ML Workshop

Machine Learning for everyone!

{SJODEVCenter;}
Amazon

Agenda

- Machine Learning
- Demo
- Exercise

Machine Learning

"Field of study that gives computers the ability to learn without being explicitly programmed”

Arthur Samuel

Machine Learning

- Machine Learning covers the field of investigating and developing algorithms that can learn from data and even make predictions based on their own findings.

- Their base is a trial error system, in which a model based on different inputs/outputs is built and analyzed by the system itself.

Machine Learning

- Supervised Learning: The machine is given some inputs and the desired outputs. Its goal to generate a process to reproduce them.

- Unsupervised Learning: The learning algorithm is left alone to produce its own inputs searching for a goal.

- Reinforcement Learning: The machine just interacts with its environment while performing a certain goal. Play against an opponent for example.

What we will do…

- Download and install Python SciPy and get the most useful package for machine learning in Python.
- Load a dataset and understand it's structure using statistical summaries and data visualization.
- Create 6 machine learning models, pick the best and build confidence that the accuracy is reliable.

Typical ML recipe

- Define Problem.
- Prepare Data.
- Evaluate Algorithms.
- Improve Results.
- Present Results.

Classical Iris Dataset

- https://archive.ics.uci.edu/ml/datasets/Iris

- The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant.

- Most well known data set for ML learning.

Classical Iris Dataset

- Attributes are numeric.
- It is a classification problem, easier type of supervised learning algorithm.
- It only has 4 attributes and 150 rows, meaning it is small and easily fits into memory.
- All of the numeric attributes are in the same units and the same scale, not requiring any normalization.

Hands on!

- Let’s install:
- Python version 2.7 or 3.5.
- scipy
- numpy
- matplotlib
- pandas
- sklearn

Hands on!

- Let’s install:
- Python version 2.7 or 3.5.
- scipy
- numpy
- matplotlib
- pandas
- sklearn
- → https://www.scipy.org/install.html ←

Hands on!

- Open a terminal and type:
  python

Hands on!

```
# Check the versions of libraries

# Python version
import sys
print('Python: {}'.format(sys.version))
# scipy
import scipy
print('scipy: {}'.format(scipy.__version__))
# numpy
import numpy
print('numpy: {}'.format(numpy.__version__))
# matplotlib
import matplotlib
print('matplotlib: {}'.format(matplotlib.__version__))
# pandas
import pandas
print('pandas: {}'.format(pandas.__version__))
# scikit-learn
import sklearn
print('sklearn: {}'.format(sklearn.__version__))
```

- Python: 2.7.10 (default, Oct 23 2015, 19:19:21)
- [GCC 4.2.1 Compatible Apple LLVM 7.0.0 (clang-700.0.59.5)]
- scipy: 0.13.0b1
- numpy: 1.8.0rc1
- matplotlib: 1.3.1
- pandas: 0.20.3
- sklearn: 0.19.1

Load the dataset

- Load the iris dataset, from the url, using Pandas

```
# Load libraries
import pandas
from pandas.tools.plotting import scatter_matrix
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import
LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
```

```
# Load dataset
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
dataset = pandas.read_csv(url, names=names)
```

Summarize dataset

- Dimensions of the dataset.
- Peek at the data itself.
- Statistical summary of all attributes.
- Breakdown of the data by the class variable.

Dimensions

```
# shape
print(dataset.shape)
```

Results in:

```
(150, 5)
```

Peek Data

```
# head
print(dataset.head(20))
```

Results in: List of the first 20 elements of the dataset

Statistical Summary

```
# descriptions
print(dataset.describe())
```

Results in:

```
       sepal-length  sepal-width  petal-length  petal-width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000
```

Class aggrupation

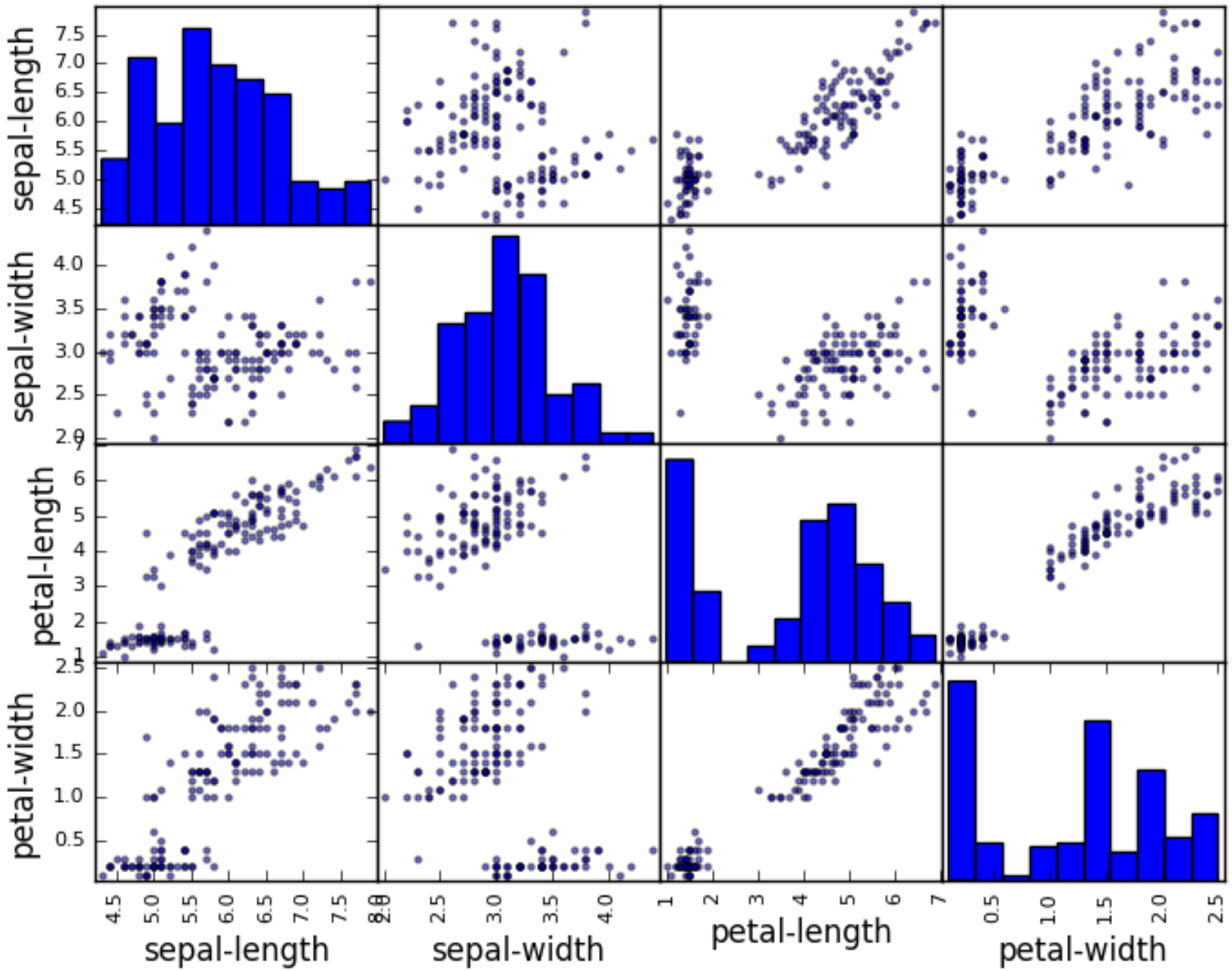
```
# class distribution
print(dataset.groupby('class').size())
```

Results in:

```
class
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
```

Look for correlation

```
# scatter plot matrix
scatter_matrix(dataset)
plt.show()
```


sepal-length
sepal-width
petal-length
petal-width
4.5
5.0
5.5
6.0
6.5
7.0
7.5
2.0
2.5
3.0
3.5
4.0
1
2
3
4
5
6
7
0.5
1.0
1.5
2.0
2.5

Create a validation dataset

```
# Split-out validation dataset
array = dataset.values
X = array[:,0:4]
Y = array[:,4]
validation_size = 0.20
seed = 7
scoring = 'accuracy'
X_train, X_validation, Y_train, Y_validation =
model_selection.train_test_split(X, Y, test_size=validation_size,
random_state=seed)
```

Build Models

- Logistic Regression (LR)
- Linear Discriminant Analysis (LDA)
- K-Nearest Neighbors (KNN).
- Classification and Regression Trees (CART).
- Gaussian Naive Bayes (NB).
- Support Vector Machines (SVM).

```
# Spot Check Algorithms
models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))
# evaluate each model in turn
results = []
names = []
for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=seed)
    cv_results = model_selection.cross_val_score(model, X_train, Y_train, cv=kfold,
scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)
```

- LR: 0.966667 (0.040825)
- LDA: 0.975000 (0.038188)
- KNN: 0.983333 (0.033333)
- CART: 0.975000 (0.038188)
- NB: 0.975000 (0.053359)
- SVM: 0.981667 (0.025000)

Make predictions

```
# Make predictions on validation dataset
knn = KNeighborsClassifier()
knn.fit(X_train, Y_train)
predictions = knn.predict(X_validation)
print(accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
```

Results

0.9

```
[[ 7  0  0]
[ 0 11  1]
[ 0  2  9]]
```

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	7
Iris-versicolor	0.85	0.92	0.88	12
Iris-virginica	0.90	0.82	0.86	11
avg / total	0.90	0.90	0.90	30

Exercise

- Think about a real life problem you can solve with Machine Learning.
- Not something complicated, i.e.: “Calculate the gender of a person based on their age and shoe size.”

Thanks!!!

- Q&A ??