Hi-Lasso-spark

Release 0.0.1

Aug 28, 2021

Contents:

Hi-LASSO_Spark(High-Demensinal LASSO Spark) is to improve the LASSO solutions for extremely high-dimensional data using pyspark.

PySpark is the Python API written in python to support Apache Spark. Apache Spark is a distributed framework that can handle Big Data analysis. Spark is basically a computational engine, that works with huge sets of data by processing them in parallel and batch systems.

CHAPTER 1

Installation guide

1.1 Dependencies

`Hi-LASSO-spark` support Python 3.6+, Additionally, you will need `numpy`, `scipy`, and `glmnet`.

1.2 Installing pyspark

To run Hi-LASSO-spark, you need to install a pyspark. In the prompt environment, run the following installation and update:

```
pip install pyspark
```

After installing pyspark, it will be installed with 2.4.6 version. To run Hi-LASSO-spark, you must upgrade to version 3.0.0.:

```
pip install pyspark --upgrade
```

After the upgrade, 3.0.0 version is executed.

1.3 Installing Hi-LASSO-spark

`Hi-LASSO-spark` is available through PyPI and can easily be installed with a pip install:

```
pip install hi_lasso_spark
```

The PyPI version is updated regularly, however for the latest update, you should clone from GitHub and install it directly:

```
git clone https://github.com/datax-lab/Hi_Lasso_spark.git
cd hi_lasso_spark
python setup.py
```

1.4 Installation error

If the *glmnet* packages failed, you can try a follow solutions.

```
error: extension '_glmnet' has Fortran sources but no Fortran compiler found
```

You should install `anaconda3` and then install conda `fortran-compiler`.

- anaconda3
- fortran compiler

```
error: Microsoft Visual C++ 14.0 is required. Get it with "Build Tools for Visual␣
↪Studio": https://visualstudio.microsoft.com/downloads/
```

You need to install *Microsoft Visual C++ 14.0*.

- reference

CHAPTER 2

API Reference

2.1 Hi-LASSO_spark model

class hi_lasso_spark.Hi_LASSO_spark.**HiLASSO_Spark**(*X*, *y*, *X_test='auto'*, *y_test='auto'*, *alpha=0.05*, *q1='auto'*, *q2='auto'*, *L=30*, *cv=5*, *node='auto'*, *logistic=False*)

Bases: object

Hi-LASSO_Spark(High-Demensinal LASSO Spark) is to improve the LASSO solutions for extremely high-dimensional data using pyspark.

PySpark is the Python API written in python to support Apache Spark. Apache Spark is a distributed framework that can handle Big Data analysis. Spark is basically a computational engine, that works with huge sets of data by processing them in parallel and batch systems.

The main contributions of Hi-LASSO are as following:

- Rectifying systematic bias introduced by bootstrapping.
- Refining the computation for importance scores.
- Providing a statistical strategy to determine the number of bootstrapping.
- Taking advantage of global oracle property.
- Allowing tests of significance for feature selection with appropriate distribution.

Parameters

- **X** (*array-like of shape (n_sample, n_feature)*) – predictor variables
- **y** (*array-like of shape (n_sample,)*) – response variables
- **q1** (*'auto' or int, optional [default = 'auto']*) – The number of predictors to randomly selecting in Procedure 1. When to set 'auto', use q1 as number of samples.

- **q2** (*'auto' or int, optional [default = 'auto']*) – The number of predictors to randomly selecting in Procedure 2. When to set 'auto', use q2 as number of samples.
- **L** (*int [default=30]*) – The expected value at least how many times a predictor is selected in a bootstrapping.
- **alpha** (*float [default=0.05]*) – significance level used for significance test for feature selection
- **node** (*Node refers to node which runs the application code in the cluster.*) – If you do not specify the number of nodes, the 8 nodes are automatically set to the default node.

Variables

- **n** (*int*) – number of samples.
- **p** (*int*) – number of features.

Examples

```
>>> from Hi_LASSO_spark_fin import HiLASSO_Spark
>>> model = HiLASSO_Spark(X, y)
>>> model.fit()
```

```
>>> model.coef_
>>> model.p_values_
>>> model.selected_var_
```

Calculate_p_value(*coef_result*)
Compute p-values of each predictor for Statistical Test of Variable Selection.

Estimate_coefficient_Adaptive(*value*)
Estimation of coefficients for each bootstrap sample using Adaptive_LASSO

Returns coef_result

Return type coefficient for Adaprive_LASSO

Estimate_coefficient_Adaptive_logistic(*value*)
Estimation of coefficients for each bootstrap sample using Adaptive_LASSO

Returns coef_result

Return type coefficient for Adaprive_LASSO

Estimate_coefficient_Elastic(*value*)
Estimation of coefficients for each bootstrap sample using Elastic_net

Returns coef_result

Return type coefficient for Elastic_Net

fit()
Fit the model with Procedure 1 and Procedure 2.

Procedure 1: Compute importance scores for predictors.

Procedure 2: Compute coefficients and Select variables.

Parallel processing of Spark One important parameter for parallel collections is the number of partitions to cut the dataset into. Spark will run one task for each partition of the cluster. Typically you want 2-4

partitions for each CPU in your cluster. Normally, Spark tries to set the number of partitions automatically based on your cluster. However, you can also set it manually by passing it as a second parameter to parallelize (e.g. sc.parallelize(data, 10)).

Variables

- **sc.parallelize()** (*method*) – The sc.parallelize() method is the SparkContext's parallelize method to create a parallelized collection. This allows Spark to distribute the data across multiple nodes, instead of depending on a single node to process the data.
- **map()** (*method*) – A map is a transformation operation in Apache Spark. It applies to each element of RDD and it returns the result as new RDD. In the Map, operation developer can define his own custom business logic. The same logic will be applied to all the elements of RDD.
- **Procedure_1_coef** (*array*) – Estimated coefficients by Elastic_net.
- **Procedure_2_coef** (*array*) – Estimated coefficients by Adaptive_LASSO.
- **coef** (*array*) – Estimated coefficients by Hi-LASSO.
- **p_values** (*array*) – P-values of each coefficients.
- **selected_var** (*array*) – Selected variables by significance test.

Returns self

Return type object

standardization(*X*, *y*)

The response is mean-corrected and the predictors are standardized :param X: predictor :type X: array-like of shape (n_sample, n_feature) :param y: response :type y: array-like of shape (n_sample,)

Returns

- *np.ndarray*
- *scaled_X, scaled_y, std*

CHAPTER 3

Getting Started

3.1 Data load

[1]:
```
import pandas as pd
X = pd.read_csv('simulation_data_x.csv')
y = pd.read_csv('simulation_data_y.csv')
```

[2]:
```
X.head()
```

[2]:
```
         V1        V2        V3        V4        V5        V6        V7  \
0  0.101741  0.144909  0.235567  0.576186  0.299443  0.296395  0.902235
1 -0.105054 -0.110128 -0.033311 -0.042925 -0.752605 -0.794815 -1.699739
2 -0.478922 -0.058475 -0.620625  1.775435  0.935760  1.268173 -2.652118
3 -0.657057 -0.508105 -0.556453  0.057045 -0.344814 -0.557824 -0.814844
4 -0.939275 -0.780565 -0.495687 -0.308105 -0.604543 -0.272711  0.080763

         V8        V9       V10  ...       V91       V92       V93       V94  \
0  0.265811  0.420927  0.684045  ... -0.058677  0.497420 -1.124986  0.338215
1 -1.891533 -1.287547 -1.154547  ... -1.614948 -1.337878  0.795742  1.101117
2 -2.327299 -2.913926 -2.140256  ...  0.081620  0.999657 -0.594758  0.057804
3  0.016355 -0.384234 -0.022224  ...  0.077500 -1.489750 -0.151010  0.347814
4  0.542582  0.580669  0.162638  ...  0.677662  0.188278  0.716616 -1.290398

        V95       V96       V97       V98       V99      V100
0 -0.942943 -1.257044 -0.531471  1.236317  0.405682  0.387636
1 -0.920702 -0.098002 -0.269719  0.333092 -0.500367  1.340876
2 -1.259650  0.321864  0.992930  0.552269  1.253700  0.974151
3  2.281268 -1.275026 -0.141539 -0.335557  0.196004 -1.347782
4 -0.579556 -0.692827 -1.040820 -0.674525  1.355343  1.754870

[5 rows x 100 columns]
```

[3]:
```
y.head()
```

[3]:
```
         V1
0  2.811113
1  1.049249
2 -4.496389
3 -3.846408
4 -2.805357
```

3.2 General Usage

[4]:
```
from Hi_LASSO_pyspark import HiLASSO_Spark

model = HiLASSO_Spark(X, y, alpha=0.05, q1='auto', q2='auto', L=30, cv=5, node='auto',
↪ logistic=False)
model.fit()
```

```
C:\Users\Seungha\anaconda3\lib\site-packages\sklearn\externals\joblib\__init__.py:15:␣
↪FutureWarning: sklearn.externals.joblib is deprecated in 0.21 and will be removed␣
↪in 0.23. Please import this functionality directly from joblib, which can be␣
↪installed with: pip install joblib. If this warning is raised when loading pickled␣
↪models, you may need to re-serialize those models with scikit-learn 0.21+.
  warnings.warn(msg, category=FutureWarning)
C:\Users\Seungha\anaconda3\lib\site-packages\sklearn\externals\six.py:31:␣
↪FutureWarning: The module is deprecated in version 0.21 and will be removed in␣
↪version 0.23 since we've dropped support for Python 2.7. Please rely on the␣
↪official version of six (https://pypi.org/project/six/).
  "(https://pypi.org/project/six/).", FutureWarning)
```

```
Procedure_1_fin.
Procedure_2_fin.
```

[4]:
```
<Hi_LASSO_spark.HiLASSO_Spark at 0x280ac36e308>
```

[5]:
```
model.coef_
```

[5]:
```
array([ 6.21282205e-01,  2.27730867e+00, -2.12828920e-01,  7.91275602e-01,
        9.36945797e-02, -6.15373475e-02,  9.00107483e-01,  7.81416406e-01,
        2.33909585e-01, -2.91365004e-03, -3.78273386e-01,  0.00000000e+00,
       -2.41606493e-02,  0.00000000e+00, -6.58164921e-01,  1.54145412e-03,
       -1.16992273e-02,  2.51639371e-02,  0.00000000e+00,  7.55799788e-03,
        1.68828138e-01, -3.05823959e-02,  0.00000000e+00,  1.73225751e-03,
        3.01344168e-01,  0.00000000e+00,  1.91514369e-02,  3.97503818e-02,
        0.00000000e+00,  4.43804365e-02,  0.00000000e+00,  0.00000000e+00,
        4.32576398e-02,  5.52723676e-01,  0.00000000e+00,  0.00000000e+00,
        4.03342118e-04, -2.10036305e-03,  0.00000000e+00,  0.00000000e+00,
        6.95194714e-02,  0.00000000e+00,  5.69989592e-01,  0.00000000e+00,
       -6.26655745e-02,  2.29603115e-02, -1.44772894e-02,  0.00000000e+00,
        0.00000000e+00,  0.00000000e+00,  0.00000000e+00, -3.37721796e-02,
       -1.72170081e-01,  0.00000000e+00,  5.22259759e-03,  0.00000000e+00,
        1.57643093e-02,  1.84508844e-02, -3.35463900e-02,  8.92966775e-02,
        0.00000000e+00,  0.00000000e+00,  1.16838784e-03,  0.00000000e+00,
       -5.10785040e-03, -8.20826615e-04,  1.37965408e-02, -7.84592615e-03,
        4.05342662e-04, -8.13447460e-04,  1.62454783e-03,  2.11658823e-01,
        1.37239634e-02,  0.00000000e+00,  1.96548381e-02,  3.97642011e-04,
        3.98597583e-02,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       -1.87735414e-01,  9.89520027e-02,  9.42610997e-02,  0.00000000e+00,
       -4.74286582e-03,  3.51989385e-03, -1.06679404e+00, -1.44361223e-01,
```

(continues on next page)

(continued from previous page)

```
        0.00000000e+00,  0.00000000e+00,  1.13453582e-03,  0.00000000e+00,
        0.00000000e+00,  9.41690572e-02,  0.00000000e+00,  8.94030396e-02,
        0.00000000e+00,  0.00000000e+00, -5.98969457e-02,  1.38721768e-02])
```

[6]:
```
model.p_values
```

[6]:
```
array([4.62571924e-063, 5.33340116e-216, 5.58692822e-002, 7.50873638e-129,
       5.41813130e-001, 9.94065083e-001, 7.50873638e-129, 6.68678693e-094,
       4.69898115e-011, 9.99999953e-001, 3.47337427e-096, 1.00000000e+000,
       9.99559305e-001, 1.00000000e+000, 1.77097129e-119, 1.00000000e+000,
       9.99999988e-001, 9.99985528e-001, 1.00000000e+000, 1.00000000e+000,
       2.63765514e-007, 9.99910617e-001, 1.00000000e+000, 1.00000000e+000,
       3.22315280e-033, 1.00000000e+000, 9.99998180e-001, 9.98226547e-001,
       1.00000000e+000, 7.93160337e-001, 1.00000000e+000, 1.00000000e+000,
       4.72204837e-001, 4.95865112e-102, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       7.69851716e-002, 1.00000000e+000, 4.14070187e-122, 1.00000000e+000,
       9.94240213e-006, 9.99796163e-001, 9.99962949e-001, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 9.59297160e-001,
       1.20662551e-010, 1.00000000e+000, 9.99999988e-001, 1.00000000e+000,
       9.99985528e-001, 9.99910617e-001, 9.99559305e-001, 1.20674890e-003,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 4.77949646e-023,
       9.99994693e-001, 1.00000000e+000, 9.99796163e-001, 1.00000000e+000,
       4.72204837e-001, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.55932686e-032, 2.46381754e-006, 2.00854891e-003, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 2.61910668e-202, 1.81491953e-022,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.28372860e-004, 1.00000000e+000, 4.40268352e-009,
       1.00000000e+000, 1.00000000e+000, 8.81497258e-001, 9.99999953e-001])
```

[7]:
```
model.selected_var
```

[7]:
```
array([ 0.62128221,  2.27730867,  0.        ,  0.7912756 ,  0.        ,
        0.        ,  0.90010748,  0.78141641,  0.23390958,  0.        ,
       -0.37827339,  0.        ,  0.        ,  0.        , -0.65816492,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.16882814,  0.        ,  0.        ,  0.        ,  0.30134417,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.55272368,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.56998959,  0.        , -0.06266557,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        , -0.17217008,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.21165882,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
       -0.18773541,  0.098952  ,  0.        ,  0.        ,  0.        ,
        0.        , -1.06679404, -0.14436122,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.09416906,  0.        ,
        0.08940304,  0.        ,  0.        ,  0.        ,  0.        ])
```

CHAPTER 4

Getting Started

4.1 Data load

[1]:
```
from pyspark.sql.session import SparkSession

spark = SparkSession.builder.getOrCreate()

from pyspark import SparkFiles
from pyspark.sql.functions import col

url_X = 'https://raw.githubusercontent.com/seunghajeong/data_repo/master/simulation_
↪data_x.csv'
url_y = 'https://raw.githubusercontent.com/seunghajeong/data_repo/master/simulation_
↪data_y.csv'
spark.sparkContext.addFile(url_X)
X_Sparkdataframe = spark.read.csv(SparkFiles.get("simulation_data_x.csv"),␣
↪header=True)
X_Sparkdataframe = X_Sparkdataframe.select(*(col(c).cast("float").alias(c) for c in X_
↪Sparkdataframe.columns))
X = X_Sparkdataframe.toPandas()
spark.sparkContext.addFile(url_y)
y_Sparkdataframe = spark.read.csv(SparkFiles.get("simulation_data_y.csv"),␣
↪header=True)
y_Sparkdataframe = y_Sparkdataframe.select(*(col(c).cast("float").alias(c) for c in y_
↪Sparkdataframe.columns))
y = y_Sparkdataframe.toPandas()
```

[3]:
```
X.head()
```

[3]:
```
         V1        V2        V3        V4        V5        V6        V7  \
0  0.101741  0.144909  0.235567  0.576186  0.299443  0.296395  0.902235
1 -0.105054 -0.110128 -0.033311 -0.042925 -0.752605 -0.794815 -1.699739
2 -0.478922 -0.058475 -0.620625  1.775435  0.935760  1.268173 -2.652118
3 -0.657057 -0.508105 -0.556453  0.057045 -0.344814 -0.557824 -0.814844
```

(continues on next page)

(continued from previous page)

```
4 -0.939275 -0.780565 -0.495687 -0.308105 -0.604543 -0.272711  0.080763

         V8        V9       V10  ...       V91       V92       V93       V94  \
0  0.265811  0.420927  0.684045  ... -0.058677  0.497420 -1.124986  0.338215
1 -1.891533 -1.287547 -1.154547  ... -1.614948 -1.337878  0.795742  1.101117
2 -2.327299 -2.913926 -2.140256  ...  0.081620  0.999657 -0.594758  0.057804
3  0.016355 -0.384234 -0.022224  ...  0.077500 -1.489750 -0.151010  0.347814
4  0.542582  0.580669  0.162638  ...  0.677662  0.188278  0.716616 -1.290398

        V95       V96       V97       V98       V99      V100
0 -0.942943 -1.257044 -0.531471  1.236317  0.405682  0.387636
1 -0.920702 -0.098002 -0.269719  0.333092 -0.500367  1.340876
2 -1.259650  0.321864  0.992930  0.552269  1.253700  0.974150
3  2.281268 -1.275026 -0.141539 -0.335557  0.196004 -1.347782
4 -0.579556 -0.692827 -1.040820 -0.674525  1.355343  1.754870

[5 rows x 100 columns]
```

[4]:
```
y.head()
```

[4]:
```
         V1
0  2.811113
1  1.049249
2 -4.496389
3 -3.846408
4 -2.805357
```

4.2 General Usage

[5]:
```
from Hi_LASSO_pyspark import HiLASSO_Spark

model = HiLASSO_Spark(X, y, alpha=0.05, q1='auto', q2='auto', L=30, cv=5, node='auto',
 ↪ logistic=False)
model.fit()
```

```
C:\Users\Seungha\anaconda3\lib\site-packages\sklearn\externals\joblib\__init__.py:15:␣
 ↪FutureWarning: sklearn.externals.joblib is deprecated in 0.21 and will be removed␣
 ↪in 0.23. Please import this functionality directly from joblib, which can be␣
 ↪installed with: pip install joblib. If this warning is raised when loading pickled␣
 ↪models, you may need to re-serialize those models with scikit-learn 0.21+.
  warnings.warn(msg, category=FutureWarning)
C:\Users\Seungha\anaconda3\lib\site-packages\sklearn\externals\six.py:31:␣
 ↪FutureWarning: The module is deprecated in version 0.21 and will be removed in␣
 ↪version 0.23 since we've dropped support for Python 2.7. Please rely on the␣
 ↪official version of six (https://pypi.org/project/six/).
  "(https://pypi.org/project/six/).", FutureWarning)
```

```
Procedure_1_fin.
Procedure_2_fin.
```

[5]:
```
<Hi_LASSO_pyspark.HiLASSO_Spark at 0x172ece75f48>
```

[6]:
```
model.coef_
```

[6]:
```
array([ 1.15905047e+00,  1.73054066e+00, -2.78546358e-01,  7.08538060e-01,
        2.93630644e-01, -3.09802967e-02,  9.87172481e-01,  4.98625094e-01,
        2.80438607e-01,  2.04938322e-03, -5.33703582e-01,  0.00000000e+00,
       -2.25060983e-02,  0.00000000e+00, -5.63347651e-01,  7.20865170e-03,
       -1.05214712e-02,  1.45435365e-02,  0.00000000e+00,  8.20807070e-03,
        2.18642615e-01, -5.03987466e-02,  0.00000000e+00,  7.34933400e-03,
        1.43663963e-01,  0.00000000e+00,  6.34305539e-03,  2.00344988e-02,
        3.13616708e-05,  6.35422045e-02, -2.01727289e-03,  0.00000000e+00,
        4.78273926e-02,  4.29157582e-01,  0.00000000e+00,  0.00000000e+00,
        0.00000000e+00, -1.40783507e-02,  0.00000000e+00,  0.00000000e+00,
        8.87616324e-02,  0.00000000e+00,  3.38209725e-01,  0.00000000e+00,
       -4.18006928e-02,  3.83894299e-02, -3.01721439e-02,  0.00000000e+00,
        0.00000000e+00, -1.45920949e-03,  0.00000000e+00, -3.20830882e-02,
       -1.49872298e-01,  0.00000000e+00, -3.98133525e-03,  0.00000000e+00,
        2.45903335e-03,  7.68685032e-03, -5.65818992e-02,  1.12274264e-01,
        2.13740191e-03,  0.00000000e+00, -4.18373709e-04,  0.00000000e+00,
       -7.87575129e-04,  0.00000000e+00,  4.93090821e-02,  0.00000000e+00,
       -4.18499004e-04, -3.56030163e-04,  3.92035324e-02,  2.35306927e-01,
        1.71041040e-02,  0.00000000e+00,  1.03596675e-02,  0.00000000e+00,
        2.09837938e-02, -2.43313781e-03,  0.00000000e+00,  0.00000000e+00,
       -2.72681876e-01,  1.64381733e-01,  1.56982030e-01,  1.43836815e-03,
        4.66357974e-03,  0.00000000e+00, -1.12394212e+00, -8.07932918e-02,
        0.00000000e+00,  0.00000000e+00, -1.70966580e-03,  0.00000000e+00,
       -2.23018886e-03,  1.26032087e-01,  0.00000000e+00,  1.03304962e-01,
        0.00000000e+00,  0.00000000e+00, -5.00787433e-02,  8.02132494e-03])
```

[7]:
```
model.p_values
```

[7]:
```
array([5.45546950e-131, 2.01625779e-164, 1.31670825e-004, 8.80886752e-097,
       4.31004255e-027, 9.99999051e-001, 8.02834634e-123, 6.37878173e-048,
       6.27334431e-017, 1.00000000e+000, 1.70639215e-121, 1.00000000e+000,
       9.99733947e-001, 1.00000000e+000, 6.12580122e-098, 1.00000000e+000,
       1.00000000e+000, 9.99999704e-001, 1.00000000e+000, 1.00000000e+000,
       1.31517681e-007, 8.32544869e-001, 1.00000000e+000, 1.00000000e+000,
       2.85508681e-007, 1.00000000e+000, 1.00000000e+000, 9.99948594e-001,
       1.00000000e+000, 5.20943089e-003, 1.00000000e+000, 1.00000000e+000,
       1.85618624e-002, 3.49112719e-075, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       4.16256090e-004, 1.00000000e+000, 1.28847785e-054, 1.00000000e+000,
       5.31842478e-001, 2.19719125e-001, 9.81054785e-001, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 9.34796624e-001,
       2.85508681e-007, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.01690916e-001, 5.94884253e-008,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 8.73984401e-001, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 9.70377124e-001, 7.00544469e-023,
       9.99948594e-001, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       9.99997164e-001, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       2.73297075e-068, 8.03655110e-012, 1.39764626e-010, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 4.97483677e-211, 2.64252727e-008,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.39764626e-010, 1.00000000e+000, 4.94110926e-009,
       1.00000000e+000, 1.00000000e+000, 9.93078320e-001, 1.00000000e+000])
```

[8]:
```
model.selected_var
```

[8]:
```
array([ 1.15905047,  1.73054066, -0.27854636,  0.70853806,  0.29363064,
        0.        ,  0.98717248,  0.49862509,  0.28043861,  0.        ,
```

(continues on next page)

(continued from previous page)

```
       -0.53370358,  0.        ,  0.        ,  0.        , -0.56334765,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.21864261,  0.        ,  0.        ,  0.        ,  0.14366396,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.42915758,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.08876163,  0.        ,  0.33820973,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        , -0.1498723 ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.11227426,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.23530693,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
       -0.27268188,  0.16438173,  0.15698203,  0.        ,  0.        ,
        0.        , -1.12394212, -0.08079329,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.12603209,  0.        ,
        0.10330496,  0.        ,  0.        ,  0.        ,  0.        ])
```

CHAPTER 5

Getting Started

5.1 Data load

[1]:
```
#Import package
import numpy as np
import pandas as pd

from pyspark.sql.session import SparkSession

spark = SparkSession.builder.getOrCreate()
```

[2]:
```
#If using .json file
from pyspark.sql.functions import *
from pyspark.sql.types import *
sch = StructType([StructField('V1',DoubleType()), StructField('V2',DoubleType()),␣
↪StructField('V3',DoubleType()), StructField('V4',DoubleType()), StructField('V5',
↪DoubleType()), StructField('V6',DoubleType()), StructField('V7',DoubleType()),␣
↪StructField('V8',DoubleType()), StructField('V9',DoubleType()), StructField('V10',
↪DoubleType()), StructField('V11',DoubleType()), StructField('V12',DoubleType()),␣
↪StructField('V13',DoubleType()), StructField('V14',DoubleType()), StructField('V15',
↪DoubleType()), StructField('V16',DoubleType()), StructField('V17',DoubleType()),␣
↪StructField('V18',DoubleType()), StructField('V19',DoubleType()), StructField('V20',
↪DoubleType()), StructField('V21',DoubleType()), StructField('V22',DoubleType()),␣
↪StructField('V23',DoubleType()), StructField('V24',DoubleType()), StructField('V25',
↪DoubleType()), StructField('V26',DoubleType()), StructField('V27',DoubleType()),␣
↪StructField('V28',DoubleType()), StructField('V29',DoubleType()), StructField('V30',
↪DoubleType()), StructField('V31',DoubleType()), StructField('V32',DoubleType()),␣
↪StructField('V33',DoubleType()), StructField('V34',DoubleType()), StructField('V35',
↪DoubleType()), StructField('V36',DoubleType()), StructField('V37',DoubleType()),␣
↪StructField('V38',DoubleType()), StructField('V39',DoubleType()), StructField('V40',
↪DoubleType()), StructField('V41',DoubleType()), StructField('V42',DoubleType()),␣
↪StructField('V43',DoubleType()), StructField('V44',DoubleType()), StructField('V45',
↪DoubleType()), StructField('V46',DoubleType()), StructField('V47',DoubleType()),␣
↪StructField('V48',DoubleType()), StructField('V49',DoubleType()), StructField('V50',
↪DoubleType()), StructField('V51',DoubleType()), StructField('V52',DoubleType()),␣
↪StructField('V53',DoubleType()), StructField('V54',DoubleType()), StructField('V55',
↪DoubleType()), StructField('V56',DoubleType()), StructField('V57',DoubleType()),␣
↪StructField('V58',DoubleType()), StructField('V59',DoubleType()), StructField('V60',
↪DoubleType()), StructField('V61',DoubleType()), StructField('V62',DoubleType()),␣
↪StructField('V63',DoubleType()), StructField('V64',DoubleType()), StructField('V65',
↪DoubleType()), StructField('V66',DoubleType()), StructField('V67',DoubleType()),␣
```

(continues on next page)

(continued from previous page)

```
])
```

[3]:
```
X_Sparkdataframe = spark.read.schema(sch).json("simulation_data_x.json")
X_Sparkdataframe = X_Sparkdataframe.select(*(col(c).cast("float").alias(c) for c in X_
↪Sparkdataframe.columns))
X = X_Sparkdataframe.toPandas()
y_Sparkdataframe = spark.read.json("simulation_data_y.json")
y_Sparkdataframe = y_Sparkdataframe.select(*(col(c).cast("float").alias(c) for c in y_
↪Sparkdataframe.columns))
y = y_Sparkdataframe.toPandas()
```

[4]:
```
X.head()
```

[4]:

	V1	V2	V3	V4	V5	V6	V7	\
0	0.101741	0.144909	0.235567	0.576186	0.299443	0.296395	0.902235	
1	-0.105054	-0.110128	-0.033311	-0.042925	-0.752605	-0.794815	-1.699739	
2	-0.478922	-0.058475	-0.620625	1.775435	0.935760	1.268173	-2.652118	
3	-0.657057	-0.508105	-0.556453	0.057045	-0.344814	-0.557824	-0.814844	
4	-0.939275	-0.780565	-0.495687	-0.308105	-0.604543	-0.272711	0.080763	

	V8	V9	V10	...	V91	V92	V93	V94	\
0	0.265811	0.420927	0.684045	...	-0.058677	0.497420	-1.124986	0.338215	
1	-1.891533	-1.287547	-1.154547	...	-1.614948	-1.337878	0.795742	1.101117	
2	-2.327299	-2.913926	-2.140256	...	0.081620	0.999657	-0.594758	0.057804	
3	0.016355	-0.384234	-0.022224	...	0.077500	-1.489750	-0.151010	0.347814	
4	0.542582	0.580669	0.162638	...	0.677662	0.188278	0.716616	-1.290398	

	V95	V96	V97	V98	V99	V100
0	-0.942943	-1.257044	-0.531471	1.236317	0.405682	0.387636
1	-0.920702	-0.098002	-0.269719	0.333092	-0.500367	1.340876
2	-1.259650	0.321864	0.992930	0.552269	1.253700	0.974150
3	2.281268	-1.275026	-0.141539	-0.335557	0.196004	-1.347782
4	-0.579556	-0.692827	-1.040820	-0.674525	1.355343	1.754870

[5 rows x 100 columns]

[5]:
```
y.head()
```

[5]:

	V1
0	2.811113
1	1.049249
2	-4.496389
3	-3.846408
4	-2.805357

5.2 General Usage

[6]:
```
from Hi_LASSO_pyspark import HiLASSO_Spark

model = HiLASSO_Spark(X, y, alpha=0.05, q1='auto', q2='auto', L=30, cv=5, node='auto',
↪ logistic=False)
model.fit()
```

```
C:\Users\Seungha\anaconda3\lib\site-packages\sklearn\externals\joblib\__init__.py:15:␣
↪FutureWarning: sklearn.externals.joblib is deprecated in 0.21 and will be removed␣
↪in 0.23. Please import this functionality directly from joblib, which can be␣
↪installed with: pip install joblib. If this warning is raised when loading pickled␣
↪models, you may need to re-serialize those models with scikit-learn 0.21+.
  warnings.warn(msg, category=FutureWarning)
C:\Users\Seungha\anaconda3\lib\site-packages\sklearn\externals\six.py:31:␣
↪FutureWarning: The module is deprecated in version 0.21 and will be removed in␣
↪version 0.23 since we've dropped support for Python 2.7. Please rely on the␣
↪official version of six (https://pypi.org/project/six/).
  "(https://pypi.org/project/six/).", FutureWarning)
```

```
Procedure_1_fin.
Procedure_2_fin.
```

[6]:
```
<Hi_LASSO_pyspark.HiLASSO_Spark at 0x1b7628c91c8>
```

[7]:
```
model.coef_
```

[7]:
```
array([ 9.07767822e-01,  1.84831819e+00, -1.50886668e-01,  6.51613666e-01,
        3.60881908e-01, -5.43158333e-02,  1.10656217e+00,  5.76287223e-01,
        1.45350133e-01,  1.12891593e-02, -4.92306272e-01,  4.00331044e-03,
       -2.92755363e-02,  0.00000000e+00, -5.97072304e-01, -4.01800075e-03,
       -1.85472135e-02,  1.50973881e-02,  0.00000000e+00,  1.00734445e-03,
        1.82650706e-01, -2.44372503e-02,  0.00000000e+00,  4.87803321e-04,
        2.12434938e-01, -1.35364354e-03,  6.26210409e-03,  7.30404076e-02,
        0.00000000e+00,  4.32746612e-02,  6.84953978e-04,  0.00000000e+00,
        5.08000749e-02,  4.16104086e-01,  0.00000000e+00,  0.00000000e+00,
        6.12355843e-05, -2.15697968e-03,  0.00000000e+00,  0.00000000e+00,
        8.90277566e-02,  0.00000000e+00,  4.82355939e-01,  0.00000000e+00,
       -6.06668658e-02,  4.24659000e-02, -1.41782146e-02, -3.61172307e-03,
        0.00000000e+00, -1.65201959e-04,  0.00000000e+00, -1.42850442e-02,
       -1.45138412e-01,  0.00000000e+00,  9.37401342e-05,  0.00000000e+00,
        4.69654532e-03,  2.33781070e-02, -1.21564959e-01,  1.19742376e-01,
        4.37066230e-03,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       -7.42586867e-03, -7.69866705e-04,  4.79935847e-03, -8.54272613e-03,
       -4.44192587e-04,  0.00000000e+00,  6.32937349e-03,  1.72394695e-01,
        1.89504145e-02,  0.00000000e+00,  4.31866689e-03,  0.00000000e+00,
        2.51616680e-02,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       -2.12432332e-01,  1.61906818e-01,  1.70911957e-01,  0.00000000e+00,
        5.26630954e-04,  0.00000000e+00, -1.13084992e+00, -8.78953629e-02,
       -2.32628707e-03,  0.00000000e+00,  3.15969130e-03,  0.00000000e+00,
       -6.36326525e-04,  1.61205647e-01,  0.00000000e+00,  5.21366390e-02,
        2.89292450e-03,  0.00000000e+00, -5.86326872e-02,  2.83126071e-02])
```

[8]:
```
model.p_values
```

[8]:
```
array([7.53704419e-089, 1.50423843e-183, 7.54534150e-001, 9.43249973e-088,
       2.82878036e-034, 9.98485520e-001, 7.92473484e-147, 3.05039555e-068,
       6.11829481e-003, 1.00000000e+000, 1.65138684e-117, 1.00000000e+000,
       9.45720852e-001, 1.00000000e+000, 4.57443521e-091, 1.00000000e+000,
       1.00000000e+000, 9.99999998e-001, 1.00000000e+000, 1.00000000e+000,
       1.27876186e-005, 9.99999540e-001, 1.00000000e+000, 1.00000000e+000,
       5.63291884e-020, 1.00000000e+000, 1.00000000e+000, 1.13863358e-001,
       1.00000000e+000, 7.54534150e-001, 1.00000000e+000, 1.00000000e+000,
       5.62464251e-001, 3.12847085e-079, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       5.00176690e-004, 1.00000000e+000, 1.90247524e-094, 1.00000000e+000,
       2.85225204e-004, 6.30455341e-001, 9.99233039e-001, 1.00000000e+000,
```

(continues on next page)

(continued from previous page)

```
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 9.99998548e-001,
       1.63646066e-007, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 9.99999865e-001, 4.69500947e-005, 6.18737269e-009,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 9.99999998e-001,
       1.00000000e+000, 1.00000000e+000, 1.00000000e+000, 1.75747652e-010,
       9.94847160e-001, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       9.99988256e-001, 1.00000000e+000, 1.00000000e+000, 1.00000000e+000,
       1.47539558e-054, 7.77001029e-016, 1.38617488e-012, 1.00000000e+000,
       9.99969672e-001, 1.00000000e+000, 4.20762516e-215, 1.44173110e-008,
       1.00000000e+000, 1.00000000e+000, 9.99999998e-001, 1.00000000e+000,
       1.00000000e+000, 2.37906131e-015, 1.00000000e+000, 9.49056045e-003,
       1.00000000e+000, 1.00000000e+000, 3.58419958e-001, 9.99995730e-001])
```

[9]:
```
model.selected_var
```

[9]:
```
array([ 0.90776782,  1.84831819,  0.        ,  0.65161367,  0.36088191,
        0.        ,  1.10656217,  0.57628722,  0.        ,  0.        ,
       -0.49230627,  0.        ,  0.        ,  0.        , -0.5970723 ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.18265071,  0.        ,  0.        ,  0.        ,  0.21243494,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.41610409,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.48235594,  0.        , -0.06066687,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        , -0.14513841,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        , -0.12156496,  0.11974238,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.17239469,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
       -0.21243233,  0.16190682,  0.17091196,  0.        ,  0.        ,
        0.        , -1.13084992, -0.08789536,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.16120565,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ])
```

CHAPTER 6

What is Hi-LASSO?

Hi-LASSO(High-Dimensional LASSO) can theoretically improves a LASSO model providing better performance of both prediction and feature selection on extremely high-dimensional data. Hi-LASSO alleviates bias introduced from bootstrapping, refines importance scores, improves the performance taking advantage of global oracle property, provides a statistical strategy to determine the number of bootstrapping, and allows tests of significance for feature selection with appropriate distribution. In Hi-LASSO will be applied to use the pool of the python library to process parallel multiprocessing to reduce the time required for the model.

CHAPTER 7

Credit

Hi-LASSO was primarily developed by Dr. Youngsoon Kim, with significant contributions and suggestions by Dr. Joongyang Park, Dr. Mingon Kang, and many others. The pyspark package was developed by Seungha Jeong. Initial supervision for the project was provided by Dr. Mingon Kang.

Development of Hi-LASSO is carried out in the DataX lab lab at University of Nevada, Las Vegas (UNLV).

If you use Hi-LASSO in your research, generally it is appropriate to cite the following paper: Y. Kim, J. Hao, T. Mallavarapu, J. Park and M. Kang, "Hi-LASSO: High-Dimensional LASSO," in IEEE Access, vol. 7, pp. 44562-44573, 2019, doi: 10.1109/ACCESS.2019.2909071.

Python Module Index

h

Index