04_model_training

February 28, 2024

1 Training a Machine Learning model with scikit-learn

Lesson 4 from Introduction to Machine Learning with scikit-learn

Note: This notebook uses Python 3.9.1 and scikit-learn 0.23.2. The original notebook (shown in the video) used Python 2.7 and scikit-learn 0.16.

1.1 Agenda

- What is the **K-nearest neighbors** classification model?
- What are the four steps for **model training and prediction** in scikit-learn?
- How can I apply this pattern to **other Machine Learning models**?

1.2 Reviewing the iris dataset

[1]:
```
# added empty cell so that the cell numbering matches the video
```

[2]:
```
from IPython.display import IFrame
IFrame('https://www.dataschool.io/files/iris.txt', width=300, height=200)
```

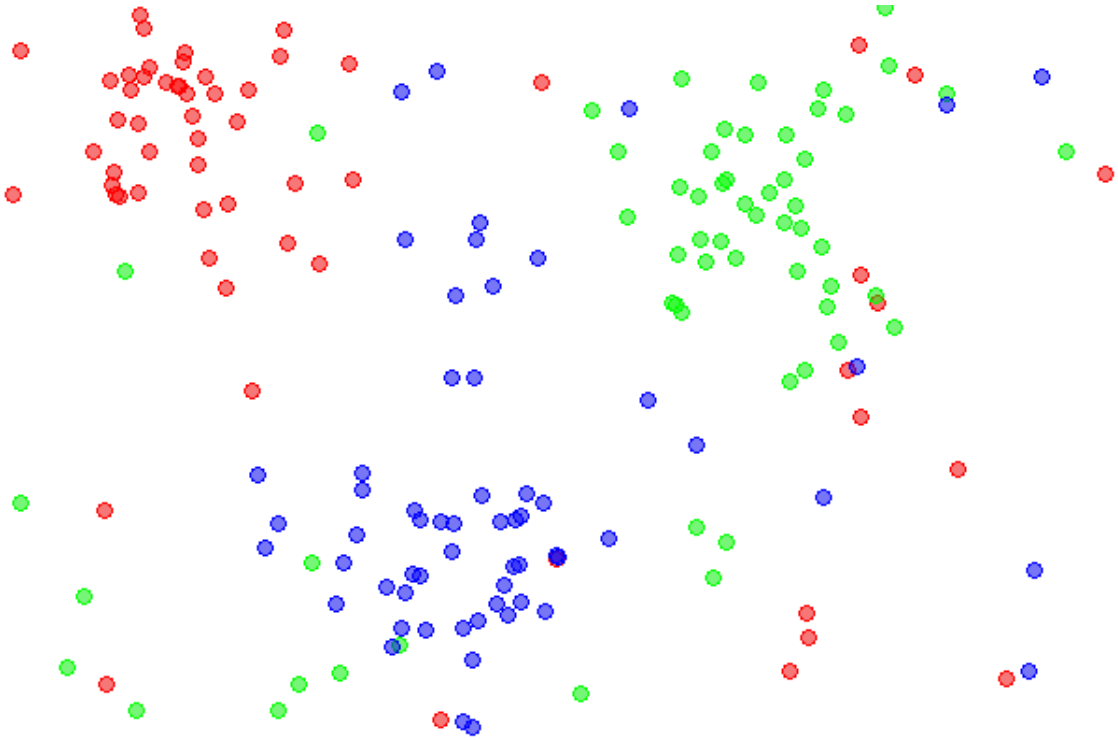
[2]:
```
<IPython.lib.display.IFrame at 0x7f8c18558700>
```

- 150 **observations**
- 4 **features** (sepal length, sepal width, petal length, petal width)
- **Response** variable is the iris species
- **Classification** problem since response is categorical
- More information in the UCI Machine Learning Repository

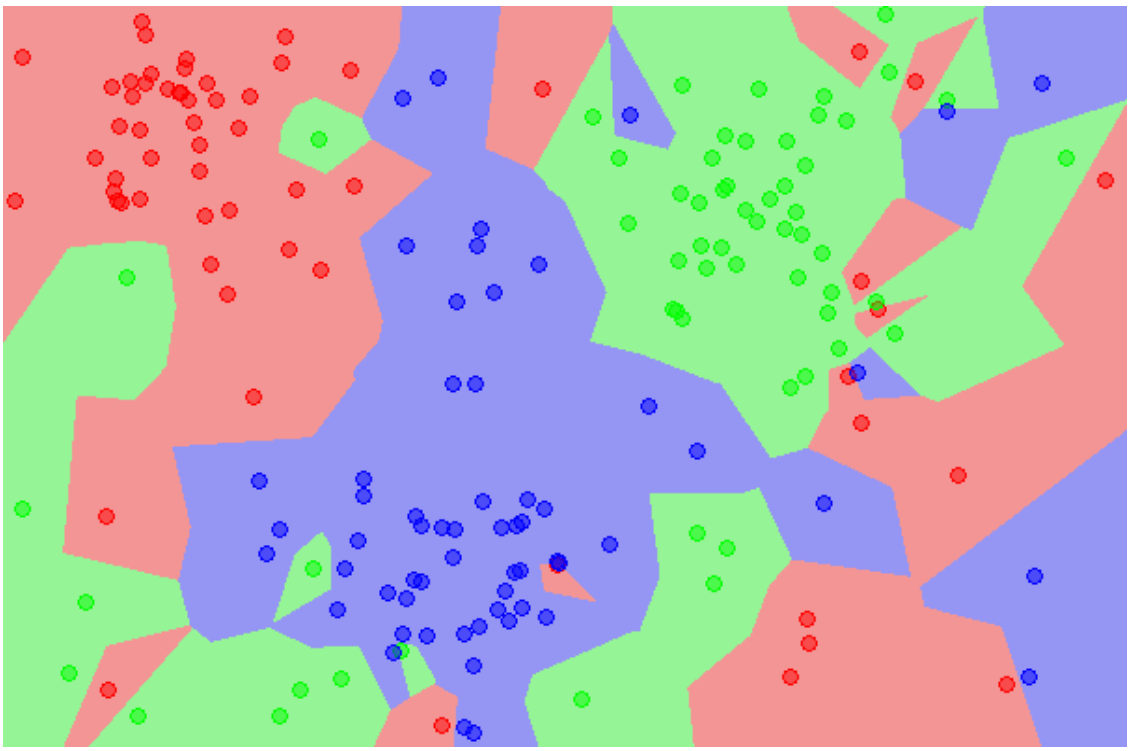
1.3 K-nearest neighbors (KNN) classification

1. Pick a value for K.
2. Search for the K observations in the training data that are "nearest" to the measurements of the unknown iris.
3. Use the most popular response value from the K nearest neighbors as the predicted response value for the unknown iris.

1.3.1 Example training data



1.3.2 KNN classification map (K=1)



1.3.3 KNN classification map (K=5)

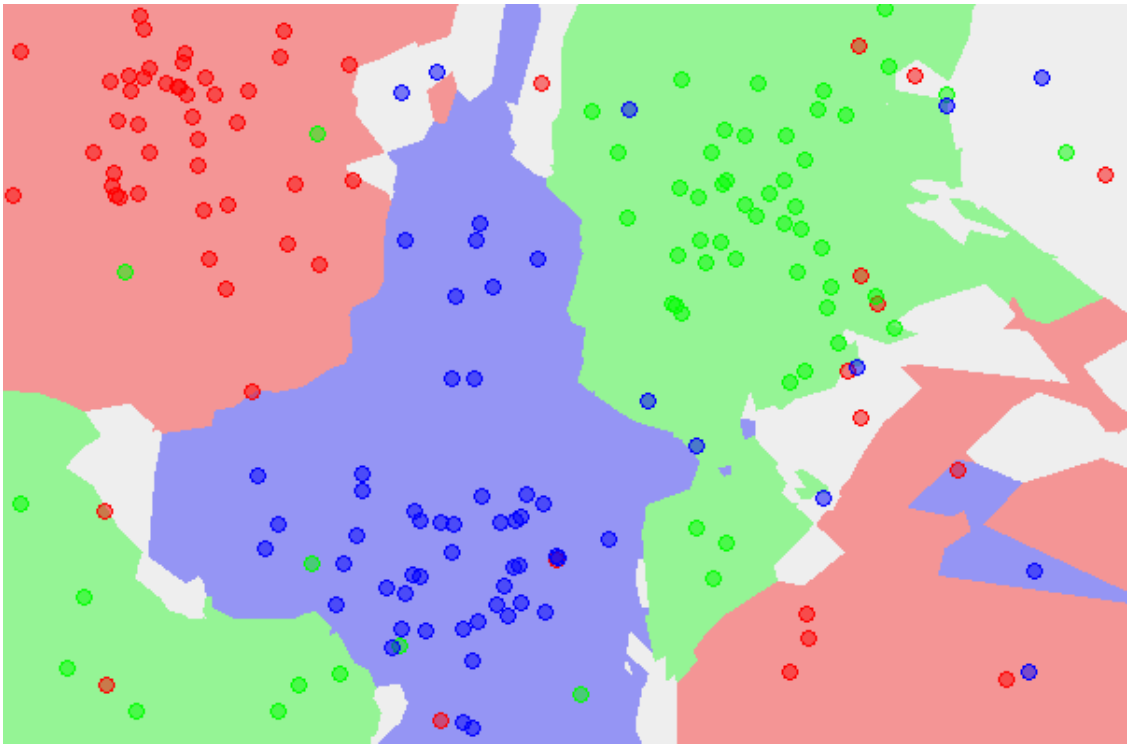


Image Credits: Data3classes, Map1NN, Map5NN by Agor153. Licensed under CC BY-SA 3.0

1.4 Loading the data

[3]:
```
# import load_iris function from datasets module
from sklearn.datasets import load_iris

# save "bunch" object containing iris dataset and its attributes
iris = load_iris()

# store feature matrix in "X"
X = iris.data

# store response vector in "y"
y = iris.target
```

[4]:
```
# print the shapes of X and y
print(X.shape)
print(y.shape)
```

```
(150, 4)
(150,)
```

1.5 scikit-learn 4-step modeling pattern

Step 1: Import the class you plan to use

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

Step 2: "Instantiate" the "estimator"

- "Estimator" is scikit-learn's term for model
- "Instantiate" means "make an instance of"

[6]:
```
knn = KNeighborsClassifier(n_neighbors=1)
```

- Name of the object does not matter
- Can specify tuning parameters (aka "hyperparameters") during this step
- All parameters not specified are set to their defaults

[7]:
```
print(knn)
```

```
KNeighborsClassifier(n_neighbors=1)
```

Step 3: Fit the model with data (aka "model training")

- Model is learning the relationship between X and y
- Occurs in-place

[8]:
```
knn.fit(X, y)
```

[8]:
```
KNeighborsClassifier(n_neighbors=1)
```

Step 4: Predict the response for a new observation

- New observations are called "out-of-sample" data
- Uses the information it learned during the model training process

[9]:
```
knn.predict([[3, 5, 4, 2]])
```

[9]:
```
array([2])
```

- Returns a NumPy array
- Can predict for multiple observations at once

[10]:
```
X_new = [[3, 5, 4, 2], [5, 4, 3, 2]]
knn.predict(X_new)
```

[10]:
```
array([2, 1])
```

1.6 Using a different value for K

[11]:
```
# instantiate the model (using the value K=5)
knn = KNeighborsClassifier(n_neighbors=5)

# fit the model with data
```

```
knn.fit(X, y)

# predict the response for new observations
knn.predict(X_new)
```

[11]:
```
array([1, 1])
```

1.7 Using a different classification model

[12]:
```
# import the class
from sklearn.linear_model import LogisticRegression

# instantiate the model
logreg = LogisticRegression(solver='liblinear')

# fit the model with data
logreg.fit(X, y)

# predict the response for new observations
logreg.predict(X_new)
```

[12]:
```
array([2, 0])
```

1.8 Resources

- Nearest Neighbors (user guide), KNeighborsClassifier (class documentation)
- Logistic Regression (user guide), LogisticRegression (class documentation)
- Videos from An Introduction to Statistical Learning
  - Classification Problems and K-Nearest Neighbors (Chapter 2)
  - Introduction to Classification (Chapter 4)
  - Logistic Regression and Maximum Likelihood (Chapter 4)

1.9 Comments or Questions?

- Email: kevin@dataschool.io
- Website: https://www.dataschool.io
- Twitter: @justmarkham