Python for Data Scientists
L15 (1) : Follow-up lecture on assignment 7

more libraries!!

Data exploration and analysis

- NumPy
- Pandas
- Scipy

Data exploration and analysis

NumPy:

- introduces objects for multidimensional arrays and matrices
- provides functions that allow to easily perform advanced mathematical and statistical operations on ndarrays
- provides vectorization of mathematical operations on arrays and matrices which significantly improves the performance
- many other python libraries are built on NumPy

Link: http://www.numpy.org/

Data exploration and analysis

Pandas:

- adds data structures and tools designed to work with table-like data
- provides tools for data manipulation: reshaping, merging, sorting, slicing, aggregation etc.
- allows handling missing data

Link: http://pandas.pydata.org/

Data exploration and analysis

SciPy:

- collection of algorithms for linear algebra, differential equations, numerical integration, optimization, statistics and more

- works alongside NumPy arrays to provide a platform that provides numerous mathematical methods like, numerical integration and optimization.

Link: https://www.scipy.org/scipylib/

Plotting and visualization libraries

- Matplotlib
- Seaborn
- Plotly

Plotting and visualization libraries

matplotlib:

- python 2D plotting library which produces publication quality figures in a variety of hardcopy formats
- line plots, scatter plots, barcharts, histograms, pie charts etc.
- relatively low-level; some effort needed to create advanced visualization

Link: https://matplotlib.org/

Plotting and visualization libraries

Seaborn:

- based on matplotlib
- provides high level interface for drawing attractive statistical graphics
- Similar (in style) to the popular ggplot2 library in R

Link: https://seaborn.pydata.org/

Plotting and visualization libraries

Plotli:

- Python graphing library which makes interactive and publication-quality graphs
- supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific, and 3-dimensional use-cases.
- Built on the Plotly JavaScript library

Link: https://plotly.com/python/

Classical Machine learning libraries

- Scikit-Learn
- StatsModels
- XGBoost

Classical Machine learning libraries

SciKit-Learn:

- provides machine learning algorithms: classification, regression, clustering, model validation etc.

- built on NumPy, SciPy and matplotlib libraries

Link: http://scikit-learn.org/

Classical Machine learning libraries

SciKit-Learn: example

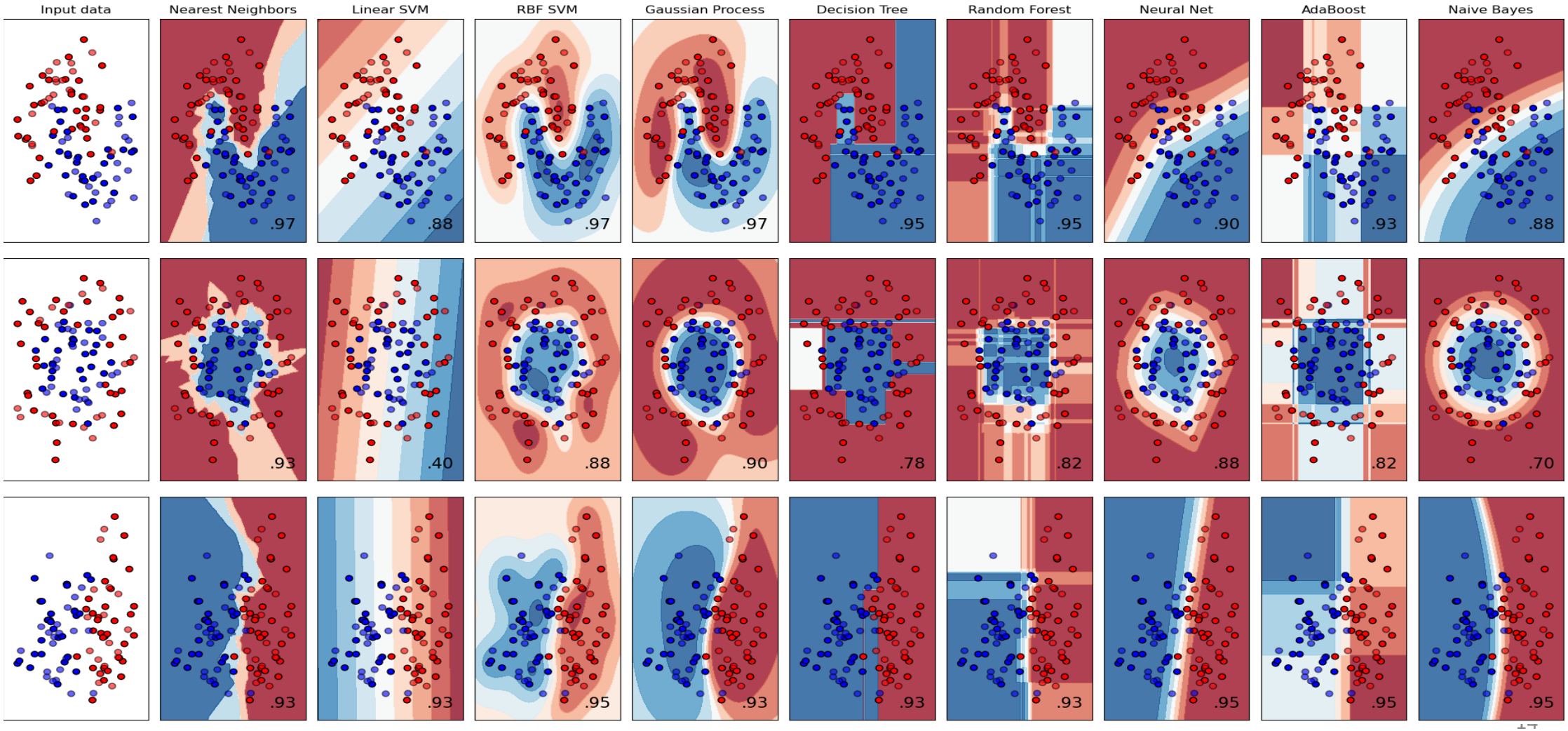
```
# Code source: Gaël Varoquaux
#              Andreas Müller
# Modified for documentation by Jaques Grobler
# License: BSD 3 clause

import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import make_moons, make_circles, make_classification
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis

h = .02  # step size

names = ["Nearest Neighbors", "Linear SVM", "RBF SVM", "Gaussian Process",
         "Decision Tree", "Random Forest", "Neural Net", "AdaBoost",
         "Naive Bayes", "QDA"]
```

```
classifiers = [
    KNeighborsClassifier(3),
    SVC(kernel="linear", C=0.025),
    SVC(gamma=2, C=1),
    GaussianProcessClassifier(1.0 * RBF(1.0)),
    DecisionTreeClassifier(max_depth=5),
    RandomForestClassifier(max_depth=5, n_estimators=10,
max_features=1),
    MLPClassifier(alpha=1, max_iter=1000),
    AdaBoostClassifier(),
    GaussianNB()]

X, y = make_classification(n_features=2, n_redundant=0,
n_informative=2,
                           random_state=1, n_clusters_per_class=1)
rng = np.random.RandomState(2)
X += 2 * rng.uniform(size=X.shape)
linearly_separable = (X, y)

datasets = [make_moons(noise=0.3, random_state=0),
            make_circles(noise=0.2, factor=0.5, random_state=1),
            linearly_separable
            ]
```


https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html#sphx-glr-auto-examples-classification-plot-classifier-comparison-py

Classical Machine learning libraries

SciKit-Learn: example



https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html#sphx-glr-auto-examples-classification-plot-classifier-comparison-py

Classical Machine learning libraries

StatModels:

- provides classes and functions for the estimation of many different statistical models, as well as for conducting statistical tests, and statistical data exploration.

- provides a complement to scipy for statistical computations including descriptive statistics and estimation and inference for statistical models.

Link: https://www.statsmodels.org/stable/

Classical Machine learning libraries

XGBoost:

- Faster than other ensemble classifiers (Originally written in C++)
- The core XGBoost algorithm is parallelizable.
- Shown better performance on a variety of machine learning benchmark datasets.
- XGBoost has parameters for: cross-validation, regularization, user-defined objective functions, missing values, tree parameters, scikit-learn compatible API, …

Link:
https://xgboost.readthedocs.io/en/latest/python/index.html

Deep learning libraries

- Keras
- TensorFlow
- Pytorch

Deep learning libraries

Keras:

- high-level neural networks API for Python.
- running on top of the machine learning platform TensorFlow (open-source machine learning platform).
- contains numerous implementations of commonly used neural-network building blocks
- Provides tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code.

Link: https://keras.io/

Deep learning libraries

TensorFlow

- is a Python library for fast numerical computing created and released by Google.

- used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensodFlow

Link: https://www.tensorflow.org/

Deep learning libraries

TensorFlow and Keras example:

```
from numpy import loadtxt
import keras
import tensorflow
from keras.models import Sequential
from keras.layers import Dense


# load the dataset to train the model
dataset = loadtxt('pima-indians-diabetes.data.csv',
delimiter=',')

# Split into feature- and classification data
X = dataset[:,0:8]
y = dataset[:,8]

# define the keras model
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

```
# compile the keras model
model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])

# fit the keras model on the dataset
model.fit(X, y, epochs=150, batch_size=10, verbose=0)

# evaluate the keras model
_, accuracy = model.evaluate(X, y)
print('Accuracy: %.2f' % (accuracy*100))
```

Accuracy: 77.47

https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/

Deep learning libraries

Pytorch*:*

- developed by Facebook's AI Research lab (FAIR)
- being Pythonic, smoothly integrates with the Python data science stack.
- used for applications such as computer vision and natural language processing

Link: https://pytorch.org/

NLP libraries

- NLTK
- SpaCy
- Gensim

NLP libraries

NLTK*:*

- work with human language data
- provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet
- Provides text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, ...

Link: https://www.nltk.org/

NLP libraries

SpaCy:

- written in the programming languages Python and Cython.
- construct linguistically sophisticated statistical models for a variety of NLP problems.
- helps you build applications that process and “understand” large volumes of text : information extraction or natural language understanding systems, or to pre-process text for deep learning.

Link: https://spacy.io/

NLP libraries

Gensim:

- *Used for topic modelling*, *document indexing* and *similarity retrieval* with large corpora.
- Specially used for *natural language processing* and *information retrieval* community.
- Efficient multicore implementations of popular algorithms, such as online Latent Semantic Analysis, Latent Dirichlet, or word2vec deep learning.

Link: https://pypi.org/project/gensim/

NLP libraries

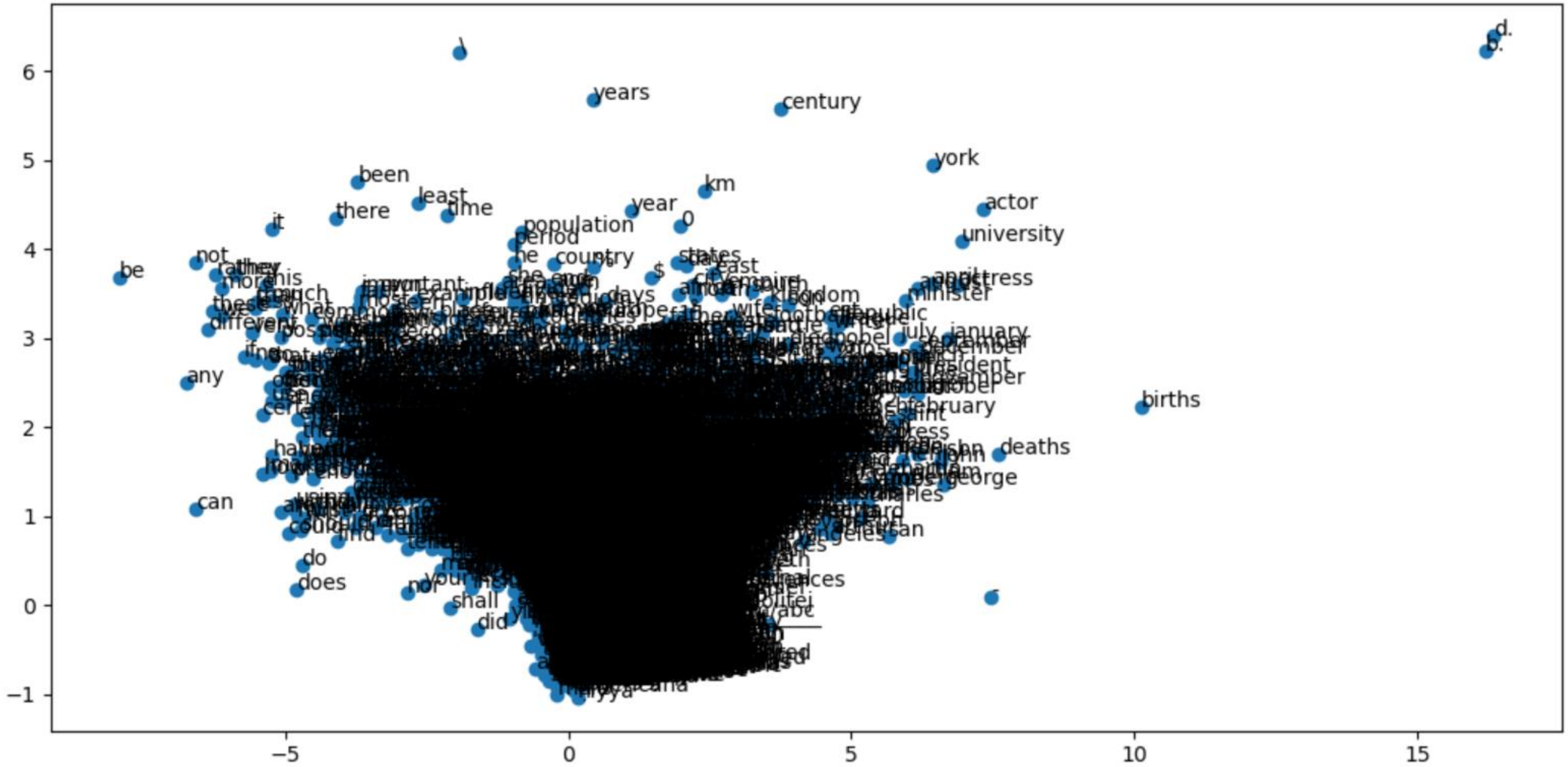
Gensim: Example

```
import gensim as gensim
from sklearn.decomposition import PCA
from matplotlib import pyplot

text_file_name = 'smallWikipedia.txt'
sentences = gensim.models.word2vec.LineSentence(text_file_name, limit=100000)

simple_model = gensim.models.Word2Vec(sentences, size=10, window=5, min_count=5, workers=2)
word_vectors = simple_model.wv

print("The word vector for cat is : ",word_vectors['cat'])

X = simple_model[simple_model.wv.vocab]
pca = PCA(n_components=2)
result = pca.fit_transform(X)
# create a scatter plot of the projection
pyplot.scatter(result[:, 0], result[:, 1])
words = list(simple_model.wv.vocab)
for i, word in enumerate(words):
   pyplot.annotate(word, xy=(result[i, 0], result[i, 1]))
pyplot.show()
```

https://machinelearningmastery.com/develop-word-embeddings-python-gensim/

NLP libraries

Gensim: Example



Data storage and big data frameworks

- Apache Spark
- HDFS

Data storage and big data frameworks

*A*pache Spark*:*

- fast and general engine for big data processing, with built-in modules for streaming, SQL, machine learning and graph processing.

- does in-memory computations to analyze data in real-time

Link: https://www.tutorialspoint.com/pyspark/index.htm

Data storage and big data frameworks

HDFS*:*

- provides machine learning algorithms: classification, regression, clustering, model validation etc.
- built on NumPy, SciPy and matplotlib libraries

Link: https://pypi.org/project/hdfs/