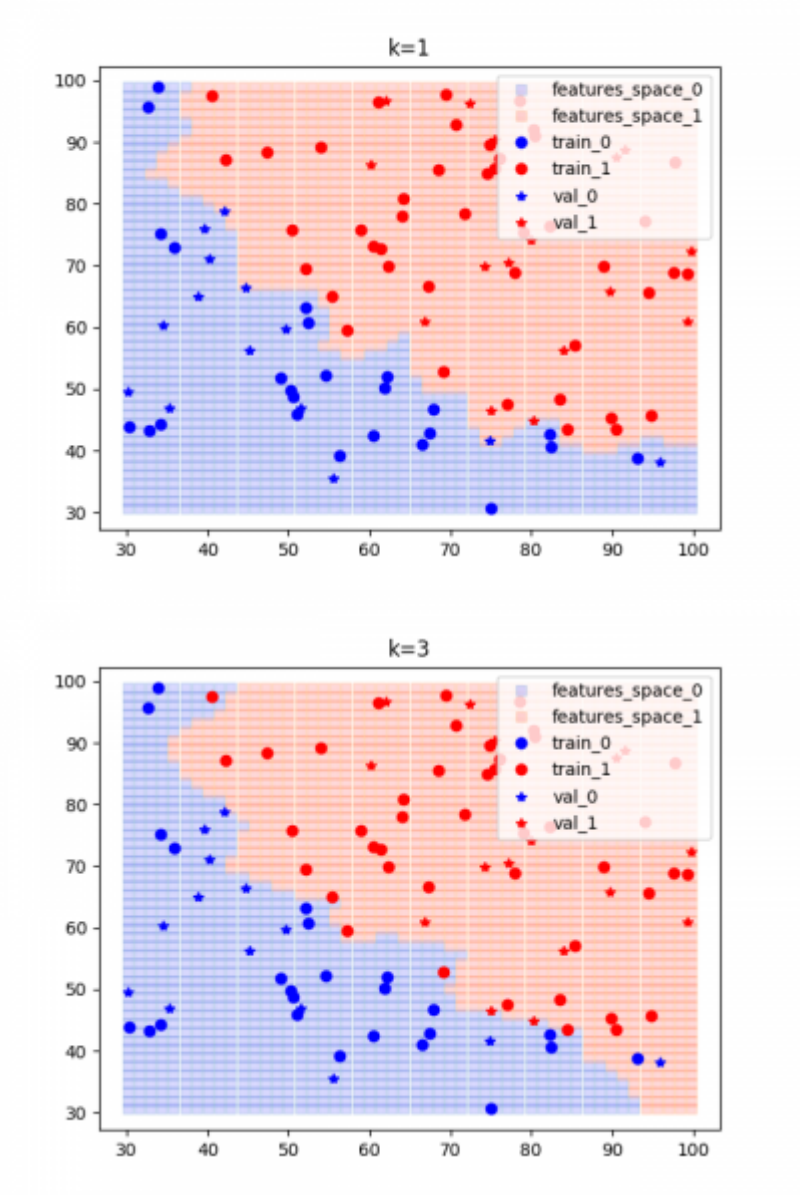
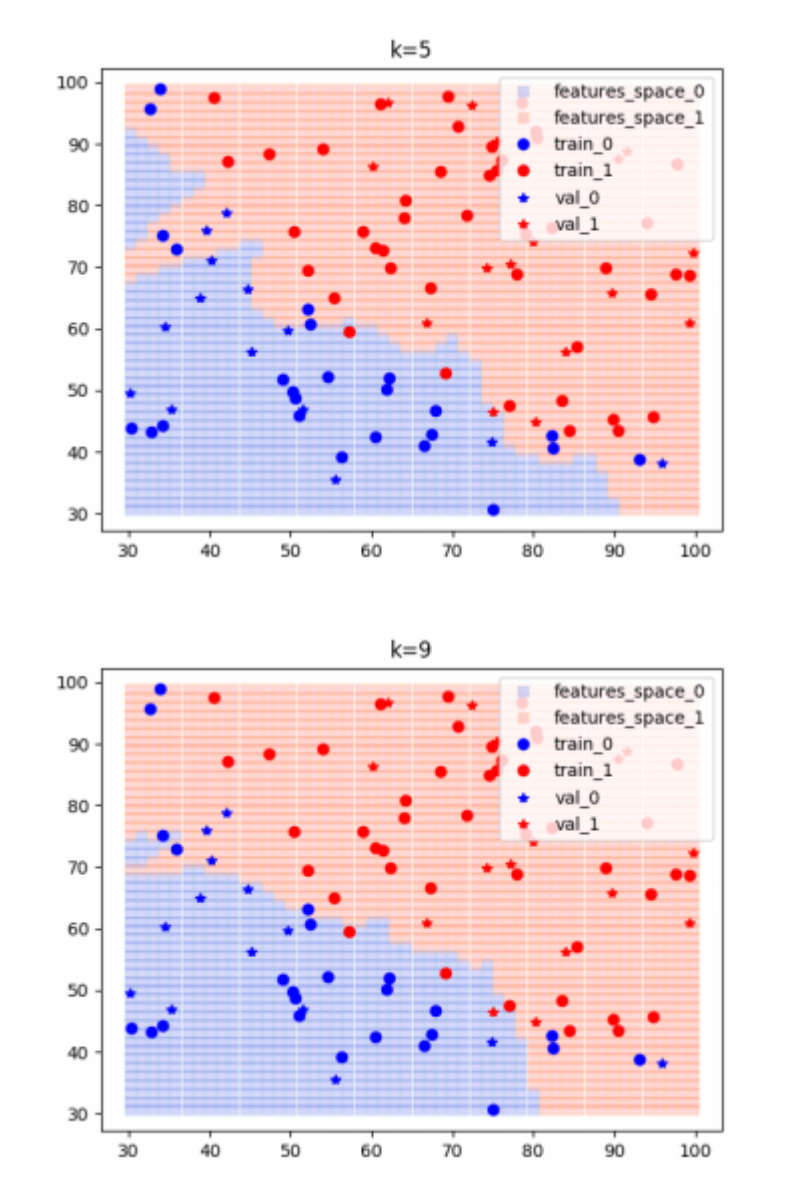
Classification

Instrukcja po polsku (fragmenty mogą być nieaktualne):
Zagadnienie klasyfikacji

Lecture e-learning

Additional materials::
Drzewa decyzyjne - lecture PL showing basic information and tree structure
kNN - lecture ANG - algorithm kNN





Materials

Additional material: Drzewa decyzyjne

Ex. 1 - kNN - own implementation

Please provide your own implementation of the “K nearest neighbors” algorithm.

Task:

1. Iris database - please divide into training and test sets
2. Find the best value of k for the selected test set (show on the graph change of k and the error - for both the test and training sets)

1. We choose the value of k (np. from 1 to n, where n will be the value for which the algorithm's results will deteriorate again)
2. For each example in the test set, we look for the k observations that are closest to our analyzed example. Use the Eukleidesa distance to determine the distance.
3. The advantage of a given class among the “k” closest neighbors (from the training set) proves that the example belongs to this class.
4. Use the most frequent value of the “k” nearest neighbors“ as the value for an unclassified example.
5. For each case of k, after classifying all examples in the test set, count the error (e.g. the number of examples wrongly classified to the set size) and write the pair (k - error) to collect data for summary graphs.

3. Suggest the best k for the set.
Useful libraries and functions:

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from scipy.spatial import distance
from sklearn.metrics import accuracy_score
```

Example:

```
db_iris = datasets.load_iris()

#separate the set for teaching and testing, test_size - percentage share
(example 70% for training and 30% for testing)
features_train, features_test, labels_train, labels_test =
train_test_split(iris.data, iris.target, test_size=0.5)

#An example of using the Euclidean distance
a = (1, 2, 3)
b = (4, 5, 6)
dst = distance.euclidean(a, b)

# Checking the effectiveness of the classifier
output = accuracy_score(labels_test, predictions)
print(output)
```

Ex. 2 - kNN - Python (sklearn)

Please solve the above problem using the library `sklearn.neighbors.KNeighborsClassifier` in order to check the correct implementation of the kNN algorithm
Useful libraries and functions:

```
from sklearn import datasets
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
```

Ex.3 - Decision trees

In the scikit-learn library, decision trees are implemented by the class: `DecisionTreeClassifier`.
For implementation details, please see here:
Decision Trees - Python
Klasa sklearn.tree.DecisionTreeClassifier

Example - Iris dataset

```
from sklearn.datasets import load_iris
from sklearn import tree
iris = load_iris()
clf = tree.DecisionTreeClassifier()
clf.fit(iris.data, iris.target)
```

Predicting examples to belong to classes:

```
clf.predict(iris.data[:1, :])
```

or estimating the probability of belonging to classes:

```
clf.predict_proba(iris.data[:1, :])
```

Please illustrate the result using the 'Graphiz' tool and installing the 'pydot' library to python (please do the task at home and generate a '*.png' or '*.pdf' file)

```
from six import StringIO
import pydot
dot_data = StringIO()
tree.export_graphviz(clf, out_file=dot_data)
graph = pydot.graph_from_dot_data(dot_data.getvalue())
graph[0].write_pdf("iris.pdf")
```

W nowszej wersji Pythona funkcja `write_pdf` jest zastąpiona przez `write`.

Exercise

Classification of handwritten numbers. Images have been normalized to size $ 28$ x $28 $ px.
The data that we will use is an image converted to a one-dimensional format by arranging successive lines. This data comes from the MNIST handwritten number database.

Dataset: https://keras.io/examples/vision/mnist_convnet/
More Information about MNIST Dataset in Python – Basic Importing and Plotting:
https://www.journaldev.com/45249/mnist-dataset-in-python

If you download mnist from another source, make sure it is converted from 28 x 28 px and correctly transposed.

```
# -*- coding: utf-8 -*-
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import classification_report, confusion_matrix,
f1_score
from sklearn import tree
from sklearn.model_selection import train_test_split
from scipy.io import loadmat



# wczytywanie danych (np. z biblioteki keras)
dane = #TODO

#Ex.1.Divide the data into parameters X and answer y:

X = #TODO
y = #TODO

# Standardization
for i in range(X.shape[0]):
    X[i,:] = X[i,:]/np.std(X[i,:])

# Convert digit 10 -> 0 (error in the dataset)
y[np.where(y==10)]=0

# the height and width of the picture with the number
h = 28
w = 28

# Ex 2. Please display the number of digits and the number of pixels per
image
#TODO
```

Auxiliary function `plot_mnist` to display pictures from the database:

```
def plot_mnist(images, titles, h, w, n_row=3, n_col=4):
    plt.figure(figsize=(1.8 * n_col, 2.4 * n_row))
    plt.subplots_adjust(bottom=0, left=.01, right=.99, top=.90, hspace=.05)
    for i in range(n_row * n_col):
        plt.subplot(n_row, n_col, i + 1)
        plt.imshow(images[i].reshape((h, w)).T, cmap=plt.cm.gray)
        plt.title(titles[i], size=12)
        plt.xticks(())
        plt.yticks(())
```

Ex. 3. Please display sample digits from the dataset (function`plot_mnist`).

Ex. 4. Please divide the dataset into learner (70%) and training.
Ex. 5. Create an instance of the classifier, then train and predict for test data.
Tree parameters:

```
DEPTH = 10
```

Zad 6. Please provide F1 result, confusion matrix and classification report.