# Final Project

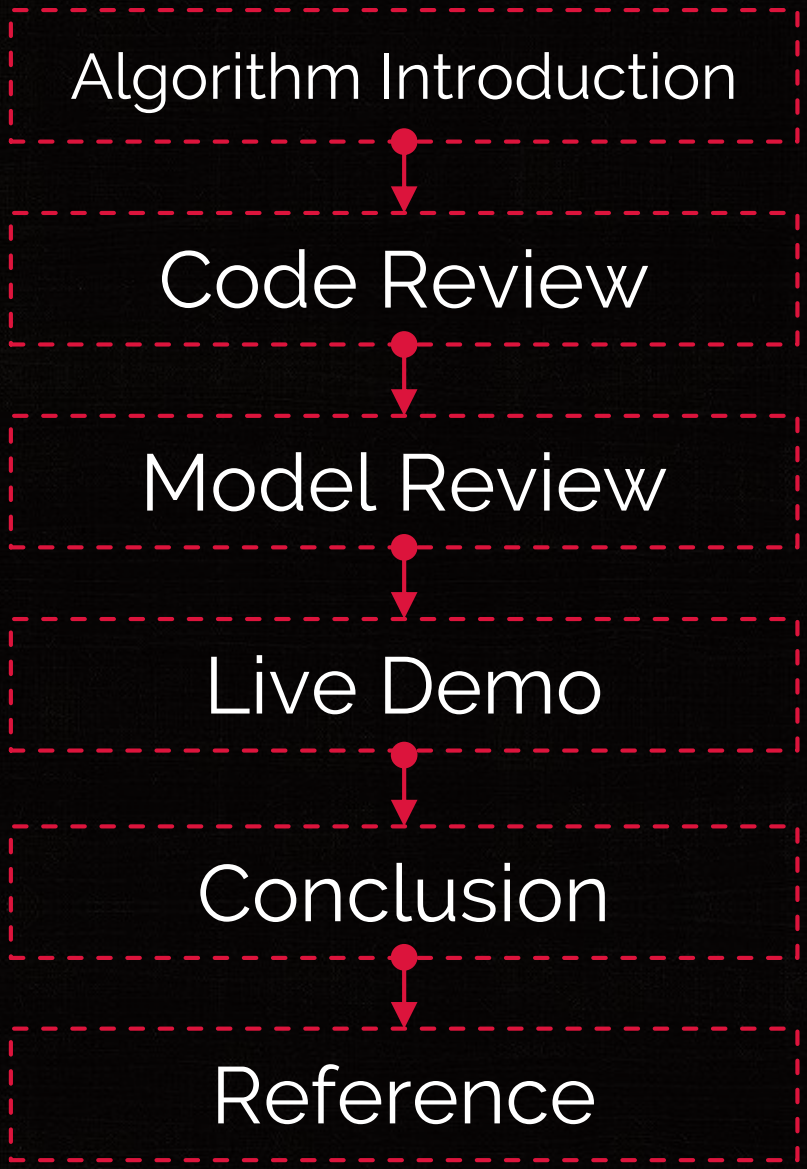
# Lasso regression

hello!

# I AM BERRIE CHEN

Name: 陳昱芝
ID: r06525089
抽籤編號: 10[演算法文件]
go40631@gmail.com

# Outline


Algorithm Introduction
Code Review
Model Review
Live Demo
Conclusion
Reference

# 1. Algorithm Introduction

# Algorithm Introduction

LASSO (least absolute shrinkage and selection operator):

- A regularization regression method.
- Perform both feature selection and regularization.

$$\min_w \frac{1}{2n_{samples}}||Xw - y||_2^2 + \alpha||w||_1$$

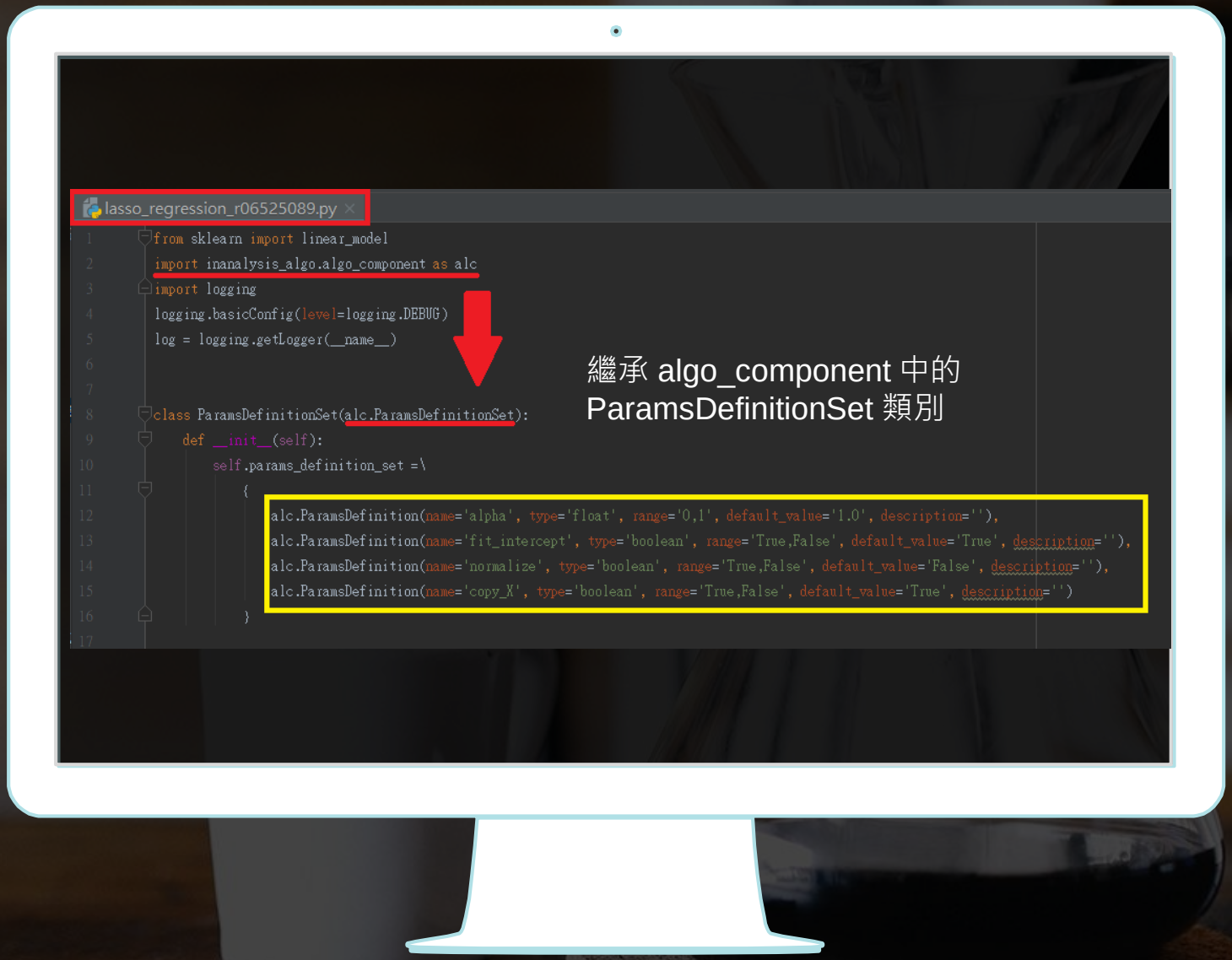
最小平方和　　正規化 (regularizer)

- Trained with L1 prior as regularizer.

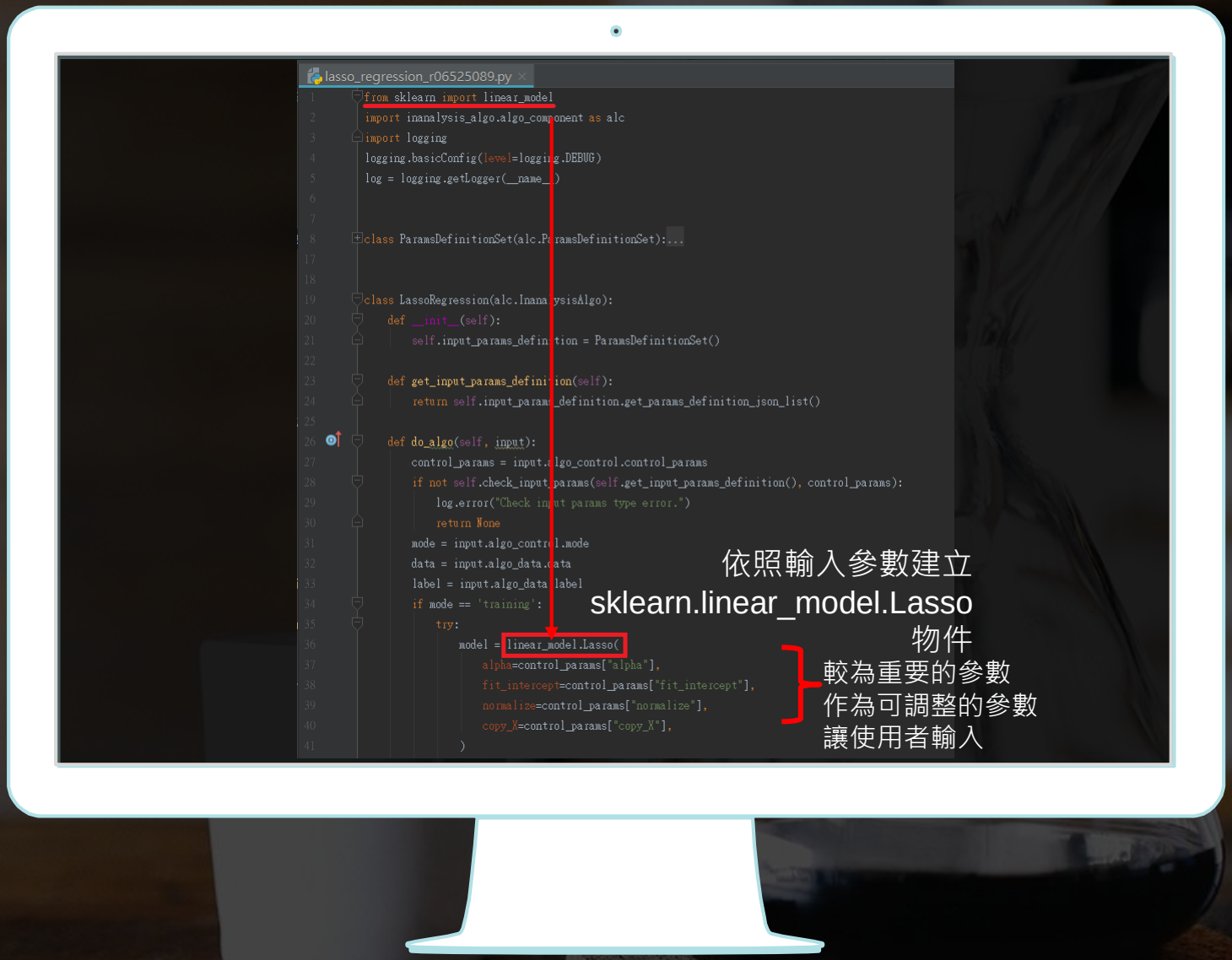
# 2. Code Review


algo_component.py
import ...
logging.basicConfig(level=logging.DEBUG)
log = logging.getLogger(__name__)
class ParamsDefinition:
def __init__(self, name, type, range, default_value, description):
self.name = name
self.type = type
self.range = range
self.default_value = default_value
self.description = description
def get_params_definition(self):
return self.__dict__
class ParamsDefinitionSet:
def __init__(self):
self.params_definition_set = []
raise NotImplementedError
def get_params_definition_set(self):
definition_set_json_list = []
for params_object in self.params_definition_set:
definition_set_json_list.append(params_object.get_params_definition())
return definition_set_json_list
在各演算法
子類別中實作

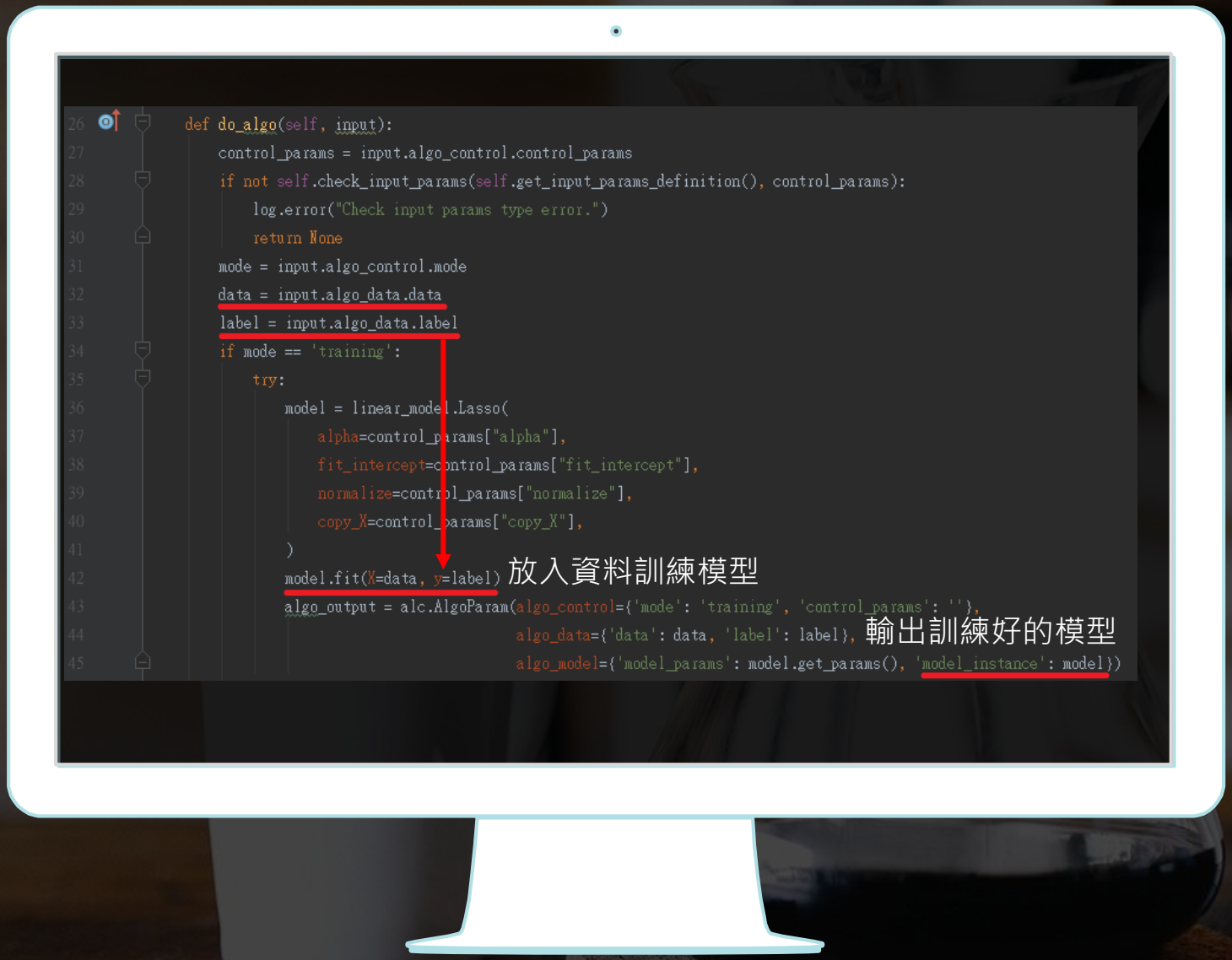
Code Review


lasso_regression_r06525089.py
from sklearn import linear_model
import inanalysis_algo.algo_component as alc
import logging
logging.basicConfig(level=logging.DEBUG)
log = logging.getLogger(__name__)
class ParamsDefinitionSet(alc.ParamsDefinitionSet):
def __init__(self):
self.params_definition_set =\
{
alc.ParamsDefinition(name='alpha', type='float', range='0,1', default_value='1.0', description=''),
alc.ParamsDefinition(name='fit_intercept', type='boolean', range='True,False', default_value='True', description=''),
alc.ParamsDefinition(name='normalize', type='boolean', range='True,False', default_value='False', description=''),
alc.ParamsDefinition(name='copy_X', type='boolean', range='True,False', default_value='True', description='')
}
繼承 algo_component 中的
ParamsDefinitionSet 類別

# Code Review


lasso_regression_r06525089.py
from sklearn import linear_model
import inanalysis_algo.algo_component as alc
import logging
logging.basicConfig(level=logging.DEBUG)
log = logging.getLogger(__name__)
class ParamsDefinitionSet(alc.ParamsDefinitionSet):...
class LassoRegression(alc.InanalysisAlgo):
def __init__(self):
self.input_params_definition = ParamsDefinitionSet()
def get_input_params_definition(self):
return self.input_params_definition.get_params_definition_json_list()
def do_algo(self, input):
control_params = input.algo_control.control_params
if not self.check_input_params(self.get_input_params_definition(), control_params):
log.error("Check input params type error.")
return None
mode = input.algo_control.mode
data = input.algo_data.data
label = input.algo_data.label
if mode == 'training':
try:
model = linear_model.Lasso(
alpha=control_params["alpha"],
fit_intercept=control_params["fit_intercept"],
normalize=control_params["normalize"],
copy_X=control_params["copy_X"],
)
依照輸入參數建立
sklearn.linear_model.Lasso
物件
較為重要的參數
作為可調整的參數
讓使用者輸入

# Code Review


def do_algo(self, input):
control_params = input.algo_control.control_params
if not self.check_input_params(self.get_input_params_definition(), control_params):
log.error("Check input params type error.")
return None
mode = input.algo_control.mode
data = input.algo_data.data
label = input.algo_data.label
if mode == 'training':
try:
model = linear_model.Lasso(
alpha=control_params["alpha"],
fit_intercept=control_params["fit_intercept"],
normalize=control_params["normalize"],
copy_X=control_params["copy_X"],
)
model.fit(X=data, y=label)
algo_output = alc.AlgoParam(algo_control={'mode': 'training', 'control_params': ''},
algo_data={'data': data, 'label': label},
algo_model={'model_params': model.get_params(), 'model_instance': model})
放入資料訓練模型
輸出訓練好的模型

# Code Review


def test_correct_lasso_regression_parameter_type(self):
# given: collect input parameter, create algorithm object
arg_dict = {
"alpha": 1.0,
"fit_intercept": True,
"normalize": False,
"copy_X": True
}
algo_name = 'lasso'
algo_input = alc.AlgoParam(algo_control={'mode': 'training', 'control_params': arg_dict},
algo_data={'data': self.boston_data, 'label': self.boston_label},
algo_model={'model_params': None, 'model_instance': None})
in_algo = AlgoUtils.algo_factory(algo_name)
input_params_definition = in_algo.get_input_params_definition()
# when: checkout input type
check_result = in_algo.check_input_params(input_params_definition, algo_input.algo_control.control_params)
# then: type match
self.assertTrue(check_result is True)
self.assertEqual(Algorithm.get_project_type(algo_name), "regression")
參數測試 Happy Path Test

# Code Review


def test_error_lasso_regression_parameter_alpha_string_type(self):
# given: error input parameter "alpha" needs to be type(float)
arg_dict = {
"alpha": 'string',
"fit_intercept": True,
"normalize": False,
"copy_X": True
}
algo_input = alc.AlgoParam(algo_control={'mode': 'training', 'control_params': arg_dict},
algo_data={'data': self.boston_data, 'label': self.boston_label},
algo_model={'model_params': None, 'model_instance': None})
in_algo = AlgoUtils.algo_factory('lasso')
input_params_definition = in_algo.get_input_params_definition()
# when: checkout input type
check_result = in_algo.check_input_params(input_params_definition, algo_input.algo_control.control_params)
# then: type match
self.assertTrue(check_result is False)
參數測試 Sad Path Test

Code Review


def test_error_lasso_regression_parameter_fit_intercept_string_type(self):
# given: error input parameter "fit_intercept" needs to be type(boolean)
arg_dict = {
"alpha": 1.0,
"fit_intercept": 'string',
"normalize": False,
"copy_X": True
}
algo_input = alc.AlgoParam(algo_control={'mode': 'training', 'control_params': arg_dict},
algo_data={'data': self.boston_data, 'label': self.boston_label},
algo_model={'model_params': None, 'model_instance': None})
in_algo = AlgoUtils.algo_factory('lasso')
input_params_definition = in_algo.get_input_params_definition()
# when: checkout input type
check_result = in_algo.check_input_params(input_params_definition, algo_input.algo_control.control_params)
# then: type match
self.assertTrue(check_result is False)
參數測試 Sad Path Test

Code Review


def test_error_lasso_regression_parameter_fit_intercept_float_type(self):
# given: error input parameter "fit_intercept" needs to be type(boolean)
arg_dict = {
"alpha": 1.0,
"fit_intercept": 1.0,
"normalize": False,
"copy_X": True
}
algo_input = alc.AlgoParam(algo_control={'mode': 'training', 'control_params': arg_dict},
algo_data={'data': self.boston_data, 'label': self.boston_label},
algo_model={'model_params': None, 'model_instance': None})
in_algo = AlgoUtils.algo_factory('lasso')
input_params_definition = in_algo.get_input_params_definition()
# when: checkout input type
check_result = in_algo.check_input_params(input_params_definition, algo_input.algo_control.control_params)
# then: type match
self.assertTrue(check_result is False)
參數測試 Sad Path Test

# Code Review

```
118     def test_correct_lasso_regression_do_algo(self):
119         # given: collect input parameter, create algorithm object
120         arg_dict = {
121             "alpha": 1.0,
122             "fit_intercept": True,
123             "normalize": False,
124             "copy_X": True
125         }
126         algo_input = alc.AlgoParam(algo_control={'mode': 'training', 'control_params': arg_dict},
127                                    algo_data={'data': self.boston_data, 'label': self.boston_label},
128                                    algo_model={'model_params': None, 'model_instance': None})
129         in_algo = AlgoUtils.algo_factory('lasso')
130         log.debug(algo_input)
131         # when: do decision tree algorithm
132         algo_output = in_algo.do_algo(algo_input)
133
134         # then:
135         self.assertTrue(algo_output is not None)
136         self.assertTrue(algo_output.algo_model.model_instance is not None)
137         self.assertTrue(algo_output.algo_model.model_instance.predict(X=self.boston_data) is not None)
138         #self.assertTrue(algo_output.algo_model.model_instance.score(X=self.boston_data, y=self.boston_label) is float)
```

演算法測試 Happy Path Test

模型訓練成功

用訓練好的模型作預測

Sklearn官方文件
Lasso之Methods

**Methods**

| |
|---|
| `fit` (X, y[, check_input]) |
| `get_params` ([deep]) |
| `path` (X, y[, l1_ratio, eps, n_alphas, ...]) |
| `predict` (X) |
| `score` (X, y[, sample_weight]) |
| `set_params` (**params) |

Code Review

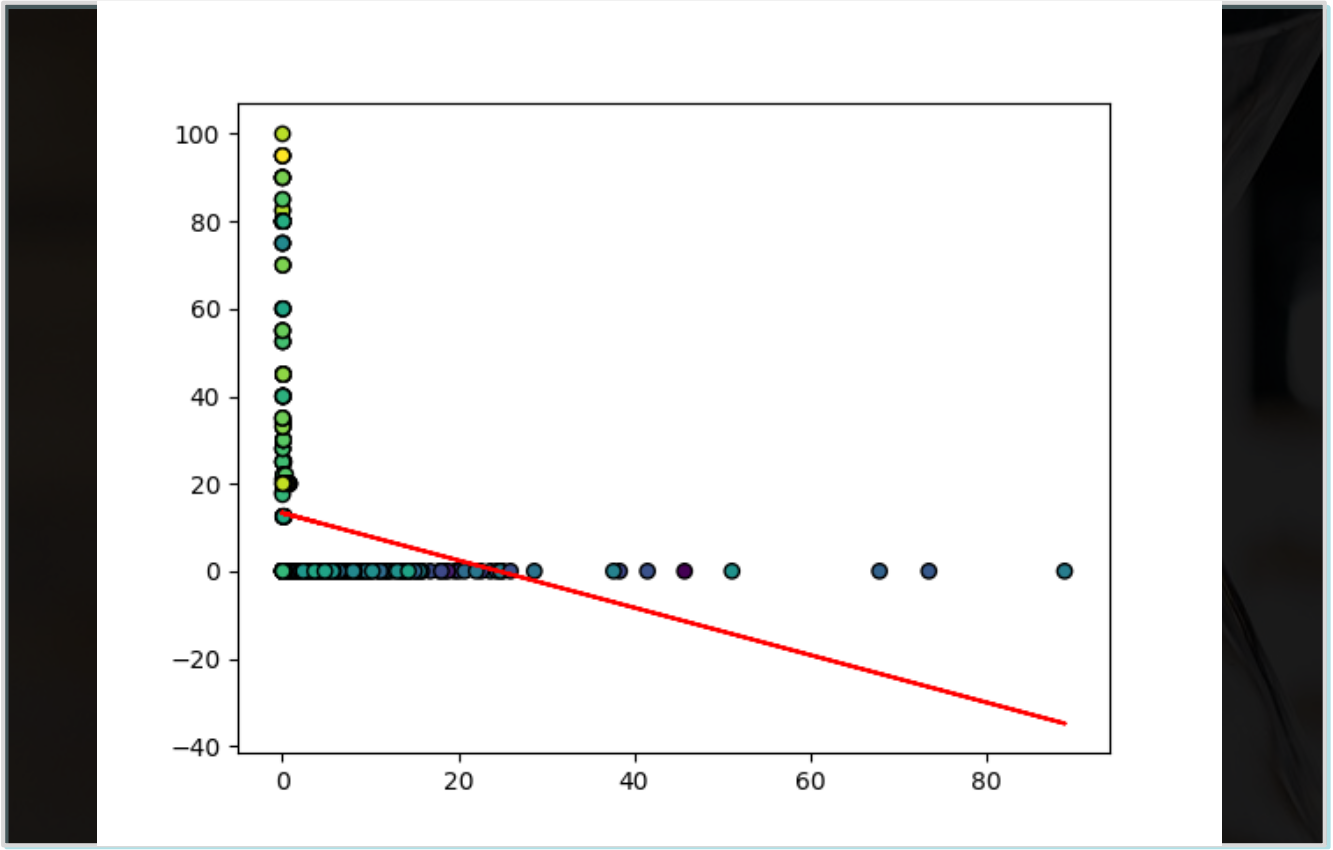
# 3. Model Review

```python
import matplotlib.pyplot as plt
import inanalysis_algo.algo_component as alc
from inanalysis_algo.utils import AlgoUtils
from sklearn.datasets import load_boston
import pandas as pd
import numpy as np


def lasso_model_preview(data, predict_result, x_axis_name, y_axis_name):
    plt.title("Boston model plot")
    plt.xlabel(x_axis_name)
    plt.ylabel(y_axis_name)
    fig, ax = plt.subplots()
    fit = np.polyfit(data[x_axis_name], data[y_axis_name], deg=1)
    ax.plot(data[x_axis_name], fit[0] * data[x_axis_name] + fit[1], color='red')
    ax.scatter(data[x_axis_name], data[y_axis_name], c=predict_result, edgecolor='black', linewidth='1')
    plt.savefig(x_axis_name + " _ " + y_axis_name + ".png")
    plt.close('all')
```

Regression line

# Model Review




# Model Review

# 4. Live Demo

http://ntuesoe.com:8008/

# 5. Conclusion

# What I have learned:

- 運用套件的method
- 寫單元測試(Unit Test)
- survey使用工具背後的相關技術
- 容易又快速的開發演算法程式

# 6. Reference

# Reference


http://scikit-learn.org/stable/modules/linear_model.html#lasso

https://dotblogs.com.tw/dash_analysis/2017/11/03/161734


thanks!
ANY
QUESTIONS?