**DARPA D3M TA2 Organization-**

**TA2 core functionalities-**

1. Run as server waiting to hear TA3 client GRPC requests on port 45042.
2. For a given dataset-problem specification,
   1. Search for valid pipelines (solutions).
   2. Produce top 20 (atmost) pipelines. Pipelines can be output as JSON files for evaluation purposes.
   3. Enable scoring, fit, produce on any pipeline. These calls are invoked from TA3.
3. For any given dataset-problem evaluation, TA2 is run on the TRAIN subset to produce top pipelines. Each of these pipelines is scored independently using D3M’s reference runtime framework on the TEST subset.

Pipeline and solution are aliases and mean the same concept in TA2. A pipeline or solution is an end-to-end flowchart (DAG) composed of TA1 primitives, their hyperparameters and their connections to produce predictions on the specified input.

**TA2 code structure-**

1. src/solution\_templates.py:
   1. Contains pipeline templates for different task types, data types and different models being evaluated.
   2. Creates the set of all the pipelines for a dataset. These pipelines have not been scored/fitted as yet. However, common steps of pipelines have been already run in *run\_basic\_solution()* before we copy the basic solution for multiple models (classifiers/regressors).
   3. Pipeline templates for classification/regression are not complete. They require the model (classifier/regressor) to be appended followed by ‘construct\_predictions’ primitive step.
   4. Even for graph-based problems, like vertex nomination, graph matching etc., we add classification templates since they can work with the main table to produce good predictions.
2. src/solutiondescription.py:
   1. Contains class ‘SolutionDescription’ for a single pipeline/solution.
   2. Contains methods to initialize, score, fit, produce, and describe a pipeline.
   3. *initialize\_solution()* is important for constructing the complete pipeline from the respective template. This creates the entire pipeline end-to-end making the appropriate connections.
   4. *run\_basic\_solution()* is used to run common preprocessing/featurizing steps before we copy and spawn multiple processes for evaluating different models (classifiers/regressors). This results in huge savings in time usage when processing complex data types.
   5. *score\_solution()* is used to evaluate pipeline by running k-fold CV on the classifier/regressor model in the pipeline. This corresponds to the second-last step in the pipelines (before construct\_predictions).
3. src/api\_v3/core.py:

Contains server startup and methods for TA2-TA3 API (GRPC calls). See <https://gitlab.com/datadrivendiscovery/ta3ta2-api>. All GRPC messages are a part of core.proto file. Core.proto, pipeline.proto, primitive.proto and value.proto files contain all the information about these message structures.

1. src/search.py:
   1. Runs TA2 in stand-alone evaluation. This is used only for internal purposes. It is not invoked by any TA3 or DMC evaluations.
2. src/\*pb2\*.py:

These are auto-generated files by the rebuild\_grpc.sh script. These handle the GRPC communication between TA2-TA3.

1. src/main.py:

Contains TA2 startup code.

1. src/primitivedescription.py:

Contains code for evaluating models/primitives using k-fold cross-validation.