Linear Regression
&
Ridge Regression

Hsieh Yuan-Hao

2024/07/26

Introduction of Linear Regression

- A linear model makes a prediction by simply computing a weighted sum of the input features, plus a constant called the bias term (also called the intercept term), as shown in following function.

Equation

$$\hat{y} = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

1) $\hat{y}$ is the predicted value.
2) p is the number feature value.
3) x_i is the i^{th} feature value.
4) β_j is the j^{th} model parameter

Vector form

- This function can be written much more concisely using a vector form, as shown in following function:

Equation

$$\hat{y} = h_{\boldsymbol{\beta}}(X) = \boldsymbol{\beta}^T \boldsymbol{x}$$

1) $\boldsymbol{\beta}$ is the model's parameter vector, containing the bias term β_0 and the feature weights β_1 to β_p
2) $\boldsymbol{x}$ is the instance's feature vector, containing x_0 to x_p, with x_0 always equal to 1
3) $\boldsymbol{\beta}^T \boldsymbol{x}$ is the dot product of vectors $\boldsymbol{\beta}$ and $\boldsymbol{x}$, which is of course equal to $\hat{y} = \beta_0 x_0 + \beta_1 x_1 + \cdots + \beta_p x_p$
4) h_β is the hypothesis function, using the model parameters $\boldsymbol{\beta}$, which is often use in machine learning.

Training

- Training a model means setting its parameters so that the model best fits the training set. So first, we need to set a loss function.
- The most common performance measure of a regression model is the Root Mean Square Error (RMSE) or Mean Square Error(MSE)
- How to use Linear Regression in python

```
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression()
lin_reg.fit(X, y)
lin_reg.intercept_, lin_reg.coef_
```

Loss Function

- Root Mean Square Error(RMSE)

$$RMSE(\boldsymbol{X}, \beta, y) = \sqrt{MSE(\boldsymbol{X}, \beta, y)} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\beta^T \boldsymbol{X}^{(i)} - y^{(i)})^2}$$

- Mean Square Error (MSE)

$$MSE(\boldsymbol{X}, \beta, y) = \frac{1}{n}\sum_{i=1}^{n}(\beta^T \boldsymbol{X}^{(i)} - y^{(i)})^2$$

Note: n is the number of sample data.

Normal Equation

- To find the value of β that minimizes the lost function, there is a closed-form solution, in other words, a mathematical equation that gives the result directly. This is call the Normal Equation, form, as shown in following

$$\hat{\beta} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

1) $\hat{\beta}$ is the value of β that minimizes the lost function.
2) $\boldsymbol{y}$ is the vector of target values containing $y^{(1)}$ and $y^{(n)}$
3) $\boldsymbol{X}^T \boldsymbol{X}$ is invertible if and only if the columns of $\boldsymbol{X}$ are linearly independent.

Normal Equation Derivation

$$
\begin{aligned}
L(\beta) &= \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{\beta}^T\boldsymbol{X}^{(i)} - y^{(i)})^2 \\
&= \frac{1}{n}\|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|^2 \\
&= \frac{1}{n}(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y})^T(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}) \\
&= \frac{1}{n}(\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{y}) \\
\Rightarrow \frac{\partial}{\partial\beta}L(\beta) &= \frac{1}{n}(2\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} - 2\boldsymbol{X}^T\boldsymbol{y} + 0) \\
&= \frac{1}{n}(\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{X}^T\boldsymbol{y})
\end{aligned}
$$

Let

$$
\frac{1}{n}(\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{X}^T\boldsymbol{y}) = 0 \quad \textit{then}, \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{X}^T\boldsymbol{y} = 0
$$

$$
\Rightarrow \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{X}^T\boldsymbol{y} \Rightarrow \boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}
$$

Implementation(Data Set)

- **linear regression dataset sample.csv**

	Subject_ID	Learning_Hours	Salary
2	1	1.2	66000
3	2	1.3	60000
4	3	1.5	58000
5	4	1.6	70000
6	5	2.1	66000
7	6	2.6	78000
8	7	2.8	68000
9	8	3	70000
10	9	3.5	82000
11	10	3.6	78000
12	11	3.7	88000
13	12	3.8	90000
14	13	4	80000
15	14	4.2	75000
16	15	4.7	80000
17	16	5	78000
18	17	5.1	98000
19	18	5.3	100000
20	19	5.5	130000
21	20	5.6	150000
22	21	5.7	146000
23	22	5.8	136000
24	23	5.9	168000
25	24	6.1	188000
26	25	6.5	200000

- $\boldsymbol{X}$:將數據集中的Learning Hours 當特徵值（自變數）
- $\boldsymbol{y}$:將數據集中的Salary 當成目標值（依變數）
- 將數據集拆成訓練集與測試集，training set 佔0.7，testing set 佔0.3

Outputs

From Simple Linear Regression

- Interception : 5968.56
  coefficient : [26828.07]
  Salary = 26828.07*Learn-hours + 5968.56
- Score: 0.8302
  Accuracy: 83.02%
  Note:
  1) score() : 它會利用R平方來判斷我們模型的精準程度，也就是預測的準確度執行結果
  2) 程式碼:

```
score = regressor.score(X_test, y_test)
print('Score: ', score)
print('Accuracy: ' + str(score*100) + '%')
```

- From testing data set ***X***, to predict its ***y***.

Visualization

- Training data set

- Testing data set

Progression from linear regression to ridge regression

1) Multicollinearity:
   - Problem: When predictor variables are highly correlated, the variance of the estimated coefficients increases, making the model unstable.
   - Solution: Introduce regularization to penalize large coefficients and stabilize the model.
2) Overfitting:
   - Problem: Linear regression can overfit the training data, especially when the number of predictors is large relative to the number of observations.
   - Solution: Introduce regularization to penalize large coefficients and stabilize the model.
3) Advantages of Ridge Regression:
   - Stabilizes the estimation process in the presence of multicollinearity.
   - Reduces overfitting by controlling the complexity of the model.

Introduction of Ridge Regression

- For a linear model, regularization is a typically achieved by constraining the weights of the model. Ridge regression (also call Tikhonov regularization) is a regularization term equal to ($\lambda \sum_{j=1}^{p} \beta_j^2$) is added to the loss function. This forces the learning algorithm to not only fit the data but also keep the model weights as small as possible.
- Note:
  1) λ is a non-negative hyperparameter (also known as the regularization parameter)
  2) Choosing λ: Use techniques like cross-validation to select the optimal λ that balances bias and variance.
  3) β_j is the j^{th} model parameter
  4) p is the number feature value.

Loss Function

- The hyperparameter λ controls how much you want to regularize the model. If λ is very large, then all weights end up very close to zero and the results is a flat line going through the data's mean. The Ridge Regression loss function is as follows

$$L(\boldsymbol{\beta}) = MSE(\boldsymbol{\beta}) + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{1}$$

- Note:
  1) The bias term β_0 is not regularized (so the sum starts at j = 1, not 0)
  2) It is important to scale the data (e.g. using a StandardScaler) before performing Ridge Regression, as it is sensitive to the scale of the input features.
     - For example, consider two features x_1 and x_2,where x_1 ranges from 0 to 1, and x_2 ranges from 0 to 1000.Without scaling, the coefficients for x_2 will be penalized more heavily than those for x_1,leading to an imbalance.

Close-form solution (1/2)

- Ridge Regression closed-form solution is as follows

$$\hat{\boldsymbol{\beta}}^{ridge} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{2}$$

  note: where $\boldsymbol{I}$ is the identify matrix.

- In the previous parts we get formula $MSE(\boldsymbol{X}, \boldsymbol{\beta}, y) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{\beta}^T\boldsymbol{X}^{(i)} - y^{(i)})^2$ we can also express it as $MSE(\boldsymbol{X}, \boldsymbol{\beta}, \boldsymbol{y}) = (\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y})^T(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y})$, so we can rewrite Ridge Regression loss function as following

$$L(\beta) = (\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y})^T(\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta} \tag{3}$$

Close-form solution (2/2)

- Now we hope to find a set of parameters β that minimizes $L(\beta)$, so we partial differential beta and make it to zero

$$\frac{\partial L}{\partial \beta} = 2\boldsymbol{X}^T\boldsymbol{X}\beta - 2\boldsymbol{X}^T\boldsymbol{y} + 2\lambda\beta = 0$$

$$\Rightarrow (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})\beta = \boldsymbol{X}^T\boldsymbol{y}$$

$$\Rightarrow \hat{\beta}^{ridge} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

- How to use Ridge Regression in Python

```
from sklearn.linear_model import Ridge
regressor = Ridge()
regressor.fit(X_train, y_train)
```

Implementation(The same data)

From Ridge Regression

- Interception : 6274.93
  coefficient : [26789.23]
  Salary = 26789.23*Learn-hours + 6274.93
- Score: 0.8298
  Accuracy: 82.98%
- From testing data set $\boldsymbol{X}$, to predict its $\boldsymbol{y}$.

```
y_pred = regressor.predict(X_test)
print('Predict : ', y_pred)
```

Visualization

- Training data set

- Testing data set

Reference

- Hastie, Trevor, et al. The elements of statistical learning: data mining, inference, and prediction. Vol. 2. New York: springer, 2009.
- Bishop, Christopher M., and Nasser M. Nasrabadi. Pattern recognition and machine learning. Vol. 4. No. 4. New York: springer, 2006.
- Hoerl, Arthur E., and Robert W. Kennard. "Ridge regression: Biased estimation for nonorthogonal problems." Technometrics 12.1 (1970): 55-67.
- Machine Learning Linear Regression迴歸模型强大的Sklearn簡單線性迴歸模型、多項式迴歸模型、多元迴歸模型完整實作教學