김동환

2021년 1월

**점경사평균법을 이용한 선형회귀 연구**

**(PIM LinearRegression)**

1. 서론
   1. 연구 목적
2. 선행연구
   1. 용어와 정의
3. 기술설명

3.1 점경사평균법

3.1.1 데이터 전처리

3.1.2 두 점들을 기준으로 기울기 측정

3.1.3 절편 구하기

4. 결론

5. 이용한 패키지

6. 점경사평균법 오픈소스

차례

* 1. 용어와 정의

- 점경사평균법: 두 점으로 기울기(경사)를 구하고 구한 기울기(경사) 들의 평균으로 선형회귀의 기울기를 구하는 알고리즘이다. (자세한 내용은 기술설명을 참고)

- PIM Linear Regression: PIM는 Point Inclination Mean의 약자이고, PIM Linear Regression 또한 점경사평균법과 같이 두 점으로 기울기를 구하고 구한 기울기들의 평균으로 선형회귀의 기울기를 구하는 알고리즘이다. (자세한 내용은 기술설명을 참고)

* 1. 연구 목적
* 현재 머신러닝의 회귀 부분에 많은 역할을 하고 있는 선형회귀를 ‘경사하강법’ 이라는 알고리즘을 이용하여 만들 수 있다. 많은 회귀 문제들이 전반적으로 경사하강법을 이용하여 해결되고 있다. 저자는 경사하강법을 공부하고 직접 경사하강법 알고리즘을 이용한 머신러닝을 개발하던 도중 회귀 분야에서 좀 더 새로운 접근법으로 회귀 문제를 해결하는 알고리즘이 없을까 하는 생각에 연구하게 되었다. 또한 점경사평균법을 연구하고, 개발함으로서 머신러닝의 회귀를 더 재밌게 배우고, 많은 사람들이 점경사평균법을 조금씩 변형해가며, 재밌는 기술들이 나오고 성과를 만들어내고, 4차산업혁명에 더 많은 기여를 했으면 하기 때문에 연구하게 되었다.

1. 선행연구
2. 서론
3. 기술설명

3.1 점경사평균법

- 점경사평균법은 데이터 전처리, 두 점들의 증가량을 이용한 기울기 측정, 절편 구하기. 의 순서로 설계되어있다. 파이썬으로 작성되었다.

3.1.1 데이터 전처리

- 먼저 **데이터 전처리** 과정에서는 x 값이 비슷한 y 값들끼리의 평균값을 구하고, 실제 데이터에 반영한다. 데이터 전처리를 하는 이유는 연산처리속도를 줄이고, 메모리 효율을 높이기 위해서이다. 해당 코드는 다음과 같다.

data = list(zip(X, y))

data = pd.DataFrame(data, columns=["X", "y"])

**def** duplicate(x):

duplicate\_data = data[(data["X"] > (x - dp)) & (data["X"] < (x + dp))]["y"]

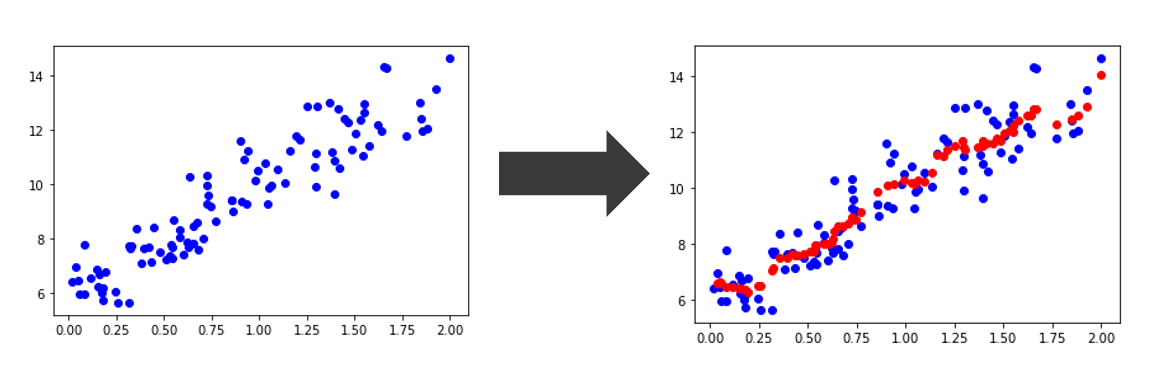
**return** sum(duplicate\_data) / len(duplicate\_data)

data["y"] = data["X"].apply(**lambda** x: duplicate(x))

data = data.drop\_duplicates(["y"], keep="first")

data = data.reset\_index(drop=True)

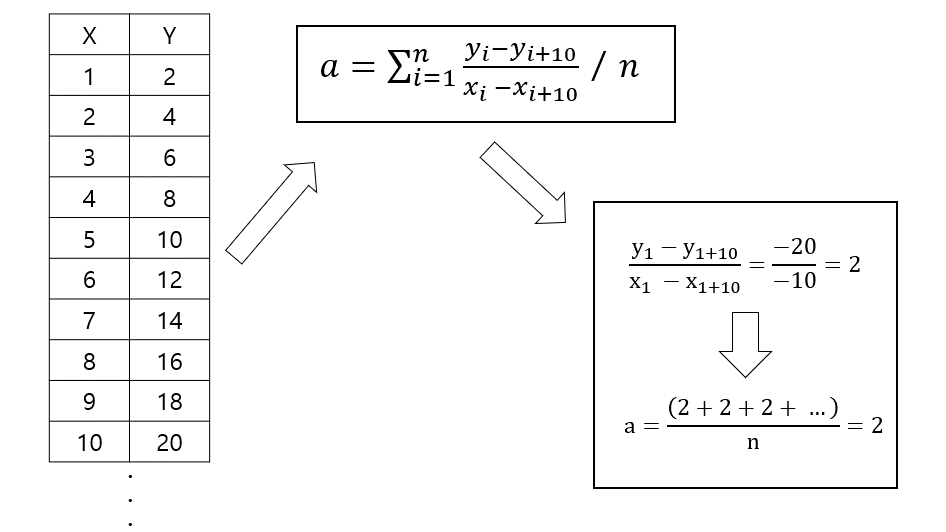
파란색을 원래 데이터, 빨간색을 전처리한 데이터로 표시하면 다음과 같은 그래프를 그릴 수 있다.



3.1.2 두 점들을 기준으로 기울기 측정

- **두 점들의 증가량을 이용한 기울기 측정**을 해야 한다. 기울기 측정은 많은 점들중 두 점을 기준으로 기울기를 측정하고 이 과정을 반복한 후에 기울기들의 평균으로 최종 기울기를 구한다. a 를 기울기라고 하고, 수식으로 표현하면 다음과 같다.

이 수식을 이해하기 쉽게 그림으로 표현하면 다음과 같이 표현할 수 있다.



해당 코드는 다음과 같다.

**for** i **in** range(data.shape[0]):

**try**:

up.append((data.iloc[i, 1] - data.iloc[i + uprate, 1]) / (data.iloc[i, 0] - data.iloc[i + uprate, 0]))

**except**:

**pass**

**for** i **in** reversed(range(data.shape[0])):

**try**:

up.append((data.iloc[i, 1] - data.iloc[i - uprate, 1]) / (data.iloc[i, 0] - data.iloc[i - uprate, 0]))

**except**:

**pass**

*# 기울기 구하기*

a = round(sum(up) / len(up), 3)

3.1.3 절편 구하기

- **절편**은 y 값과 예측값의 차이의 평균으로 구할 수 있다. b를 절편으로 두고, 수식으로 표현하면 다음과 같다.

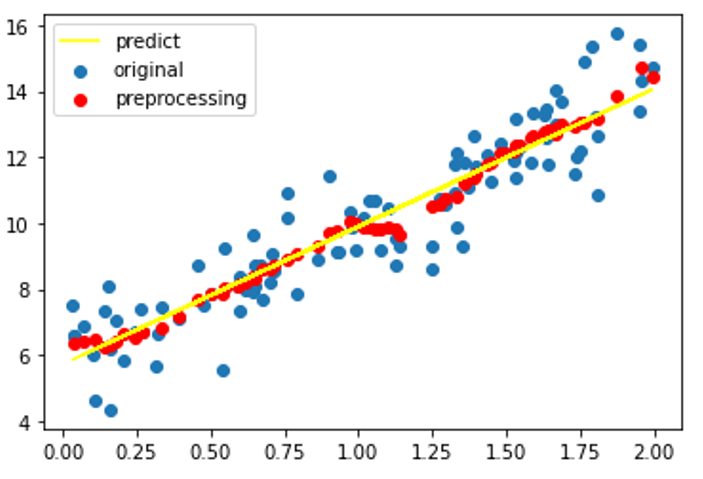
해당 코드는 다음과 같다.

*# 절편 구하기*

b\_lst = data.iloc[:, 1] - (a \* data.iloc[:, 0])

b = round(sum(b\_lst) / len(b\_lst), 3)

numpy 라는 패키지를 이용하여 임의의 데이터를 생성하고, 임의의 데이터로 학습한 모델을 평가하면 아래와 같은 그래프가 나온다. (그래프는 matplotlib 라는 패키지를 이용)



**파란색 산점도**는 원래 있었던 데이터이고**, 빨간색 산점도**는 점경사평균법의 첫번째 과정을 거쳐 전처리가 된 데이터이다. 그리고 **노란색 그래프**는 학습된 모델이 예측하고 있는 그래프이다. 완성된 PimLinearRegression의 코드는 차례 6에서 확인할 수 있다.

1. 결론

- numpy: Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). DOI: 0.1038/s41586-020-2649-2. (Publisher link).

([https://www.nature.com/articles/s41586-020-2649-2)](https://www.nature.com/articles/s41586-020-2649-2)

- pandas: Wes McKinney. Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference, 51-56 (2010) (publisher link)

(<http://conference.scipy.org/proceedings/scipy2010/mckinney.html>)

- matplotlib: John D. Hunter. Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering, 9, 90-95 (2007), DOI:10.1109/MCSE.2007.55 (publisher link)

(<https://ieeexplore.ieee.org/document/4160265/>)

1. 이용한 패키지

* 점경사평균법(PIM Linear Regression)은 보다 어렵지 않게 머신러닝의 회귀 분야에 대해 재미있게 공부할 수 있게 해주고, 많은 것들을 예측할 수 있는 머신러닝 알고리즘이다. 이 점경사평균법의 핵심 내용은 이름에서 알 수 있듯이, 점들 중에 두 점을 기준으로 기울기(경사)를 구하고 그 기울기(경사)의 평균을 구하여 회귀를 만들어내는 것이라고 할 수 있다. 점경사평균법 알고리즘이 많이 알려져서 보다 많은 사람들이 점경사평균법을 이용한 머신러닝을 개발하고, 인공지능에 관심을 갖고 기여를 했으면 하는 생각이고, 또한 인공지능 개념이 어려워 인공지능 공부를 그만두는 사람보다 인공지능이 재밌어서 더 많은 인공지능을 개발하는 사람들이 많아졌으면 하는 마음에 만들게 되었다.

6. 점경사평균법 오픈소스

* 현재 점경사평균법은 github에 MIT License 로 등록하여, 오픈소스로 공개되어 있는 상태이다. 아래 링크를 통해 연구개발이 완료된 점경사평균법을 확인할 수 있다. (<https://github.com/SimplePro/custom_linear_regression/>)