CS480_Project_v4

August 12, 2024

[ ]:
```
! pip install --quiet "xlrd" "ipython[notebook]==7.34.0, <8.17.0" "openpyxl"␣
↪"fastparquet" "lightgbm" "pyarrow" "setuptools>=68.0.0, <68.3.0"  "xgboost"␣
↪"catboost" "tensorboard" "lightning>=2.0.0" "urllib3" "torch==2.3.0"␣
↪"matplotlib" "optuna" "pytorch-lightning>=1.4, <2.1.0" "seaborn"␣
↪"torchvision" "torchmetrics>=0.7, <1.3" "matplotlib>=3.0.0, <3.9.0"
```

[ ]:
```
import os

import lightning as L
import matplotlib.pyplot as plt
import matplotlib_inline.backend_inline
import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import seaborn as sns
from lightning.pytorch.callbacks import LearningRateMonitor, ModelCheckpoint,␣
↪EarlyStopping
from torchvision import transforms
import pandas as pd
import numpy as np
from sklearn.metrics import r2_score, mean_squared_error

TRAIN_IMAGES_DATA_PATH = "./train_images"
TEST_IMAGES_DATA_PATH = "./test_images"
CHECKPOINT_PATH = os.environ.get("PATH_CHECKPOINT", "saved_models/")

plt.set_cmap("cividis")
%matplotlib inline
matplotlib_inline.backend_inline.set_matplotlib_formats("svg", "pdf")  # For␣
↪export
sns.reset_orig()

L.seed_everything(42)
torch.backends.cudnn.deterministic = True
torch.backends.cudnn.benchmark = False
```

```
device = (
    "cuda"
    if torch.cuda.is_available()
    else "mps"
    if torch.backends.mps.is_available()
    else "cpu"
)
print(f"Using {device} device")
```

```
Seed set to 42
```

```
Using mps device
```

```
<Figure size 640x480 with 0 Axes>
```

[ ]:
```
from torch.utils.data import random_split
from torch.utils.data import DataLoader
from data_set import PlantDataset

from sklearn.preprocessing import MinMaxScaler, StandardScaler

img_transform_train = transforms.Compose([
    transforms.Resize(size=(224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])

img_transform_test = test_transform = transforms.Compose(
    [
        transforms.Resize(size=(224,224)),
        transforms.ToTensor(),
        transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.
 ↪225]),
    ]
)

# do not normalize CSV features until split to avoid data leakage
partial_training_data = PlantDataset('train.csv', 'train_images', num_labels=6,␣
 ↪image_transform=img_transform_train)
partial_test_data = PlantDataset('test.csv', 'test_images', num_labels=0,␣
 ↪image_transform=img_transform_test)
```

[ ]:
```
from data_set import AugmentedDataset

# Augment data sets
model = torch.hub.load('facebookresearch/dinov2', 'dinov2_vits14_reg').
 ↪to(device)
```

```
full_training_data = AugmentedDataset(partial_training_data, model,␣
 ↪"train_embeddings.parquet", device=device)
test_data = AugmentedDataset(partial_test_data, model, "test_embeddings.
 ↪parquet", device=device)
```

```
Using cache found in
/Users/sriharivishnu/.cache/torch/hub/facebookresearch_dinov2_main
/Users/sriharivishnu/.cache/torch/hub/facebookresearch_dinov2_main/dinov2/layers
/swiglu_ffn.py:51: UserWarning: xFormers is not available (SwiGLU)
  warnings.warn("xFormers is not available (SwiGLU)")
/Users/sriharivishnu/.cache/torch/hub/facebookresearch_dinov2_main/dinov2/layers
/attention.py:33: UserWarning: xFormers is not available (Attention)
  warnings.warn("xFormers is not available (Attention)")
/Users/sriharivishnu/.cache/torch/hub/facebookresearch_dinov2_main/dinov2/layers
/block.py:40: UserWarning: xFormers is not available (Block)
  warnings.warn("xFormers is not available (Block)")
```

[ ]:
```
# train_size = len(full_training_data)

# val_size = int(0.1 * len(full_training_data))
# train_size = train_size - val_size

# L.seed_everything(42)
# train_data, _ = random_split(full_training_data, [train_size, val_size])
# L.seed_everything(42)
# _, val_data = random_split(full_val_data, [train_size, val_size])

# # Define datasets
# X_train, Y_train = full_training_data.csv_aug.iloc[train_data.indices].
 ↪copy(), full_training_data.labels.iloc[train_data.indices].copy()
# X_val, Y_val = full_training_data.csv_aug.iloc[val_data.indices].copy(),␣
 ↪full_training_data.labels.iloc[val_data.indices].copy()
# X_test = test_data.csv_aug.copy()

# # Normalize data

# def log_transform(x):
#     return np.log10(x)

# def inverse_log_transform(x):
#     return 10**x

# from sklearn.pipeline import Pipeline
# from sklearn.preprocessing import RobustScaler, FunctionTransformer
# X_pipeline = Pipeline([('scaler', RobustScaler())])
# # ('log', FunctionTransformer(log_transform, inverse_log_transform))
# Y_pipeline = Pipeline([('scaler', StandardScaler())])
```

```
# X_train[X_train.columns] = X_pipeline.fit_transform(X_train)
# Y_train[Y_train.columns] = Y_pipeline.fit_transform(Y_train)

# X_val[X_val.columns] = X_pipeline.transform(X_val)
# Y_val[Y_val.columns] = Y_pipeline.transform(Y_val)

# X_test[X_test.columns] = X_pipeline.transform(X_test)
# X_train
```

```
Seed set to 42
Seed set to 42
```

[ ]:
```
       WORLDCLIM_BIO1_annual_mean_temperature  \
331                                 0.452227
37716                              -0.139200
24869                              -1.551352
38577                               0.866574
15658                              -0.354403
…                                        …
36103                               0.767314
21747                              -1.315561
27730                              -0.545294
13788                               0.763963
6707                               -0.324525

       WORLDCLIM_BIO12_annual_precipitation  \
331                                0.451728
37716                             -0.242720
24869                             -0.674094
38577                             -0.434391
15658                             -0.139690
…                                       …
36103                             -1.011357
21747                              0.336684
27730                             -0.563163
13788                              2.816521
6707                               1.740798

       WORLDCLIM_BIO13.BIO14_delta_precipitation_of_wettest_and_dryest_month  \
331                                                 0.308841
37716                                              -0.522154
24869                                              -0.050457
38577                                               0.290840
15658                                              -0.467796
…                                                        …
36103                                              -0.632652
```

```
21747                                          0.051645
27730                                         -0.265236
13788                                          1.981584
6707                                           0.550868

       WORLDCLIM_BIO15_precipitation_seasonality  \
331                                      -0.097577
37716                                    -0.681169
24869                                     1.305822
38577                                     0.978099
15658                                    -0.561669
…                                            …
36103                                     0.932182
21747                                    -0.201485
27730                                     0.226353
13788                                    -0.079494
6707                                     -0.453219

       WORLDCLIM_BIO4_temperature_seasonality  \
331                                   -0.139388
37716                                 -0.191477
24869                                  2.452484
38577                                 -0.021739
15658                                  0.199484
…                                         …
36103                                 -0.129143
21747                                  0.382205
27730                                  0.935151
13788                                 -0.963297
6707                                  -0.259296

       WORLDCLIM_BIO7_temperature_annual_range  SOIL_bdod_0.5cm_mean_0.01_deg  \
331                                   -0.122118                       0.083333
37716                                 -0.203778                      -1.291667
24869                                  2.433335                       0.041667
38577                                  0.144648                       1.291667
15658                                 -0.078876                       0.000000
…                                         …                             …
36103                                 -0.017085                       1.125000
21747                                 -0.140544                      -0.708333
27730                                  1.289888                       0.875000
13788                                 -0.901999                      -0.458333
6707                                  -0.477283                      -1.375000

       SOIL_bdod_100.200cm_mean_0.01_deg  SOIL_bdod_15.30cm_mean_0.01_deg  \
331                                -0.40                            -0.10
37716                              -0.80                            -1.35
```

```
24869                                0.50                                0.25
38577                                0.55                                1.05
15658                                0.20                                0.20
…                                     …                                   …
36103                                0.40                                0.90
21747                               -0.30                               -0.70
27730                                0.85                                1.20
13788                               -1.00                               -0.60
6707                                -1.65                               -1.70

       SOIL_bdod_30.60cm_mean_0.01_deg  …      1526      1527      1528  \
331                              -0.15  … -0.035865  0.302029 -0.028921
37716                            -1.20  …  1.030290 -1.405956 -0.473035
24869                             0.25  … -0.038121  0.712303 -0.196621
38577                             0.85  … -1.071165  0.485870 -0.897550
15658                             0.35  … -0.638814  0.159860  0.338273
…                                  …  …       …        …        …
36103                             0.50  …  0.181640 -0.516820  0.659337
21747                            -0.55  …  0.000491  0.794809  1.103474
27730                             1.00  … -0.164043  1.075969  0.572995
13788                            -0.80  … -1.131211  0.145449 -0.066013
6707                             -1.85  … -0.108057 -0.641189  0.125625

           1529      1530      1531      1532      1533      1534      1535
331   -1.047511  0.447232 -0.904520 -0.359341  1.008899 -0.872129  0.327152
37716 -0.119309  1.238882  0.353272  0.504527  0.236974  1.044594 -0.892983
24869  0.343609 -0.799219  0.075258 -0.692054 -0.176775 -0.928963 -0.459565
38577 -0.546678 -0.063755  0.177236 -0.154138  0.496182  1.650604 -0.567907
15658  0.763841 -0.793664 -0.294099  0.495492 -0.538217 -1.026370 -0.227260
…         …       …         …         …         …         …
36103 -1.764627 -0.063433  0.301784 -0.873698  0.503186  0.619144 -1.124390
21747  0.353248 -0.895271 -0.276857 -0.959812  0.489866 -0.275102 -0.472095
27730 -0.557596  0.717719  1.060919  0.992661 -0.223296  1.465056  1.803320
13788  0.159567  0.452972  0.618598  0.359918 -0.188242 -1.309235 -0.536108
6707   0.205087 -0.982614  0.323461  0.455706 -0.361441  0.610351  0.090198

[39027 rows x 1699 columns]
```

[ ]:
```
from sklearn.utils import Bunch
from sklearn.utils.validation import _check_method_params, has_fit_parameter
from sklearn.base import is_classifier, _routing_enabled
from joblib import Parallel, delayed
from sklearn.multioutput import _fit_estimator, process_routing,␣
 ↪check_classification_targets
from sklearn.multioutput import MultiOutputRegressor

class PlantTraitRegressor(MultiOutputRegressor):
```

```
    def fit(self, X, y, sample_weight=None, **fit_params):
        """Fit the model to data, separately for each output variable.

        Parameters
        ----------
        X : {array-like, sparse matrix} of shape (n_samples, n_features)
            The input data.

        y : {array-like, sparse matrix} of shape (n_samples, n_outputs)
            Multi-output targets. An indicator matrix turns on multilabel
            estimation.

        sample_weight : array-like of shape (n_samples,), default=None
            Sample weights. If `None`, then samples are equally weighted.
            Only supported if the underlying regressor supports sample
            weights.

        **fit_params : dict of string -> object
            Parameters passed to the ``estimator.fit`` method of each step.

            .. versionadded:: 0.23

        Returns
        -------
        self : object
            Returns a fitted instance.
        """
        if not hasattr(self.estimator, "fit"):
            raise ValueError("The base estimator should implement a fit method")

        y = self._validate_data(X="no_validation", y=y, multi_output=True)

        if is_classifier(self):
            check_classification_targets(y)

        if y.ndim == 1:
            raise ValueError(
                "y must have at least two dimensions for "
                "multi-output regression but has only one."
            )

        if _routing_enabled():
            if sample_weight is not None:
                fit_params["sample_weight"] = sample_weight
            routed_params = process_routing(
                self,
                "fit",
```

```
                **fit_params,
            )
        else:
            if sample_weight is not None and not has_fit_parameter(
                self.estimator, "sample_weight"
            ):
                raise ValueError(
                    "Underlying estimator does not support sample weights."
                )

            fit_params_validated = _check_method_params(X, params=fit_params)
            routed_params = Bunch(estimator=Bunch(fit=fit_params_validated))
            if sample_weight is not None:
                routed_params.estimator.fit["sample_weight"] = sample_weight


        eval_set = routed_params.estimator.fit.pop('eval_set')

        if type(eval_set) is list:
            X_val, Y_val = eval_set[0]
            Y_val = self._validate_data(X="no_validation", y=Y_val,␣
↪multi_output=True)

            self.estimators_ = Parallel(n_jobs=self.n_jobs)(
                delayed(_fit_estimator)(
                    self.estimator, X, y[:, i], eval_set=[(X_val, Y_val[:,␣
↪i])], **routed_params.estimator.fit
                )
                for i in range(y.shape[1])
            )
        else:
            X_val, Y_val = eval_set
            Y_val = self._validate_data(X="no_validation", y=Y_val,␣
↪multi_output=True)

            self.estimators_ = Parallel(n_jobs=self.n_jobs)(
                delayed(_fit_estimator)(
                    self.estimator, X, y[:, i], eval_set=(X_val, Y_val[:, i]),␣
↪**routed_params.estimator.fit
                )
                for i in range(y.shape[1])
            )

        if hasattr(self.estimators_[0], "n_features_in_"):
            self.n_features_in_ = self.estimators_[0].n_features_in_
        if hasattr(self.estimators_[0], "feature_names_in_"):
            self.feature_names_in_ = self.estimators_[0].feature_names_in_
```

```
        return self
```

[ ]:
```
from data_set import PandasDataset

class MLPModel(L.LightningModule):
    def __init__(self, input_dim=1699, output_dim=6, lr=5e-4, **kwargs):
        super().__init__()
        self.save_hyperparameters()
        self.body = nn.Sequential(
            nn.Linear(input_dim, 1024),
            nn.GELU(),
            nn.Linear(1024, 256),
            nn.GELU(),
            nn.Linear(256, output_dim)
        )

    def forward(self, row):
        x = self.body(row)
        return x

    def configure_optimizers(self):
        optimizer = optim.AdamW(self.parameters(), lr=self.hparams.lr,␣
↪weight_decay=0.0005)
        return [optimizer], []

    def _calculate_loss(self, batch, mode="train"):
        rows, labels = batch

        preds = self.forward(rows)
        loss = F.mse_loss(torch.squeeze(preds), torch.squeeze(labels))
        self.log(f"{mode}_loss", loss)
        self.log(f"{mode}_r2", r2_score(labels.cpu().numpy(), preds.detach().
↪cpu().numpy()))
        return loss

    def training_step(self, batch, batch_idx):
        return self._calculate_loss(batch, mode="train")

    def validation_step(self, batch, batch_idx):
        return self._calculate_loss(batch, mode="val")

    def test_step(self, batch, batch_idx):
        pass

    def predict(self, X : pd.DataFrame):
```

```
        with torch.no_grad():
            return self.forward(torch.tensor(X.values, dtype=torch.float32).
 ↪to(device)).cpu().numpy()

    def score(self, X : pd.DataFrame, Y: pd.DataFrame):
        return r2_score(Y, self.predict(X))

class MyDataModule(L.LightningDataModule):
    def __init__(self, X_train, Y_train, X_val, Y_val, batch_size=512):
        super().__init__()
        self.mlp_train_set = PandasDataset(X_train, Y_train)
        self.mlp_val_set = PandasDataset(X_val, Y_val)
        self.batch_size = batch_size

    def train_dataloader(self):
        return DataLoader(self.mlp_train_set, batch_size=self.batch_size,␣
 ↪shuffle=True, num_workers=2)

    def val_dataloader(self):
        return DataLoader(self.mlp_val_set, batch_size=self.batch_size,␣
 ↪num_workers=2)

    def test_dataloader(self):
        pass


import os
def train_mlp_model(X_train, Y_train, X_val, Y_val, batch_size=256,␣
 ↪dry_run=False, run_num=0, **kwargs):
    mlp_data_module = MyDataModule(X_train, Y_train, X_val, Y_val,␣
 ↪batch_size=batch_size)
    trainer = L.Trainer(
        default_root_dir=os.path.join(CHECKPOINT_PATH, f"{run_num}/mlp/"),
        accelerator="auto",
        devices=1,
        max_epochs=5 if not dry_run else 1,
        callbacks=[
            ModelCheckpoint(save_weights_only=True, mode="max",␣
 ↪monitor="epoch"),
            EarlyStopping(monitor='val_r2', patience=1, mode="max"),
            LearningRateMonitor("epoch"),
        ],
        enable_progress_bar=False
    )
    trainer.logger._log_graph = True
    trainer.logger._default_hp_metric = None
```

```
    model = MLPModel(
        input_dim=X_train.shape[1],
        output_dim=Y_val.shape[1] if not dry_run else 1,
        **kwargs
    )
    trainer.fit(model, datamodule=mlp_data_module)

    # Load the best checkpoint after training
    model = MLPModel.load_from_checkpoint(trainer.checkpoint_callback.
 ↪best_model_path)

    return model
```

[ ]:
```
# Opens tensorboard in notebook. Adjust the path to your CHECKPOINT_PATH!
%reload_ext tensorboard
%tensorboard --logdir ./saved_models
```

[ ]:
```
from sklearn.model_selection import KFold, ShuffleSplit
import xgboost
import catboost
import lightgbm

from sklearn.linear_model import Ridge, Lasso
from sklearn.neighbors import KNeighborsRegressor
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import RobustScaler

# do a quick run through
dry_run = False

kf = ShuffleSplit(
    n_splits=5 if not dry_run else 1,
    random_state=42,
    test_size=0.1
)
preds_test = np.zeros((len(test_data), 6 if not dry_run else 1))

if dry_run:
    print ("Dry running code...")


for i, (train_index, test_index) in enumerate(kf.split(full_training_data.
 ↪csv_aug, full_training_data.labels)):
    # train sets
    X_train = full_training_data.csv_aug.iloc[train_index].copy()
```

```
    Y_train = full_training_data.labels.iloc[train_index].copy()

    # validation sets
    X_val = full_training_data.csv_aug.iloc[test_index].copy()
    Y_val = full_training_data.labels.iloc[test_index].copy()

    if dry_run:
        Y_train = pd.DataFrame(Y_train.iloc[:, 0])
        Y_val = pd.DataFrame(Y_val.iloc[:, 0])

    # Test set
    X_test = test_data.csv_aug.copy()

    # for boosting algorithms, it can be beneficial to engineer some features
    poly = PolynomialFeatures(2)

    # Randomly select 1000 extra polynomial features
    num_extra_features = 1000
    poly.fit(X_train.iloc[:, :163])
    random_extra_features = np.random.choice(range(163, poly.
↪n_output_features_), num_extra_features, replace=False)

    # Augment each of the corresponding feature sets
    X_train_extra_features = pd.DataFrame(np.concatenate((X_train.values, poly.
↪transform(X_train.iloc[:, :163])[:, random_extra_features]), axis=1))
    X_val_extra_features = pd.DataFrame(np.concatenate((X_val.values,  poly.
↪transform(X_val.iloc[:, :163])[:, random_extra_features]), axis=1))
    X_test_extra_features = pd.DataFrame(np.concatenate((X_test.values, poly.
↪transform(X_test.iloc[:, :163])[:, random_extra_features]), axis=1))

    columns_no_embed = X_train_extra_features.columns

    # for catboost, add embeddings
    X_train_extra_features['emb'] = list(X_train.iloc[:, 163:].values)
    X_val_extra_features['emb'] = list(X_val.iloc[:, 163:].values)
    X_test_extra_features['emb'] = list(X_test.iloc[:, 163:].values)

    # For the other models, need to normalize
    X_pipeline = Pipeline([('scaler', RobustScaler())])
    Y_pipeline = Pipeline([('scaler', StandardScaler())])

    X_train[X_train.columns] = X_pipeline.fit_transform(X_train)
    Y_train[Y_train.columns] = Y_pipeline.fit_transform(Y_train)

    X_val[X_val.columns] = X_pipeline.transform(X_val)
    Y_val[Y_val.columns] = Y_pipeline.transform(Y_val)
```

```
    X_test[X_test.columns] = X_pipeline.transform(X_test)

    best_xgb = {
        "objective": "reg:squarederror",
        "n_estimators": 1000 if not dry_run else 1,
        "learning_rate": 0.029604246449770312,
        "max_depth": 8,
        "subsample": 0.8080014405993786,
        "colsample_bytree": 0.6684075982840267,
        "min_child_weight": 20
    }
    xgb = MultiOutputRegressor(
        xgboost.XGBRegressor(
            **best_xgb
        ),
        n_jobs=3 if not dry_run else 1
    )
    print("Xgb: ",
          xgb.fit(
              X_train_extra_features[columns_no_embed],
              Y_train, verbose=False
            ).score(X_val_extra_features[columns_no_embed], Y_val))
    os.makedirs(f"{os.getcwd()}/saved_models/{i}/xgb/", exist_ok=True)
    for j in range(len(xgb.estimators_)):
        xgb.estimators_[j].save_model(f"{os.getcwd()}/saved_models/{i}/xgb/{j}.
↪mdl")

    best_cat = {'learning_rate': 0.05, 'depth': 9}
    cat = PlantTraitRegressor(
        catboost.CatBoostRegressor(
            iterations=2000 if not dry_run else 1,
            embedding_features=["emb"],
            eval_metric="R2",
            early_stopping_rounds=1000,
            use_best_model=True,
            verbose=False,
            **best_cat
        ),
        n_jobs=2 if not dry_run else 1
    )
    print("Cat: ", cat.fit(X_train_extra_features, Y_train,␣
↪eval_set=(X_val_extra_features, Y_val)).score(X_val_extra_features, Y_val))
    os.makedirs(f"{os.getcwd()}/saved_models/{i}/cat/", exist_ok=True)
    for j in range(len(cat.estimators_)):
        cat.estimators_[j].save_model(f"{os.getcwd()}/saved_models/{i}/cat/{j}.
↪mdl")
```

```
    best_lgb = {
        "objective": "regression",
        "metric": "rmse",
        "n_estimators": 1500 if not dry_run else 1,
        "bagging_freq": 1,
        "learning_rate": 0.010144890360462996,
        "num_leaves": 724,
        "subsample": 0.9896282659716074,
        "colsample_bytree": 0.2884524600576782,
        "min_data_in_leaf": 61,
        "verbosity": -1,
    }
    lgb = PlantTraitRegressor(
        lightgbm.LGBMRegressor(
            linear_tree = True,
            **best_lgb
        ),
        n_jobs=2 if not dry_run else 1
    )
    print("Lgb: ", lgb.fit(
        X_train_extra_features[columns_no_embed], Y_train,
        eval_set=(X_val_extra_features[columns_no_embed], Y_val),
        callbacks=[lightgbm.early_stopping(stopping_rounds=100)]
    ).score(X_val_extra_features[columns_no_embed], Y_val))
    os.makedirs(f"{os.getcwd()}/saved_models/{i}/lgb/", exist_ok=True)
    for j in range(len(lgb.estimators_)):
        lgb.estimators_[j].booster_.save_model(f"{os.getcwd()}/saved_models/{i}/
↪lgb/{j}.mdl")

    ridge = Ridge()
    print("Ridge: ", ridge.fit(X_train, Y_train).score(X_val, Y_val))

    reg = KNeighborsRegressor(
        n_neighbors=7 if not dry_run else 1,
        metric="manhattan",
        weights='distance'
    )
    print("Reg: ", reg.fit(X_train, Y_train).score(X_val, Y_val))

    mlp = train_mlp_model(
        X_train=X_train,
        Y_train=Y_train,
        X_val=X_val,
        Y_val=Y_val,
        batch_size=256,
        lr=5e-4,
        dry_run=dry_run,
```

```
        run_num=i,
    )
    print ("MLP: ", mlp.score(X_val, Y_val))

    l_train_X = np.column_stack((
        Y_pipeline.transform(xgb.
↪predict(X_val_extra_features[columns_no_embed])),
        Y_pipeline.transform(cat.predict(X_val_extra_features)),
        Y_pipeline.transform(lgb.
↪predict(X_val_extra_features[columns_no_embed])),
        ridge.predict(X_val),
        reg.predict(X_val),
        mlp.predict(X_val))
    )

    meta = Lasso(alpha=0.00006)
    print (f"Done: {i} with score", meta.fit(l_train_X, Y_val).score(l_train_X,␣
↪Y_val))

    l_test_X = np.column_stack((
        Y_pipeline.transform(xgb.
↪predict(X_test_extra_features[columns_no_embed])),
        Y_pipeline.transform(cat.predict(X_test_extra_features)),
        Y_pipeline.transform(lgb.
↪predict(X_test_extra_features[columns_no_embed])),
        ridge.predict(X_test),
        reg.predict(X_test),
        mlp.predict(X_test))
    )

    if dry_run:
        preds = Y_pipeline.inverse_transform(meta.predict(l_test_X).reshape(-1,␣
↪1))
    else:
        preds = Y_pipeline.inverse_transform(meta.predict(l_test_X))
    preds_test += preds / kf.get_n_splits()

    print ("================\n\n")
```

```
Xgb:  0.4791228771209717
```

```
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/xgboost/core.py:158: UserWarning: [15:57:36] WARNING:
/Users/runner/work/xgboost/xgboost/src/c_api/c_api.cc:1374: Saving model in the
UBJSON format as default.  You can use file extension: `json`, `ubj` or
`deprecated` to choose between formats.
  warnings.warn(smsg, UserWarning)
```

```
Cat:  0.5037602154782522
Training until validation scores don't improve for 100 rounds
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.701749
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.731844
Training until validation scores don't improve for 100 rounds
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.594255
Training until validation scores don't improve for 100 rounds
Early stopping, best iteration is:
[1336]  valid_0's rmse: 0.818875
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.786314
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.693728
Lgb:  0.4785356161693053
Ridge:  0.4067244261518747

GPU available: True (mps), used: True
TPU available: False, using: 0 TPU cores
HPU available: False, using: 0 HPUs
Missing logger folder: saved_models/0/mlp/lightning_logs

  | Name | Type       | Params | Mode
-------------------------------------------
0 | body | Sequential | 2.0 M  | train
-------------------------------------------
2.0 M     Trainable params
0         Non-trainable params
2.0 M     Total params
8.019     Total estimated model params size (MB)

Reg:  0.48507974888286204

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/loggers/tensorboard.py:194: Could not log
computational graph to TensorBoard: The `model.example_input_array` attribute is
not set or `input_array` was not given.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'val_dataloader' to speed up the dataloader
worker initialization.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'train_dataloader' to speed up the
```

```
dataloader worker initialization.
`Trainer.fit` stopped: `max_epochs=5` reached.

MLP:  0.46301111578941345

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(

Done: 0 with score 0.5434211970382934

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(

===============


Xgb:  0.4756641685962677

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/xgboost/core.py:158: UserWarning: [18:59:10] WARNING:
/Users/runner/work/xgboost/xgboost/src/c_api/c_api.cc:1374: Saving model in the
UBJSON format as default.  You can use file extension: `json`, `ubj` or
`deprecated` to choose between formats.
  warnings.warn(smsg, UserWarning)

Cat:  0.5024788636625849
Training until validation scores don't improve for 100 rounds
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1499]  valid_0's rmse: 0.722972
Training until validation scores don't improve for 100 rounds
```

```
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.697954
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1488]  valid_0's rmse: 0.596224
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1492]  valid_0's rmse: 0.801484
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.778007
Did not meet early stopping. Best iteration is:
[1494]  valid_0's rmse: 0.682096
Lgb:  0.4744044489803018
Ridge:  0.405946217035337

GPU available: True (mps), used: True
TPU available: False, using: 0 TPU cores
HPU available: False, using: 0 HPUs
Missing logger folder: saved_models/1/mlp/lightning_logs

  | Name | Type       | Params | Mode
-------------------------------------------
0 | body | Sequential | 2.0 M  | train
-------------------------------------------
2.0 M     Trainable params
0         Non-trainable params
2.0 M     Total params
8.019     Total estimated model params size (MB)

Reg:  0.4911360464170051

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/loggers/tensorboard.py:194: Could not log
computational graph to TensorBoard: The `model.example_input_array` attribute is
not set or `input_array` was not given.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'val_dataloader' to speed up the dataloader
worker initialization.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'train_dataloader' to speed up the
dataloader worker initialization.

MLP:  0.46521177887916565

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
```

```
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(

Done: 1 with score 0.5391624153654578

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(

===============


Xgb:  0.466921329498291

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/xgboost/core.py:158: UserWarning: [22:17:03] WARNING:
/Users/runner/work/xgboost/xgboost/src/c_api/c_api.cc:1374: Saving model in the
UBJSON format as default.  You can use file extension: `json`, `ubj` or
`deprecated` to choose between formats.
  warnings.warn(smsg, UserWarning)

Cat:  0.4984494012624445
Training until validation scores don't improve for 100 rounds
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1499]  valid_0's rmse: 0.704489
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.739364
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.761805
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
```

```
[1499]  valid_0's rmse: 0.619189
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.77195
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.690084
Lgb:  0.46916977204105237
Ridge:  0.39634323076380795

GPU available: True (mps), used: True
TPU available: False, using: 0 TPU cores
HPU available: False, using: 0 HPUs
Missing logger folder: saved_models/2/mlp/lightning_logs

  | Name | Type       | Params | Mode
-------------------------------------------
0 | body | Sequential | 2.0 M  | train
-------------------------------------------
2.0 M     Trainable params
0         Non-trainable params
2.0 M     Total params
8.019     Total estimated model params size (MB)

Reg:  0.48011064097364103

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/loggers/tensorboard.py:194: Could not log
computational graph to TensorBoard: The `model.example_input_array` attribute is
not set or `input_array` was not given.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'val_dataloader' to speed up the dataloader
worker initialization.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'train_dataloader' to speed up the
dataloader worker initialization.

MLP:  0.44507554173469543

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
```

```
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/linear_model/_coordinate_descent.py:697: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations,
check the scale of the features or consider increasing regularisation. Duality
gap: 1.474e+01, tolerance: 3.839e-01
  model = cd_fast.enet_coordinate_descent(

Done: 2 with score 0.5300080843996272

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(

================


Xgb:  0.4625978469848633

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/xgboost/core.py:158: UserWarning: [01:29:10] WARNING:
/Users/runner/work/xgboost/xgboost/src/c_api/c_api.cc:1374: Saving model in the
UBJSON format as default.  You can use file extension: `json`, `ubj` or
`deprecated` to choose between formats.
  warnings.warn(smsg, UserWarning)

Cat:  0.4884226635379461
Training until validation scores don't improve for 100 rounds
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1499]  valid_0's rmse: 0.692786
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.722833
Training until validation scores don't improve for 100 rounds
Training until validation scores don't improve for 100 rounds
Early stopping, best iteration is:
[1281]  valid_0's rmse: 0.813829
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.605392
```

```
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.774824
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.702162
Lgb:  0.46225274970394903
Ridge:  0.39188798408729597
```

```
GPU available: True (mps), used: True
TPU available: False, using: 0 TPU cores
HPU available: False, using: 0 HPUs
Missing logger folder: saved_models/3/mlp/lightning_logs
```

```
  | Name | Type       | Params | Mode
-------------------------------------------
0 | body | Sequential | 2.0 M  | train
-------------------------------------------
2.0 M     Trainable params
0         Non-trainable params
2.0 M     Total params
8.019     Total estimated model params size (MB)
```

```
Reg:  0.47525342427510764
```

```
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/loggers/tensorboard.py:194: Could not log
computational graph to TensorBoard: The `model.example_input_array` attribute is
not set or `input_array` was not given.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'val_dataloader' to speed up the dataloader
worker initialization.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'train_dataloader' to speed up the
dataloader worker initialization.
```

```
MLP:  0.4389471113681793
```

```
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
```

```
  warnings.warn(

Done: 3 with score 0.5226707581591652

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(

================


Xgb:  0.4851963222026825

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/xgboost/core.py:158: UserWarning: [04:28:03] WARNING:
/Users/runner/work/xgboost/xgboost/src/c_api/c_api.cc:1374: Saving model in the
UBJSON format as default.  You can use file extension: `json`, `ubj` or
`deprecated` to choose between formats.
  warnings.warn(smsg, UserWarning)

Cat:  0.5071858612432029
Training until validation scores don't improve for 100 rounds
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.697047
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.700483
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.594073
Training until validation scores don't improve for 100 rounds
Early stopping, best iteration is:
[1269]  valid_0's rmse: 0.785954
Training until validation scores don't improve for 100 rounds
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.760828
Did not meet early stopping. Best iteration is:
[1500]  valid_0's rmse: 0.685345

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/joblib/externals/loky/process_executor.py:752: UserWarning: A worker
```

```
stopped while some jobs were given to the executor. This can be caused by a too
short worker timeout or by a memory leak.
  warnings.warn(

Lgb:  0.48737798120138826
Ridge:  0.41208815518543895

GPU available: True (mps), used: True
TPU available: False, using: 0 TPU cores
HPU available: False, using: 0 HPUs
Missing logger folder: saved_models/4/mlp/lightning_logs

  | Name | Type       | Params | Mode
-------------------------------------------
0 | body | Sequential | 2.0 M  | train
-------------------------------------------
2.0 M     Trainable params
0         Non-trainable params
2.0 M     Total params
8.019     Total estimated model params size (MB)

Reg:  0.4922613173423169

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/loggers/tensorboard.py:194: Could not log
computational graph to TensorBoard: The `model.example_input_array` attribute is
not set or `input_array` was not given.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'val_dataloader' to speed up the dataloader
worker initialization.
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/lightning/pytorch/trainer/connectors/data_connector.py:419: Consider
setting `persistent_workers=True` in 'train_dataloader' to speed up the
dataloader worker initialization.
`Trainer.fit` stopped: `max_epochs=5` reached.

MLP:  0.4572415351867676

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/joblib/externals/loky/process_executor.py:752: UserWarning: A worker
stopped while some jobs were given to the executor. This can be caused by a too
short worker timeout or by a memory leak.
```

```
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(

Done: 4 with score 0.5459703325317419

/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/joblib/externals/loky/process_executor.py:752: UserWarning: A worker
stopped while some jobs were given to the executor. This can be caused by a too
short worker timeout or by a memory leak.
  warnings.warn(
/Users/sriharivishnu/mambaforge/envs/cs480/lib/python3.12/site-
packages/sklearn/base.py:493: UserWarning: X does not have valid feature names,
but StandardScaler was fitted with feature names
  warnings.warn(

================
```

[ ]:
```
Y_pipeline.inverse_transform(meta.predict(l_test_X).reshape(-1,1))
```

[ ]:
```
array([[1.10234921],
       [0.99312656],
       [1.03718978],
       …,
       [1.11389608],
       [1.18449267],
       [1.05394351]])
```

[ ]:
```
preds_test.shape
```

[ ]:
```
(6391, 6)
```

[ ]:
```
r2_score(Y_pipeline.inverse_transform(reg.predict(X_test)), preds_test)
```

[ ]:
```
0.907657301423881
```

[ ]:
```
import csv
with open('submission.csv', 'w', newline='') as csvfile:
    writer = csv.writer(csvfile)
    writer.writerow(['id', 'X4', 'X11', 'X18', 'X26', 'X50', 'X3112'])
    for i in range(len(preds_test)):
      writer.writerow([test_data.plant.ids[i], preds_test[i][0],␣
 ↪preds_test[i][1], preds_test[i][2], preds_test[i][3], preds_test[i][4],␣
 ↪preds_test[i][5]])
```