**Ensemble Learning Course**

**Prof. Dr. Mehmet Fatih Amasyalı**

**Homework 02**

**Prepared by Rayene Bech – 18011115**

**02/12/2022**

# Outline

# 1. Introduction

To study the performance of different ensemble learning algorithms, these ten regression datasets were chosen:

1- 2dplanes.arff
2- abalone.arff
3- ailerons.arff
4- autoHorse.arff
5- autoMpg.arff
6- auto_price.arff
7- bank32nh.arff
8- bank8FM.arff
9- elevators.arff
10- housing.arff

One **base regression model** (a regression neural network with a single hidden layer with neurons equal to half of the data features) and four **ensemble algorithms** were analyzed on each of the datasets listed above. Each ensemble algorithm has 10 estimators. The architecture of each estimator is identical to the base model. These ensemble algorithms are:

1- **Bagging**: For each base estimator random **subsets of data** were taking from the whole dataset **with replacement.**
2- **Random Subspaces:** For each base estimator **random subsets of features** were taking from the whole features.
3- **Pasting**: For each base estimator random **subsets of data** were taking from the whole dataset **without replacement.**
4- **AdaBoost:** Starts by fitting one estimator on the dataset, then fitting additional estimators on the same dataset **but focusing on the errors** of the previous estimator's predictions.

# 2. Hyperparameters Tuning:

All the models were trained on Epoch= 200. However, the training is stopped if the validation score doesn't change for successive 5 epochs (Early Stopping). The reported Epoch in the tables shows the value of the epoch when the training was stopped – 5.
As an optimizer, the Adam optimizer was chosen in all experiments.

Dataset 1: **2dplanes.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 1.6 | 94.636 | 4 | 3.6 | 94.861 | 21 |
| 64 | 1.3 | 94.734 | 13 | 2.6 | 94.894 | 32 |
| 128 | 0.7 | 94.754 | 13 | 1.9 | 94.891 | 41 |
| 256 | 0.6 | 94.859 | 18 | 2.6 | 94.888 | 71 |

Dataset 2: **abalone.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 0.7 | 51.165 | 33 | 1.2 | 49.997 | 72 |
| 64 | 0.5 | 50.644 | 52 | 1.2 | 48.604 | 148 |
| 128 | 0.3 | 49.877 | 71 | 0.9 | 46.448 | 200 |
| 256 | 0.5 | 50.618 | 144 | 0.5 | 36.44 | 200 |

Dataset 3: **ailerons.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 0.7 | 80.583 | 6 | 2.2 | 81.857 | 29 |
| 64 | 0.4 | 81.045 | 8 | 1.3 | 81.752 | 34 |
| 128 | 0.5 | 80.919 | 19 | 1.3 | 81.361 | 59 |
| 256 | 0.6 | 80.757 | 37 | 0.7 | 80.699 | 44 |

Dataset 4: **autoHorse.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 0.04 | 73.814 | 20 | 0.2 | 69.251 | 97 |
| 64 | 0.04 | 67.019 | 21 | 0.1 | 67.190 | 122 |
| 128 | 0.02 | 66.30180 | 19 | 0.1 | 68.424 | 134 |
| 256 | 0.02 | 62.985 | 31 | 0.1 | 63.795 | 195 |

Dataset 5: **autoMpg.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 0.1 | 85.809 | 46 | 0.3 | 86.461 | 155 |
| 64 | 0.03 | 77.555 | 22 | 0.2 | 83.6 | 200 |
| 128 | 0.03 | 76.063 | 31 | 0.1 | 65.479 | 129 |
| 256 | 0.04 | 75.423 | 43 | 0.1 | 47.759 | 129 |

Dataset 6: **auto_price.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 0.08 | 80.468 | 72 | 0.01 | -1.85 | 6 |
| 64 | 0.02 | 0.19 | 23 | 0.01 | -2.21 | 10 |
| 128 | 0.007 | -1.54 | 3 | 0.01 | -2.16 | 19 |
| 256 | 0.005 | -1.54 | 3 | 0.01 | -2.16 | 19 |

Dataset 7: **bank32nh.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 0.7 | 59.299 | 15 | 1.2 | 57.919 | 26 |
| 64 | 0.4 | 58.921 | 18 | 0.7 | 57.915 | 38 |
| 128 | 0.3 | 58.794 | 24 | 0.7 | 57.890 | 57 |
| 256 | 0.2 | 56.785 | 17 | 0.6 | 57.123 | 73 |

Dataset 8: **bank8FM.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 0.8 | 94.610 | 20 | 1.3 | 94.532 | 35 |
| 64 | 0.3 | 94.392 | 15 | 1.3 | 94.569 | 69 |
| 128 | 0.5 | 94.585 | 42 | 1.2 | 95.023 | 135 |
| 256 | 0.4 | 94.494 | 69 | 1.1 | 94.7558 | 200 |

Dataset 9: **elevators.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 1.9 | 88.398 | 22 | 3.1 | 86.769 | 37 |
| 64 | 1.4 | 88.833 | 37 | 1.5 | 85.447 | 40 |
| 128 | 1.1 | 88.106 | 57 | 1.2 | 85.084 | 68 |
| 256 | 0.69 | 85.276 | 58 | 1.4 | 85.386 | 127 |

Dataset 10: **housing.arff**

| lr | 0.01 | | | 0.001 | | |
|---|---|---|---|---|---|---|
| Batch Size | Time (s) | Val Score(%) | Epoch | Time (s) | Val Score(%) | Epoch |
| 32 | 0.05 | 83.179 | 17 | 0.2 | 82.753 | 32 |
| 64 | 0.05 | 84.288 | 32 | 0.2 | 81.413 | 152 |
| 128 | 0.05 | 82.631 | 48 | 0.2 | 77.034 | 195 |
| 256 | 0.05 | 81.076 | 63 | 0.1 | 65.477 | 195 |

Afterwards, all the experiments related to the datasets are configured with the hyperparameters highlighted in green.

# 3. T-test results

To compare the performance of models and different ensemble models, a 10-fold 5*2 t-test was implemented.

In each iteration of this t-test, we compare two models A and B. Firstly, we assume that model A and model B test loss values are drawn from a normal distribution (with unknown variance). We define two hypothesizes accordingly:

- **Null hypothesis**: The loss values of A and B are drawn from the same distribution (Since they were trained and tested on the same datasets). Hence, the difference in the loss values has an expected value equal to 0 (E[diff]=0). This implies that there is no difference between the two models.

- **Alternative hypothesis**: The loss values of A and B are drawn from two different distributions, i.e. E[diff] ≠ 0. Thus, the two models are different, and one is performing better than the other.

We define a confidence level of 95%. This means the value of alpha is:

alpha = 100% - 95% = 5% = 0.05

For each iteration of the 10 (5*2) K-fold iterations, we train and test the 2 models A and B. If model A performs better than model B, we add +1 to the "Win" value. Otherwise, we add +1 to the "Loss" value.

After testing on all iterations, we compute the p-value and the loss mean of each model.

- **If p-value < alpha**: we reject the Null Hypothesis ➔ The two models are different. Then we decide which model is better by comparing the mean losses.

- **If p-value > alpha**: We accept the Null hypothesis**.**

| | Boosting Vs NN | | | | | Bagging Vs NN | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Datasets | w | l | P-value | e-mean | b-mean | w | l | P-value | e-mean | b-mean |
| **2dplanes.arff** | 10 | 0 | 0.002 | 0.002 | 0.003 | 0 | 10 | 0.0133 | 0.003 | 0.002 |
| **abalone.arff** | 0 | 10 | 0.742 | 0.007 | 0.006 | 6 | 4 | 0.0519 | 0.006 | 0.006 |
| **ailerons.arff** | 10 | 0 | 0.035 | 0.002 | 0.003 | 10 | 0 | 0.0895 | 0.002 | 0.002 |
| **autoHorse.arff** | 10 | 0 | 0.263 | 0.005 | 0.008 | 10 | 0 | 0.0425 | 0.004 | 0007 |
| **autoMpg.arff** | 10 | 0 | 0.005 | 0.007 | 0.011 | 10 | 0 | 0.103 | 0.007 | 0.010 |
| **auto_price.arff** | 10 | 0 | 0.823 | 0.009 | 0.025 | 9 | 1 | 0.024 | 0.009 | 0.01 |
| **bank32nh.arff** | 2 | 8 | 0.115 | 0.010 | 0.010 | 10 | 0 | 0.587 | 0.009 | 0.011 |
| **bank8FM.arff** | 6 | 4 | 0.843 | 0.002 | 0.002 | 10 | 0 | 0.002 | 0.002 | 0.003 |
| **elevators.arff** | 8 | 2 | 0.127 | 0.001 | 0.001 | 7 | 3 | 0.298 | 0.002 | 0.002 |
| **housing.arff** | 0 | 10 | 0.218 | 0.017 | 0.013 | 0 | 10 | 0.106 | 0.013 | 0.012 |
| **(win-tie-loss)** | | | 3-7-0 | | | | | 3-6-1 | | |

| | Random Subspace Vs NN | | | | | Pasting Vs NN | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Datasets | w | l | P-value | e-mean | b-mean | w | l | P-value | e-mean | b-mean |
| **2dplanes.arff** | 0 | 10 | 6.6e-6 | 0.006 | 0.003 | 0 | 10 | 0.002 | 0.003 | 0.001 |
| **abalone.arff** | 6 | 4 | 0.607 | 0.006 | 0.006 | 5 | 5 | 0.575 | 0.006 | 0.006 |
| **ailerons.arff** | 0 | 10 | 0.001 | 0.0032 | 0.0025 | 10 | 0 | 0.496 | 0.002 | 0.002 |
| **autoHorse.arff** | 10 | 0 | 0.163 | 0.005 | 0.007 | 10 | 0 | 0.024 | 0.004 | 0.007 |
| **autoMpg.arff** | 10 | 0 | 0.042 | 0.007 | 0.012 | 10 | 0 | 0.017 | 0.006 | 0.010 |
| **auto_price.arff** | 10 | 0 | 0.785 | 0.011 | 0.025 | 10 | 0 | 0.022 | 0.009 | 0.012 |
| **bank32nh.arff** | 0 | 10 | 0.003 | 0.012 | 0.010 | 10 | 0 | 0.834 | 0.009 | 0.011 |
| **bank8FM.arff** | 0 | 10 | 1.9e-6 | 0.009 | 0.002 | 10 | 0 | 0.0003 | 0.002 | 0.003 |
| **elevators.arff** | 0 | 10 | 0.008 | 0.002 | 0.001 | 7 | 3 | 0.290 | 0.002 | 0.002 |
| **housing.arff** | 0 | 10 | 0.003 | 0.026 | 0.013 | 1 | 9 | 0.053 | 0.013 | 0.012 |
| **(win-tie-loss)** | | | 1-3-6 | | | | | 4-5-1 | | |

| | Boosting Vs Bagging | | | | | Boosting Vs Random Subspace | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Datasets | w | l | P-value | boosting-mean | bagging-mean | w | l | P-value | boosting-mean | RS-mean |
| **2dplanes.arff** | 10 | 0 | 0.006 | 0.001 | 0.003 | 10 | 0 | 2.3e-8 | 0.001 | 0.005 |
| **abalone.arff** | 0 | 10 | 0.775 | 0.007 | 0.006 | 0 | 10 | 0.858 | 0.007 | 0.006 |
| **ailerons.arff** | 6 | 4 | 0.283 | 0.002 | 0.002 | 10 | 0 | 0.001 | 0.002 | 0.003 |
| **autoHorse.arff** | 1 | 9 | 0.832 | 0.005 | 0.005 | 1 | 9 | 0.713 | 0.005 | 0.005 |
| **autoMpg.arff** | 6 | 4 | 0.645 | 0.007 | 0.007 | 5 | 5 | 0.409 | 0.007 | 0.007 |
| **auto_price.arff** | 8 | 2 | 0.852 | 0.009 | 0.014 | 7 | 3 | 0.957 | 0.009 | 0.011 |
| **bank32nh.arff** | 1 | 9 | 0.054 | 0.010 | 0.009 | 10 | 0 | 0.034 | 0.010 | 0.012 |
| **bank8FM.arff** | 1 | 9 | 0.270 | 0.002 | 0.002 | 10 | 0 | 4e-7 | 0.002 | 0.009 |
| **elevators.arff** | 10 | 0 | 0.168 | 0.001 | 0.001 | 10 | 0 | 0.001 | 0.001 | 0.002 |
| **housing.arff** | 1 | 9 | 0.461 | 0.017 | 0.014 | 10 | 0 | 0.059 | 0.017 | 0.026 |
| **(win-tie-loss)** | | | 1-9-0 | | | | | 5-5-0 | | |

| Datasets | Pasting Vs Bagging | | | | | Pasting Vs Boosting | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | w | l | P-value | pasting-mean | bagging-mean | w | l | P-value | pasting-mean | boosting-mean |
| **2dplanes.arff** | 4 | 6 | 0.911 | 0.003 | 0.003 | 0 | 10 | 0.020 | 0.003 | 0.001 |
| **abalone.arff** | 6 | 4 | 0.397 | 0.006 | 0.006 | 9 | 1 | 0.962 | 0.006 | 0.007 |
| **ailerons.arff** | 5 | 5 | 0.417 | 0.002 | 0.002 | 5 | 5 | 0.494 | 0.002 | 0.002 |
| **autoHorse.arff** | 8 | 2 | 0.139 | 0.004 | 0.004 | 9 | 1 | 0.178 | 0.004 | 0.005 |
| **autoMpg.arff** | 5 | 5 | 0.573 | 0.007 | 0.007 | 6 | 4 | 0.245 | 0.007 | 0.007 |
| **auto_price.arff** | 7 | 3 | 0.961 | 0.012 | 0.014 | 2 | 8 | 0.791 | 0.013 | 0.009 |
| **bank32nh.arff** | 2 | 8 | 0.061 | 0.010 | 0.009 | 9 | 1 | 0.076 | 0.010 | 0.010 |
| **bank8FM.arff** | 6 | 4 | 0.899 | 0.002 | 0.002 | 8 | 2 | 0.298 | 0.002 | 0.002 |
| **elevators.arff** | 5 | 5 | 0.677 | 0.001 | 0.001 | 1 | 9 | 0.154 | 0.002 | 0.001 |
| **housing.arff** | 4 | 6 | 0.696 | 0.014 | 0.014 | 9 | 1 | 0.74 | 0.014 | 0.017 |
| **(win-tie-loss)** | | | 0-10-0 | | | | | 0-9-1 | | |

| Datasets | Bagging Vs Random Subspace | | | | | Pasting Vs Random Subspace | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | w | l | P-value | bagging-mean | RS-mean | w | l | P-value | pasting-mean | RS-mean |
| **2dplanes.arff** | 10 | 0 | 6.6e-5 | 0.003 | 0.006 | 10 | 0 | 0.0003 | 0.003 | 0.005 |
| **abalone.arff** | 3 | 7 | 0.759 | 0.006 | 0.006 | 6 | 4 | 0.456 | 0.006 | 0.006 |
| **ailerons.arff** | 10 | 0 | 0.0002 | 0.002 | 0.003 | 10 | 0 | 0.0005 | 0.002 | 0.003 |
| **autoHorse.arff** | 4 | 6 | 0.897 | 0.005 | 0.005 | 7 | 3 | 0.193 | 0.004 | 0.005 |
| **autoMpg.arff** | 7 | 3 | 0.320 | 0.007 | 0.007 | 6 | 4 | 0.611 | 0.007 | 0.007 |
| **auto_price.arff** | 3 | 7 | 0.786 | 0.014 | 0.011 | 3 | 7 | 0.687 | 0.012 | 0.011 |
| **bank32nh.arff** | 10 | 0 | 4.1e-5 | 0.009 | 0.012 | 10 | 0 | 0.0001 | 0.010 | 0.012 |
| **bank8FM.arff** | 10 | 0 | 2.2e-6 | 0.002 | 0.009 | 10 | 0 | 4.7e-6 | 0.002 | 0.009 |
| **elevators.arff** | 10 | 0 | 8.5e-5 | 0.001 | 0.002 | 10 | 0 | 9.1e-5 | 0.001 | 0.002 |
| **housing.arff** | 10 | 0 | 0.027 | 0.014 | 0.026 | 10 | 0 | 0.039 | 0.014 | 0.026 |
| **(win-tie-loss)** | | | 6-4-0 | | | | | 6-4-0 | | |

## Notes:

- Pasting and Bagging models almost performed the same on all datasets: The data replacement didn't change the performance of the Bagging model.

- The Random Subspace model performed poorly compared to the base model (won only one time and lost 6 times).

- The Random Subspace model was first trained by choosing 50% of the features for each estimator. But the performance of the model was bad for almost all the datasets. Hence, we trained again this ensemble model using 70% of the features for each estimator.

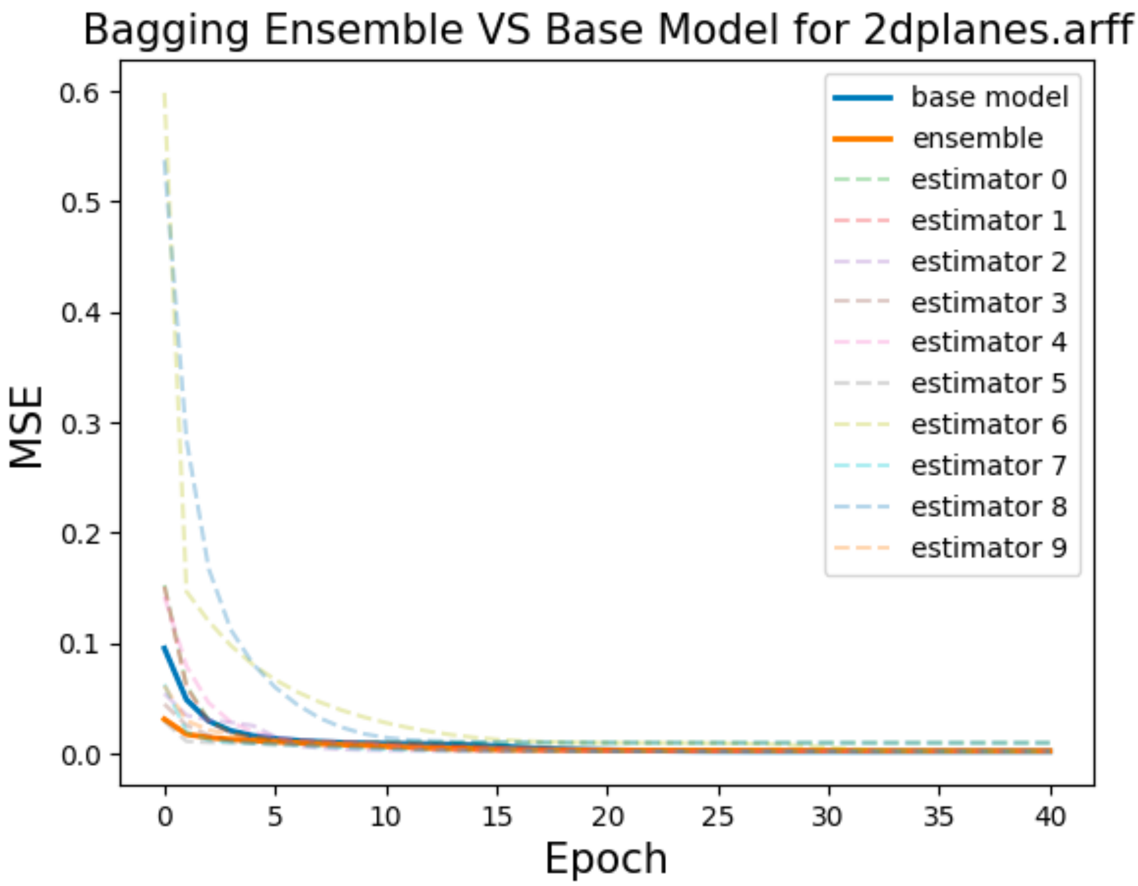
- Bagging, Pasting and Boosting methods performed better than Random Subspace model in almost half of the datasets.

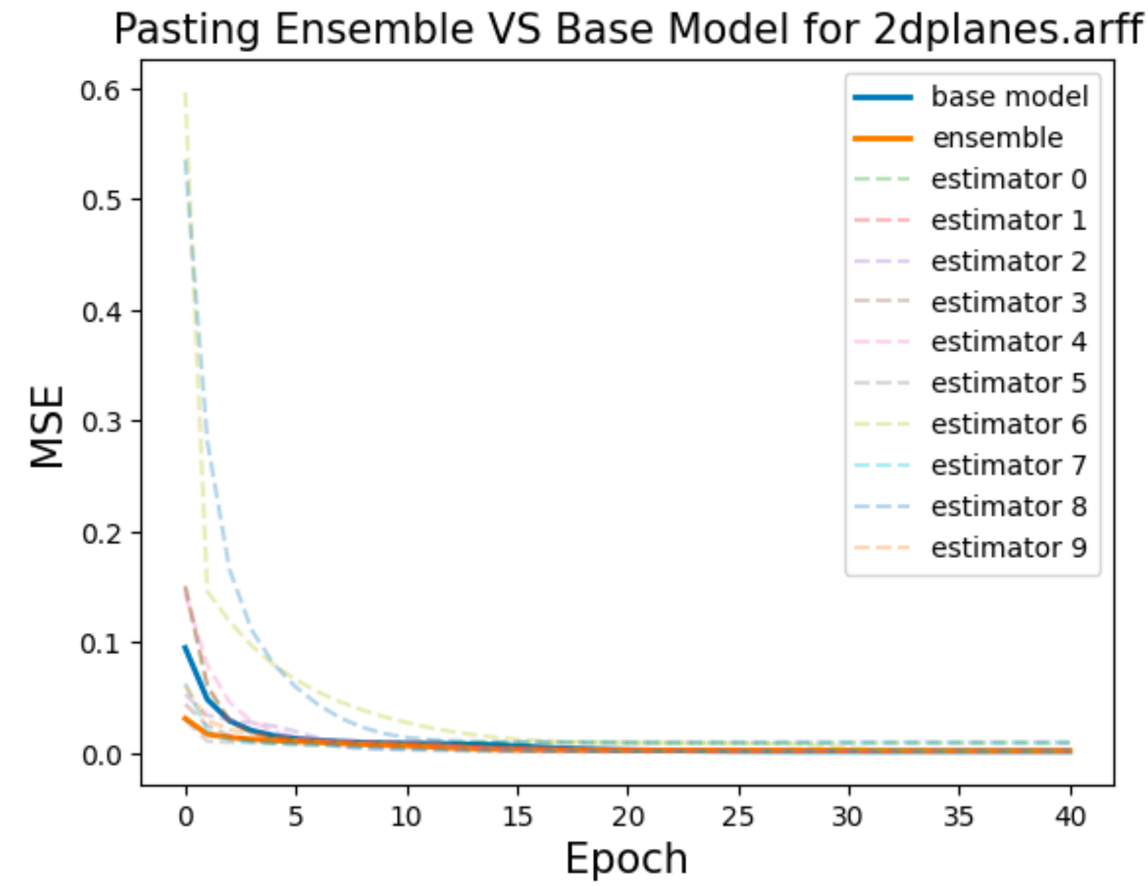
- There was a slight difference between the performance of the Boosting model and the Bagging model.

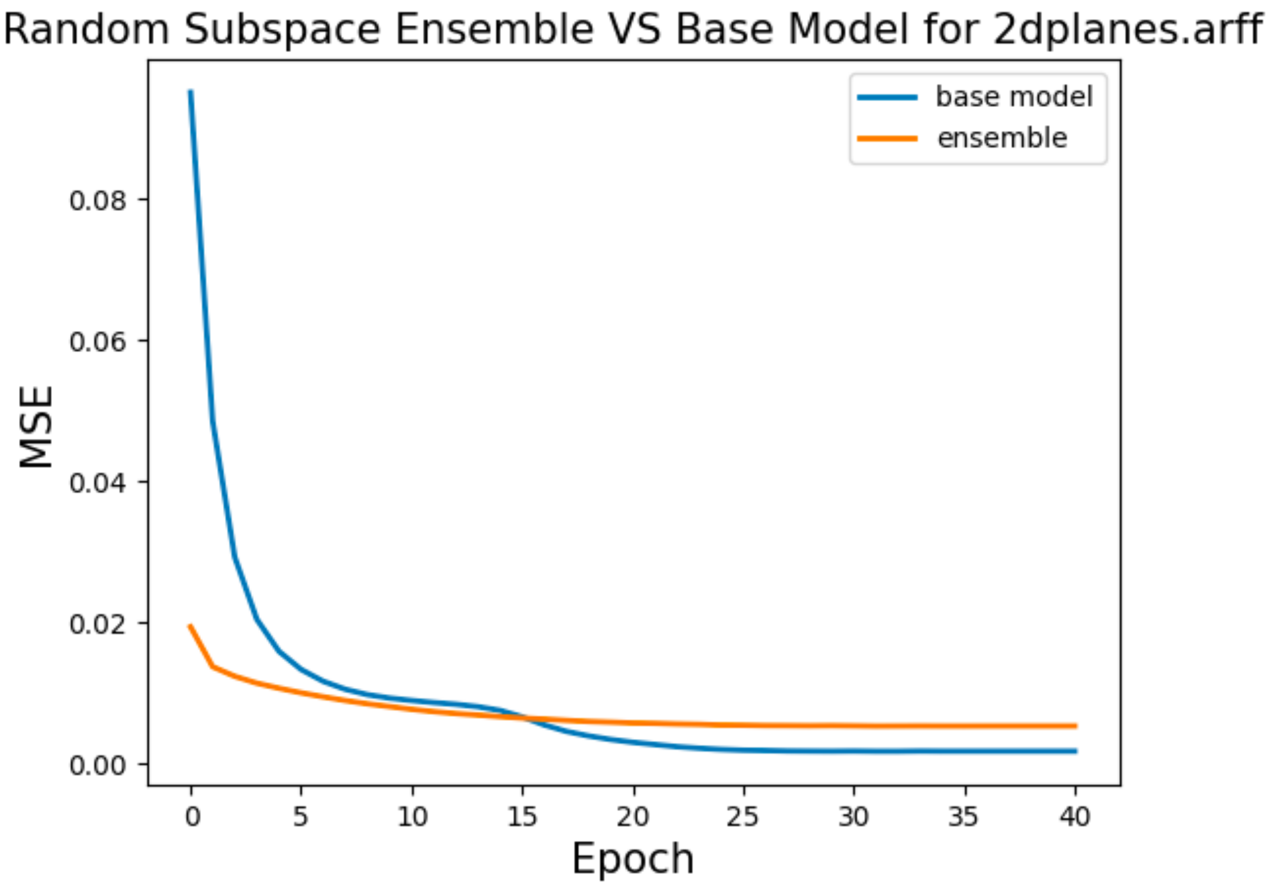
# 4. Test Loss/Epoch Graphs (MSE):

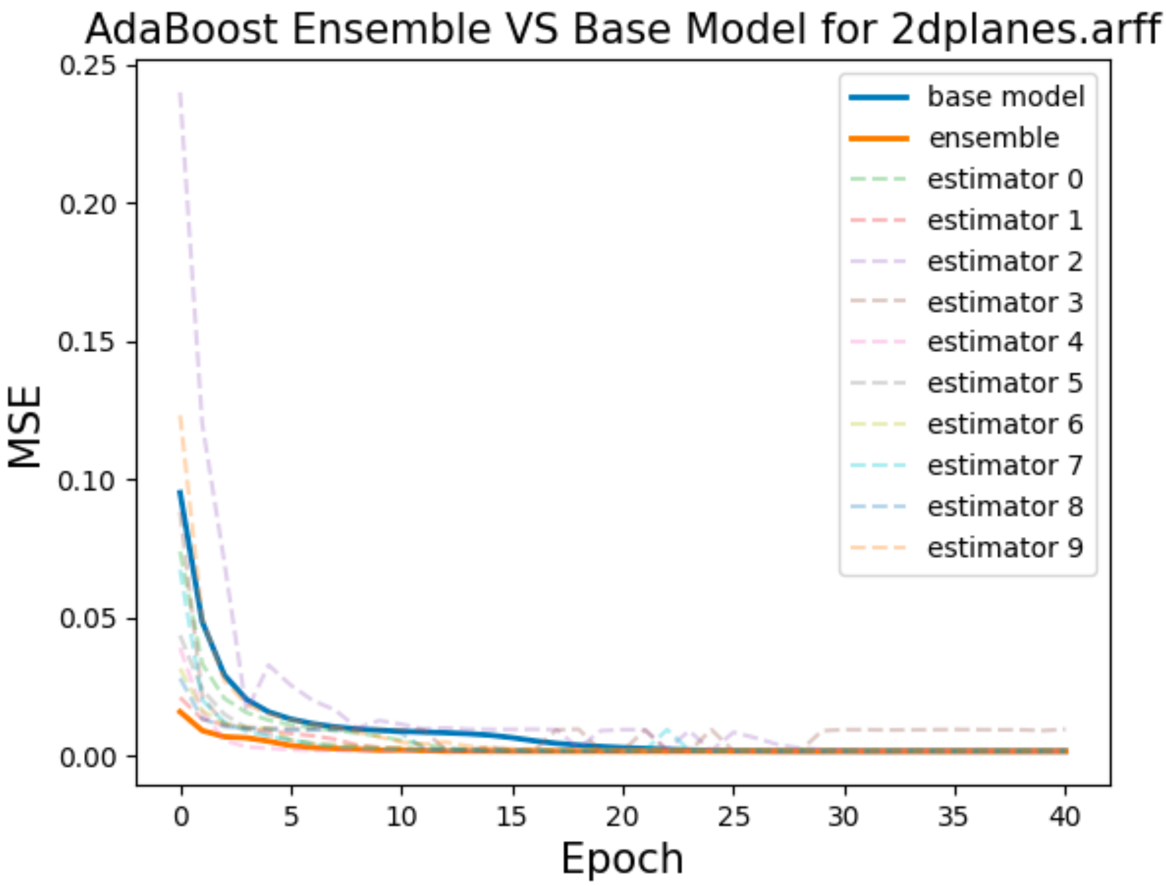
- Two datasets were chosen for this section: “2dplanes.arff” and “elevators.arff”.
- The metric used to compute the loss in these graphs is the Mean Squared Error (MSE).

- **Dataset 01: 2dplanes.arff:**



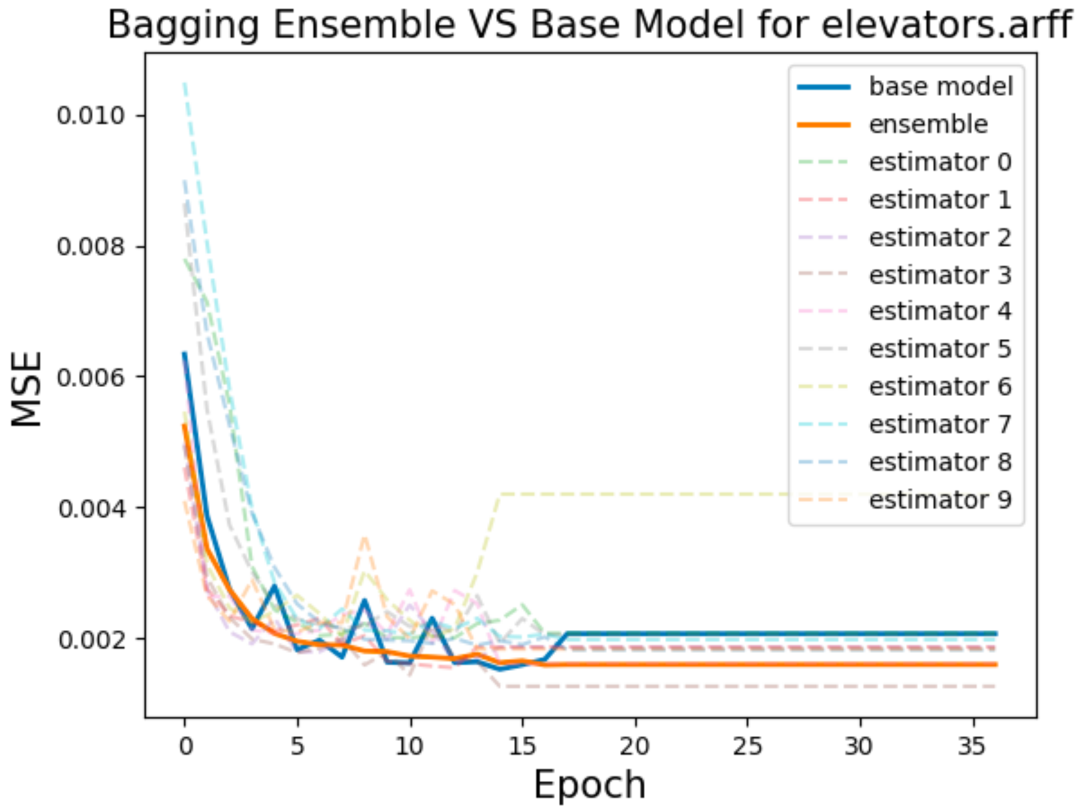

Pasting Ensemble VS Base Model for 2dplanes.arff
MSE
Epoch
base model
ensemble
estimator 0
estimator 1
estimator 2
estimator 3
estimator 4
estimator 5
estimator 6
estimator 7
estimator 8
estimator 9
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20
25
30
35
40

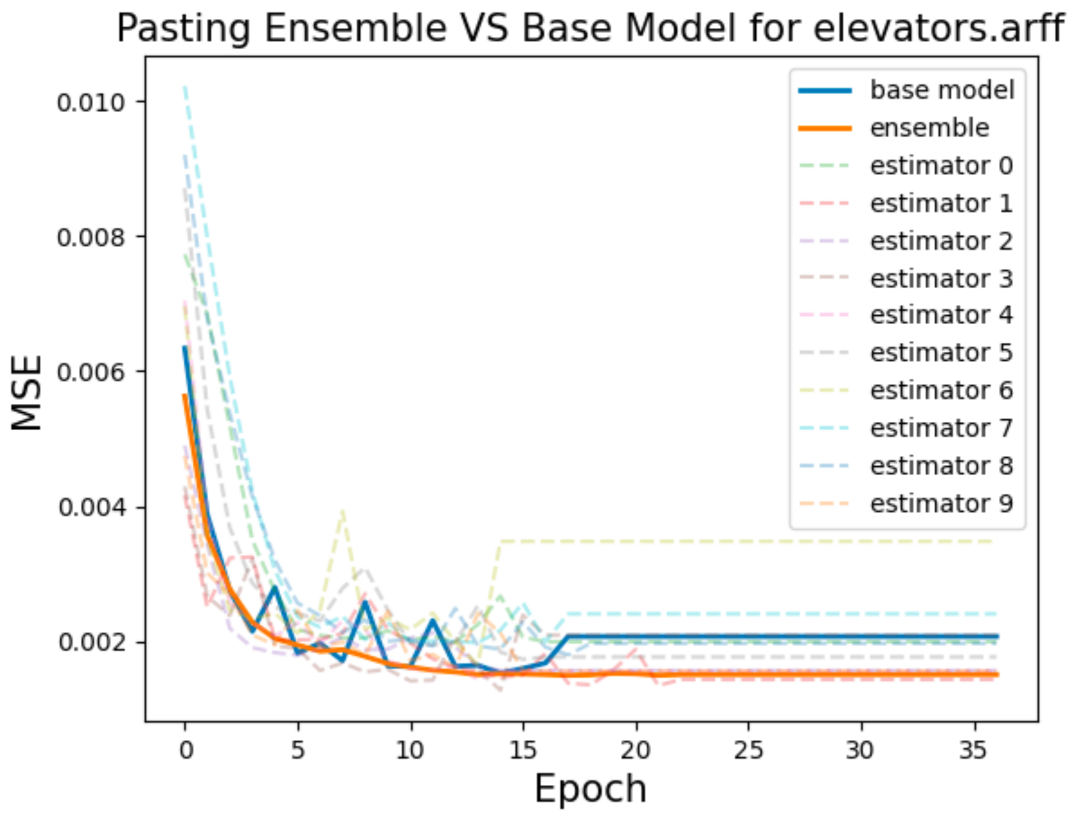

Random Subspace Ensemble VS Base Model for 2dplanes.arff
MSE
Epoch
base model
ensemble
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25
30
35
40

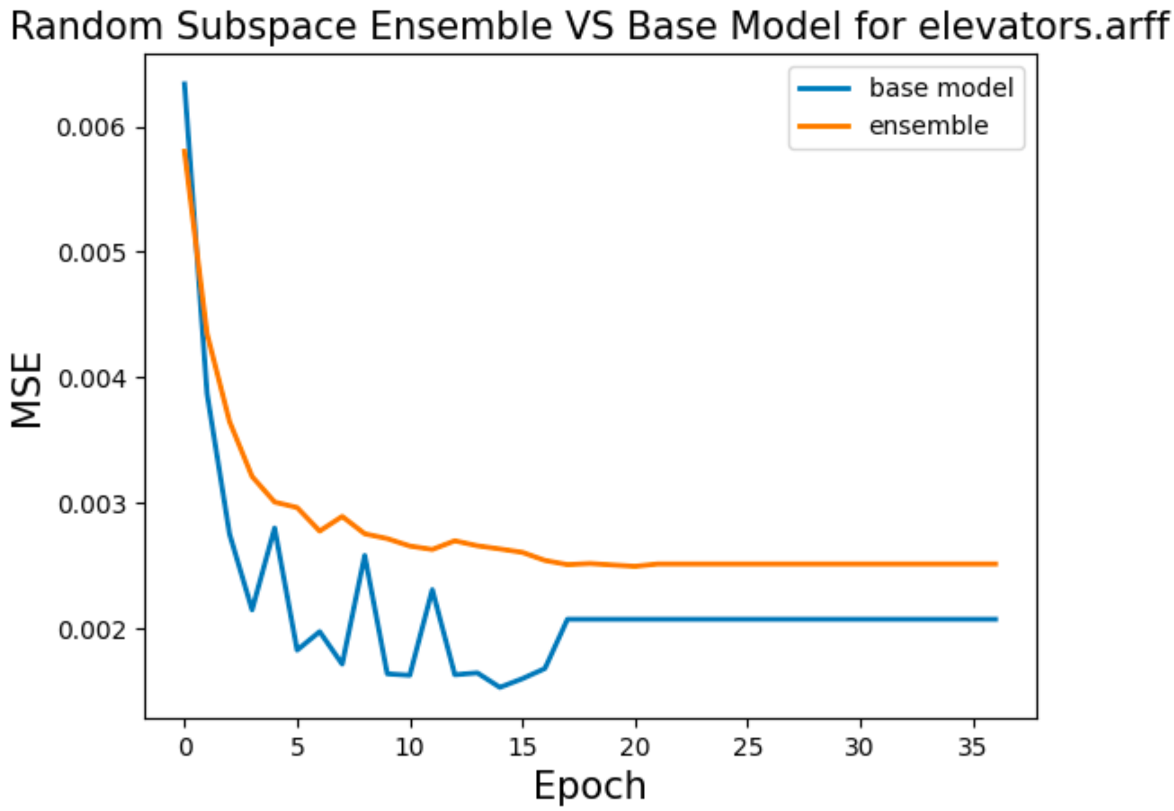


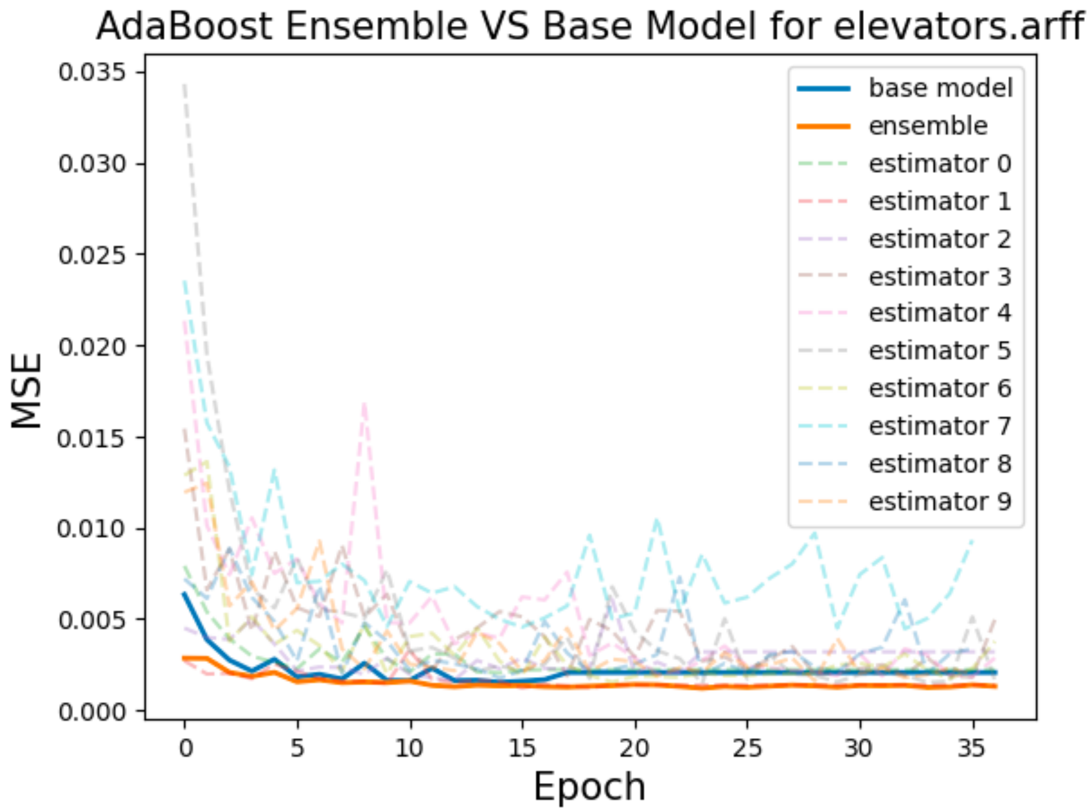

- We notice that for all models, starting from the earliest epochs, the ensemble model's loss is less than the base model's loss. (Fast Convergence)
- The loss of the ensemble was lower than the loss of its weak learners (estimators) in the first epochs despite some of them had very big loss values.
- The weak learners of the Adaboost ensemble had lower loss values compared to the bagging 's weak learners (Having much training data?)

- **Dataset 02: elevators.arff**





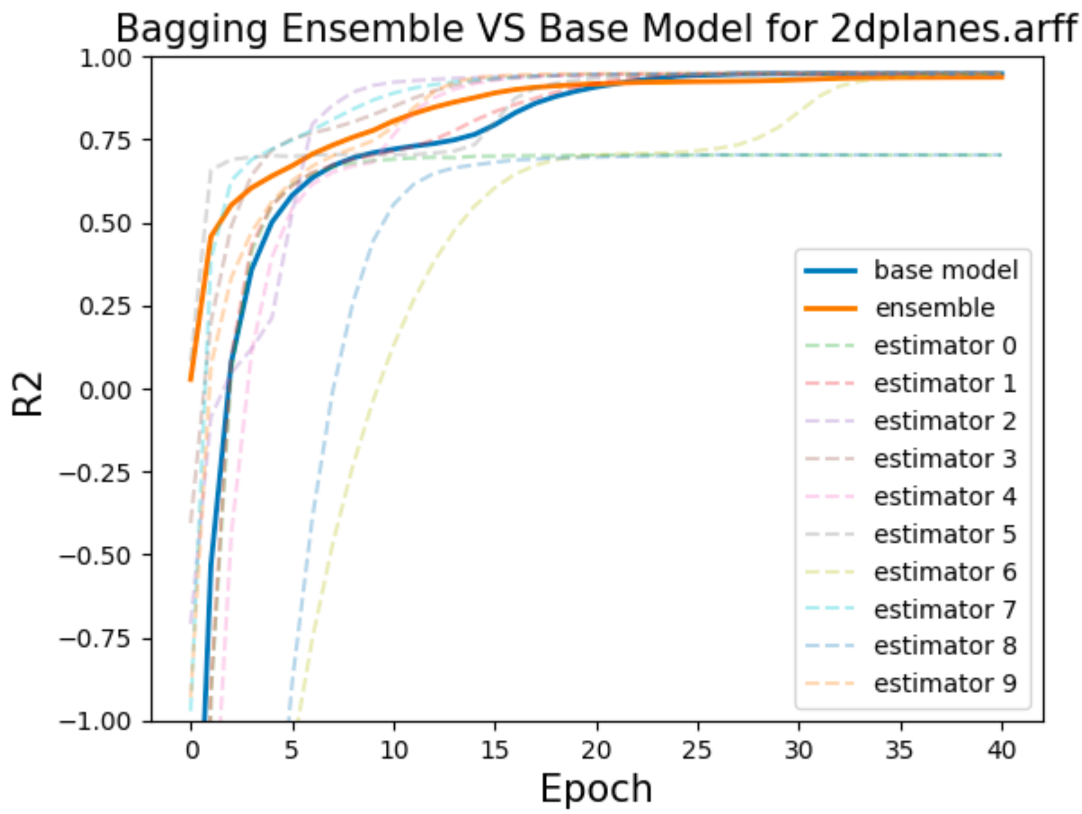



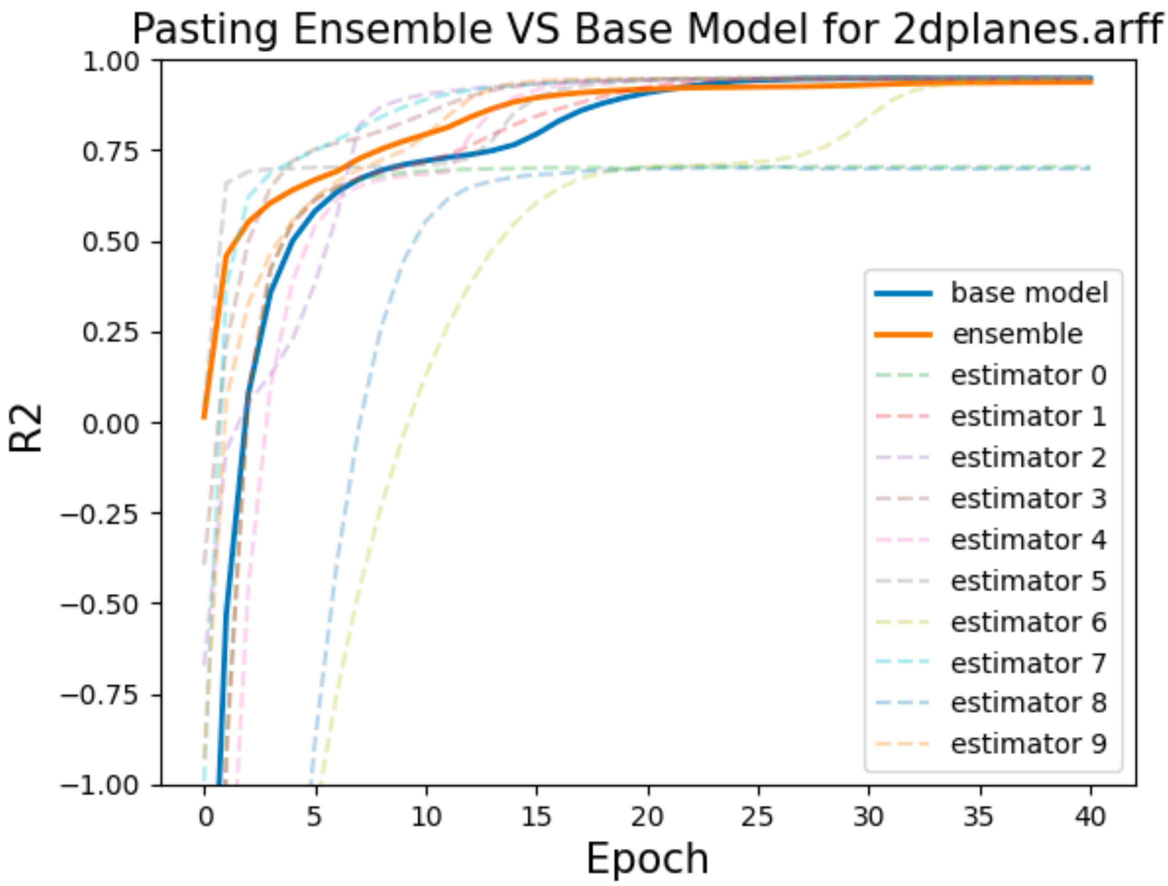

- This dataset shows clearly that the Random Subspace performed worser than the base model.
- AdaBoost and Bagging models had very close loss values to the base model.
- Again we see that Adaboost model converged faster than other ensemble methods and even faster than the bagging models.

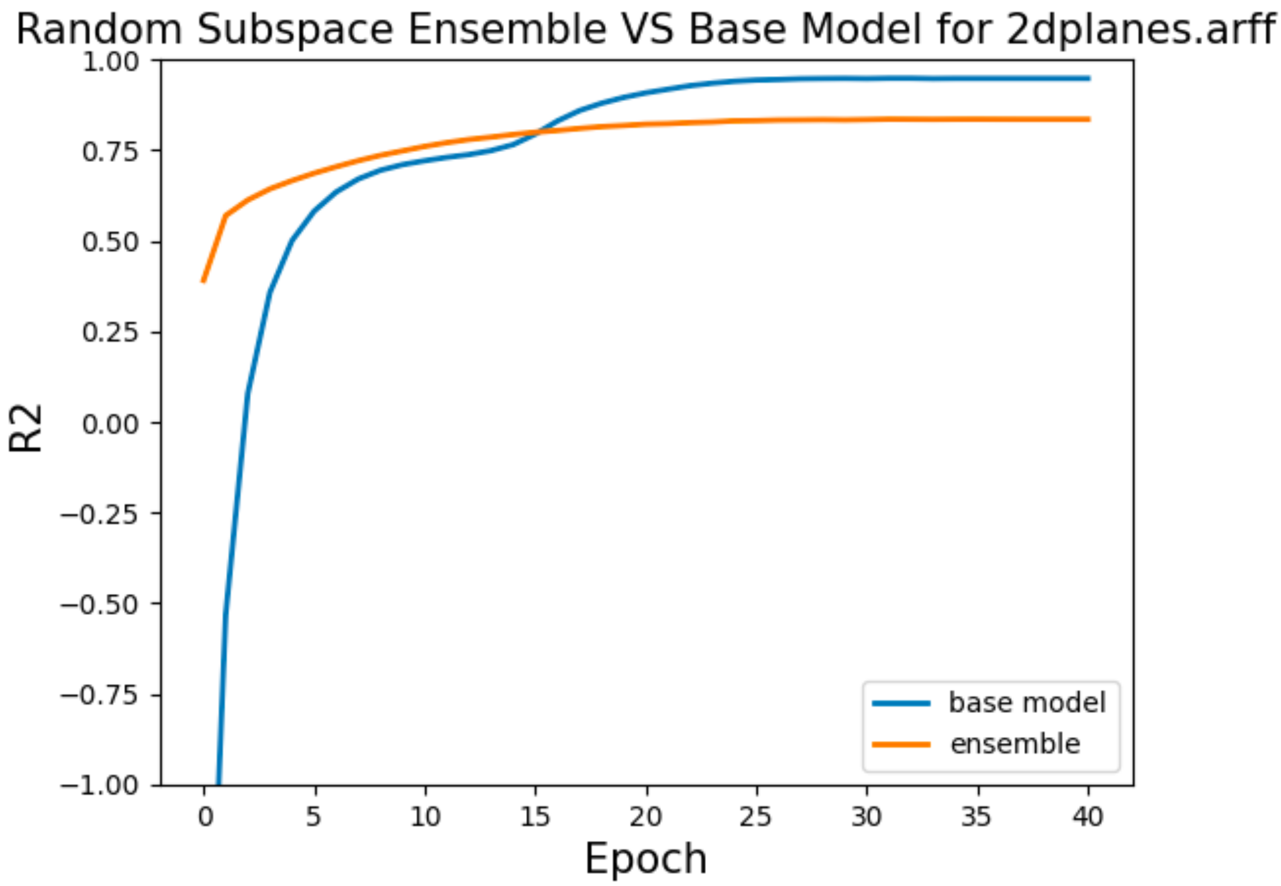
# 5. Test Score/Epoch graphs (R2)

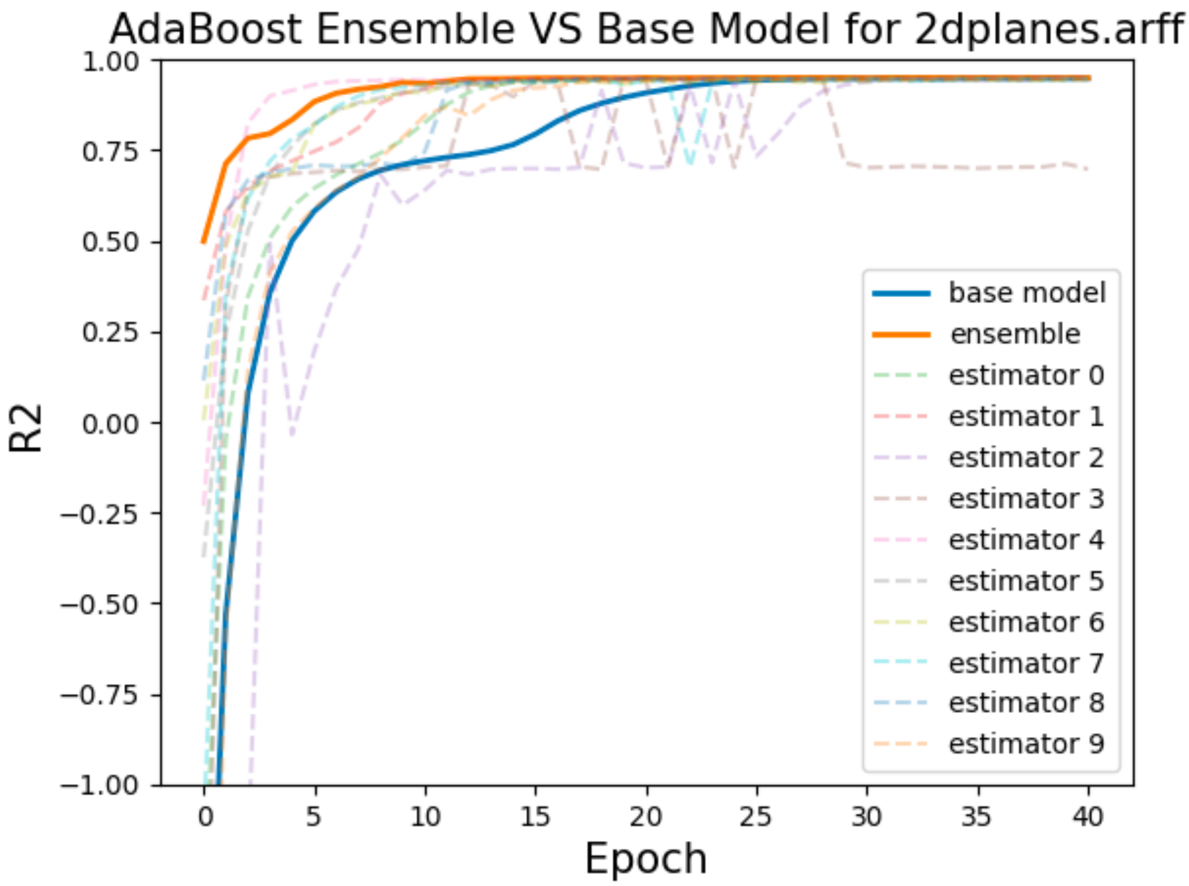
- The same previous two datasets were used in this section: "2dplanes.arff" and "elevators.arff".
- The metric used to compute the loss in these graphs is R-Squared (R2).

- **Dataset 01: 2dplanes.arff:**

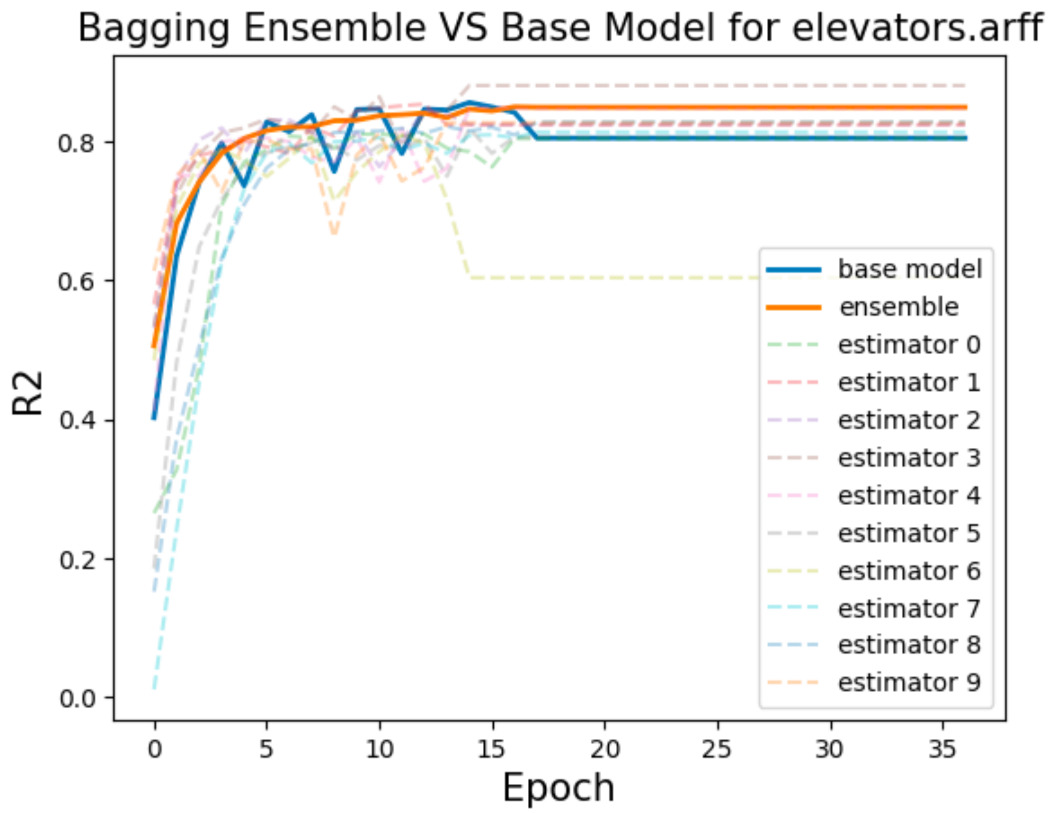


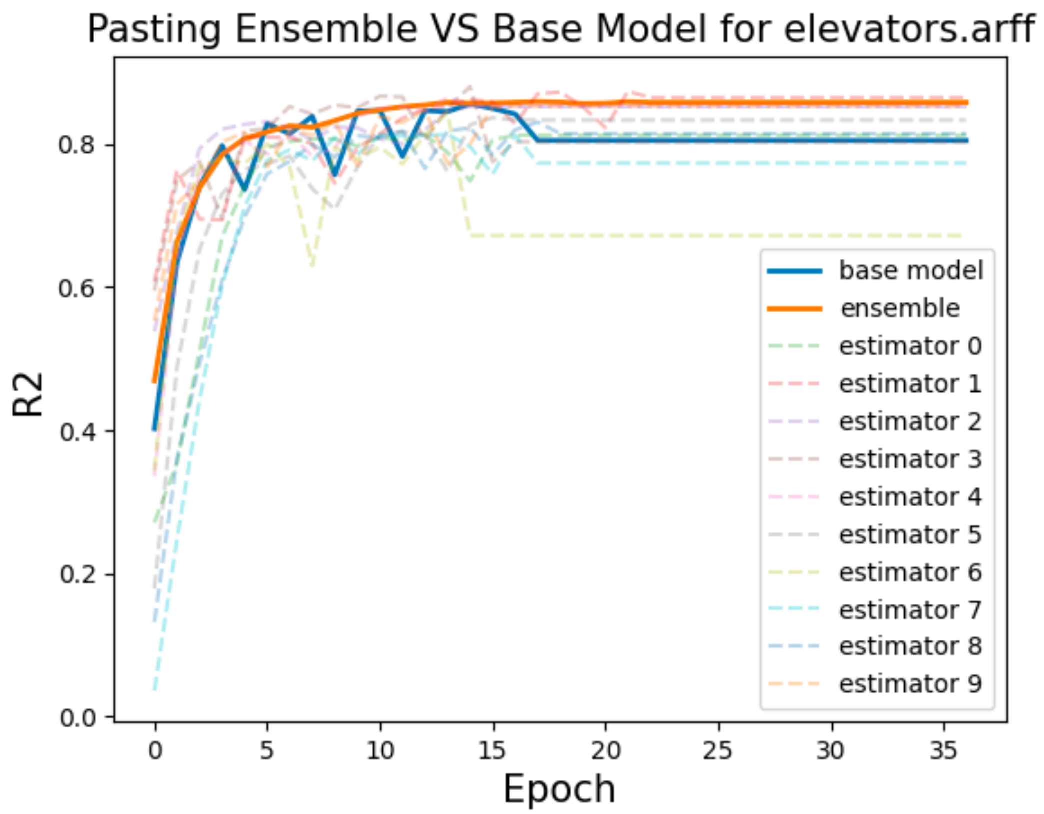

Pasting Ensemble VS Base Model for 2dplanes.arff
R2
Epoch
base model
ensemble
estimator 0
estimator 1
estimator 2
estimator 3
estimator 4
estimator 5
estimator 6
estimator 7
estimator 8
estimator 9

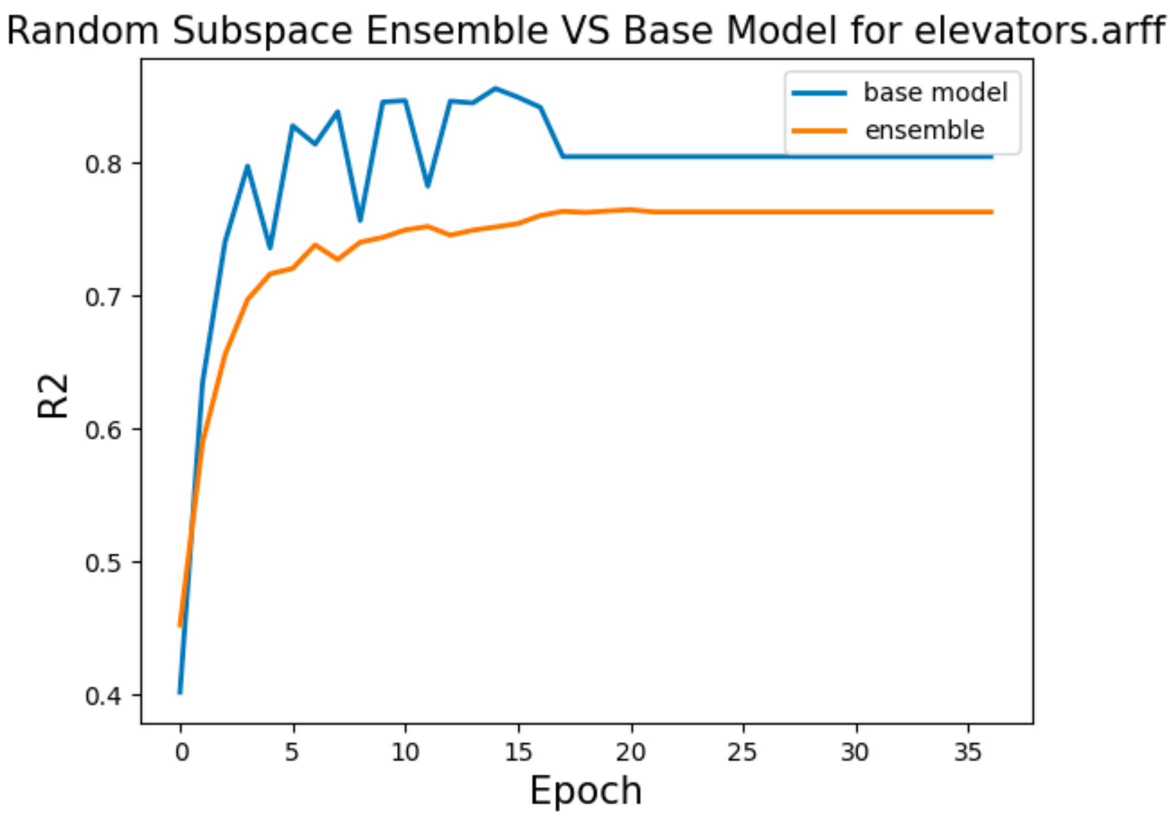

Random Subspace Ensemble VS Base Model for 2dplanes.arff
R2
Epoch
base model
ensemble

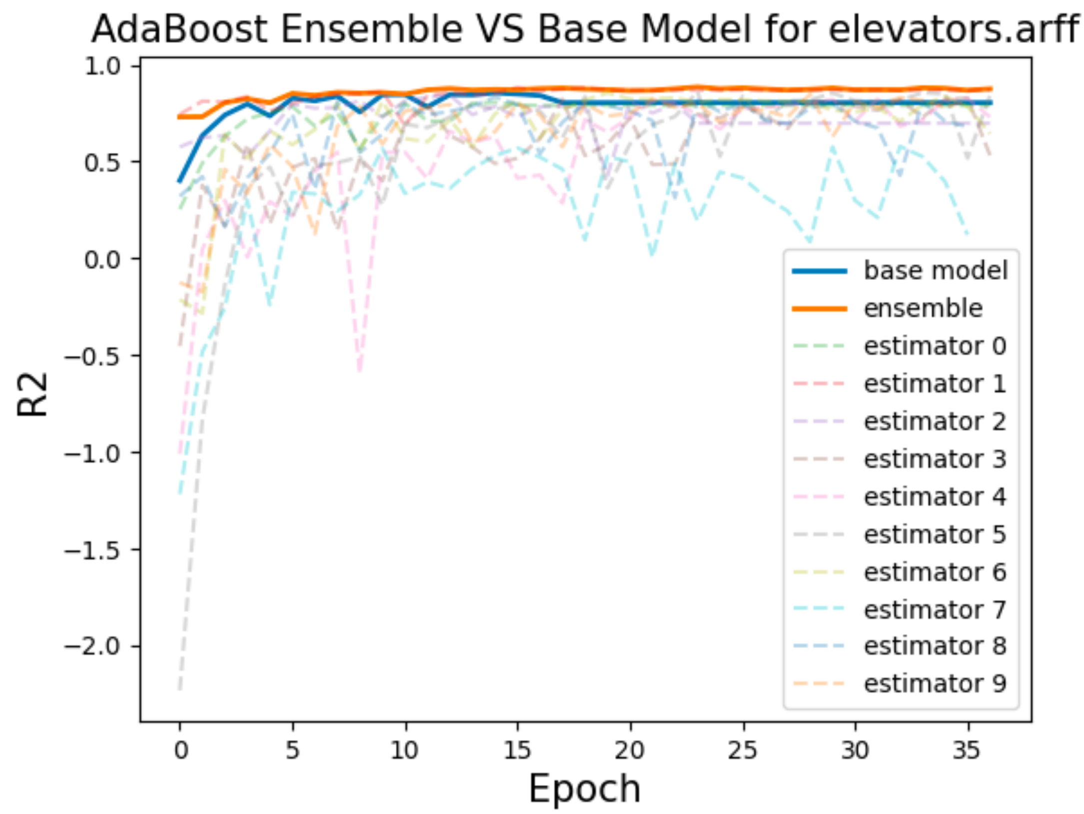

AdaBoost Ensemble VS Base Model for 2dplanes.arff
R2
Epoch
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
15
20
25
30
35
40
base model
ensemble
estimator 0
estimator 1
estimator 2
estimator 3
estimator 4
estimator 5
estimator 6
estimator 7
estimator 8
estimator 9

**Dataset 02: elevators.arff**



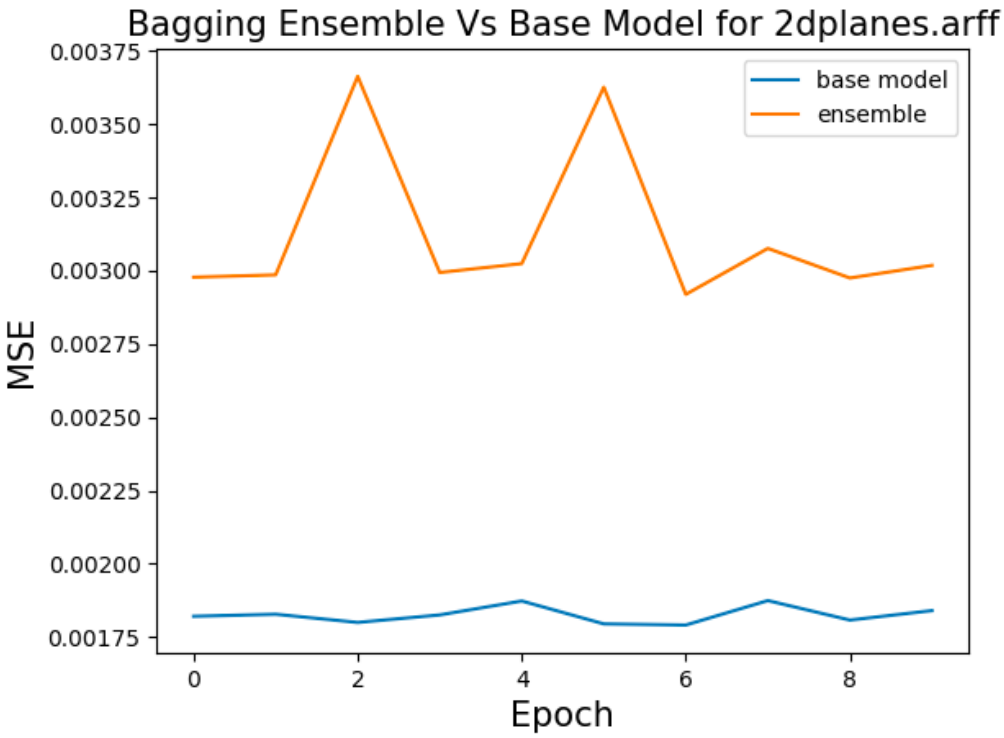

Random Subspace Ensemble VS Base Model for elevators.arff
base model
ensemble
R2
0.4
0.5
0.6
0.7
0.8
0
5
10
15
20
25
30
35
Epoch

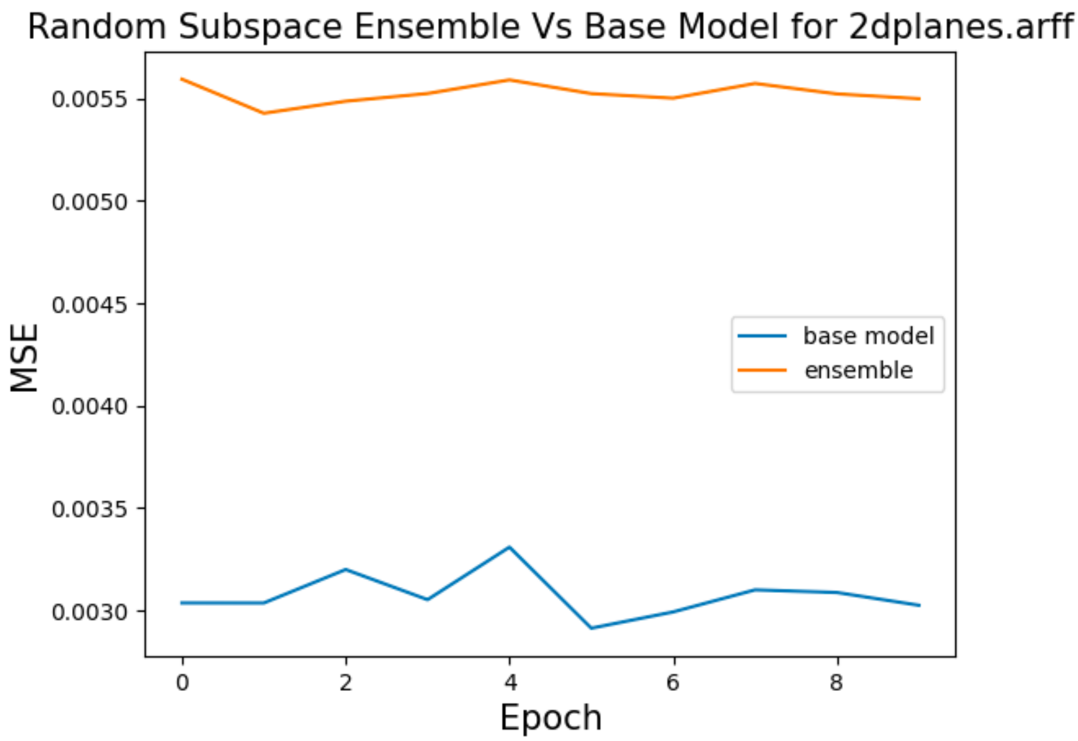

AdaBoost Ensemble VS Base Model for elevators.arff
base model
ensemble
estimator 0
estimator 1
estimator 2
estimator 3
estimator 4
estimator 5
estimator 6
estimator 7
estimator 8
estimator 9
R2
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
0
5
10
15
20
25
30
35
Epoch

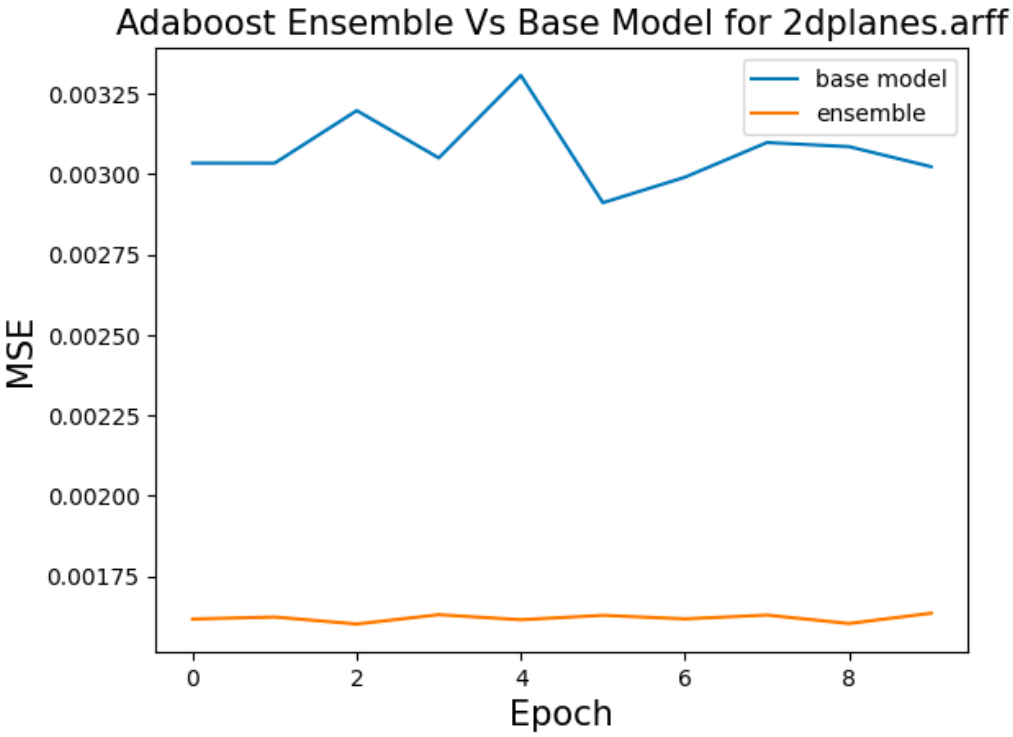
# 6. Graphs for Test Loss / K-Fold Iteration

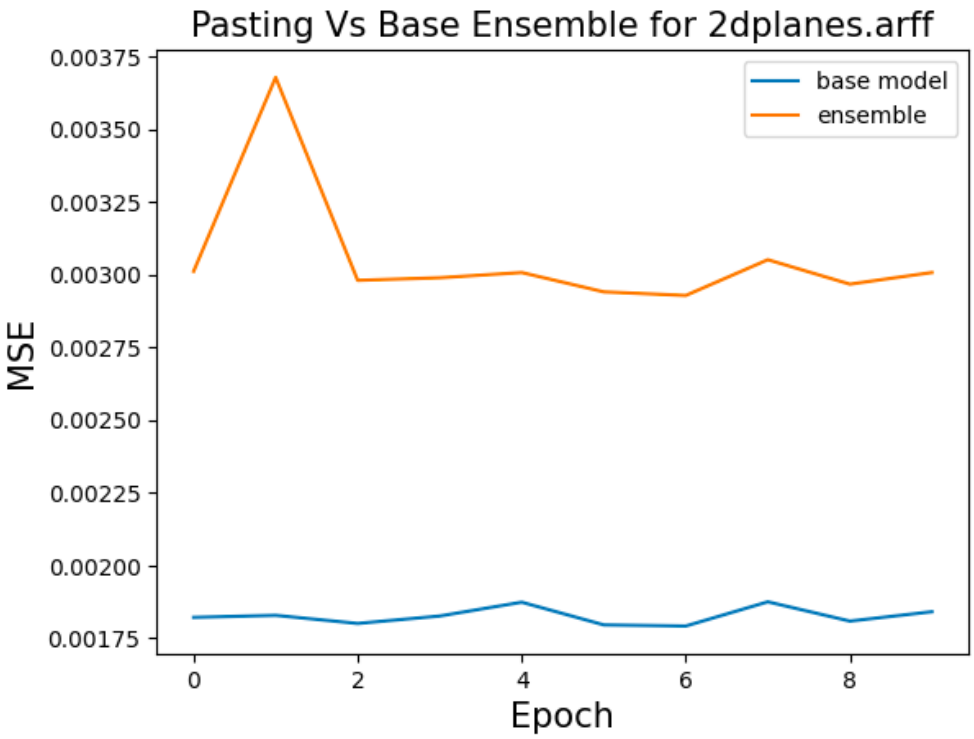
After each test iteration in the K-Fold 5*2 t-test, the test loss of both the base model and the ensemble models were plotted for the same 2 datasets.

- **Dataset 01: 2dplanes.arff:**

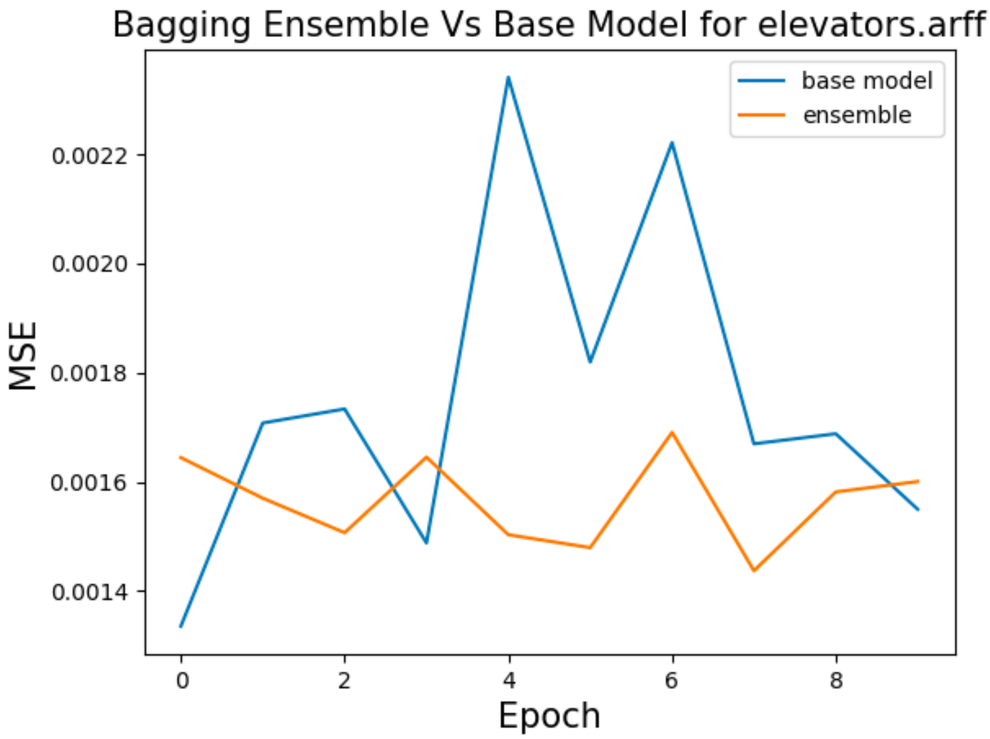


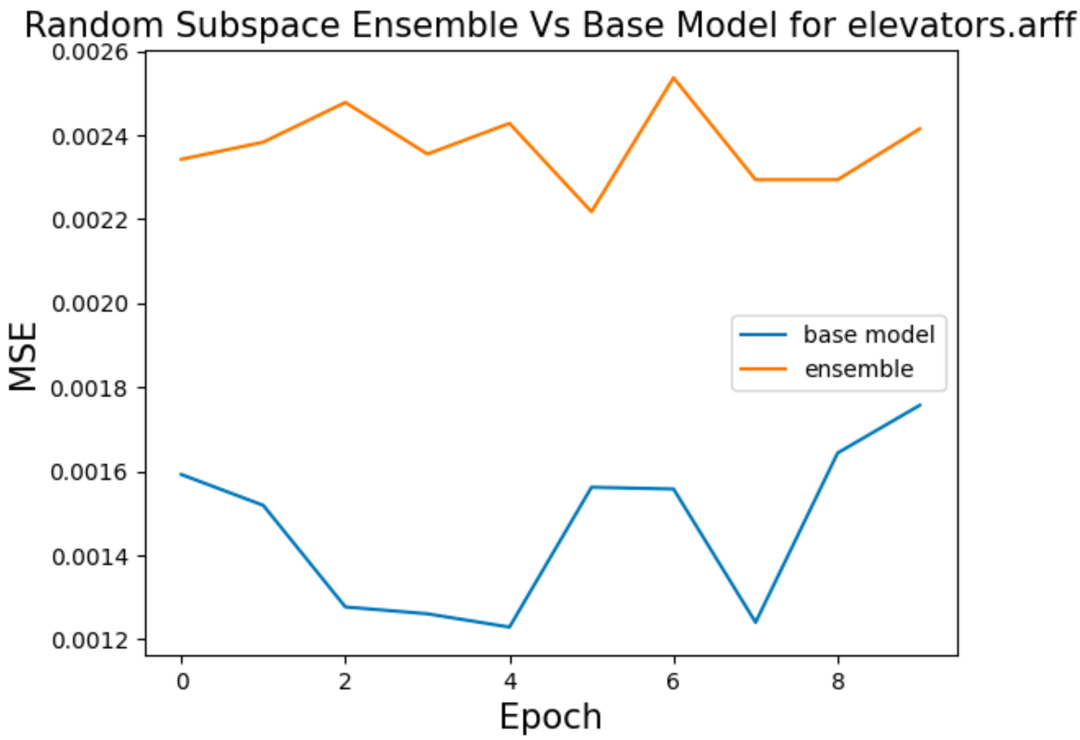



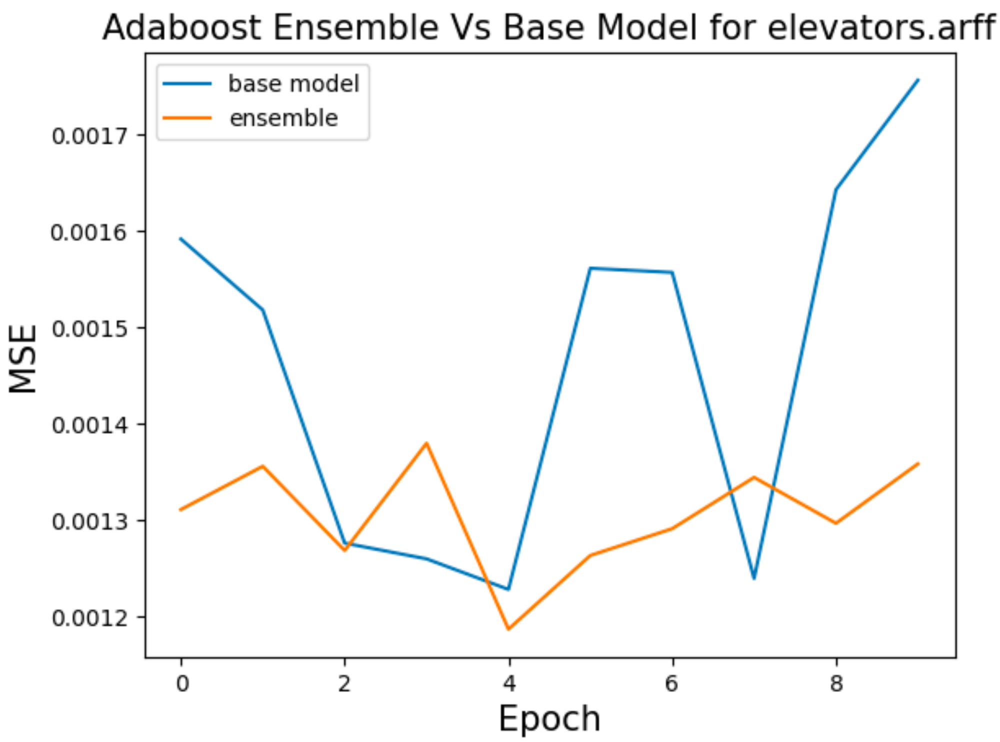

Adaboost Ensemble Vs Base Model for 2dplanes.arff
base model
ensemble
MSE
0.00325
0.00300
0.00275
0.00250
0.00225
0.00200
0.00175
0
2
4
6
8
Epoch

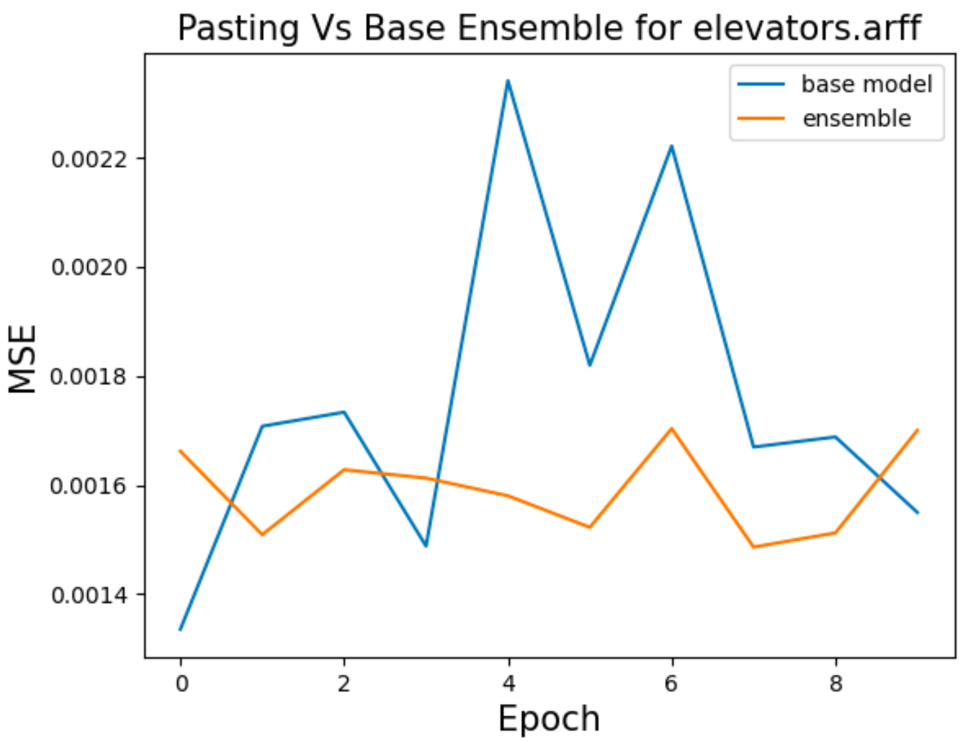

Pasting Vs Base Ensemble for 2dplanes.arff
base model
ensemble
MSE
0.00375
0.00350
0.00325
0.00300
0.00275
0.00250
0.00225
0.00200
0.00175
0
2
4
6
8
Epoch

**Dataset 02: elevators.arff**




Adaboost Ensemble Vs Base Model for elevators.arff
base model
ensemble
MSE
0.0017
0.0016
0.0015
0.0014
0.0013
0.0012
0
2
4
6
8
Epoch


Pasting Vs Base Ensemble for elevators.arff
base model
ensemble
MSE
0.0022
0.0020
0.0018
0.0016
0.0014
0
2
4
6
8
Epoch

# 7. Conclusion:

- Using Bagging and Boosting algorithms helped in enhancing the performance of the model.

- AdaBoost ensemble model converged after few epochs compared to the base model and to the bagging ensemble models. However, it takes more time to be trained since we cannot parallelize the training of the weak learners (because the data weights of each weak leaner are based on the errors of the previous weak learner).

- Pasting (without replacements) and Bagging (with replacement) had very close values for all datasets. Hence, the replacement method did not affect the performance of the Bagging model.

- Random Subspace models did not perform well for the used datasets. Not having a high number of features, reducing the number of features used for each weak learner hurt the performance and made it worse than all other ensemble models and even worse than the base model.

# 8. References


1. Raschka, S. (no date) *PAIRED_TTEST_5X2CV: 5x2cv paired t test for classifier comparisons*, *paired_ttest_5x2cv: 5x2cv paired *t* test for classifier comparisons - mlxtend*. Available at: http://rasbt.github.io/mlxtend/user_guide/evaluate/paired_ttest_5x2cv/ (Accessed: November 30, 2022).
2. Silvan (2019) *Ensemble methods: Bagging and pasting in Scikit-Learn*, *Medium*. Medium. Available at: https://medium.com/@silvaan/ensemble-methods-bagging-and-pasting-in-scikit-learn-723f4183cdf4 (Accessed: November 28, 2022).
3. *Sklearn.ensemble.baggingregressor* (no date) *scikit*. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingRegressor.html#sklearn.ensemble.BaggingRegressor (Accessed: November 25, 2022).
4. Politi, M. (2022) *Paired T-test to evaluate machine learning classifiers using Python*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/paired-t-test-to-evaluate-machine-learning-classifiers-1f395a6c93fa (Accessed: November 30, 2022).