In [21]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from seaborn import load_dataset
from sklearn.linear_model import LogisticRegression,LinearRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score,confusion_matrix
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
```

In [6]:
```
df = sns.load_dataset('iris')
```

In [8]:
```
df.head()
```

Out[8]:

	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 [10]:
```
encoder = LabelEncoder()
df['species'] = encoder.fit_transform(df['species'])
```

In [11]:
```
df.head()
```

Out[11]:

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

In [15]:
```
df1 = df[['sepal_length','sepal_width','species']]
```

In [17]:
```
df1
```

Out[17]:

	sepal_length	sepal_width	species
0	5.1	3.5	0
1	4.9	3.0	0
2	4.7	3.2	0
3	4.6	3.1	0
4	5.0	3.6	0
...	...	...	...
145	6.7	3.0	2
146	6.3	2.5	2
147	6.5	3.0	2
148	6.2	3.4	2
149	5.9	3.0	2

150 rows × 3 columns

In [18]:
```
x = df1.iloc[:,0:2]
y = df.iloc[:,-1]
```

In [19]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
```

In [24]:
```
clf = LogisticRegression(multi_class='multinomial')
```

In [25]:
```
clf.fit(x_train,y_train)
```

Out[25]:
```
LogisticRegression(multi_class='multinomial')
```

In [26]:
```
y_pred = clf.predict(x_test)
```

In [27]:
```
print(accuracy_score(y_test,y_pred))
```

```
0.8666666666666667
```

In [29]:
```
pd.DataFrame(confusion_matrix(y_test,y_pred))
```

Out[29]:

	0	1	2
0	9	0	0
1	0	10	3
2	0	1	7

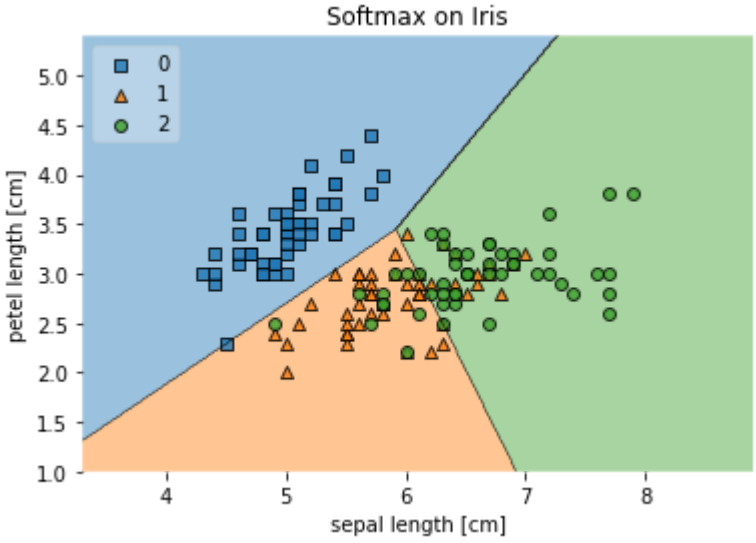
In [30]:
```
query = np.array([[3.4,2.7]])
clf.predict_proba(query)
```

Out[30]:
```
array([[9.91895784e-01, 7.97898898e-03, 1.25227481e-04]])
```

In [31]:
```
clf.predict(query)
```

Out[31]:
```
array([0])
```

In [34]:
```
from mlxtend.plotting import plot_decision_regions
plot_decision_regions(x.values,y.values,clf,legend=2)
plt.xlabel('sepal length [cm]')
plt.ylabel('petel length [cm]')
plt.title('Softmax on Iris')
plt.show()
```



In [ ]: