나이브 베이시안, 선형회귀, 로지스틱스는 손으로 짜야하고

디시젼 트리 부터는 라이브러리 쓰면 됨!

선형회귀 모형으로 하면 옵티마이제이션으로 가는 것임.

결론적으로 변수 값(변수의 기울기 값)

규제화(릿지, 랏소 모델-패널티)

머신러닝으로 학습을 시켰는데 설명이 안되는 경우, -> 데이터에서 여러 개의 모델이 있을 때, 데이터에서 영향력이 있는 피쳐는 다른 데이터에도 영향력이 있을 것임.

<https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDRegressor.html>

엘 2 모델이 릿지 모델

알파\_ 정규화 텀임.

릿지 모델에서는 L = ​I~M \*ㅛ1-ㅌ1세터1-세터2)제곱 + 알파

계수를 업데이트 해주는 것

랏소 모델은 알파

수식에서 알파가 그 알파임

learning\_rate: 학습 율 ( 계속 업데이트 됨)

그러나 학습 율에 상수 값을 주면 고정됨.

**‘invscaling’: [default]**

eta = eta0 / pow(t, power\_t)

통해서 학습율이 업데이트 됨.

옵티마이제이션\_ 글로벌 옵티멈(전역\_전체에 걸쳐서 해)

업데이트 할때마다 샘플을 랜덤하게 뽑을 것

글로벌 옵티멈으로 못갈수도 있음 그래서\_뜨레드 홀드 만큼의 변화가 없으면 멈춰가\_ **early\_stopping : *bool, default=False(임계치, 변화에 대한 임계치)***

***정규화를 하는 이유는 옵티마이제이션을 할 때, 옵티마이제이션의 속도를 조절하기 위해서 정규화 하는 것임.***

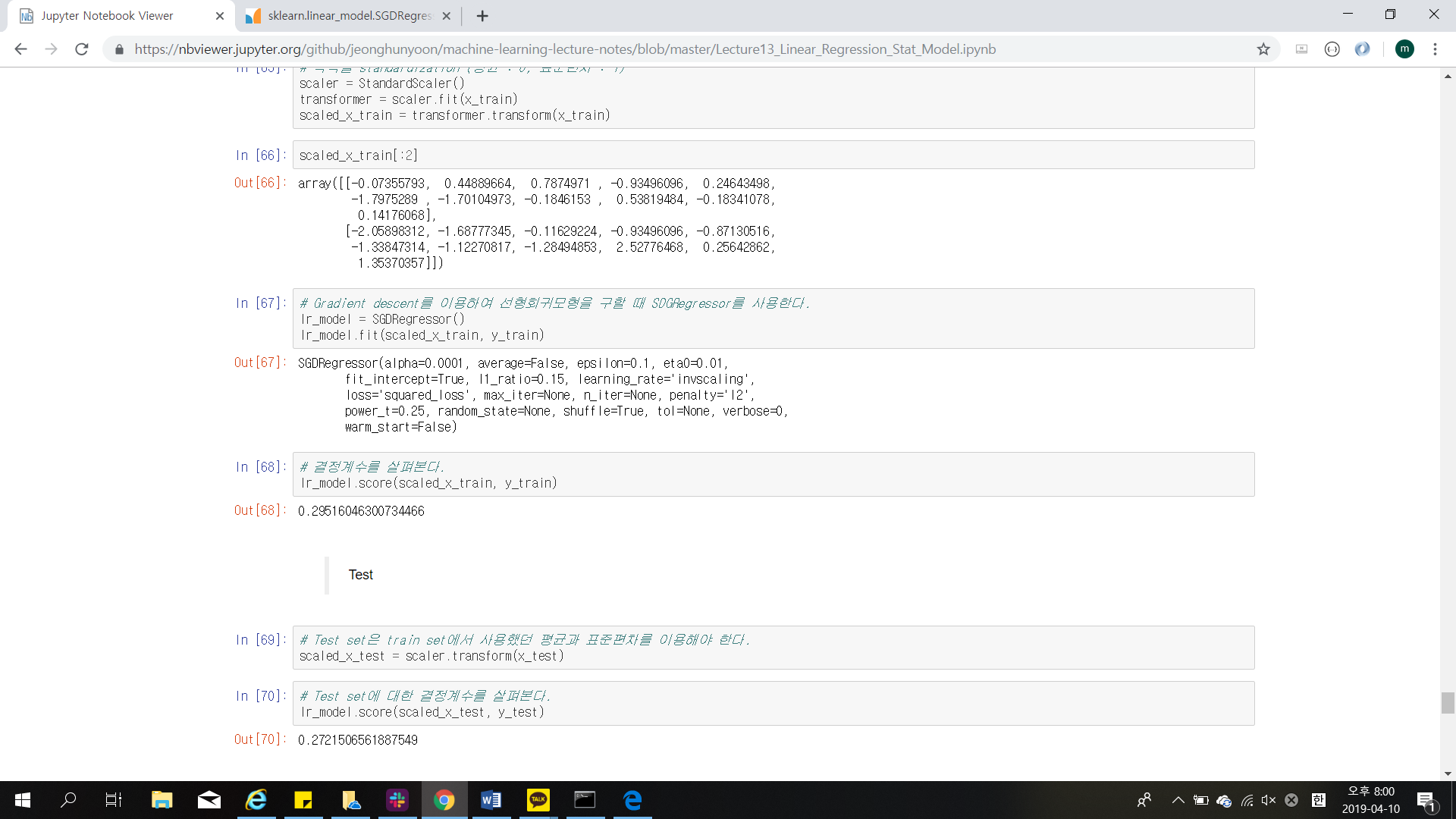
***변수 간의 스케일을 맞춰줘서 제대로 비교하는 것임!***

***머신러닝에서 정규화 하는 이유는 스케일하려는 것이 아니라 옵티마이제이션을 하기 위해서 임.***

* 스케일이 크면 클수록 미분 값이 커짐/스케일이 다르면 스케일이 큰놈을 기준으로 세터 제로와 세터 1 점의 변화 량을 볼 수 있음.
* 등고선의 의미는 로스 값의 변화량임
* 머신러닝에서 그래디언트를 사용할 때는 스케일을 해서 정규화를 해줘야함.

\*\* 스케일을 해주는 라이브러리가-> **from** **sklearn.preprocessing** **import** StandardScaler

트랜스 포머가 엑스\_ 트레인 값의 평균, 분산 값 등을 다 가지고 있음



엔\_이터: 몇번 돌아 가냐

중요한 것은 고도화하냐 못하는 것이냐 (하이퍼 파라미터\_사람이 조절할 수 있는 파라미터 이냐가 중요함)

성능이 가장 좋은

알파 값에 변화를 주면서 성능이 얼마나 좋은 지 판단 한다.

알파 값에 따라 트레인 이랑 테스트 셋에서의 알 스퀘어 값을 찾아서 가장 성능이 좋은 것을 찾아야 한다.

유의하지 않다고 판단했던 변수들은 사라지지 않고 쓴다!

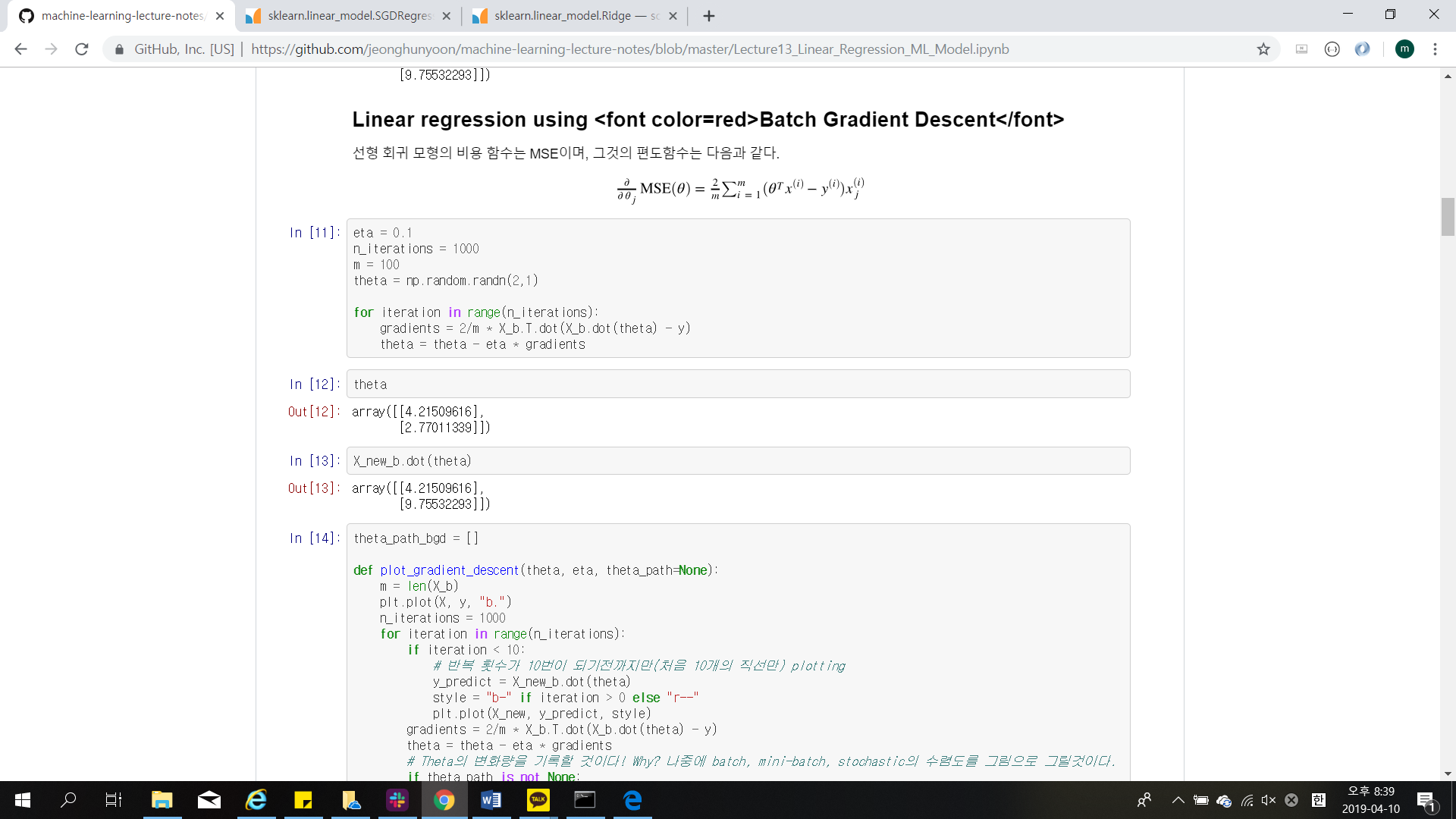
옵티마이제션을 해서 하는 게 아니라 바로

1년 동안 300만 유저가 만든 패팅방이 1400만임.-> 샘플링(어떠케 샘플링 할까?

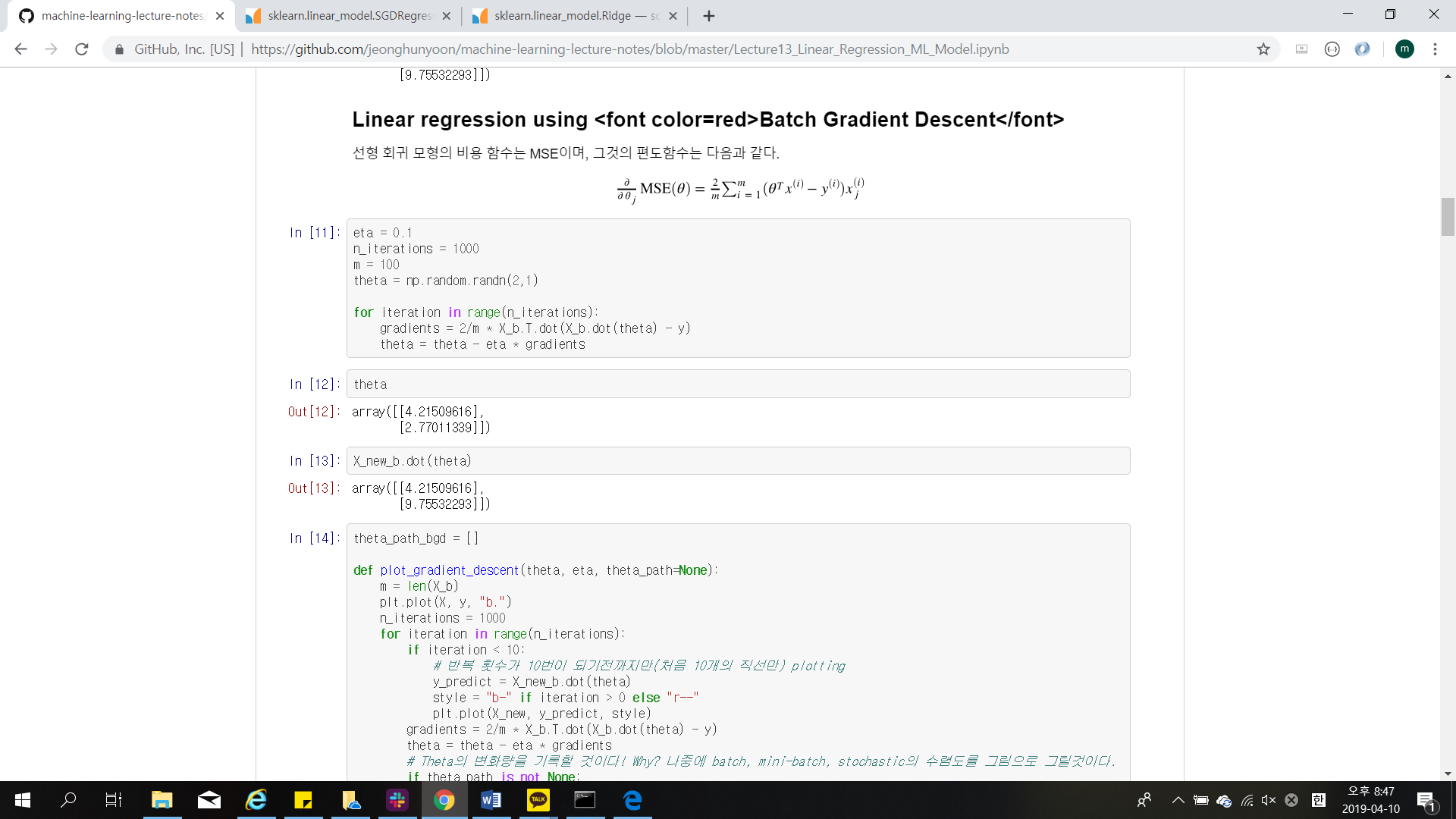
1. 1차 샘플을 뽑는다-> 1000만 -이걸 어떠케 샘플링(이 가장 중요하다) 할까?

최근 일주일 정도 안에 제재가 들어온 방이 90만 개-> 여기에 자주 등장하는 단어들

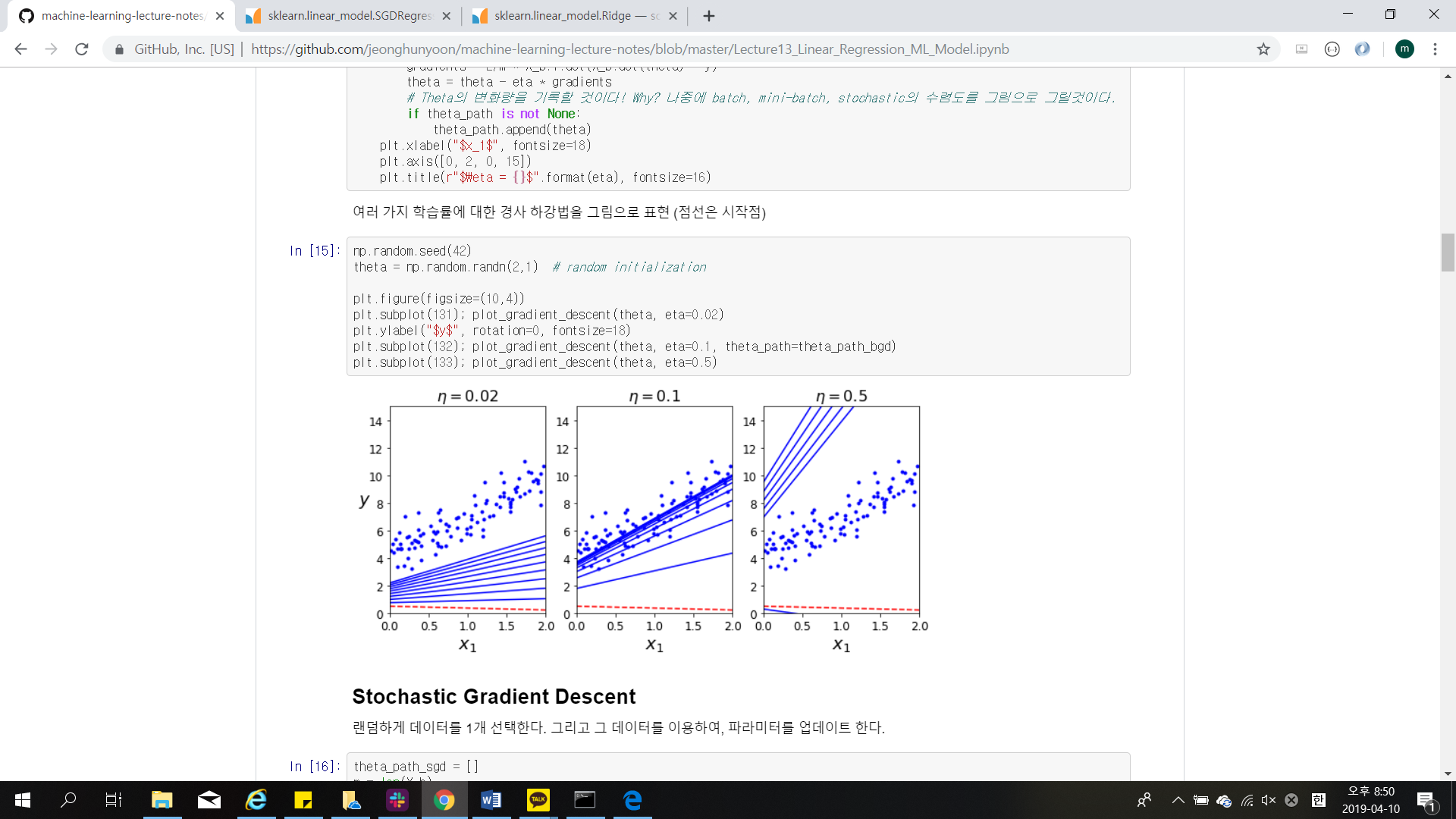
층하 계층(stratified sampling) 길이 분포/정말 좋은 데이터를 어떻게 골라내나



* 사진으로 식을 이해하고 손으로 쓸 것



* 이 수학식에서 중요한 것은 2/엠이 아니라 뒤에 수학 식임.

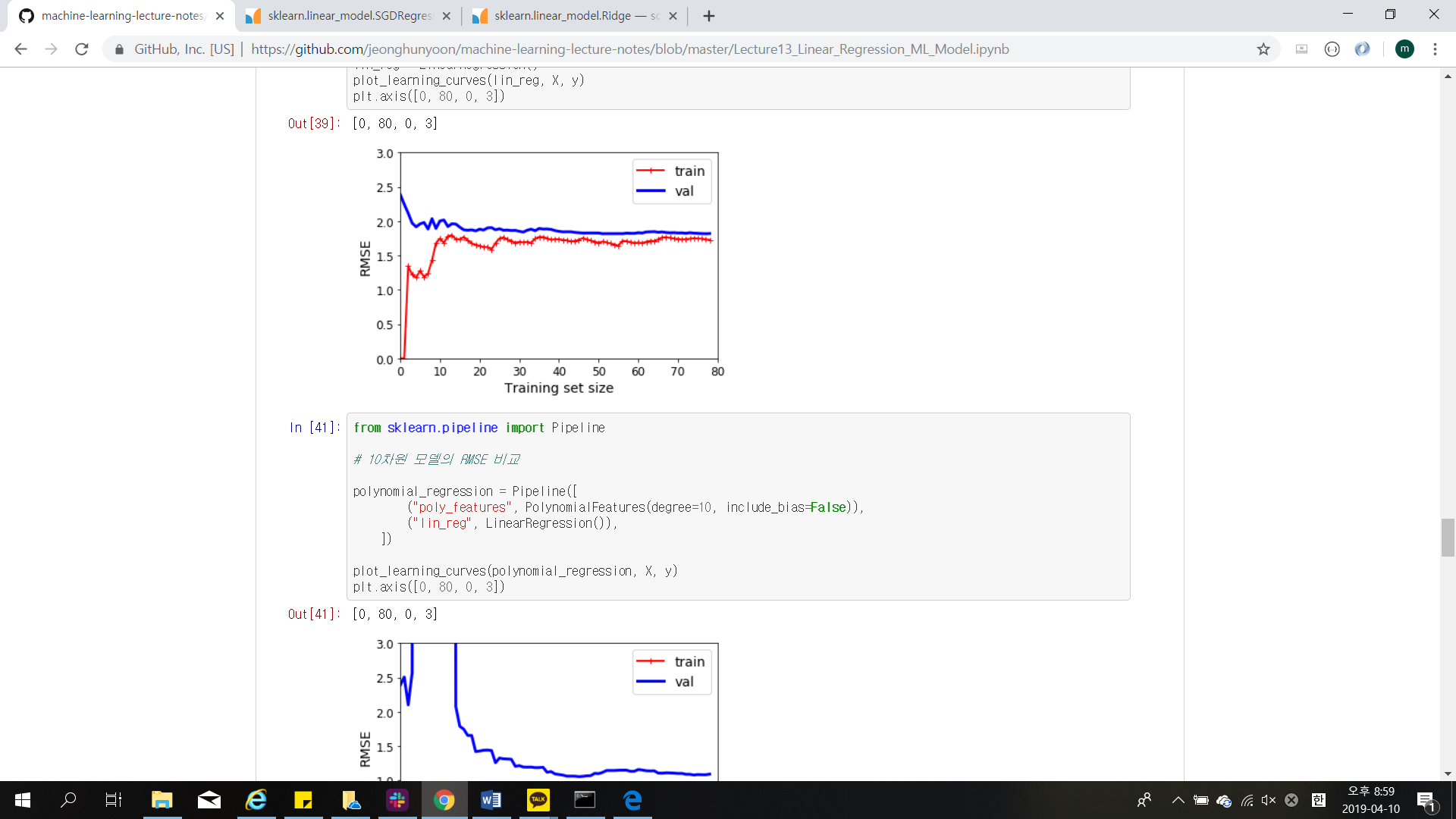


학습률을 0.02 로 주었을 때 초반이 촘촘함

세터가 0.1 일때는

그레디언트는 방향임

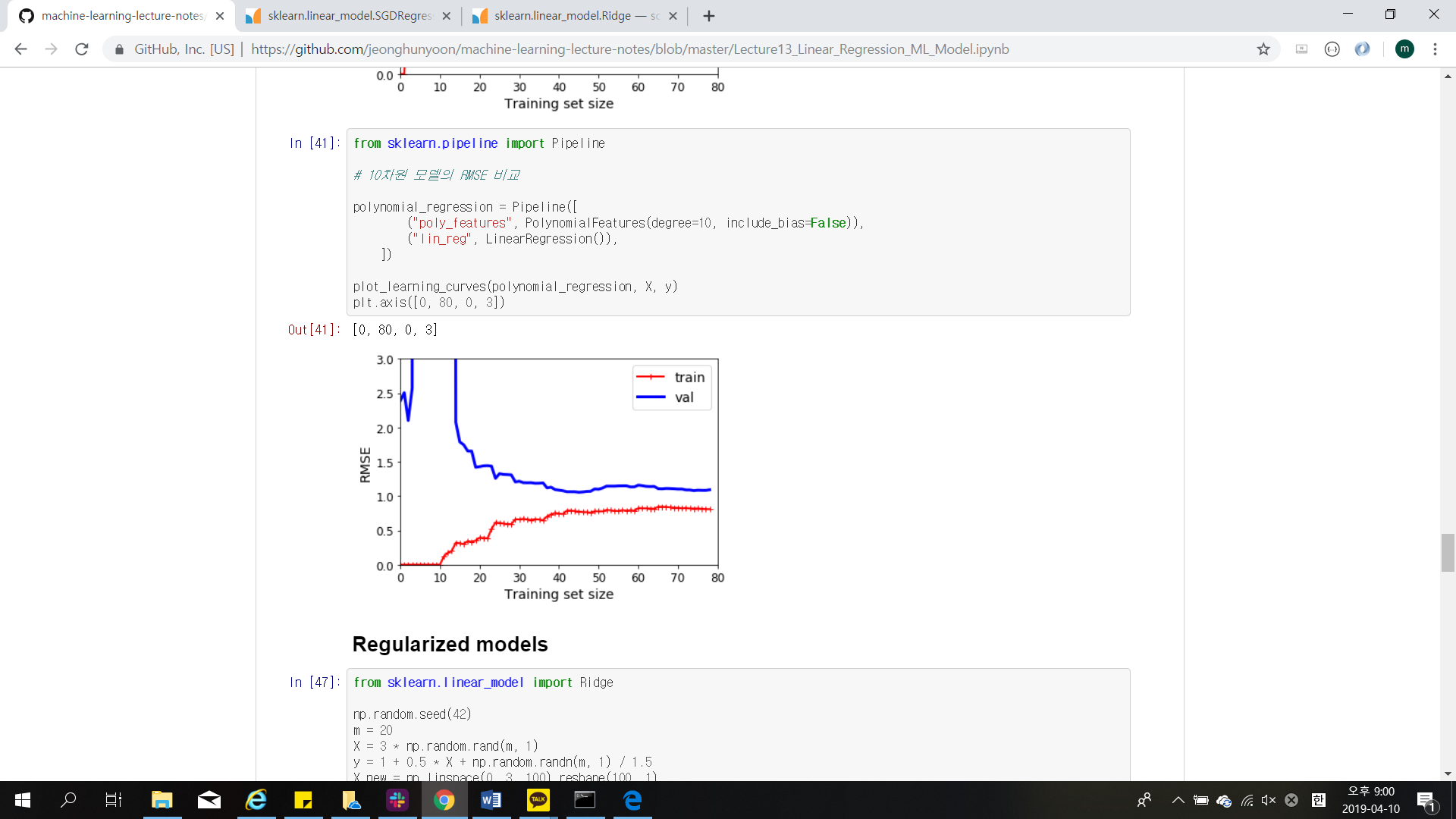
그래서 학습 률이 정말 중요함!



알엠에스이가 높으면 높을수록 예측을 못한 것

그래서 언더 피팅

언더 피팅은 벨리데이션이 높고

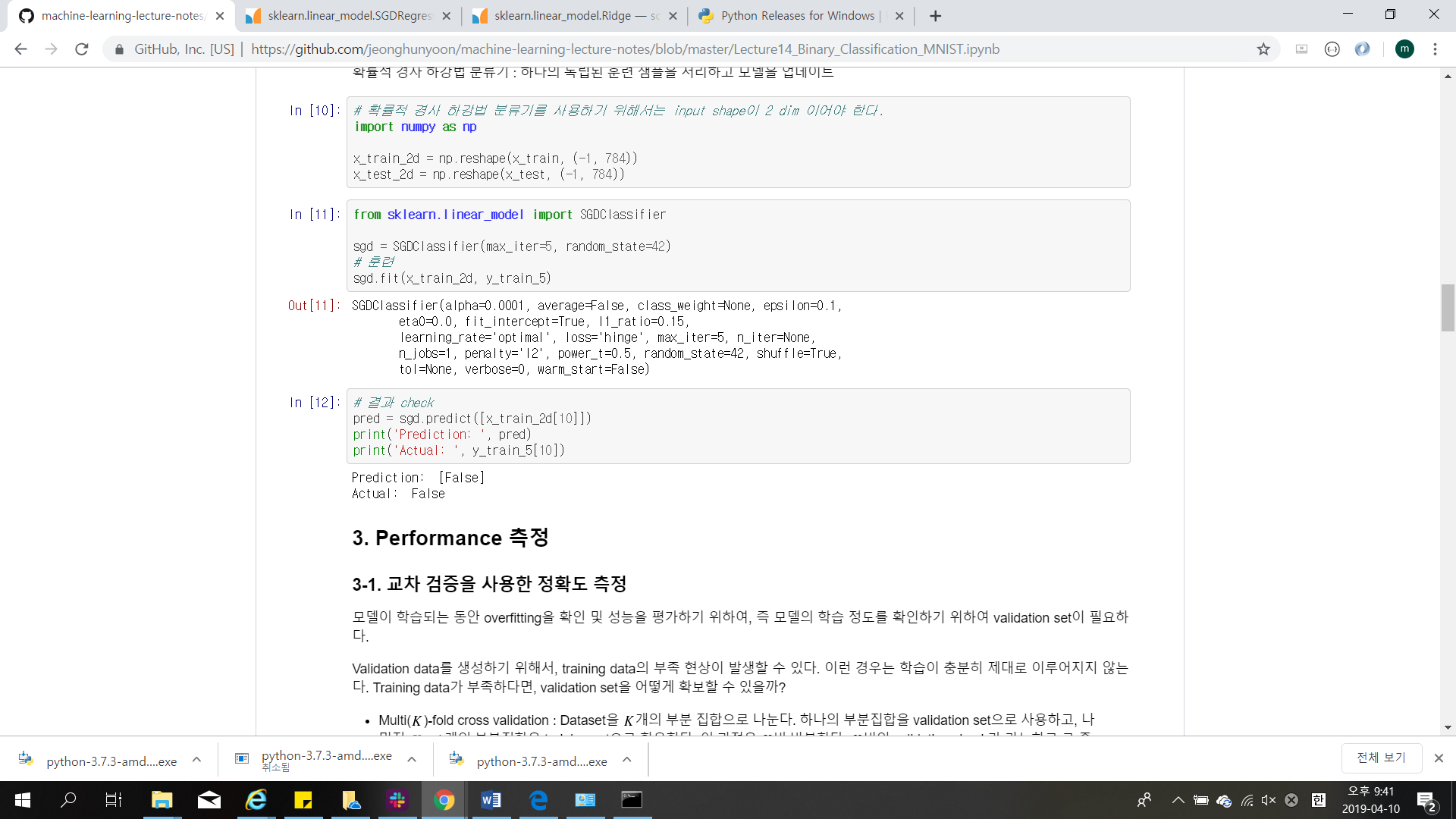


트레인과 밸리데이션의

* 그림만 보고 언더피팅과 오버 피팅을 구분할 수 있어야 함.
* 얼리 스토핑은 rmse가 어느 정도 되었을 때 멈추는 것.
* 훈련 오차와 검증 오차 사이가 좁아야 함.

전체 데이터 셋의 비율에 따라 트레이닝 셋과 학습 셋도 그 비율을 맞출 것

각 클래스에서의 비율을 맞춰야 함.



클래스 웨이트를 주게 되면 해당 클래스 가 틀리면 로스 값이 커짐!

클래스 웨이트를 잘 줘야함\_민감도라기 보다는 분류기가 어느 쪽을 더 타겟팅 하는 지 조절 하는 것임.

프리시젼 리콜\_정상인 사람을 스패머로 예측 하면 안됨(정상인에 웨이트를 줄 것)

* 강사님 화재 모델은/ 내가 타겟팅하게 민감하다!/내가 타겟팅하는 클래스를 위해서 웨이트를 주게 됨.

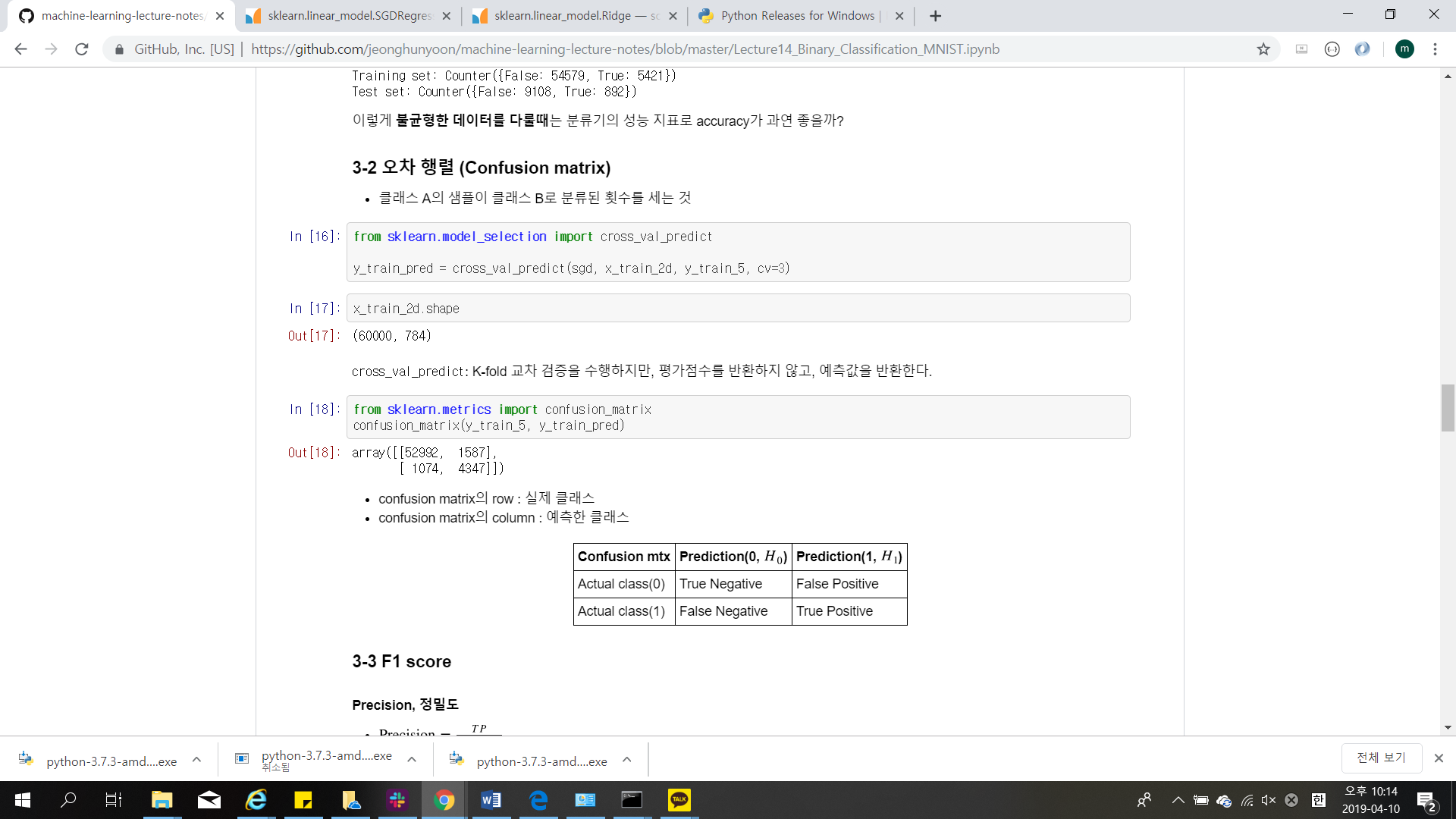
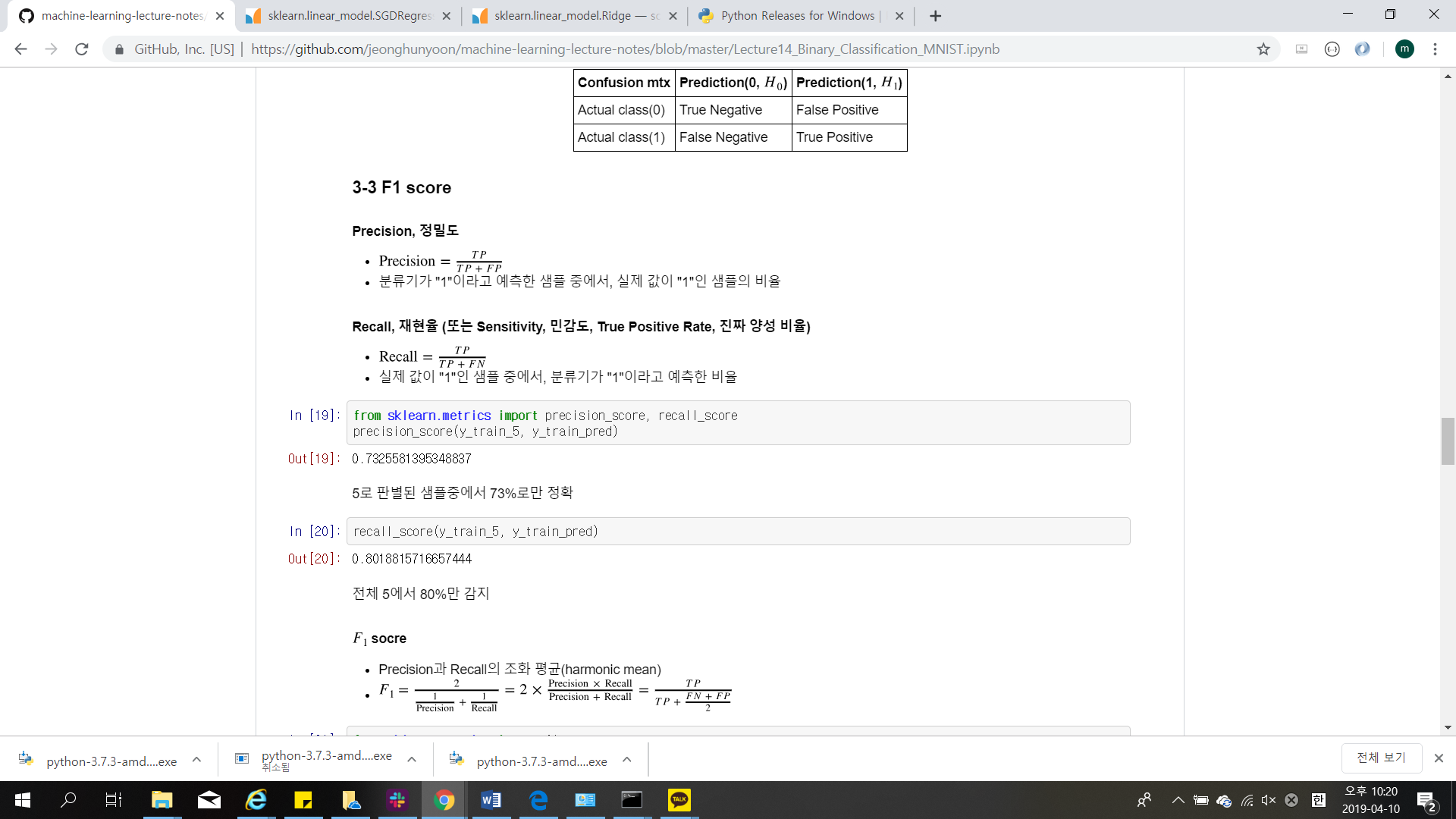
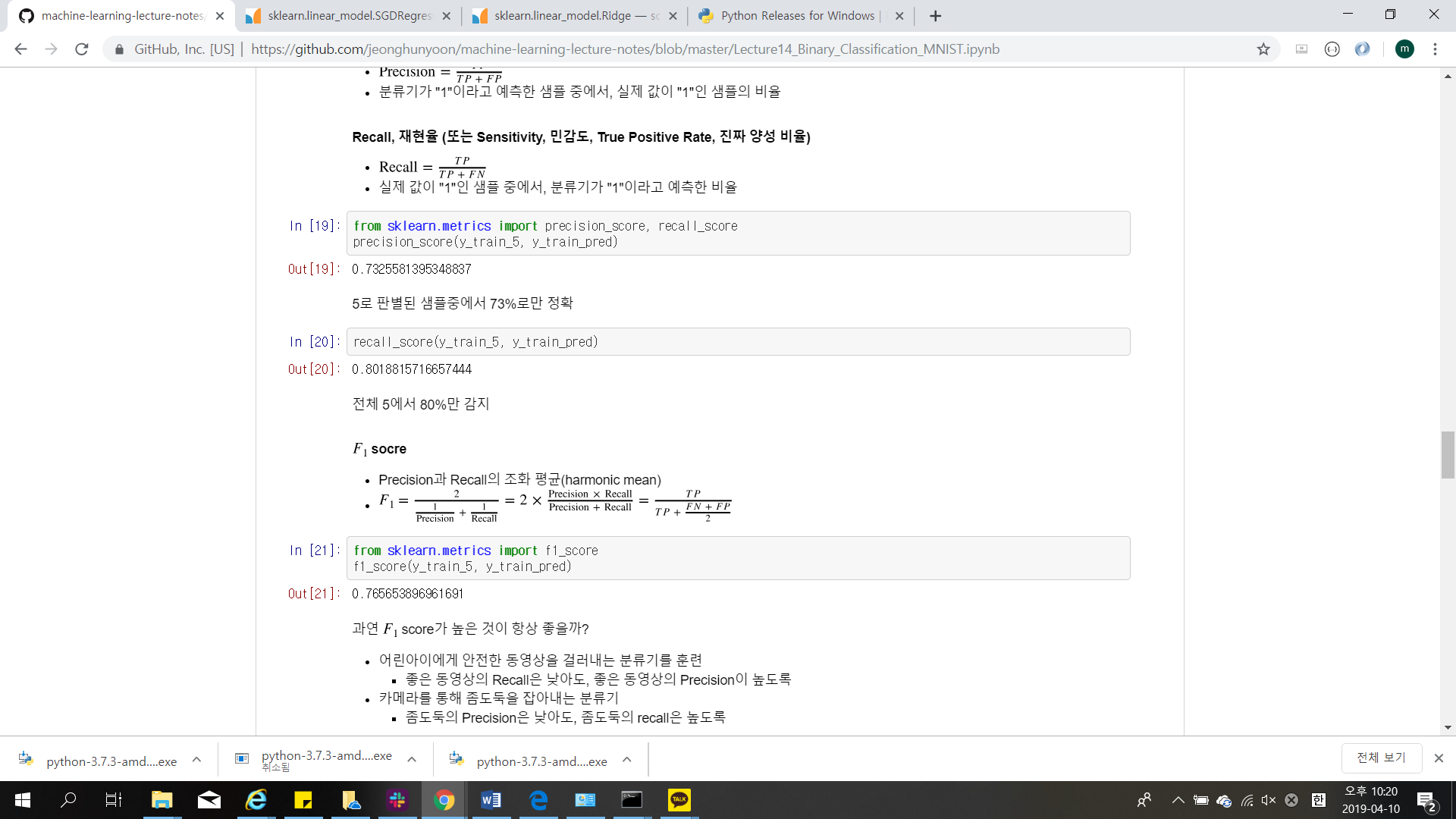
<https://github.com/jeonghunyoon/machine-learning-lecture-notes/blob/master/Lecture14_Binary_Classification_MNIST.ipynb>

## -> 3. Performance 측정

**3-1. 교차 검증을 사용한 정확도 측정**

트레이닝 세트 자체가 워낙 부족하기 때문에 밸리데이션 세트를 하기 어려움.

트레이닝 세트와 테스트 세트만 두고 검증하는 것은 좋지 않음

* 왜? 테스트 세트에 맞춰서 파라미터가 맞춰짐.
* 그래서 밸리데이션 세트에 내 하이퍼테시스에 물들지 않도록 아트가 필요함.
* 여기서 아트란 1) 알엠에스이 2) 멀티-폴드 크로스 벨리데이션(여러 벨리데이션 세트세 만들 것)-케이 폴드를 많이 사용함.
* 트레인 세트가 얼마 안될때는 트레인 세트를 케이개로 나눠서 차례대로 구분해서 하나씩 벨리데이션 세트로 구분하고 퍼포먼스를 분류하다. 그래서 케이번 벨리데이션 가능 하다.\
* 케이 폴드 사용 할 때에는 데이터를 클론할 것. 클론한 데이터를 사용할 것.
* Accuracy 만 가지고 모델의 성능을 판단해서는 안됨.
* 임 발란스 한 데이터를 얼마나 잘 관리하는 지가 능력자 인지 아닌 지 임.
* 
* x-train\_5 는 참값, y\_train\_pred는 예측 값을 넣을 것
* cross\_val\_predict(세개의 모델)-> 세개의 모델중 과반수 이상이 트루이면 트루임(반대의 경우도 마찬가지임)/ 세 개의 모델을 조합해서 사용하는 것임
* 
* 5로 예측 했는데, 실제로 5인 경우
* 
* 실제로 5였는데 5라고 예측한 것이 80인 것임
* F1-Score: precision + recall 조화 평균
* 프리시젼과 리콜은 반비례 관계에 있음