Broad Functionality: **Challengers**

Principle: **Deployment and Prediction**

Understanding Functionality: Champion/Challenger is an approach to both monitor and measure, the prediction of different possible machine learning models build of the same logic of the decision. The ultimate aim to incorporate this method is to keep track of best performing model.

Does MLflow supports the functionality *directly*: **No**

Does MLflow supports the functionality *indirectly*: **Yes**

- Explanation: MLflow doesn't support the champion/challenger framework directly at the development/production stage. However, A/B testing like experiments can be designed and batch prediction is done by the model at production i.e., Champion can be replayed manually by the challenger model. Since MLflow supports logging of all the results, inferential analysis can be conducted on a weekly/bi-weekly basis to understand the health of models. And based on recall/precision/accuracy champion model can either continue to stay as champion or get replaced by the best performing challenger model.
- Extra remark: This functionality is implicitly executable because MLflow enables saves multiple models are artifacts and analysis can be conducted as MLflow logging lets users save all the metrics and hyperparameters during an experiment run.

Pseudocode or code for MLflow supports the functionality *directly* or *indirectly*:

```
def update_learning(recall, recall_live):
    s3_client = boto3.Session(profile_name=None).client('s3')
    s3_resource = boto3.resource('s3')
    artifact_bucket = 'YOUR ARTIFACT BUCKET ON S3'
    if recall > recall_live:
        # Push live champion to history
        try:
            object = s3_client.get_object(Bucket=artifact_bucket,
                                          Key='mlflow/' + proj_id +
'/live_model_run_history')
            live_hist = object['Body'].read().decode("utf-8")
        except botocore.exceptions.ClientError as e:
            live_hist = ''
        object = s3_resource.Object(artifact_bucket, 'mlflow/' + proj_id +
'/live_model_run_history')
        object.put(Body=live_run + " \n" + live_hist)
        # Promote active challenger to live champion
        object = s3_resource.Object(artifact_bucket, 'mlflow/' + proj_id +
'/live_model_run')
        object.put(Body=active_run)
        print("INFO: New model learnt.")
        return 1
    else:
        print("INFO: Old model has better performance, keeping it.")
    return 0
```