SGN-41007 Pattern Recognition and Machine Learning

Exercise Set 3: January 25.1–27.1.2017

Exercises consist of both pen&paper and computer assignments. Pen&paper questions are solved at home before exercises, while computer assignments are solved during exercise hours. The computer assignments are marked by text **`python`** and Pen&paper questions by text **`pen&paper`**

1. **`pen&paper`** *Design an optimal detector for step signal.*

   The lecture slides describe an optimal detector for a known waveform $s[n]$. Apply it to design the optimal detector for a step edge:

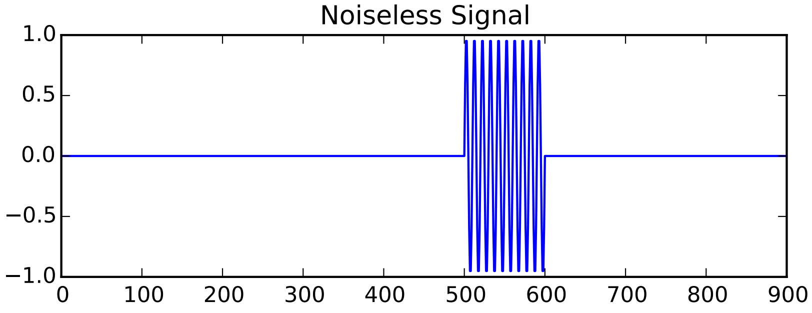
   $$s[n] = \begin{cases} -1, & \text{for } 0 \leq n < 10 \\ 1, & \text{for } 10 \leq n < 20 \end{cases}$$

   Simplify the expression as far as you can.

2. **`python`** *Implement a sinusoid detector.*

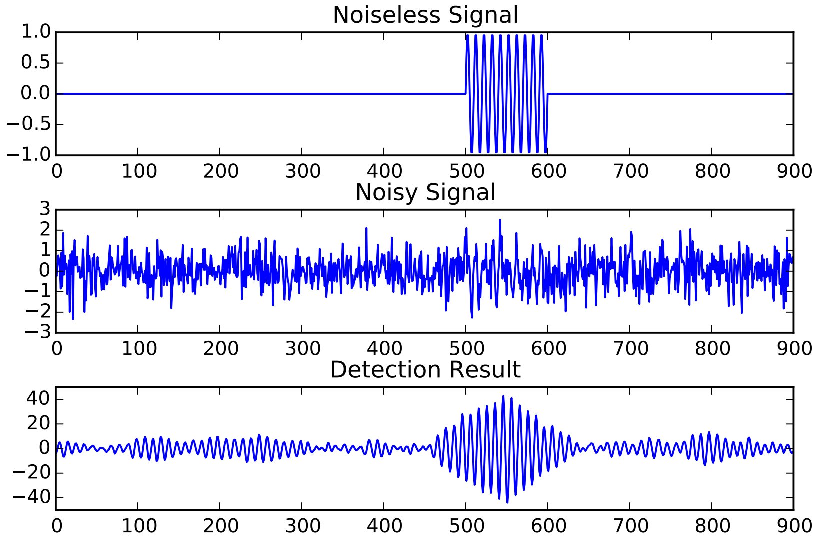
   In this exercise we generate a noisy sinusoid with known frequency and see how the sinusoid detector of the lecture slides performs.

   a) Create a vector of zero and sinusoidal components that looks like the plot below. Commands: `np.zeros`, `np.concatenate`. Sinusoid is generated by `np.cos(2 * np.pi * 0.1 * n)`.

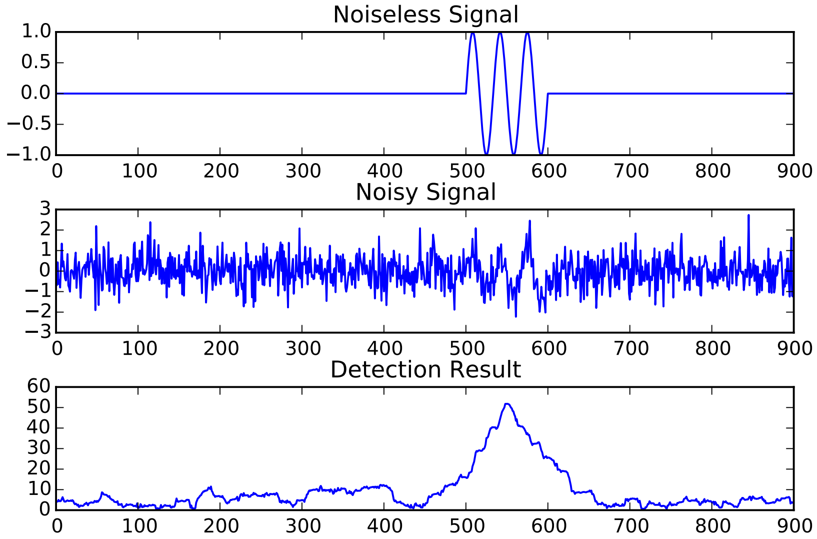
   b) Create a noisy version of the signal by adding Gaussian noise with variance 0.5: `y_n = y + np.sqrt(0.5) * np.random.randn(y.size)`.

   c) Implement the two detectors and reproduce the below plot.



3. `python` *Same as previous but different frequency and detector.*

Change the code of the previous exercise such that the frequency is 0.03 and the detector is the random signal version.



4. `python` *Load a dataset of images split to training and testing.*

We will train a classifier to classify hand written digits. Scikit-learn provides a number of sample datasets. Load the `digits`-dataset as follows.

```
from sklearn.datasets import load_digits
digits = load_digits()
```

The result is a `dict` structure that can be accessed using *keys*. Find all keywords of the dict with `print(digits.keys())`. The interesting ones for us are: `'images'`,`'data'` and `'target'`.

Plot the first image of the 1797 numbers like this.

```
import matplotlib.pyplot as plt
plt.gray()
plt.imshow(digits.images[0])
plt.show()
```

Check that this corresponds to the label `digits.target[0]`.

The images are vectorized as rows in the matrix `digits.data`, whose size is 1797×64 (1797 images of size 8×8).

Split the data to training and testing sets, such that the training set consists of 80% and test set 20% of the data. Use `sklearn.cross_validation.train_test_split` to do this and create variables `x_train`, `y_train`, `x_test`, `y_test`.

5. **python** *Train a classifier using the image data.*

In this exercise we will train a nearest neighbor classifier with the data arrays of exercise 4.

- Initiate a KNN classifier with

```
from sklearn.neighbors import KNeighborsClassifier
clf = KNeighborsClassifier()
```

- Train the classifier using the training data.
- Predict the labels for the test data.
- Compute the accuracy using `sklearn.metrics.accuracy_score`.