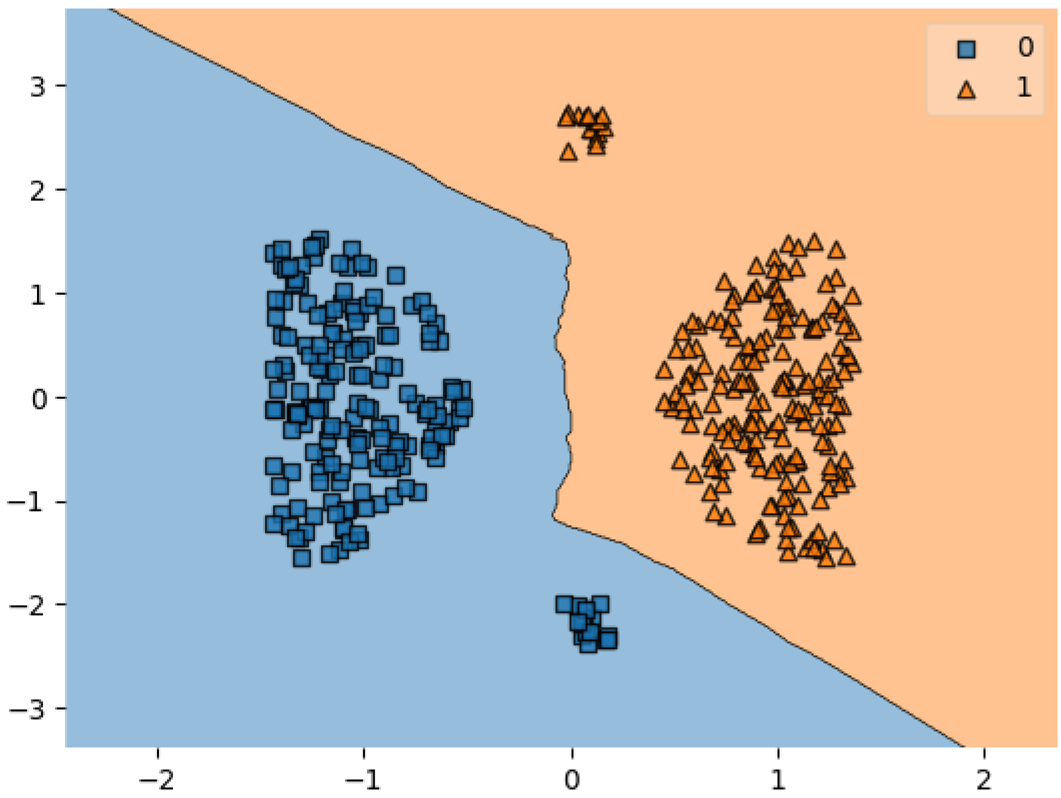
```
1.0
```



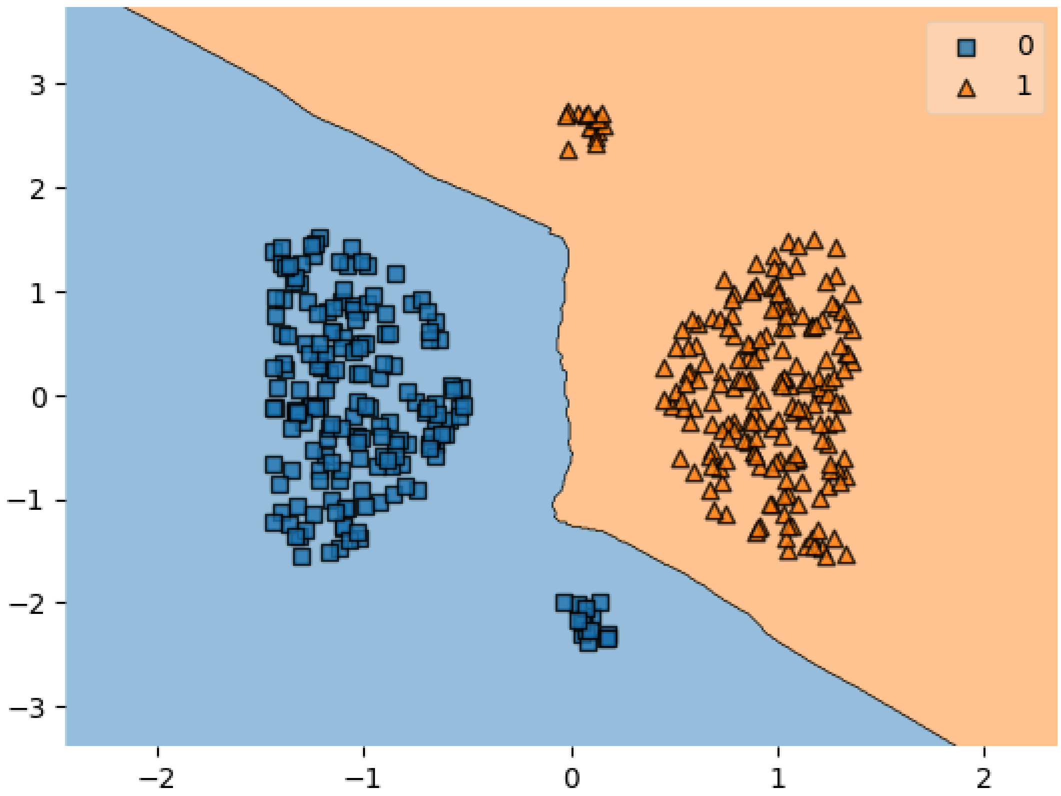
```
1.0
```



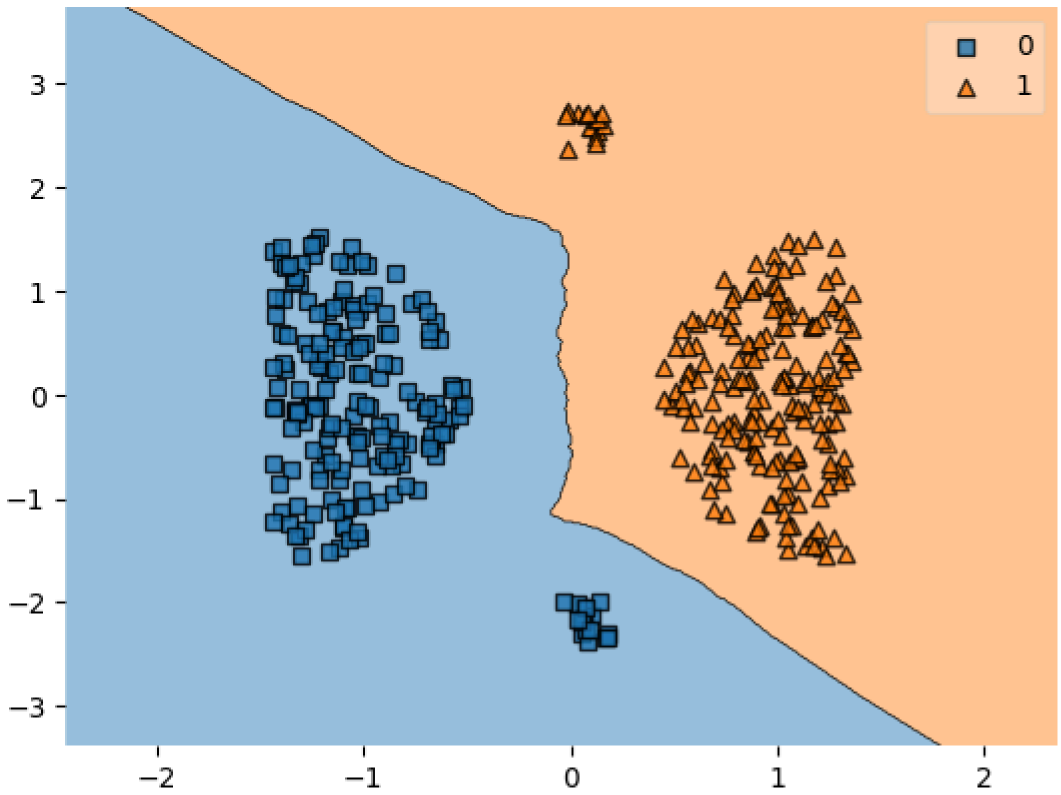
```
1.0
```



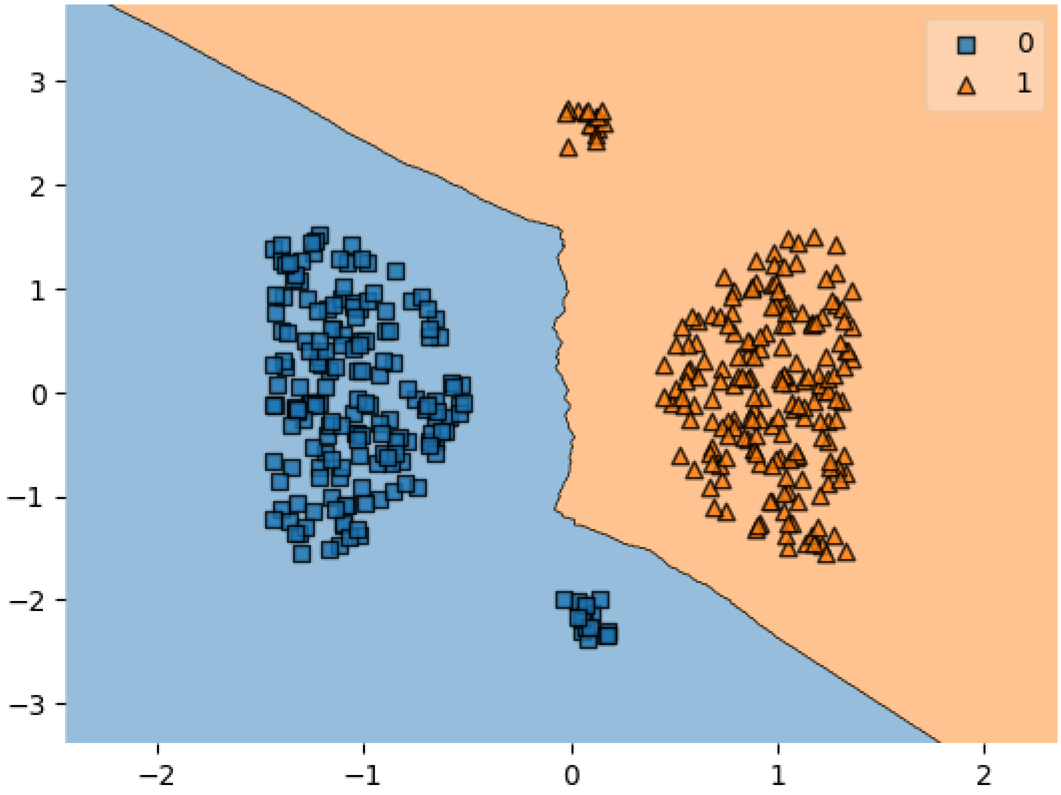
```
1.0
```



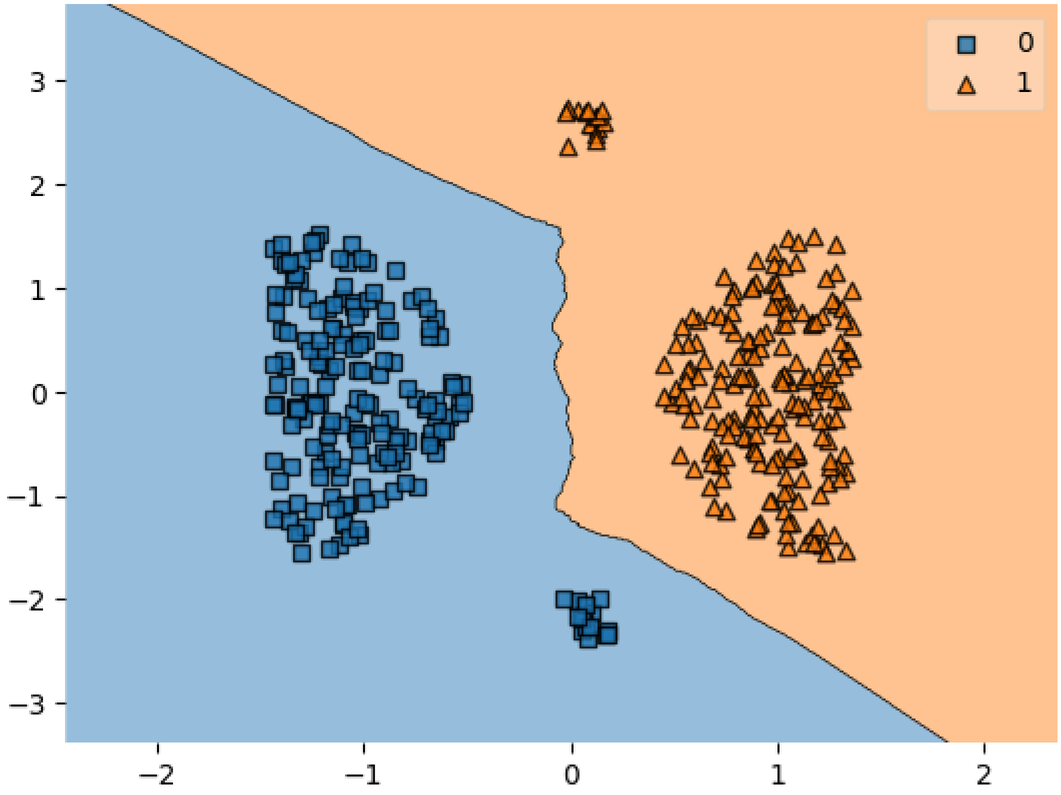
```
1.0
```



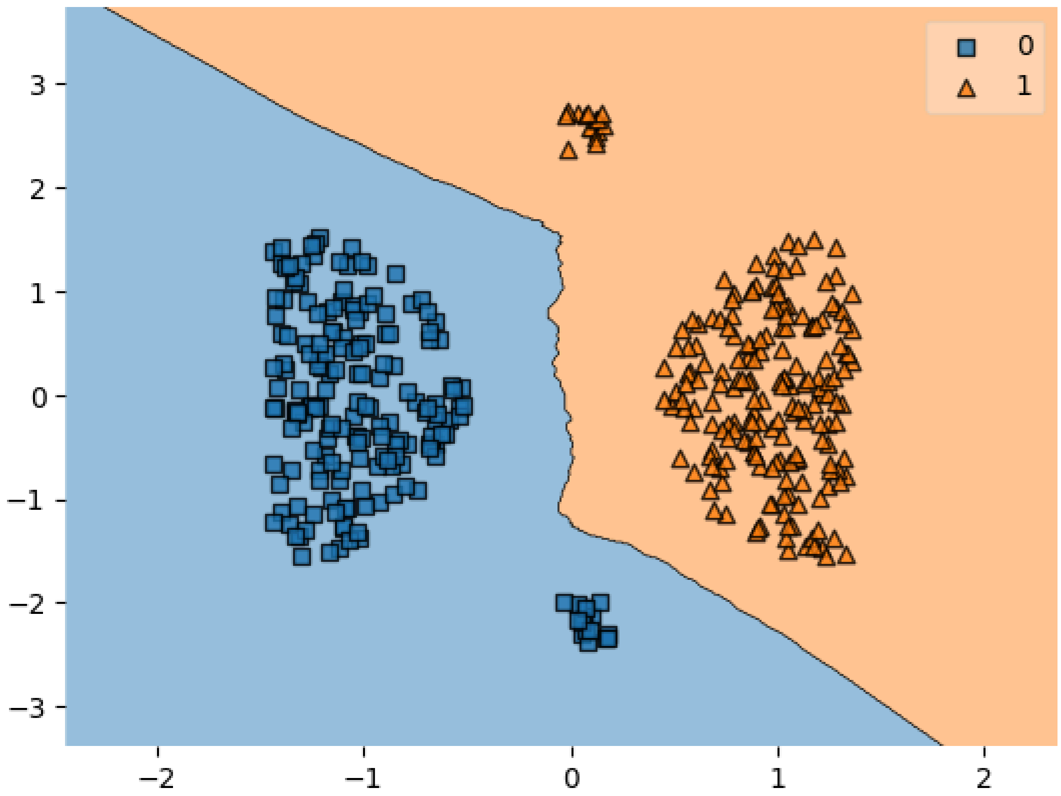
```
1.0
```



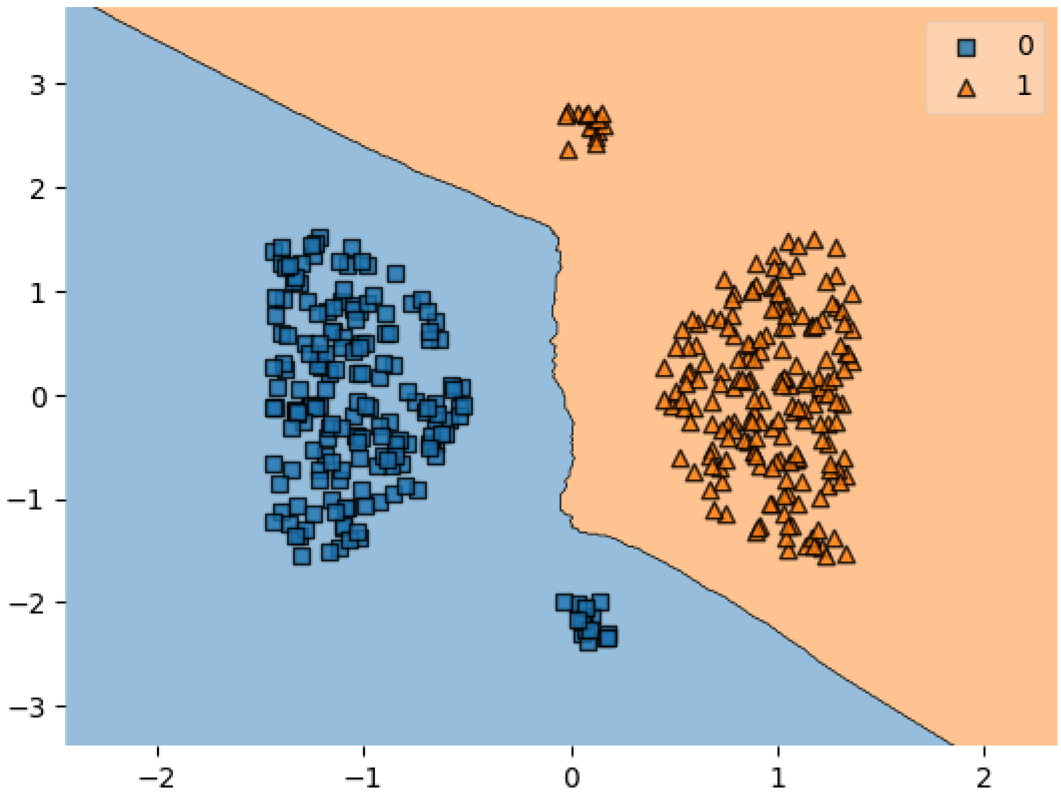
```
1.0
```



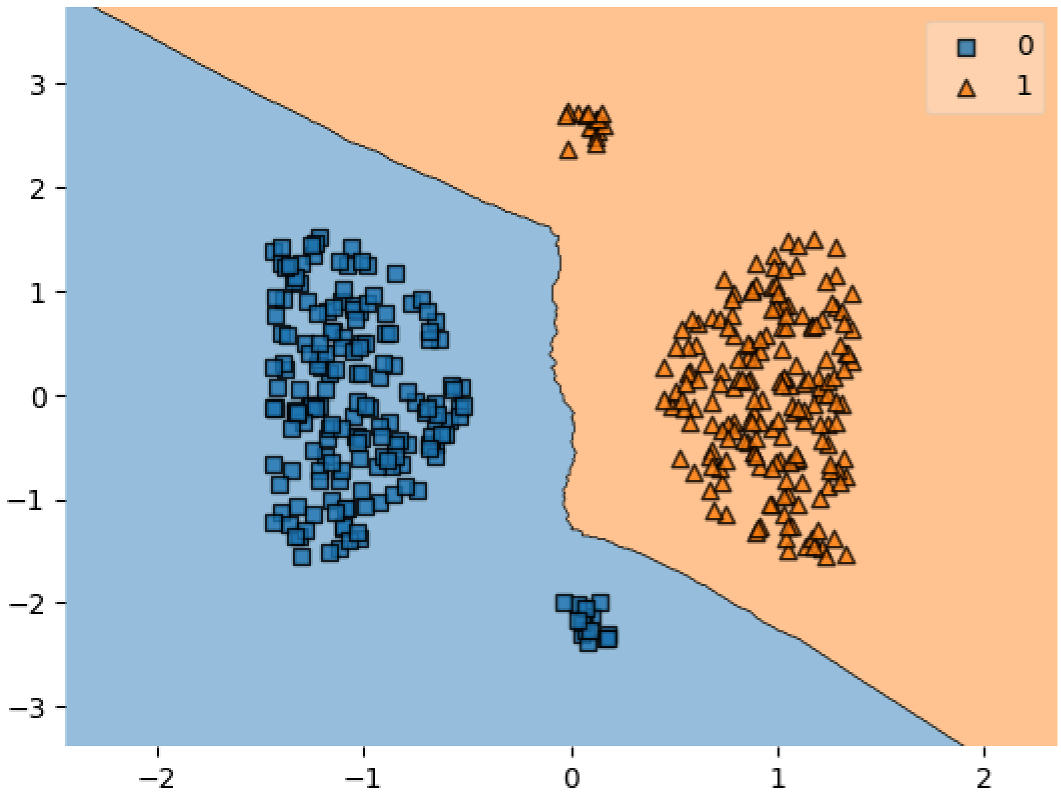
```
1.0
```



```
1.0
```



```
1.0
```



from the above plot at where k=any I'am able to make the correct decision

6. overlap

```
In [89]: df=pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\6.over
```

```
In [90]: df
```

Out[90]:

	a	b	c
0	7.0	3.2	0
1	6.4	3.2	0
2	6.9	3.1	0
3	5.5	2.3	0
4	6.5	2.8	0
...	...	...	...
95	6.7	3.0	1
96	6.3	2.5	1
97	6.5	3.0	1
98	6.2	3.4	1
99	5.9	3.0	1

100 rows × 3 columns

In [91]:

```
df.describe()
```

Out[91]:

	a	b	c
count	100.000000	100.000000	100.000000
mean	6.262000	2.872000	0.500000
std	0.662834	0.332751	0.502519
min	4.900000	2.000000	0.000000
25%	5.800000	2.700000	0.000000
50%	6.300000	2.900000	0.500000
75%	6.700000	3.025000	1.000000
max	7.900000	3.800000	1.000000

In [92]:

```
fv=df.iloc[:,:2]
cv=df.iloc[:,-1]
```

In [93]:

```
x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,random_state=2,s
x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

In [94]:

```
std=StandardScaler()
px_trainf=std.fit_transform(x_trainf)
px_test=std.transform(x_test)
px_cv=std.transform(x_cv)
```

In [95]:

```
knn=KNeighborsClassifier(n_neighbors=1)
model=knn.fit(px_trainf,y_trainf)
predicted=model.predict(px_cv)
```

In [96]:

```
accuracy_score(y_cv,predicted)
```

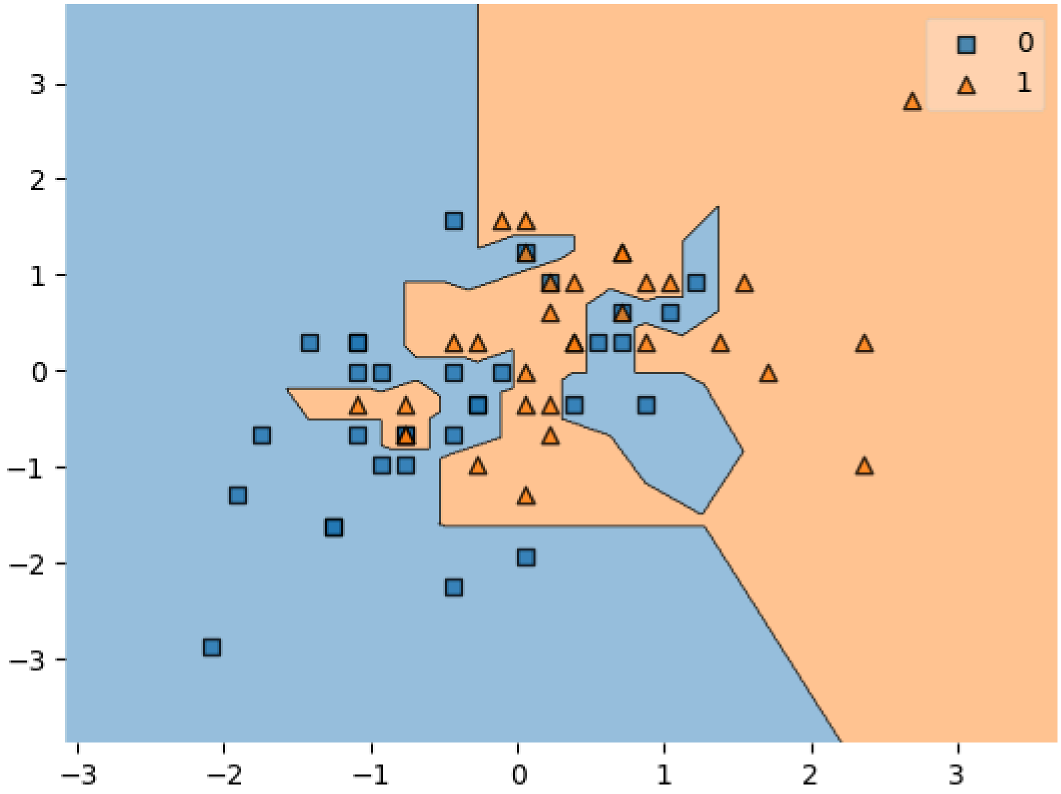
Out[96]:

```
0.625
```

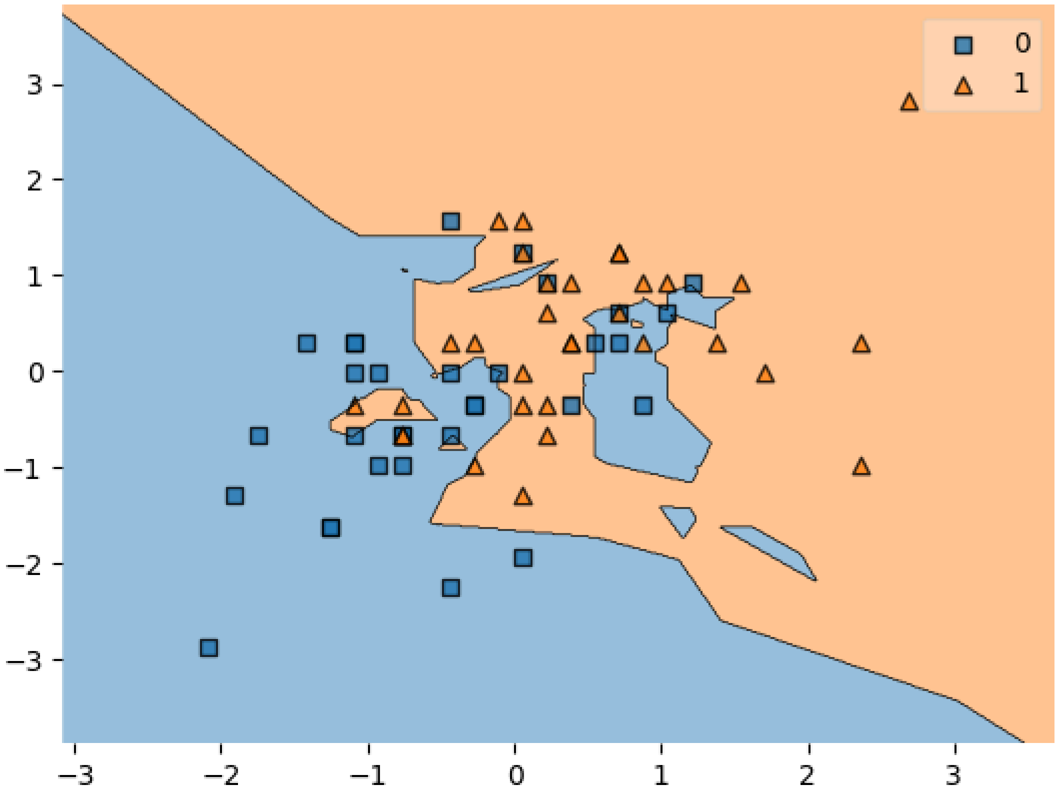
In [97]:

```
for i in range(1,20,2):
    knn=KNeighborsClassifier(n_neighbors=i)
    model=knn.fit(px_trainf,y_trainf)
    predicted=model.predict(px_cv)
    print(f"k is equal to = {i} , accuracy score",accuracy_score(y_cv,predicted))
    plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
    plt.show()
```

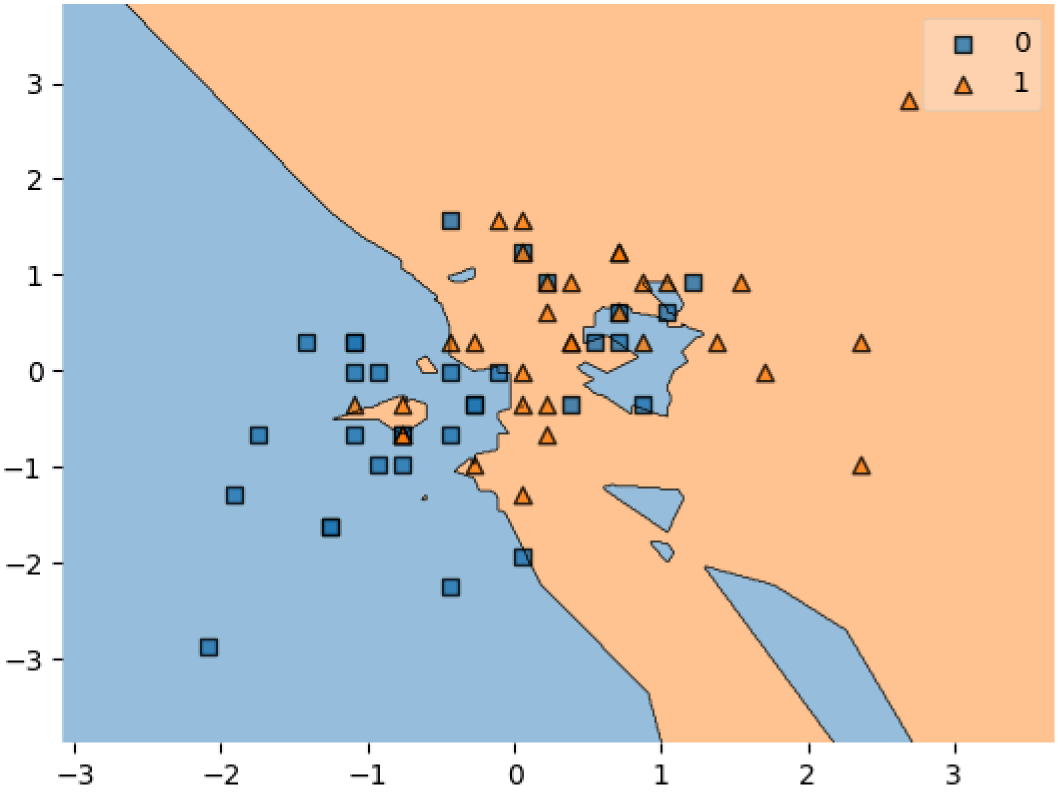
```
k is equal to = 1 , accuracy score 0.625
```



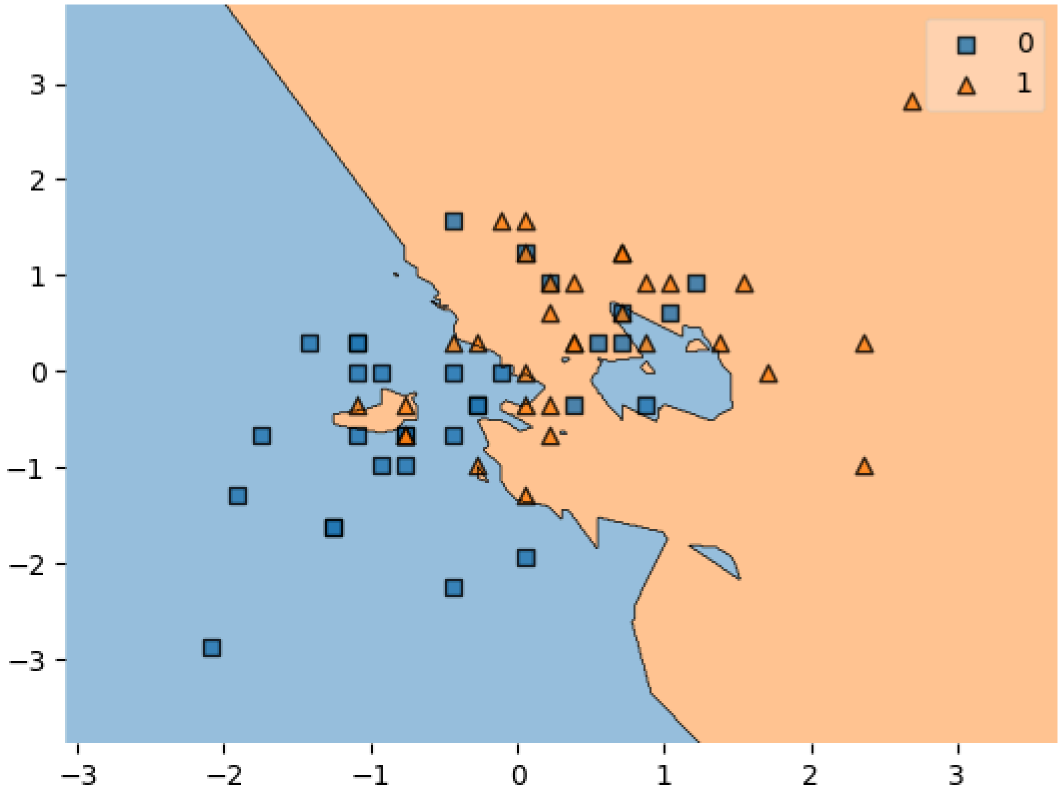
```
k is equal to = 3 , accuracy score 0.5625
```



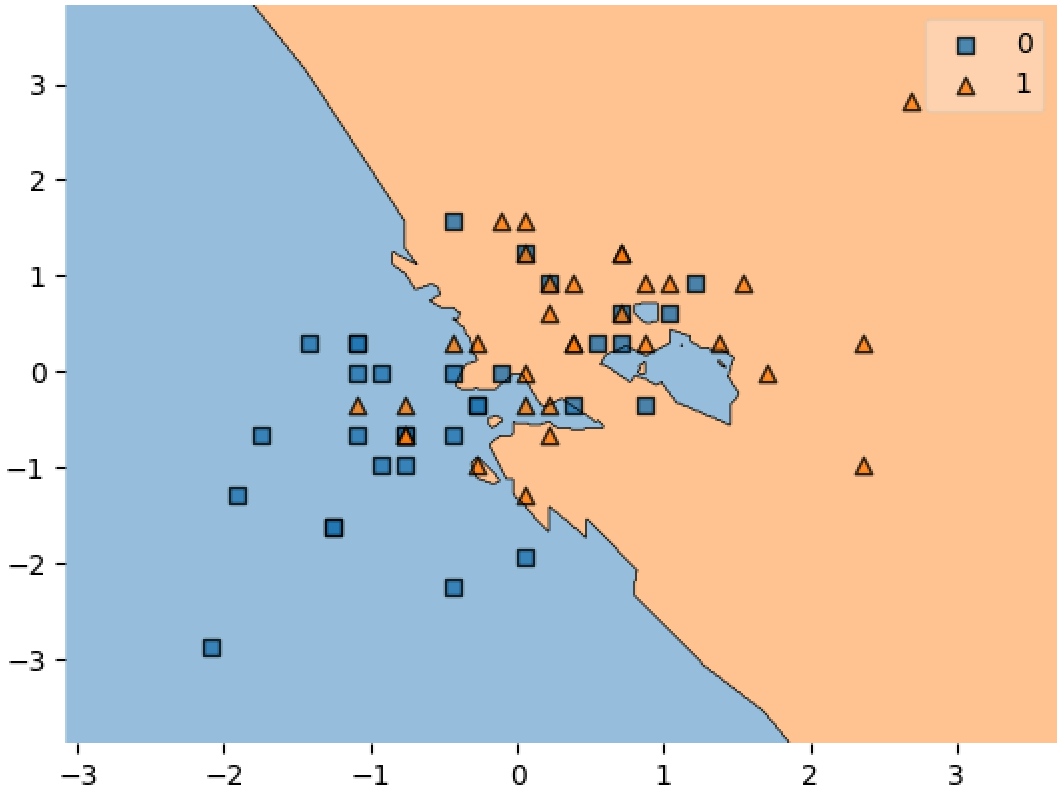
```
k is equal to = 5 , accuracy score 0.5625
```



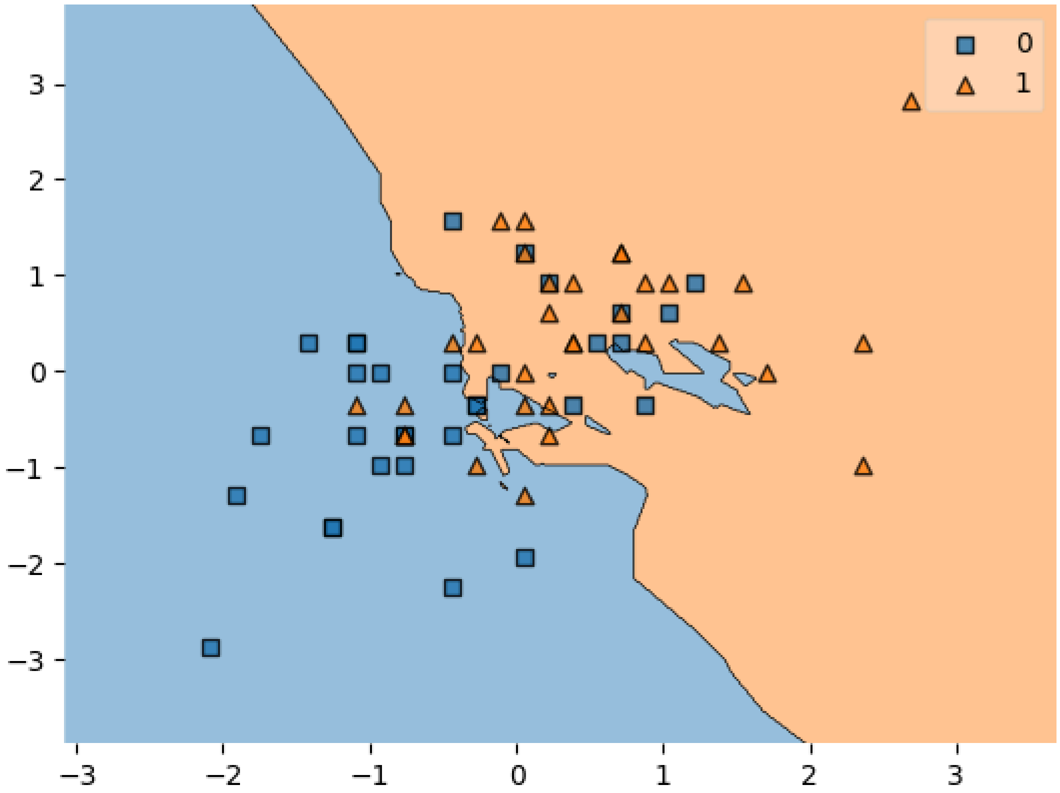
```
k is equal to = 7 , accuracy score 0.625
```



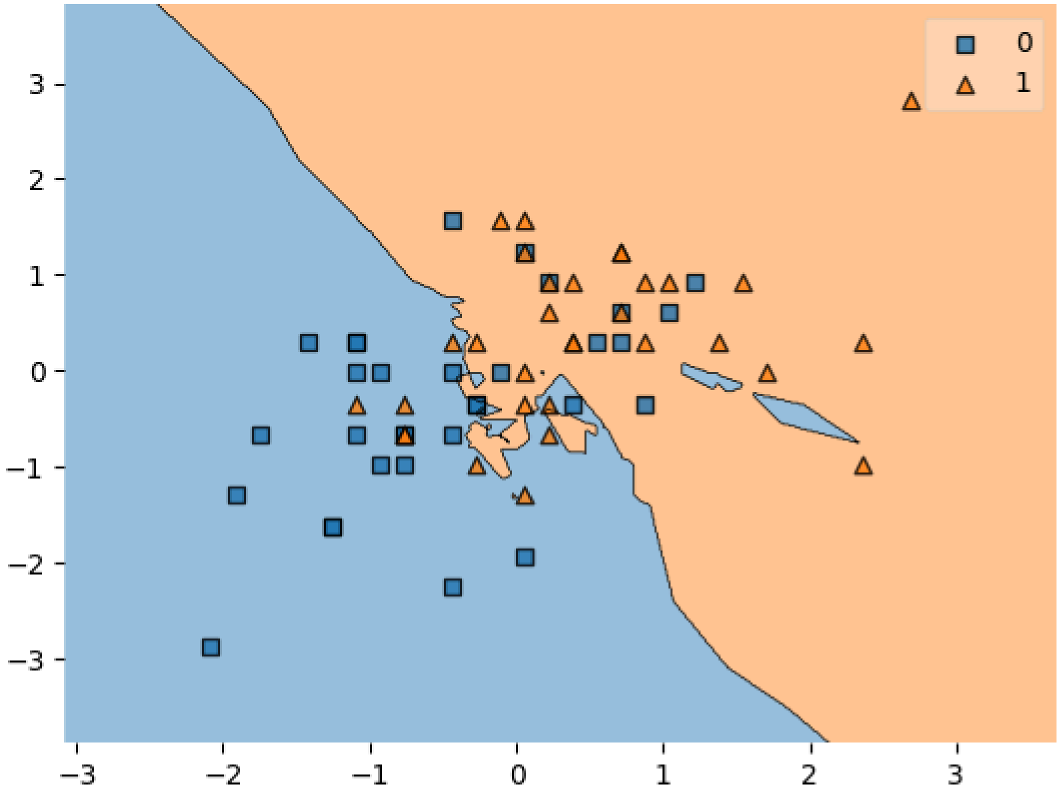
```
k is equal to = 9 , accuracy score 0.625
```



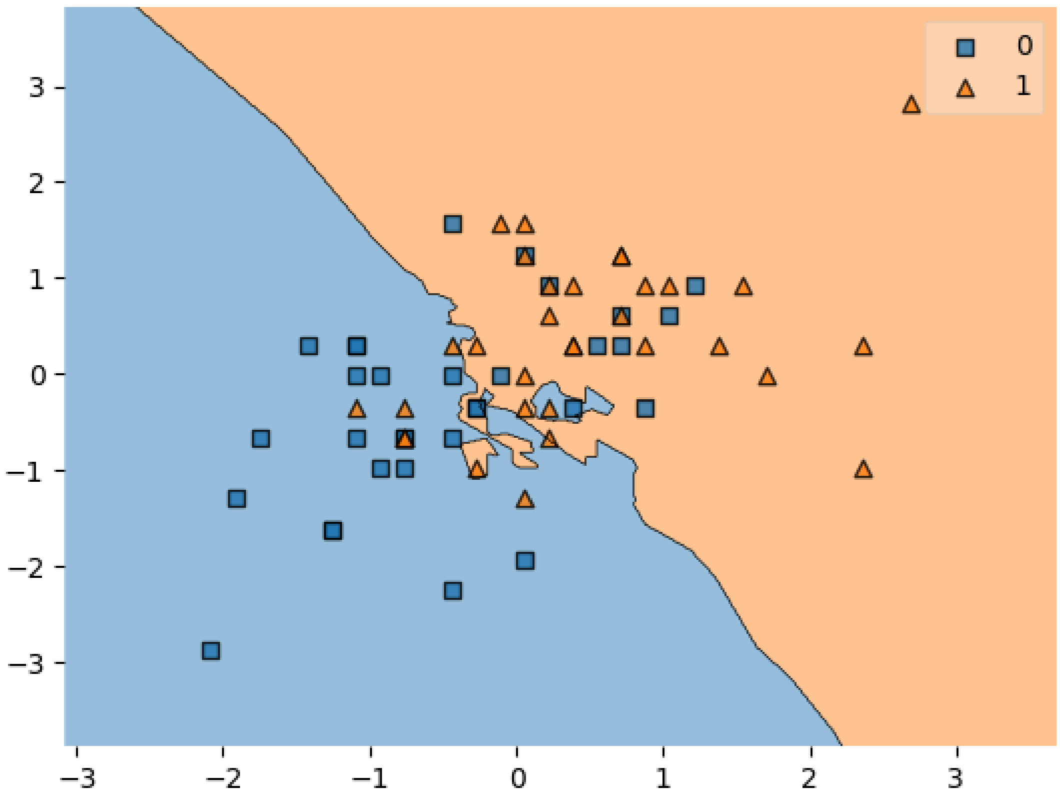
```
k is equal to = 11 , accuracy score 0.625
```



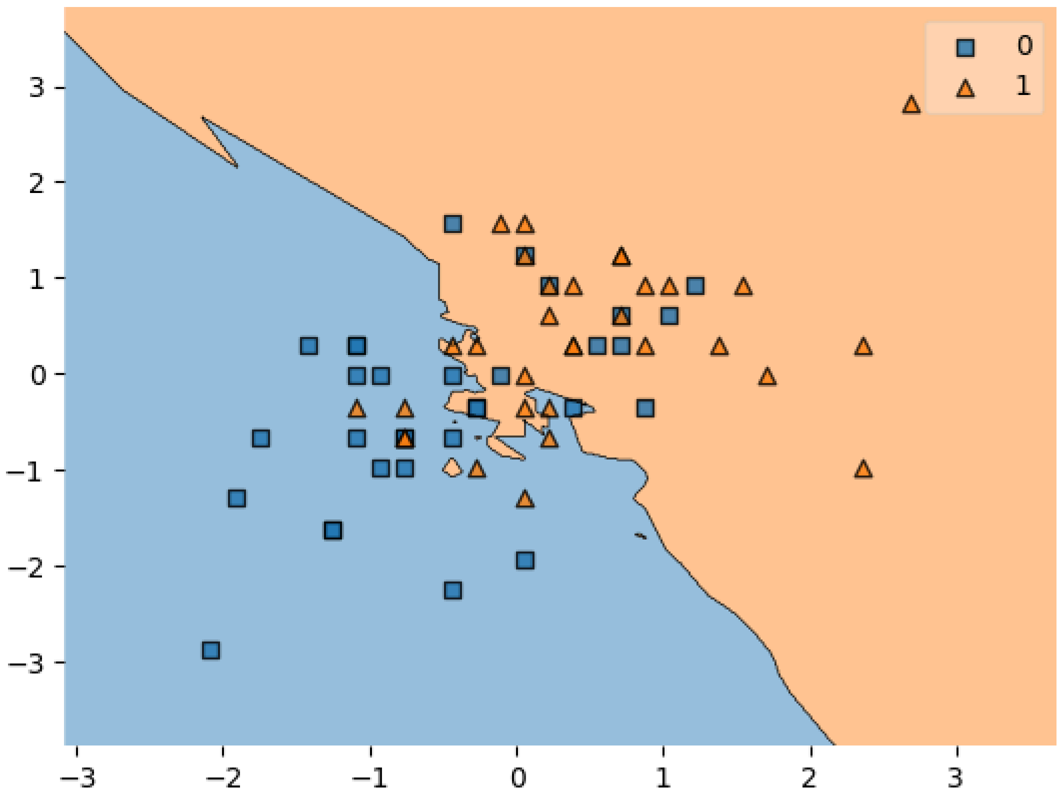
```
k is equal to = 13 , accuracy score 0.5625
```



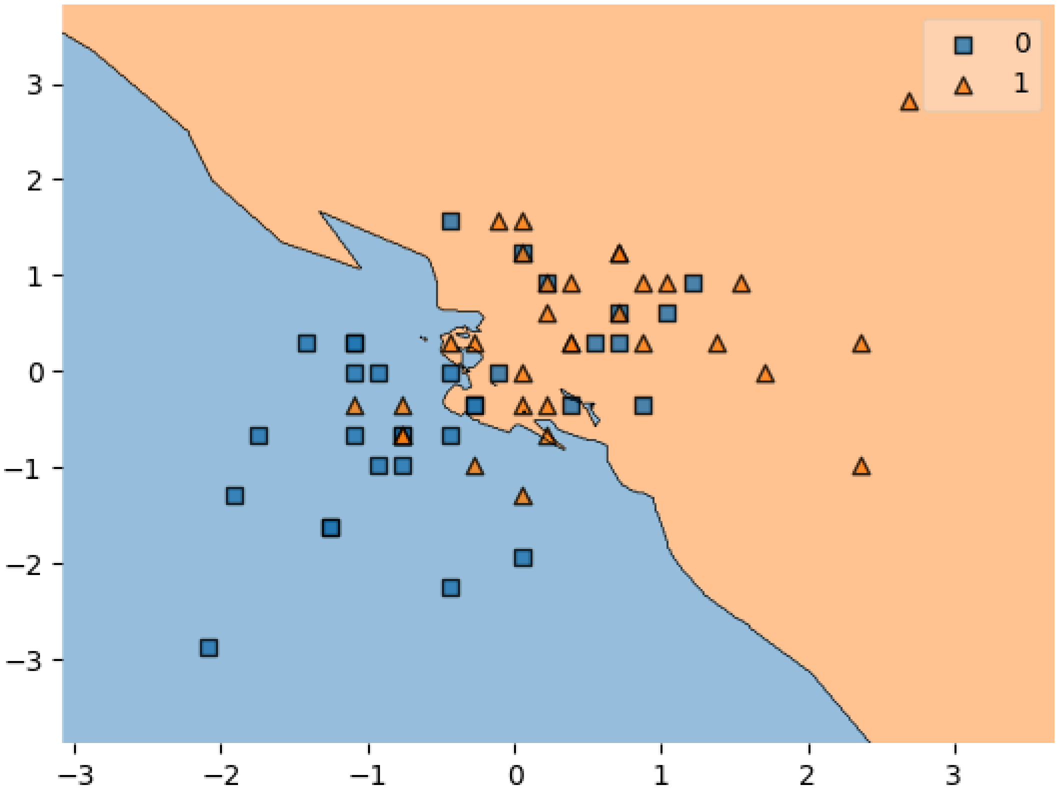
```
k is equal to = 15 , accuracy score 0.5625
```



```
k is equal to = 17 , accuracy score 0.6875
```



```
k is equal to = 19 , accuracy score 0.625
```



from the above plot at where k=19 & 17 I'am able to make the correct decision

7.XOR

```
In [118…  df=pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\7.xor.
```

```
In [119…  fv=df.iloc[:,:2]
          cv=df.iloc[:,-1]
```

```
In [120…  cv=cv.astype(int)
```

```
In [121…  x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,random_state=2,s
          x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

```
In [122…  std=StandardScaler()
          px_trainf=std.fit_transform(x_trainf)
          px_test=std.transform(x_test)
          px_cv=std.transform(x_cv)
```

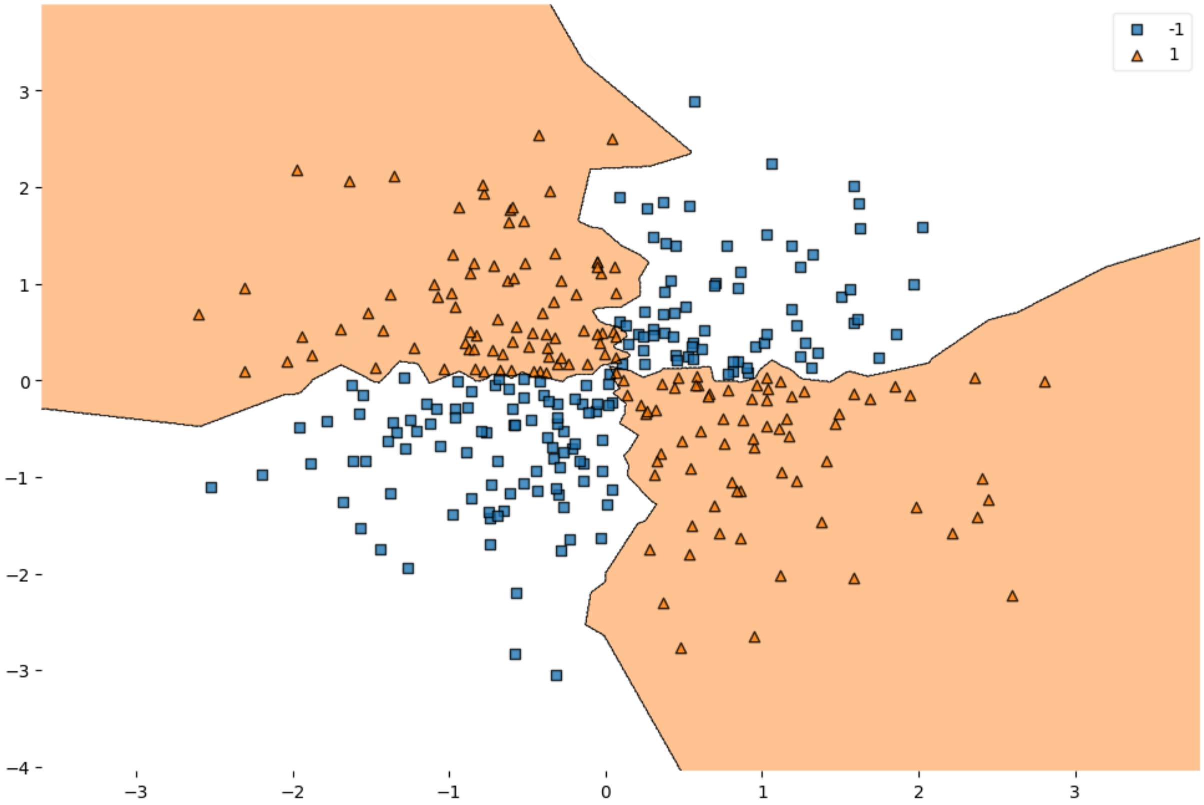
```
In [123…  knn=KNeighborsClassifier(n_neighbors=1)
          model=knn.fit(px_trainf,y_trainf)
          predicted=model.predict(px_cv)
```

```
In [124…  accuracy_score(y_cv,predicted)
```

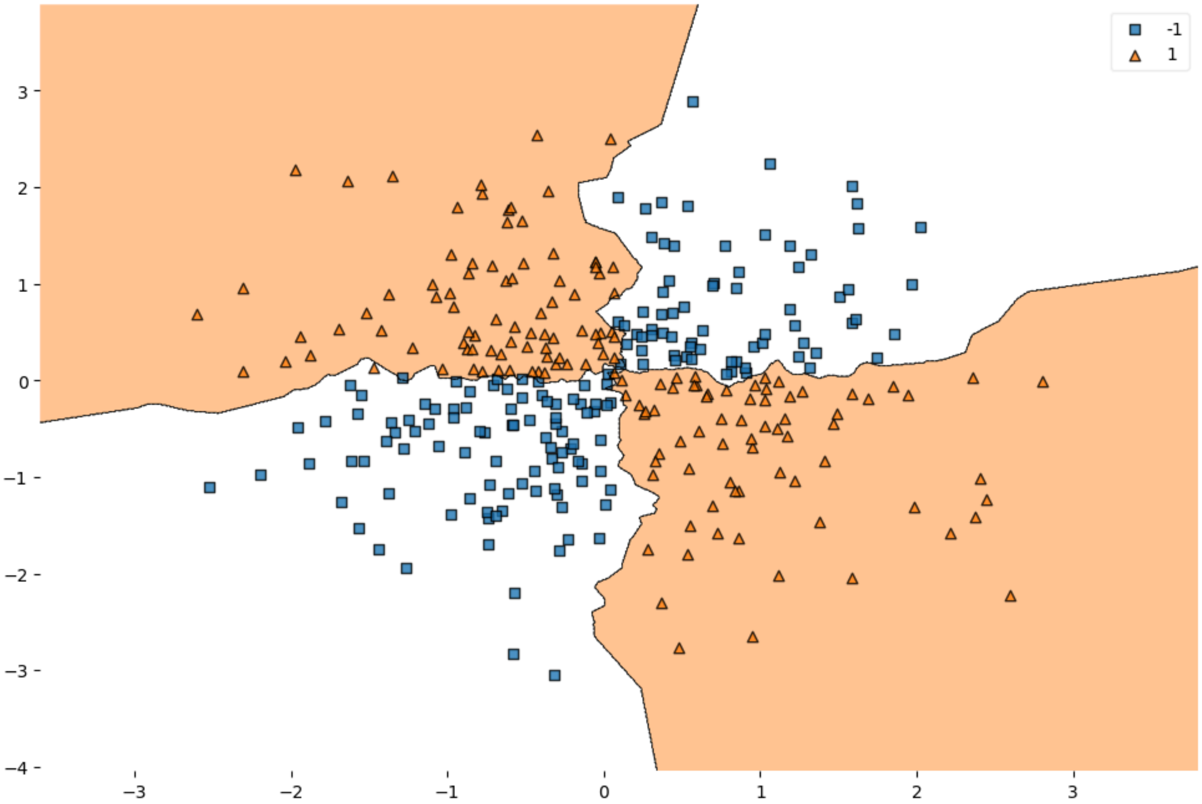
```
Out[124…  0.925
```

```
In [125…  for i in range(1,20,2):
              knn=KNeighborsClassifier(n_neighbors=i)
              model=knn.fit(px_trainf,y_trainf)
              predicted=model.predict(px_cv)
              plt.figure(figsize=(12,8))
              print(f"Accuracy for n_neighbors={i}: {accuracy_score(y_cv,predicted)}")
              plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
              plt.show()
```

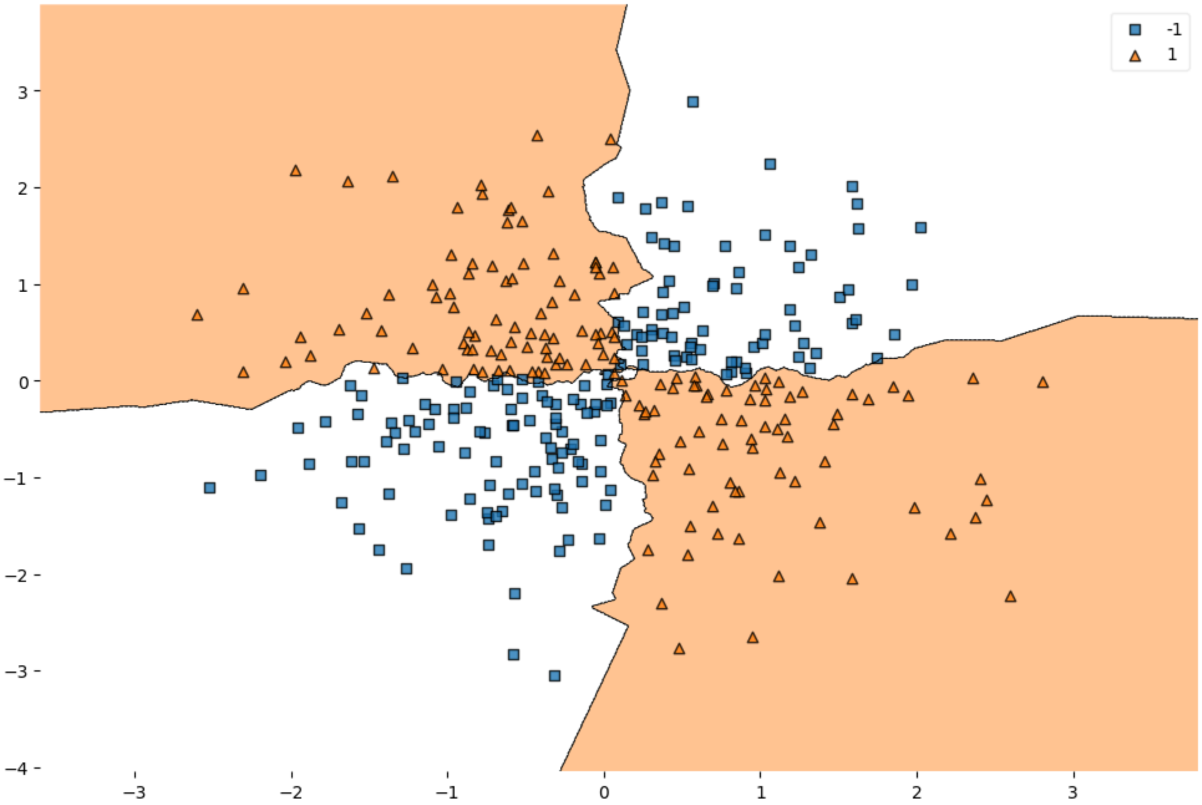
```
Accuracy for n_neighbors=1: 0.925
```



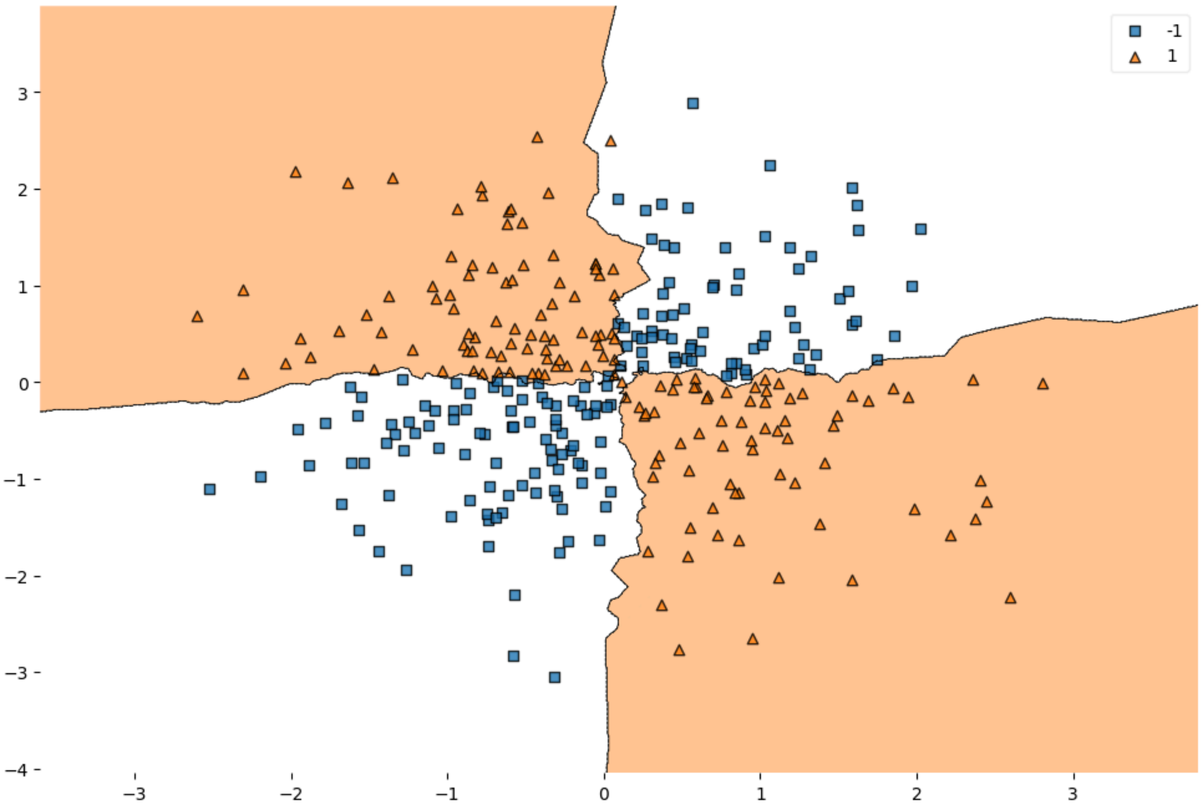
```
Accuracy for n_neighbors=3: 0.9375
```



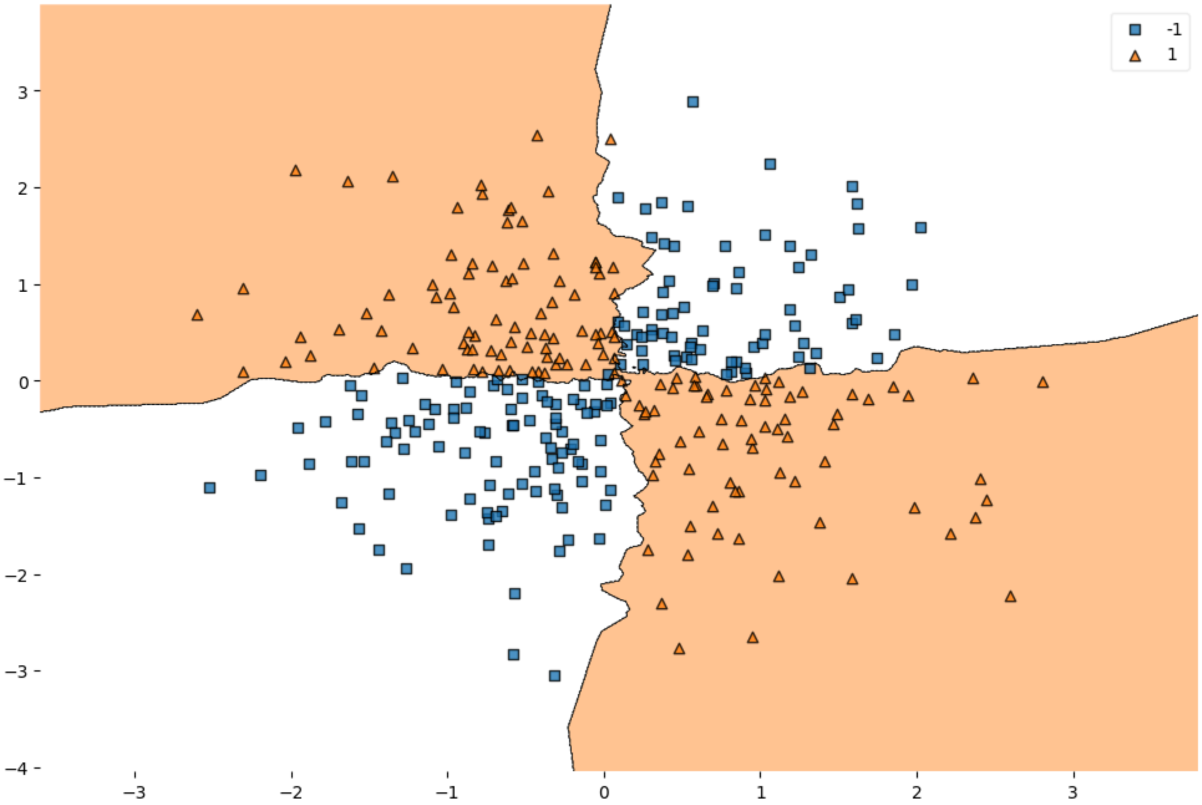
```
Accuracy for n_neighbors=5: 0.9375
```



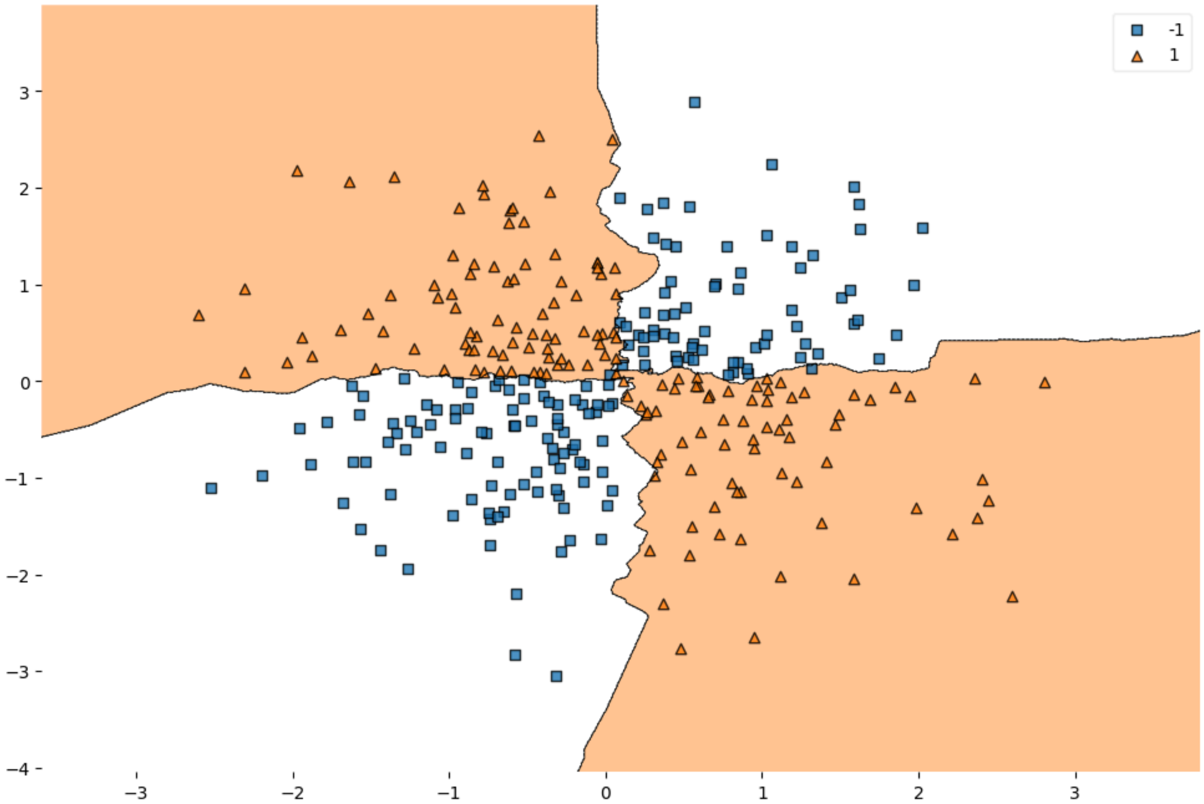
```
Accuracy for n_neighbors=7: 0.9375
```



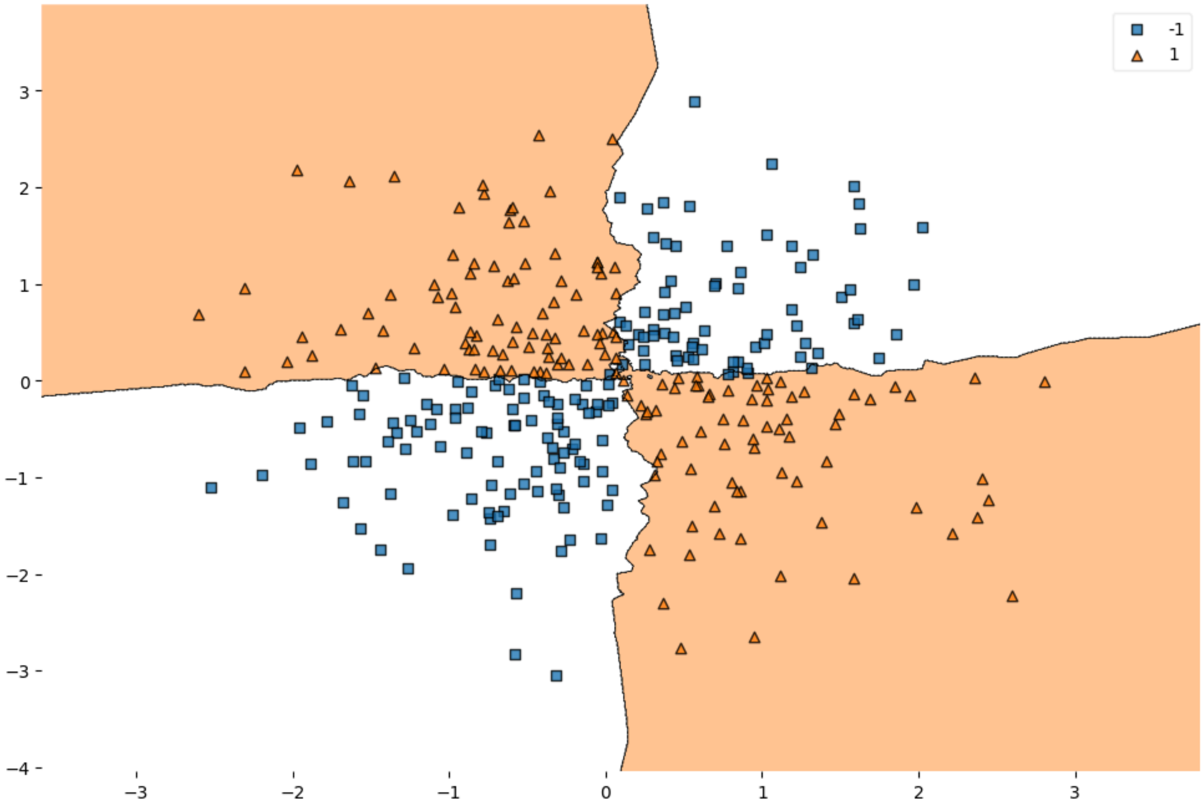
```
Accuracy for n_neighbors=9: 0.95
```



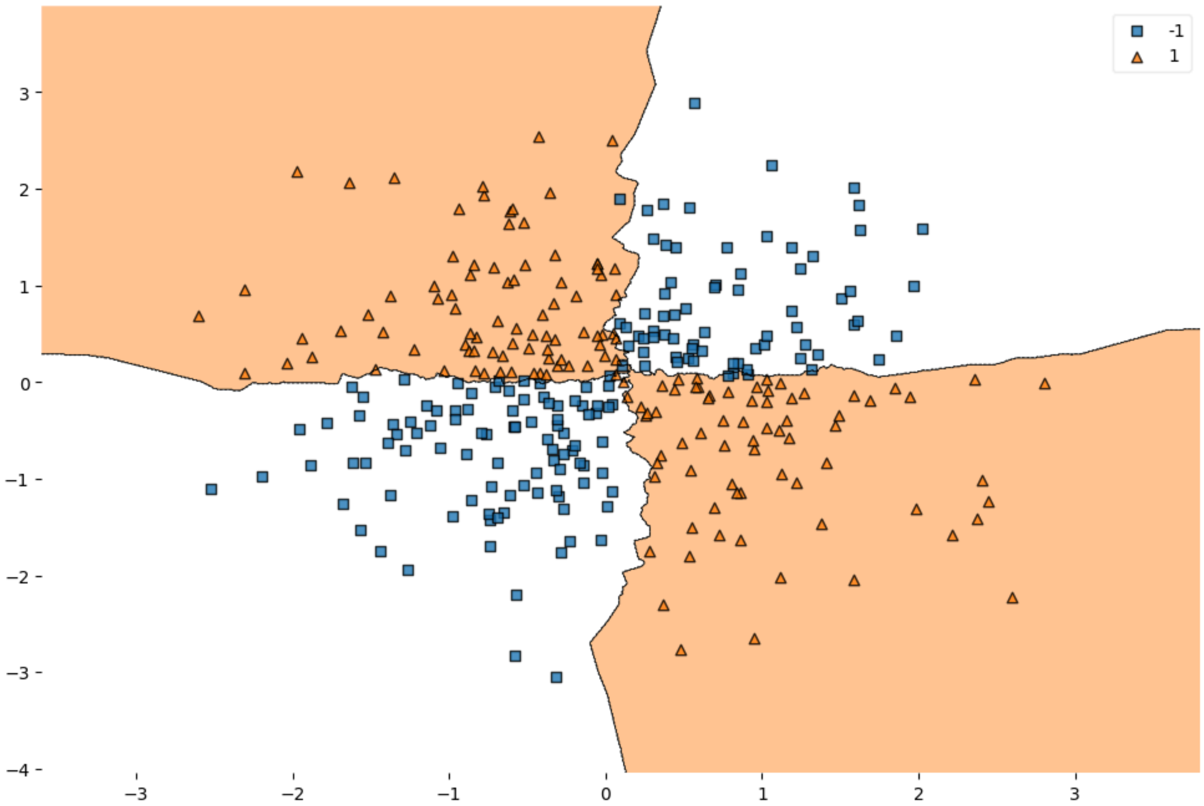
```
Accuracy for n_neighbors=11: 0.9375
```



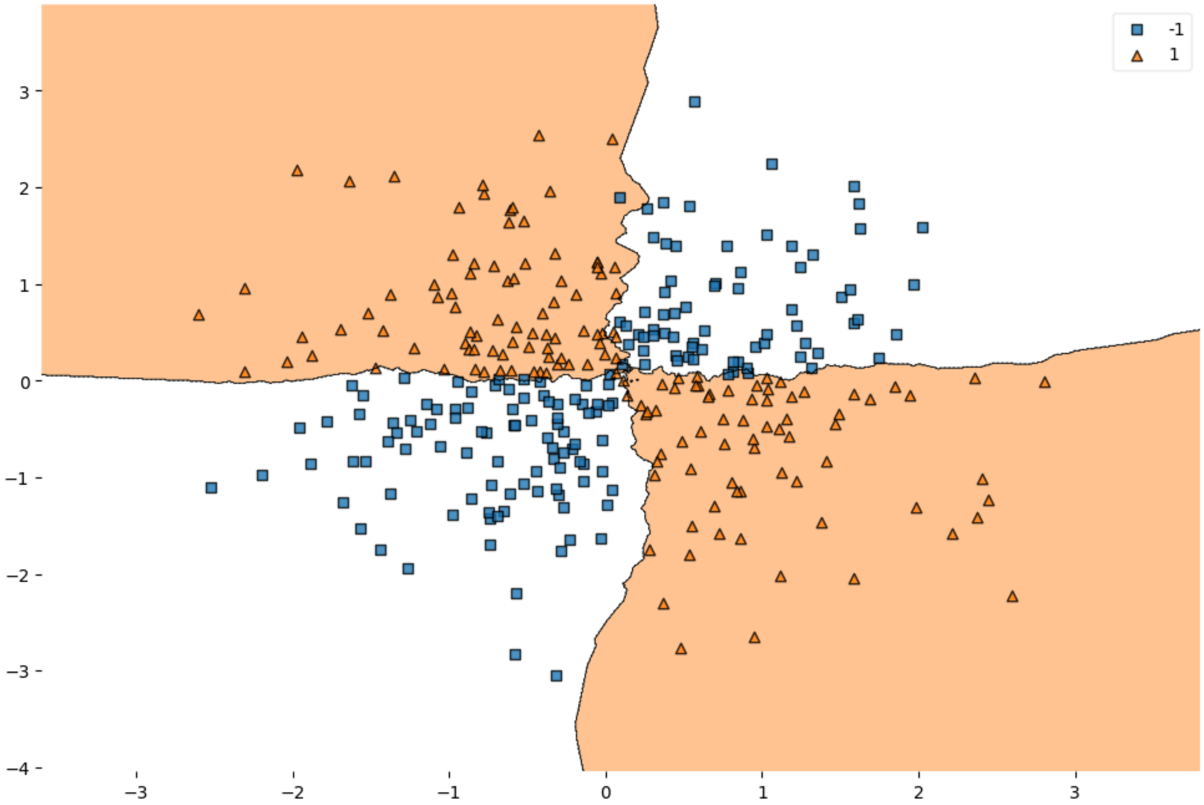
```
Accuracy for n_neighbors=13: 0.95
```



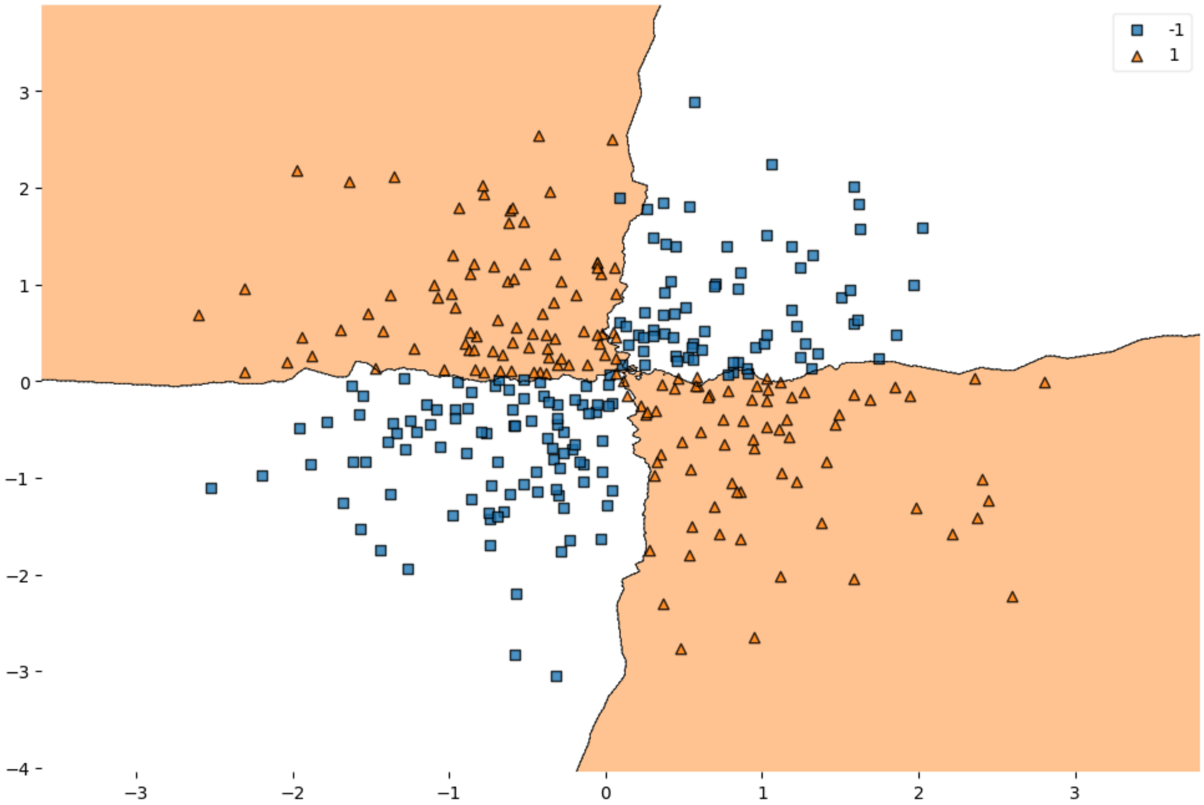
```
Accuracy for n_neighbors=15: 0.9375
```



```
Accuracy for n_neighbors=17: 0.925
```



```
Accuracy for n_neighbors=19: 0.9375
```



from the above plot at where k=3,9 I'am able to make the correct decision

8.Twospirals

In [131…

```
pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\8.twospir
```

Out[131...

	a	b	c
0	-2.543456	-10.816358	0
1	9.434466	-2.572000	0
2	3.368646	-10.194671	0
3	1.341407	-4.204140	0
4	9.547758	-2.220580	0
...	...	...	...
1995	-3.213608	1.543994	1
1996	5.577210	2.359087	1
1997	-1.393598	-7.876754	1
1998	-7.708972	-4.298002	1
1999	4.610779	10.629477	1

2000 rows × 3 columns

In [132...

```
fv=df.iloc[:,:2]
cv=df.iloc[:,-1]
```

In [133...

```
cv=cv.astype("int")
```

In [134...

```
x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,random_state=2,s
x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

In [135...

```
std=StandardScaler()
px_trainf=std.fit_transform(x_trainf)
px_test=std.transform(x_test)
px_cv=std.transform(x_cv)
```

In [136...

```
knn=KNeighborsClassifier(n_neighbors=1)
model=knn.fit(px_trainf,y_trainf)
predicted=model.predict(px_cv)
```

In [137...

```
accuracy_score(y_cv,predicted)
```

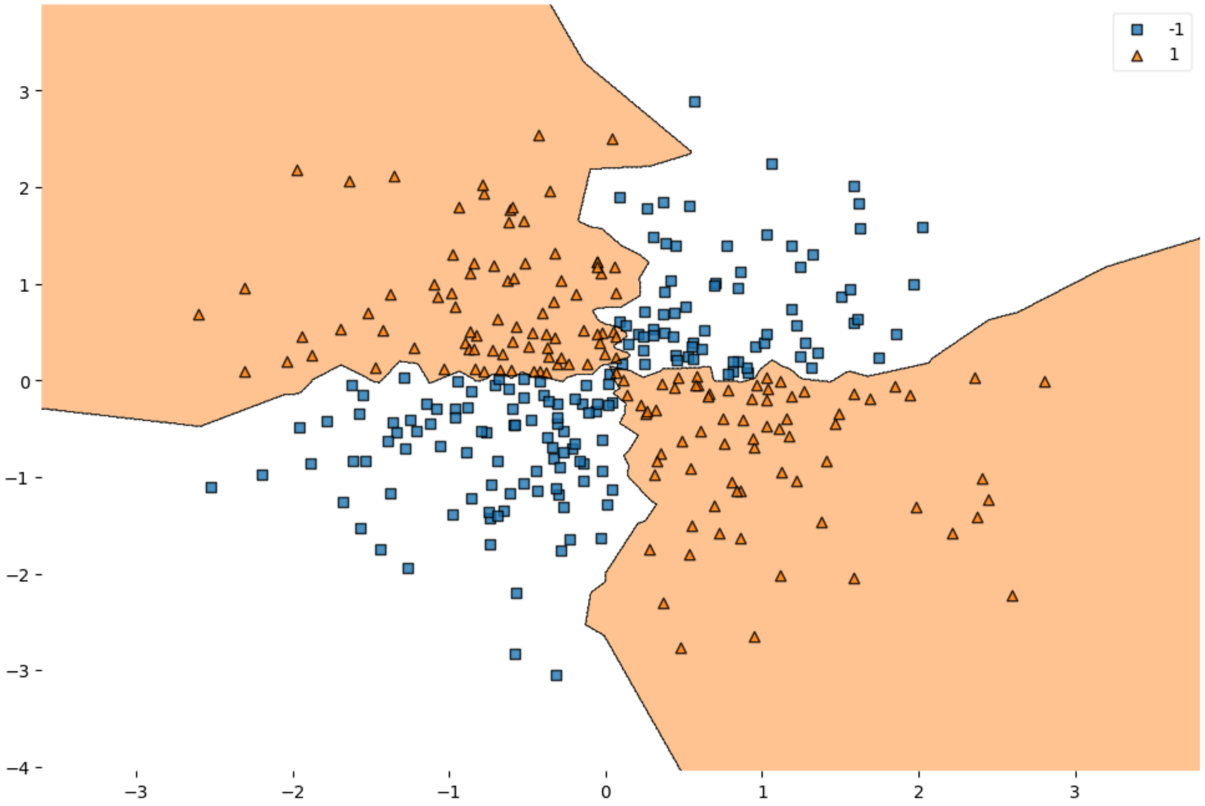
Out[137...

```
0.925
```

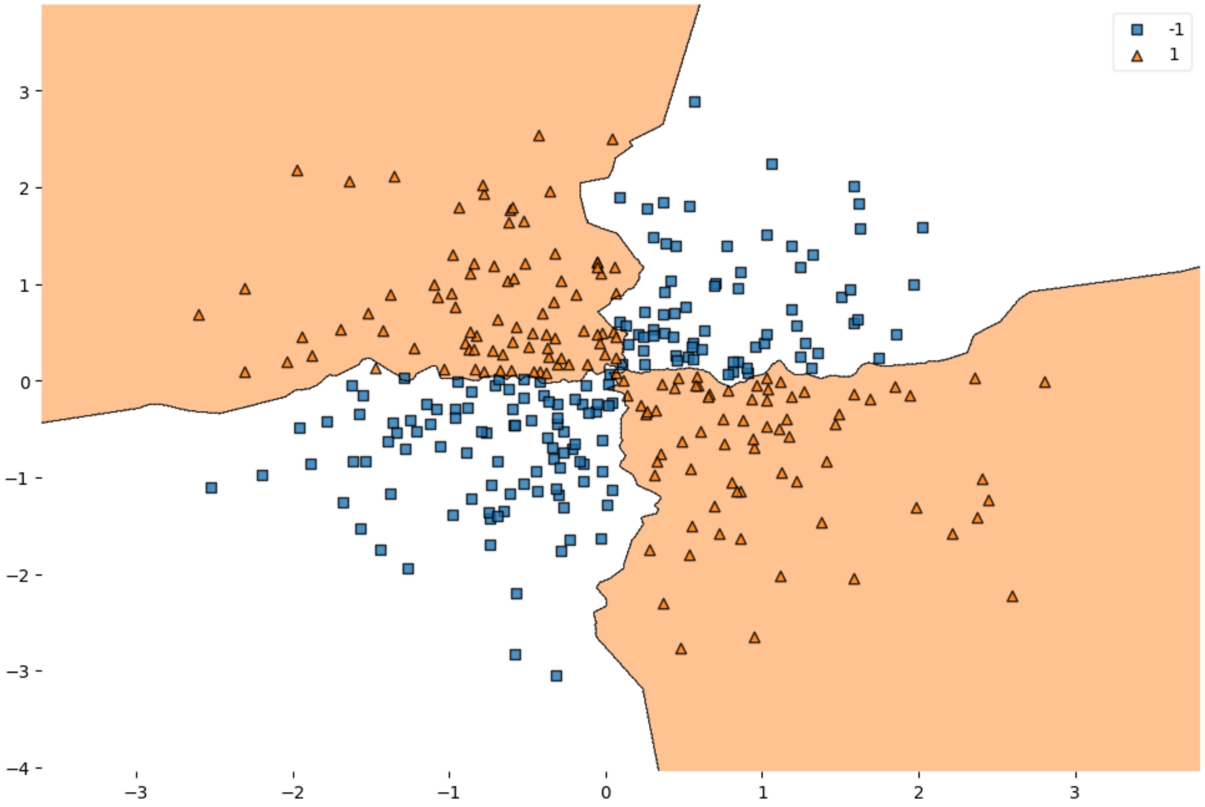
In [138...

```
for i in range(1,20,2):
    knn=KNeighborsClassifier(n_neighbors=i)
    model=knn.fit(px_trainf,y_trainf)
    predicted=model.predict(px_cv)
    print(f"Accuracy for n_neighbors={i}: {accuracy_score(y_cv,predicted)}")
    plt.figure(figsize=(12,8))
    plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
    plt.show()
```

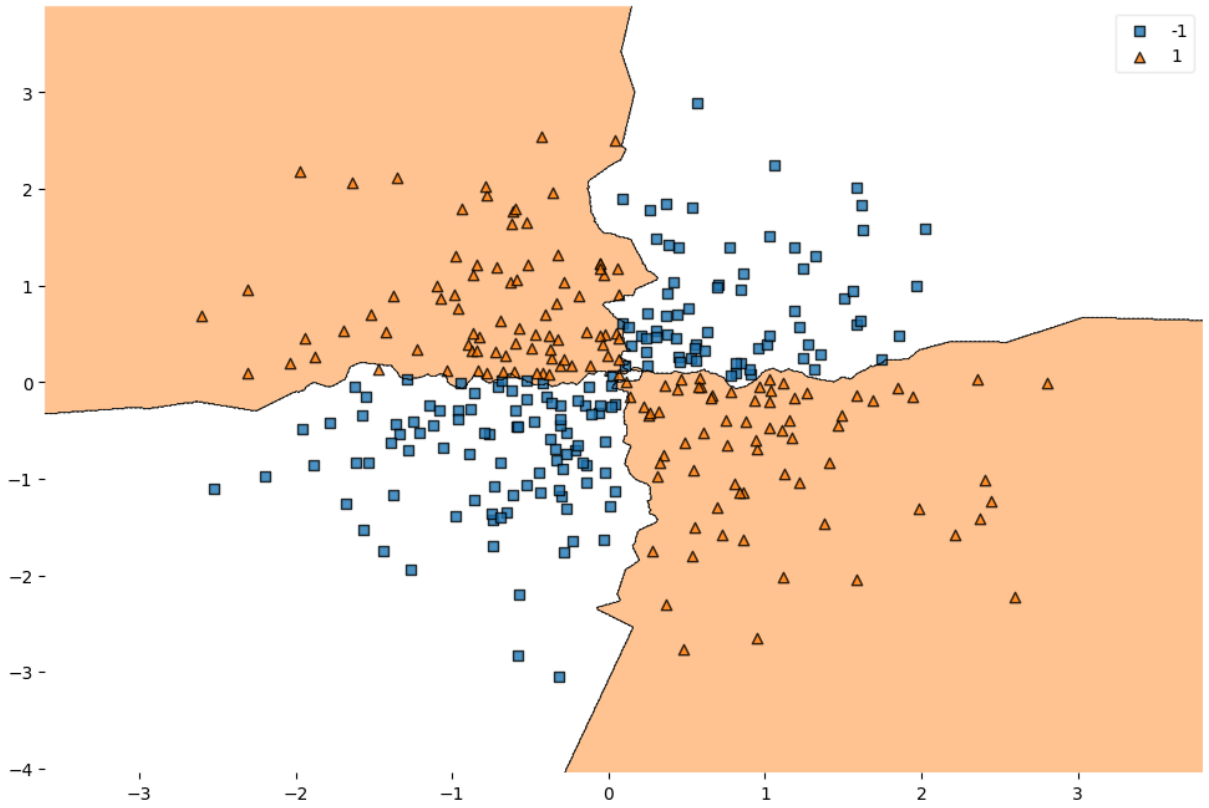
```
Accuracy for n_neighbors=1: 0.925
```



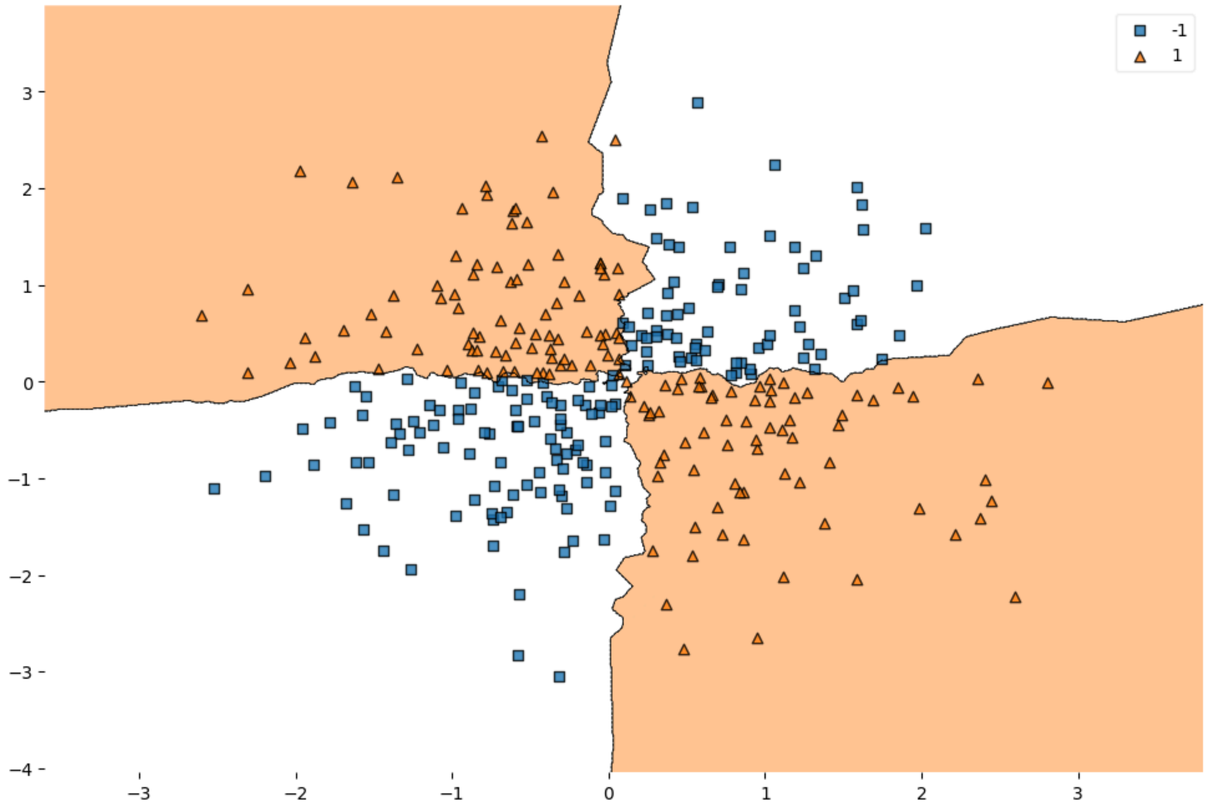
```
Accuracy for n_neighbors=3: 0.9375
```



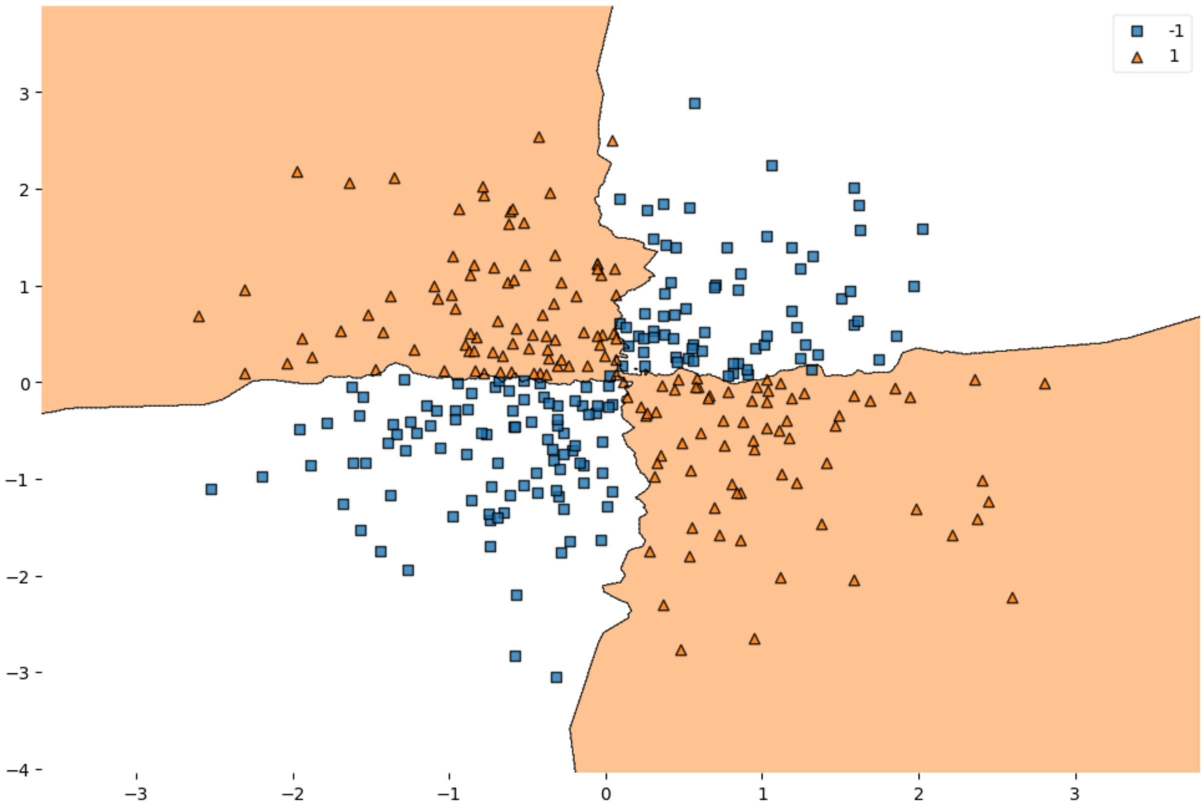
```
Accuracy for n_neighbors=5: 0.9375
```



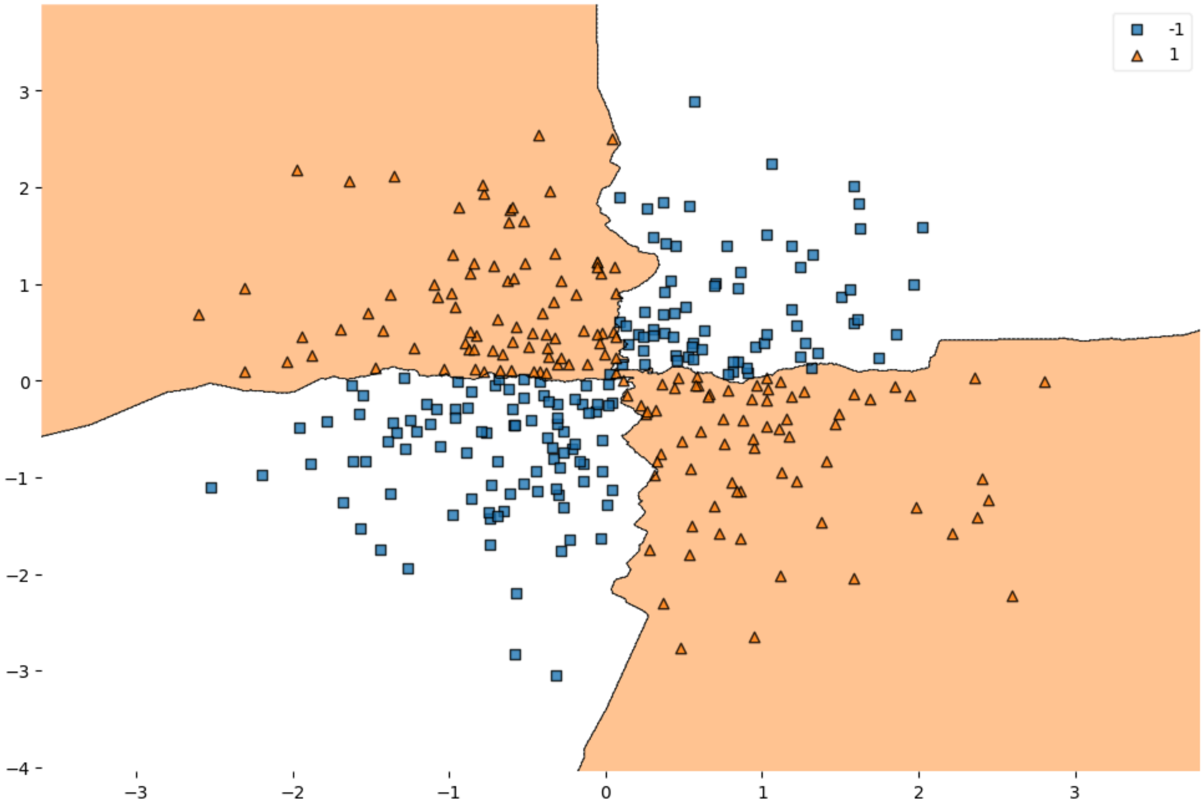
```
Accuracy for n_neighbors=7: 0.9375
```



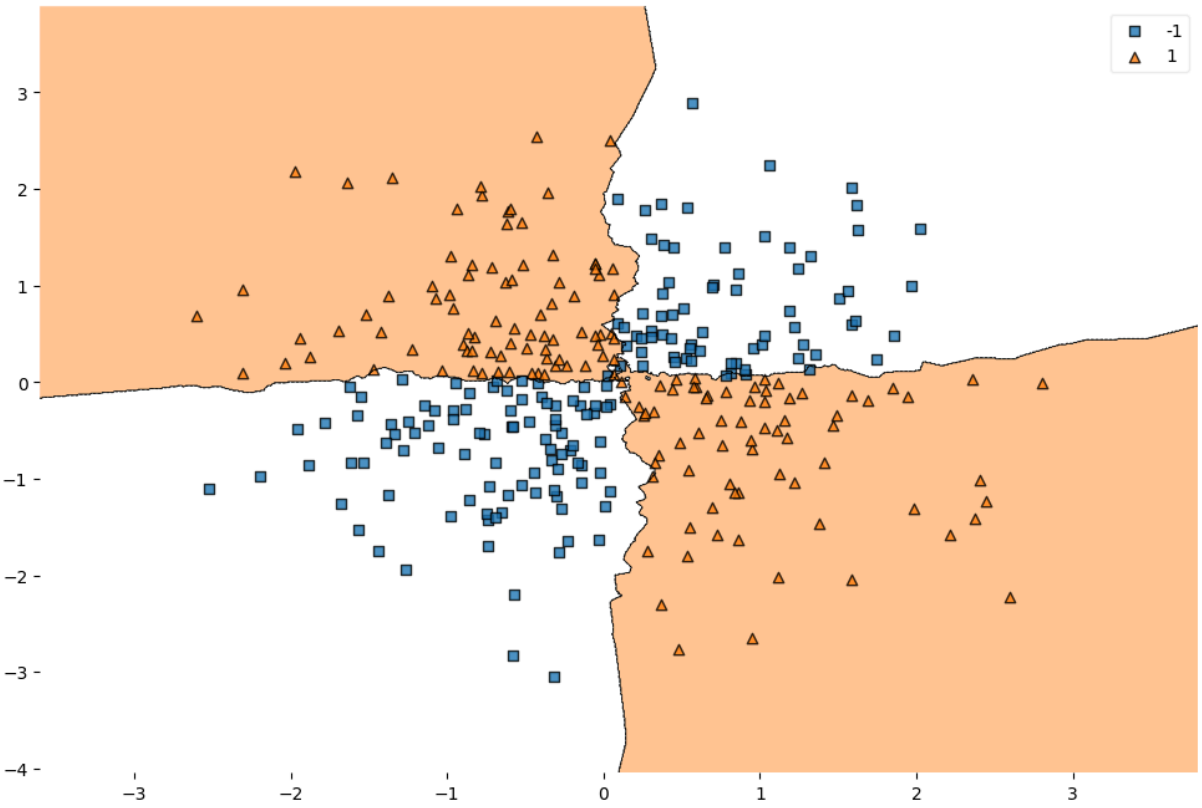
```
Accuracy for n_neighbors=9: 0.95
```



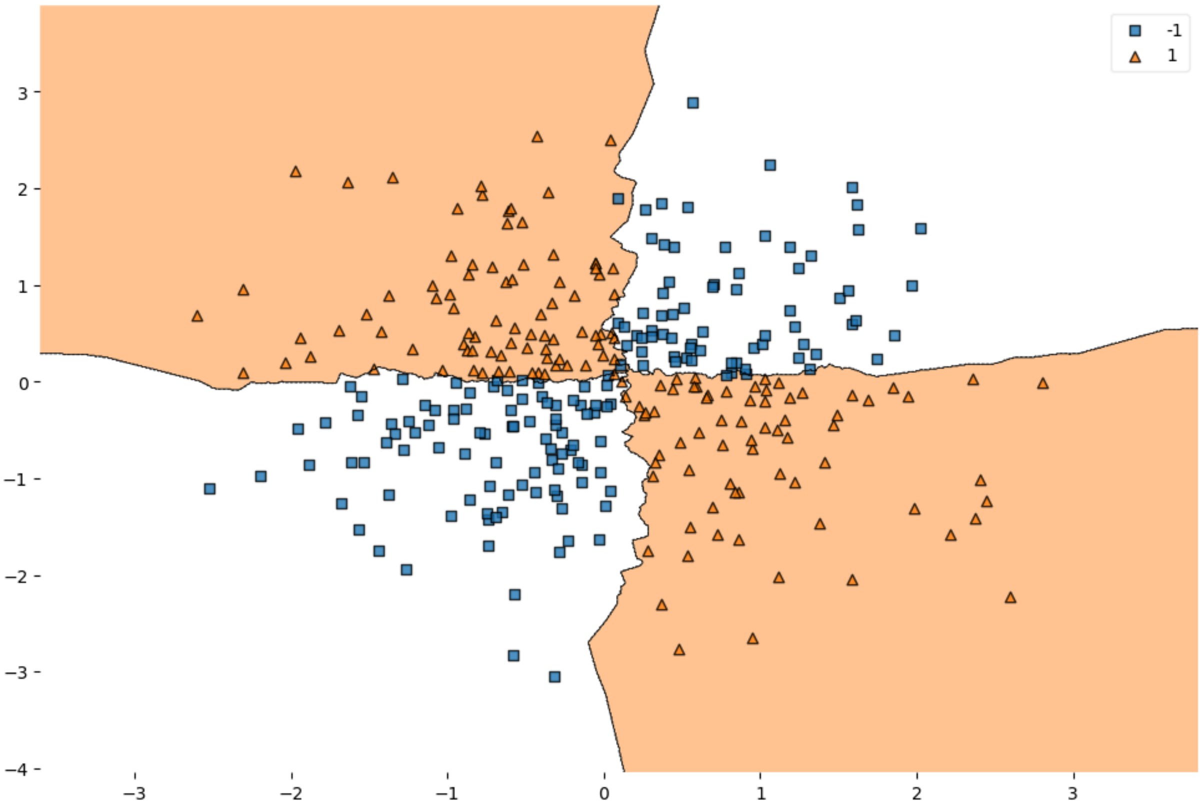
```
Accuracy for n_neighbors=11: 0.9375
```



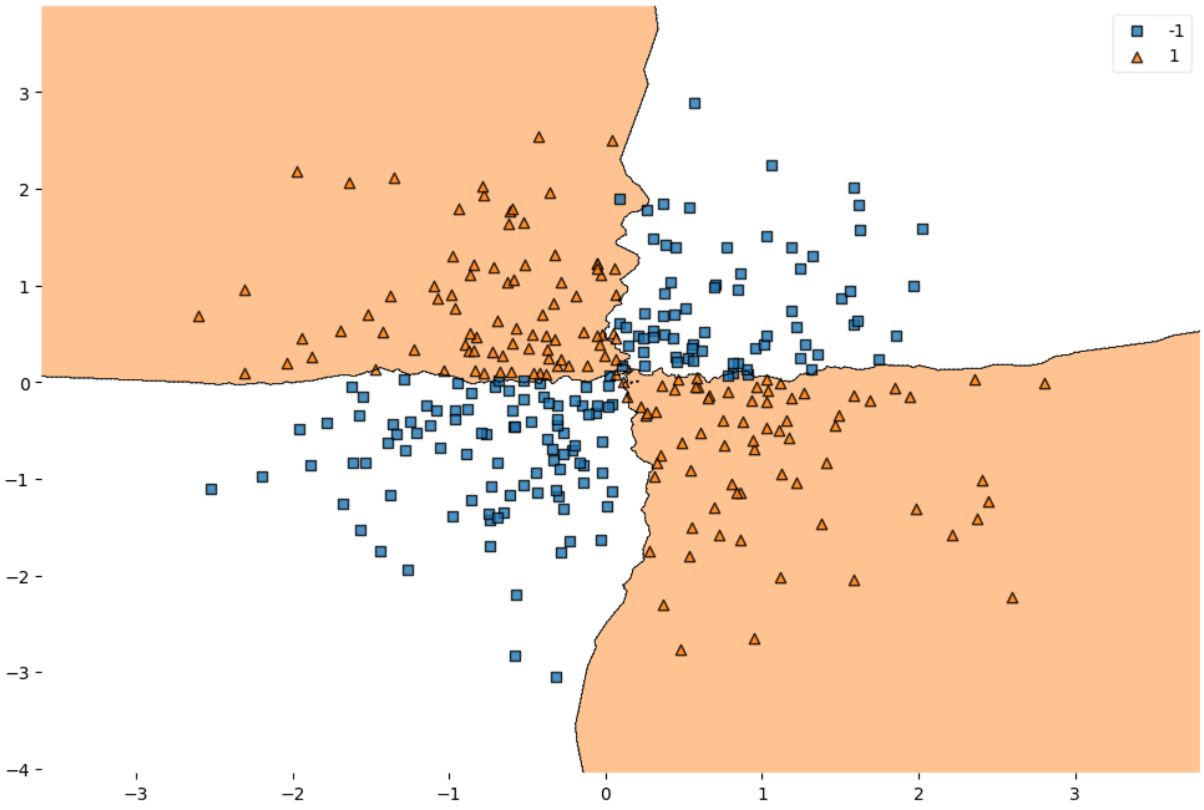
```
Accuracy for n_neighbors=13: 0.95
```



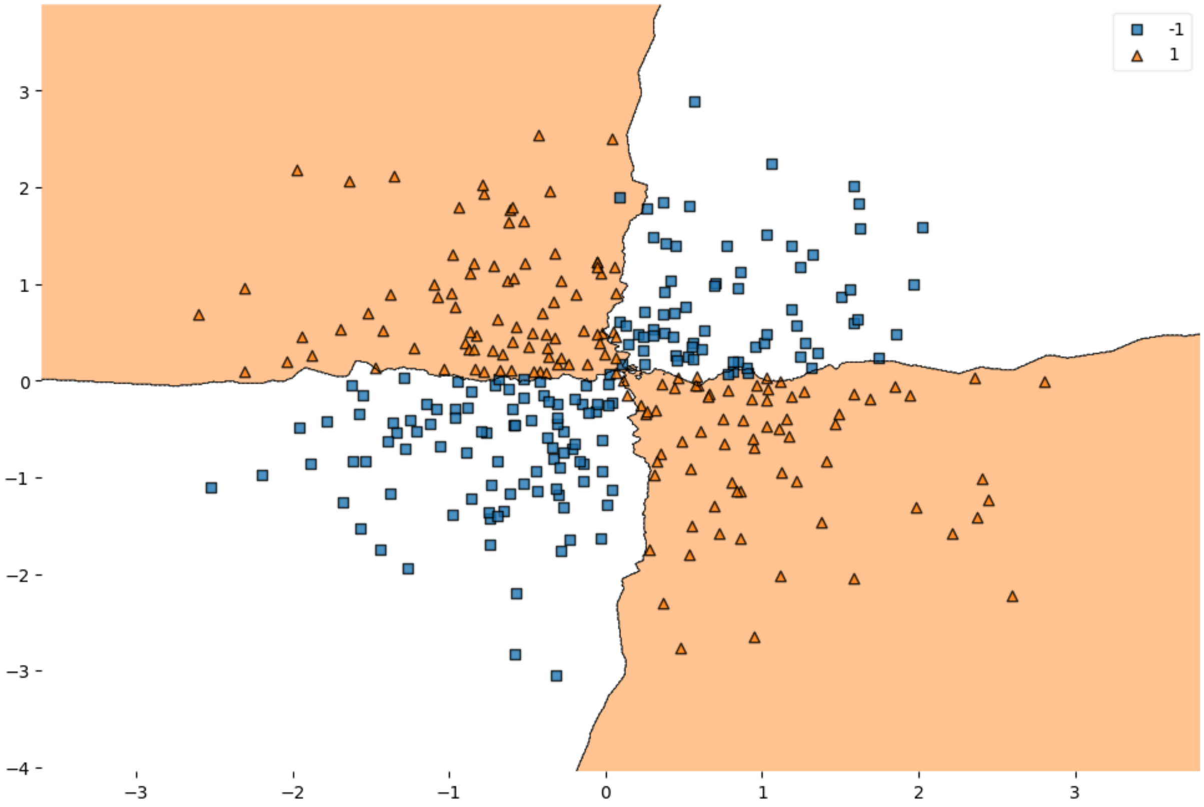
```
Accuracy for n_neighbors=15: 0.9375
```



```
Accuracy for n_neighbors=17: 0.925
```



```
Accuracy for n_neighbors=19: 0.9375
```



from the above plot at where k=any I'am able to make the correct decision

9.Random

In [142…

```
df=pd.read_csv(r"C:\Users\pavan\OneDrive\Documents\Multiple CSV\Multiple CSV\9.rand
```

In [143…

```
fv=df.iloc[:,:2]
cv=df.iloc[:,-1]
```

In [144…

```
cv=cv.astype("int")
```

In [145…

```
x_train,x_test,y_train,y_test=train_test_split(fv,cv,test_size=0.2,random_state=2,s
x_trainf,x_cv,y_trainf,y_cv=train_test_split(x_train,y_train,test_size=0.2,stratify
```

In [146…

```
std=StandardScaler()
px_trainf=std.fit_transform(x_trainf)
px_test=std.transform(x_test)
px_cv=std.transform(x_cv)
```

In [147…

```
knn=KNeighborsClassifier(n_neighbors=1)
model=knn.fit(px_trainf,y_trainf)
predicted=model.predict(px_cv)
```

In [148…

```
accuracy_score(y_cv,predicted)
```

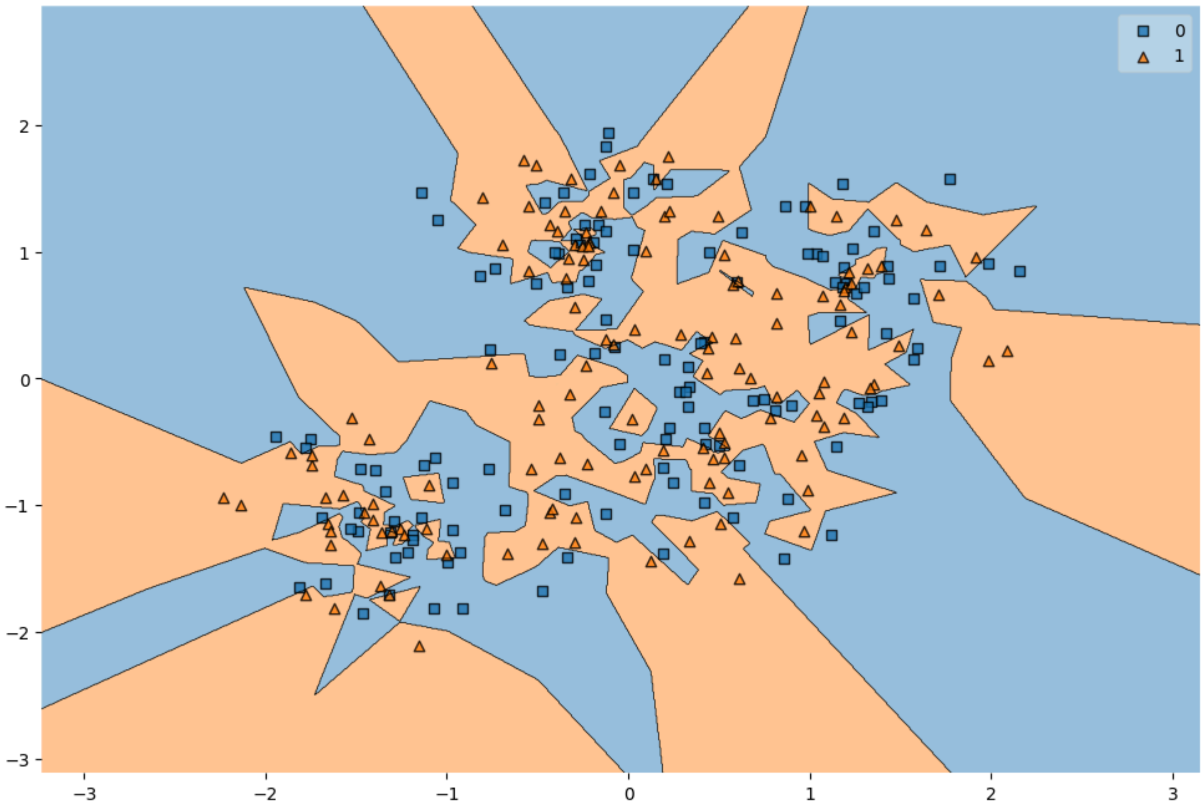
Out[148…

```
0.53125
```

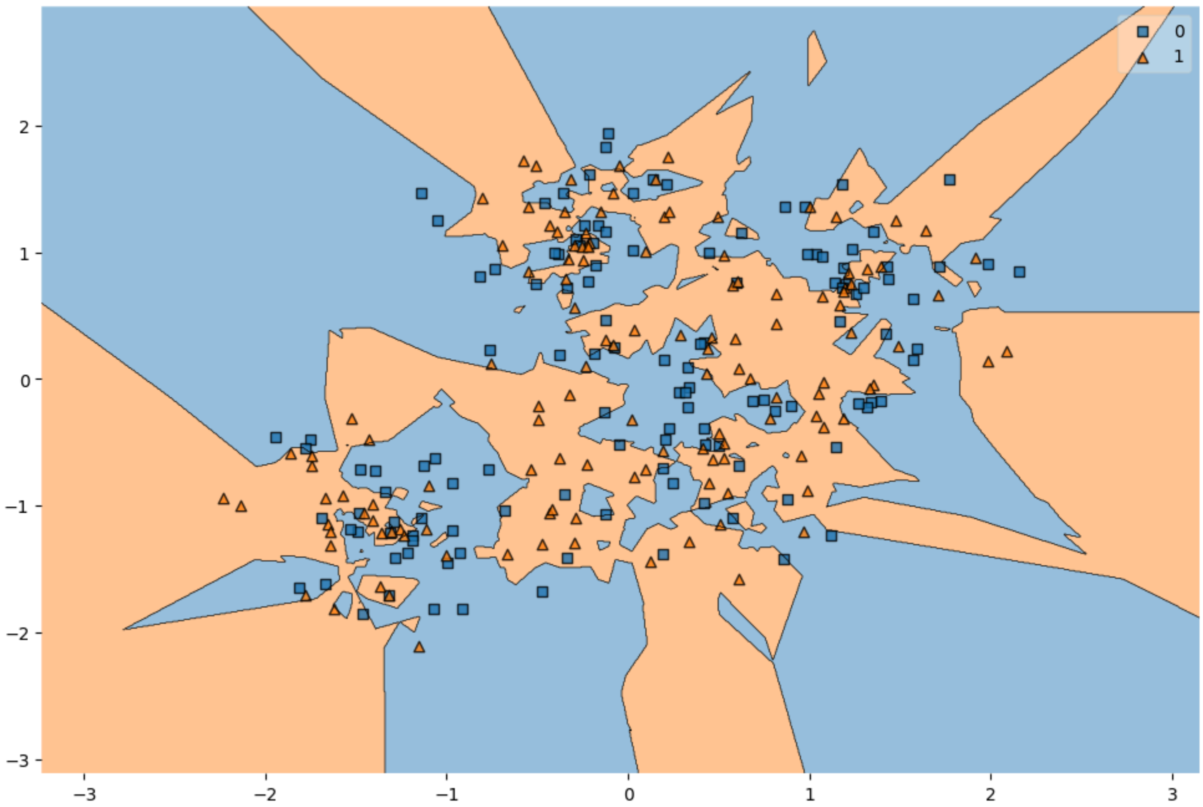
In [149…

```
for i in range(1,20,2):
    knn=KNeighborsClassifier(n_neighbors=i)
    model=knn.fit(px_trainf,y_trainf)
    predicted=model.predict(px_cv)
    print(f"Accuracy for n_neighbors={i}: {accuracy_score(y_cv,predicted)}")
    plt.figure(figsize=(12,8))
    plot_decision_regions(X=px_trainf,y=y_trainf.values,clf=knn)
    plt.show()
```

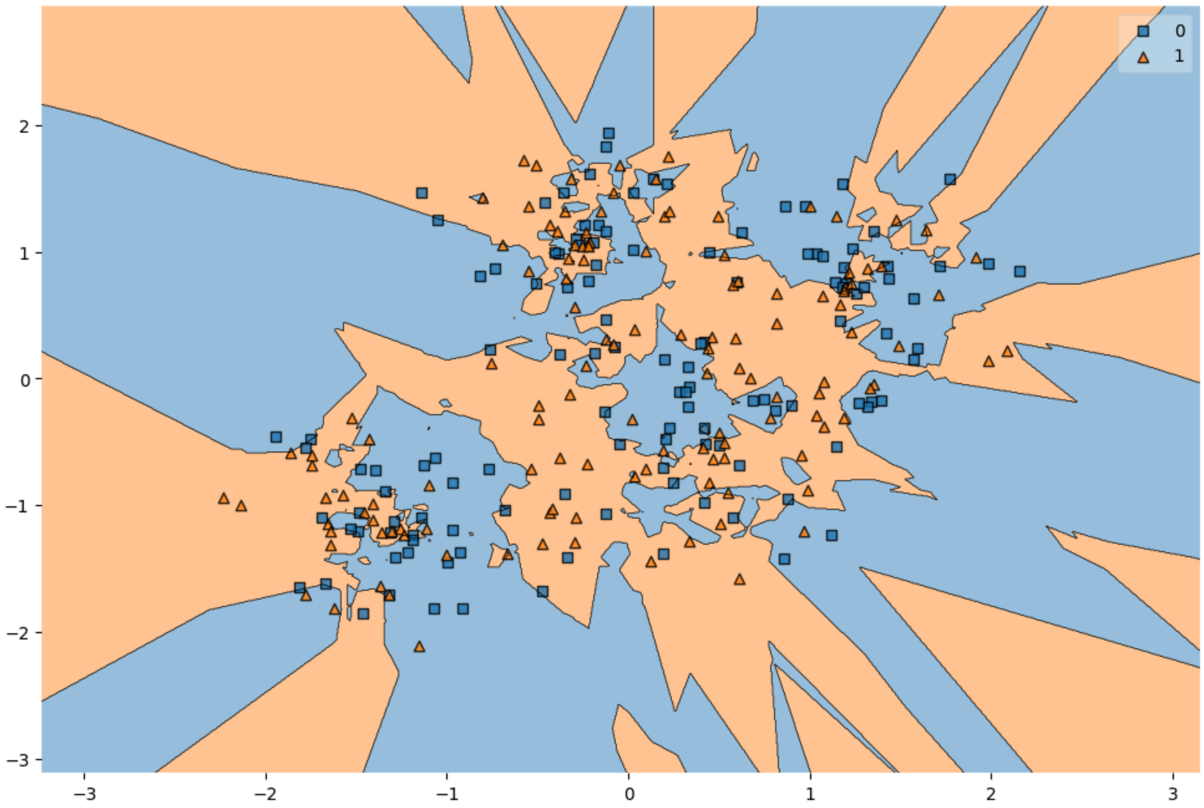
```
Accuracy for n_neighbors=1: 0.53125
```



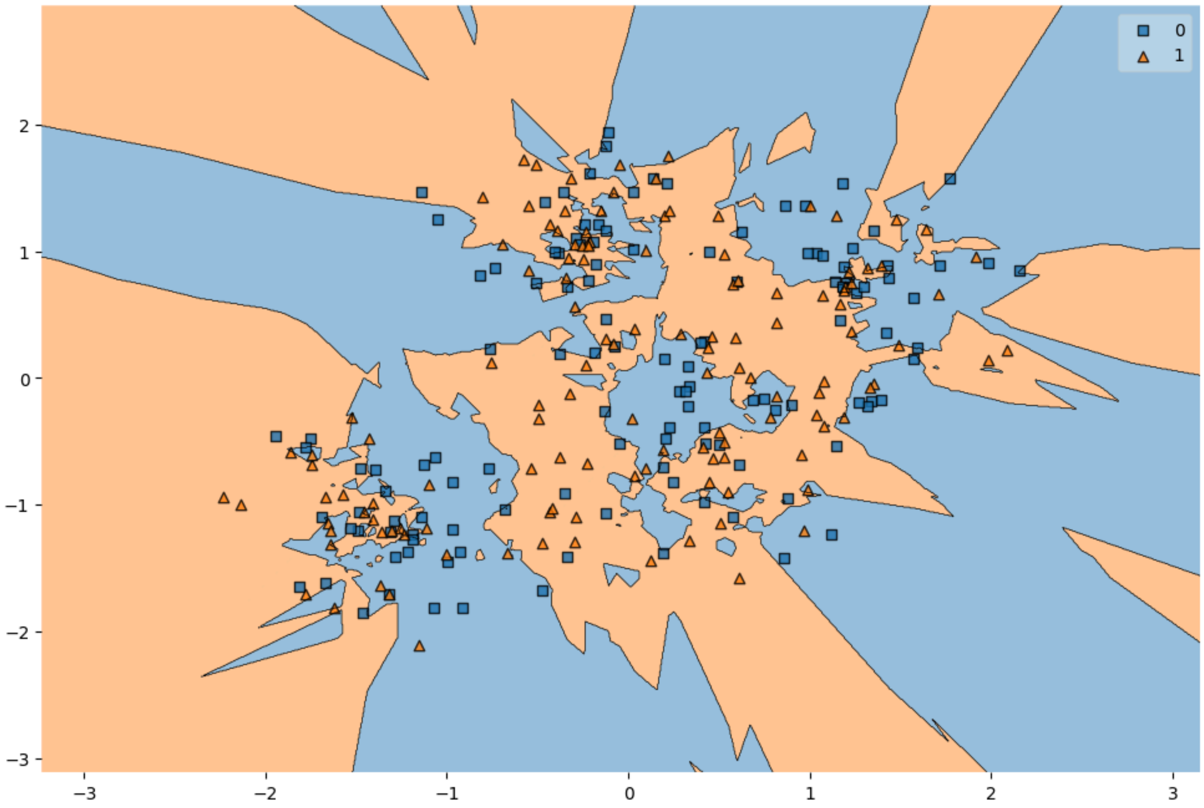
```
Accuracy for n_neighbors=3: 0.546875
```



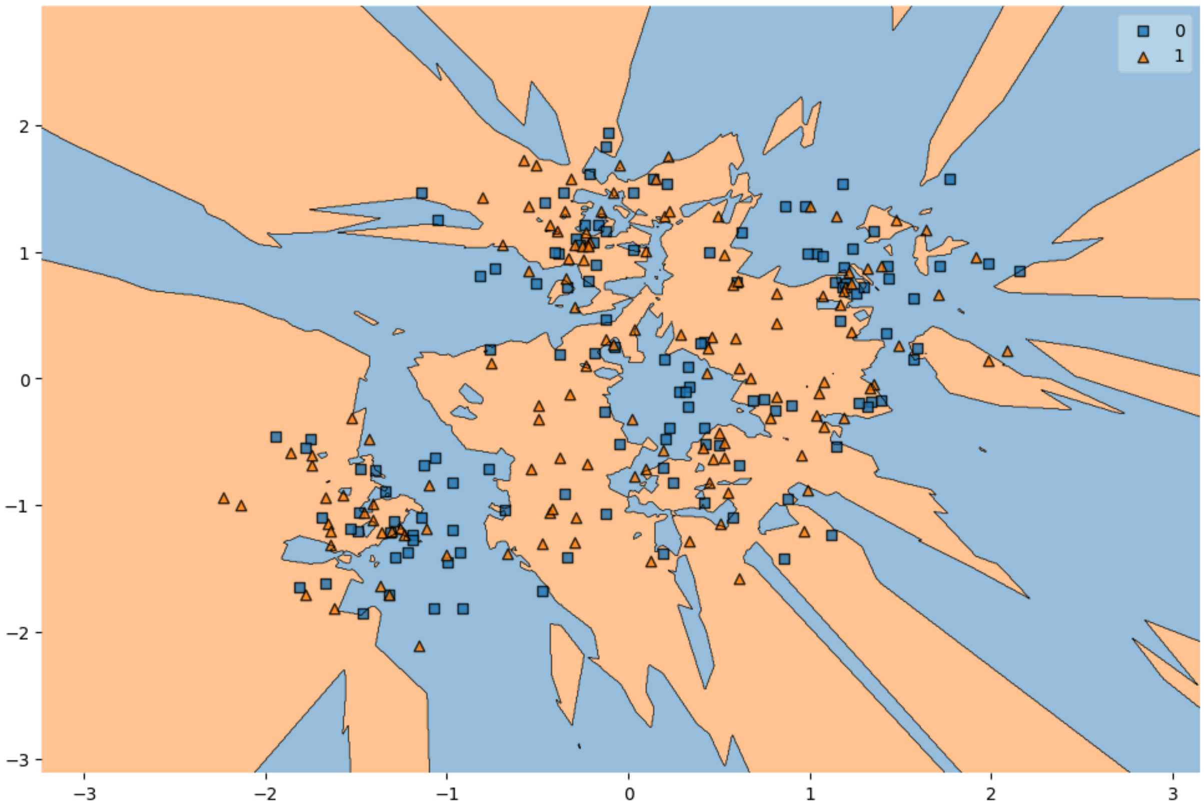
```
Accuracy for n_neighbors=5: 0.578125
```



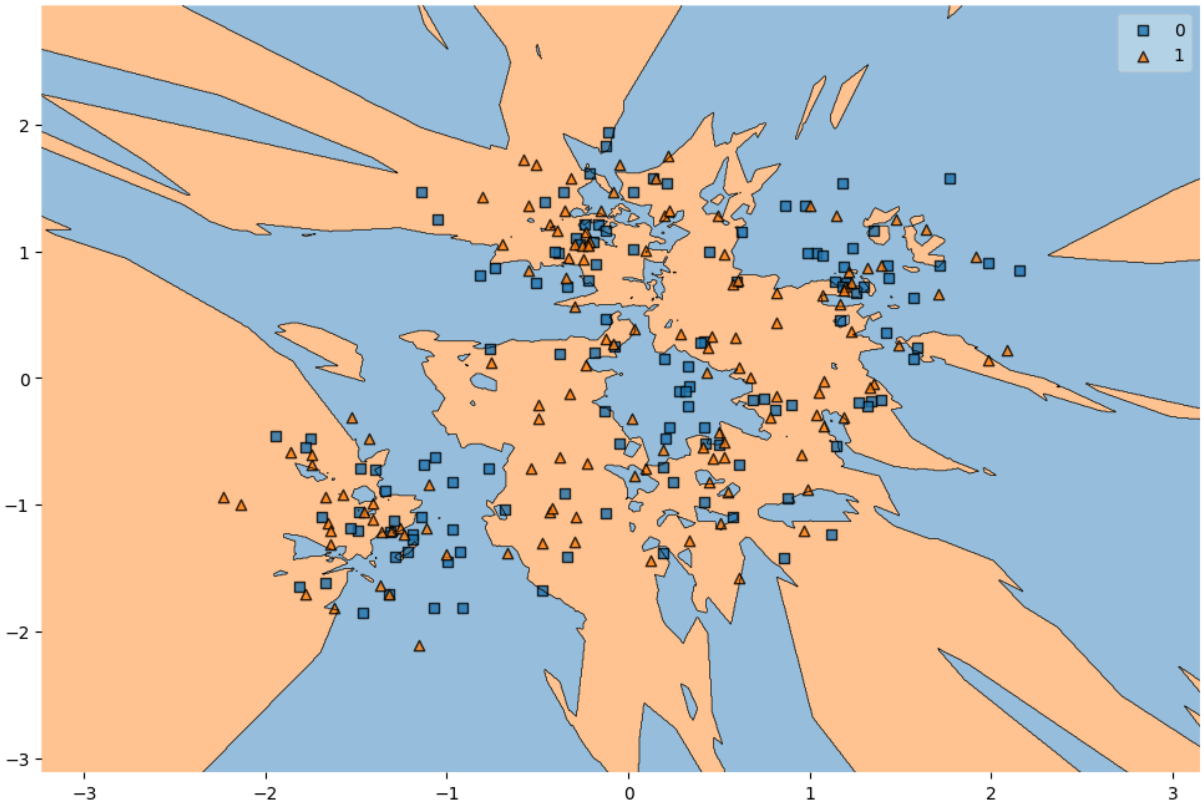
```
Accuracy for n_neighbors=7: 0.59375
```



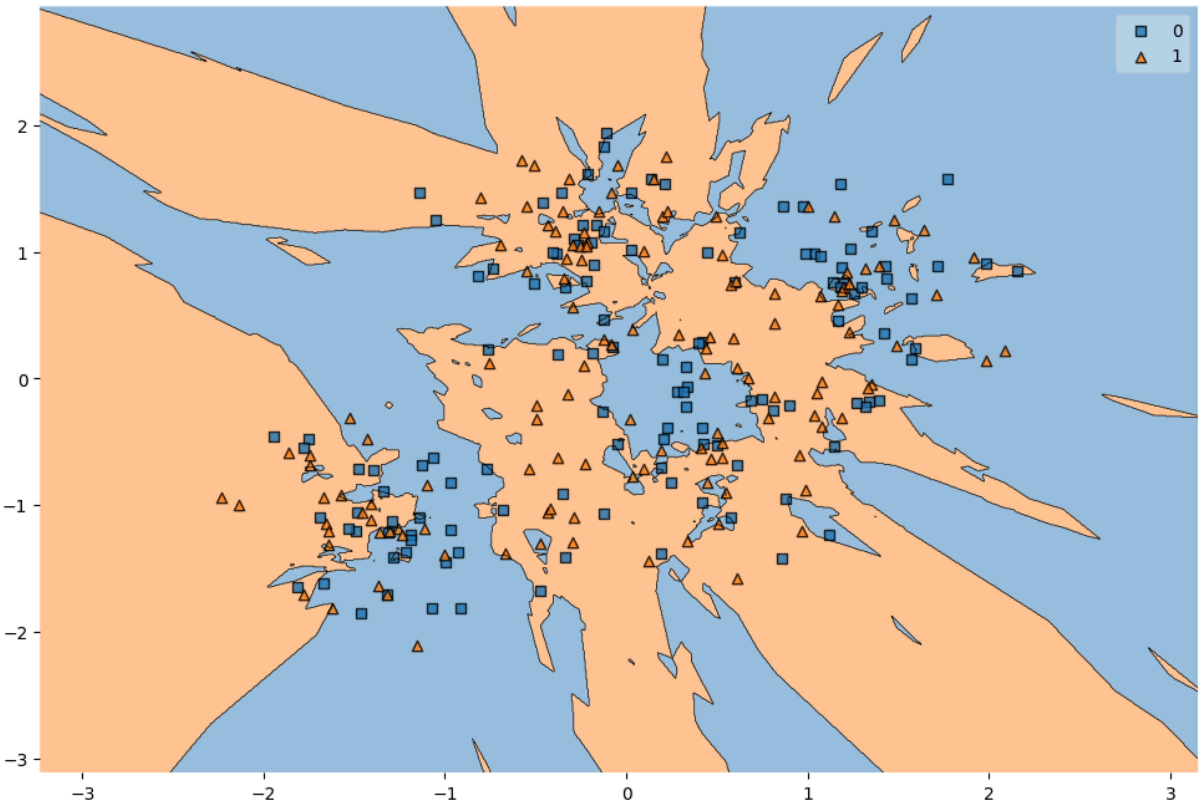
```
Accuracy for n_neighbors=9: 0.578125
```



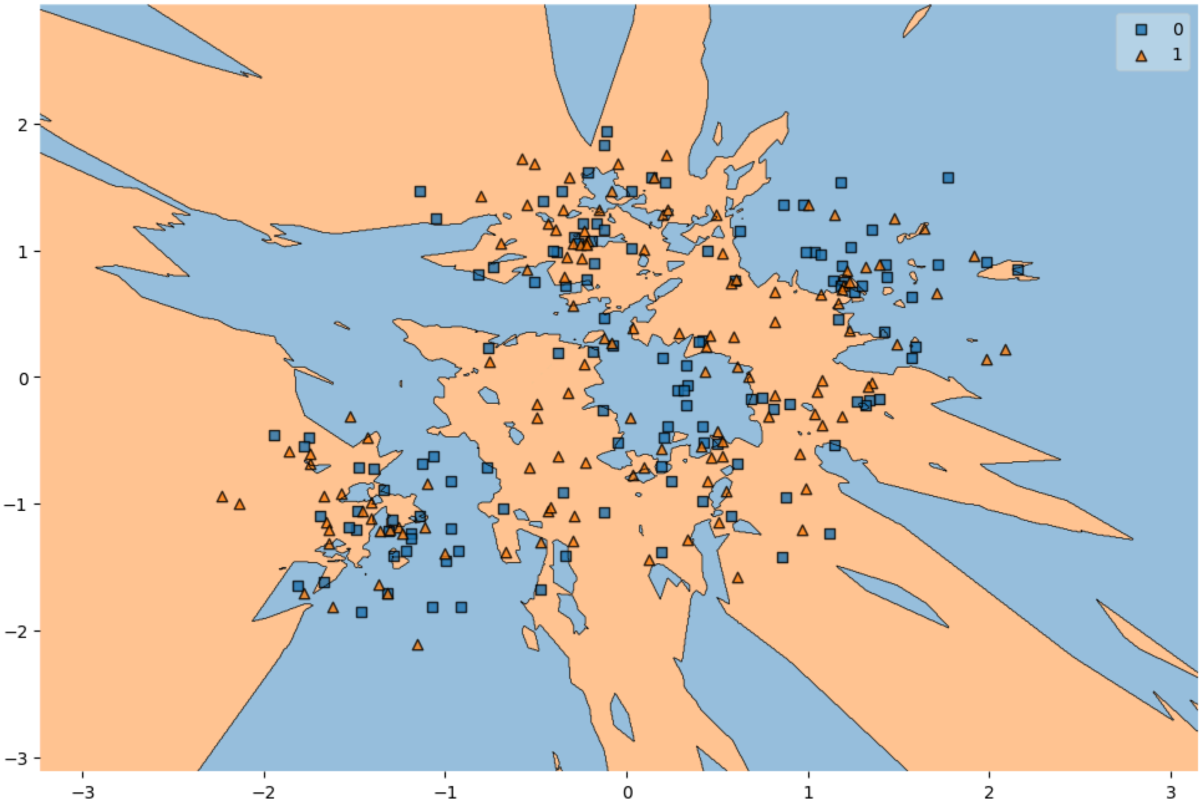
```
Accuracy for n_neighbors=11: 0.515625
```



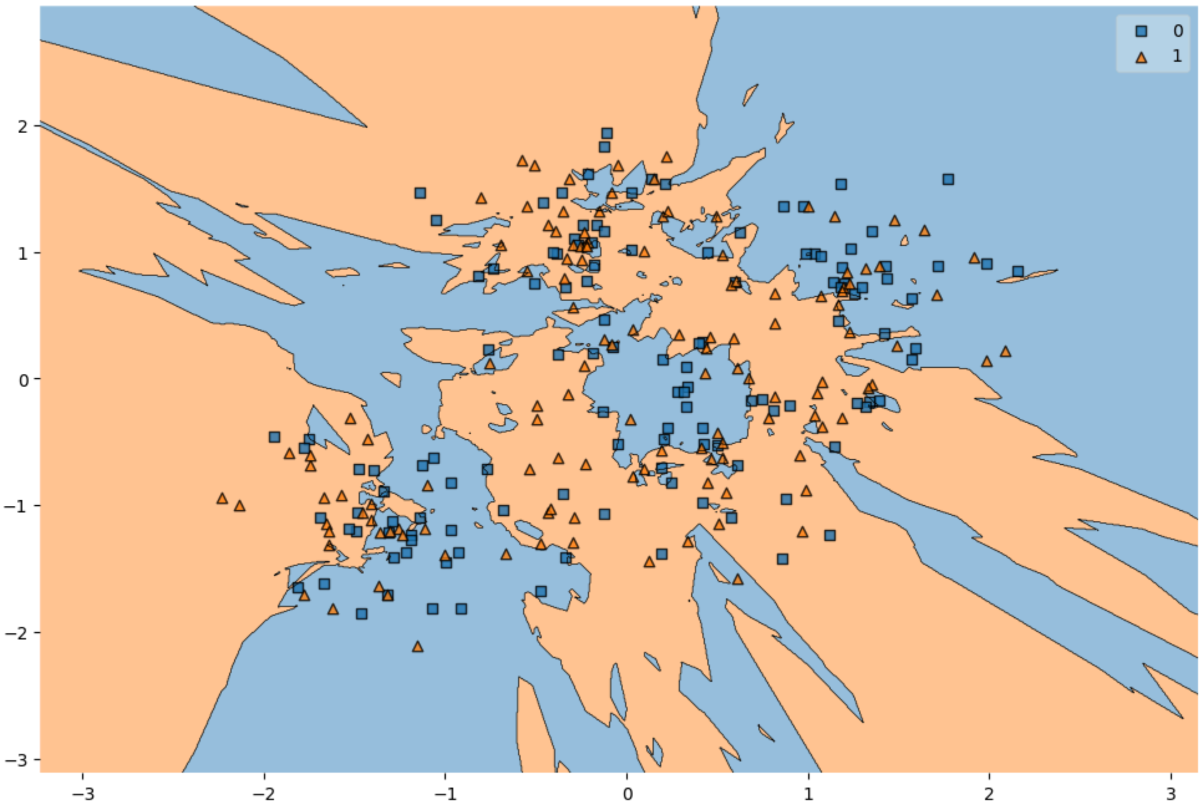
```
Accuracy for n_neighbors=13: 0.546875
```



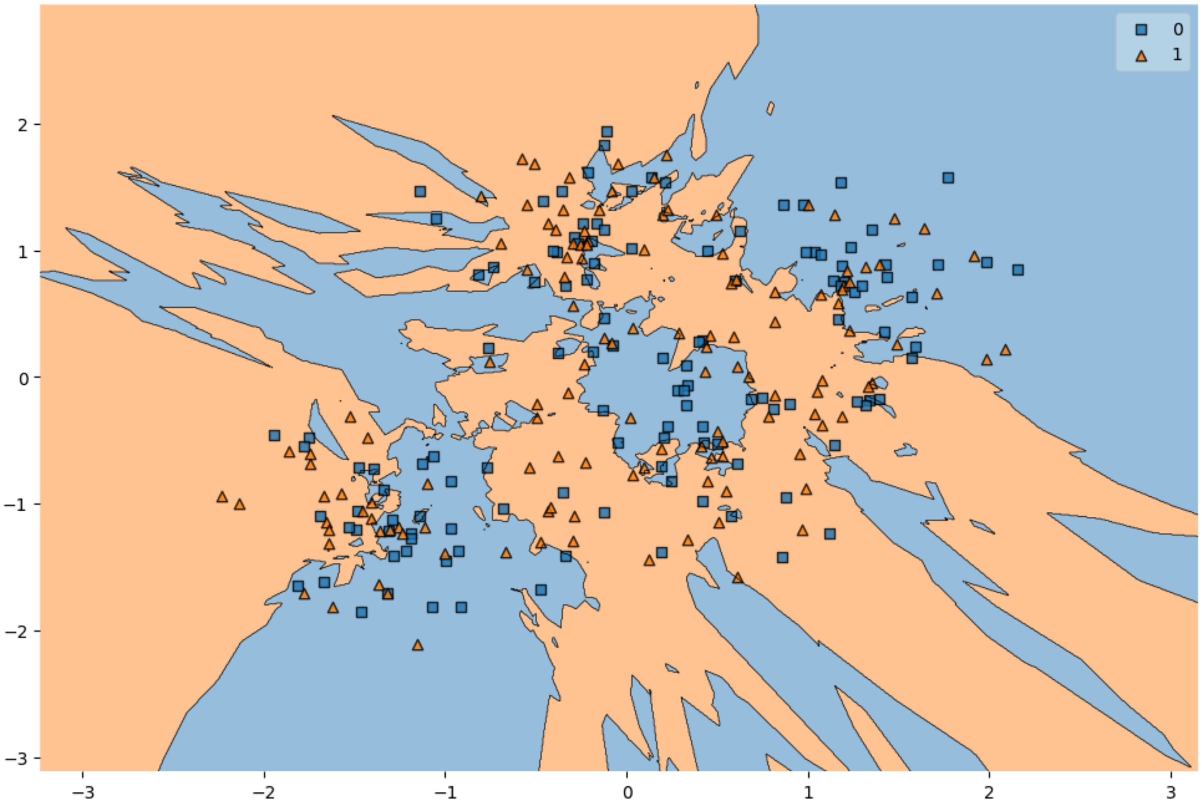
```
Accuracy for n_neighbors=15: 0.515625
```



```
Accuracy for n_neighbors=17: 0.53125
```



```
Accuracy for n_neighbors=19: 0.515625
```



from the above plot at where k=7 I'am able to make the correct decision

In [ ]:

In [5]:

In [ ]: