Instituto Superior de Agronomia, ULisboa

Green Data Science

Practical Machine Learning/Aprendizagem Automática Aplicada

Questionnaire #8, May 12, 2023

Name:_____

Topic: Training, validation and test sets

1. Complete the missing words in the text below with the following words: hyperparameters, recall, test ,difference, fine-tuning , validation, unbiased, loss, independent¸ predicted, over-fitting, biased.

<<When developing a machine learning algorithm, it is crucial to carefully divide your data into three distinct sets: train, validation, and test. These sets play a vital role in the model development process, allowing you to assess and fine-tune the performance of your algorithm.

The training data set forms the foundation of your machine learning algorithm. It comprises a large portion of your available data and is used to train the model. This data set contains input features (or independent variables) and corresponding target values (or dependent variables). The training process involves iteratively adjusting the model's parameters to minimize the loss (or difference) between its predicted output and the actual target values.

The validation data set serves as a checkpoint during the model development process. It helps you assess how well your algorithm generalizes to unseen data and provides a measure of its performance. Unlike the training data set, the validation set is not used for training the model but rather for evaluating its performance and fine-tuning hyperparameters. By measuring metrics such as accuracy, precision, recall, or mean squared error on the validation set, you can make informed decisions about adjusting your model to optimize its performance. This step is essential as it helps you avoid over-fitting, where the model becomes overly specialized to the training data and performs poorly on new data.

The test data set is crucial for obtaining an unbiased evaluation of your model's performance. It represents real-world scenarios where the model is deployed and needs to make predictions or classifications. The test set should be completely independent of the training and validation sets to ensure an objective assessment. By evaluating your model on this data set, you gain insights into its generalization ability and its performance on unseen examples.

It's important to note that the test data set should only be used sparingly and at the end of the development process. It is not meant for iterative fine-tuning, as this could lead to overfitting to the test set itself. If adjustments are made based on test set performance, the model's generalization ability may become compromised.>>

2. The following line of code creates 4 mutually exclusive and complementary subsets from a data set of examples x and respective labels y

   x_other, x_test, y_other, y_test = train_test_split(x, y, test_size=0.1)

   Suppose that you have 1000 examples in x and y, and want to create a training set with 600 examples (x_train and y_train), a validation (or "development") set with 200 examples (x_valid and y_valid) , and a test set with 200 examples (x_test and y _test).

   Write 2 lines of code that produce x_train,y_train,x_valid,y_valid,x_test,y_test from x and y.

Response:

```
x_other, x_test, y_other, y_test = train_test_split(x, y, test_size=0.2)
x_train, x_valid, y_train, y_valid = train_test_split(x_other, y_other, test_size=0.25)
```

3. Suppose that your data set has more than 10000 examples (but you don't know in advance its size n). Write a function in Python with two arguments (x,y as above) that returns x_train,y_train,x_valid,y_valid,x_test,y_test where the test set has approximately 1000 examples, and the training set and the validation set have respectively ~80% and ~20% of the remaining (n-1000) examples. Use the train_test_split function above to define your function. You can get n with property .shape[0] or with function len().

Response:

```
def myfunction(x,y):
    n=len(x)
    x_other, x_test, y_other, y_test = train_test_split(x, y, test_size=1000/n)
    x_train, x_valid, y_train, y_valid = train_test_split(x_other, y_other, test_size=0.2)
    return x_train,y_train,x_valid,y_valid,x_test,y_test
```