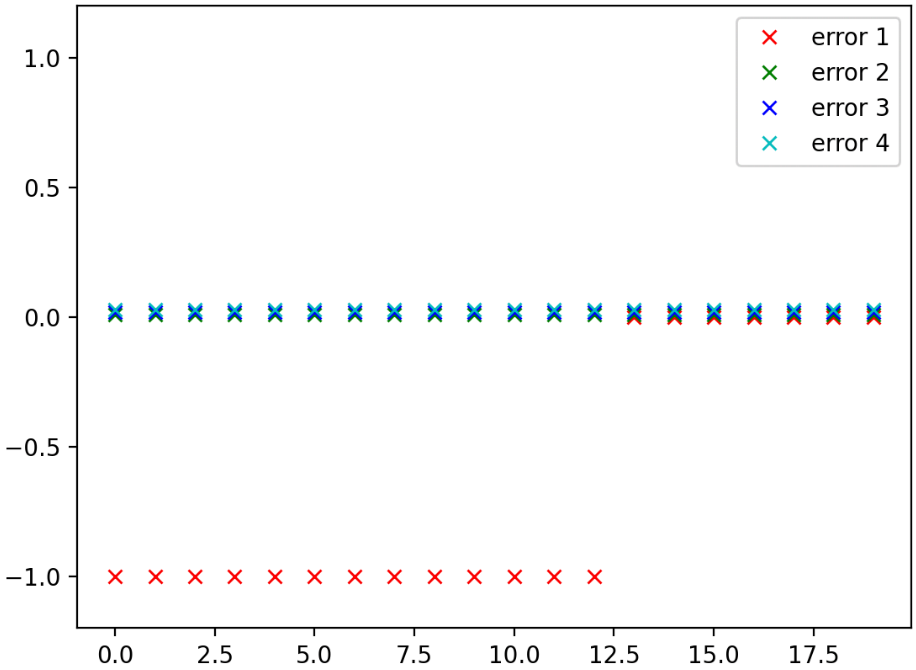
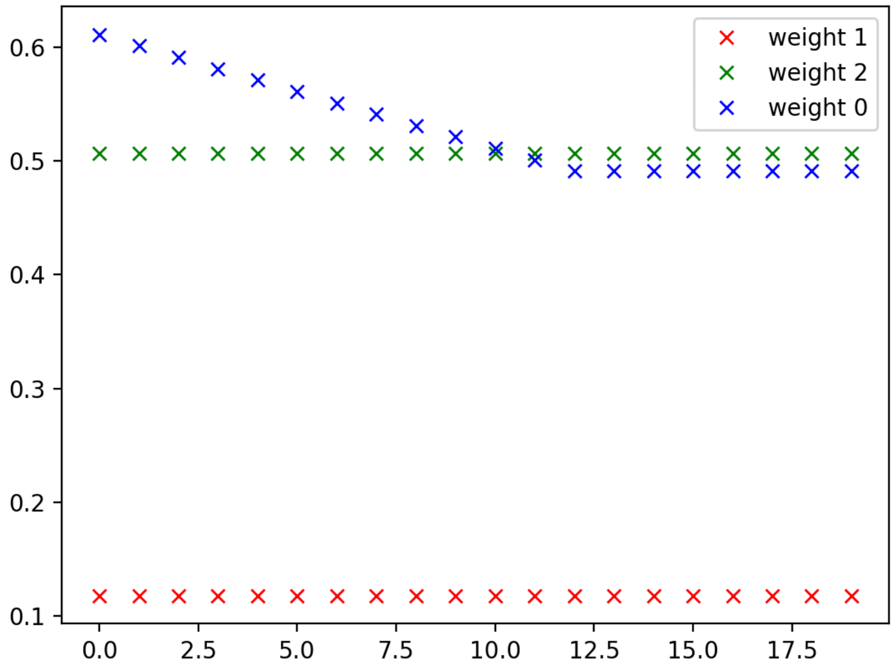
Exercise 1 - perceptron OR function

(a) Plot the value of the error at the end of each epoch, how does it behave?

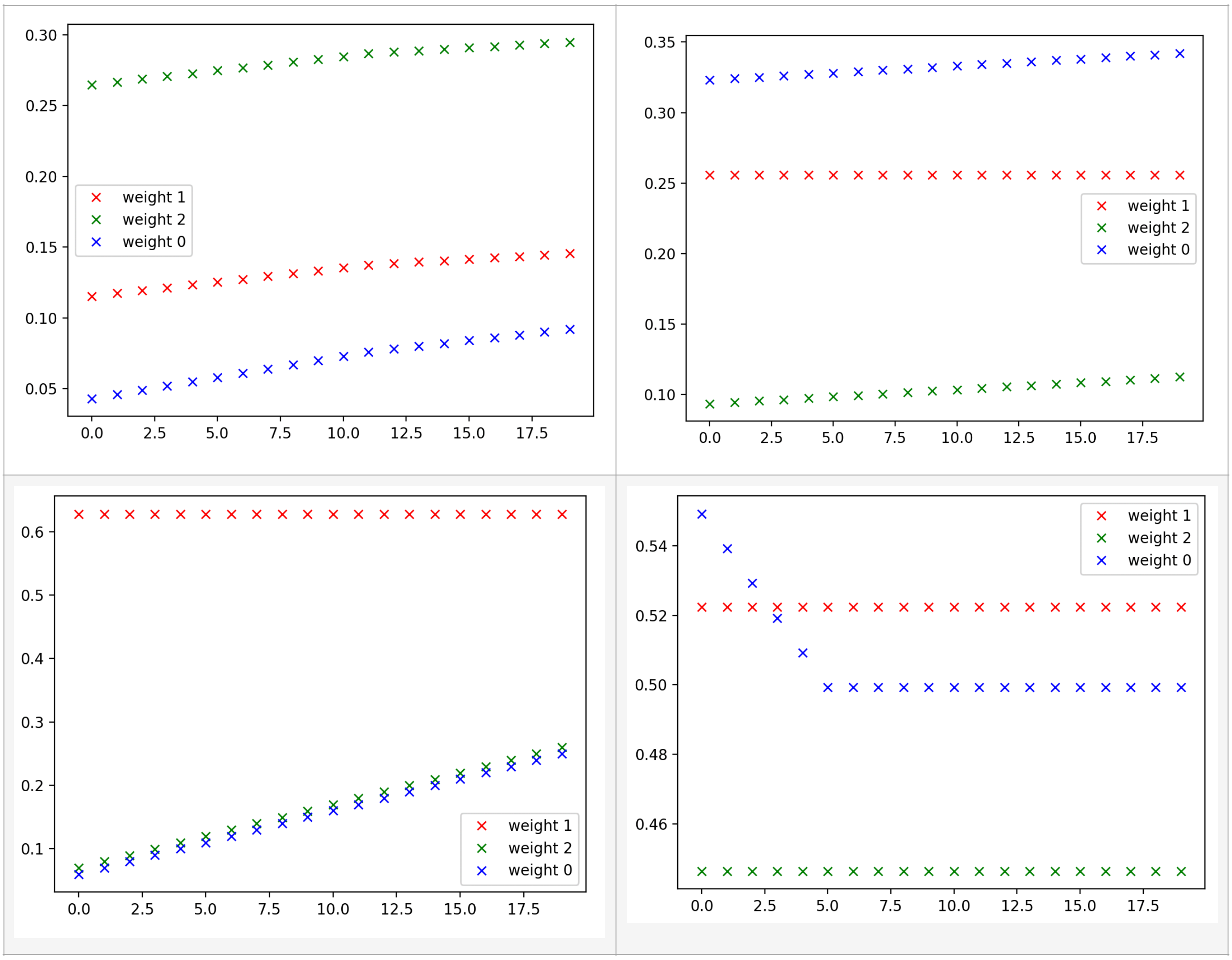


As we can see we start with error 1 being equal = 1 and with time it converges to zero. When we look at the weights:



We can see that only a small correction was needed to weight 0 (bias).

(b) Plot the value of each weight at the end of each training epoch. Are the values converging? if so, do they converge to similar values in different runs (with different random initialisations)?

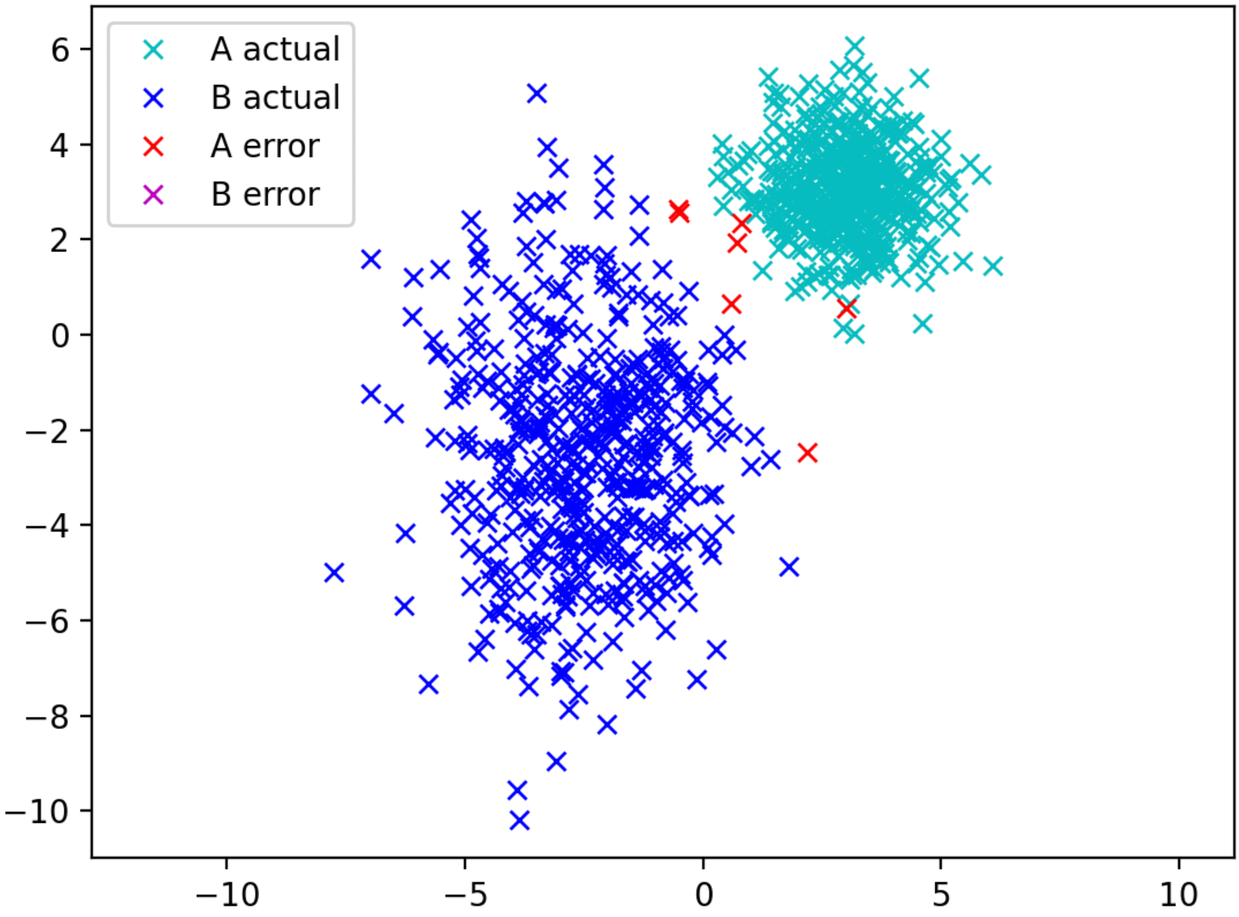


(c) What is the effect of increasing/decreasing the α parameter? Can you tell (approximately) what is the "best" value for α?
The parameter a influences the rate of change of each weight when correcting errors. From our experiments we concluded α = 10e-2 seems to give the best results.

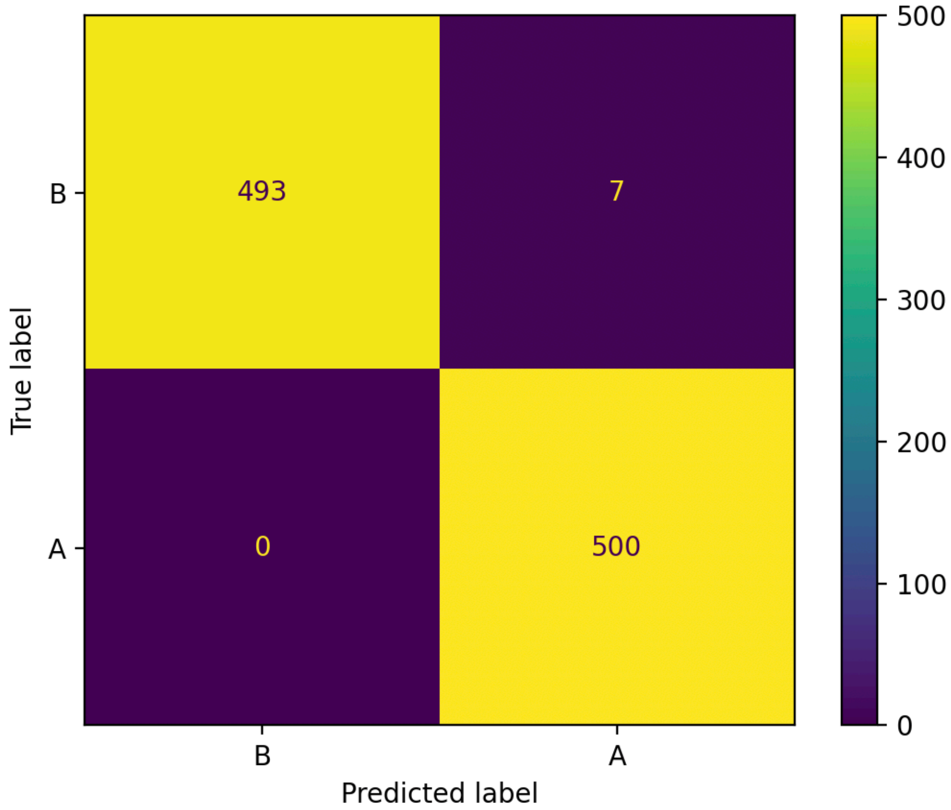
(d) How many epochs (iterations through the whole set) did it take to get all examples right? (i.e. ∀i : di = oi). Repeat the experiment 30 times with different random values for the initial weights and present the average and standard deviation of the number of epochs it took to converge.
average needed iterations: 7.47
standard dev of iterations: 5.739

8. Use the dataset generated in the previous item as the training set for the same per-

ceptron and train it to partition the two datasets (adjust number of epochs if neces-
sary). Notice that the same program learned two different tasks depending on the
dataset used.
Print with different colors:
(a) Points generated by the first distribution and labeled 1 by the perceptron;
(b) Points generated by the first distribution and labeled 0 by the perceptron;
(c) Points generated by the second distribution, labeled 1 by the perceptron;
(d) Points generated by the second distribution, labeled 0 by the perceptron.



9. Print/Plot the confusion matrix for the above test. Can you relate each



of the numbers in the confusion matrix to the points of a given color on the previously generated figure?

We can easily recognise the 7 outliers in the previous diagram.

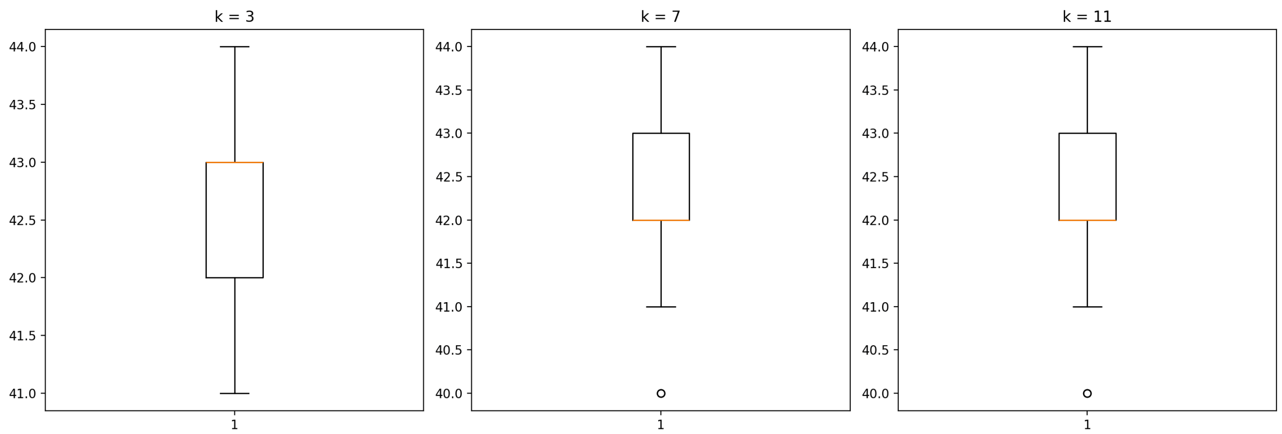
10. Print the metrics (accuracy, precision, recall, and F1) for all the tests: metrics should be an average for 30 tests with the same parameters but different initial weights.

accuracy: 0.9984
recall: 0.9994
precision: 0.997
F1: 0.998

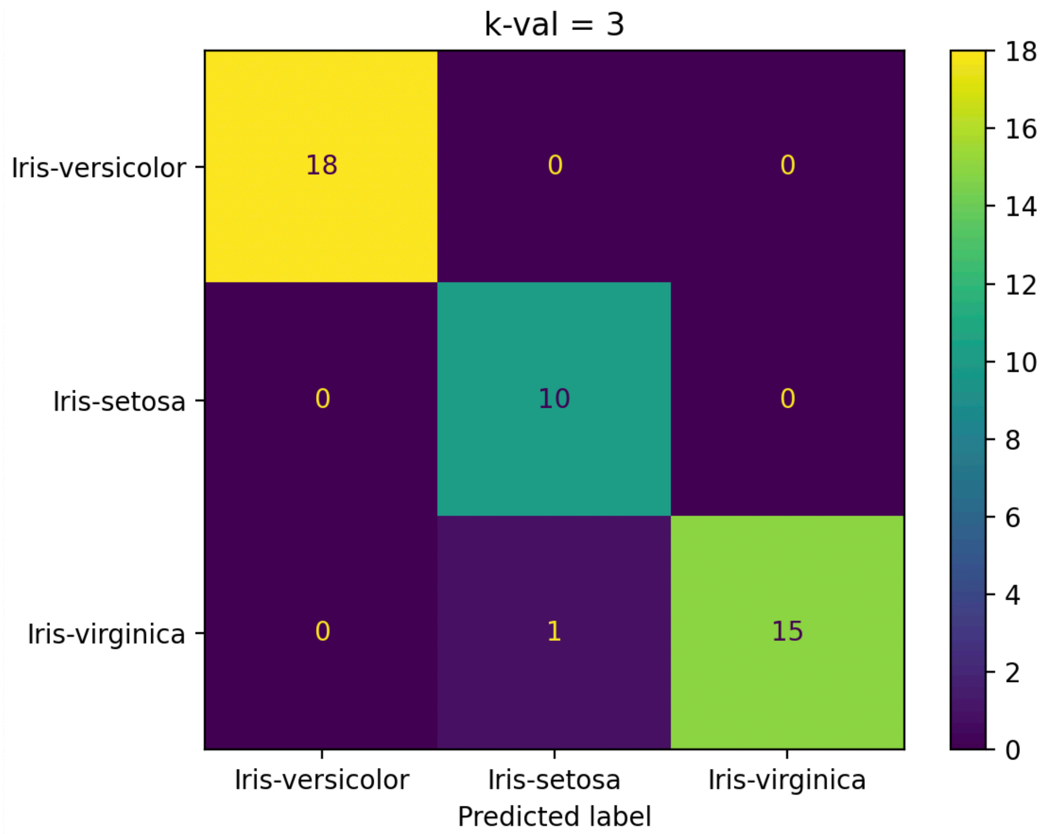
With alpha = 10e-2

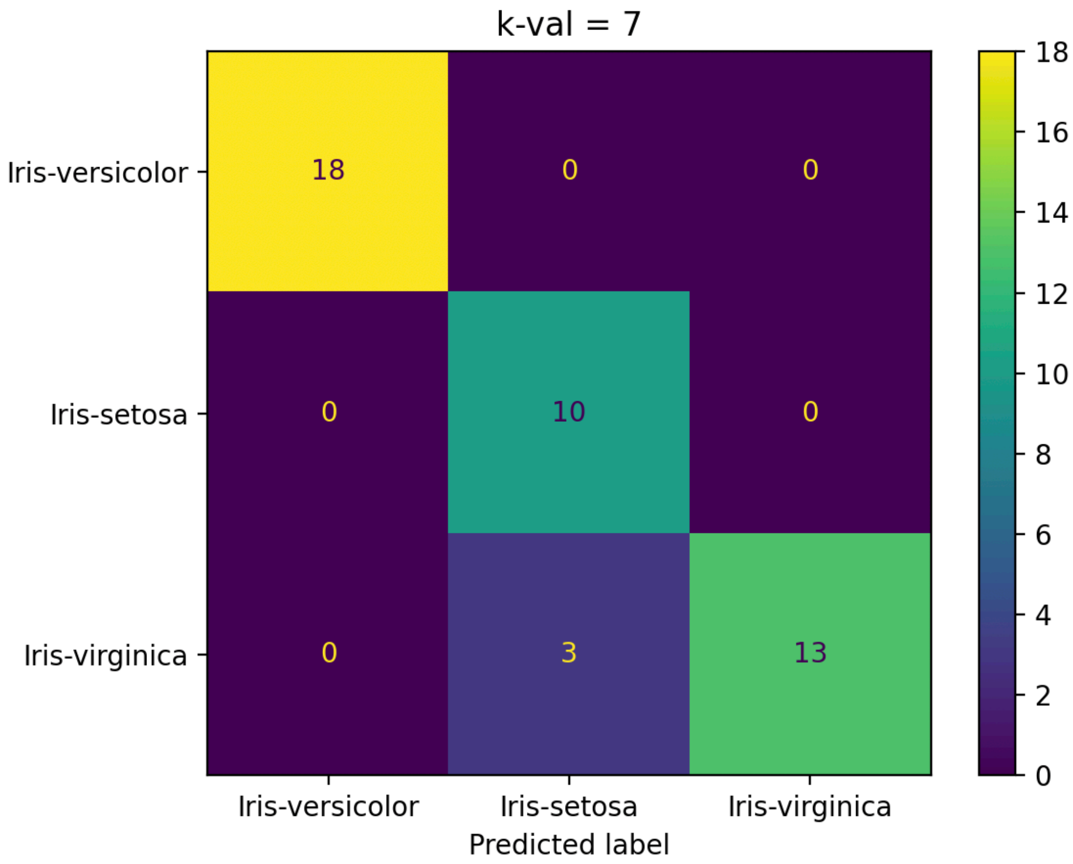
Exercise 2 - k-NN classifier

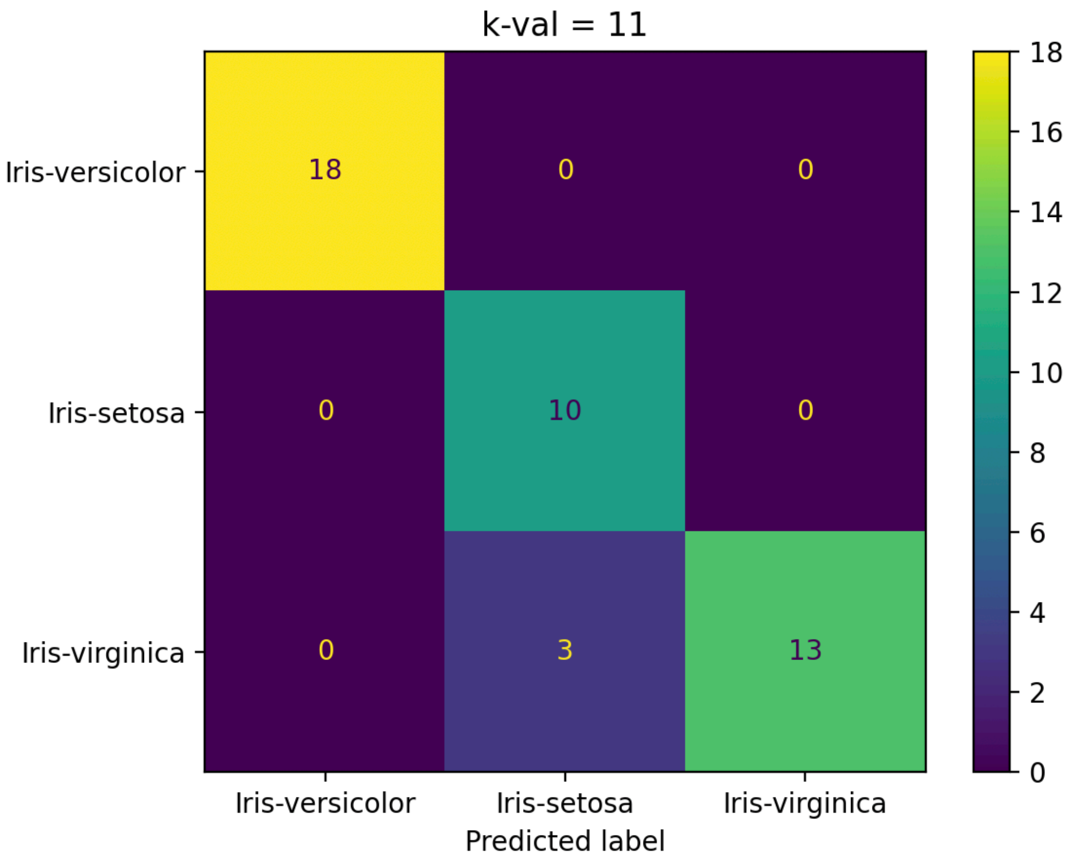
1. Split the dataset randomly in two subsets (70% / 30%). Use the bigger subset as the training set and the smaller as the test set. Run all test examples through the classifier and calculate the number of correct predictions over the total number of examples of the test set. Compare the scores of k-NN classifiers for k = 3, 7, and11. Repeat 30 times, with different dataset splits, for each value of k. Use a boxplot with whiskers graphic to allow easy comparison.



2. Plot the confusion matrix of one of the tests for each value of k.




k-val = 7
Iris-versicolor
Iris-setosa
Iris-virginica
18
0
0
0
10
0
0
3
13
Iris-versicolor
Iris-setosa
Iris-virginica
Predicted label
18
16
14
12
10
8
6
4
2
0

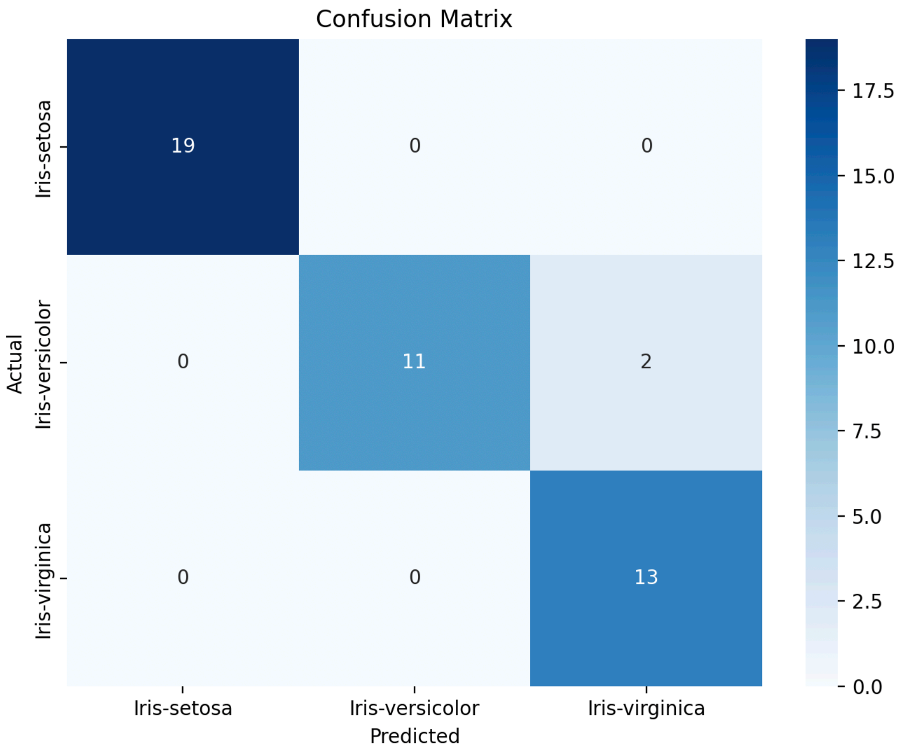

k-val = 11
Iris-versicolor
Iris-setosa
Iris-virginica
18
0
0
0
10
0
0
3
13
Iris-versicolor
Iris-setosa
Iris-virginica
Predicted label
18
16
14
12
10
8
6
4
2
0

3. Considering the dataset presented in Fig. 3, why should k always be an odd number?
K should alway be an odd number to not allow any draws/ties when deciding the class of the element.

Exercise 3 - naive Bayers

1. **Transform by discretizing all columns' values into categories with three possible values (low / medium / high). Use a sensible partition for each column. As in the previous exercise, split the dataset randomly in two subsets (70% / 30%). Repeat the process of the previous exercise to obtain evaluation metrics and an example of a confusion matrix (this time, there is no parameter to vary, so only one cycle of 30 repetitions with different dataset partitions).**

Using the Naive Bayers classifier we get this confusion matrix:



With these classifications:

	precision	recall	f1-score
Iris-setosa	1.00	1.00	1.00
Iris-versicolor	1.00	0.85	0.92
Iris-virginica	0.87	1.00	0.93

Exercise 4 - entropy

Calculate the entropy of the 4 datasets

Calculated entropies:

```
Entropy of full dataset: 0.9182958340544896
Entropy of Low subset: 0.5699613760403499
Entropy of Medium subset: 0.43408112000433474
Entropy of High subset: 0
```

Split:

Information gain from splitting on first column: 0.5586522722255499

The split improves our ability to classify elements of S.

The gain of 0.5586522722255499 indicates that the split effectively reduces uncertainty in classification.

Information gains

Using the entropy of each subset we are able to calculate their information gain:

```
Information Gain for feature 0: 0.5586522722255499
Information Gain for feature 1: 0.3081172725955511
Information Gain for feature 2: 0.9182958340544896
Information Gain for feature 3: 0.9182958340544896
```

The feature with the greatest information gain is feature 2 with a gain of 0.918.