DV2599: Assignment 2

1st Emil Karlström
DVAMI19h
Blekinge Tekniska Högskola
Karlskrona, Sweden
emkl19@student.bth.se

2nd Samuel Jonsson
DVAMI19h
Blekinge Tekniska Högskola
Karlskrona, Sweden
sajs19@student.bth.se

I. Introduction

The the algorithms that shall be tested in this report are

- *K Nearest Neighbour*: A non-parametric algorithm that uses data point proximity to group data points into classes. Usually used for classification, but can also be used for linear regression problems.
- *Decision Tree*[1]: An algorithm that uses a tree-like structure of internal nodes and decision tests that end in "leaf nodes", where each leaf node is a classification of a data point. The data is propagated through the tree through the internal nodes at which you compare the data to a "learned" decision and forward the data to a following node based on the decision taken. This repeats until you reach a leaf node at which point the leaf node represents the classification or regression result of the classifier.
- *Logarithmic Regression Model*: A specialised regression model that attempts to model data where changes in the data happens at a fast pace in the beginning and slows down over time.

II. The Friedman Test

The Friedman test is a statistical test in which the values in each row of a set of data is ranked from best-to-worst performant, usually indicated as 1 being the best and up to N being the worst, assuming the data has N columns. It calculates a test statistic that determines whether there are significant differences in the columns of the data.

III. The Nemenyi Test

Assuming there are significant differences in the data, which is determined in this case by comparing the test statistic acquired by the Friedman Test to an upper critical value based on the level of significance as well as the shape of the data. The Nemenyi test compares all the columns pair-wise and finds out which specific pairs of columns there are significant differences in.

[1]As the DecisionTreeClassifier used when conducting the tests have an element of randomness, on some iterations, there might be small differences in accuracy and F-measure scores. It was however not great enough to significantly affect the tests, so it was decided that it should be ignored

IV. Training Time Performance (Seconds)

TABLE I
Algorithm Performance based on training time

Fold	kNN	Decision Tree	Log-Reg
1	0.0070	0.0163	0.233
2	0.0070	0.0194	0.0252
3	0.0087	0.0162	0.0270
4	0.0083	0.0217	0.0265
5	0.0118	0.0201	0.0310
6	0.0097	0.0188	0.0335
7	0.0109	0.0164	0.0290
8	0.0119	0.0171	0.0256
9	0.0078	0.0215	0.0329
10	0.0080	0.0200	0.0283
Avg.	0.0091	0.0188	0.0282
Std.	0.0017	0.0019	0.0030

TABLE II
Friedman test: Training time Performance

Fold	kNN	Decision Tree	Log-Reg
1	0.0070(3)	0.0163(2)	0.233(1)
2	0.0070(3)	0.0194(2)	0.0252(1)
3	0.0087(3)	0.0162(2)	0.0270(1)
4	0.0083(3)	0.0217(2)	0.0265(1)
5	0.0118(3)	0.0201(2)	0.0310(1)
6	0.0097(3)	0.0188(2)	0.0335(1)
7	0.0109(3)	0.0164(2)	0.0290(1)
8	0.0119(3)	0.0171(2)	0.0256(1)
9	0.0078(3)	0.0215(2)	0.0329(1)
10	0.0080(3)	0.0200(2)	0.0283(1)
Avg. Rank	3	2	1

V. Accuracy (Percentage)

TABLE III
Algorithm Predictive Performance

Fold	kNN	Decision Tree	Log-Reg
1	0.08705	0.8849	0.9065
2	0.9058	0.8696	0.9420
3	0.9203	0.8841	0.9275
4	0.8841	09130	0.9275
5	0.8768	0.8478	0.9276
6	0.8623	0.8551	0.9638
7	0.8986	0.8768	0.9203
8	0.9420	0.9058	0.9348
9	0.9058	0.8841	0.9203
10	0.8696	0.8188	0.9203
Avg.	0.8936	0.8740	0.9291
Std.	0.0230	0.0264	0.0140

TABLE IV
Friedman test: Accuracy

Fold	kNN	Decision Tree	Log-Reg
1	0.08705(3)	0.8849(2)	0.9065(1)
2	0.9058(2)	0.8696(3)	0.9420(1)
3	0.9203(2)	0.8841(3)	0.9275(1)
4	0.8841(3)	09130(2)	0.9275(1)
5	0.8768(2)	0.8478(3)	0.9276(1)
6	0.8623(2)	0.8551(3)	0.9638(1)
7	0.8986(3)	0.8768(3)	0.9203(1)
8	0.9420(1)	0.9058(3)	0.9348(2)
9	0.9058(2)	0.8841(3)	0.9203(1)
10	0.8696(2)	0.8188(3)	0.9203(1)
Avg. Rank	2.1	2.8	1.1

VI. F-measure (Percentage)

TABLE V
F-measure

Fold	kNN	Decision Tree	Log-Reg
1	0.8393	0.8750	0.8960
2	0.8687	0.8624	0.9273
3	0.8911	0.8545	0.9074
4	0.8491	0.9043	0.9123
5	0.8132	0.8214	0.9020
6	0.8403	0.8777	0.9624
7	0.8679	0.8649	0.9009
8	0.9184	0.8932	0.9159
9	0.8807	0.8852	0.9060
10	0.8393	0.8254	0.9091
Avg.	0.8608	0.8664	0.9139
Std.	0.0278	0.0245	0.0173

TABLE VI
Friedman test: F-measure

Fold	kNN	Decision Tree	Log-Reg
1	0.8393(3)	0.8750(2)	0.8960(1)
2	0.8687(2)	0.8624(3)	0.9273(1)
3	0.8911(2)	0.8545(3)	0.9074(1)
4	0.8491(3)	0.9043(2)	0.9123(1)
5	0.8132(3)	0.8214(2)	0.9020(1)
6	0.8403(3)	0.8777(2)	0.9624(1)
7	0.8679(2)	0.8649(3)	0.9009(1)
8	0.9184(1)	0.8932(3)	0.9159(2)
9	0.8807(3)	0.8852(2)	0.9060(1)
10	0.8393(2)	0.8254(3)	0.9091(1)
Avg. Rank	2.4	2.5	1.1

VII. Friedman and Nemenyi results

To determine whether each set of average ranks display significant differences, we calculated the formula:

$$\left(\frac{12}{n*k*(k+1)}\right)*\sum_j(\sum_i(R_i))^2-(3*n*(k+1))$$

Where $\sum_j(\sum_i(R_i))^2$ is the sum of the summed ranks for each test squared, $n=10$ is the number of data sets and $k=3$ is the number of algorithms used for each Friedman test. This gave us the following values:

TABLE VII
Friedman value for each measure

Measure	Friedman Value
Time Performance	20.0
Accuracy	14.59
F-measure	12.59

These were compared with the value found on this table for $n=10$, $k=3$ and $\alpha=0.05$, which results in 6.20. As all Friedman values are greater than this upper critical value, we determined that they displayed significant difference.

The Critical Difference (CD) is calculated according to the formula from the book:

$$CD=q_\alpha\sqrt{\frac{k(k+1)}{6n}}=\approx 1.0478$$

when $\alpha=0.05, k=3$ and $n=10$. q_α is acquired from below equation 12.1 on page 356 in the book.

As all values are the same as in the book, we can assume we reached the correct value. Using the Critical Difference we then compared the difference in value of the average ranks for each Friedman test. this resulted in the following matrix:

TABLE VIII
Whether the algorithm pairs are significant

Algorithms	**Training Time**	**Accuracy**	**F-measure**
kNN & DTC	-	-	-
kNN & Log-Reg	2.0	1.2	1.1
DTC & Log-Reg	-	1.6	1.3

Each each number represents the difference between average ranks of the algorithms where they are greater than the critical difference.