



機器學習基礎與演算法

Chapter 2 迴歸 (Regression)



「版權聲明頁」

本投影片已經獲得作者授權台灣人工智慧學校得以使用於教學用途，如需取得重製權以及公開傳輸權需要透過台灣人工智慧學校取得著作人同意；如果需要修改本投影片著作，則需要取得改作權；另外，如果有需要以光碟或紙本等實體的方式傳播，則需要取得人工智慧學校散佈權。

課程內容

2. 迴歸 (Regression)

2-1 [實作課程] Linear Regression

2-2 [實作課程] Weight Regularization

Code 放在Hub中的course內

- 為維護課程資料，courses中的檔案皆為read-only，如需修改請cp至自身環境中
- 打開terminal，輸入

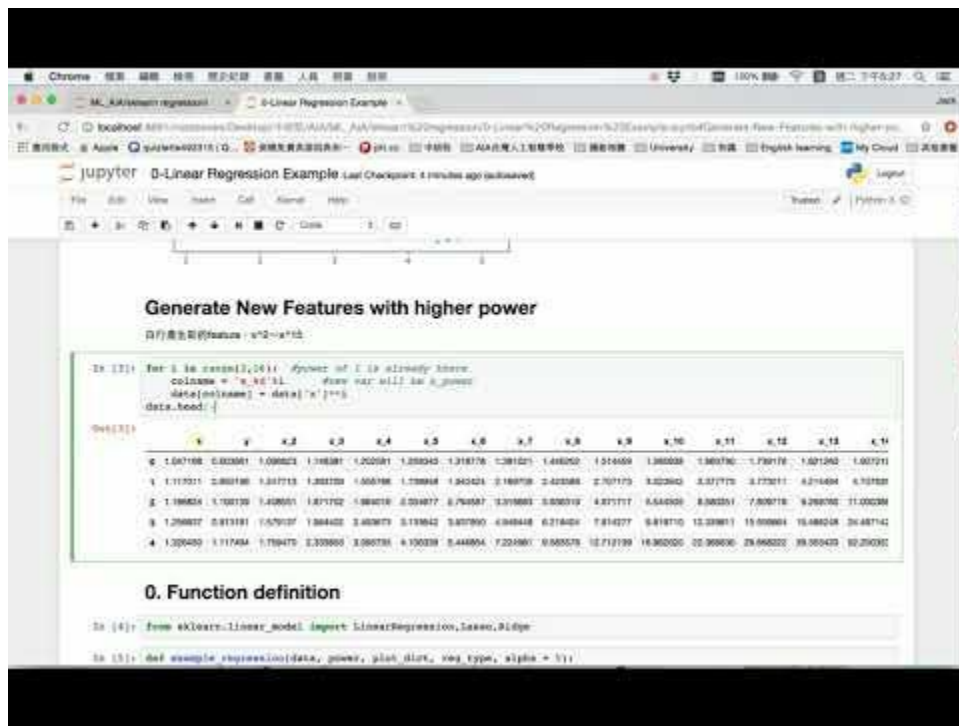
`cp -r courses-tpe/ML/Chapter2` <存放至本機的名稱>



Chapter 2 迴歸(Regression)

- 範例程式(example)的檔名會以藍色字體顯示且旁邊附上
- 練習(exercise)的檔案以紅色字體顯示且旁邊附上

Section 2-1 [實作課程] Linear Regression - part 1



Generate New Features with higher power

自行產生新的feature: $x^2 \sim x^{15}$

```
In [17]: for i in range(15): #power of 1 is already there.  
         colname = 'x_{}'.format(i) #new var will be x_power  
         data[colname] = data['x']**i  
         data.loc[:, colname]
```

	x	x ²	x ³	x ⁴	x ⁵	x ⁶	x ⁷	x ⁸	x ⁹	x ¹⁰	x ¹¹	x ¹²	x ¹³	x ¹⁴
0	1.047198	0.000961	1.000023	1.018381	1.070043	1.118776	1.181021	1.448050	1.514404	1.580038	1.589780	1.730178	1.801242	1.807211
1	1.112011	0.000180	1.337713	1.382109	1.604786	1.738948	1.943421	2.188978	2.422388	2.707175	3.022043	3.377775	3.773211	4.214404
2	1.188404	1.000739	1.408601	1.671702	1.984018	2.304877	2.744581	3.318663	4.030019	4.877717	5.944509	7.340251	9.080718	11.200336
3	1.258837	0.811191	1.579707	1.844402	2.403873	3.138642	4.073900	5.348448	7.114024	9.514277	12.733811	16.999861	22.488248	29.687142
4	1.320409	1.171404	1.788475	2.330860	3.080781	4.100338	5.448804	7.221580	9.585578	12.712739	16.802005	22.068036	29.560433	39.250393

0. Function definition

```
In [4]: from sklearn.linear_model import LinearRegression, Lasso, Ridge  
  
In [5]: def example_regression(data, power, plot_data, var_type, alpha = 1):
```



Linear Regression

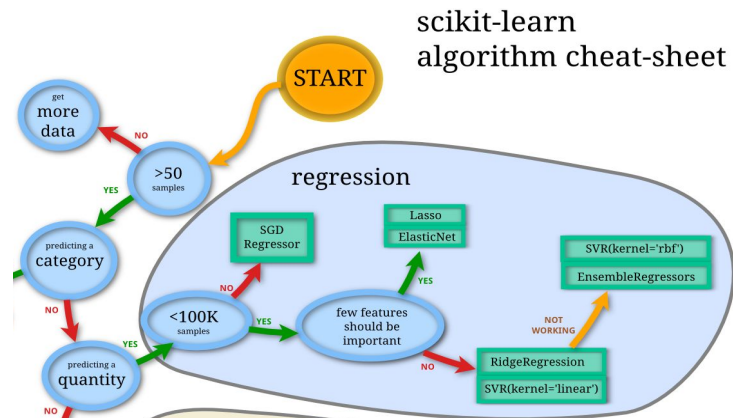
➤ sklearn.LinearRegression使用時機：

- Label為連續值
- 資料量較少(<100K)
- 假設資料的Features和Label之間有線性關係

$$y = a_1x_1 + a_2x_2 + a_3x_3 + \dots$$

➤ 可以利用Features Transformation提升模型的複雜度。

$$y = a_1x_1 + a_2x_2 + a_3x_3 + a_{11}x_1^2 + a_{12}x_1x_2 + \dots$$



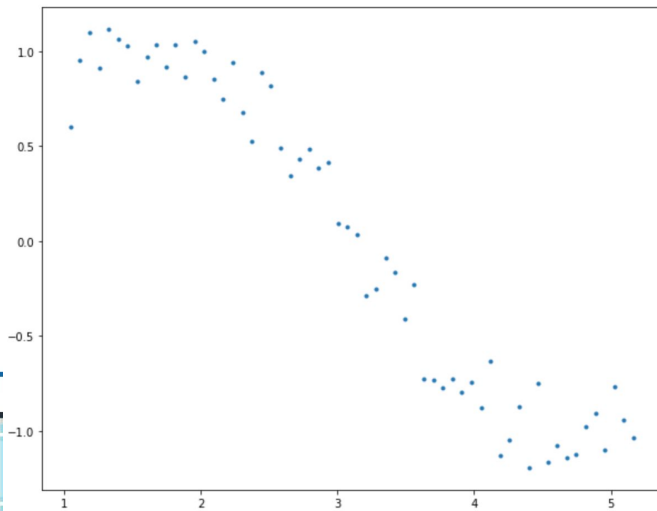
0-Linear Regression Example



Generate Sin(x) Dataset

```
In [2]: # define input array with angles from 60deg to 300deg converted to radians
x = np.array([i*np.pi/180 for i in range(60,300,4)])
np.random.seed(100) #Setting seed for reproducibility
y = np.sin(x) + np.random.normal(0,0.15,len(x))

data = pd.DataFrame(np.column_stack([x,y]),columns=['x','y'])
plt.plot(data['x'],data['y'],'.')
```

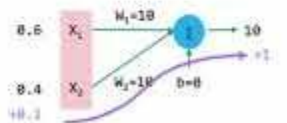
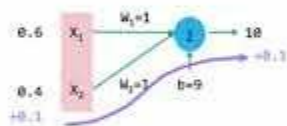


- 學習點:
 - Feature Transformation
 - Linear Regression

Section 2-1 [實作課程] Linear Regression - part 2

模型抗雜訊能力

- 模型參數越多，越容易用不相關的參數去擬合資料雜訊。當資料有雜訊時，預測值就會受到較大的影響。



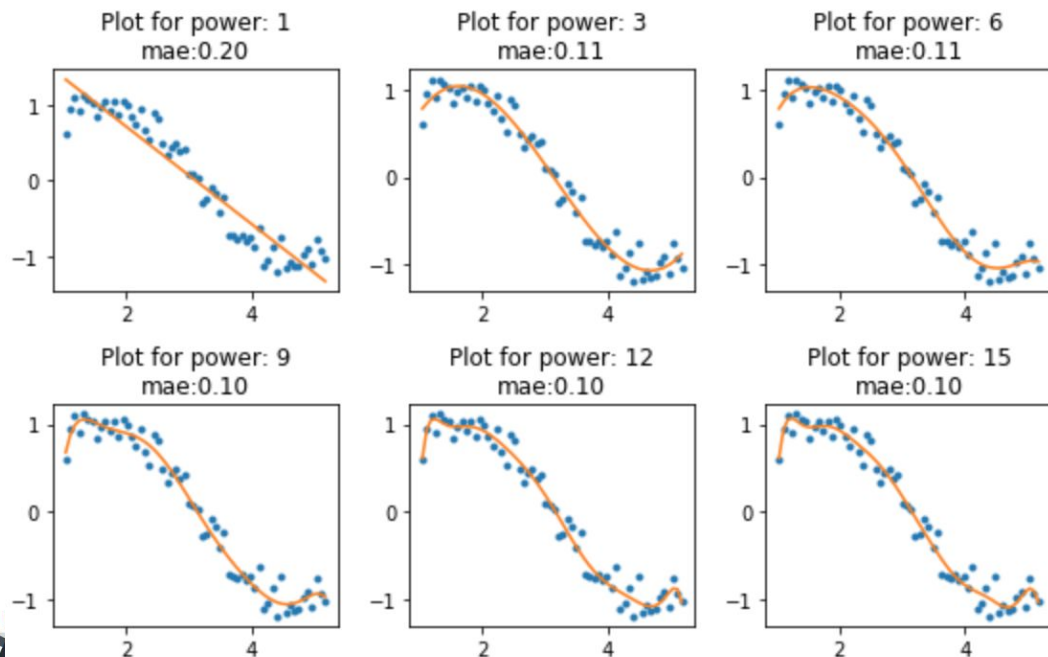
誤差越小越好！？

row	intercept	coef_x_1	coef_x_2	coef_x_3	coef_x_4
row_1	3.7	0	-0.03	NaN	NaN
row_2	3.7	1.9	-0.04	-0.017	NaN
row_3	1.1	-1.4	3.4	-1.4	5.15
row_4	1.1	-1.1	3.8	-1.1	0.002
row_5	1	0.7	-0.86	1.8	-0.97
row_6	5	-6.1	19	15	7.4
row_7	0.89	-15	64	-81	98
row_8	0.89	-49	5.1e+02	-0.9e+02	5.1e+02
row_9	0.89	-70	5.3e+02	-5.1e+02	5.4e+02
row_10	0.89	-4.5e+02	1.9e+03	-3.4e+03	3.5e+03
row_11	0.89	-5.4e+02	2.3e+03	-4.2e+03	4.4e+03
row_12	0.89	-9.9e+02	4.6e+03	-9.4e+03	1.1e+04
row_13	0.89	-1.4e+03	6.8e+03	-1.3e+04	2e+04
row_14	0.87	2.5e+03	-1.7e+04	4.3e+04	-8.3e+04
row_15	0.87	1.8e+03	-1.2e+04	3.5e+04	-6.5e+04

越高次方參數越大



0-Linear Regression Example

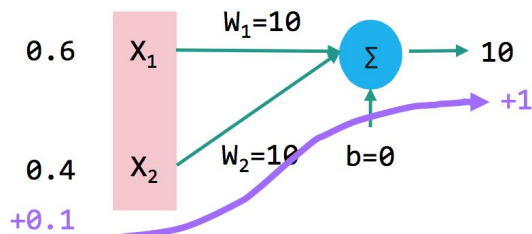
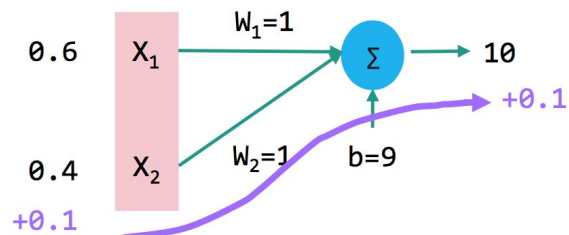


誤差越小越好！？

	rss	intercept	coef_x_1	coef_x_2	coef_x_3	coef_x_4
pow_1	3.7	2	-0.65	NaN	NaN	NaN
pow_2	3.7	1.9	-0.54	-0.017	NaN	NaN
pow_3	1.1	-1.4	3.4	-1.4	0.15	NaN
pow_4	1.1	-1.1	2.9	-1.1	0.087	0.0051
pow_5	1	0.7	-0.86	1.8	-0.97	0.18
pow_6	1	-6.1	16	-15	7.4	-2
pow_7	0.98	-19	54	-61	36	-13
pow_8	0.94	-66	2.1e+02	-2.9e+02	2.1e+02	-93
pow_9	0.94	-70	2.3e+02	-3.1e+02	2.4e+02	-1.1e+02
pow_10	0.88	-4.6e+02	1.9e+03	-3.4e+03	3.5e+03	-2.3e+03
pow_11	0.88	-5.4e+02	2.3e+03	-4.2e+03	4.4e+03	-3e+03
pow_12	0.88	-9.9e+02	4.6e+03	-9.4e+03	1.1e+04	-9.2e+03
pow_13	0.88	-1.4e+03	6.8e+03	-1.5e+04	2e+04	-1.7e+04
pow_14	0.87	2.5e+03	-1.7e+04	4.9e+04	-8.3e+04	9.5e+04
pow_15	0.87	1.8e+03	-1.2e+04	3.5e+04	-5.9e+04	6.5e+04

模型抗雜訊能力

- 模型參數越多，越容易用不相關的參數去擬合資料雜訊。當資料有雜訊時，預測值就會受到較大的影響。



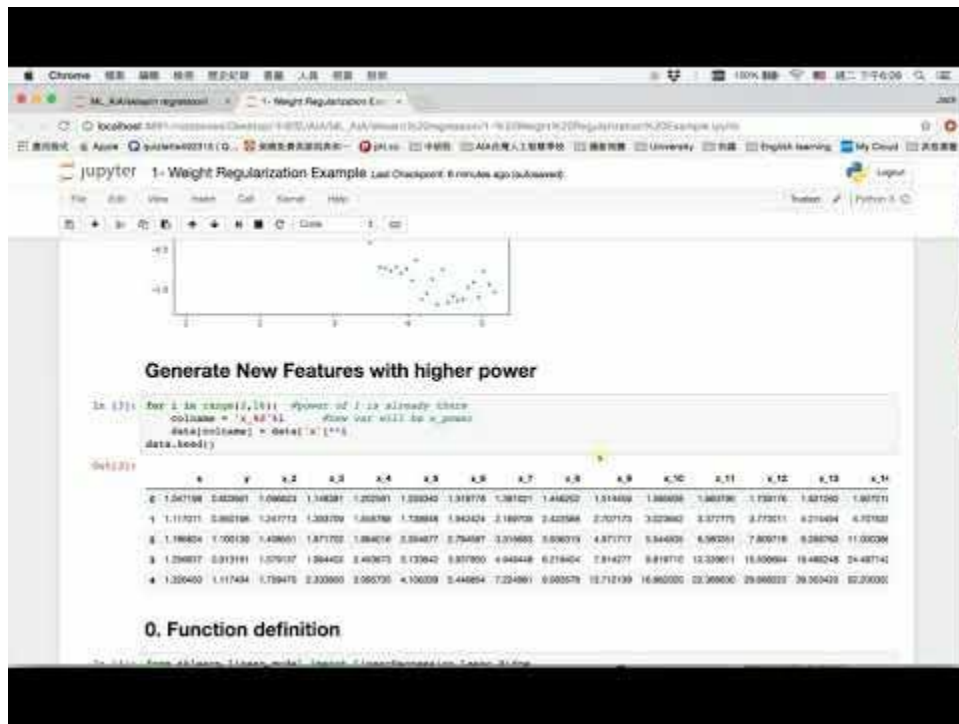
越高次方參數越大

誤差越小越好！？

受到

	rss	intercept	coef_x_1	coef_x_2	coef_x_3	coef_x_4
pow_1	3.7	2	-0.65	NaN	NaN	NaN
pow_2	3.7	1.9	-0.54	-0.017	NaN	NaN
pow_3	1.1	-1.4	3.4	-1.4	0.15	NaN
pow_4	1.1	-1.1	2.9	-1.1	0.087	0.0051
pow_5	1	0.7	-0.86	1.8	-0.97	0.18
pow_6	1	-6.1	16	-15	7.4	-2
pow_7	0.98	-19	54	-61	36	-13
pow_8	0.94	-66	2.1e+02	-2.9e+02	2.1e+02	-93
pow_9	0.94	-70	2.3e+02	-3.1e+02	2.4e+02	-1.1e+02
pow_10	0.88	-4.6e+02	1.9e+03	-3.4e+03	3.5e+03	-2.3e+03
pow_11	0.88	-5.4e+02	2.3e+03	-4.2e+03	4.4e+03	-3e+03
pow_12	0.88	-9.9e+02	4.6e+03	-9.4e+03	1.1e+04	-9.2e+03
pow_13	0.88	-1.4e+03	6.8e+03	-1.5e+04	2e+04	-1.7e+04
pow_14	0.87	2.5e+03	-1.7e+04	4.9e+04	-8.3e+04	9.5e+04
pow_15	0.87	1.8e+03	-1.2e+04	3.5e+04	-5.9e+04	6.5e+04

Section 2-2 [實作課程] Weight Regularization - part 1



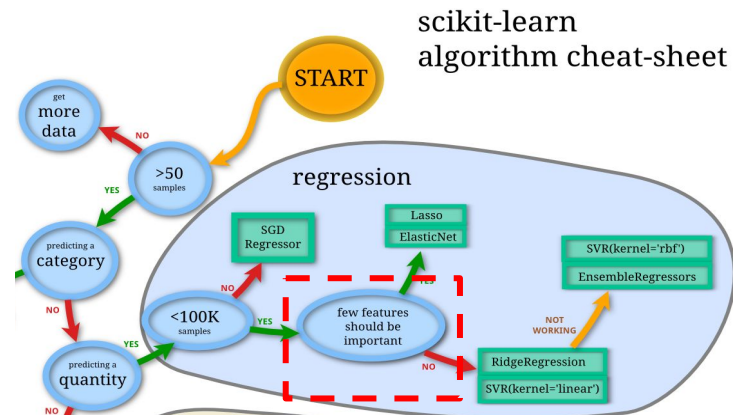
Weight Regularization

- Ridge Regression (L2)

$$\text{Cost} = \text{Prediction error} + \alpha \sum (\text{weights})^2$$

- Lasso Regression (L1)
--- Least Absolute Shrinkage and Selection Operator

$$\text{Cost} = \text{Prediction error} + \alpha \sum |\text{weights}|$$



1-Weight Regularization Example



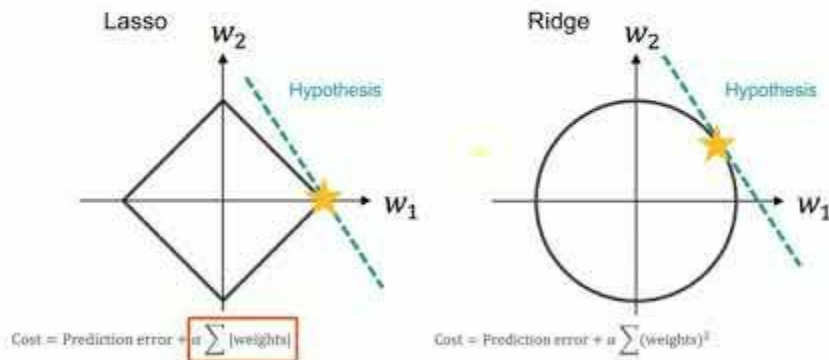
- Lasso Regularization --- L1
- Ridge Regularization --- L2
- 學習點:
 - 調整alpha並觀察結果
 - 比較L1&L2 Regularization的差異



Section 2-2 [實作課程] Weight Regularization - part 2

為什麼L1會產生許多零的Coefficient

幾何觀點: Hypothesis較容易碰到方形頂點處。 $H: y = b + w_1 \cdot x_1 + w_2 \cdot x_2$



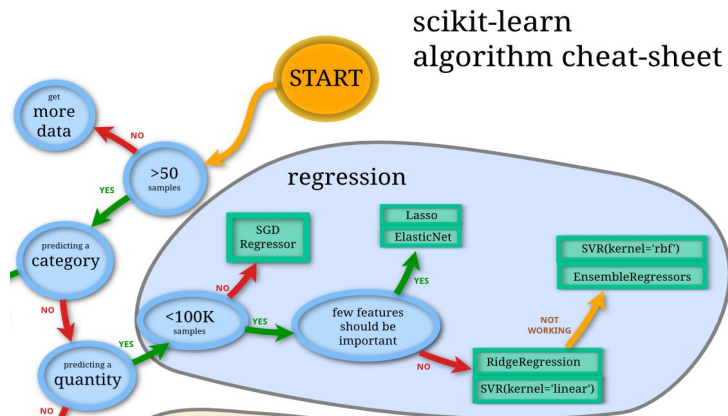
L1, L2 Regularization Summary

- 將alpha值調大對高次項的(較不相關的features)限縮越嚴謹, 抵抗資料雜訊的能力越強。

參數越小

擬合程度變小

	rss	intercept	coef_x_1	coef_x_2	coef_x_3	coef_x_4	coef_x_5	coef_x_6	coef_x_7	coef_x_8	coef_x_9
alpha_1e-3	1.1	0.69	0.43	-0.11	-0.024	-0.0029	-0.00015	3.6e-05	1.5e-05	3.6e-06	6.9e-07
alpha_2e-3	1.2	0.86	0.27	-0.088	-0.02	-0.0026	-0.00018	2.1e-05	1.2e-05	3e-06	6.1e-07
alpha_3e-3	1.3	0.97	0.18	-0.078	-0.017	-0.0023	-0.00018	1.4e-05	9.6e-06	2.6e-06	5.4e-07
alpha_4e-3	1.4	1	0.12	-0.072	-0.016	-0.0021	-0.00017	1e-05	8.2e-06	2.3e-06	4.8e-07
alpha_5e-3	1.4	1.1	0.07	-0.067	-0.014	-0.002	-0.00016	7.5e-06	7.1e-06	2e-06	4.3e-07
alpha_6e-3	1.5	1.2	0.032	-0.064	-0.013	-0.0018	-0.00016	5.5e-06	6.3e-06	1.8e-06	3.9e-07
alpha_7e-3	1.5	1.2	0.0019	-0.061	-0.013	-0.0017	-0.00015	3.9e-06	5.6e-06	1.7e-06	3.6e-07
alpha_8e-3	1.6	1.2	-0.023	-0.058	-0.012	-0.0017	-0.00015	2.6e-06	5.1e-06	1.5e-06	3.4e-07
alpha_9e-3	1.6	1.2	-0.044	-0.057	-0.011	-0.0016	-0.00014	1.5e-06	4.6e-06	1.4e-06	3.1e-07
alpha_10e-3	1.6	1.3	-0.061	-0.055	-0.011	-0.0015	-0.00014	6.4e-07	4.2e-06	1.3e-06	2.9e-07
alpha_11e-3	1.7	1.3	-0.076	-0.053	-0.01	-0.0015	-0.00014	-1.3e-07	3.9e-06	1.2e-06	2.8e-07
alpha_12e-3	1.7	1.3	-0.089	-0.052	-0.01	-0.0014	-0.00013	-7.9e-07	3.6e-06	1.2e-06	2.6e-07
alpha_13e-3	1.7	1.3	-0.1	-0.051	-0.0098	-0.0014	-0.00013	-1.4e-06	3.3e-06	1.1e-06	2.5e-07
alpha_14e-3	1.7	1.3	-0.11	-0.05	-0.0095	-0.0013	-0.00013	-1.9e-06	3.1e-06	1e-06	2.4e-07
alpha_15e-3	1.7	1.3	-0.12	-0.049	-0.0093	-0.0013	-0.00013	-2.4e-06	2.9e-06	9.7e-07	2.3e-07

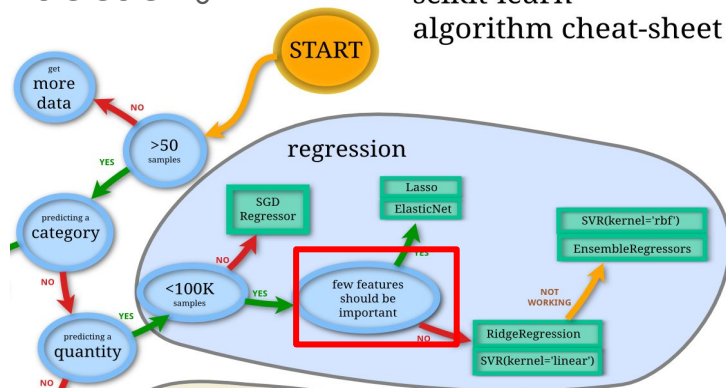


L1, L2 Regularization Summary

- Lasso(L1) Regularization相較於Ridge(L2) Regularization會產生較多零的 coefficient, 這個特性可以用來做重要Feature Extraction。

	rss	intercept	coef_x_1	coef_x_2	coef_x_3	coef_x_4	coef_x_5	coef_x_6	coef_x_7	coef_x_8	coef_x_9	coef_x_10	coef_x_11	coef_x_12	coef_x_13
pow_1	3.7	2	-0.64	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
pow_2	3.7	1.9	-0.54	-0.016	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
pow_3	3.7	1.9	-0.54	-0.016	-0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
pow_4	3.1	1.5	-0.2	-0.11	-0	0.0015	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
pow_5	2.4	1.4	-0	-0.15	-0	0	0.00042	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
pow_6	2.2	1.4	-0	-0.15	-0	-0	0	7.8e-05	NaN	NaN	NaN	NaN	NaN	NaN	NaN
pow_7	2	1.3	-0	-0.13	-0.0025	-0	-0	0	1.6e-05	NaN	NaN	NaN	NaN	NaN	NaN
pow_8	1.9	1.3	-0	-0.12	-0.0043	-0	-0	0	0	3.1e-06	NaN	NaN	NaN	NaN	NaN
pow_9	1.8	1.3	-0	-0.12	-0.0044	-0	-0	0	0	0	5.9e-07	NaN	NaN	NaN	NaN
pow_10	1.9	1.3	-0	-0.12	-0.0025	-0	-0	0	0	0	0	1.1e-07	NaN	NaN	NaN
pow_11	1.9	1.3	-0	-0.13	-0.00044	-0	-0	0	0	0	0	0	1.9e-08	NaN	NaN

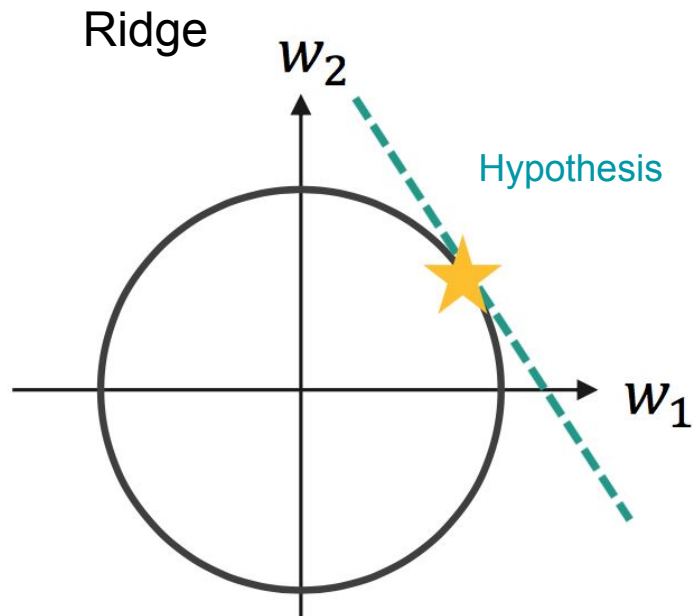
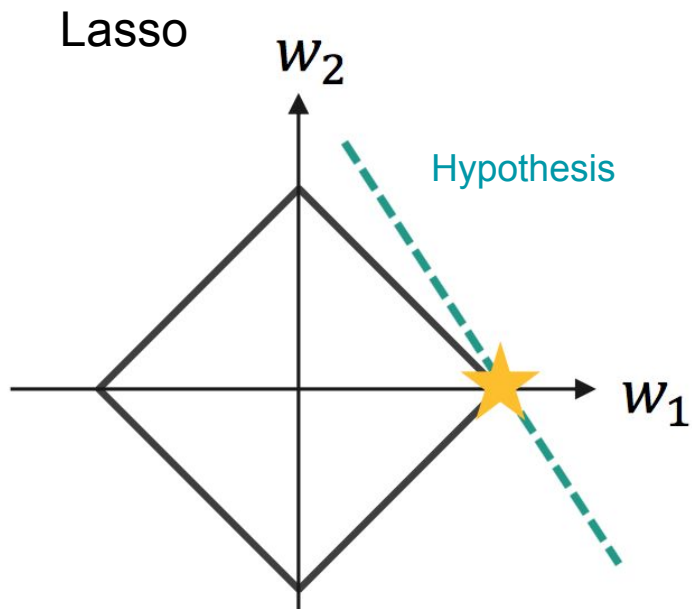
scikit-learn
algorithm cheat-sheet



為什麼L1會產生許多零的Coefficient

幾何觀點：Hypothesis較容易碰到方形頂點處。

$$H: y = b + w_1 x_1 + w_2 x_2$$



$$\text{Cost} = \text{Prediction error} + \alpha \sum |weights|$$

$$\text{Cost} = \text{Prediction error} + \alpha \sum (weights)^2$$

Short Summary

- 用 Linear Regression 解連續值的預測
- 利用 Feature Transform 增加模型複雜度
- 利用 L1&L2 Regularization 控制模型複雜度



Linear Regression Exercise 動手時間



- Boston house price prediction
 - 學習點: 1. Feature Transform 2. Weight Regularization
 - 其他: 1. 解讀重要Features 2. 選擇Alpha參數

Notes

Data Set Characteristics:

:Number of Instances: 506

:Number of Attributes: 13 numeric/categorical predictive

:Median Value (attribute 14) is usually the target

:Attribute Information (in order):

- CRIM per capita crime rate by town
- ZN proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS proportion of non-retail business acres per town
- CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX nitric oxides concentration (parts per 10 million)
- RM average number of rooms per dwelling
- AGE proportion of owner-occupied units built prior to 1940
- DIS weighted distances to five Boston employment centres
- RAD index of accessibility to radial highways
- TAX full-value property-tax rate per \$10,000
- PTRATIO pupil-teacher ratio by town
- B $1000(B_k - 0.63)^2$ where B_k is the proportion of blacks by town
- LSTAT % lower status of the population
- MEDV Median value of owner-occupied homes in \$1000's

