Fit, score and evaluate independent boosted trees

In this notebook 8 of the below mentioned models are fitted with `GradientBoostingregressor()` to predict the harvest yield. The model scores are computed and a plot for each model showing the feature importance and the partial dependence is made.

The models we fit have the following features over the growth season:

1. `NDVI`
2. `NDRE`
3. `MSAVI2`
4. `NDVI`, `DTM`, `N`, and `E`
5. `NDRE` , `DTM`, `N`, and `E`
6. `MSAVI2`, `DTM`, `N`, and `E`
7. `NDVI`, `NDRE`, and `MSAVI2`
8. `NDVI`, `NDRE`, `MSAVI2`, `DTM`, `N`, and `E`

The correlation factors, R^2, that have a value around 0.5 indicate that the models are well described by the features. The highest test score, $R^2 = 0.625814$, is for the *8th* model trained on `NDVI`, `NDRE`, `MSAVI2`, `DTM`, `N`, and `E`. The second highest $R^2 = 0.612129$ is for the *fifth* model that is trained on the features `NDRE` , `DTM`, `N`, and `E`. The lowest test score, $R^2 = 0.477224$, is for the *3rd* model trained only on `MSAVI2`. That means if we follow model 8 then we can predict the harvest yield with a mean absolute error $MAE = 1298.919713 kg/ha$ and model 5 gives $MAE = 1334.772035 kg/ha$. The highest mean absolute error is also from the least precise model 3 with $MAE = 1602.946569 kg/ha$.

The feature importance graph for model 8 shows that `N`, and `E` were the features that contributed most to the model. The next features that contributed to model 8 are the vegetations indexes `NDVI`, `NDRE`, and `MSAVI2` features from the beginning of the growth season. Furthermore the field `DTM` was also one of the most important features. Generally, it seems that the features from the beginning and the end of the growth season are the ones that contribute most to the harvest yield prediction models.

The partial dependence graphs for model 8 shows that the most important features `N` and `E` the harvest yield is descending for `N` > 0.4 and ascending for `E` > 0.4. We can however not conclude that the coordinates are the most contributing to the model because we have data from few coordinates.

We see on the partial dependency graphs that it seems like there's a growing linear dependency between the harvest yield and, respectively, the NDVI feature on July 27 for model 4, NDVI feature on August 10 for model 1 and also for the NDRE feature on August 24 for model 5. That means that in the end of the growth season a higher observed NDVI/NDRE leads to a higher harvest yield.

For model 8 we see that in the beginning of the season it seems like the harvest yield increases with a higher NDVI, but then later on towards the middle of the growth season we see that a higher observed NDVI leads to a decrease in harvest yield. Which can indicate that the crop needs to ripe. Besides on model 8, we also observe these dives in harvest yield for the features during the middle of the growth season on model 1, 2, 4, 5, and 7.

Generally, we observe that at the beginning and end of the growth season it seems that a high observed value of NDVI and NDRE leads to a higher harvest yield in combination with a steady NDVI/NDRE value during the middle of the growth season.

In [2]:

```
import pathlib
import site
BDICG_git_repo_path = '../MVP4_partially_complete_dataset/'
site.addsitedir(BDICG_git_repo_path)

import dask
import dask.distributed
import geopandas as gpd
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import train_test_split
from sklearn.ensemble.partial_dependence import plot_partial_dependence
from sklearn.metrics import mean_absolute_error
import seaborn as sns

from field_raster_model import (remote_mkdir)
sns.set()
pd.set_option('display.max_columns', 500)
```

Dask client

In [3]:

```
client = dask.distributed.Client('localhost:8786')
#client.restart()
client.upload_file(BDICG_git_repo_path + 'field_raster_model.py')
client
```

```
distributed.comm.tcp - WARNING - Could not set timeout on TCP stream: [Errno 92] Protocol not availa
ble
distributed.comm.tcp - WARNING - Could not set timeout on TCP stream: [Errno 92] Protocol not availa
ble
```

Out[3]:

Client

- **Scheduler:** tcp://localhost:8786
- **Dashboard:** http://localhost:8787/status (http://localhost:8787/status)

Cluster

- **Workers:** 20
- **Cores:** 400
- **Memory:** 8.01 TB

In [4]:

```
def get_interpreter():
    import sys
    return sys.executable

print('Local interpreter: {}'.format(get_interpreter()))
print('Client interpreter: {}'.format(dask.delayed(get_interpreter)().compute()))
```

```
Local interpreter: /home/donj/anaconda3/envs/py36_v3/bin/python
Client interpreter: /opt/miniconda3/envs/py36_v3/bin/python
```

In [5]:

```
# Output paths
OUTPUT_FEATURE_PATH = '/scratch/BDICG/feature_set_test_2018_12_13_13_35_59/'
DF_SAMPLES_FILE_PATH = pathlib.Path(OUTPUT_FEATURE_PATH)/'df_samples.parquet.brotli'
FIG_PATH = pathlib.Path('./feature_importance_plots')
FIG_PATH.mkdir(exist_ok=True)

print('OUTPUT_FEATURE_PATH: {}'.format(OUTPUT_FEATURE_PATH))
```

```
OUTPUT_FEATURE_PATH: /scratch/BDICG/feature_set_test_2018_12_13_13_35_59/
```

In [6]:

```
feature_set = dask.delayed(pd.read_parquet)(DF_SAMPLES_FILE_PATH, engine='pyarrow')
feature_set.head().compute()
```

Out[6]:

				('S2_L1C_B01', '03-09 - March 9')	('S2_L1C_B01', '03-23 - March 23')	('S2_L1C_B01', '04-06 - April 6')	('S2_L1C_B '04-20 - Ap 20')
ID_DDS_field	**harvest_year**	**N**	**E**				
4	**2017**	**6178582.5**	**565572.5**	0.138164	0.129424	0.128273	0.127705
			565577.5	0.138264	0.129524	0.128373	0.127805
			565582.5	0.138264	0.129524	0.128373	0.127805
			565587.5	0.138350	0.129623	0.128473	0.127905
		6178587.5	**565572.5**	0.138079	0.129325	0.128208	0.127678

Feature extraction and model fitting

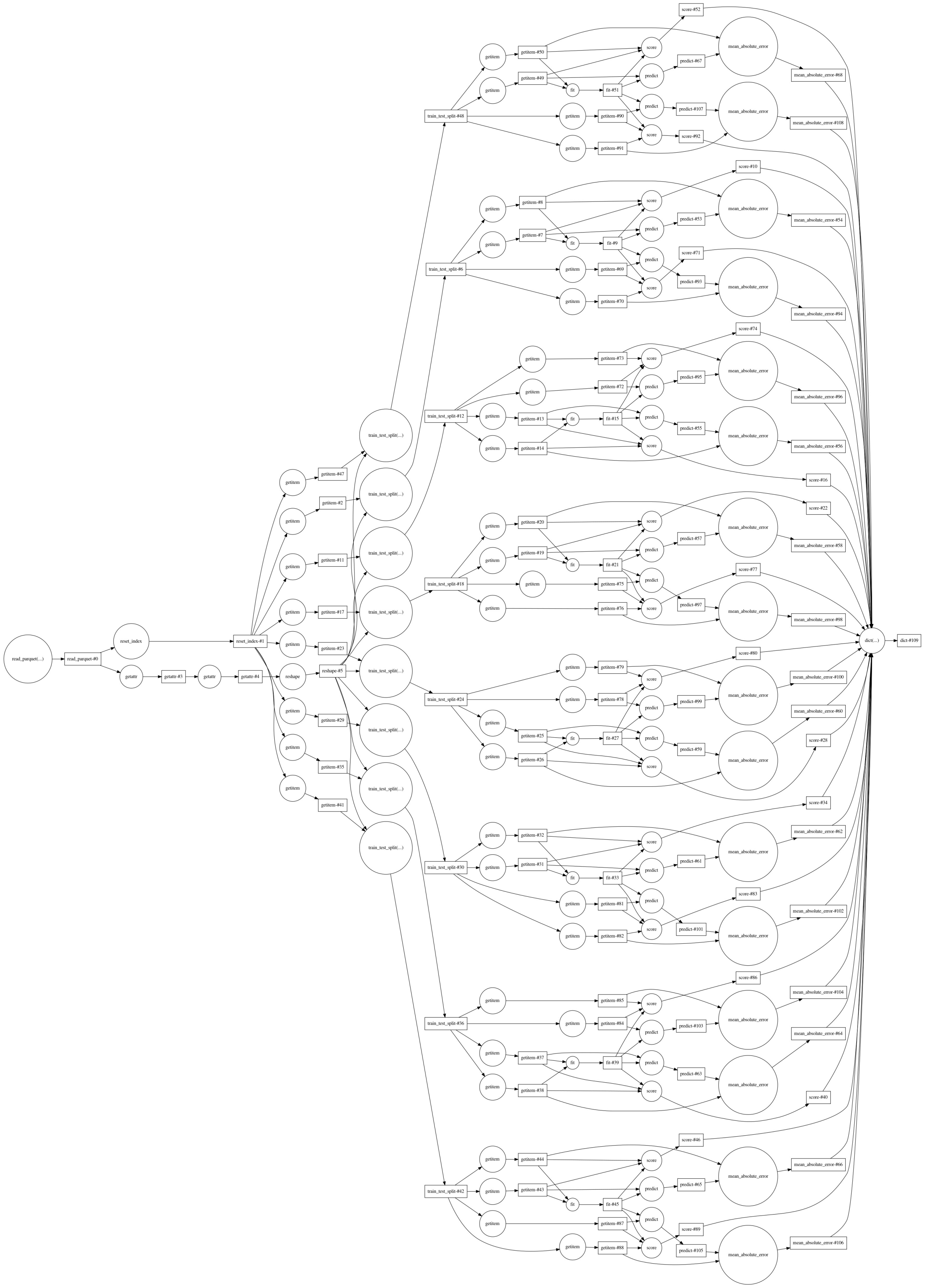
In [7]:

```
y = feature_set.harvest_dry_yield.values.reshape(-1, 1) # Response variable
```

In [8]:

```
models = {
    'NDVI': ['NDVI'],
    'NDRE': ['NDRE'],
    'MSAVI2': ['MSAVI2'],
    'NDVI_DTM_N_E': ['NDVI', 'DTM'],
    'NDRE_DTM_N_E': ['NDRE', 'DTM'],
    'MSAVI2_DTM_N_E': ['MSAVI2', 'DTM'],
    'NDVI_NDRE_MSAVI2': ['NDVI', 'NDRE', 'MSAVI2'],
    'NDVI_NDRE_MSAVI2_DTM_N_E': ['NDVI', 'NDRE', 'MSAVI2', 'DTM']
}

col_names = dict()
X_train_split = dict()
regressors = dict()
scores = {('train', 'R^2'): dict(), ('train', 'MAE'): dict(),
          ('test', 'R^2'): dict(), ('test', 'MAE'): dict()}

feature_set_columns = feature_set.columns.compute()
feature_set_and_index = feature_set.reset_index()
for model, model_features in models.items():
    # Identify names of features to fit model to
    feature_names = []
    for feature in model_features:
        feature_names.extend([col for col in feature_set_columns if feature in col])
    if model.endswith('N_E'):
        feature_names.extend(['N', 'E'])

    # Extract features
    X = feature_set_and_index[feature_names]
    col_names[model] = X.columns

    # Split the training and testing feature set
    X_train, X_test, y_train, y_test = dask.delayed(
                            train_test_split, nout=4)(X, y, random_state=42)
    X_train_split[model] = X_train

    regressor = GradientBoostingRegressor()
    regressor = dask.delayed(regressor.fit)(X_train, y_train)
    regressors[f'regressor_{model}'] = regressor

    # The predicted response variable, harvest_dry_yield.
    y_pred_train = dask.delayed(regressor.predict)(X_train)
    y_pred_test = dask.delayed(regressor.predict)(X_test)

    # The R^2 and the mean absolute error, mae.
    scores[('train','R^2')][model] = dask.delayed(regressor.score)(X_train, y_train)
    scores[('train', 'MAE')][model] = dask.delayed(mean_absolute_error)(y_train, y_pred_train)
    scores[('test', 'R^2')][model] = dask.delayed(regressor.score)(X_test, y_test)
    scores[('test', 'MAE')][model] = dask.delayed(mean_absolute_error)(y_test, y_pred_test)

regressors = dask.delayed(regressors)
scores = dask.delayed(scores)
scores.visualize(rankdir='LR')
```

Out[8]:



In [9]:

```
regressors = client.persist(regressors)
scores = client.compute(scores)
dask.distributed.progress(scores)
# Running time: ~5min 45.0s
```

```
distributed.comm.tcp - WARNING - Could not set timeout on TCP stream: [Errno 92] Protocol not availa
ble
```

In [10]:

```
pd.DataFrame(scores.result())
```

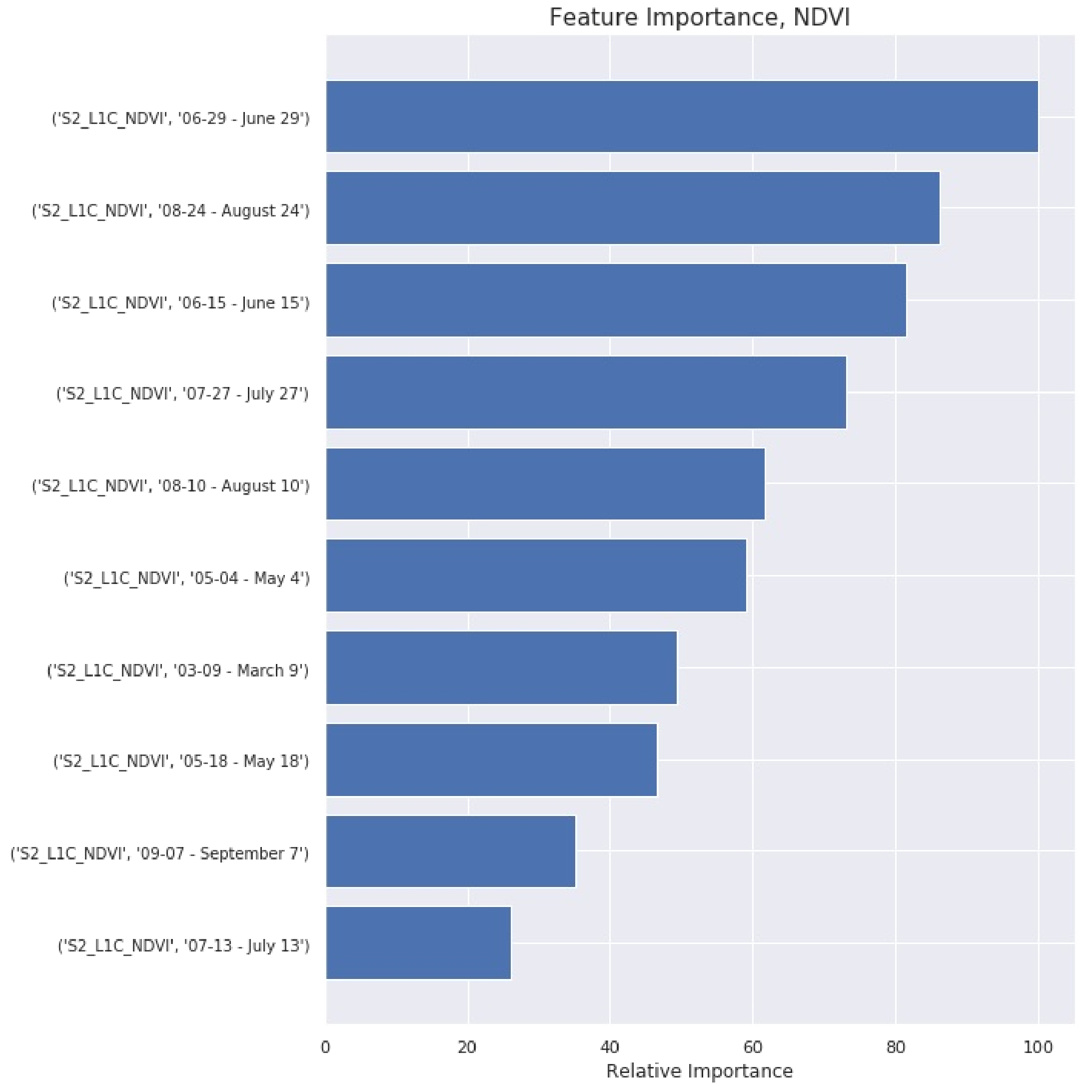
Out[10]:

	train		test	
	R^2	**MAE**	**R^2**	**MAE**
MSAVI2	0.482783	1594.655003	0.477233	1602.935351
MSAVI2_DTM_N_E	0.592200	1382.589521	0.591083	1386.304829
NDRE	0.500029	1547.966726	0.496541	1555.538042
NDRE_DTM_N_E	0.612631	1331.784758	0.612130	1334.770375
NDVI	0.504780	1540.306923	0.500461	1547.333477
NDVI_DTM_N_E	0.609829	1335.577543	0.608868	1338.973770
NDVI_NDRE_MSAVI2	0.556239	1440.296569	0.551393	1448.683882
NDVI_NDRE_MSAVI2_DTM_N_E	0.627061	1295.158336	0.625815	1298.917392

Plot of the feature importance and the partial dependence

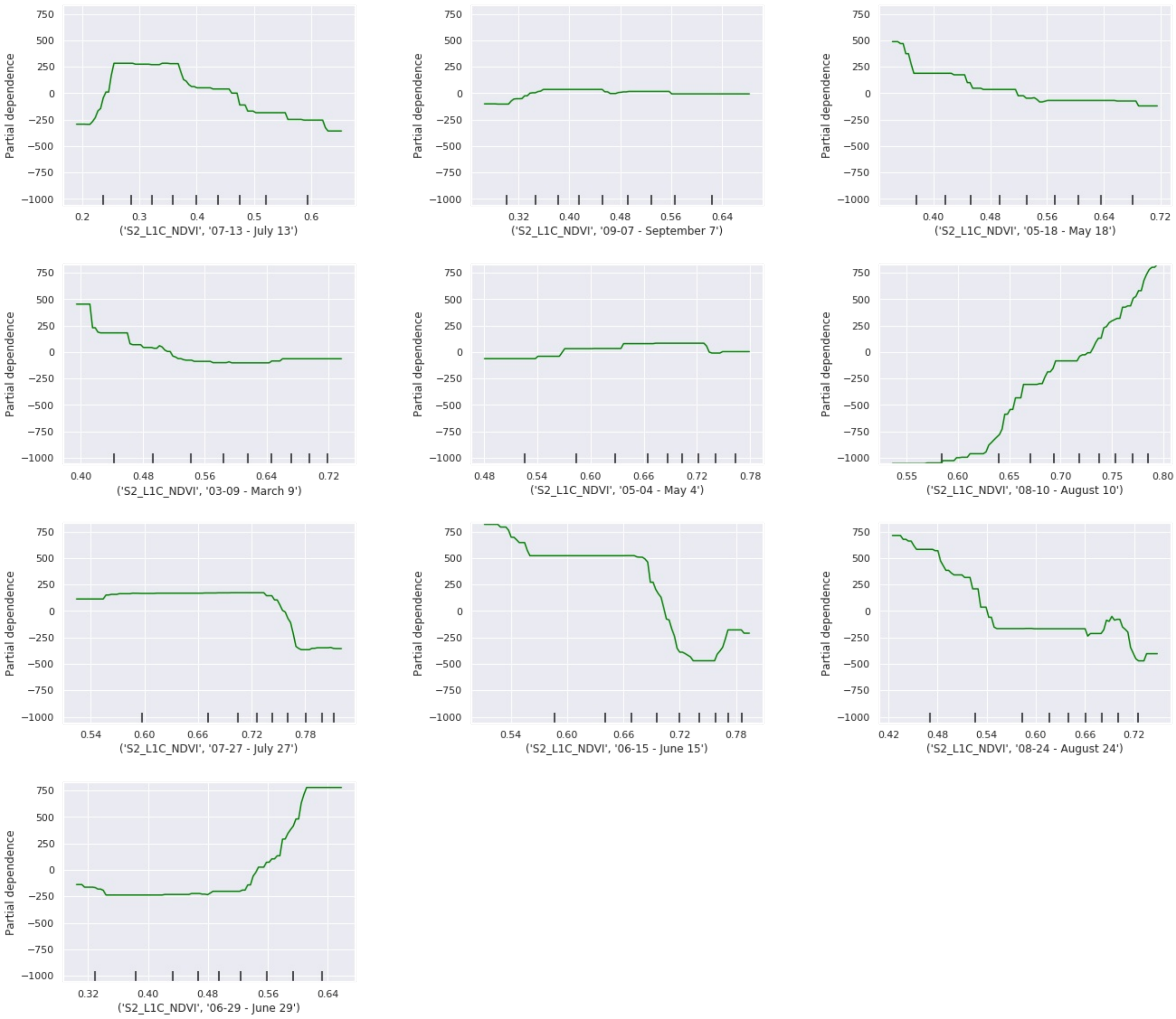
In [13]:

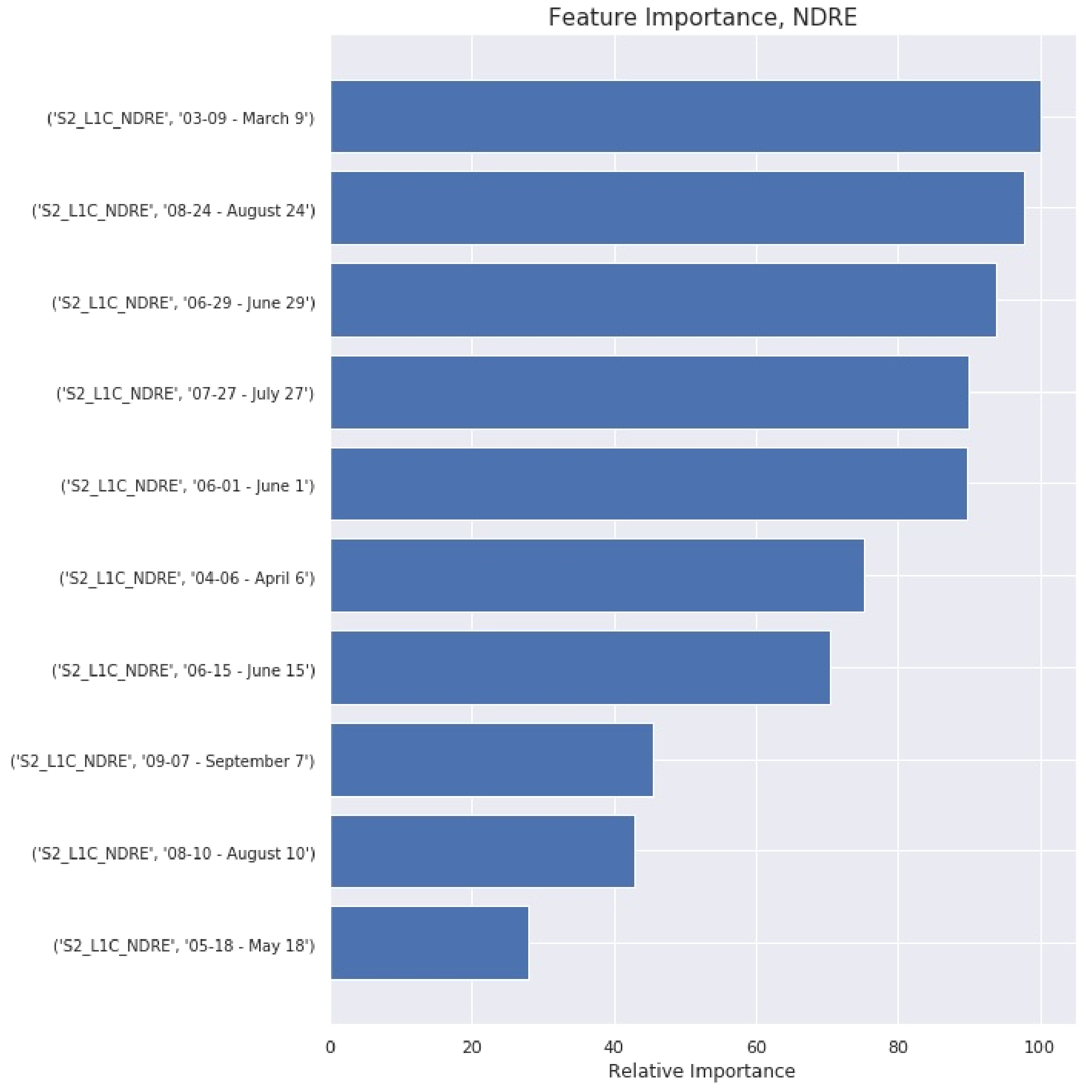
```
for model in models.keys():
    fig, ax = plt.subplots(1, figsize=(10,10))
    regressor = regressors[f'regressor_{model}'].compute()
    feature_importances = regressor.feature_importances_

    #Feature importances relative to the max importance
    feature_importances = 100.0 * (feature_importances / feature_importances.max())

    # Sorted index
    idx = np.argsort(feature_importances)
    pos = np.arange(idx.shape[0]) + .5
    feature_set_names = col_names[model][idx].compute()

    # Choosing the most impartant features to plot
    sorted_idx = np.sort(idx)
    most_important_features = feature_importances[idx[sorted_idx]][-10:]
    most_important_features_names = feature_set_names[idx[sorted_idx]][-10:]
    pos = pos[-10:]

    ax.barh(pos, most_important_features, align='center')
    ax.set_yticks(pos)
    ax.set_yticklabels(most_important_features_names, fontsize=10)
    ax.set_xlabel('Relative Importance')
    ax.set_title(f'Feature Importance, {model}', fontsize=15)
    fig.tight_layout()
    fig.savefig(FIG_PATH/f'Feature importance_{model}.pdf')

    # Partial dependence plot
    fig, axs = plot_partial_dependence(regressor, X_train_split[model].compute(), sorted_idx[:10],
                                       feature_names=most_important_features_names, figsize=(20,20))
    fig.suptitle(f'Partial dependence plots, {model}', fontsize=15)
    plt.subplots_adjust(top=0.9)
    fig.savefig(FIG_PATH/f'Partial_dependence_{model}.pdf')
```



Feature Importance, NDVI
('S2_L1C_NDVI', '06-29 - June 29')
('S2_L1C_NDVI', '08-24 - August 24')
('S2_L1C_NDVI', '06-15 - June 15')
('S2_L1C_NDVI', '07-27 - July 27')
('S2_L1C_NDVI', '08-10 - August 10')
('S2_L1C_NDVI', '05-04 - May 4')
('S2_L1C_NDVI', '03-09 - March 9')
('S2_L1C_NDVI', '05-18 - May 18')
('S2_L1C_NDVI', '09-07 - September 7')
('S2_L1C_NDVI', '07-13 - July 13')
0
20
40
60
80
100
Relative Importance

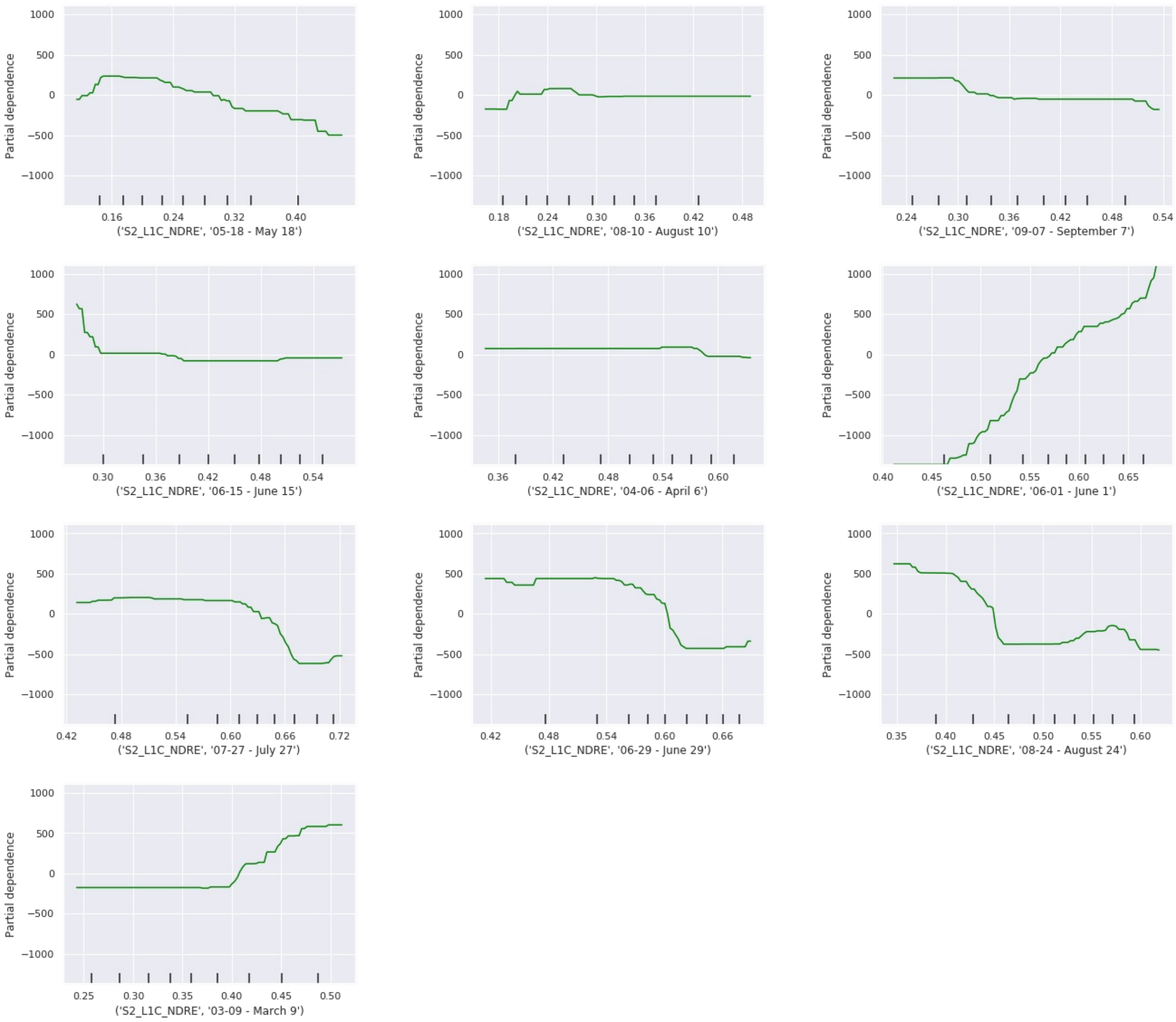
Partial dependence plots, NDVI

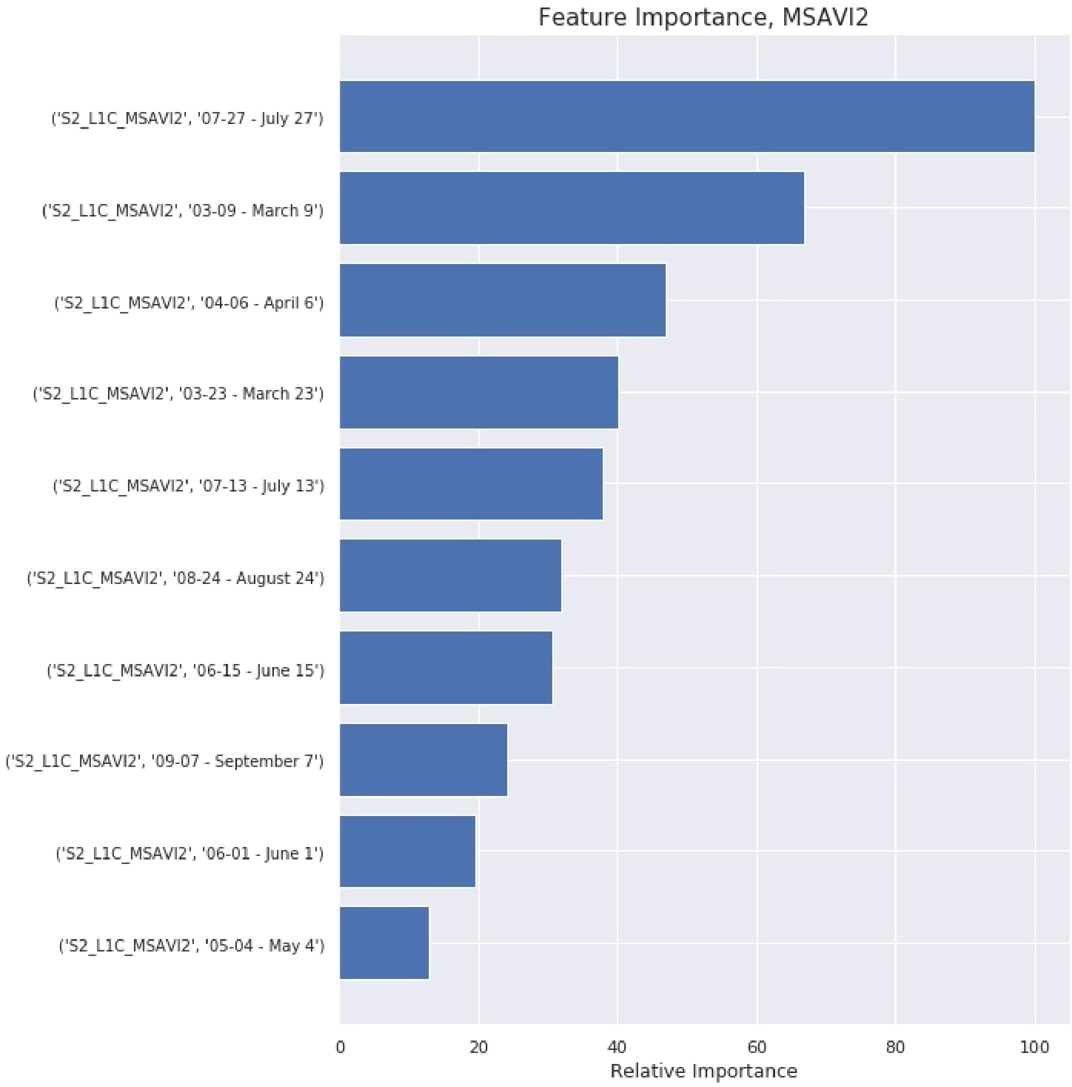




Feature Importance, NDRE
('S2_L1C_NDRE', '03-09 - March 9')
('S2_L1C_NDRE', '08-24 - August 24')
('S2_L1C_NDRE', '06-29 - June 29')
('S2_L1C_NDRE', '07-27 - July 27')
('S2_L1C_NDRE', '06-01 - June 1')
('S2_L1C_NDRE', '04-06 - April 6')
('S2_L1C_NDRE', '06-15 - June 15')
('S2_L1C_NDRE', '09-07 - September 7')
('S2_L1C_NDRE', '08-10 - August 10')
('S2_L1C_NDRE', '05-18 - May 18')
0
20
40
60
80
100
Relative Importance

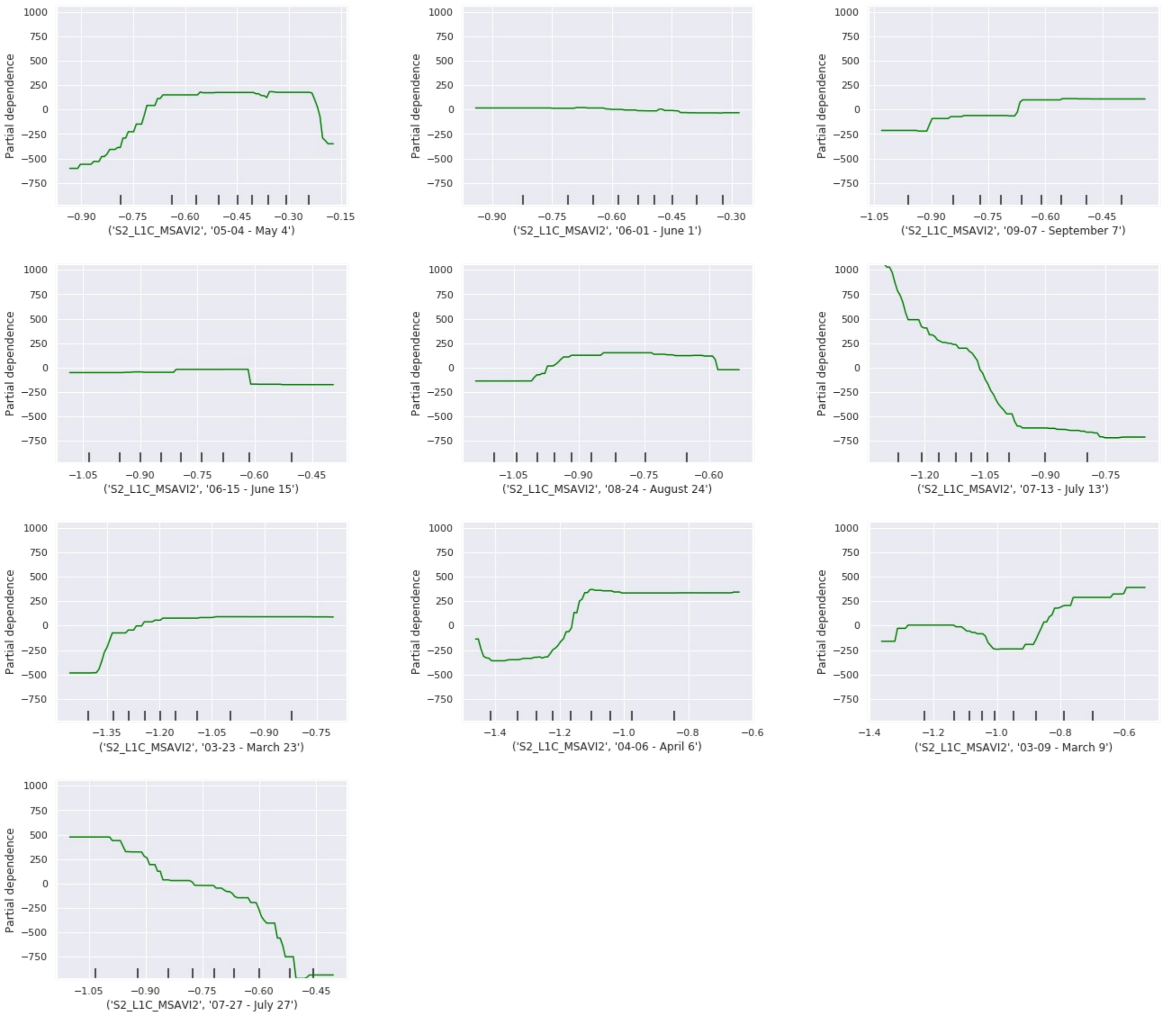
Partial dependence plots, NDRE

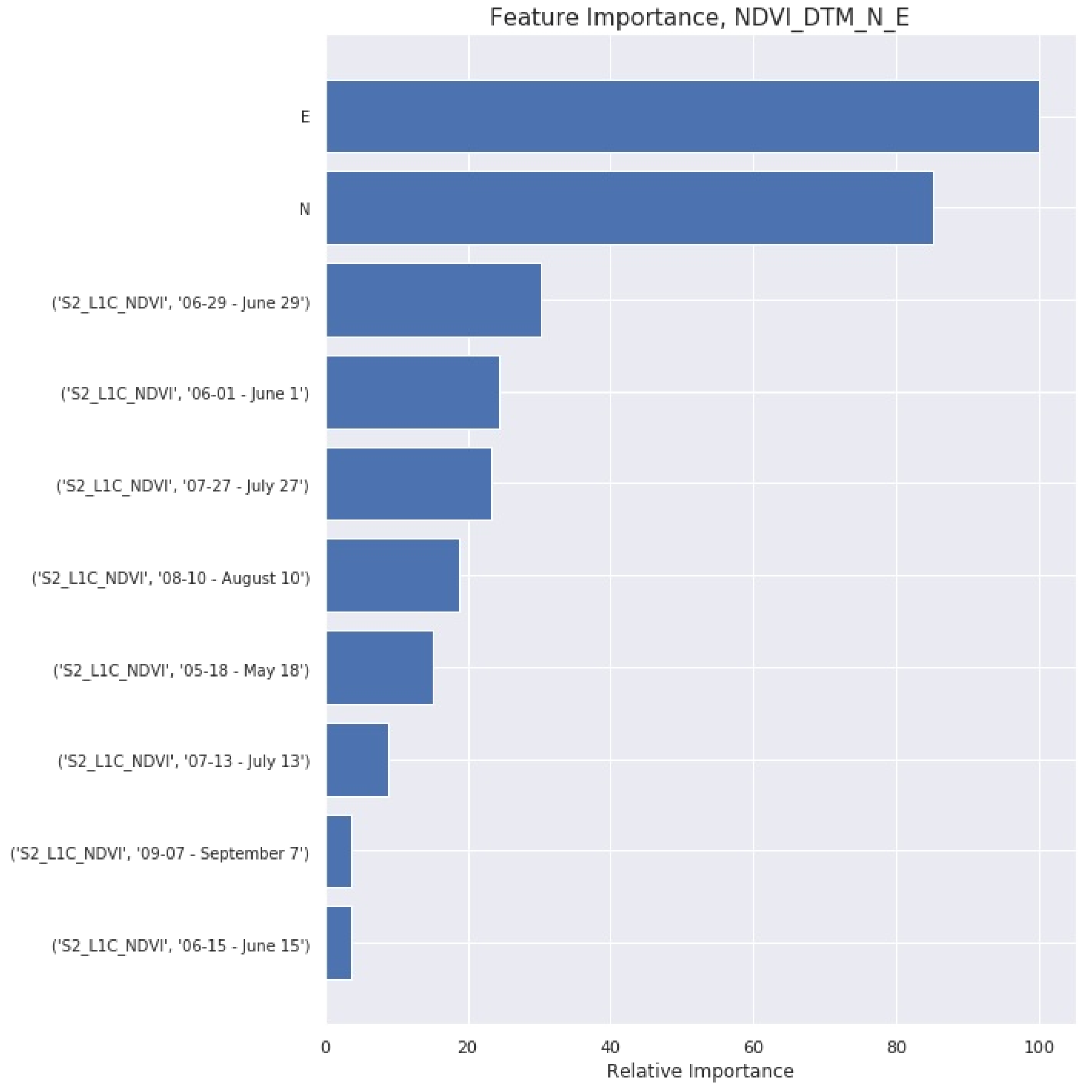




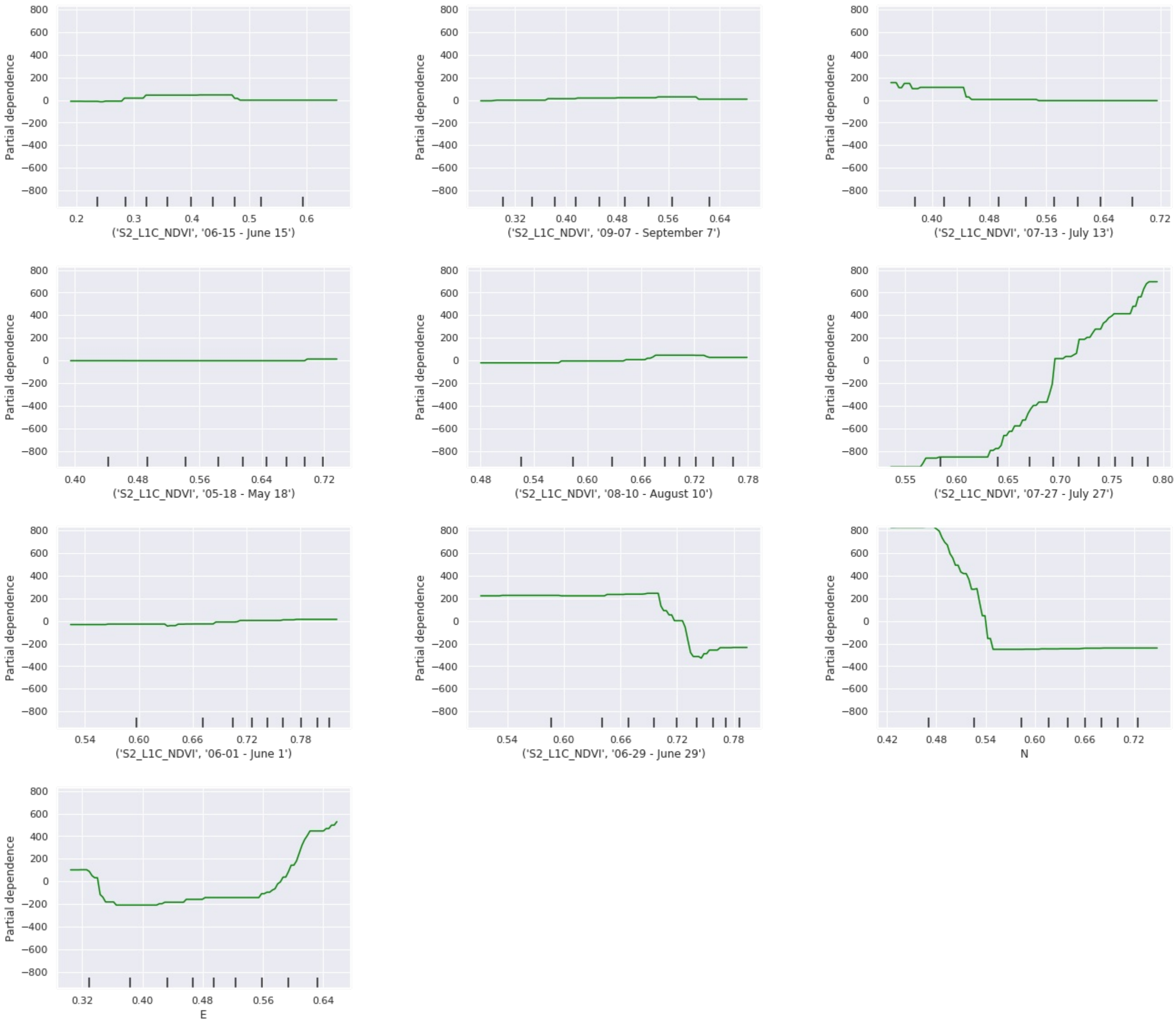
Feature Importance, MSAVI2
('S2_L1C_MSAVI2', '07-27 - July 27')
('S2_L1C_MSAVI2', '03-09 - March 9')
('S2_L1C_MSAVI2', '04-06 - April 6')
('S2_L1C_MSAVI2', '03-23 - March 23')
('S2_L1C_MSAVI2', '07-13 - July 13')
('S2_L1C_MSAVI2', '08-24 - August 24')
('S2_L1C_MSAVI2', '06-15 - June 15')
('S2_L1C_MSAVI2', '09-07 - September 7')
('S2_L1C_MSAVI2', '06-01 - June 1')
('S2_L1C_MSAVI2', '05-04 - May 4')
0
20
40
60
80
100
Relative Importance

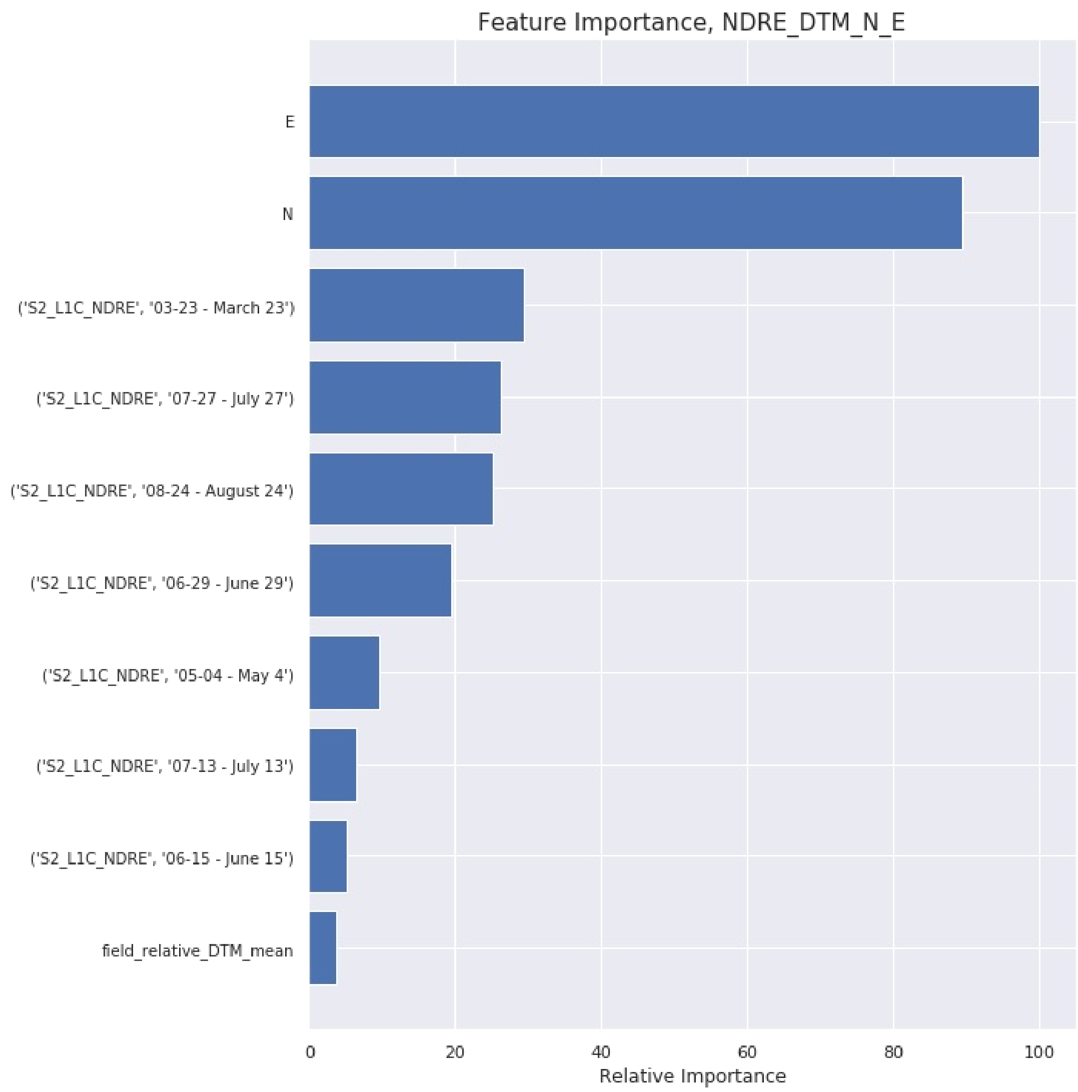
Partial dependence plots, MSAVI2




Feature Importance, NDVI_DTM_N_E
E
N
('S2_L1C_NDVI', '06-29 - June 29')
('S2_L1C_NDVI', '06-01 - June 1')
('S2_L1C_NDVI', '07-27 - July 27')
('S2_L1C_NDVI', '08-10 - August 10')
('S2_L1C_NDVI', '05-18 - May 18')
('S2_L1C_NDVI', '07-13 - July 13')
('S2_L1C_NDVI', '09-07 - September 7')
('S2_L1C_NDVI', '06-15 - June 15')
0
20
40
60
80
100
Relative Importance

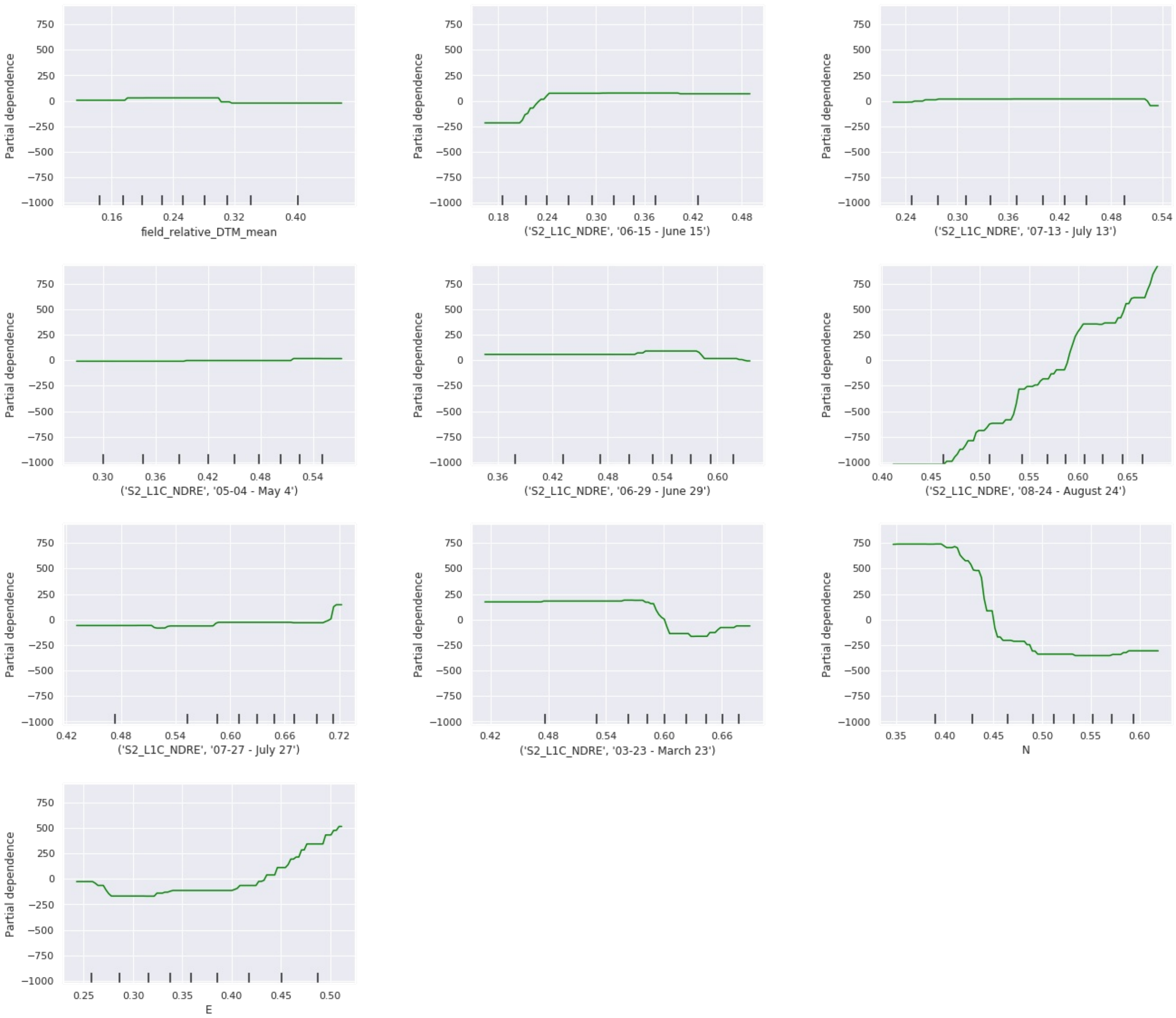
Partial dependence plots, NDVI_DTM_N_E

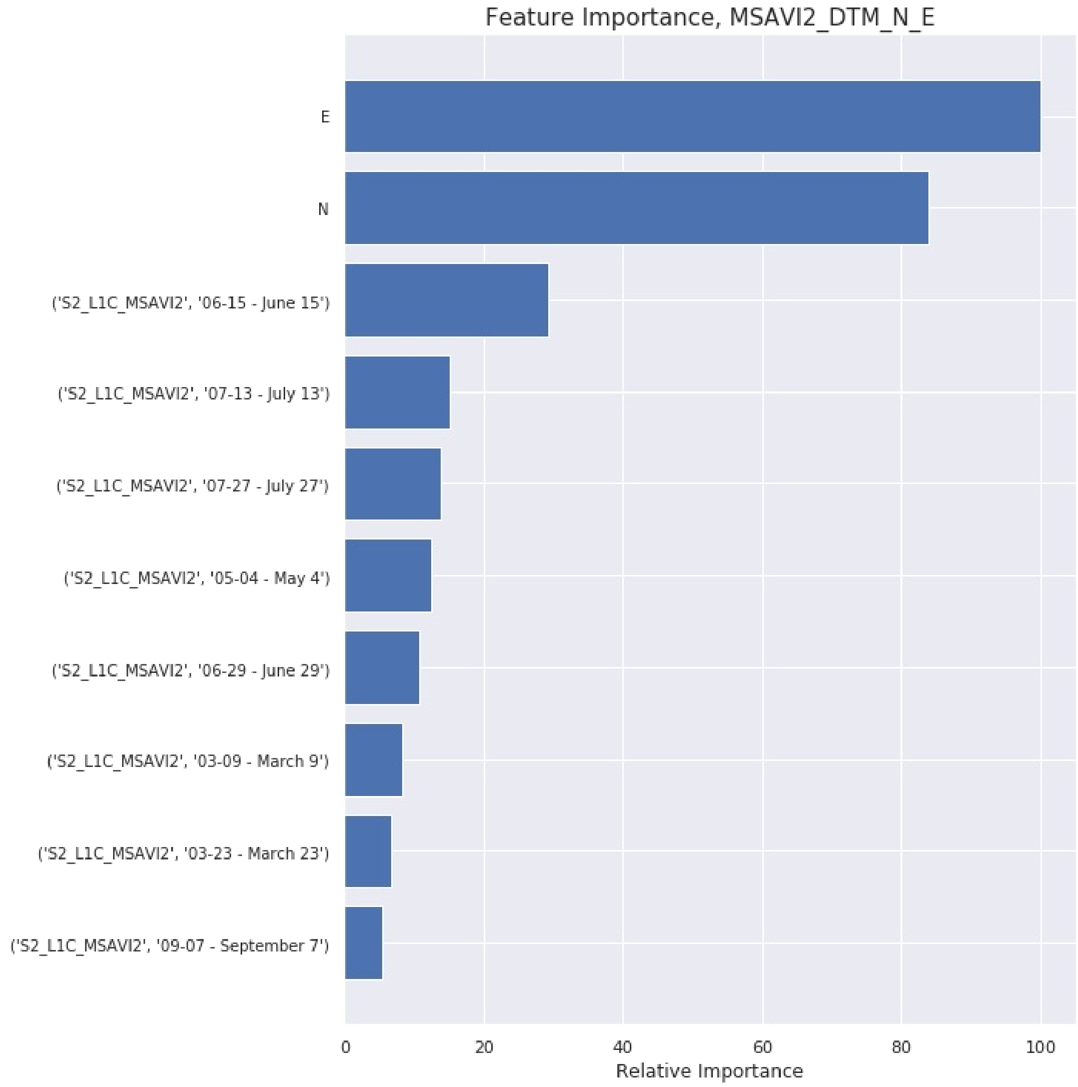




Feature Importance, NDRE_DTM_N_E
E
N
('S2_L1C_NDRE', '03-23 - March 23')
('S2_L1C_NDRE', '07-27 - July 27')
('S2_L1C_NDRE', '08-24 - August 24')
('S2_L1C_NDRE', '06-29 - June 29')
('S2_L1C_NDRE', '05-04 - May 4')
('S2_L1C_NDRE', '07-13 - July 13')
('S2_L1C_NDRE', '06-15 - June 15')
field_relative_DTM_mean
0
20
40
60
80
100
Relative Importance

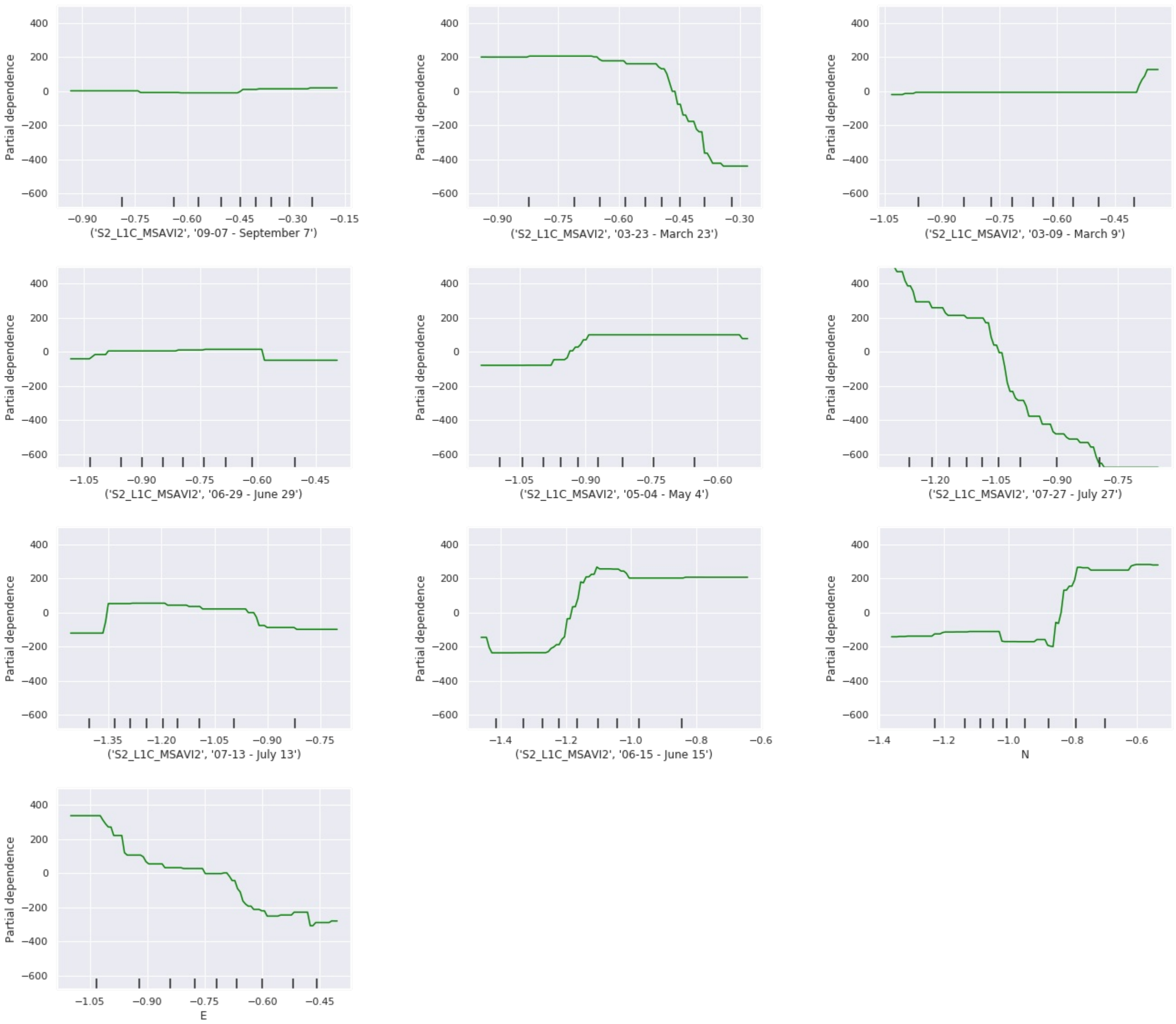
Partial dependence plots, NDRE_DTM_N_E

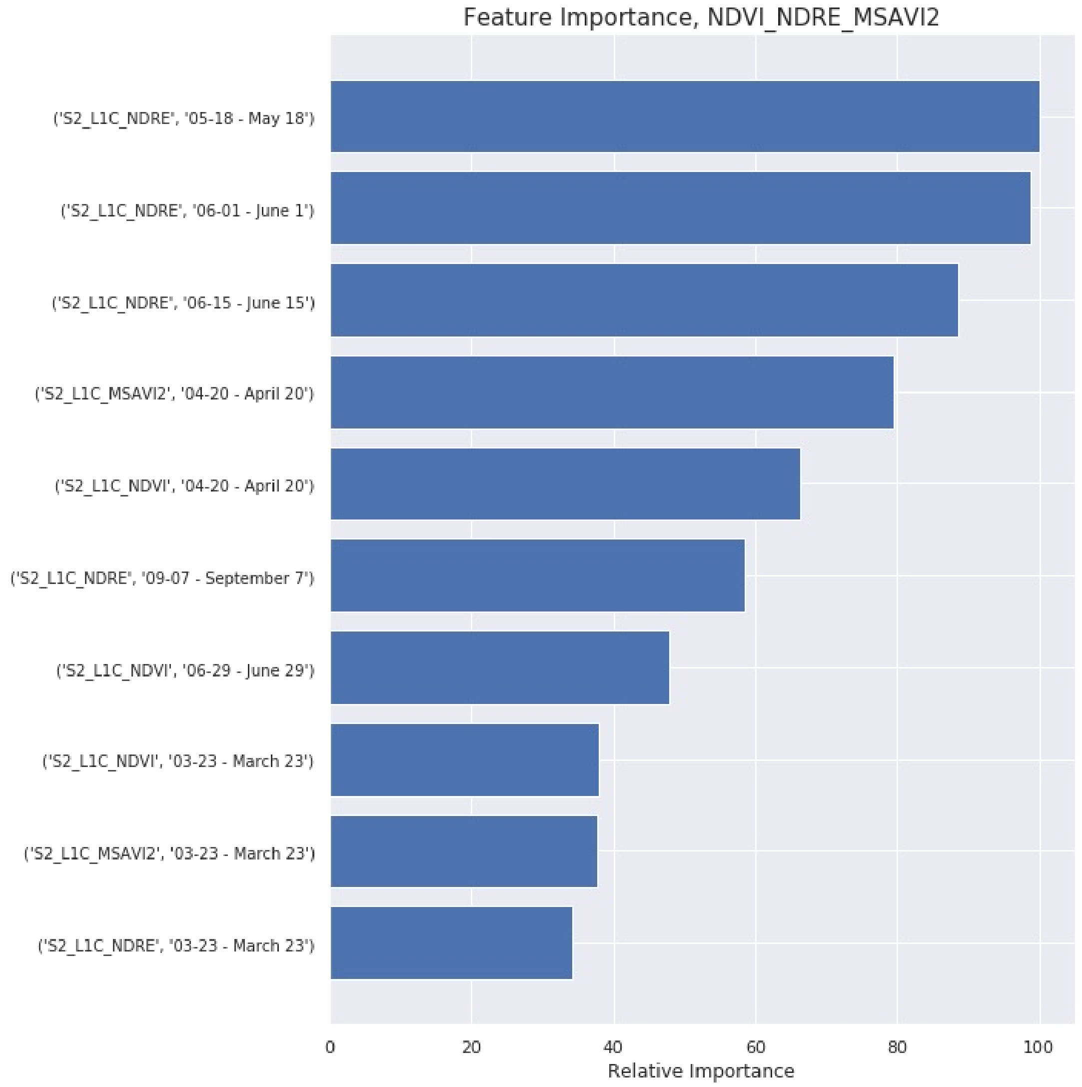




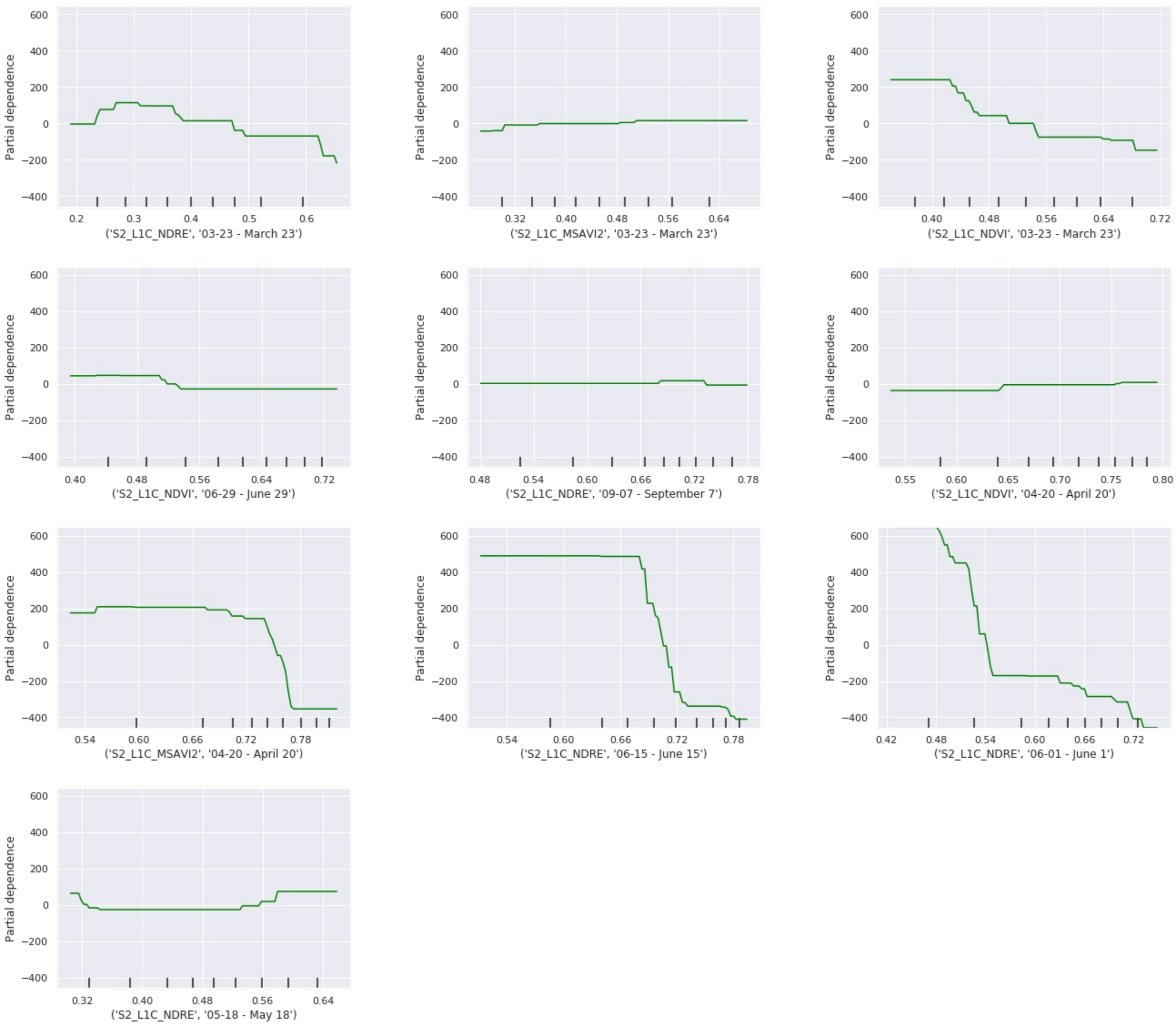
Feature Importance, MSAVI2_DTM_N_E
E
N
('S2_L1C_MSAVI2', '06-15 - June 15')
('S2_L1C_MSAVI2', '07-13 - July 13')
('S2_L1C_MSAVI2', '07-27 - July 27')
('S2_L1C_MSAVI2', '05-04 - May 4')
('S2_L1C_MSAVI2', '06-29 - June 29')
('S2_L1C_MSAVI2', '03-09 - March 9')
('S2_L1C_MSAVI2', '03-23 - March 23')
('S2_L1C_MSAVI2', '09-07 - September 7')
0
20
40
60
80
100
Relative Importance

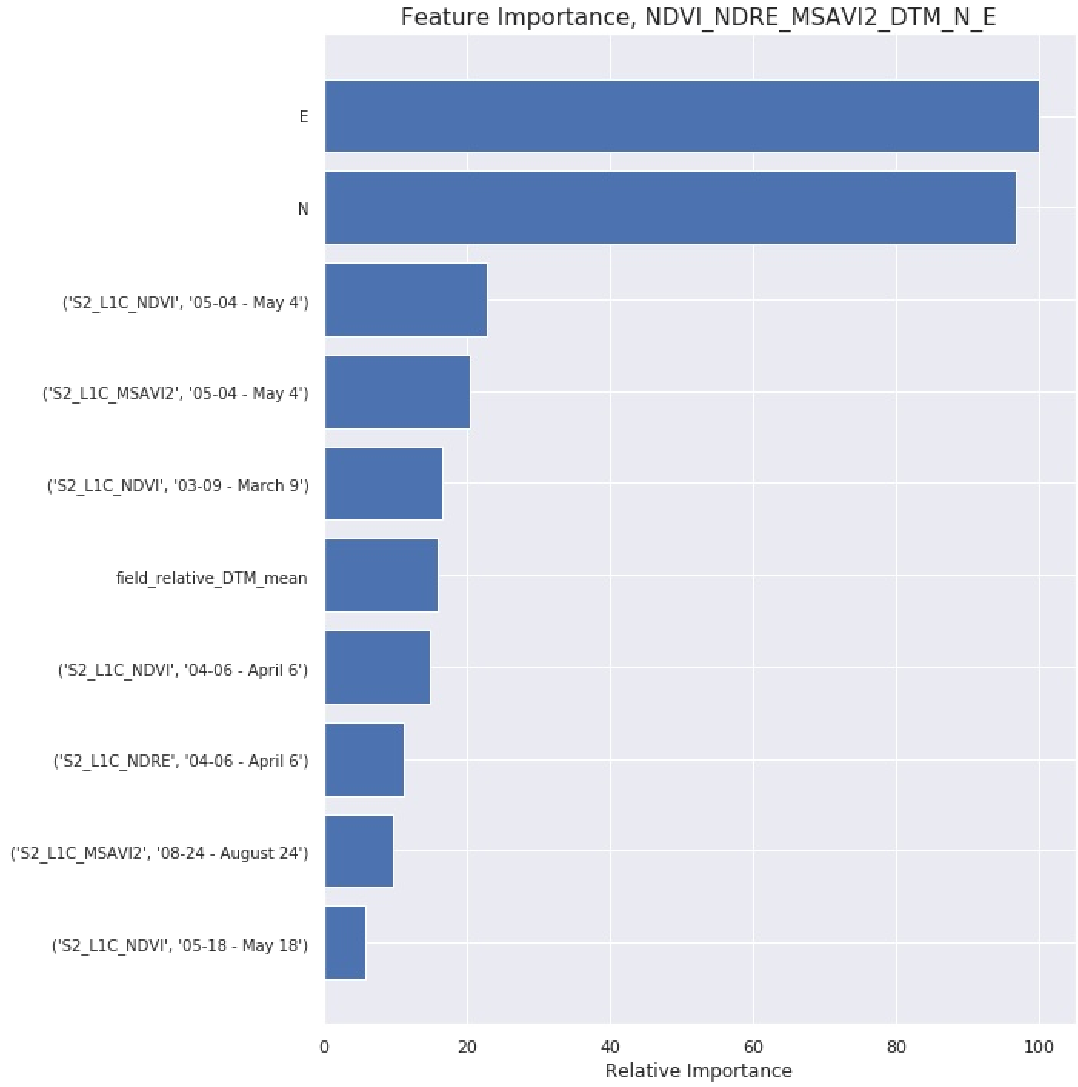
Partial dependence plots, MSAVI2_DTM_N_E




Feature Importance, NDVI_NDRE_MSAVI2
('S2_L1C_NDRE', '05-18 - May 18')
('S2_L1C_NDRE', '06-01 - June 1')
('S2_L1C_NDRE', '06-15 - June 15')
('S2_L1C_MSAVI2', '04-20 - April 20')
('S2_L1C_NDVI', '04-20 - April 20')
('S2_L1C_NDRE', '09-07 - September 7')
('S2_L1C_NDVI', '06-29 - June 29')
('S2_L1C_NDVI', '03-23 - March 23')
('S2_L1C_MSAVI2', '03-23 - March 23')
('S2_L1C_NDRE', '03-23 - March 23')
0
20
40
60
80
100
Relative Importance

Partial dependence plots, NDVI_NDRE_MSAVI2





Feature Importance, NDVI_NDRE_MSAVI2_DTM_N_E
E
N
('S2_L1C_NDVI', '05-04 - May 4')
('S2_L1C_MSAVI2', '05-04 - May 4')
('S2_L1C_NDVI', '03-09 - March 9')
field_relative_DTM_mean
('S2_L1C_NDVI', '04-06 - April 6')
('S2_L1C_NDRE', '04-06 - April 6')
('S2_L1C_MSAVI2', '08-24 - August 24')
('S2_L1C_NDVI', '05-18 - May 18')
0
20
40
60
80
100
Relative Importance

Partial dependence plots, NDVI_NDRE_MSAVI2_DTM_N_E