Question 1

What back-propagation is usually used for in neural networks?

Correct answer:

- To calculate gradient of the loss function with respect to the parameters of the network

Incorrect answers:

- To propagate signal through network from input to output only. This is called "forward pass"
- Make several random perturbations of parameters and go back to the best one. This one doesn't involve gradients and have nothing to do with back-propagation
- Select gradient update direction by flipping a coin. In back-propagation gradients are calculated exactly, not random

Question 2

Suppose we've trained a RandomForest model with 100 trees. Consider two cases:

1. We drop the first tree in the model

2. We drop the last tree in the model

We then compare models performance *on the train set*. Select the right answer.

Correct answers:

- In the case1 performance **will be roughly the same** as in the case2. In RandomForest model we average 100 similar performing trees, trained independently. So the order of trees does not matter in RandomForest and performance drop will be very similar on average.

Incorrect answers:

- In the case1 performance **will drop more** than in the case2. In RandomForest model we average 100 similar performing trees, trained independently. So the order of trees does not matter in RandomForest.
- In the case1 performance **will drop less** than in the case2. Similar to the previous one.

Question 3

Suppose we've trained a GBDT model with 100 trees with a fairly high learning rate. Consider two cases:

1. We drop the first tree in the model

2. We drop the last tree in the model

We then compare models performance *on the train set*. Select the right answer.

Correct answers:

- In the case1 performance **will drop more** than in the case2. In GBDT model we have sequence of trees, each improve predictions of all previous. So, if we drop first tree — sum of all the rest trees will be biased and overall performance should drop. If we drop the last tree -- sum of all previous tree won't be affected, so performance will change insignificantly (in case we have enough trees)

Incorrect answers:

- In the case1 performance **will drop less** than in the case2.
- In the case1 performance **will be roughly the same** as in the case2.

Question 4

Consider the two cases:

1. We fit two RandomForestClassifiers 500 trees each and average their predicted probabilities on the test set.

2. We fit a RandomForestClassifier with 1000 trees and use it to get test set probabilities.

All hyperparameters except number of trees are the same for all models.Select the right answer.

Correct answers:

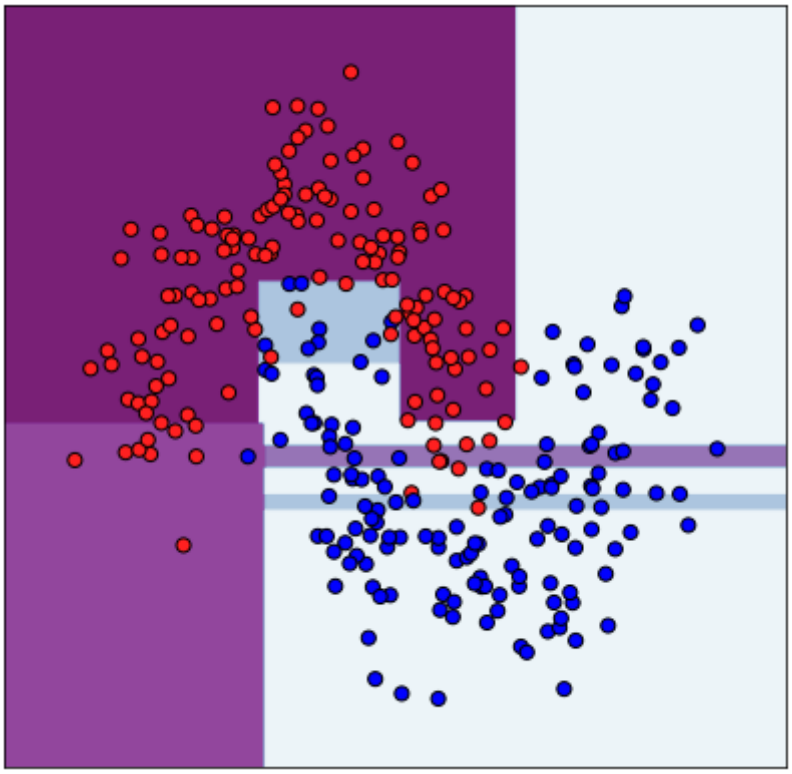
- The quality of predictions in the case1 **will be roughly the same** as the quality of the predictions in the case2. Each tree in forest is independent from the others, so two RF with 500 trees is essentially the same as single RF model with 1000 trees

Incorrect answers:

- The quality of predictions in the case1 **will be higher** than the quality of the predictions in the case2.
- The quality of predictions in the case1 **will be lower** than the quality of the predictions in the case2.

Question 5

What model was most probably used to produce such decision surface? Color (from white to purple) shows predicted probability for a point to be of class "red".



Correct answers:

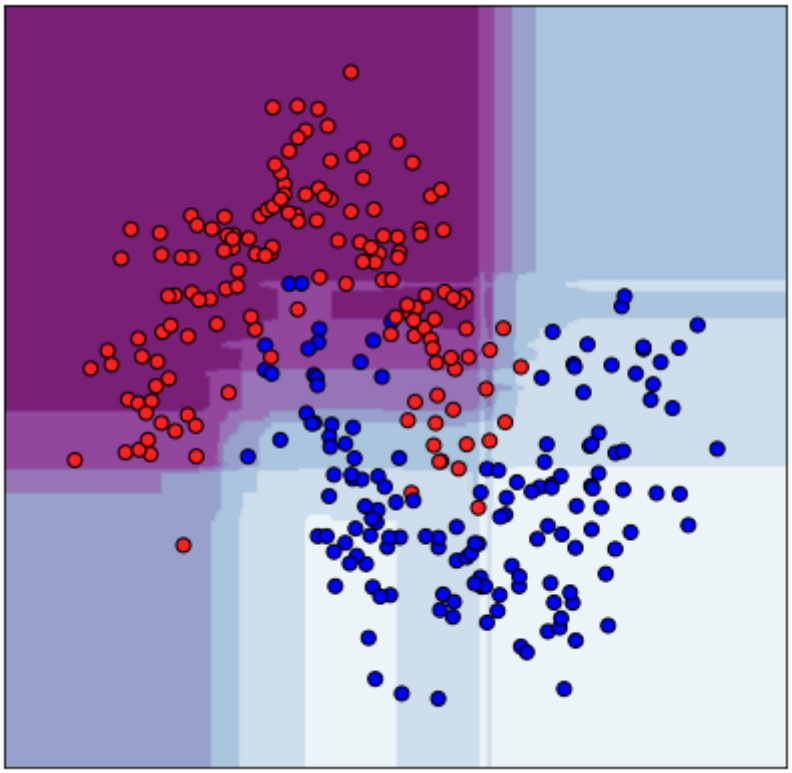
- Decision Tree. Decision surface consists of lines parallel to the axis and it is sharp.

Incorrect answers:

- Linear model. Decision surface is not linear.
- Random Forest. Decision surface consists of lines parallel to the axis and it is sharp -- in case of RF boundaries should be much more shooth.
- k-NN. Decision surface doesn't depend on distance from objects

Question 6

What model was most probably used to produce such decision surface?



Correct answers:

- Random Forest. Decision surface consists of lines parallel to the axis and its boundaries are smooth

Incorrect answers:

- Linear model. Decision surface is not linear
- Decision Tree. Decision surface consists of lines parallel to the axis and it is *not* sharp
- k-NN. Decision surface doesn't depend on distance from objects

Mark as completed