Exercise 2

Exercise 1

Algorithm 1: KNN results

```
$ python ex02/src/knn_classifier.py -d manhattan 42
INFO:root:Test accuracy (K=42, d=manhattan): 0.911

$ python ex02/src/knn_classifier.py -d euclidean 42
INFO:root:Test accuracy (K=42, d=euclidean): 0.911
```

Exercise 2

The distance metric d is a *categorical* hyperparameter with two distinct values. K is a *integer* hyperparameter.

Algorithm 2: Exhaustive search results

```
$ python ex02/src/exhaustive_search.py
The accuracy of KNN with (K, d) = (6, euclidean) is 0.940.
The accuracy of KNN with (K, d) = (9, euclidean) is 0.940.
The accuracy of KNN with (K, d) = (10, euclidean) is 0.960.
The accuracy of KNN with (K, d) = (11, euclidean) is 0.940.
The accuracy of KNN with (K, d) = (12, euclidean) is 0.960.
The accuracy of KNN with (K, d) = (13, euclidean) is 0.940.
The accuracy of KNN with (K, d) = (14, euclidean) is 0.920.
The accuracy of KNN with (K, d) = (8, manhattan) is 0.960.
The accuracy of KNN with (K, d) = (9, manhattan) is 0.920.
The accuracy of KNN with (K, d) = (10, manhattan) is 0.940.
The accuracy of KNN with (K, d) = (12, manhattan) is 0.940.
```

We choose the range of K = [1, 0.9·num_trainingdata]. The factor of 0.9 is necessary, because of the 10-fold crossvalidation. The cv-accuracy goes down for larger values of K, because the area where the actual point is, gets less and less important. In the extrem case where K equals the training data size, it's completely irrelevant what the test data is. All training points are in the set of nearest neighbours and therefore the prediction solely depends on the distribution of the training data. For small K the drop off in cv-accuracy is not as big as with larger K but it's still worse. This is due to the higher sensitivity for outliers.

Exercise 3

Both exercises took between 2.5 and 3 hours each. In the first exercise it took some time to get into numpy again and in the second exercise it was more about finding the right method for the job