Introduction to Classification: K Nearest Neighbors in Python

Know your neighbors

Jake Parbs japarbs@42.student.org

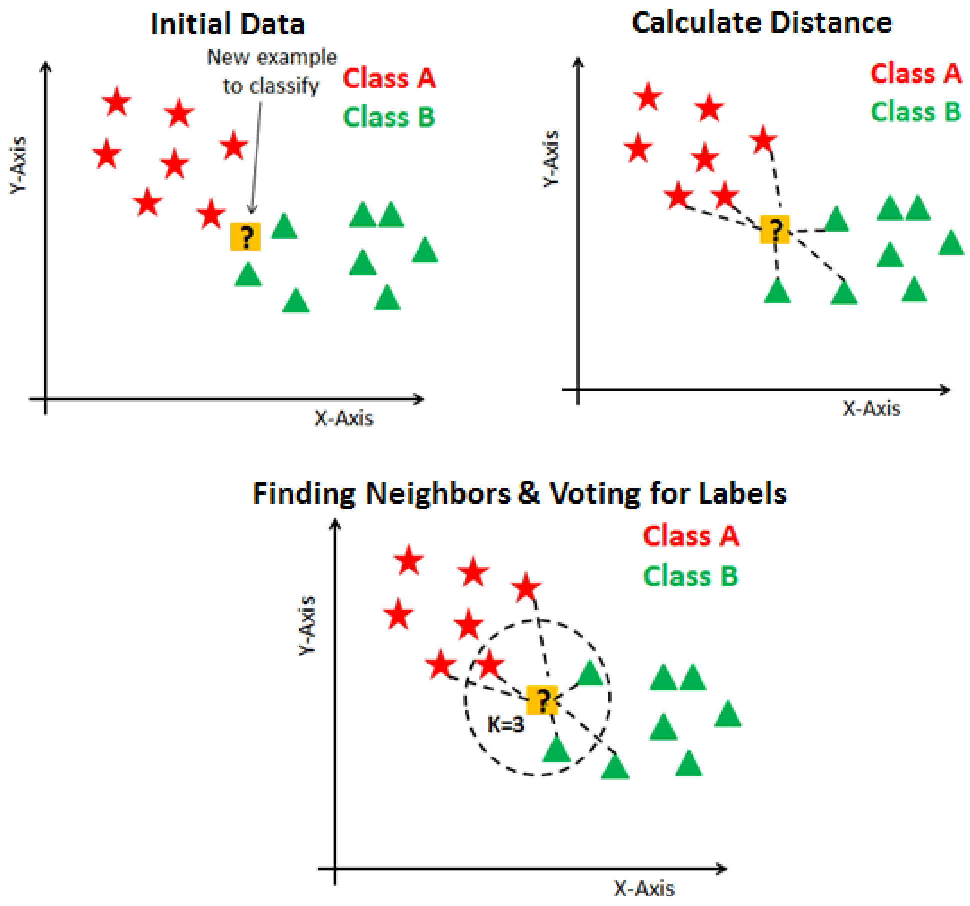
Summary: Intro to Classification, K Nearest Neighbors model and practice project.

Contents

Chapter I

Foreword

In Machine Learning there are countless models to be used for a variety of tasks. Today we'll be looking into a lightweight but powerful ML model, N Nearest Neighbors. It's one of the many classification models you'll use in Machine Learning. It's most useful for grouping together similar data points into preexisting classes. You'll learn all about KNN throughout this project with the help of Coursera.



Chapter II

Coursera setup

We will be continuing to use Coursera. Be sure to login as usual.

We will not be using the Quizzes that are built into the curriculum on Coursera. However do feel free to use the in-video questions to make sure you understand the topic.

Chapter III

Ex00: Introduction to K Nearest Neighbors

H2S	Introduction to K Nearest Neighbors
Topics to study : Introduction to K Nearest Neighbors on Coursera	
Files to turn in : None	
Notes : Watch all of the KNN section of week 3 of Coursera.	

This will be continuing from the previous PDF. Our first topic, K Nearest Neighbors is covered in the first part of Week 3 on Coursera.

Chapter IV

Ex01: Showcase Project

Run the following command within your working directory to download the dataset.

```
wget -O iris.data https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data
```

Here's the dataset's repository for all the details about this dataset.

We'll be using the same environment for the last Showcase Project in "Introduction to Machine Learning in Python". If you don't have your environment from the previous project, refer to chapter 6 of the last PDF "Showcase Project Setup".

H2S	Showcase Project
Topics to study : `Create your Showcase Project`	
Files to turn in : `showcase.py`	
Notes : `This will be what you create for your main grade.`	

When completed, the project should have the following capabilities:

- Accept an input of the 4 measurements of the Iris flower from the user to predict the class of a single observation.

- Display a scatter plot of the data with the classes of flowers color coded.

- Display the Train and Test accuracy.

- Show a read of the data in command line.

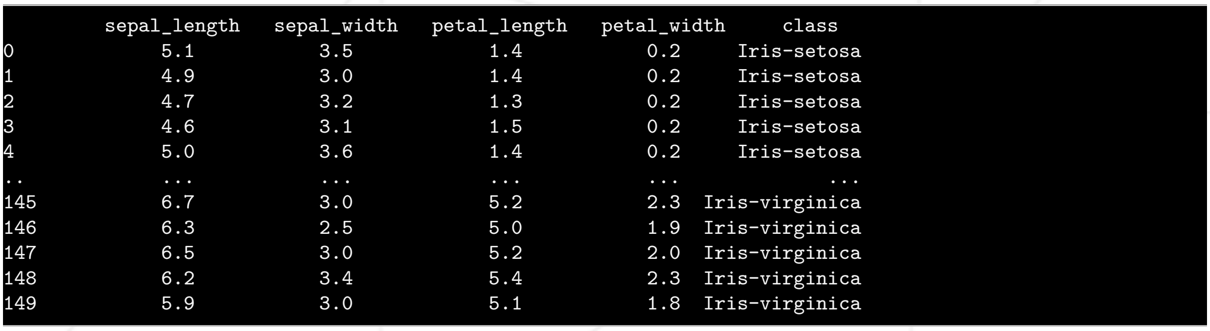
There are more detailed instructions below.

`If you're stuck, ask a peer!`

IV.1 Demonstration

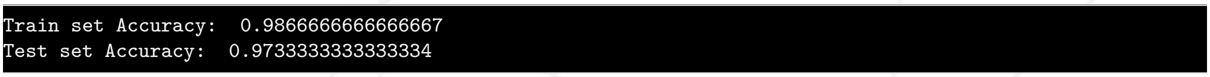
The Showcase Project should contain the following:

IV.1.1 A read of the data in command line



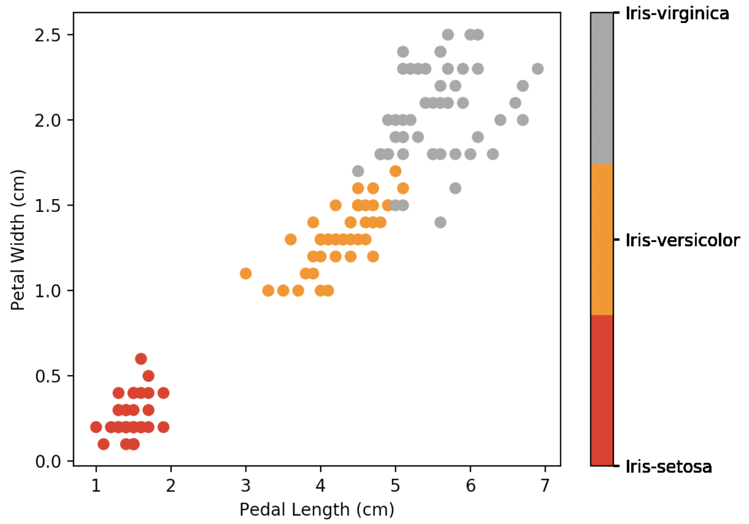
```
        sepal_length  sepal_width  petal_length  petal_width           class
0                5.1          3.5           1.4          0.2     Iris-setosa
1                4.9          3.0           1.4          0.2     Iris-setosa
2                4.7          3.2           1.3          0.2     Iris-setosa
3                4.6          3.1           1.5          0.2     Iris-setosa
4                5.0          3.6           1.4          0.2     Iris-setosa
..               ...          ...           ...          ...             ...
145              6.7          3.0           5.2          2.3  Iris-virginica
146              6.3          2.5           5.0          1.9  Iris-virginica
147              6.5          3.0           5.2          2.0  Iris-virginica
148              6.2          3.4           5.4          2.3  Iris-virginica
149              5.9          3.0           5.1          1.8  Iris-virginica
```

IV.1.2 Accuracy tests

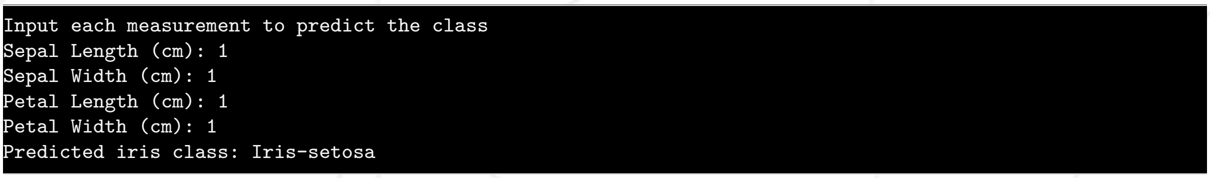


```
Train set Accuracy:  0.9866666666666667
Test set Accuracy:  0.9733333333333334
```

IV.1.3 Classified graph comparing Pedal Width and Pedal Length



IV.1.4 User input prediction



```
Input each measurement to predict the class
Sepal Length (cm): 1
Sepal Width (cm): 1
Petal Length (cm): 1
Petal Width (cm): 1
Predicted iris class: Iris-setosa
```

IV.2 Additional Information

The data provided in the CSV file is not labeled. You will have to label the data yourself!

Coursera doesn't cover classified scatter plots yet. Here is some of the code to make to make it function:

Class Conversion

The following snippet of Python will convert the string classifiers to numbered ones instead. This is needed as Matplotlib doesn't take non-numeric class values.

```
typeofiris = np.empty(150, dtype='object')
for label in range(150):
    if y[label] == 'Iris-setosa':
        typeofiris[label] = 0
    if y[label] == 'Iris-versicolor':
        typeofiris[label] = 1
    if y[label] == 'Iris-virginica':
        typeofiris[label] = 2
```

Color Map

To plot the classes you will need to input "typeofiris" to the "c" parameter of the scatter plot initialization, as well as a color map under the "cmap" parameter.

```
c=typeofiris, cmap=plt.cm.get_cmap('Set1', 3)
```

Color Bar and Labels

After that, create a color bar and labels.

```
cb = plt.colorbar()
cb.set_ticks(typeofiris)
cb.set_ticklabels(y)
```

That's all that's new!

Chapter V

Turn-in and peer-evaluation

Turn your work in using your `GiT` repository, as usual. Only work present on your repository will be graded in defense.