期末報告--
[Ridge Regression]

學號：R05525055

姓名：劉立晨

指導老師：張瑞益 博士

2018 / 01 / 10

Outline

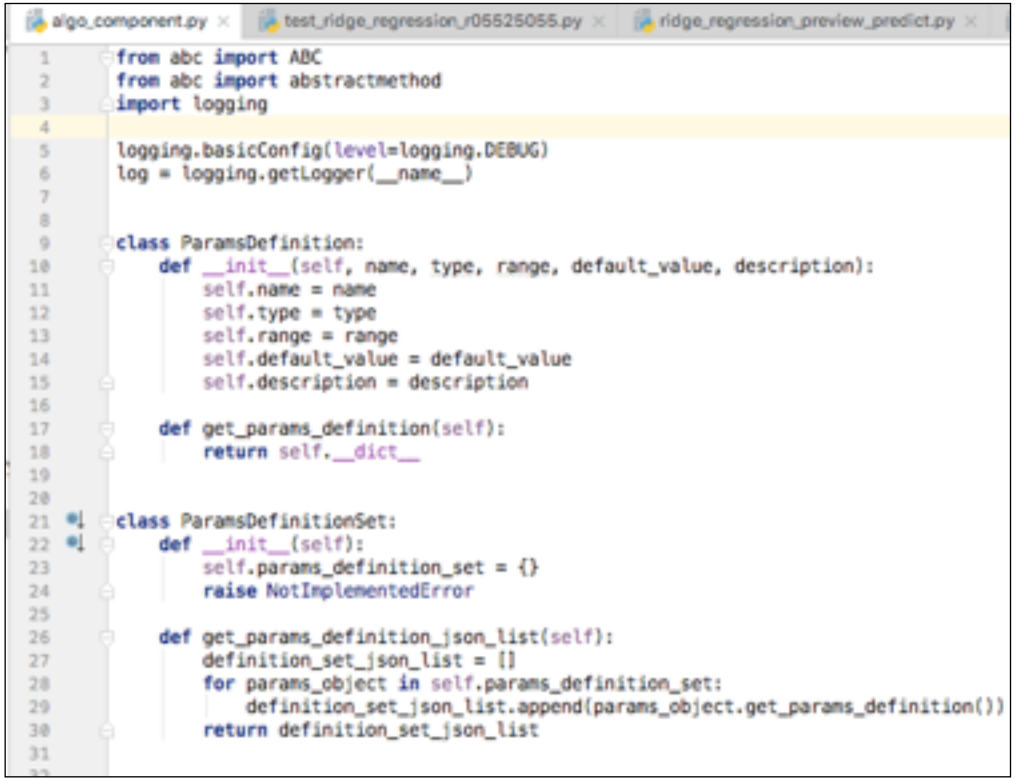
Algorithm Introduction

Ridge Regression

Ridge regression 通過收縮參數 λ 解決 overfitting 問題，達到了減少模型誤差的效果。透過控制 λ 的大小得以調節線性基函數模型的複雜性，λ 越大表示模型的複雜性越低。懲罰係數 λ 不屬於模型參數，其數值通常由交叉驗證 (cross validation) 程序決定以建立精確性最高的模型並避免 overfitting 現象發生。

參考資料：http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html

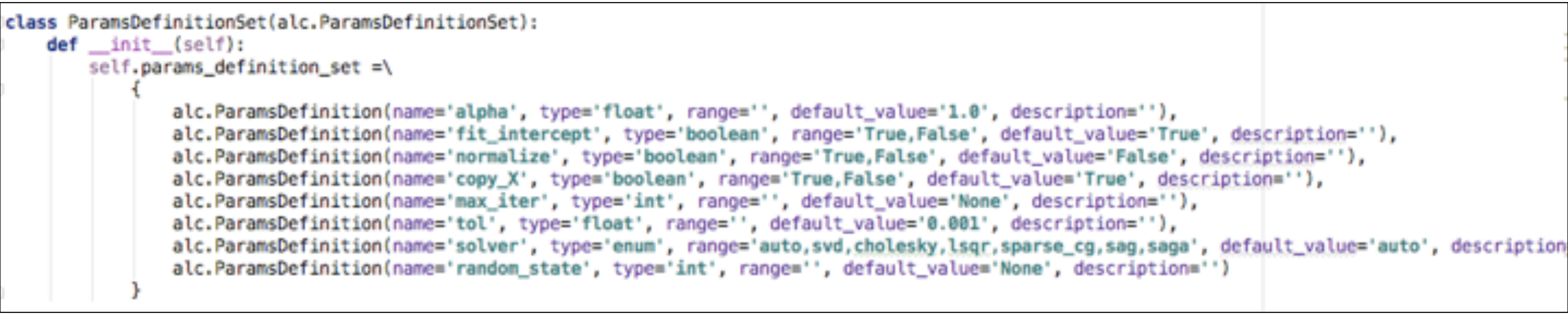
Code Review



algo_component.py | test_ridge_regression_r05525055.py | ridge_regression_preview_predict.py

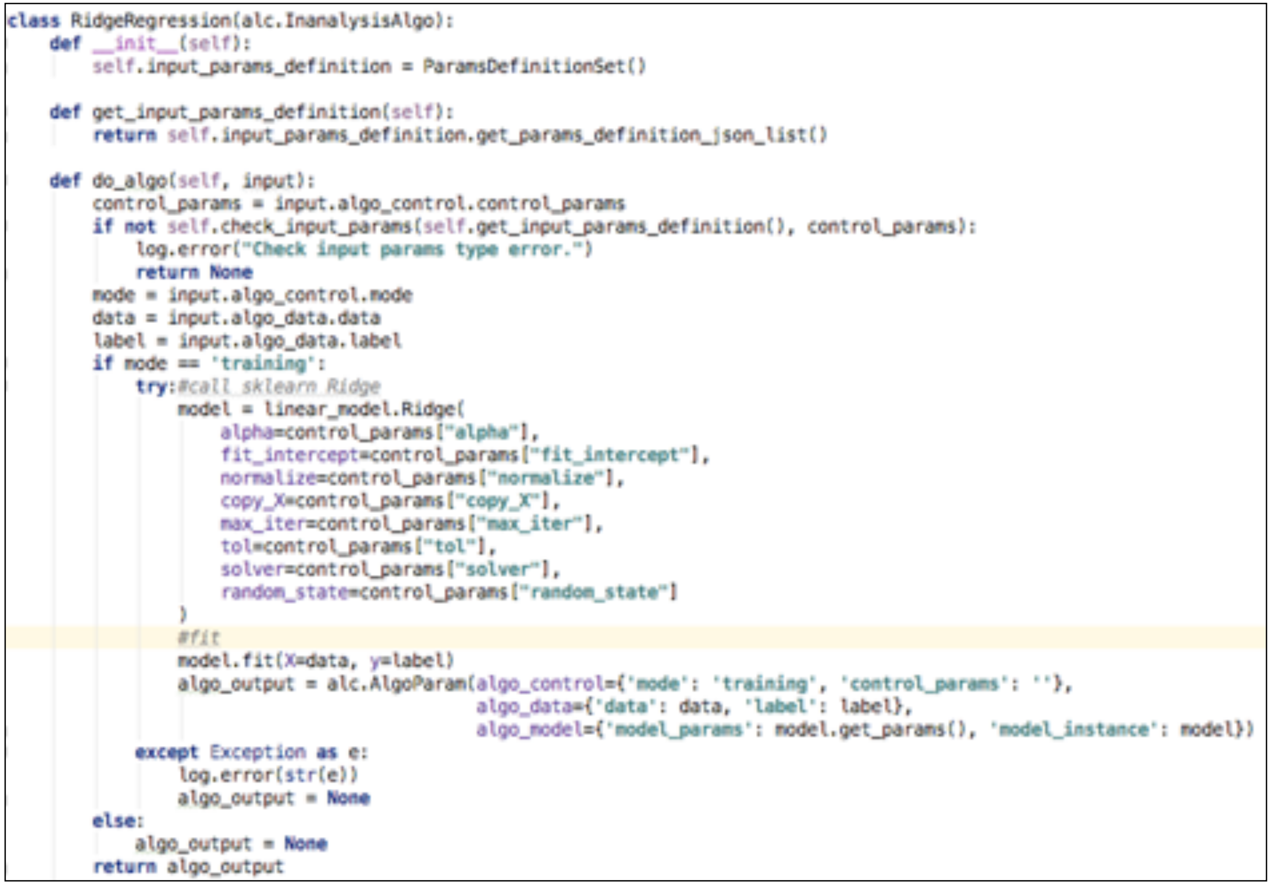
```python
from abc import ABC
from abc import abstractmethod
import logging

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_json_list(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

```

Code Review



```
class ParamsDefinitionSet(alc.ParamsDefinitionSet):
    def __init__(self):
        self.params_definition_set =\
            {
                alc.ParamsDefinition(name='alpha', type='float', range='', 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=''),
                alc.ParamsDefinition(name='max_iter', type='int', range='', default_value='None', description=''),
                alc.ParamsDefinition(name='tol', type='float', range='', default_value='0.001', description=''),
                alc.ParamsDefinition(name='solver', type='enum', range='auto,svd,cholesky,lsqr,sparse_cg,sag,saga', default_value='auto', description
                alc.ParamsDefinition(name='random_state', type='int', range='', default_value='None', description='')
            }
```

Code Review



```python
class RidgeRegression(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:#call sklearn Ridge
                model = linear_model.Ridge(
                    alpha=control_params["alpha"],
                    fit_intercept=control_params["fit_intercept"],
                    normalize=control_params["normalize"],
                    copy_X=control_params["copy_X"],
                    max_iter=control_params["max_iter"],
                    tol=control_params["tol"],
                    solver=control_params["solver"],
                    random_state=control_params["random_state"]
                )
                #fit
                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})
            except Exception as e:
                log.error(str(e))
                algo_output = None
        else:
            algo_output = None
        return algo_output
```

Data set: sklearn.datasets load_boston
Data pre-processing: filter missing value
Feature selecting: all features(房子的價格、成交日期、臥房數目、浴室數目...)
Label: price(房價)

	linear regression	Ridge, alpha = 0.1	Ridge, alpha = 0.01	Ridge, alpha = 0.001	Ridge, alpha = 0.0001	Ridge, alpha = 0.0000001
training MSE	3.11349699	3.13	3.11239414	3.11199355	3.11198925	3.11198925
prediction MSE	5.722222	5.81	5.7196368	5.7113342	5.7105351	5.7104466

alpha: Regularization strength. Regularization improves the conditioning of the problem and reduces the variance of the estimates. Larger values specify stronger regularization.

參考資料：http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html

**Bonus: Model Preview

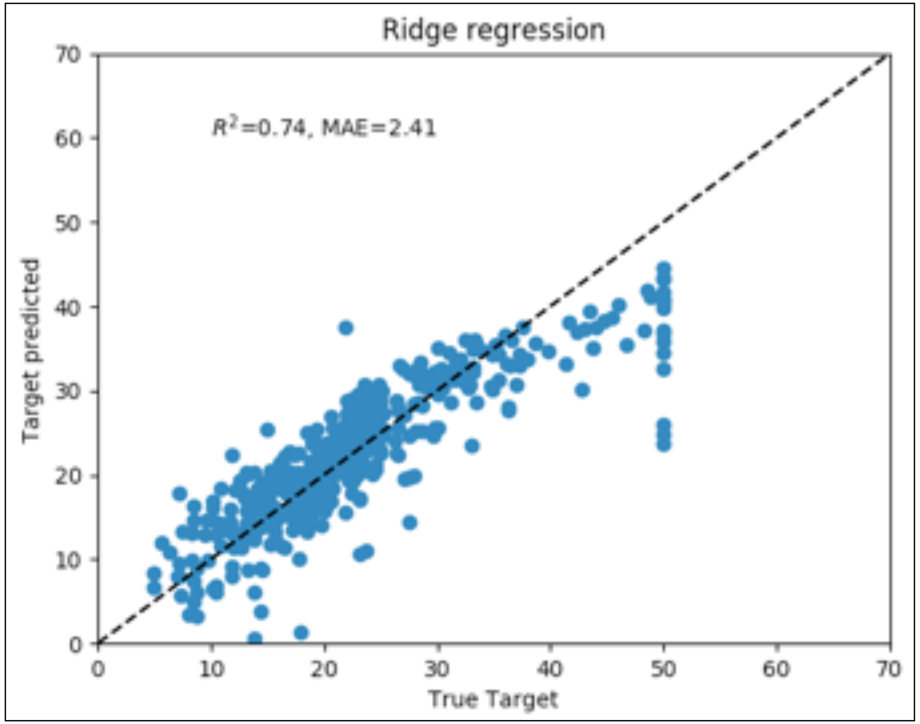
使用的套件:

```
matplotlib.pyplot
```

r2_score：**決定係數**（coefficient of determination）
以此來判斷模型的解釋力，1 是最好。

MAE: Median absolute error of regression loss

參考資料：https://matplotlib.org/api/pyplot_api.html



Live demo

Conclusion

課程一開始，老師便給我們 Data Mining 領域的宏觀視野，更介紹了機器學習的歷史演進。課堂過程中也透過影片讓我們瞭解目前 Data Mining 最新的發展。

作業的安排讓我熟悉 python 的操作，也感受到 sklearn 套件的便利性。從使用套件實作來學習的效果很好，讓我能很快進入狀況，而後再回頭去更深入理解原理。

此外從接觸 inAnalysis 平台，讓我們瞭解到任何系統在開發時非常講究的擴充性。一個規劃良好的系統會有高程度的彈性與穩定性，並能讓許多人同時參與開發。

這學期的 Data Mining 課程不僅讓我學到想學的，更讓我學到別的地方所學不到的。

Reference

1. http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html
2. http://blog.fukuball.com/lin-xuan-tian-jiao-shou-ji-qi-xue-xi-ji-shi-machine-learning-foundations-di-shi-si-jiang-xue-xi-bi-ji/
3. http://scikit-learn.org/stable/auto_examples/linear_model/plot_ridge_path.html#sphx-glr-auto-examples-linear-model-plot-ridge-path-py
4. https://matplotlib.org/api/pyplot_api.html