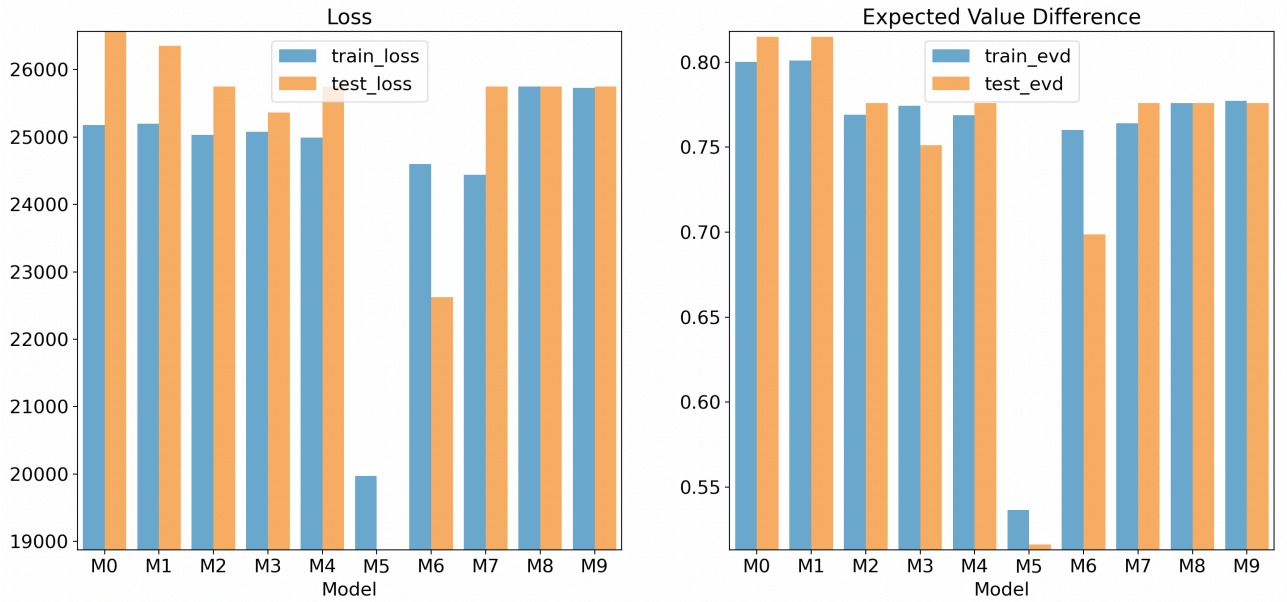
Ensemble Params & True Values

```
models_to_train = 10 # train this many models
max_epochs = 3       # for this many epochs
iters_per_epoch = 25
learning_rate = 0.1
```

True R:[3., 6., 5., 2.]
– negated likelihood(loss): 17049.900824954246
– optimal policy: [2 0 3 3]

Models Loss & EVD during training and testing

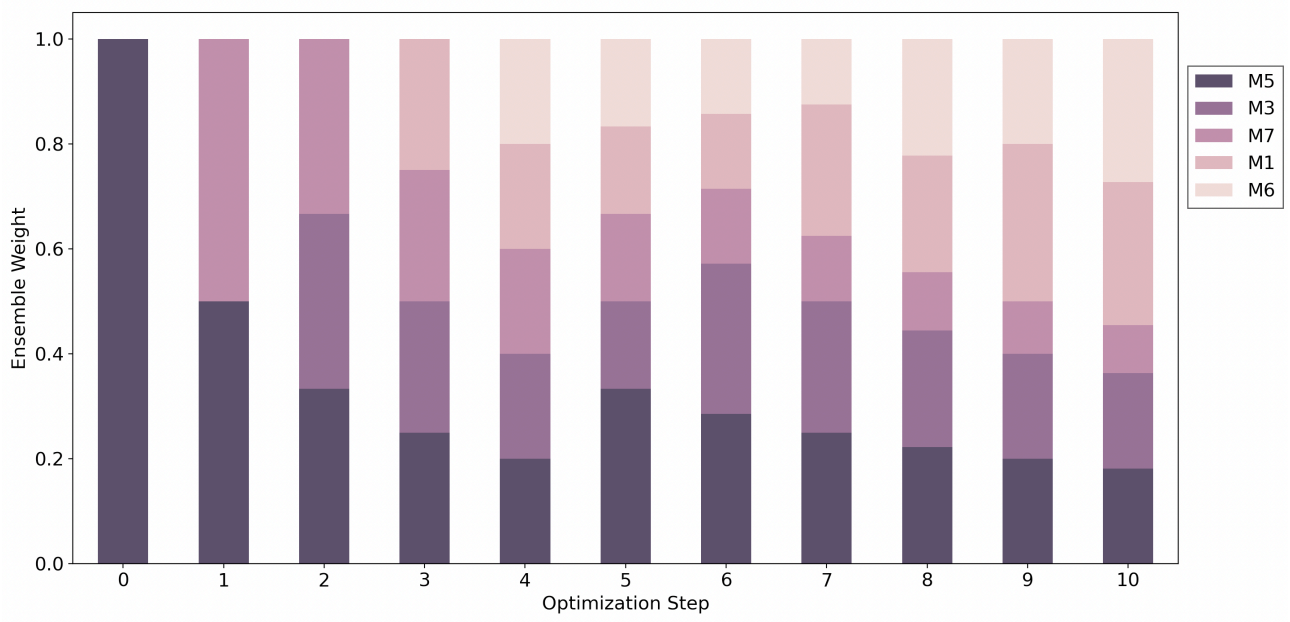


Models Weights

Ensemble Model Weight:

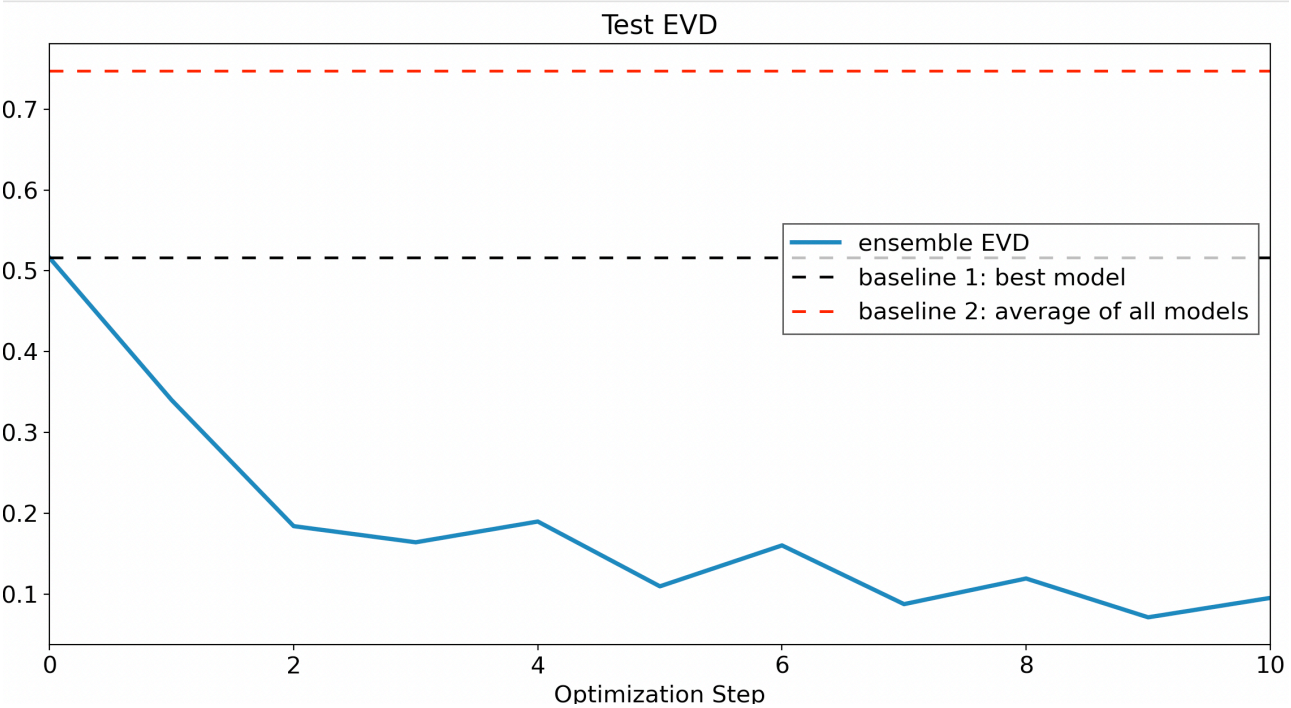
	M0	M1	M2	M3	M4	M5	M6	M7	M8	M9
0	0.0	0.000000	0.0	0.000000	0.0	1.000000	0.000000	0.000000	0.0	0.0
1	0.0	0.000000	0.0	0.000000	0.0	0.500000	0.000000	0.500000	0.0	0.0
2	0.0	0.000000	0.0	0.333333	0.0	0.333333	0.000000	0.333333	0.0	0.0
3	0.0	0.250000	0.0	0.250000	0.0	0.250000	0.000000	0.250000	0.0	0.0
4	0.0	0.200000	0.0	0.200000	0.0	0.200000	0.200000	0.200000	0.0	0.0
5	0.0	0.166667	0.0	0.166667	0.0	0.333333	0.166667	0.166667	0.0	0.0
6	0.0	0.142857	0.0	0.285714	0.0	0.285714	0.142857	0.142857	0.0	0.0
7	0.0	0.250000	0.0	0.250000	0.0	0.250000	0.125000	0.125000	0.0	0.0
8	0.0	0.222222	0.0	0.222222	0.0	0.222222	0.222222	0.111111	0.0	0.0
9	0.0	0.300000	0.0	0.200000	0.0	0.200000	0.200000	0.100000	0.0	0.0
10	0.0	0.272727	0.0	0.181818	0.0	0.181818	0.272727	0.090909	0.0	0.0

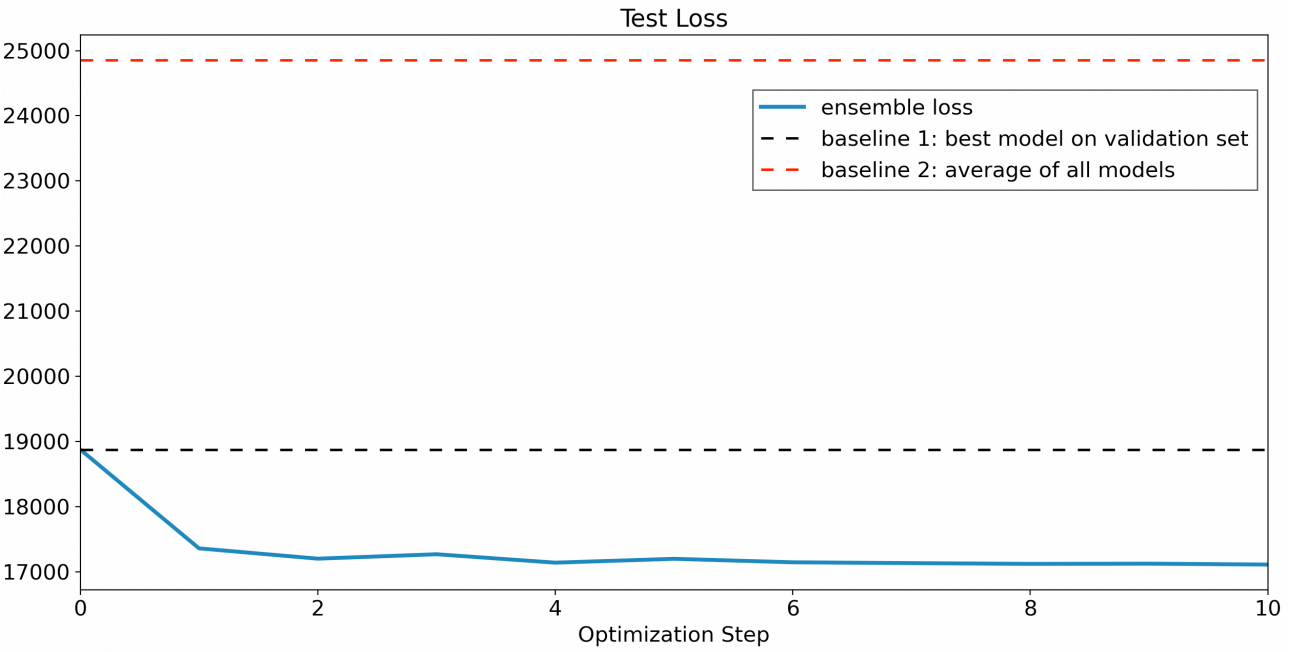
The figure below plots weights of the ensemble's constituents as a function of ensemble optimization steps with lighter hues corresponding to lower average weights over all iterations. The ensemble is initialized with the strongest single model **M5** and then sequentially adds models to minimize the loss on the validation set. Once no new model can improve the loss any further, the algorithm continues to add the models that are already in the ensemble increasing their weight.



Ensemble Loss & Expected Value Diff vs Individual Models

True R:[3., 6., 5., 2.]
– negated likelihood(loss): 17049.900824954246
– optimal policy: [2 0 3 3]





We can see the ensemble performing much better than the individual models themselves.