

Machine Learning Recipes

Hello World

INSTRUCTOR: JOSH GORDON

Notes by Behic Guven

Recipe #1

Two open source libraries:

- Scikit Learn
- TensorFlow

Examples

- Deep Blue
- AlphaGo

Can you write a code that understand the differences between apple and orange ?



To create the classifier we need to know about Supervised Learning.

Supervised Learning: Create a classifier by finding patterns in examples

Supervised Learning Recipe



How to understand the training data:

Features

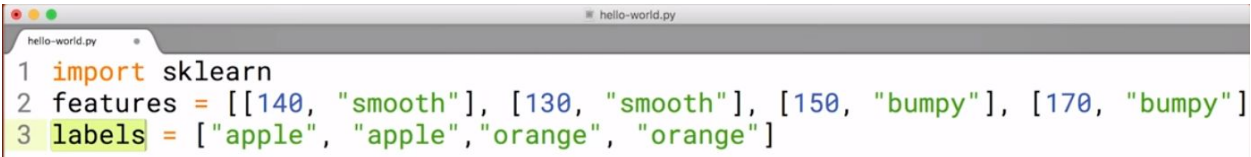
Weight	Texture	Label
150g	Bumpy	Orange
170g	Bumpy	Orange
140g	Smooth	Apple
130g	Smooth	Apple
...

The more training data the more the classifier gets trained.

Let's write around first code

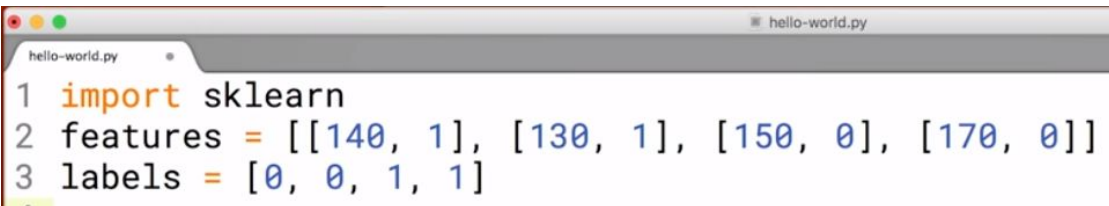
Step 1: Collect Training Data

1. Create text file and save it as helloworld.py
2. Then open the file and write "import sklearn"

3. The screenshot shows a code editor window titled 'hello-world.py'. It contains the following Python code:

```
1 import sklearn
2 features = [[140, "smooth"], [130, "smooth"], [150, "bumpy"], [170, "bumpy"]]
3 labels = ["apple", "apple", "orange", "orange"]
```

4. You can think features as inputs and labels as outputs.
5. Note: scikit-learn uses real valued features (meaning can't understand the text) so we have them numerical
6. So we will use 1 for smooth and 0 for bumpy in features and 0 for apple and 1 for orange in labels. Here is the code after the changes.

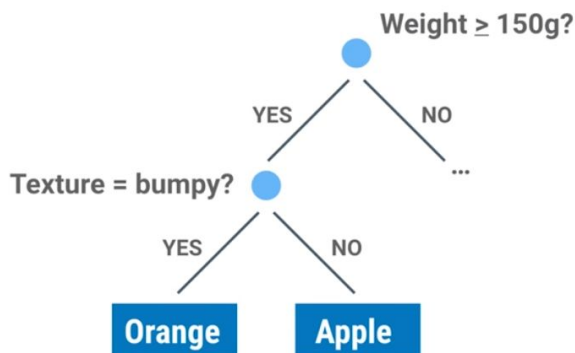
7. 

```
1 import sklearn
2 features = [[140, 1], [130, 1], [150, 0], [170, 0]]
3 labels = [0, 0, 1, 1]
```

Step 2: Train Classifier

1. You can think classifiers as box of rules. In this example we will use decision tree.

Decision Tree



- 2.
3. Here is the code after adding the tree classifier. Fit is searching for pattern in the training data.

4. 

```
1 from sklearn import tree
2 features = [[140, 1], [130, 1], [150, 0], [170, 0]]
3 labels = [0, 0, 1, 1]
4 clf = tree.DecisionTreeClassifier()
5 clf = clf.fit(features, labels)
6 print clf.predict([[150, 0]])
```

```
5. from sklearn import tree
   features = [[140,1],[130,1],[150,1],[170,0]]
   labels = [0,0,1,1]
   clf = tree.DecisionTreeClassifier()
   clf = clf.fit(features, labels)
   print(clf.predict([[160,0]]))
```

Step 3: Make Predictions

6. Print `clf.predict([[150,0]])` line of code will predict the fruit according to what it learnt from the training data. We are asking when it is 150g and bumpy.
7. And the output is 1 meaning it is more likely to be Orange

Now you can use this method for other problems. You just have to change the training data and that's it.

Important Concepts to keep in mind:

Important Concepts

- How does this work in the real world?
- How much training data do you need?
- How is the tree created?
- What makes a good feature?

Congrats you did a great job.

