Proposed PBD

Team CDEF

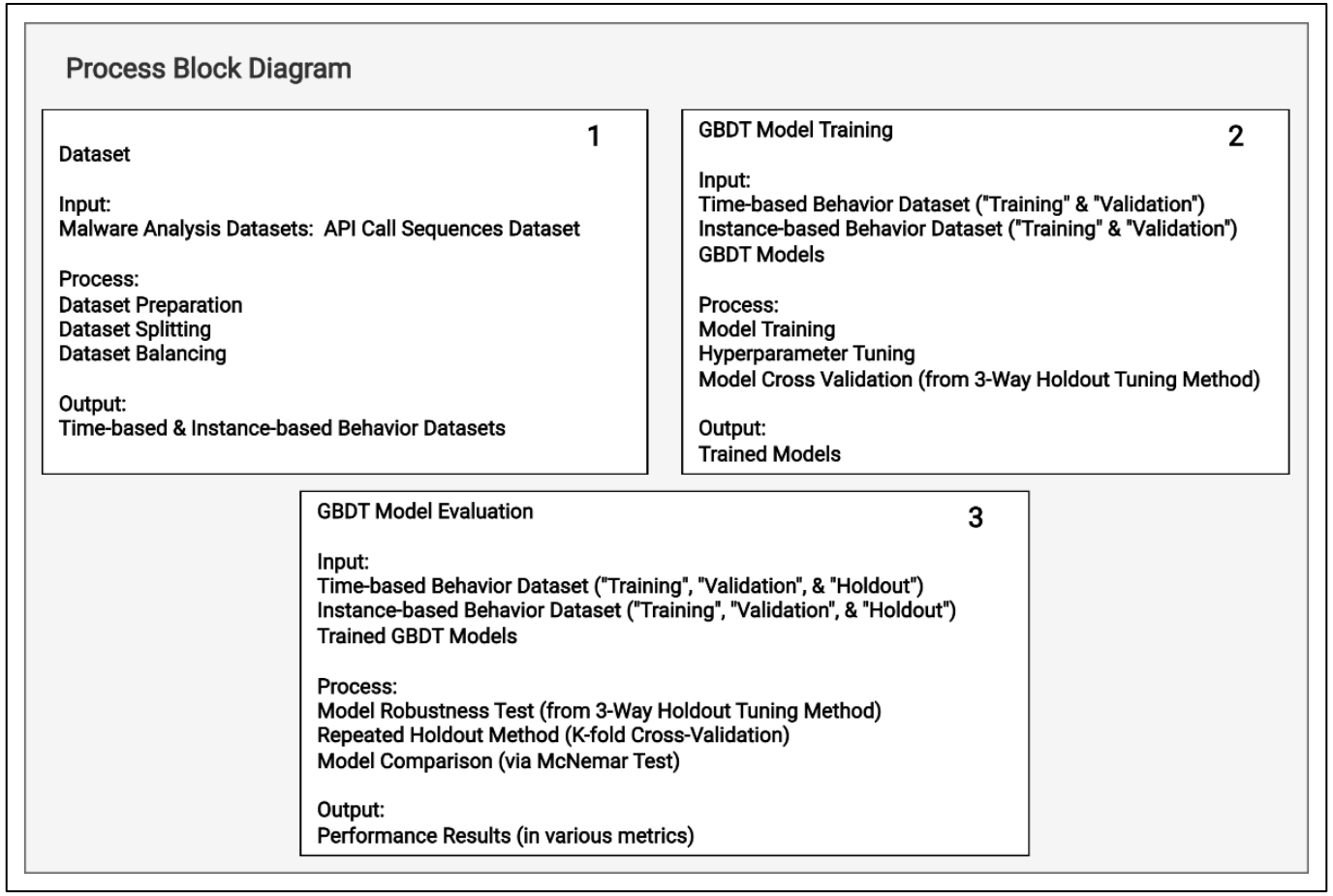
Process Block Diagram (PBD)

- The PBD will be divided into four segments.
  - **Overview**
  - **Dataset**
  - **GBDT Training**
  - **GBDT Evaluation**

- The three latter PBDs is based on the **three major aspects of building a ML model** and **the Gantt chart in CH2**.
- Do note not all concepts discussed in CH3 will be included in CH4 but the majority will be there.

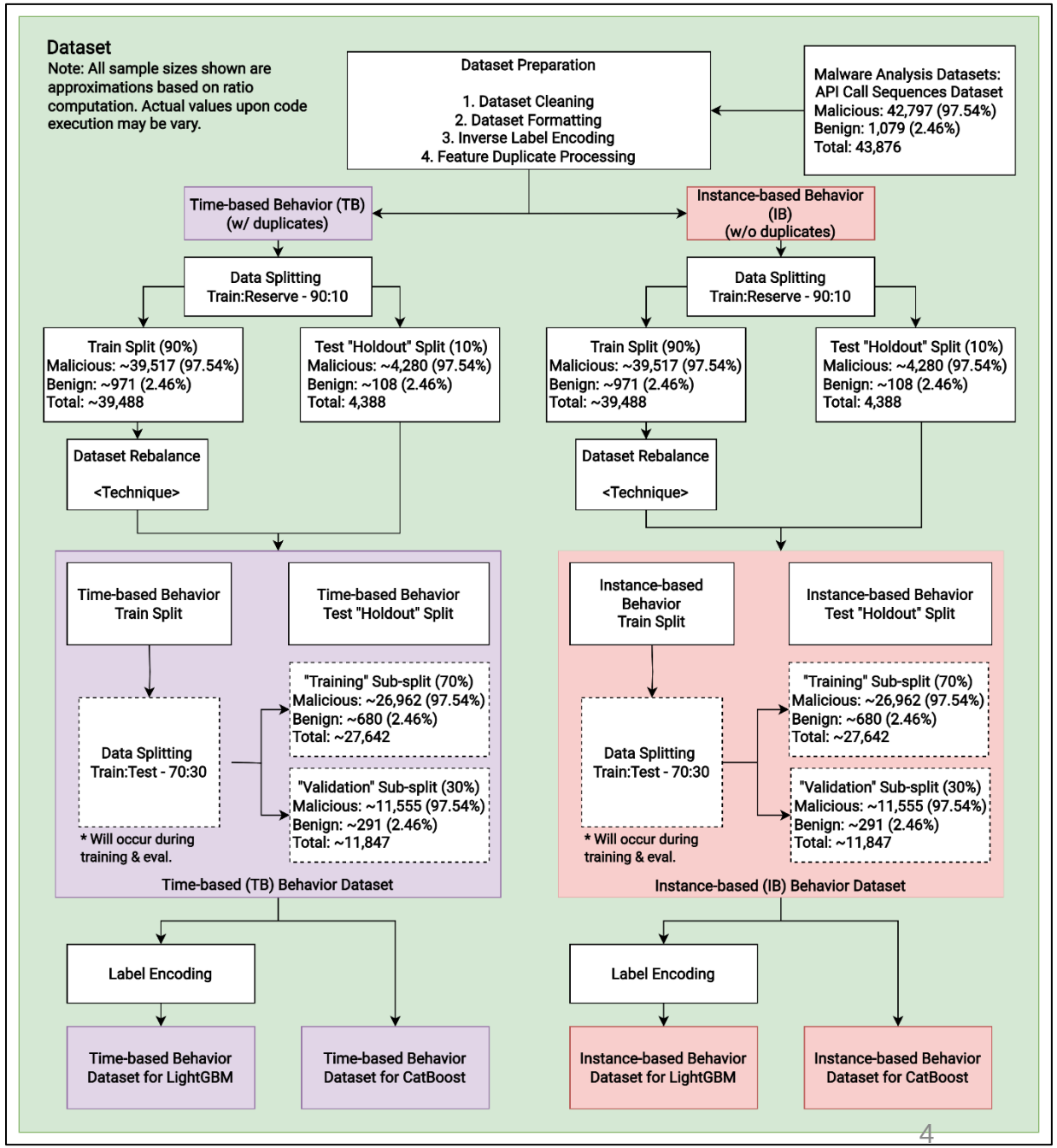
Overview PBD

- The overview PBD shows the overview of the major sections or aspects in conducting the thesis proper.
- It also represents the three major segments which will be completed in each of the three stages in thesis making (i.e., THES1-3).
- Each of the three blocks contains its own input(s), process(es), and output(s) to indicate the outcomes of each.
- The concept of **3-Way Holdout Method** will be utilized throughout the entire study.



Dataset PBD

- The Dataset PBD shows the specific dataset related process which in turn will result to the dataset being suitable for model development and evaluation on the context of the study.
- The Dataset Rebalancing Technique was left out as it still to be decided between **SMOTE or Oversampling**.
- The library Imbalanced Learn has functions for Oversampling and SMOTE. The selected ones are RandomOversampler and SMOTEN respectively.



Oversampling vs SMOTEN

Oversampling

Sample	Quantity in Resampled	% in Resampled
03384ab6368b68ed16ecb9e6352539af	90	0.22%
0822ec2ba98d291e5bfc836bc3686096	90	0.22%
f78ea80cec007b2c32fb10f9c6c82f39	88	0.21%
075323e77815ee8bcc7854ce23955a15	79	0.19%
79b78bb3d583748040c41ded09555fd3	72	0.17%

	precision	recall	f1-score	support
0	0.9110	0.8571	0.8832	12827
1	0.8653	0.9164	0.8901	12852
accuracy			0.8868	25679
macro avg	0.8882	0.8868	0.8867	25679
weighted avg	0.8881	0.8868	0.8867	25679

SMOTEN

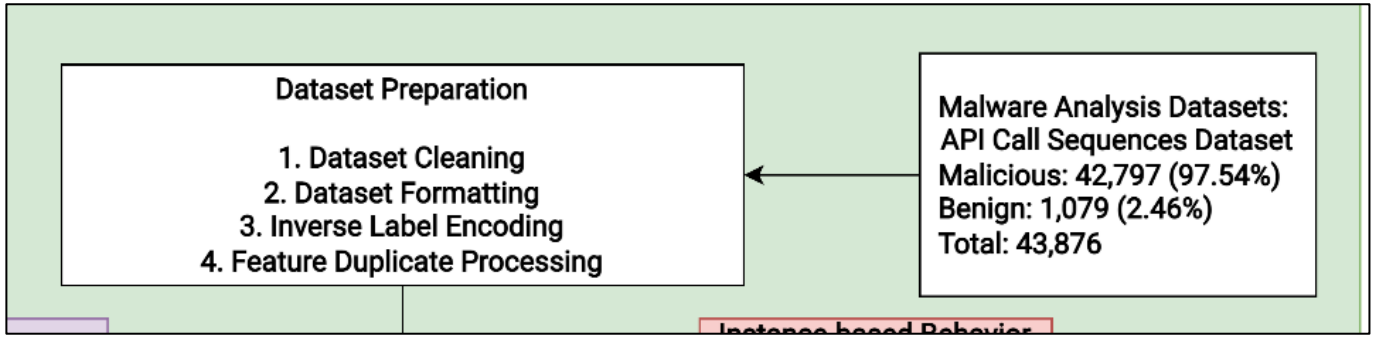
Sample	Quantity in Resampled	% in Resampled
0da6a786018d3e267de65f2532f7a1e0	5965	14.30%
302566218f78bb35439df31b54685ad0	3728	8.94%
0327301655f2e1c6bdbd4536a3349216	1895	4.54%
1707b149c4d1000242321167eec80eed	554	1.33%
3506356c329758e4f703cd2103d7daab	543	1.30%

	precision	recall	f1-score	support
0	0.9376	0.9411	0.9393	12827
1	0.9410	0.9374	0.9392	12852
accuracy			0.9393	25679
macro avg	0.9393	0.9393	0.9393	25679
weighted avg	0.9393	0.9393	0.9393	25679

(k_neighbors=2)

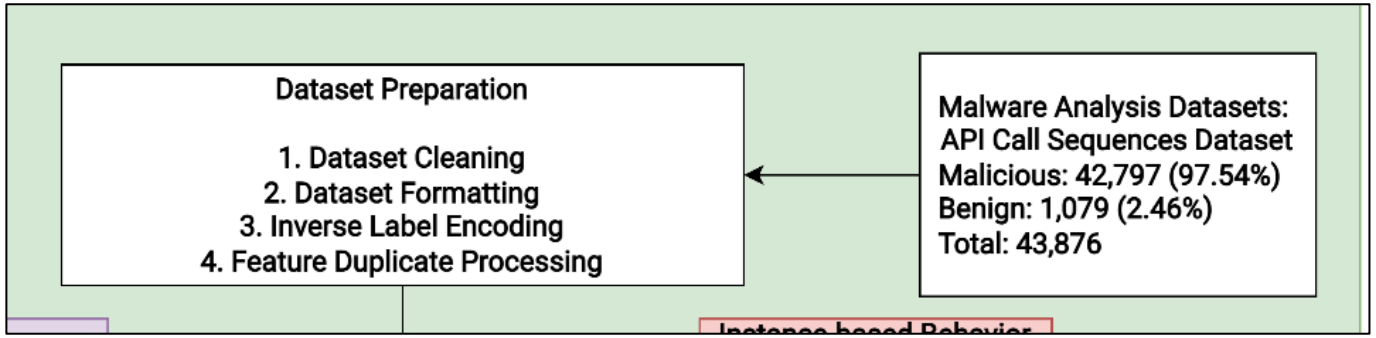
Dataset PBD

- The origin dataset (Oliveira) will undergo a set of procedures which will be summarized as “Dataset Preparation”.
- **Dataset Cleaning** – Nothing much here as the dataset is already clean enough. Sample hashes won’t be removed at the moment.
- **Dataset Formatting** – Rearranging column positions where the malware label column will be placed on the first column.



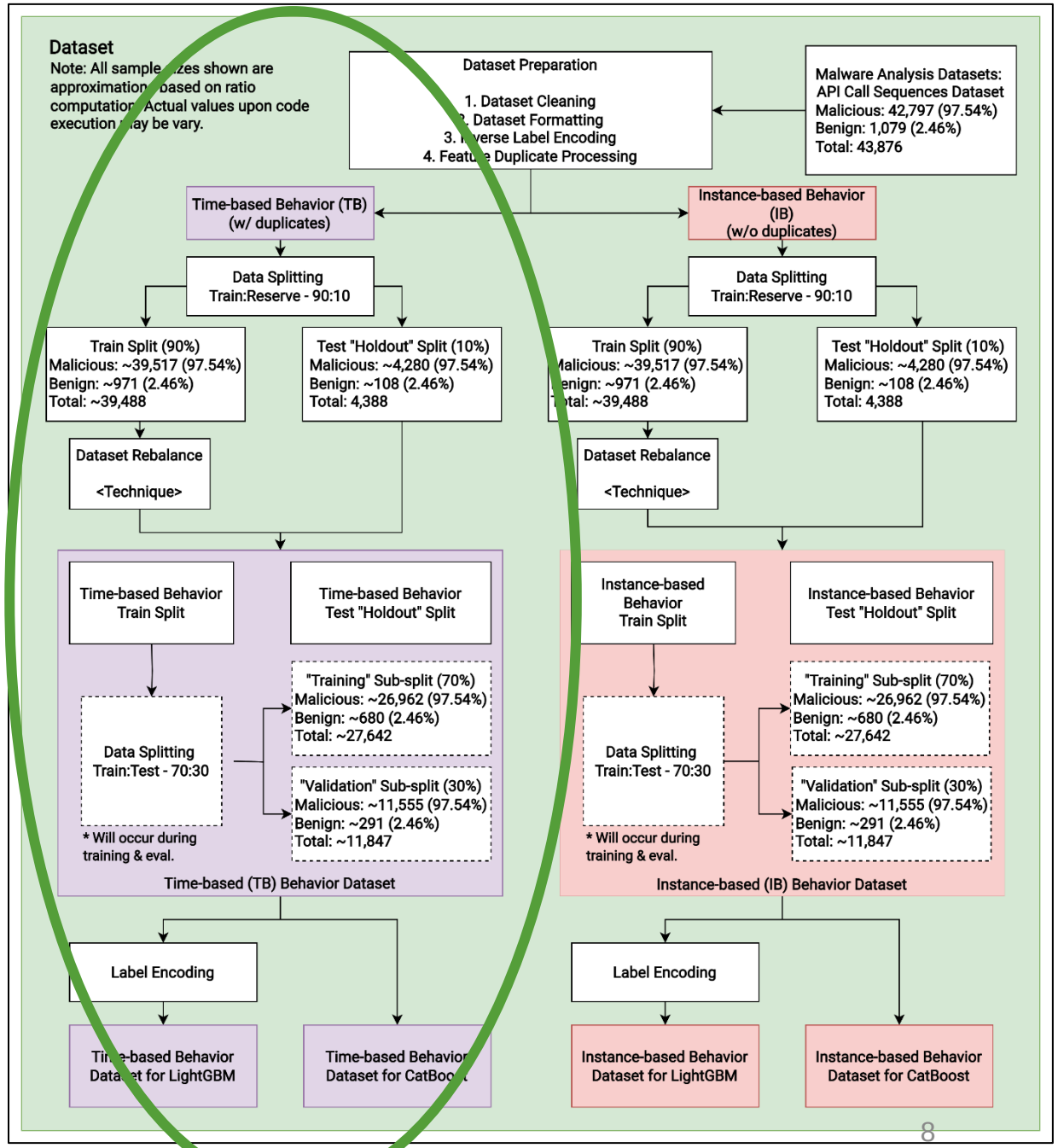
Dataset PBD

- **Inverse Label Encoding** – Converting the int categories to its equivalent string APIs using the list provided in IEEE Dataport.
- **Feature Duplicate Processing** – Creating two datasets for Time-based (w/ duplicates) and Instance-based (w/o duplicates) datasets.



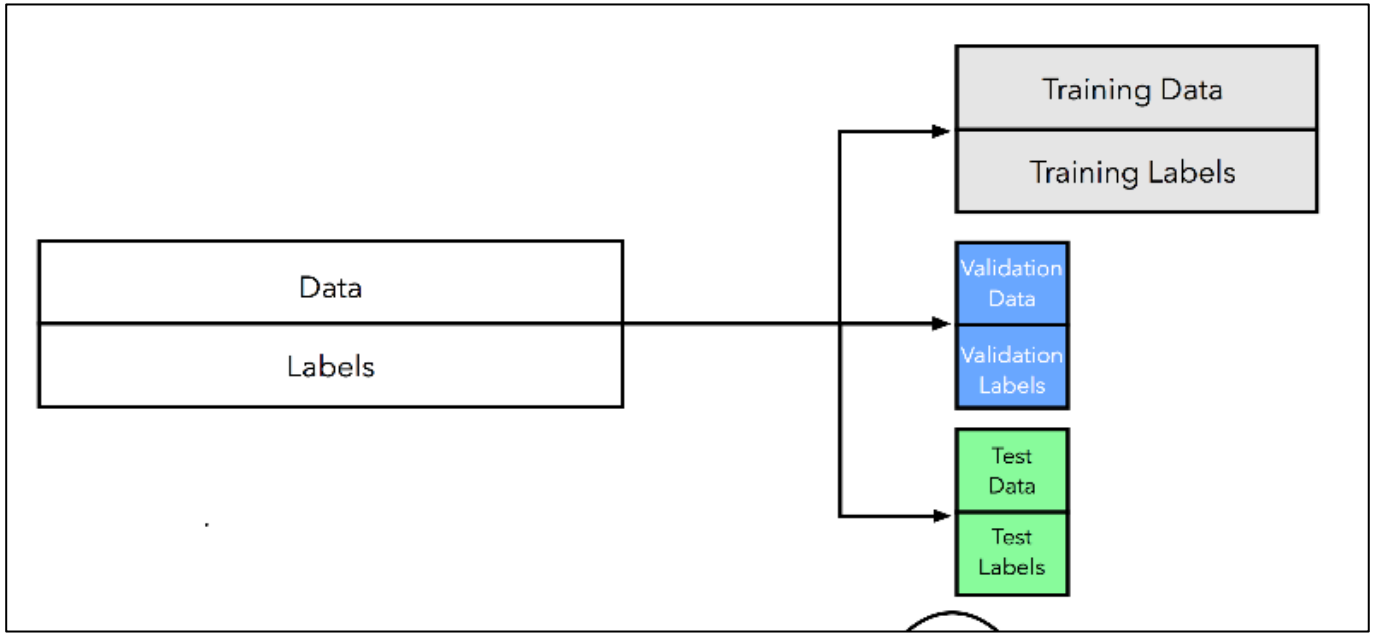
Dataset PBD

- Each of the steps will be conducted on the two datasets of different “behavior types”.
- For this presentation, we’ll be focusing on the dataset for Time-based Behavior.



Dataset PBD

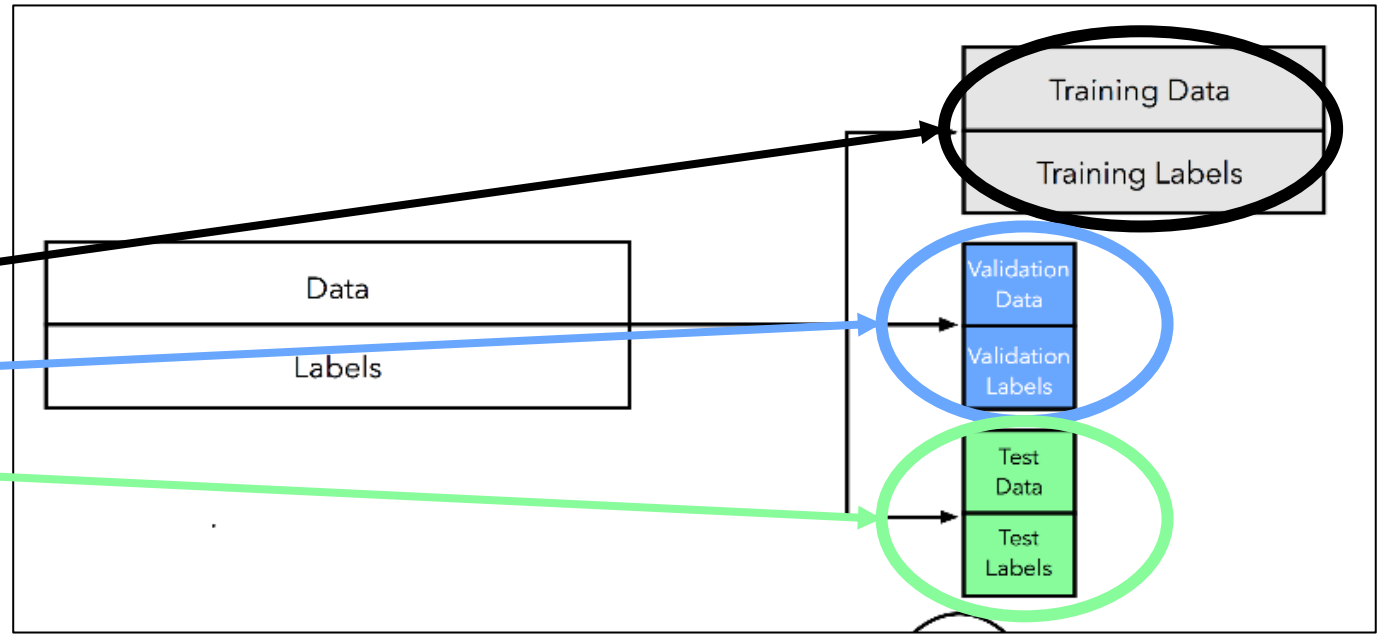
- There will be three segments of the dataset that will be made which will be divided as:
  - **“Training” aka Training**
  - **“Validation” aka Validation**
  - **“Holdout” aka Test (‘External Dataset’)**
- The reason behind the segmentation of the data is found in the **Three-Way Holdout Hypertuning Method**.



Source: (Raschka, 2018)

Dataset PBD

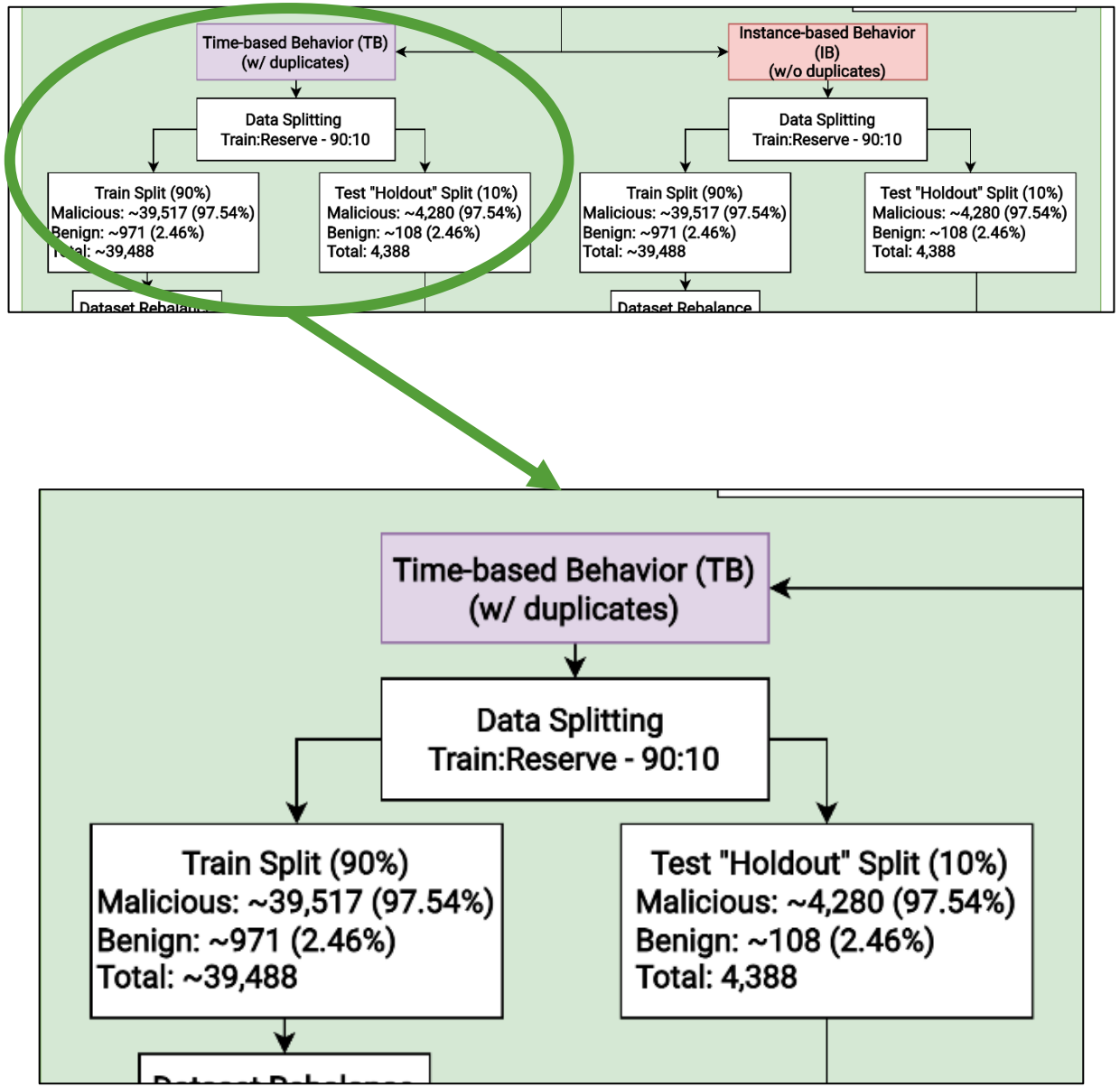
- There will be three segments of the dataset that will be made which will be divided as:
  - **“Training” aka Training**
  - **“Validation” aka Validation**
  - **“Holdout” aka Test (‘External Dataset’)**
- The reason behind the segmentation of the data is found in the **Three-Way Holdout Hypertuning Method**.



Source: (Raschka, 2018)

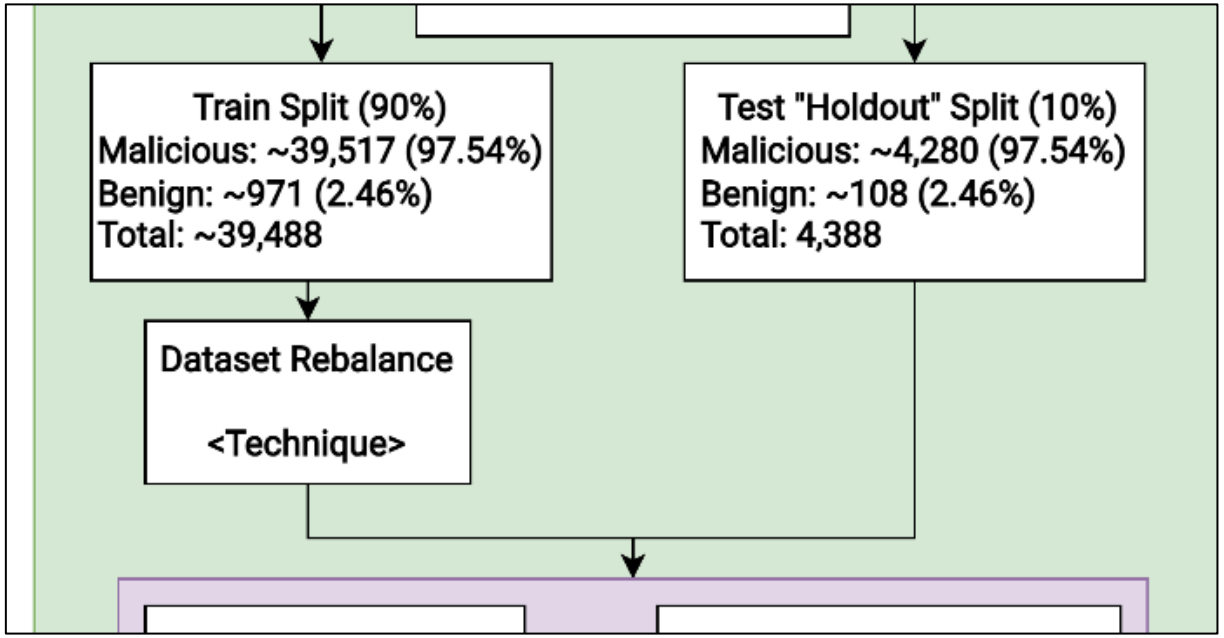
Dataset PBD

- The first data splitting will occur for splitting for the train split (i.e., “Training” & “Validation”) and the reserve (“Holdout”) split.
- The example uses a 90:10 ratio, for train:reserve, respectively.
- The ratio allows for samples for use in training and validation to be close to a total of 40K which is the nearly similar as the quantity of Oliveira itself.



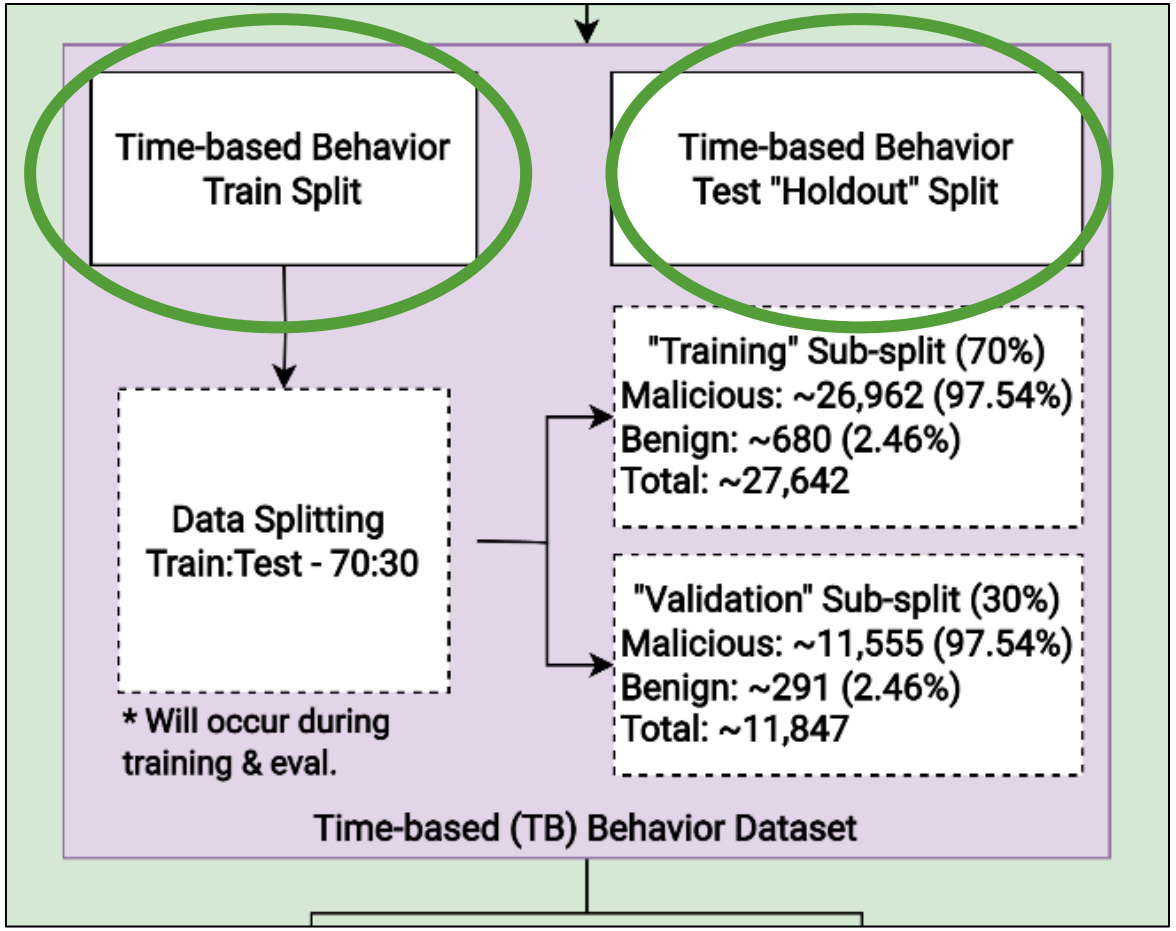
Dataset PBD

- Between the two splits, only the train split will undergo dataset rebalancing.
- The technique, as mentioned prior, is still undergoing selection process due to the pros and cons of each technique (i.e., SMOTE & Oversampling).
- Doing dataset rebalancing for “Holdout” will be pointless as doing so will only make duplicates which won’t result to improved scores.



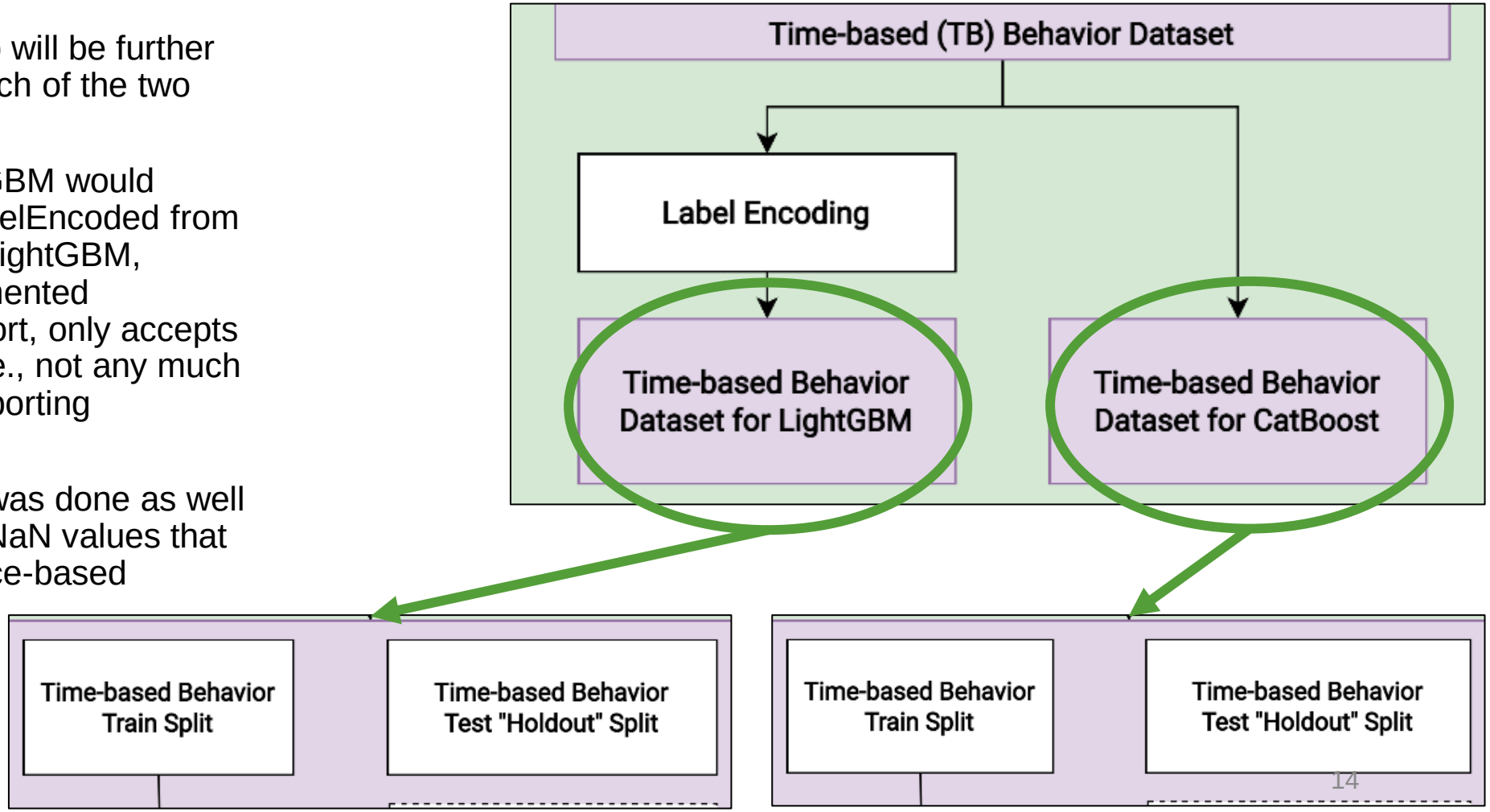
Dataset PBD

- In the end there will be two splits for each of the dataset of each behavior-type.
- It also mentions the sub-splits for “Training” and “Validation” from the original train split. This however will occur during training and evaluation as it tends to vary depending on method (e.g., K-Cross CV) that will be used.



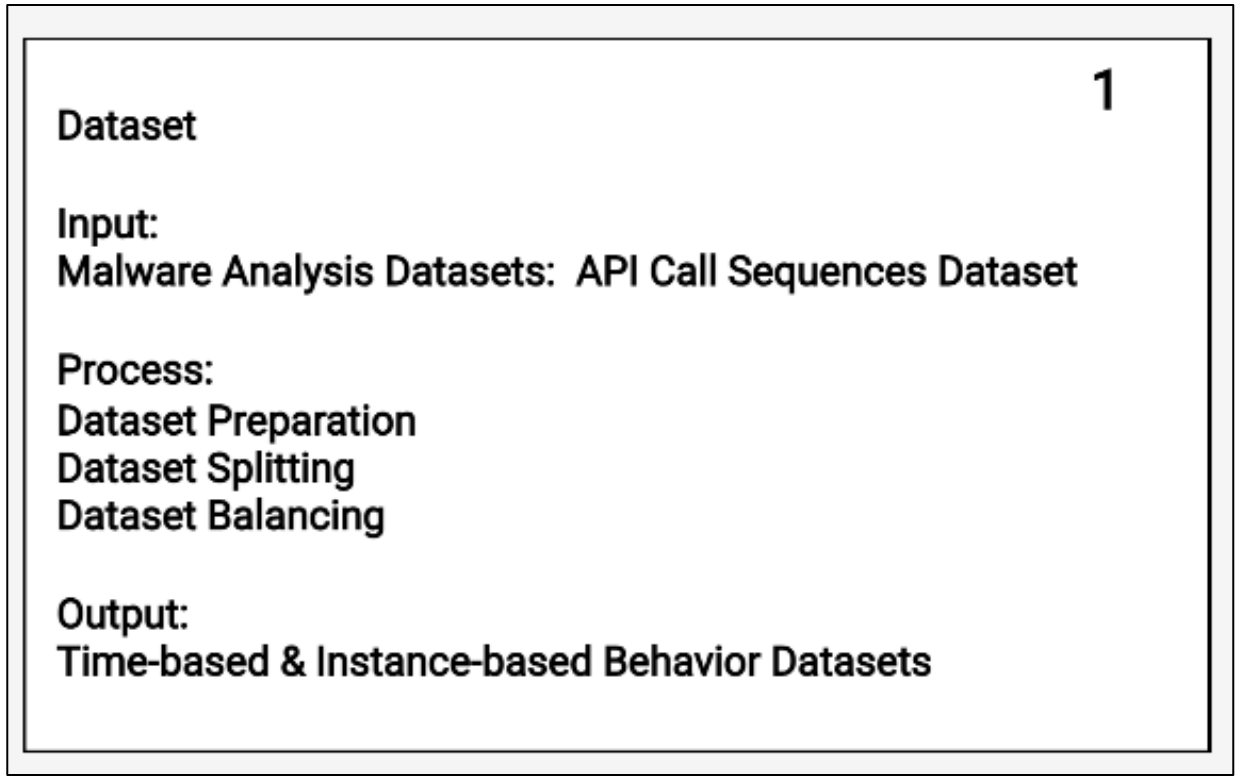
Dataset PBD

- Lastly, each of the two will be further divided into two for each of the two models.
- The dataset for LightGBM would require for it to be LabelEncoded from string to int again as LightGBM, despite having implemented categorical data support, only accepts integer-based data (i.e., not any much different from not supporting categorical data).
- Also Label Encoding was done as well to accommodate the NaN values that will exist in the Instance-based Dataset.



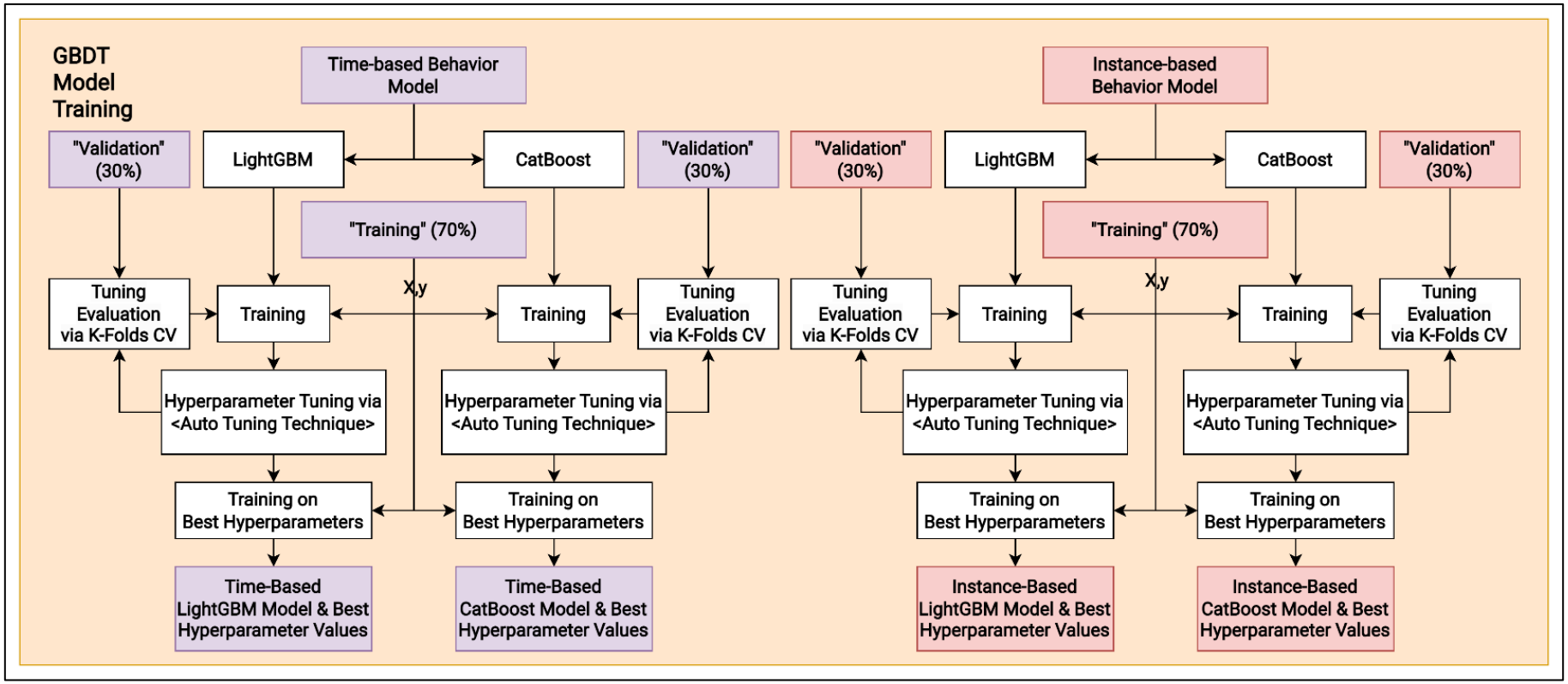
Dataset PBD

- The resulting output in this PBD will be the prepared datasets.
- Hence, there will be 8 datasets in total.
  - Time-based String API Train
  - Time-based String API “Holdout”
  - Time-based Encoded API Train
  - Time-based Encoded API “Holdout”
  - Instance-based String API Train
  - Instance-based String API “Holdout”
  - Instance-based Encoded API Train
  - Instance-based Encoded API “Holdout”



GBDT Training PBD

- The GBDT Training PBD shows the specific GBDT training-related process which in turn will result to the creation of the trained model suitable for model evaluation on the context of the study.
- The Hyperparameter Tuning Technique was left out as it still to be decided between **GridSearchCV & RandomizedSearchCV**.



GridSearchCV vs RandomizedSearchCV

GridSearchCV

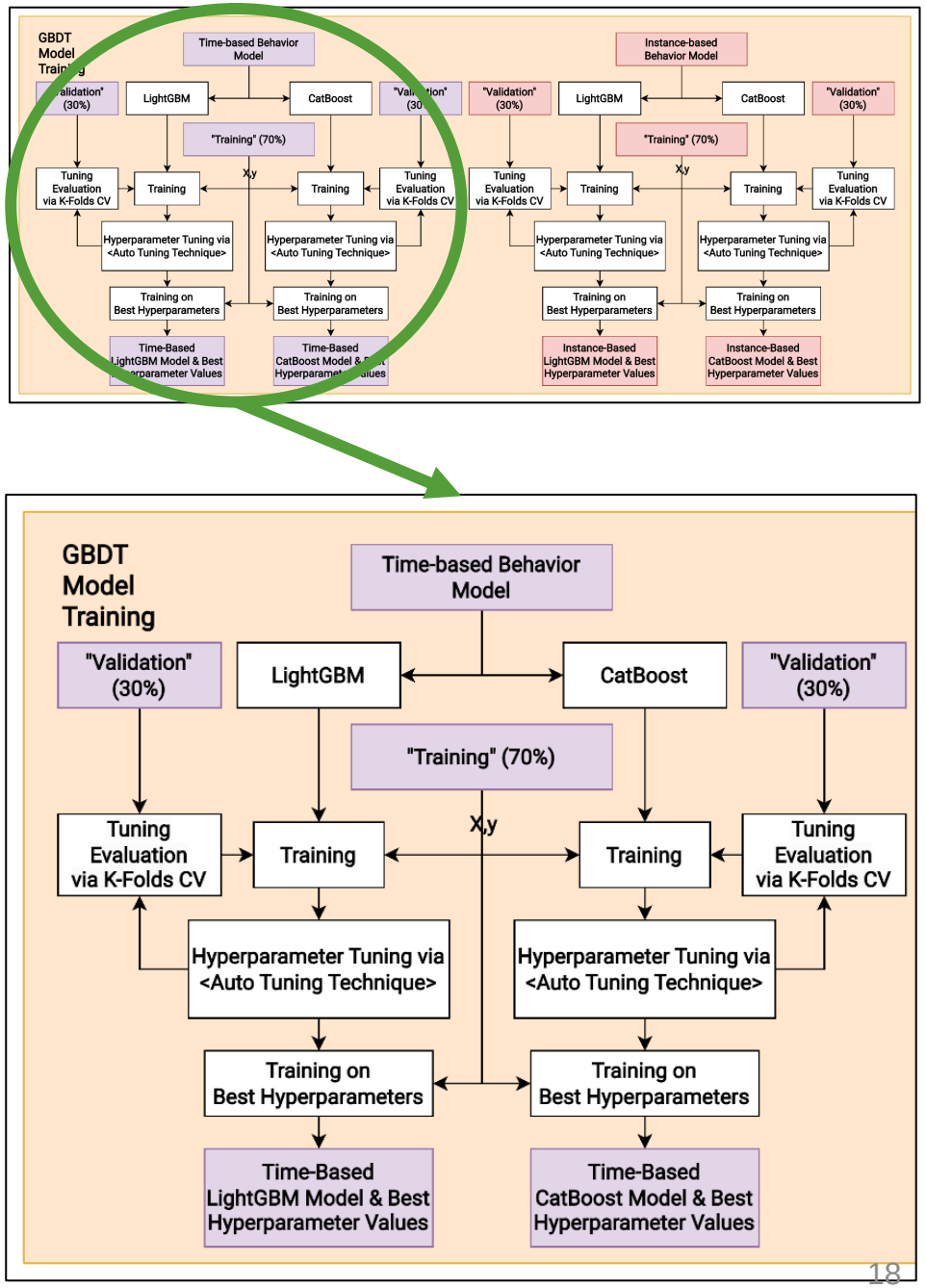
- Has built-in cross validation strategy, including Stratified K-Folds (set using *cv* parameter).
- Tries every combination of parameter set (i.e., more exhaustive).

RandomizedSearchCV

- Has built-in cross validation strategy, including Stratified K-Folds (set using *cv* parameter).
- Not all parameter values are tried out, but rather a fixed number of parameter settings is sampled from the specified distributions (set using *n_iter*).

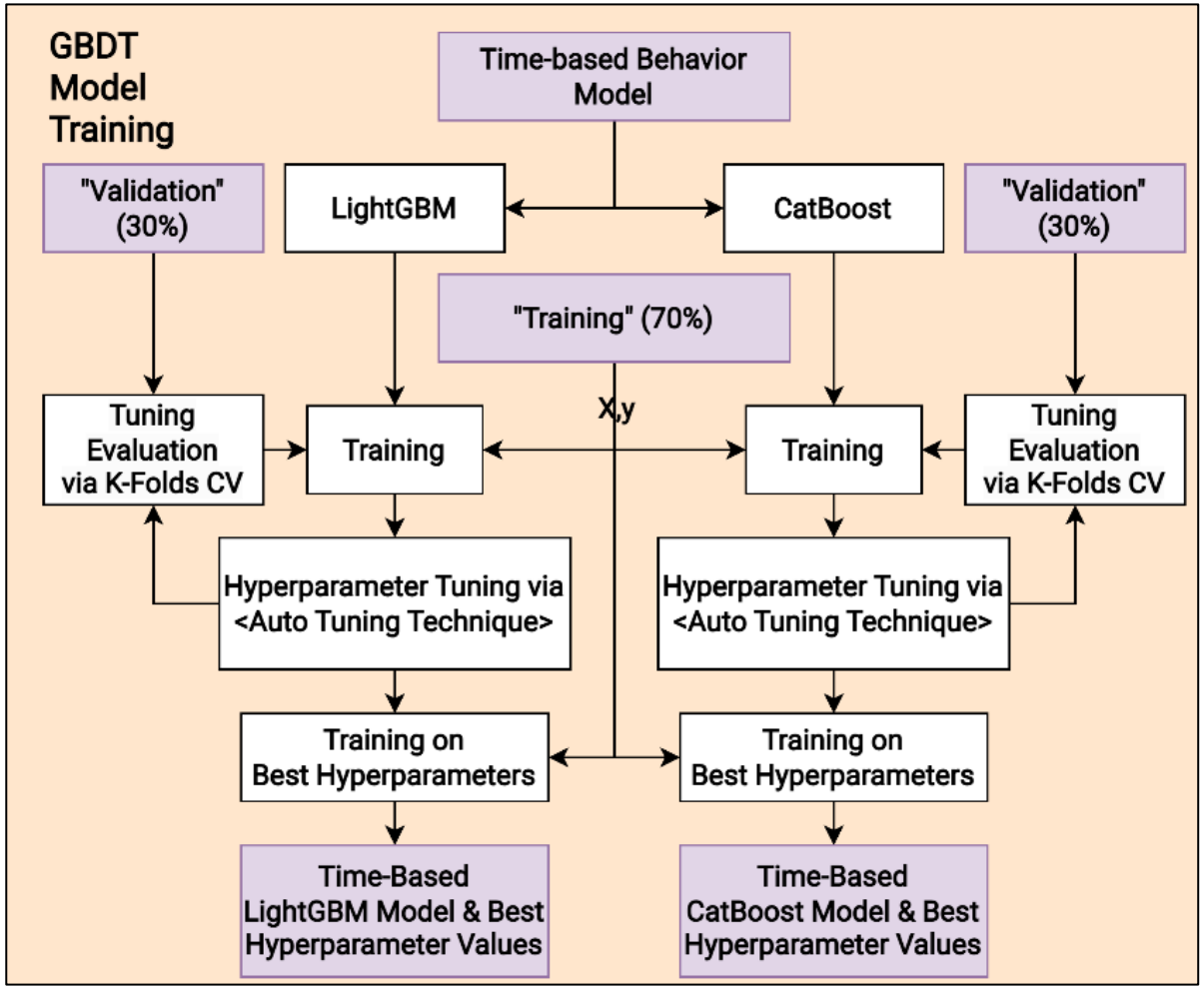
GBDT Training PBD

- Like the Dataset PBD, the steps will occur for both behavior-types.
- The focus for the presentation will be Time-based behavior Models.



GBDT Training PBD

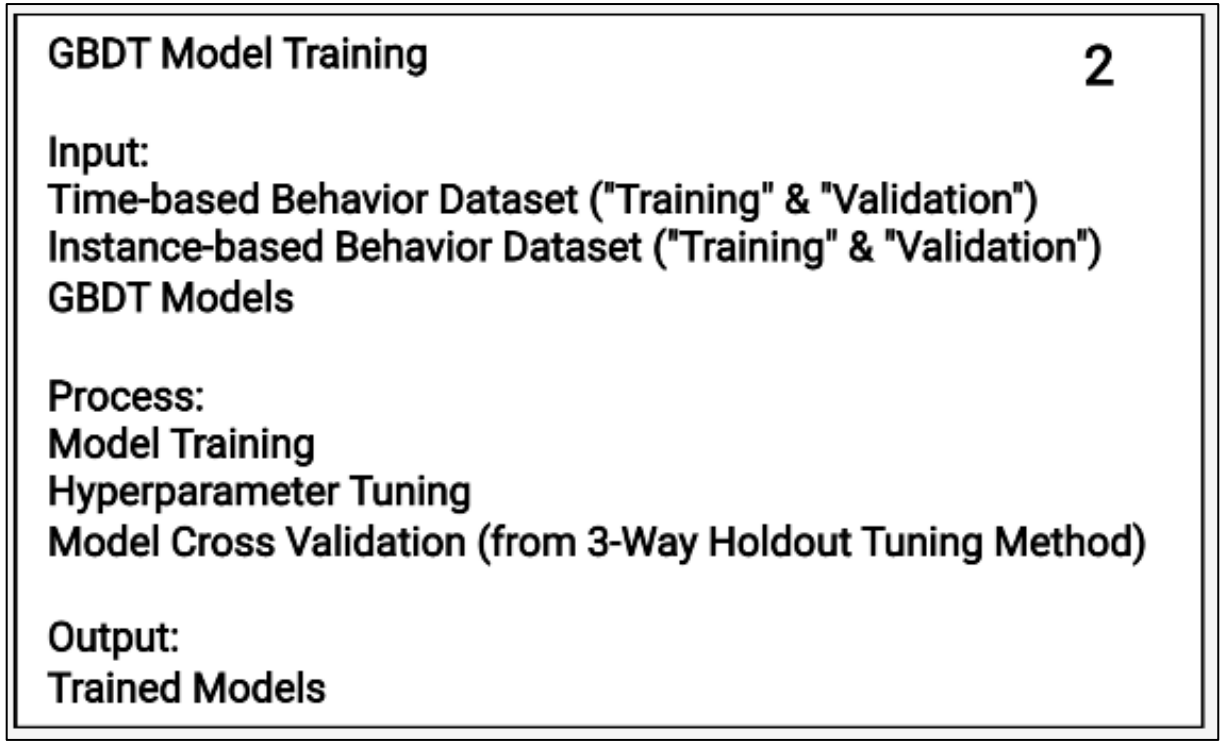
- For each of the model (LightGBM and CatBoost), the training will occur on the appropriate dataset (encoded and non-encoded respectively).
- The data splitting will occur at this stage for “Training” & “Validation” which will be done during Cross Validation.
- The splitting for “Training” and “Validation” is conducted in the hyperparameter tuning as it uses Stratified K-Folds CV as (indicated by cv parameter). Using K-Folds CV in general would ensure that the hyperparameters will be best possible without overfitting.




```
def auto_tune(setup, model, X, y, worker=-1):
    auto_tuner = RandomizedSearchCV(model, setup, refit=True, cv=5, verbose=1, n_jobs=worker, error_score=0, random_state=1)
    auto_tuner.fit(X, y)
    return auto_tuner.best_params_
```

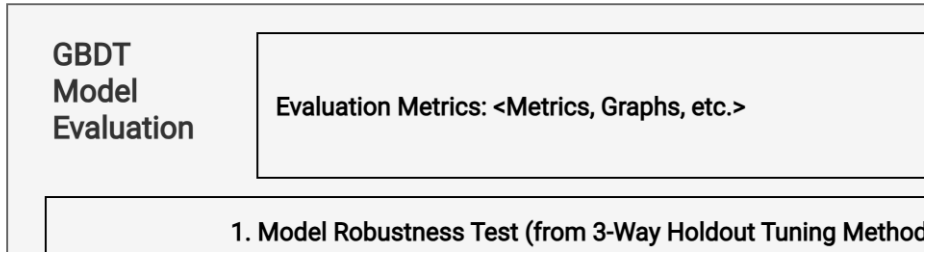
GBDT Training PBD

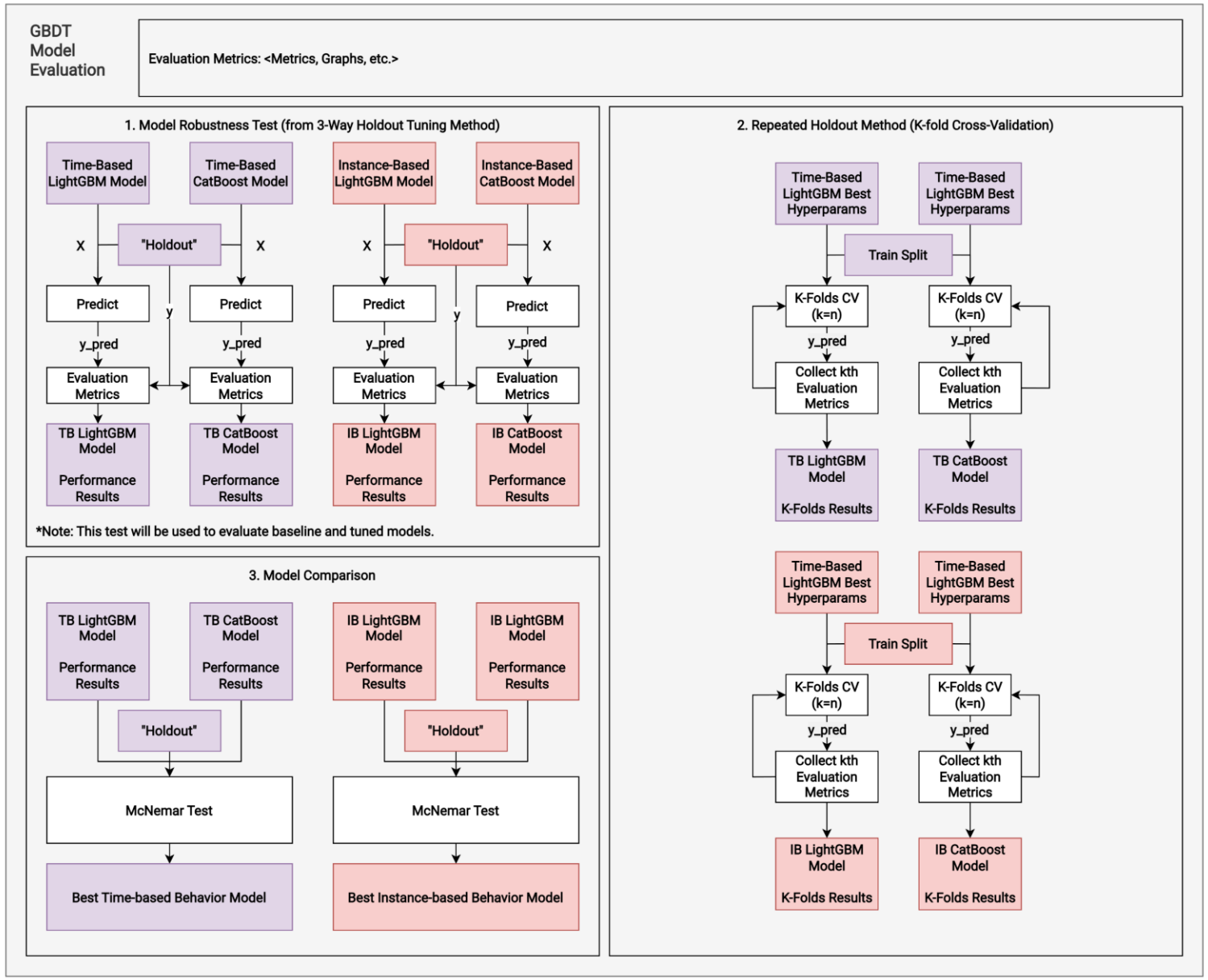
- The baseline results via K-Folds (and its averages) will be taken note for before conducting tuning.
- The resulting output in this PBD will be the trained models for both default and tuned models.
- Hence, there will be 8 trained HGBT models total.
  - Time-based LightGBM Default
  - Time-based LightGBM Tuned
  - Time-based CatBoost Default
  - Time-based CatBoost Tuned
  - Instance-based LightGBM Default
  - Instance-based LightGBM Tuned
  - Instance-based CatBoost Default
  - Instance-based CatBoost Tuned



GBDT Evaluation PBD

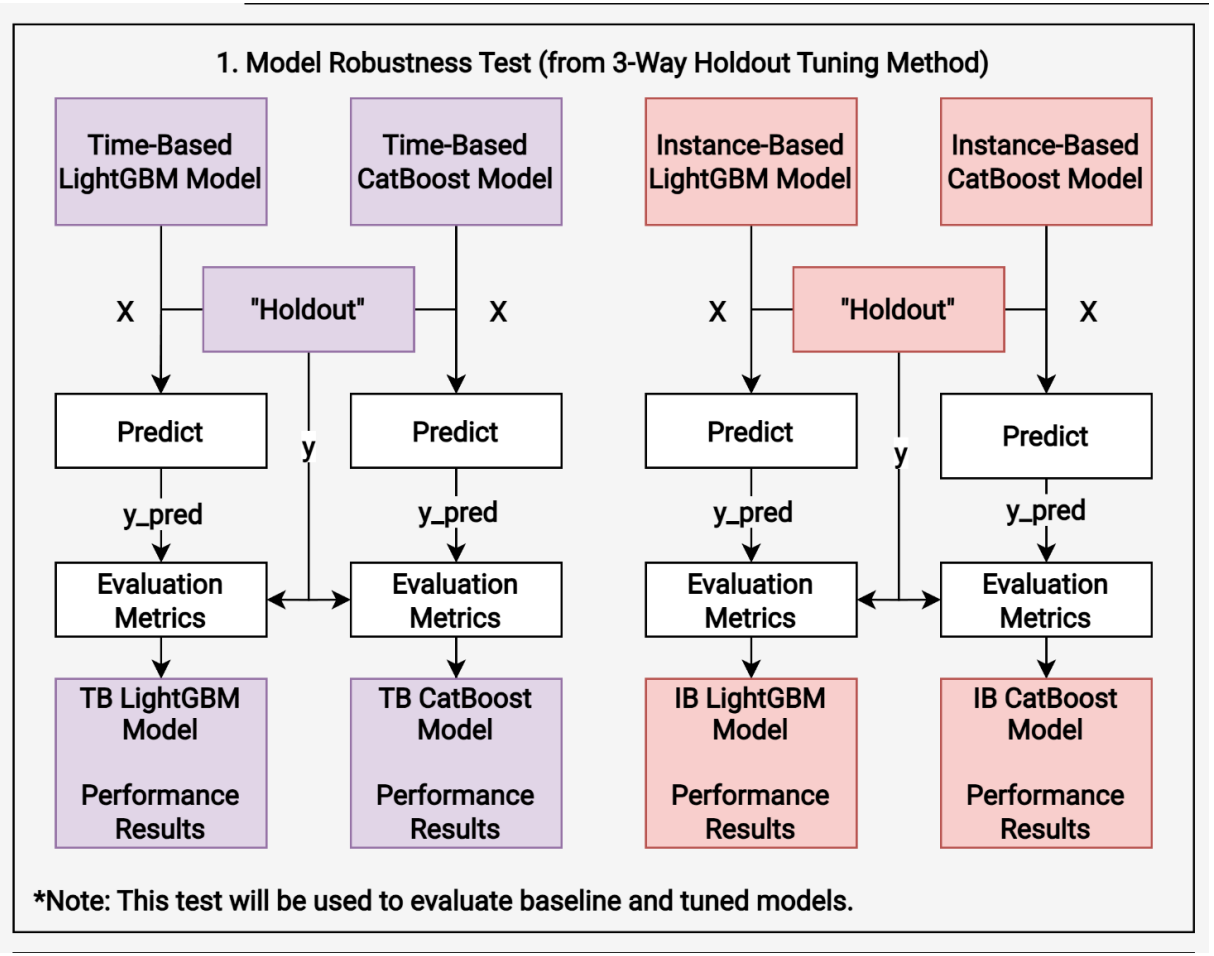
- Evaluation will have three subsections.
  - Model Robustness
  - Repeated Holdout Method (K-Folds CV)
  - Model Comparison
- The Evaluation Metrics will be up for selection but should be at what's found in CH3.





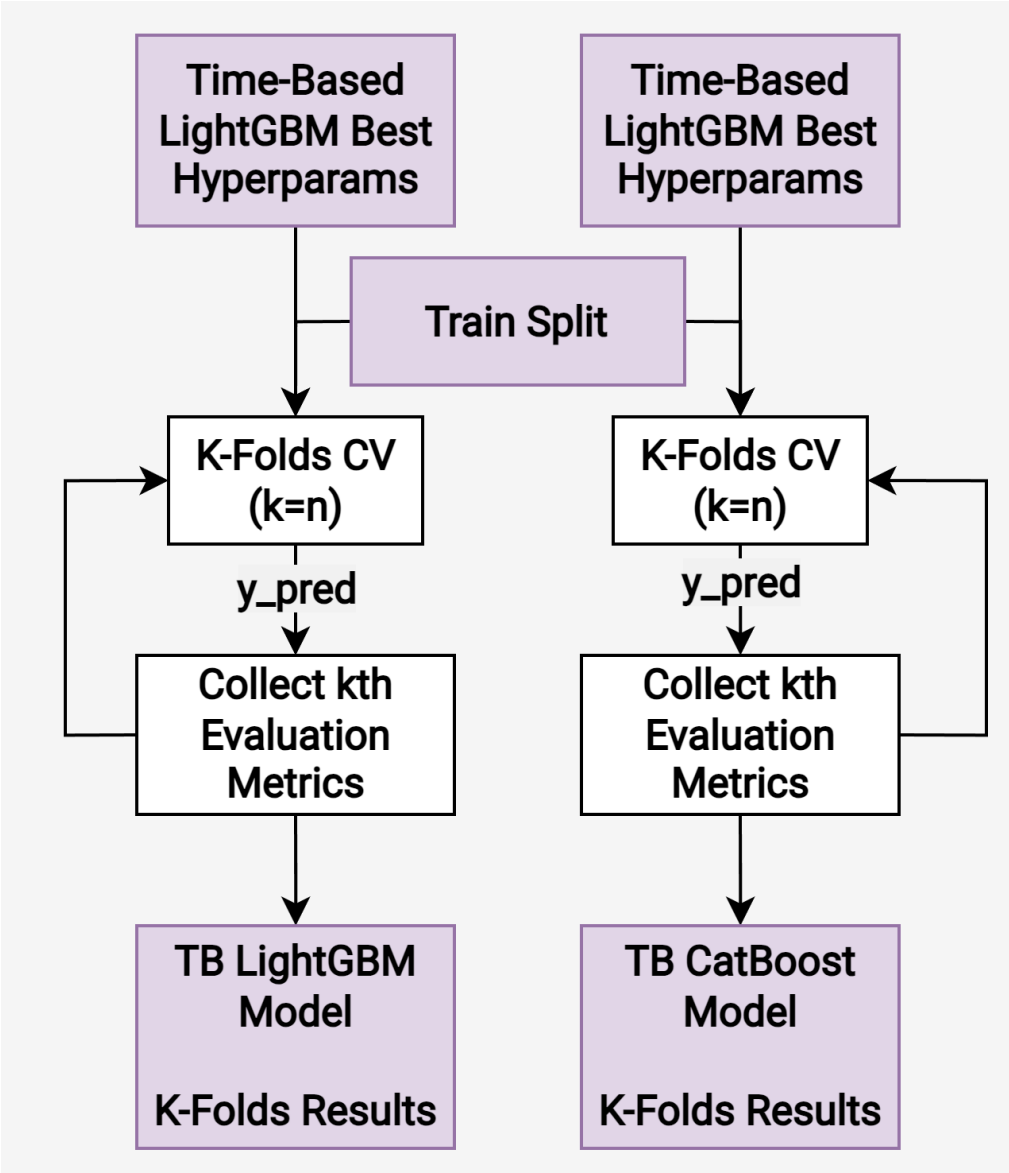
GBDT Evaluation PBD

- Designated as “Model Robustness” as the dataset is not seen by the model during training and tuning.
- Based on 3-Way Holdout Tuning Method.
- For use in both evaluating the default and tuned models (i.e., comparing the effects of tuning if any).
- The dataset that will be used as input will be the “Holdout”.
- This aims to help determine the performance of the model.



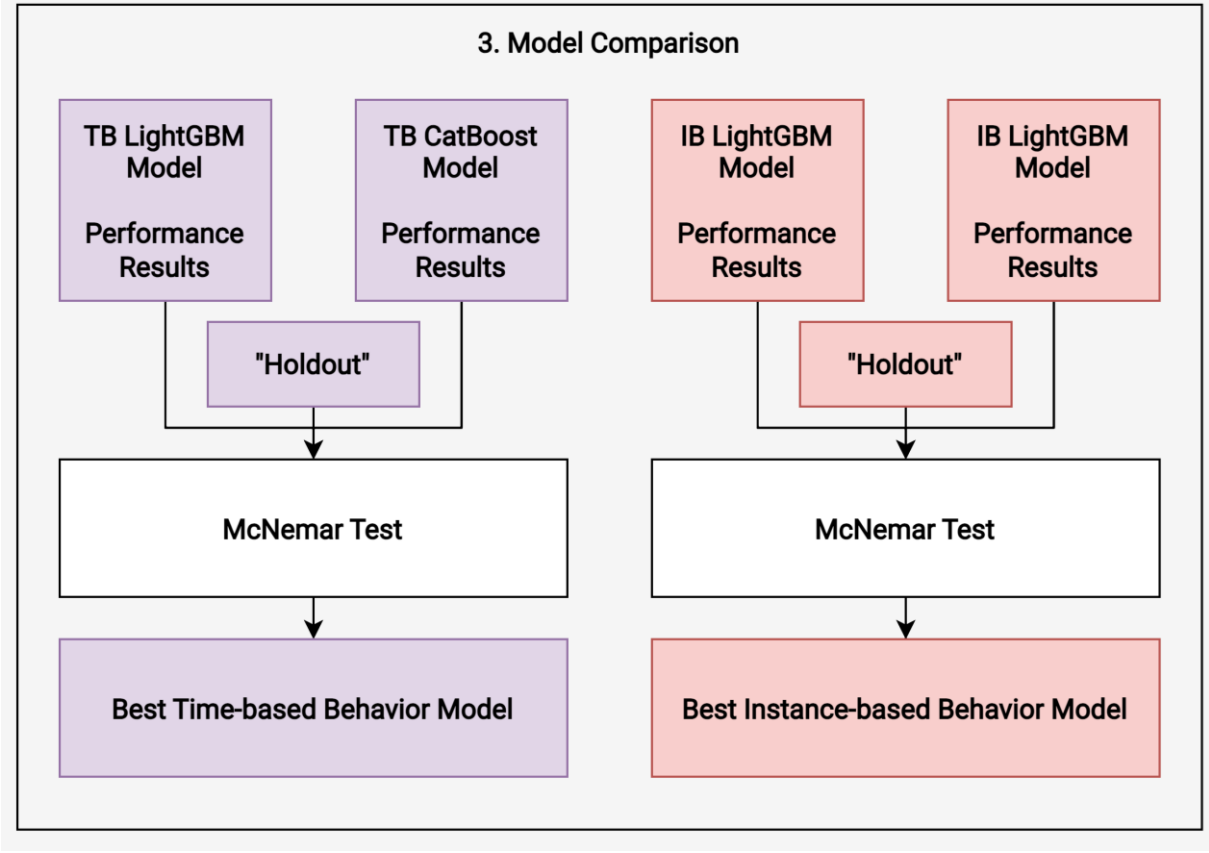
GBDT Evaluation PBD

- Same as ever, we'll be focusing on Time-based for this example.
- "Training" and "Validation" mix will be controlled by Stratified K-Folds CV where no fitting will occur, only predicting.
- This test will be done on both the default and tuned model to determine if overfitting has occurred on either of the models.
- This test aims to check if there is any instances of data where the model is overfitted from the training on the training data used earlier.



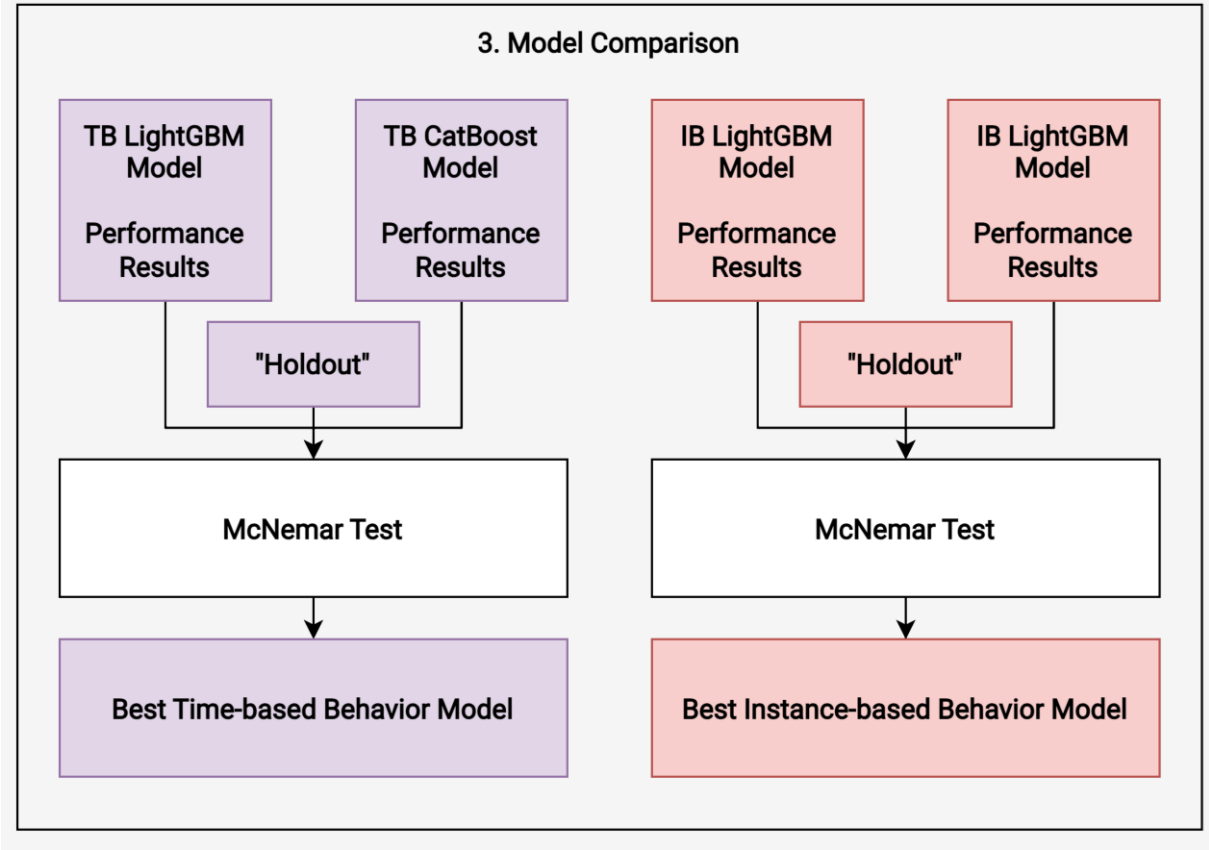
GBDT Evaluation PBD

- Last is Model Comparison where we can evaluate which model is best between LightGBM and CatBoost.
- McNemar Test will be used here which some online sources say is the similar as a “Paired Sample Z-Test”.
- Like statistical tests, it aims to determine if there are any significant differences between LightGBM and CatBoost. If there is then it is possible that the better performing model (by metrics) is actually better statistically speaking.



GBDT Evaluation PBD

- While it is possible to test between TB vs IB (i.e., which offers the best insights regardless of the GBDT model used), it does so with the complications in the differences of the input datasets which goes back to the very reason of doing the test (i.e., evaluate and compare models using the same dataset).



GBDT Evaluation PBD

- Model Robustness
  - Default
    - Time-based Behavior (LightGBM vs CatBoost)
    - Instance-based Behavior (LightGBM vs CatBoost)
  - Tuned
    - Time-based Behavior (LightGBM vs CatBoost)
    - Instance-based Behavior (LightGBM vs CatBoost)
- Stratified K-Folds CV
  - Default
    - Time-based Behavior (LightGBM vs CatBoost)
    - Instance-based Behavior (LightGBM vs CatBoost)
  - Tuned
    - Time-based Behavior (LightGBM vs CatBoost)
    - Instance-based Behavior (LightGBM vs CatBoost)

GBDT Evaluation PBD

- Model Comparison
  - Time-based Behavior (LightGBM vs CatBoost)
  - Instance-based Behavior (LightGBM vs CatBoost)

Last Note

- Remember the concept of controlled and uncontrolled variables?
- For ML development, one could argue that certain processes have a degree of randomness that result to many uncontrolled variables which could muddy the results (i.e., inconsistency). However, that is not the case.
- There is a commonly used parameter (usually) called “random_state” which allows the user to specify the randomization of the process.
- The concept is similar to Minecraft’s world seed on its world generator where you can recreate the same world on different computers using the same seed value.