ml-fuel

Release 0.2

Wikilimo Ltd., London, UK

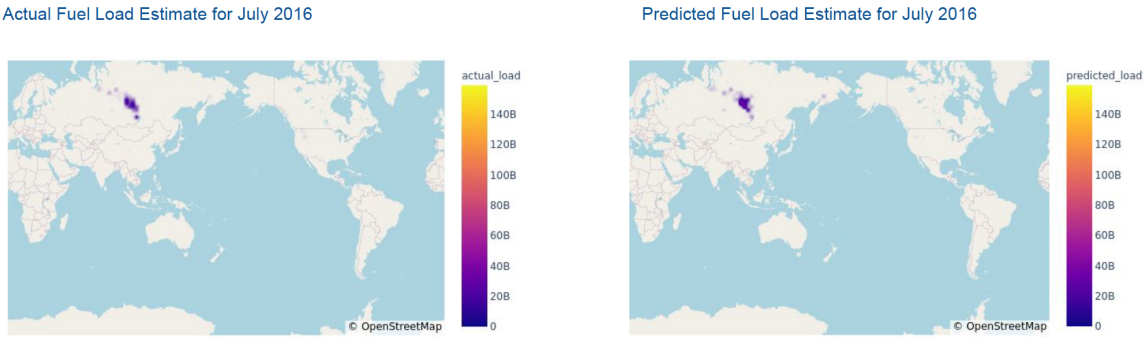
Jun 21, 2021

CONTENTS:

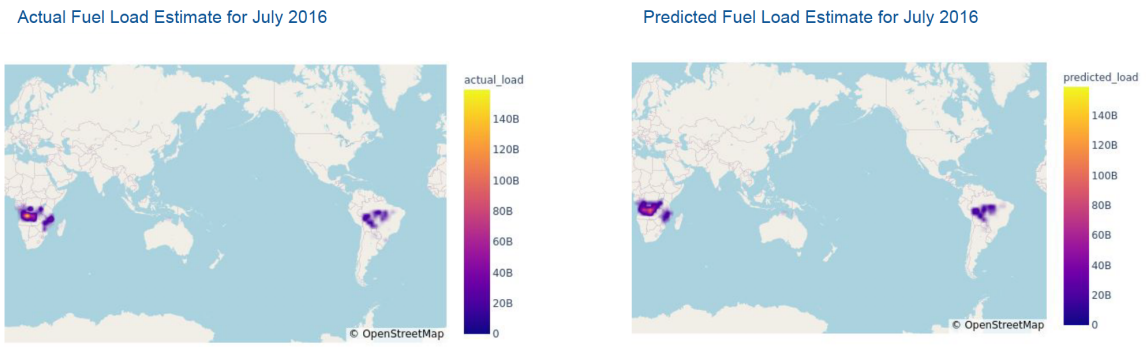
We use earth observation data with Machine Learning to predict Fuel Load, which embodies the content of burnable vegetation in an area.

Two models are developed, for the Mid-Latitudes and the Tropics using Gradient Boosted Decision Tree style Machine Learning methods. We can see below a visual comparison of the predictions made by the model and corresponding ground truth.

- CatBoost for Mid-Latitudes



- LightGBM for Tropics



We recommend you go through the Getting Started section to first set up your development environment. Once you have the input data (as `.nc` NetCDF files) and the dependencies installed, you can either refer to the example notebooks in `notebooks/` or go through the further notes on preprocessing, training and testing. More details can be found in the Module API docs in the index below.

CHAPTER

ONE

GETTING STARTED

1.1 Installation

The python environment for the repository can be created using **either** `conda` or `virtualenv`, by running from the root of the repo:

1.1.1 Using conda

```
conda create --name=ml-fuel python=3.8
conda activate ml-fuel
```

1.1.2 Using virtualenv

```
python3 -m venv env
source env/bin/activate
```

1.1.3 Install dependencies

```
pip install -U pip
pip install -r requirements.txt
```

This includes all the packages required for running the code in the repository.

1.2 Pre-trained models

Pre-trained models are available:

- `LightGBM.joblib` at `src/pre-trained_models/LightGBM.joblib`
- `CatBoost.joblib` at `src/pre-trained_models/CatBoost.joblib`

1.3 Demo Notebooks

Notebooks for training and inference:

- `LightGBM_training.ipynb` at `notebooks/LightGBM_training.ipynb`
- `LightGBM_inference.ipynb` at `notebooks/LightGBM_inference.ipynb`
- `CatBoost_training.ipynb` at `notebooks/CatBoost_training.ipynb`
- `CatBoost_inference.ipynb` at `notebooks/CatBoost_inference.ipynb`

CHAPTER

TWO

DATA DESCRIPTION

7 years of global historical data, from 2010 - 2016 will be used for developing the machine learning models. All data used in this project is propietary and NOT meant for public release. Xarray and netCDF libraries are used for working with the multi-dimensional geospatial data.

- **Datasets:**
  - Above Ground Biomass
  - Weather Anomalies
  - Climatic Regions
  - Fire Sensitivity Anomalies
  - Slope
  - Fraction of Burnable Area
  - Burned Area
  - Standardized Precipitation Index GPCC
  - Leaf Area Index
- **Resolution:**
  - Latitude: 0.25
  - Longitude: 0.25
  - Time: 1 file per month, ie. 84 timesteps from 2010-16.

The data split into training, testing and validation is currently: - Training: 2010 -> 2015 - Validation: January 2016 -> June 2016 - Testing: July 2016 -> December 2016.

To change the split, modify *data_split()* in *src/utils/generate_io_arrays.py*, and the month list in *src/test.py* used during inference.

CHAPTER

THREE

PRE-PROCESSING

Raw data should first be processed using notebooks in `notebooks/preprocess/*`. Entry point for the pre-processing script for the ML pipeline is `src/pre-processing.py`.

Args description: *`--data_path`: Path to the data files

- Input: Enter the root directory of the xarray data files as the script argument. All data files produced are stored in this directory.
  - `src/utils/data_paths.py` - defines the files paths for the features used in training and also the paths of the `fuel_load.nc` which will be created.
- Output:
  - Creates `fuel_load.nc` file for Fuel Load Data (Burned Area * Above Ground Biomass).
  - Saves the following files for the Tropics & Mid-Latitudes regions respectively, where {type} is 'tropics' or 'midlats'.
  - Save Directory `root_path/{type}`

```
- {type}_train.csv
- {type}_val.csv
- {type}_test.csv
```

  - Save Directory `root_path/infer_{type}`

```
- {type}*infers*\ July.csv
- {type}*infers*\ Aug.csv
- {type}*infers*\ Sept.csv
- {type}*infers*\ Oct.csv
- {type}*infers*\ Nov.csv
- {type}*infers*\ Dec.csv
```

Where root_path is the root save path provided for `pre-processing.py`

CHAPTER

FOUR

TRAINING

Entry-point for train `src/train.py`

Arguments description:

- `--model_name`: Name of the model to be trained (`CatBoost` or `LightGBM`).
- `--data_path`: Data directory where all the input (train, val, test) `.csv` files are stored.
- `--exp_name`: Name of the training experiment used for logging.

CHAPTER

FIVE

TRAINING CATBOOST (MID-LATITUDES)

This notebooks demonstrates training a CatBoost model with hyperparameter optimization, followed by feature importance visualization using SHAP. CatBoost is a machine learning algorithm that uses gradient boosting on decision trees. This notebook utilizes the `deepfuel-ML/src/models/catboost_module.py` script for model training.

```
import os
import pandas as pd
import numpy as np
from joblib import dump, load
import shap
```

5.1 Data directory

```
# The training, validation and test set required for model training are placed in␣
↪data/midlats/
! tree ../data/midlats
```

```
[01;34m../data/midlats[00m
├── midlats_test.csv
├── midlats_train.csv
└── midlats_val.csv

0 directories, 3 files
```

5.2 Input Features

- Latitude
- Longitude
- Leaf Area Index
- Fire Weather Index: fwinx
- Drought Code: drtcode
- Fire Danegr Severity Rating: fdsrte
- Fraction of Burnable Area: fraction_of_burnable_area
- d2m

- Evaporation Rate: erate
- fg10
- si10
- Volumetric Soil Water Level 1: swvl1
- 2m Temperature: t2m
- tprate
- Climatic Region: climatic_region
- Slope: slor
- Month: month
- Fuel Load: actual_load (target variable)

```
# Check header of training set matches input features
! head -n 1 ../data/midlats/midlats_train.csv
```

```
latitude,longitude,LAI,fwinx,drtcode,fdsrte,fraction_of_burnable_area,d2m,erate,fg10,
↪si10,swvl1,t2m,tprate,climatic_region,slor,actual_load,month
```

5.3 Model Training

```
!python '../src/train.py'  --model_name 'CatBoost' --data_path '../data/midlats/' --
↪exp_name 'CatBoost_exp'
```

```
Link for the created Neptune experiment--------
Info (NVML): NVML Shared Library Not Found. GPU usage metrics may not be reported.␣
↪For more information, see https://docs.neptune.ai/logging-and-managing-experiment-
↪results/logging-experiment-data.html#hardware-consumption
https://ui.neptune.ai/shared/step-by-step-monitoring-experiments-live/e/STEP-163
-----------------------------------
0:  learn: 0.9193915        test: 0.9374665 best: 0.9374665 (0)     total: 78.9ms  ␣
↪remaining: 1m 18s
1:  learn: 0.8572337        test: 0.8807920 best: 0.8807920 (1)     total: 90.4ms  ␣
↪remaining: 45.1s
2:  learn: 0.8133704        test: 0.8472102 best: 0.8472102 (2)     total: 105ms   ␣
↪remaining: 34.9s
3:  learn: 0.7751241        test: 0.8134123 best: 0.8134123 (3)     total: 119ms   ␣
↪remaining: 29.6s
4:  learn: 0.7455154        test: 0.7849642 best: 0.7849642 (4)     total: 134ms   ␣
↪remaining: 26.7s
5:  learn: 0.7227938        test: 0.7619387 best: 0.7619387 (5)     total: 147ms   ␣
↪remaining: 24.3s
. . .
315:        learn: 0.4626048        test: 0.6321089 best: 0.6305450 (299)   total: 4.
↪34s    remaining: 9.4s
316:        learn: 0.4623857        test: 0.6320927 best: 0.6305450 (299)   total: 4.
↪36s    remaining: 9.39s
317:        learn: 0.4622255        test: 0.6321369 best: 0.6305450 (299)   total: 4.
↪37s    remaining: 9.37s
318:        learn: 0.4618840        test: 0.6320915 best: 0.6305450 (299)   total: 4.
↪38s    remaining: 9.36s
```

(continues on next page)

(continued from previous page)

```
319:        learn: 0.4616440        test: 0.6317562 best: 0.6305450 (299)   total: 4.
↪4s     remaining: 9.35s
Stopped by overfitting detector  (20 iterations wait)

bestTest = 0.6305450201
bestIteration = 299

Shrink model to first 300 iterations.
RMSE  : 0.6305450197747737
----------------------------------------------------------------
Inference results

Training error:  2039187852.5081983
Validation error:  2854273450.7074313
Test error:  2231005975.951971
Model file save at ['/Users/rbiswas/VSCodeProjects/deepfuel-ML/src/results/pre-
↪trained_models/CatBoost.joblib']
```

The training logs can be viewed live online at the following link: https://ui.neptune.ai/shared/step-by-step-monitoring-experiments-live/e/STEP-158

5.4 Loading the trained model

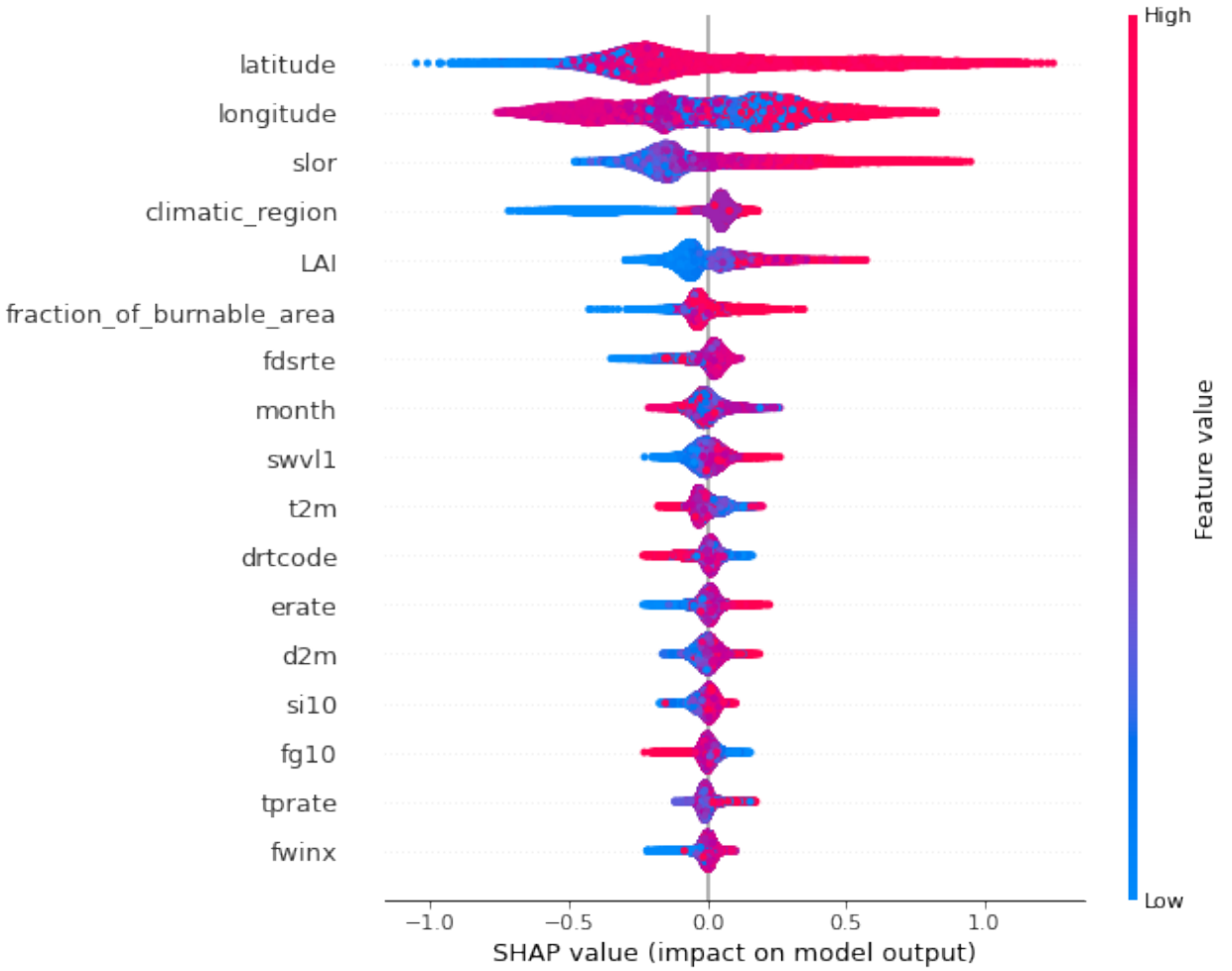
```
model = load('../src/results/pre-trained_models/CatBoost.joblib')
```

5.5 Feature importance using SHAP

SHAP (SHapley Additive exPlanations) is used to explain the output of the trained machine learning model.

```
midlat_train = pd.read_csv('../data/midlats/midlats_train.csv')
```

```
shap_values = shap.TreeExplainer(model).shap_values(midlat_train.drop([ 'actual_load
↪'], axis=1))
shap.summary_plot(shap_values, midlat_train.drop(['actual_load'], axis=1))
```



The y-axis indicates the variable name, in order of importance from top to bottom. On the x-axis (Impact on model output), the horizontal location shows whether the effect of that value is associated with a higher or lower prediction. Gradient colour indicates feature value.

CHAPTER

SIX

TRAINING LIGHTGBM (TROPICS)

This notebooks demonstrates training a LightGBM model with hyperparameter optimization, followed by feature importance visualization using SHAP. LightGBM is a gradient boosting framework that uses tree based learning algorithms. This notebook utilizes the `deepfuel-ML/src/models/lightgbm_module.py` script for model training.

```
import os
import pandas as pd
import numpy as np
from joblib import dump, load
import shap
```

6.1 Data directory

```
# The training, validation and test set required for model training are placed in␣
↪data/tropics/
! tree ../data/tropics
```

```
[01;34m../data/tropics[00m
├── tropics_test.csv
├── tropics_train.csv
└── tropics_val.csv

0 directories, 3 files
```

6.2 Input Features

- Latitude
- Longitude
- Fire Weather Index: fwinx
- Drought Code: drtcode
- Fire Danegr Severity Rating: fdsrte
- Fraction of Burnable Area: fraction_of_burnable_area
- d2m
- Evaporation Rate: erate

- fg10
- si10
- Volumetric Soil Water Level 1: swvl1
- 2m Temperature: t2m
- tprate
- Climatic Region: climatic_region
- Slope: slor
- Month: month
- Fuel Load: actual_load (target variable)

```
# Check header of training set matches input features
! head -n 1 ../data/tropics/tropics_train.csv
```

```
latitude,longitude,fwinx,drtcode,fdsrte,fraction_of_burnable_area,d2m,erate,fg10,si10,
↪swvl1,t2m,tprate,climatic_region,slor,actual_load,month
```

6.3 Model Training

```
!python  '../src/train.py'  --model_name 'LightGBM' --data_path '../data/tropics'  --
↪exp_name 'LightGBM_exp'
```

```
Link for the created Neptune experiment--------
Info (NVML): NVML Shared Library Not Found. GPU usage metrics may not be␣
↪reported. For more information, see https://docs.neptune.ai/logging\unhbox\
voidb@x\kern\z@\char'\protect\discretionary{\char\defaulthyphenchar}{}{}and\
unhbox\voidb@x\kern\z@\char'\protect\discretionary{\char\
defaulthyphenchar}{}{}managing\unhbox\voidb@x\kern\z@\char'\protect\
discretionary{\char\defaulthyphenchar}{}{}experiment\unhbox\voidb@x\
kern\z@\char'\protect\discretionary{\char\defaulthyphenchar}{}{}results/
logging\unhbox\voidb@x\kern\z@\char'\protect\discretionary{\char\
defaulthyphenchar}{}{}experiment\unhbox\voidb@x\kern\z@\char'\protect\
discretionary{\char\defaulthyphenchar}{}{}data.html#hardware\unhbox\voidb@
x\kern\z@\char'\protect\discretionary{\char\defaulthyphenchar}{}{}consumption
https://ui.neptune.ai/shared/step\unhbox\voidb@x\kern\z@\char'\protect\
discretionary{\char\defaulthyphenchar}{}{}by\unhbox\voidb@x\kern\z@\char'\
protect\discretionary{\char\defaulthyphenchar}{}{}step\unhbox\voidb@x\kern\
z@\char'\protect\discretionary{\char\defaulthyphenchar}{}{}monitoring\
unhbox\voidb@x\kern\z@\char'\protect\discretionary{\char\
defaulthyphenchar}{}{}experiments\unhbox\voidb@x\kern\z@\char'\protect\
discretionary{\char\defaulthyphenchar}{}{}live/e/STEP\unhbox\voidb@x\kern\
z@\char'\protect\discretionary{\char\defaulthyphenchar}{}{}168
-------------------------------------
/Users/rbiswas/miniconda3/envs/smos-fuel/lib/python3.8/site-packages/lightgbm/
↪engine.py:156: UserWarning: Found early_stopping_rounds in params. Will use␣
↪it instead of argument
  warnings.warn("Found {} in params. Will use it instead of argument".
↪format(alias))
[LightGBM] [Warning] Unknown parameter: lamda_l2
```

```
[LightGBM] [Warning] Unknown parameter: lamda_l1
[LightGBM] [Warning] Unknown parameter: lamda_l2
[LightGBM] [Warning] Unknown parameter: lamda_l1
[LightGBM] [Warning] Auto-choosing row-wise multi-threading, the overhead of␣
↪testing was 0.005887 seconds.
You can set force_row_wise=true to remove the overhead.
And if memory is not enough, you can set force_col_wise=true.
[LightGBM] [Info] Total Bins 3280
[LightGBM] [Info] Number of data points in the train set: 295013, number of␣
↪used features: 16
[LightGBM] [Warning] Unknown parameter: lamda_l2
[LightGBM] [Warning] Unknown parameter: lamda_l1
[LightGBM] [Info] Start training from score 1833509820.058491
[1] train's rmse: 3.70593e+09       val's rmse: 4.97223e+09
Training until validation scores don't improve for 20 rounds
[2] train's rmse: 3.69441e+09       val's rmse: 4.96466e+09
[3] train's rmse: 3.68432e+09       val's rmse: 4.96094e+09
[4] train's rmse: 3.67147e+09       val's rmse: 4.94791e+09
[5] train's rmse: 3.65895e+09       val's rmse: 4.93876e+09
[6] train's rmse: 3.65069e+09       val's rmse: 4.93606e+09
[7] train's rmse: 3.63764e+09       val's rmse: 4.92544e+09
[8] train's rmse: 3.62504e+09       val's rmse: 4.9169e+09
[9] train's rmse: 3.61277e+09       val's rmse: 4.90962e+09
[10]        train's rmse: 3.60155e+09       val's rmse: 4.90066e+09
[11]        train's rmse: 3.592e+09 val's rmse: 4.89457e+09
[12]        train's rmse: 3.58089e+09       val's rmse: 4.88582e+09
. . .
[1246]      train's rmse: 1.60812e+09       val's rmse: 4.01095e+09
[1247]      train's rmse: 1.60776e+09       val's rmse: 4.01099e+09
[1248]      train's rmse: 1.60739e+09       val's rmse: 4.01119e+09
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[1249]      train's rmse: 1.60726e+09       val's rmse: 4.01119e+09
[1250]      train's rmse: 1.60695e+09       val's rmse: 4.01122e+09
Early stopping, best iteration is:
[1230]      train's rmse: 1.6122e+09        val's rmse: 4.01059e+09
RMSE   : 4010589083.0953164
----------------------------------------------------------------
Inference results

Training error:  1612198733.154914
Validation error:  4010589083.0953164
Test error:  2508979611.028168
Model file save at ['/Users/rbiswas/VSCodeProjects/deepfuel-ML/src/results/
↪pre-trained_models/LightGBM.joblib']
```

Training logs can be viewed live at the following link: https://ui.neptune.ai/shared/step-by-step-monitoring-experiments-live/e/STEP-165

6.4 Loading the trained model

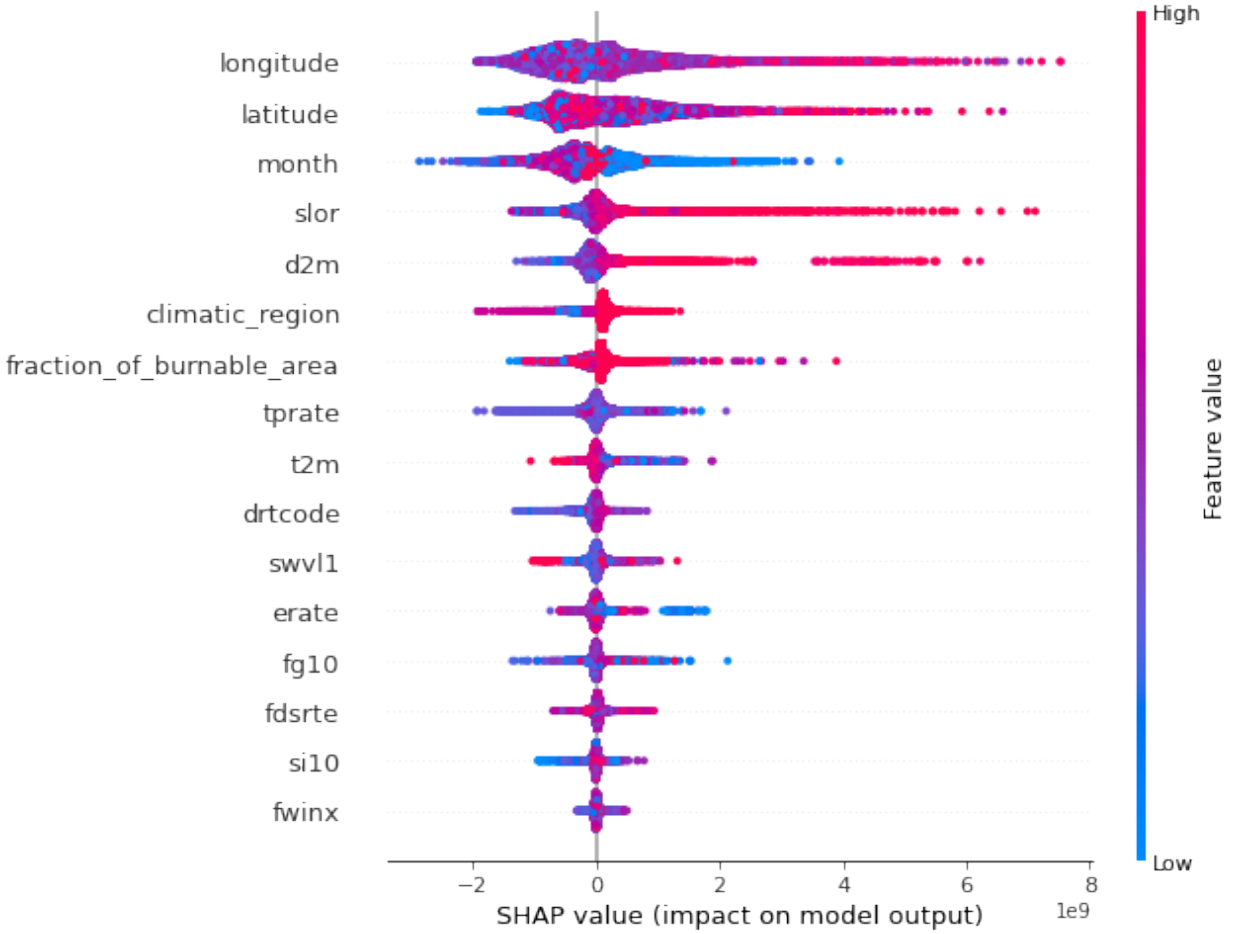
```
model = load('../src/results/pre-trained_models/LightGBM.joblib')
```

6.5 Feature importance using SHAP

SHAP (SHapley Additive exPlanations) is a game theoretic approach to explain the output of any machine learning model. It connects optimal credit allocation with local explanations using the classic Shapley values from game theory and their related extensions

```
tropic_val = pd.read_csv('../data/tropics/tropics_val.csv')
```

```
shap_values = shap.Explainer(model).shap_values(tropic_val.drop(['actual_load'],
↪axis=1))
shap.summary_plot(shap_values, tropic_val.drop(['actual_load'],axis=1))
```



The y-axis indicates the variable name, in order of importance from top to bottom. On the x-axis (Impact on model output), the horizontal location shows whether the effect of that value is associated with a higher or lower prediction. Gradient colour indicates feature value.

CHAPTER

SEVEN

INFERENCE

Entry-point for inference is `src/test.py`

Arguments description:

- `--model_name`: Name of the model to be trained (`CatBoost` or `LightGBM`).
- `--model_path`: Path to the pre-trained `.joblib` model.
- `--data_path`: Valid data directory where all the test `.csv` files are stored.
- `--results_path`: Directory where the result inference `.csv` files and `.html` visualizations are going to be stored.

CHAPTER

EIGHT

CATBOOST INFERENCE

This notebooks demonstrates generating inferences from a pretrained CatBoost model. This notebook utilizes the `deepfuel-ML/src/test.py` script for generating inferences. The script does everything from calculating error values to plotting data for visual inference.

```
import os
import pandas as pd
import numpy as np
from joblib import dump, load
from IPython.display import Image, display, HTML
```

8.1 Using `test.py`

Below is the description of its arguements: - `--model_name`: Name of the model to be trained ("CatBoost" or "LightGBM"). - `--model_path`: Path to the pre-trained model. - `--data_path`: Valid data directory where all the test .csv files are stored. - `--results_path`: Directory where the result inference .csv files and .png visualizations are going to be stored.

8.2 With Ground Truth (`actual_load` is present in the test csv)

```
!python '../src/test.py'  --model_name 'CatBoost' --model_path '../src/pre-trained_
↪models/CatBoost.joblib' --data_path '../data/infer_midlats'  --results_path '../
↪data/midlats/results'
```

```
MAPE July : 380.44795759521344
MAPE Aug : 283.7487728040964
MAPE Sept : 203.97476414457114
MAPE Oct : 117.19251658203949
MAPE Nov : 105.94428641567805
MAPE Dec : 99.29645055040669
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Nov_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Nov_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪July_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_July_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Dec_actual.html
```

(continues on next page)

(continued from previous page)

```
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Dec_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Aug_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Aug_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Oct_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Oct_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Sept_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Sept_predicted.html
```

8.3 Inference CSV

`test.py` generates `.csv` files for each month with the following columns: - `latitude` - `longitude` - `actual_load` - Actual Fuel Load value - `predicted_load` - Predicted Fuel Load value - `APE` - Average Percentage Error between actual and predicted fuel load values

```
df=pd.read_csv('../data/midlats/results/midlats_output_July.csv')
df.head()
```

8.4 Without Ground Truth (`actual_load` is not present in the test csv)

```
!python '../src/test.py'  --model_name 'CatBoost' --model_path '../src/pre-trained_
↪models/CatBoost.joblib' --data_path '../data/infer_midlats'  --results_path '../
↪data/midlats/results'
```

```
MAPE July : 380.44795759521344
MAPE Aug : 283.7487728040964
MAPE Sept : 203.97476414457114
MAPE Oct : 117.19251658203949
MAPE Nov : 105.94428641567805
MAPE Dec : 99.29645055040669
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Nov_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Nov_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪July_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_July_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Dec_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Dec_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Aug_actual.html
```

(continues on next page)

(continued from previous page)

```
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Aug_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Oct_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Oct_predicted.html
Actual FL plot successfully generated! File saved to  ../data/midlats/results/midlats_
↪Sept_actual.html
Predicted FL plot successfully generated! File saved to  ../data/midlats/results/
↪midlats_Sept_predicted.html
```

8.5 Inference CSV

```
df=pd.read_csv('../data/midlats/results/midlats_output_July.csv')
df.head()
```

8.6 Visualizing the plots generated

The plots are stored as html files that can be zoomed in upto the resolution of the data to view the predicted and actual values

CHAPTER

NINE

LIGHTGBM INFERENCE

This notebooks demonstrates generating inferences from a pretrained LightGBM model. This notebook utilizes the `deepfuel-ML/src/test.py` script for generating inferences. The script does everything from calculating error values to plotting data for visual inference.

```
import os
import pandas as pd
import numpy as np
from joblib import dump, load
import sys
import os
from IPython.display import Image, display
```

9.1 Using `test.py`

Below is the description of its arguements: - `--model_name`: Name of the model to be trained ("CatBoost" or "LightGBM"). - `--model_path`: Path to the pre-trained model. - `--data_path`: Valid data directory where all the test .csv files are stored. - `--results_path`: Directory where the result inference .csv files and .png visualizations are going to be stored.

9.2 With Ground Truth (`actual_load` is present in the test csv)

```
!python '../src/test.py'  --model_name 'LightGBM' --model_path '../src/pre-trained_
↪models/LightGBM.joblib' --data_path '../data/infer_tropics'  --results_path '../
↪data/tropics/results'
```

```
MAPE July : 358.2370533961142
MAPE Aug : 4068.041474465497
MAPE Sept : 342.60497263841376
MAPE Oct : 407.02247341732897
MAPE Nov : 553.79772310129
MAPE Dec : 433.6634326468742
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Nov_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Nov_predicted.html
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Aug_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Aug_predicted.html
```

(continues on next page)

(continued from previous page)

```
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Dec_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Dec_predicted.html
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Oct_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Oct_predicted.html
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪July_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_July_predicted.html
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Sept_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Sept_predicted.html
```

9.3 Inference CSV

`test.py` generates `.csv` files for each month with the following columns: - `latitude` - `longitude` - `actual_load` - Actual Fuel Load value - `predicted_load` - Predicted Fuel Load value - `APE` - Average Percentage Error between actual and predicted fuel load values

```
df=pd.read_csv('../data/tropics/results/tropics_output_July.csv')
df.head()
```

9.4 Without Ground Truth (`actual_load` is not present in the test csv)

```
!python '../src/test.py'  --model_name 'LightGBM' --model_path '../src/pre-trained_
↪models/LightGBM.joblib' --data_path '../data/infer_tropics'  --results_path '../
↪data/tropics/results'
```

```
MAPE July : 358.2370533961142
MAPE Aug : 4068.041474465497
MAPE Sept : 342.60497263841376
MAPE Oct : 407.02247341732897
MAPE Nov : 553.79772310129
MAPE Dec : 433.6634326468742
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Nov_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Nov_predicted.html
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Aug_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Aug_predicted.html
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Dec_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Dec_predicted.html
```

(continues on next page)

(continued from previous page)

```
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Oct_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Oct_predicted.html
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪July_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_July_predicted.html
Actual FL plot successfully generated! File saved to  ../data/tropics/results/tropics_
↪Sept_actual.html
Predicted FL plot successfully generated! File saved to  ../data/tropics/results/
↪tropics_Sept_predicted.html
```

9.5 Inference CSV

```
df=pd.read_csv('../data/tropics/results/tropics_output_July.csv')
df.head()
```

9.6 Visualizing the plots generated

The plots are stored as html files that can be zoomed in upto the resolution of the data to view the predicted and actual values.

CHAPTER

TEN

ADDING NEW FEATURES

- Make sure the new dataset to be added is a single file in `.nc` format, containing data from 2010-16 and in 0.25x0.25 grid cell resolution.
- Match the features of the new dataset with the existing features. This can be done by going through `notebooks/EDA_pre-processed_data.ipynb`.
- Add the feature path as a variable to `src/utils/data_paths.py`. Further the path variable is needed to be added to either the time dependant or independant list (depending on which category it belongs to) present inside `export_feature_paths()`.
- The model will now also be trained on the added feature while running src/train.py!

CHAPTER

ELEVEN

TRAINING AND INFERENCE USING DOCKER

This guide assumes you have `Docker` and `docker-compose` installed and setup to run as non-root user following the instructions here, here and here.

11.1 Steps

- Clone the repository.
- Download the data and place it in a `data/` directory at the root of the repository.
- Navigate to the `docker/` directory.
- Run `export UID=$(id -u)` and then `export GID=$(id -g)`.
- Run `docker-compose up --build` which will build the image, run a container and launch a Jupyter server on port `4242`.
- Use the link in the Jupyter command output to access any of the several notebooks for EDA, Training, Inference and Error Analysis.
- If you would like to run the CLI interface, use `docker-compose run ml-fuel bash` to launch an interactive terminal.
- You can now run `pre-processing.py`, `train.py` or `test.py` located in the `src/` directory. Check the docs for more details.

The steps above mount the local code repository and data directory to a volume on the container, setting up the correct permissions so that you can keep any pretrained models or inference files even after the container is shut down.

CHAPTER

TWELVE

INDICES AND TABLES

- genindex
- modindex
- search

Note: This repository was developed by Anurag Saha Roy (@lazyoracle) and Roshni Biswas (@roshni-b) for the ESA-SMOS-2020 project. Contact email: *info@wikilimo.co*. The repository is now maintained by the Wildfire Danger Forecasting team at the European Centre for Medium-range Weather Forecast.