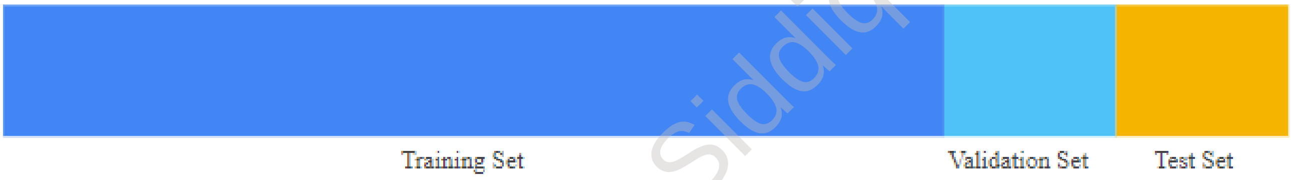
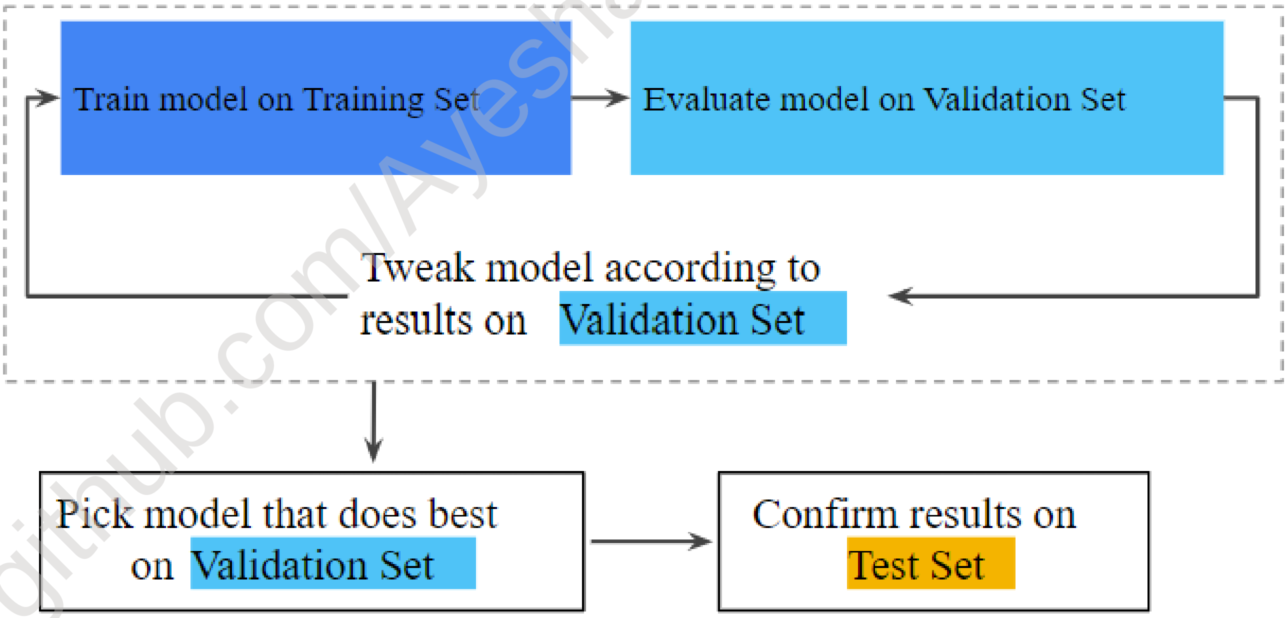
ML Concepts - Training , Test and Validation Sets

Dividing Data for Training and Testing:

- Data is split into two subsets: the training set and the test set.
- Training set is used to train the model, while the test set is used to evaluate its performance.
- The recommended split is 80% for training and 20% for testing.
- The test set should be:
  - Large enough to provide statistically meaningful results.
  - Representative of the entire dataset; it shouldn't have different characteristics than the training set.
- The goal is to create a model that generalizes well to this new data.
- Never use test data for training, as this can lead to misleadingly positive evaluation metrics.
- Accidentally training on test data can result in overestimating the model's generalization performance.




Train model on Training Set
Evaluate model on Validation Set
Tweak model according to results on Validation Set
Pick model that does best on Validation Set
Confirm results on Test Set

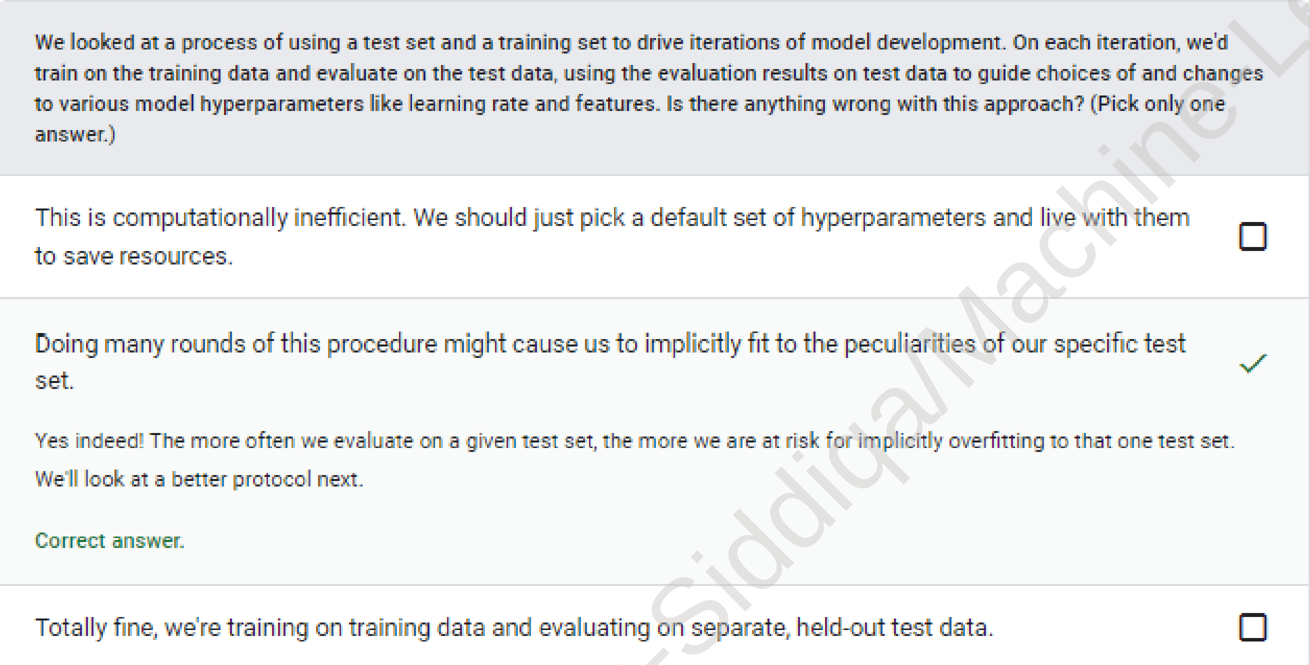
https://github.com/Ayesha-Siddiqa/Machine-Learni

Addressing Overfitting:

- While dividing data into two sets is beneficial, it may not be sufficient for preventing overfitting.
- Data set is divided into three subsets: 70% training, 15% validation, and 15% test.
- The validation set is employed to assess results obtained from the training set.

- Once the training and validation sets yield similar assessments, the model is then validated against the test set.
- The model that performs optimally on the validation set is chosen.
- The selected model is further confirmed against the independent test set.
- This approach minimizes the number of exposures to the test set, enhancing the reliability of evaluations.

Quiz:



We looked at a process of using a test set and a training set to drive iterations of model development. On each iteration, we'd train on the training data and evaluate on the test data, using the evaluation results on test data to guide choices of and changes to various model hyperparameters like learning rate and features. Is there anything wrong with this approach? (Pick only one answer.)

This is computationally inefficient. We should just pick a default set of hyperparameters and live with them to save resources. □

Doing many rounds of this procedure might cause us to implicitly fit to the peculiarities of our specific test set. ✓

Yes indeed! The more often we evaluate on a given test set, the more we are at risk for implicitly overfitting to that one test set. We'll look at a better protocol next.

Correct answer.

Totally fine, we're training on training data and evaluating on separate, held-out test data. □

ps://github.com/Ayesha-Siddiqa/Machine-Learni