Week_06_Quiz-qm2162

October 24, 2021

1 Week 6 Quiz

1.1 Qi Meng - qm2162

1.1.1 Due Sunday Oct 24 11:59pm ET

In this quiz we'll be loading some data, training a few models and plotting their decision boundaries to compare how the models perform.

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn import datasets
from mlxtend.plotting import plot_decision_regions

sns.set_style('darkgrid')
%matplotlib inline
```

[2]:
```
# For this quiz we'll be using the "iris" dataset, a very common dataset seen
# when learning about classification.

# Using sklearn.datasets, load the dataset into iris
iris = datasets.load_iris()
```

[3]:
```
# As we've seen in other datasets loaded from sklearn, the data is contained in
# iris.data and feature names in iris.feature_names.
# Create a new pd.DataFrame X_iris that contains iris.data
#    with columns=iris.feature_names
X_iris = pd.DataFrame(iris.data, columns=iris.feature_names)

# In order to make the data easier to plot, keep only the first 2 features:
#   "sepal length (cm)","sepal width (cm)"
# Store back into X_iris
X_iris = X_iris.iloc[:,:2]

# We'll check to make sure that X_iris only has 2 columns
assert X_iris.shape == (150,2)
```

```
# The target/label is contained in iris.target.
# Create a pd.Series y_iris containing iris.target.
y_iris = pd.Series(iris.target)

# Print out the number of rows per label.
# Note that this is a multiclass problem (3 classes)
#   each with 50 observations
y_iris.value_counts()
```

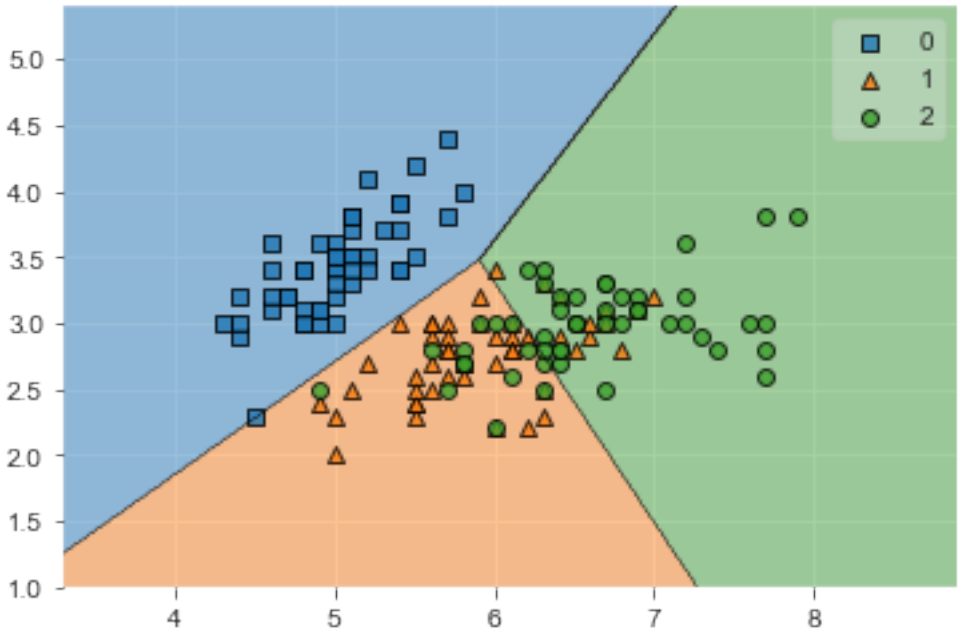
[3]:
```
0    50
1    50
2    50
dtype: int64
```

[4]:
```
# Import LogisticRegression from sklearn.linear_model
from sklearn.linear_model import LogisticRegression

# Instantiate the LogisticRegression model with default settings
#   and fit on X_iris and y_iris
# Store in logreg
logreg = LogisticRegression()
logreg.fit(X_iris, y_iris)

# Plot the training set and trained classifier with
#   plot_decision_regions() which takes numpy arrays
#   so use X_iris.values, y_iris.values
plot_decision_regions(X_iris.values, y_iris.values, logreg)
```

[4]:
```
<AxesSubplot:>
```

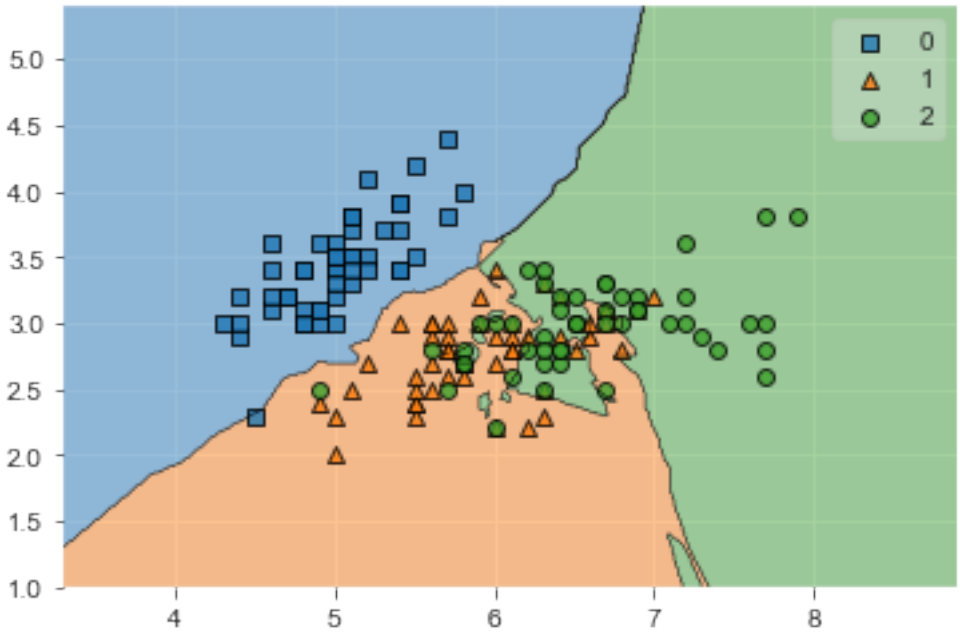


[5]:
```
# Import KNeighborsClassifier from sklearn.neighbors
from sklearn.neighbors import KNeighborsClassifier

# Instantiate the KNeighborsClassifier model with default settings
#   and fit on X_iris and y_iris
# Store in knn
knn = KNeighborsClassifier()
knn.fit(X_iris, y_iris)

# Plot the training set and trained classifier with plot_decision_regions()
plot_decision_regions(X_iris.values, y_iris.values, knn)
```

[5]: <AxesSubplot:>

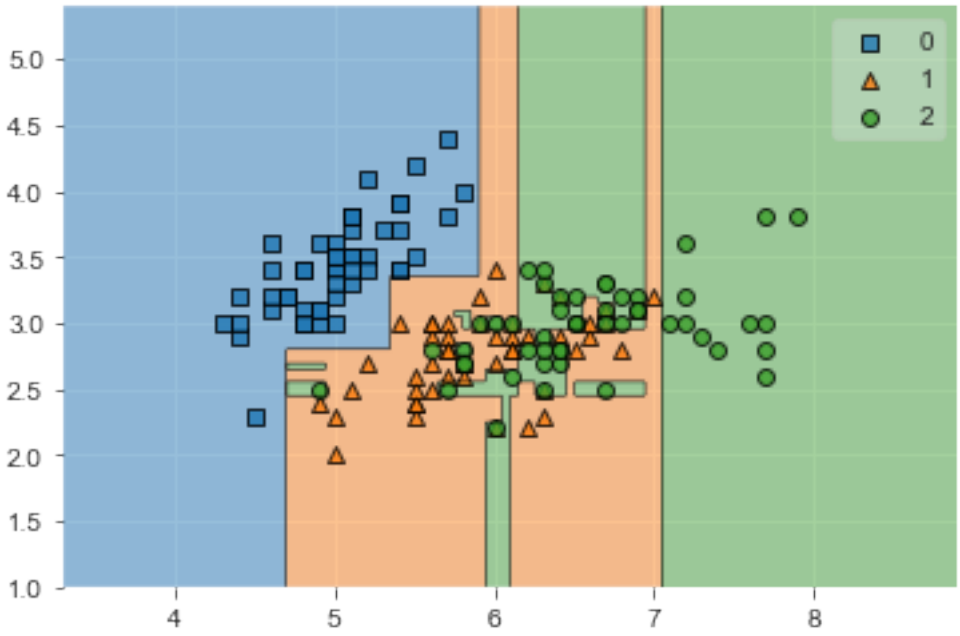


[6]:
```
# Import GradientBoostingClassifier from sklearn.ensemble
from sklearn.ensemble import GradientBoostingClassifier

# Instantiate the GradientBoostingClassifier with default settings
#   and fit on X_iris and y_iris
# Store in gbc
gbc = GradientBoostingClassifier()
gbc.fit(X_iris, y_iris)

# Plot the training set and trained classifier with plot_decision_regions()
plot_decision_regions(X_iris.values, y_iris.values, gbc)
```

[6]: <AxesSubplot:>


0
1
2
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
4
5
6
7
8