Table of Contents

Machine-Learning-Specific Refactoring Types

Data Handling Optimization

Description:s

- Optimizing data reading/writing from/to the dataset.

Example:

- https://github.com/chemprop/chemprop/commit/ae8ae8af6ad4e92e4edae62699fb655a77a3b0b7

```diff
85 chemprop/data/utils.py
@@ -97,26 +97,55 @@ def get_smiles(path: str) -> List[str]:
 97  97      :return: A list of smiles strings.
 98  98      """
 99  99      with open(path) as f:
100      -       f.readline()  # Skip header
101      -       smiles = [line.strip().split(',')[0] for line in f]
     100 +       reader = csv.reader(f)
     101 +       next(reader)  # Skip header
     102 +       smiles = [line[0] for line in reader]
102 103
103 104      return smiles
```

⇒ The refactoring is accomplished by replacing manual line reading and header skipping with the "csv.reader" object, specifically designed for CSV file reading.

Model Initialization Refinement

Description:

- Modify the hyper-parameter initializations in the models.

Example:

- https://github.com/scikit-learn/scikit-learn/commit/14ecaa19c66d2af94e110268410e903fc8ffc6dd

```diff
16 sklearn/tests/test_hmm.py
@@ -182,10 +182,13 @@ def test_base_hmm_attributes(self):
182 182
183 183
184 184 def train_hmm_and_keep_track_of_log_likelihood(hmm, obs, n_iter=1, **kwargs):
185     -   hmm.fit(obs, n_iter=1, **kwargs)
    185 +   hmm.n_iter = 1
    186 +   hmm.fit(obs)
186 187     loglikelihoods = []
187 188     for n in xrange(n_iter):
188     -       hmm.fit(obs, n_iter=1, init_params='', **kwargs)
    189 +       hmm.n_iter = 1
    190 +       hmm.init_params = ''
    191 +       hmm.fit(obs)
```

⇒ The refactoring involves breaking down the model function call into individual steps, explicitly setting the parameters "n_iter" and "init_params" before calling the "fit" method to match the library API.

Resource Allocation Optimization

Description:

- Adjusting hardware usage parameters.

Example:

- https://github.com/mars-project/mars/commit/b2316ea69686dd5246c46cd8a47f6f67d8f42faa

```diff
5 mars/deploy/kubernetes/tests/test_kubernetes.py
@@ -160,7 +160,8 @@ def _start_kube_cluster(use_test_docker_file=True, **kwargs):
 160  160  @pytest.mark.skipif(not kube_available, reason='Cannot run without kubernetes')
 161  161  def test_run_in_kubernetes(use_test_docker_file):
 162  162      with _start_kube_cluster(
 163       -            worker_mem='1G', worker_cache_mem='128m',
      163  +            supervisor_cpu=0.5, supervisor_mem='1G',
      164  +            worker_cpu=0.5, worker_mem='1G', worker_cache_mem='64m',
 164  165              extra_labels={'mars-test/group': 'test-label-name'},
```

⇒ The refactoring in this code involves adding more specific parameters for CPU allocation "supervisor_cpu" and "worker_cpu" and adjusting memory allocations "worker_cache_mem" while maintaining the overall functionality of the "_start_kube_cluster" function call.

Data Type Clarification

Description:

- Converting datatypes from one to another (e.g., from NumPy to DataFrame).

Example:

- https://github.com/q-optimize/c3/commit/2ea2f0b9639ea572458563f3b53d06505d4a4e2f

```diff
 31  38          self.pmap = pmap
 32       -      self.optim_status = {}
 33       -      self.gradients = {}
      39  +      self.optim_status: Dict[str, Any] = dict()
      40  +      self.gradients: Dict[str, np.ndarray] = {}
 34  41          self.current_best_goal = 9876543210.123456789
```

⇒ This change introduces type hinting by explicitly specifying the type of "self.optim_status" as a dictionary "Dict[str, Any]" and initializing it with an empty dictionary using "dict()". This can improve code readability and help catch type-related errors early during development.

Data Presentation Enhancement

Description:

- Refactoring data visualization code.

Example:

- https://github.com/biolab/orange3/commit/c2d3a8a30a5e68a13512bdc5ad85a5ba0b2a0053

```
65    -        box = gui.widgetBox(self.controlArea, "Selection")
      61 +     box = gui.widgetBox(self.controlArea, "Select")
```

⇒ renaming “Selection” to “Select” in the widget box title aimed at improving the clarity and consistency of the plot.

Data Path Management

Description:

- Managing dataset/plots/models storage paths.

Example:

- https://github.com/automl/auto-sklearn/commit/2388087e5b347ae7d6425afaf878a8f279bd663e

```
50    49    def main():
51       -     datasets = 'resources/datasets.csv'
      50 +     datasets = 'datasets.csv'
```

⇒ moving the CSV dataset file into a “resources” directory can make the code more organized and easier to manage, which doesn’t change external functionalities.

Mathematical Operation Refactoring

Description:

- Changing mathematical calculation to a simpler form.

Example:

- https://github.com/pyro-ppl/pyro/commit/4f5940971cb0c7c42f81450ce70b6244551ace02

```
2 pyro/distributions/hmm.py
@@ -125,7 +125,7 @@ def log_prob(self, value):
125   125        result = _sequential_logmatmulexp(result)
126   126
127   127        # Combine initial factor.
128      -       result = _logmatmulexp(self.initial_logits.unsqueeze(-2), result).squeeze(-2)
      128 +      result = self.initial_logits + result.logsumexp(-1)
```

⇒ simplifying the code by replacing a custom operation "_logmatmulexp" with a more straightforward operation "logsumexp" and using existing functionality "initial_logits" in a more direct manner

General Refactoring Types

Code Cleanup

Description:

- Removing unused code or files.

Example:

- https://github.com/stellargraph/stellargraph/commit/6a3b0e5119192df3369e85b760e1cb1106a53a42

```
3 stellargraph/layer/graph_classification.py
@@ -31,10 +31,8 @@
31  31  from .misc import deprecated_model_function
32  32  from ..mapper import GraphGenerator
33  33  from .cluster_gcn import ClusterGraphConvolution
34      - from ..core.experimental import experimental
35  34
36  35
37      - @experimental(reason="Missing unit tests.", issues=[1042])
38  36  class GraphClassificationConvolution(ClusterGraphConvolution):
39  37
40  38      """
@@ -96,7 +94,6 @@ def call(self, inputs):
96  94          return output
97  95
98  96
99      - @experimental(reason="Missing unit tests")
100 97  class GCNSupervisedGraphClassification:
101 98      """
102 99      A stack of :class:`GraphClassificationConvolution` layers together with a Keras `GlobalAveragePooling1D` layer
```

⇒ remove the experimental marking and related metadata from the specified code segment, which does not change the output of the code.

Code Simplification

Description:

- Replacing code with more simplified alternatives.

Example:

- https://github.com/AI4Finance-Foundation/ElegantRL/commit/c8cf5d0a3d2bbdfdd5380f37cf77fdbd8ffaa3e7

```
319     -       # action = dist.sample()
320     -       samples_2d = torch.multinomial(a_prob, num_samples=1, replacement=True)
321     -       action = samples_2d.reshape(state.size(0))
322     -       return action, a_prob
    371 +       # a_int = dist.sample()
    372 +       a_int = torch.multinomial(a_prob, num_samples=1, replacement=True)[:, 0]
    373 +       # samples_2d = torch.multinomial(a_prob, num_samples=1, replacement=True)
    374 +       # samples_2d.shape == (batch_size, num_samples)
    375 +       return a_int, a_prob
```

⇒ the variable "action" is replaced by "a_int", and the reshaping operation is removed.

Import/Export Optimization

Description:

- Removing or relocating unused dependencies.

Example:

- https://github.com/deepchem/deepchem/commit/25f19f49668cf0ff28a87e8aa4719f5d429d299b

```
28 deepchem/models/__init__.py
@@ -9,9 +9,9 @@
 9   9   import pandas as pd
10  10   import joblib
11  11   import os
12      - from deepchem.utils.dataset import NumpyDataset
13      - from deepchem.utils.dataset import load_sharded_dataset
14      - from deepchem.utils.dataset import save_sharded_dataset
    12  + from deepchem.utils.dataset import ShardedDataset
    13  + from deepchem.utils.dataset import load_from_disk
    14  + from deepchem.utils.dataset import save_to_disk
```

⇒ Changing multiple imports with a similar library in data-load/featurization.

Logging Enhancement

Description:

- Improving log messages for user understanding or modify them for readability/consistency.

Example:

- https://github.com/RasaHQ/rasa/commit/61ba44a1be9e674566ab20930f217e8fdd363ebb

```
2 rasa/nlu/model.py
@@ -243,7 +243,7 @@ def persist(
243 243         if persistor is not None:
244 244             persistor.persist(dir_name, model_name)
245 245         logger.info(
246     -           "Successfully saved model into " "'{}'".format(os.path.abspath(dir_name))
    246 +           "Successfully saved model into '{}'".format(os.path.abspath(dir_name))
247 247         )
248 248         return dir_name
249 249
```

⇒ improves the readability and consistency of the code by removing redundant string concatenation and using a single format specifier for the placeholder.

Requiremesnt/Configuration Update

Description:

- Updating program requirements or configurations.

Example:

- https://github.com/hi-primus/optimus/commit/436f7f1923ba6fd8cc410f16ab93461ccfc0e37a

```diff
requirements.txt
@@ -11,7 +11,6 @@ nose==1.3.7
 ipython==6.5.0
 seaborn==0.8.1
 setuptools==40.1.0
-quinn==0.2.1
 deprecated==1.2.0
 pyarrow==0.9.*
 tabulate==0.8.2
@@ -21,5 +20,4 @@ keras==2.1.5
 pillow>=4.1.1,<4.2
 pygments>=2.2.0
 six>=1.10.0
-h5py>=2.7.0
-timber
+h5py>=2.7.0
```

⇒ Remove unused dependencies from the requirements.

Condition Simplification

Description:

- Simplifying complex conditional statements.

Example:

- https://github.com/ray-project/ray/commit/8f59546ef2fba5af2666c27e5480f6819389fedf

```diff
-                if row_metadata is not None:
-                    self._row_metadata = row_metadata.copy()
-                if col_metadata is not None:
-                    self._col_metadata = col_metadata.copy()
                assert self._block_partitions.ndim == 2, \
                    "Block Partitions must be 2D."
            else:
                if row_partitions is not None:
                    axis = 0
                    partitions = row_partitions
-                   if row_metadata is not None:
-                       self._row_metadata = row_metadata.copy()
                elif col_partitions is not None:
                    axis = 1
                    partitions = col_partitions
-                   if col_metadata is not None:
-                       self._col_metadata = col_metadata.copy()

                self._block_partitions = \
                    _create_block_partitions(partitions, axis=axis,
                                             length=len(columns))

+        if row_metadata is not None:
+            self._row_metadata = row_metadata.copy()
+        if col_metadata is not None:
+            self._col_metadata = col_metadata.copy()
+
```

⇒ The code combines several nested conditions into two straightforward conditions.

File/Dependency Path Refinement

Description:

- Optimizing/updating dependency/file path.

Example:

- https://github.com/Drakkar-Software/OctoBot/commit/3ce0df4d77976a0cd78ada75803f7fa5b8b9f522

```
8 ■■■■ config/cst.py
@@ -6,7 +6,7 @@
6  6   VERSION = "{0}-{1}".format(SHORT_VERSION, VERSION_DEV_PHASE)
7  7   LONG_VERSION = "{0}_{1}-{2}".format(SHORT_VERSION, REV_VERSION, VERSION_DEV_PHASE)
8  8
9     - ORIGIN_URL = "https://github.com/Trading-Bot/CryptoBot.git"
   9  + ORIGIN_URL = "https://github.com/Drakkar-Software/Octobot.git"
10 10
```

⇒ Change the directory of Cryptobot with Octobot