Hackathon Best Practices

1. **Load the training and test data separately**
2. **Understand the data** (check for each of the following in both the train & test dataset)
    a. Check the head and tail of the data
    b. Use the info() and describe() functions for more information
    c. Look for the presence of null values in the dataset
    d. Look for the presence of bad data or unwanted characters like '$' or '#' in the numerical columns
3. **Clean the data**
    a. Treat for missing values in both the train & test set
    b. Remove bad data values in both the train & test set
    c. Encode the categorical object variables in both the train & test set
    d. Perform Feature Engineering if necessary
    e. Scale/Normalize the dataset if necessary
4. **Perform model building** using any algorithm you feel will suit this problem:
    a. Fit the model on the training data
    b. Make predictions on the testing data
    c. Store the predicted values in an array
5. **Submission (Approach 1)**
    a. Import the sample submission file
    b. Check that your test data and the sample submission file have the same ID column sequence (if not, then sort them such that each ID's individual predicted value is placed on the corresponding ID in submission file)
    c. Replace the "target" column in the submission file with your array of predictions
    d. Export this file to a CSV - this is your personal submission file
    e. Make sure it has the same headers as the sample submission file
    f. Upload to the platform and check your score to make sure
6. **Submission (Approach 2)**
    a. From your original test dataset, take the ID
    b. Create a new dataframe with the ID and the corresponding predicted values
    c. Export that dataframe to a CSV - this is your personal submission file
    d. The number of rows (including headers) & columns should match with that in the sample submission file, otherwise the platform will not accept the submission
    e. Upload to the platform and check your score to make sure
7. **Now go back to Step 3** (this is an iterative process)
    a. Check to see if scaling the data helps with performance
    b. Check to see if additional feature engineering helps with performance
    c. Try removing unnecessary variables (use feature importance) & check to see if that helps with performance

d. Play around with different algorithm choices & check to see if that helps improve final model performance
e. Try GridSearch, RandomSearch or other model tuning techniques to see if that improves the model's final performance

A few pointers to keep in mind:

1. **Do not drop null values from the test dataset.**
2. Make sure that if you perform any preprocessing step on the training dataset, it is also performed on the test dataset.
3. Make sure that the hyperparameter "**n_jobs = -1**" is set to that value while fitting the model, to ensure that parallel processing takes place and decreases the time taken to fit the model. However, this may take up all your local computational resources, and your PC may start slowing down for other running tasks.
4. It is recommended to make copies of the datasets at every checkpoint, so you don't have to restart from scratch. In case of any issues, this will allow you to directly access the dataset from the latest checkpoint and start from there instead.