Ordinary Least Squares: Takeaways

Syntax

- Finding the optimal parameter values using ordinary least squares (OLS):

```
first_term = np.linalg.inv(
    np.dot(
        np.transpose(X),
        X
    )
    )
second_term = np.dot(
    np.transpose(X),
    y
    )
a = np.dot(first_term, second_term)
print(a)
```

Concepts

- The ordinary least squares estimation provides a clear formula for directly calculating the optimal values that maximizes the cost function.
- The OLS estimation formula that results in optimal vector a:

$$a = (X^T X)^{-1} X^T y$$

- OLS estimation provides a closed form solution to the problem of finding the optimal parameter values. A closed form solution is where a solution can be computed arithmetically with a predictable amount of mathematical operations.
- The error for OLS estimation is often represented using the Greek letter for E. Since the error is the difference between the predictions made using the model and the actual labels, it's represented as a vector:

$$\epsilon = \hat{y} - y$$

- We can use the error metric to define y:

$$y = Xa - \epsilon$$

- The cost function in matrix form:

$$J(a) = \frac{1}{n}(Xa - y)^T(Xa - y)$$

- The derivative of the cost function:

$$\frac{dJ(a)}{da} = 2X^TXa - 2X^Ty$$

- Minimizing the cost function, $J(a)$.
  - Set the derivative equal to 0 and solve for a:
    - $2X^TXa - 2X^Ty = 0$
  - Compute the inverse of X and multiply both sides by the inverse:
    - $a = (X^TX)^{-1}X^Ty$
- The biggest limitation of OLS estimation is that it's computationally expensive when the data is large. Computing the inverse of a matrix has a computational complexity of approximately $O(n^3)$.
- OLS is computationally expensive, and so is commonly used when the numbers of elements in the dataset is less than a few million elements.

Resources

- [Walkthrough of the derivative of the cost function]
- [Ordinary least squares]