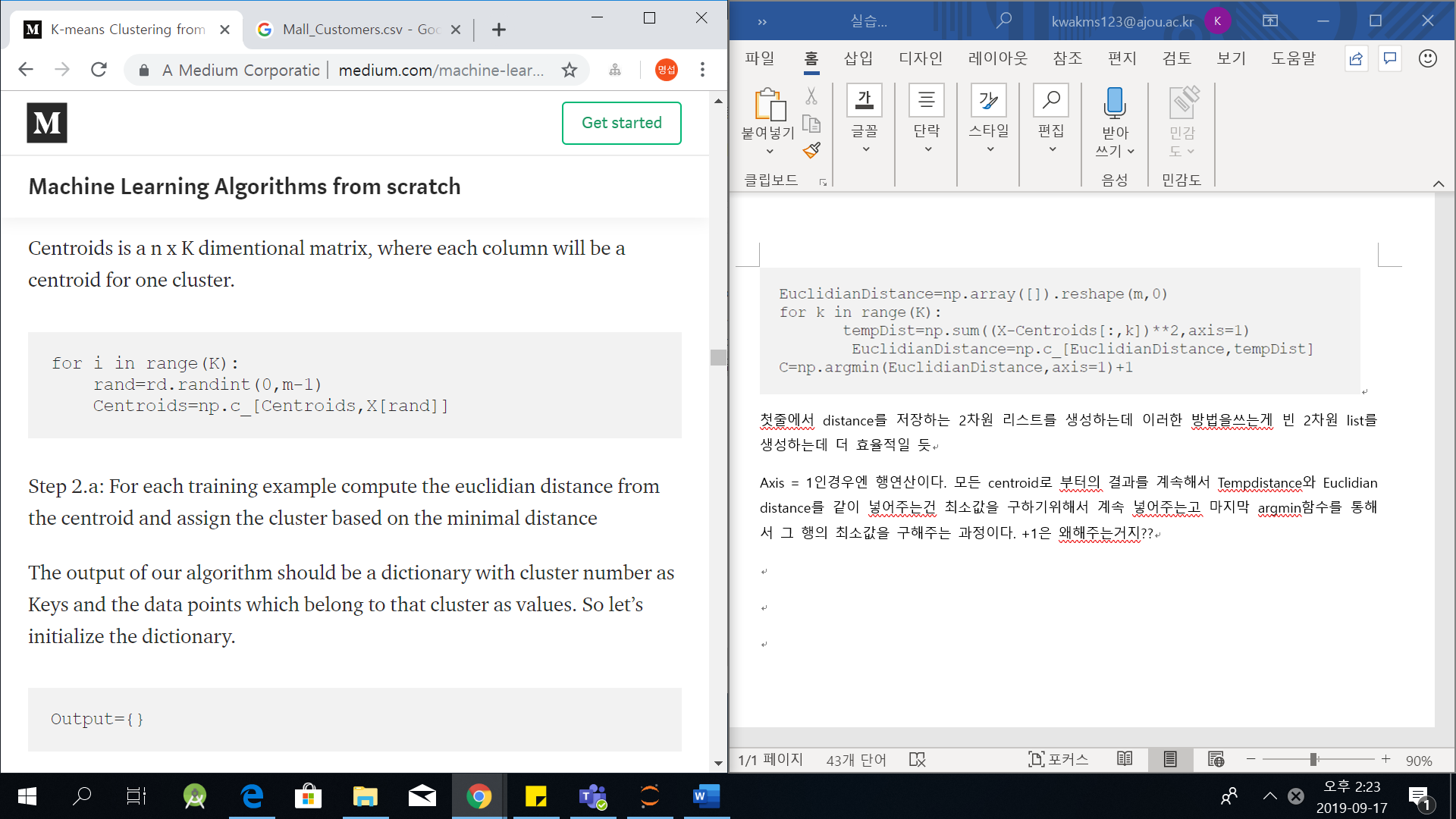
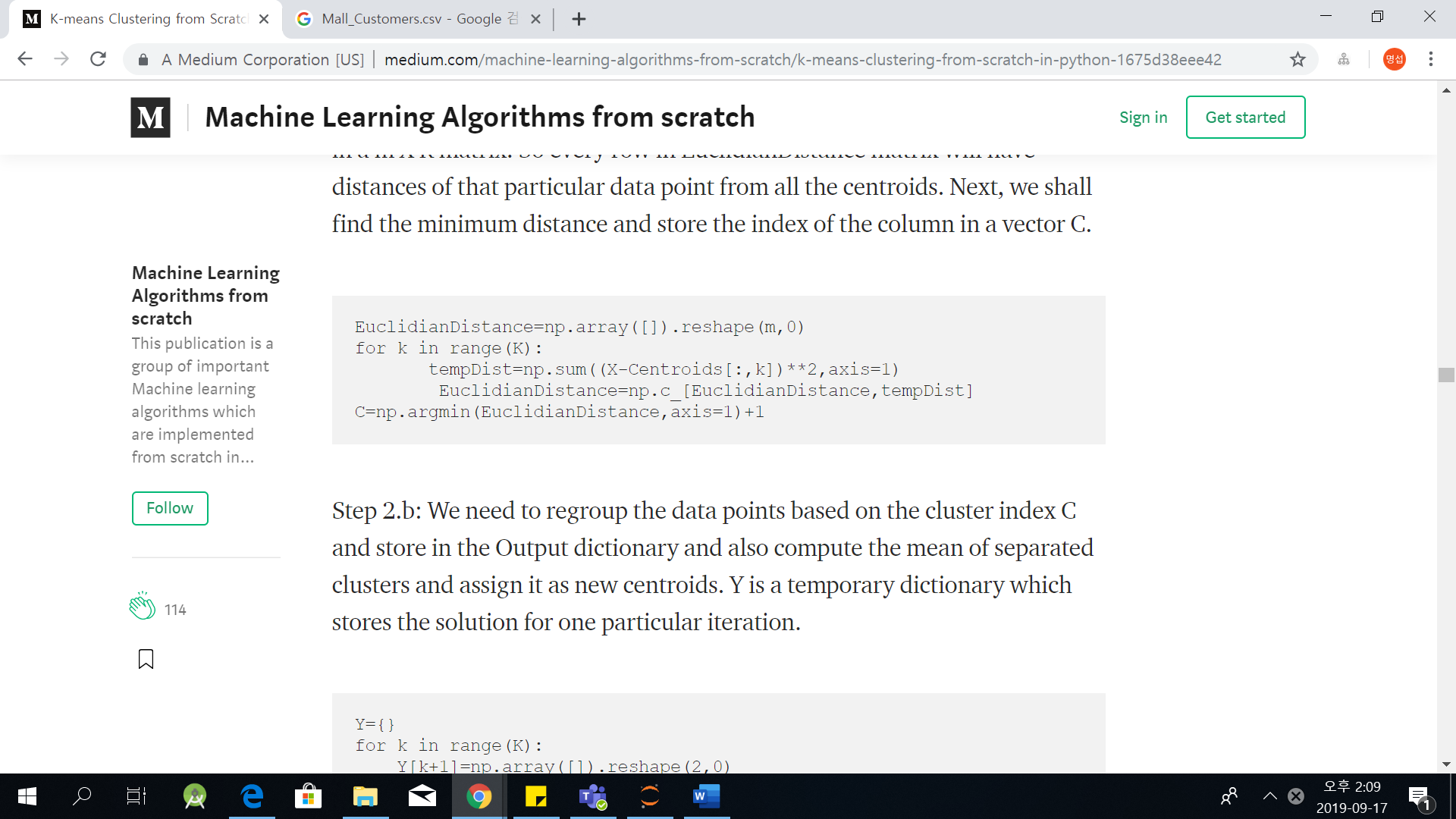
실습 보고서

201520991 곽명섭

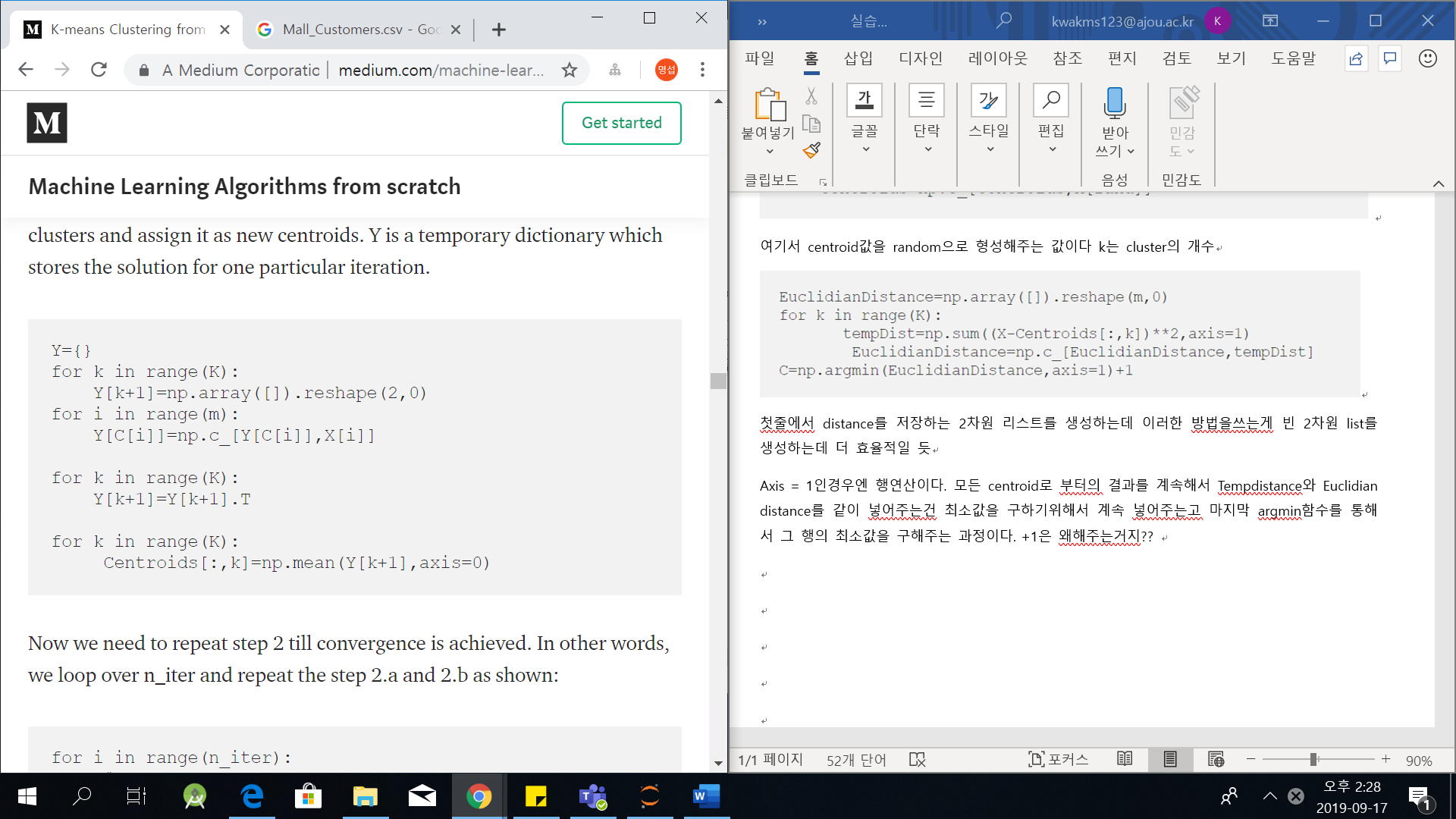


여기서 random으로 생성된 centroid를 각 column이 한 cluster의 centroid가 될 것이다. k는 cluster의 개수를 의미한다.



첫 줄을 보면 distance를 저장하는 비어 있는 2차원 리스트를 생성하는 과정이다. 사실 파이썬을 다루어 본지 얼마되지 않아서 이렇게 빈 2차원 리스트를 생성하는 것을 처음 보아서 나중에 유용하게 쓸 수 있을 것 같다.

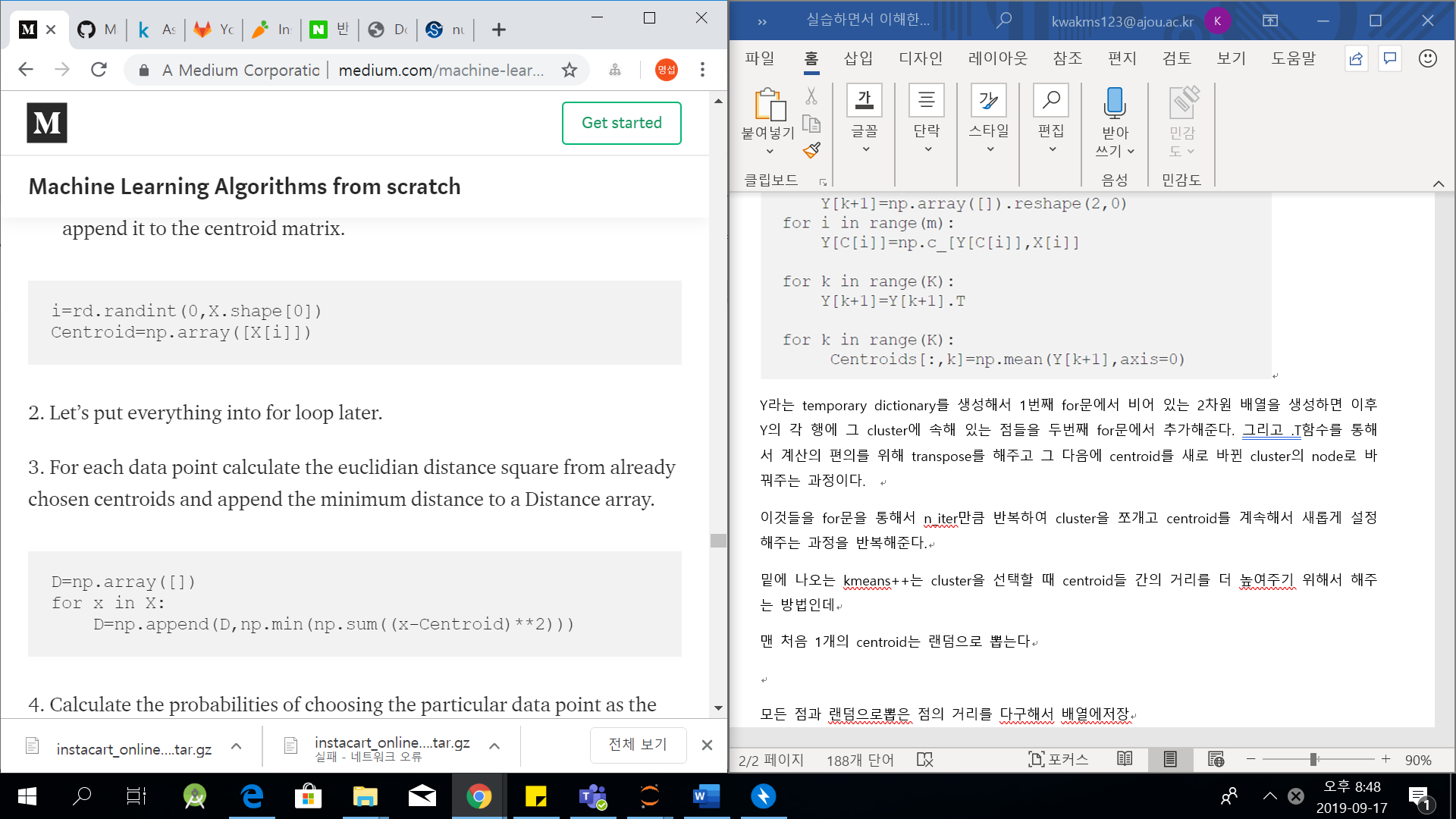
Axis = 1인 경우엔 행연산을 의미한다. 그리고 이 코드는 centroid로부터의 각 점까지 distance값의 결과를 계속해서 EuclidianDistance라는 리스트에 계속해서 추가해준다. np.c\_같은 경우엔 matrix를 transpose시킨 상태로 추가를 해주는 함수이다. EuclidianDistance 같은 경우는 index값이라고 보면 되고 tempDist에는 각 점과 모든 centroid간의 distance값이 저장이 되어있다고 보면 된다. 그리고 C에는 index를 1부터 넣어 주기 위해서 1 마지막에 1을 더해준다.



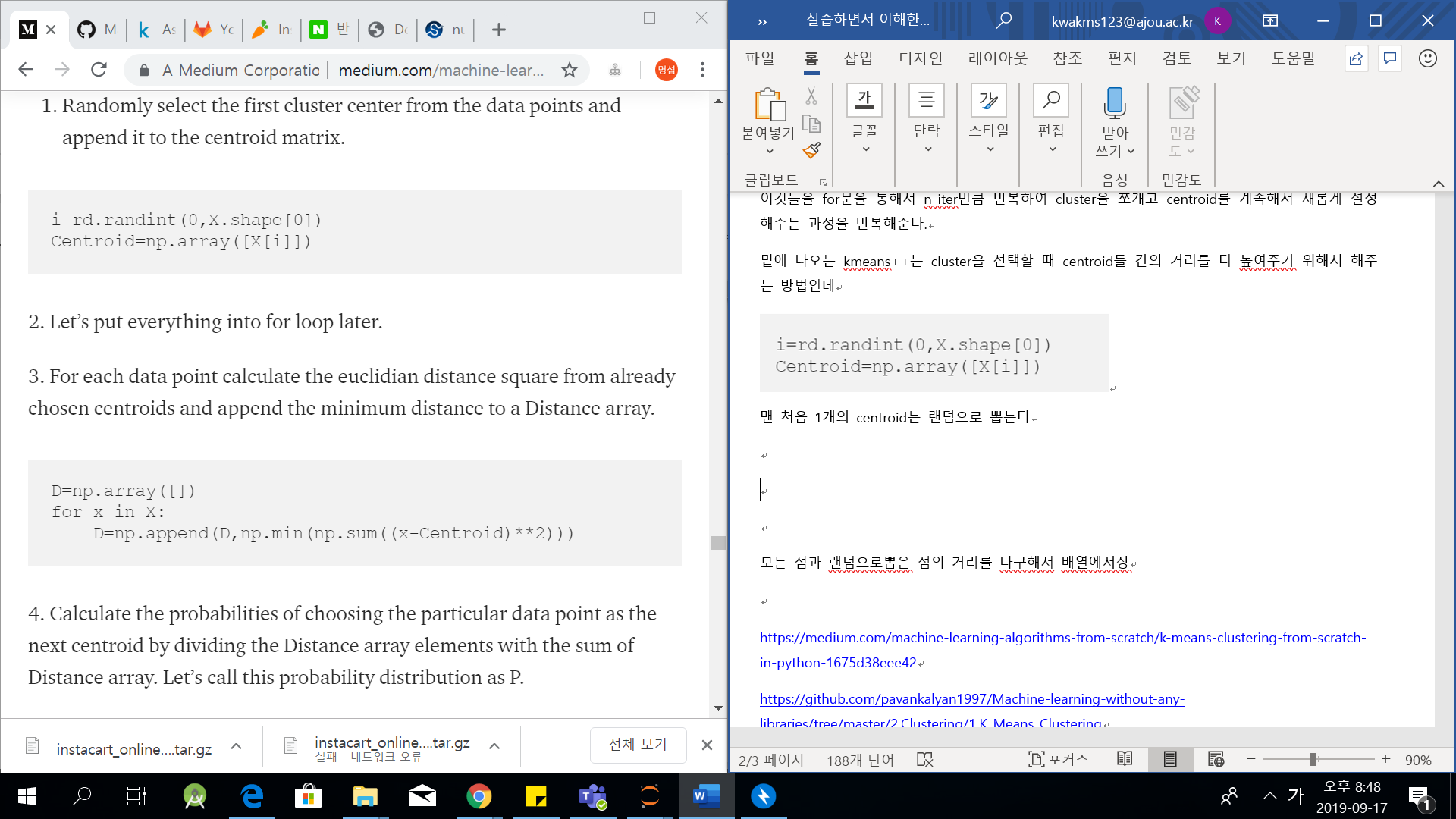
Y라는 temporary dictionary를 생성해서 1번째 for문에서 비어 있는 2차원 배열을 생성하면 이후 Y의 각 행에 그 cluster에 속해 있는 점들을 두번째 for문에서 추가해준다. 그리고 .T함수를 통해서 계산의 편의를 위해 transpose를 해주고 그 다음에 centroid를 새로 바뀐 cluster의 node로 바꿔주는 과정이다.

이것들을 for문을 통해서 n\_iter만큼 반복하여 cluster을 쪼개고 centroid를 계속해서 새롭게 설정해주는 과정을 반복해준다.

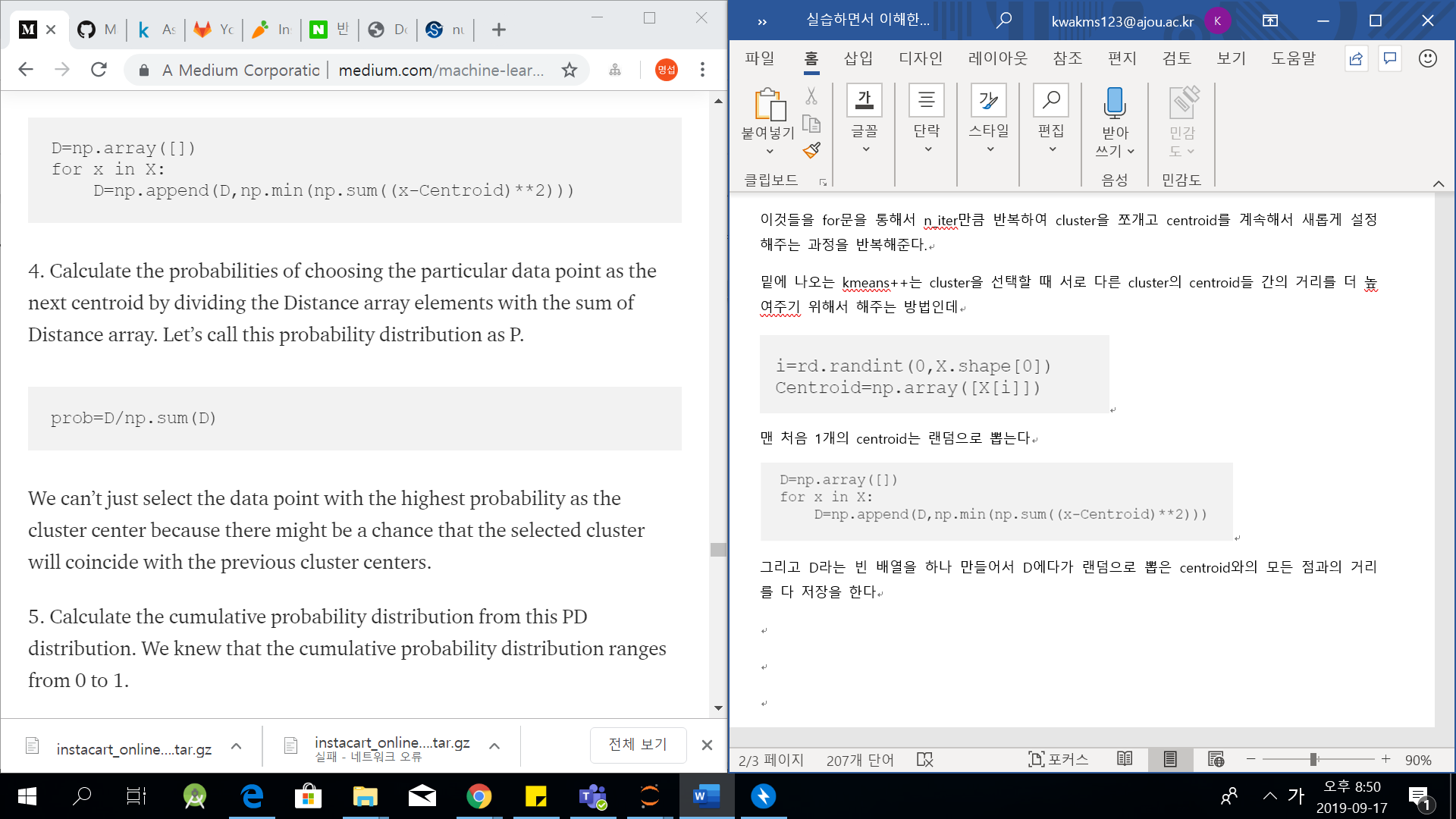
밑에 나오는 kmeans++는 cluster을 선택할 때 서로 다른 cluster의 centroid들 간의 거리를 더 높여주기 위해서 해주는 방법인데



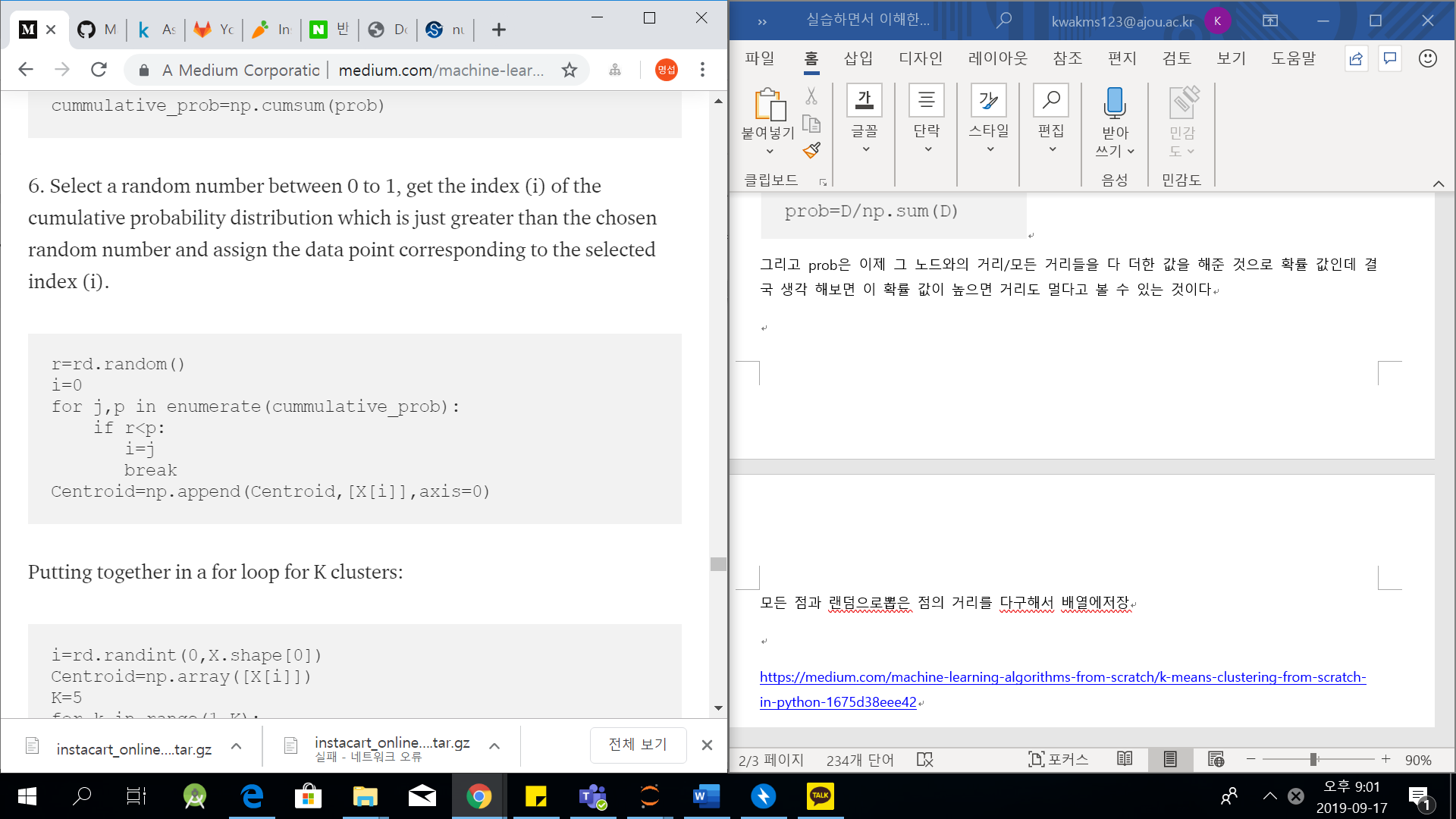
맨 처음 1개의 centroid는 랜덤으로 뽑는다



그리고 D라는 빈 배열을 하나 만들어서 D에다가 랜덤으로 뽑은 centroid와의 모든 점과의 거리를 다 저장을 한다



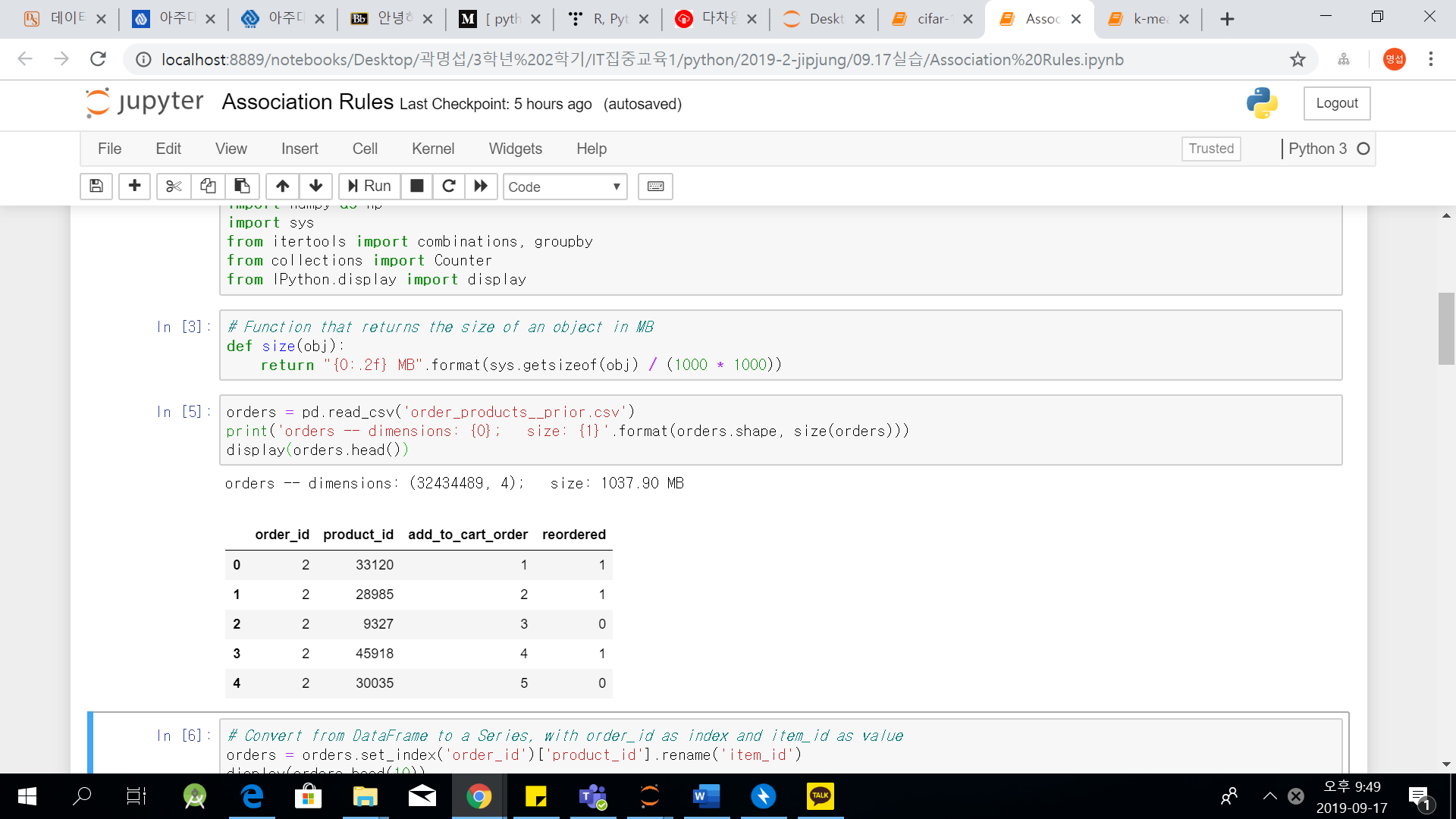
그리고 prob은 이제 그 노드와의 거리/모든 거리들을 다 더한 값을 해준 것으로 확률 값인데 결국 생각 해보면 이 확률 값이 높으면 거리도 멀다고 볼 수 있는 것이다



여기가 약간은 어려운 부분인데 이 부분이 0과 1사이의 random값을 설정을 해주는데 어떤 cummulative한 함수에서 기울기가 더 가파를수록 r이 할당 받은 확률자체가 더 낮기 때문에 이 과정은 결국 더 centroid간의 거리가 퍼질 확률이 높게 잡아주는 과정이다.

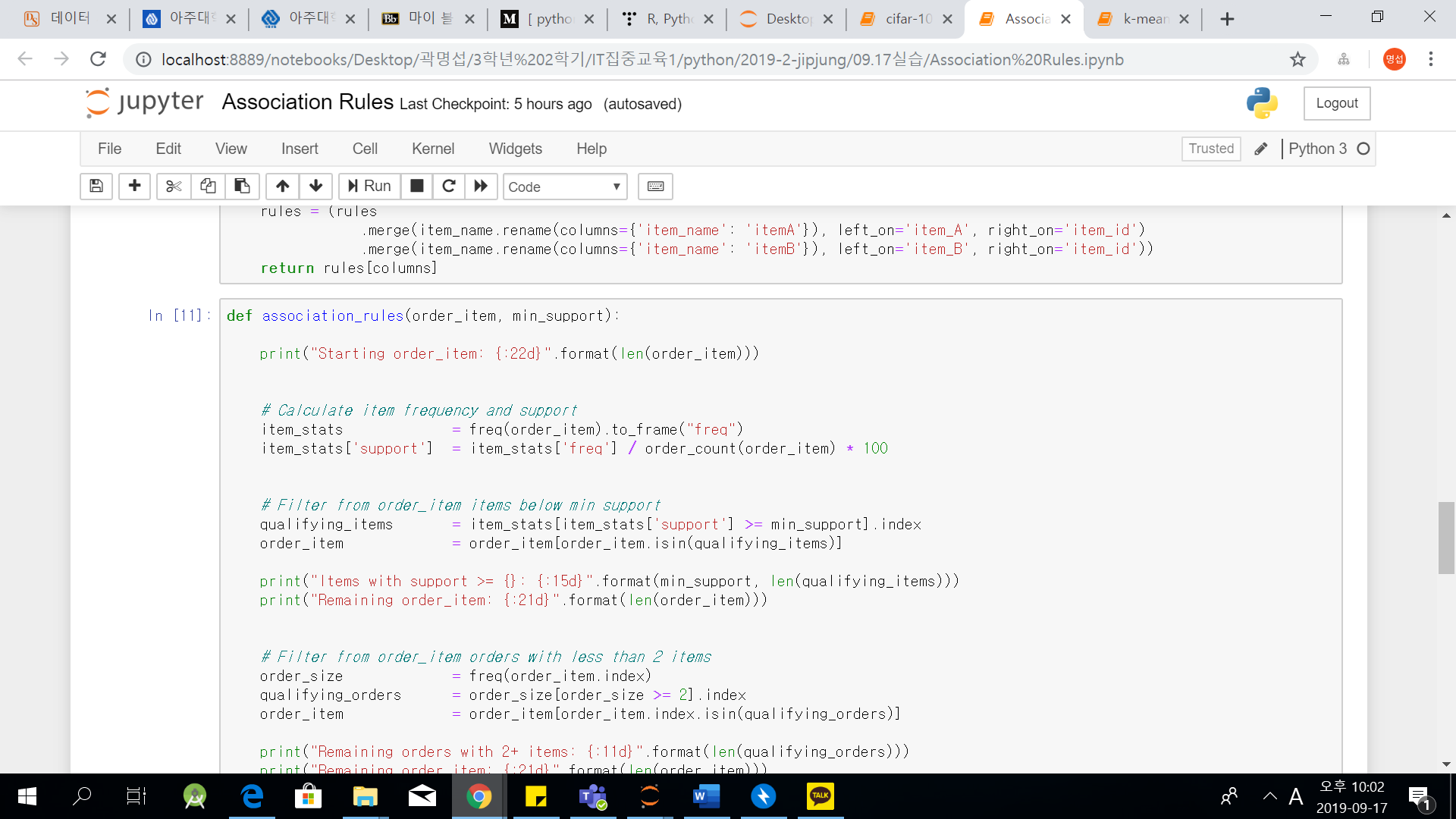
그리고 실제로 kmeans++를 사용해서 하면 결과적으로도 더 성능이 좋아지는 것을 알 수 있다.

Association rules

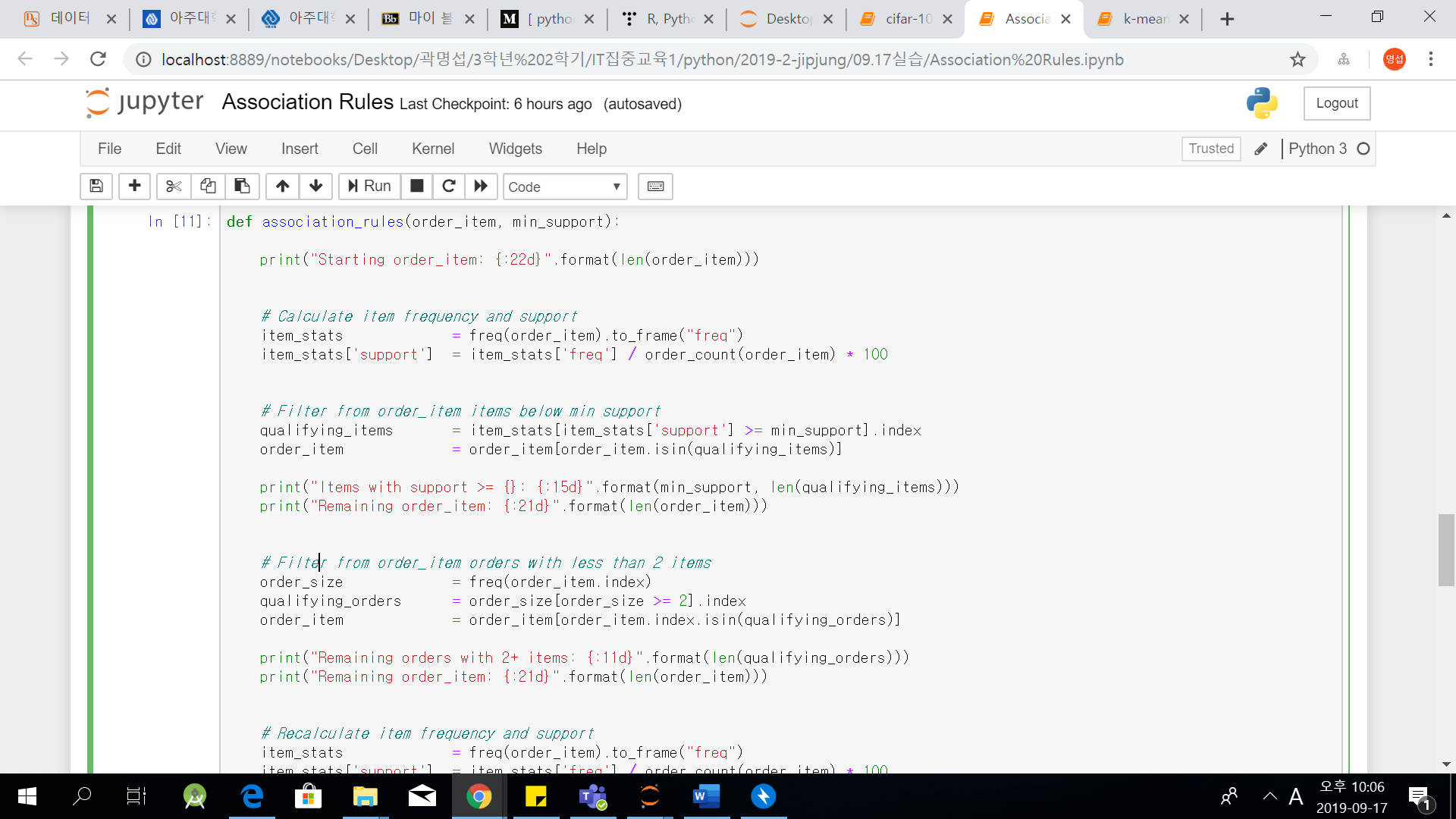


csv파일을 읽어 온 후에 display로 보여주는 과정이다

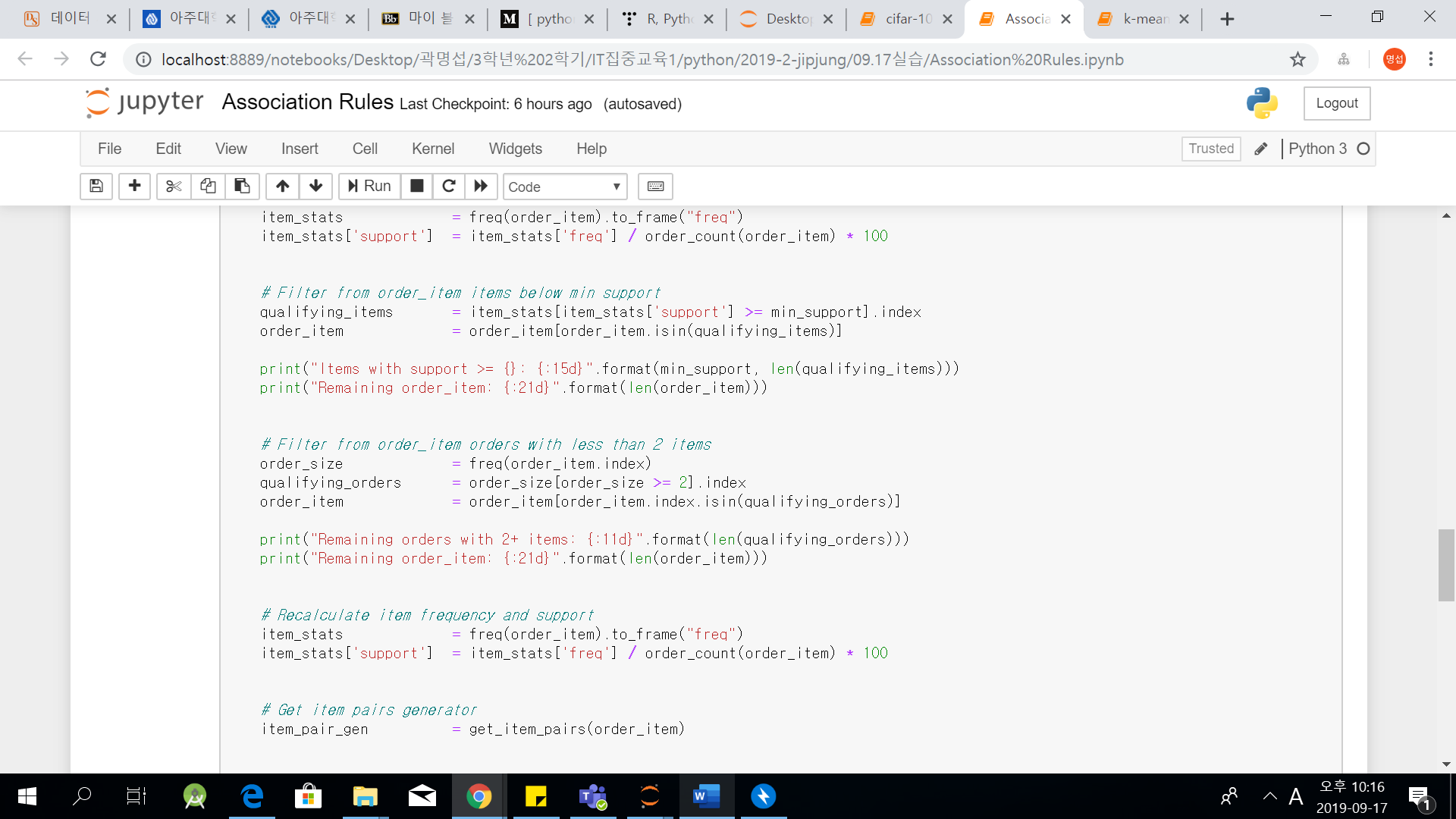
이후에는 association rule함수의 핵심적인 것들만 설명을 하자면



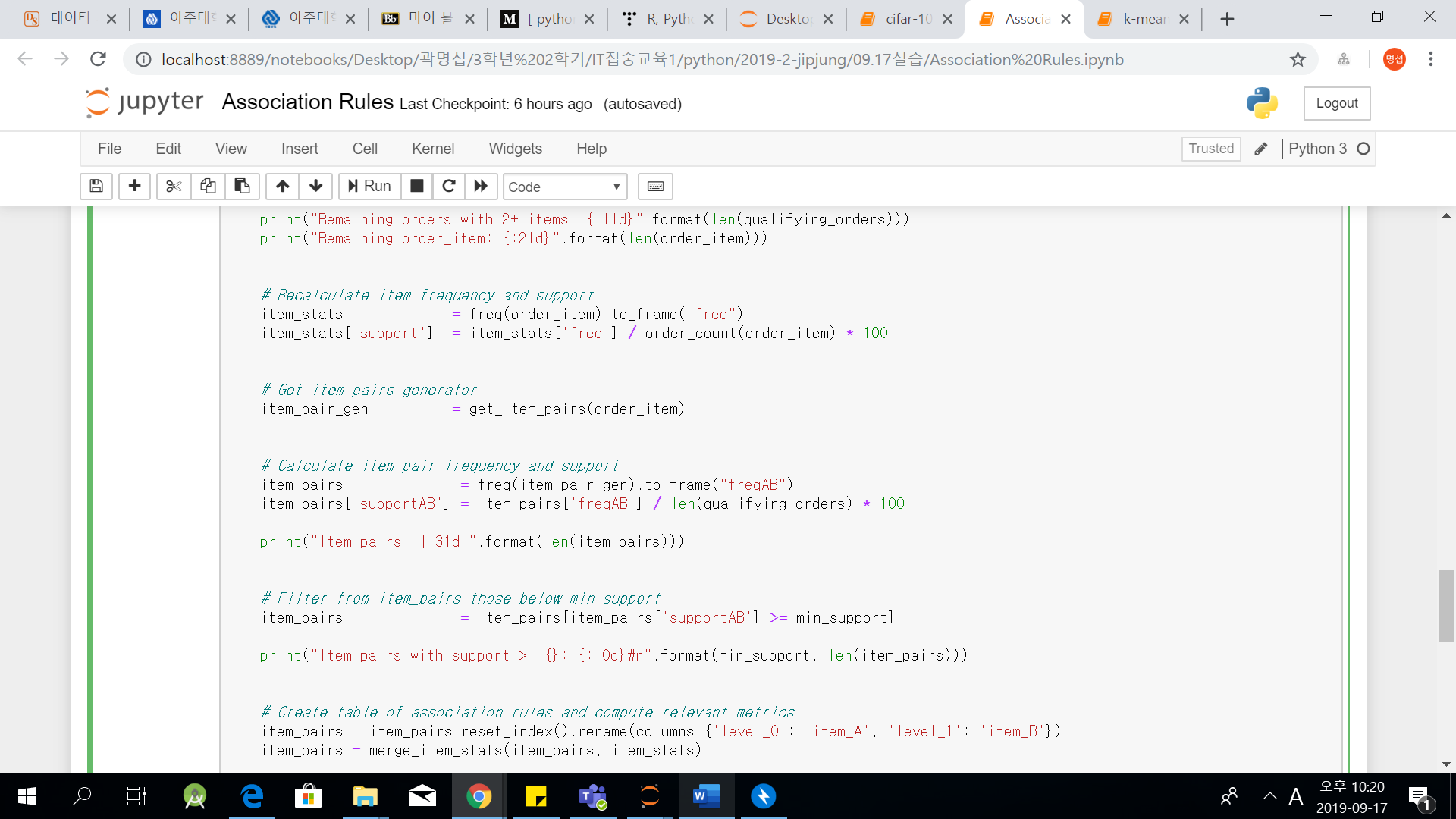
Item의 frequency와 support를 계산해주는 과정인데 support같은경우엔 빈도수를 확률로 나타낸 것이다. Support는 관계를 설정하기 위한 상품들이 동시에 발생될 확률을 의미 하며, Confidence는 특정 상품이 선택된 뒤, 다른 상품이 선택될 확률을 의미 합니다.



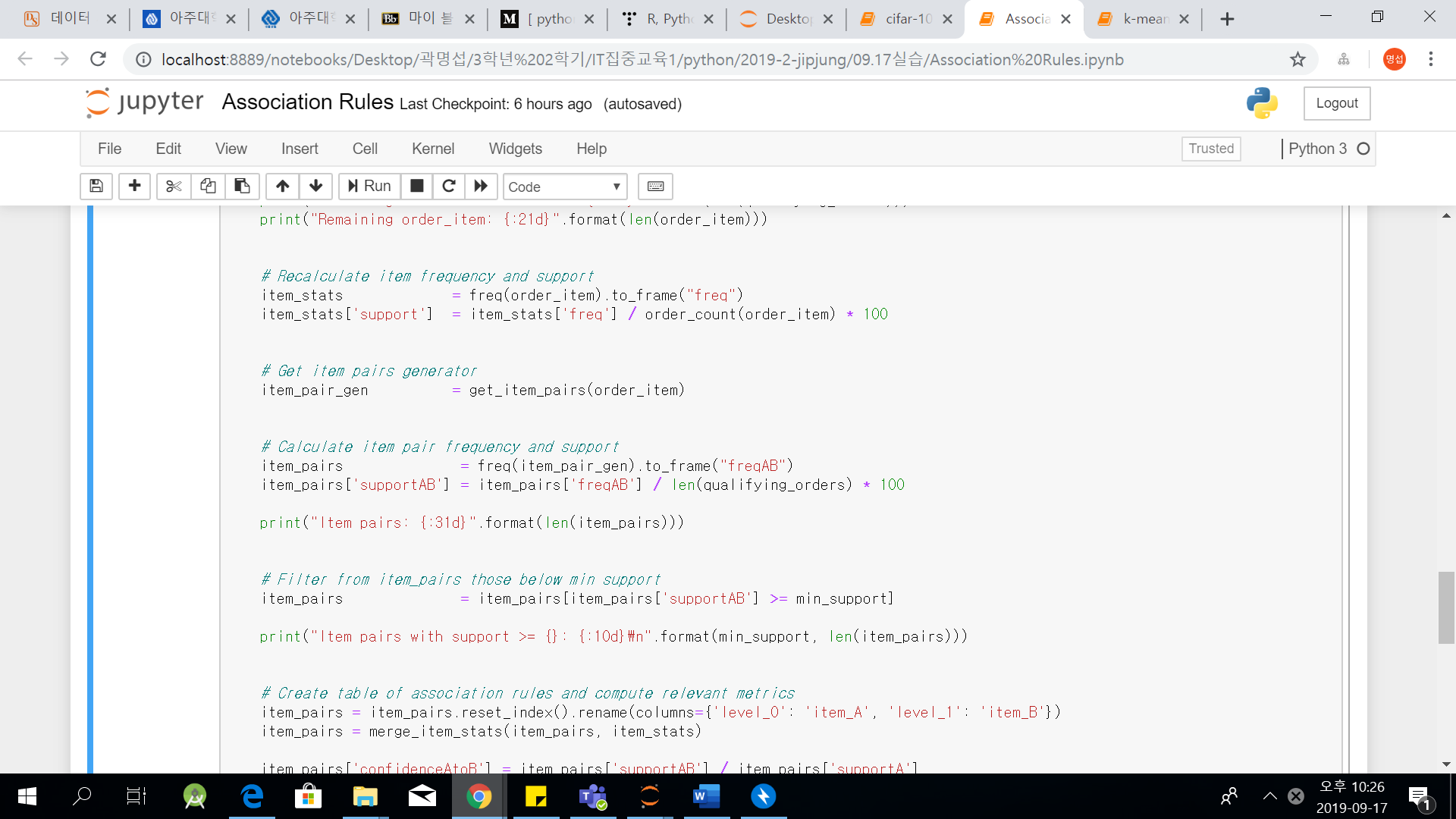
Item의 support가 min support이상인것들로만 구성되도록 filtering하는 과정이다. 그리고 filtering한 item을 order item에 넣어준다.



2개의 item이상인 order만을 filtering 하는 과정이다



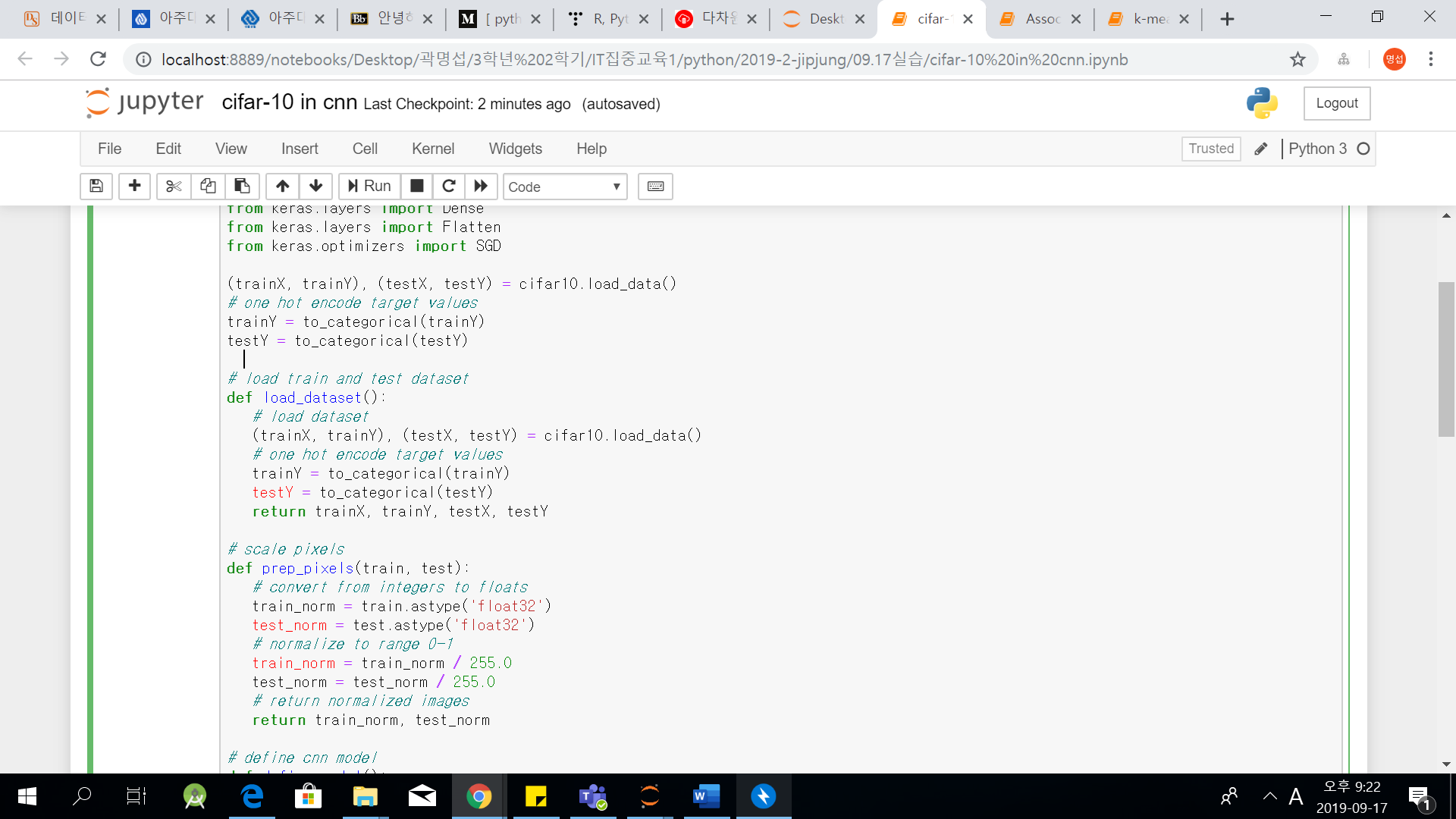
그리고 item의 frequency와 support를 다시 계산해주고



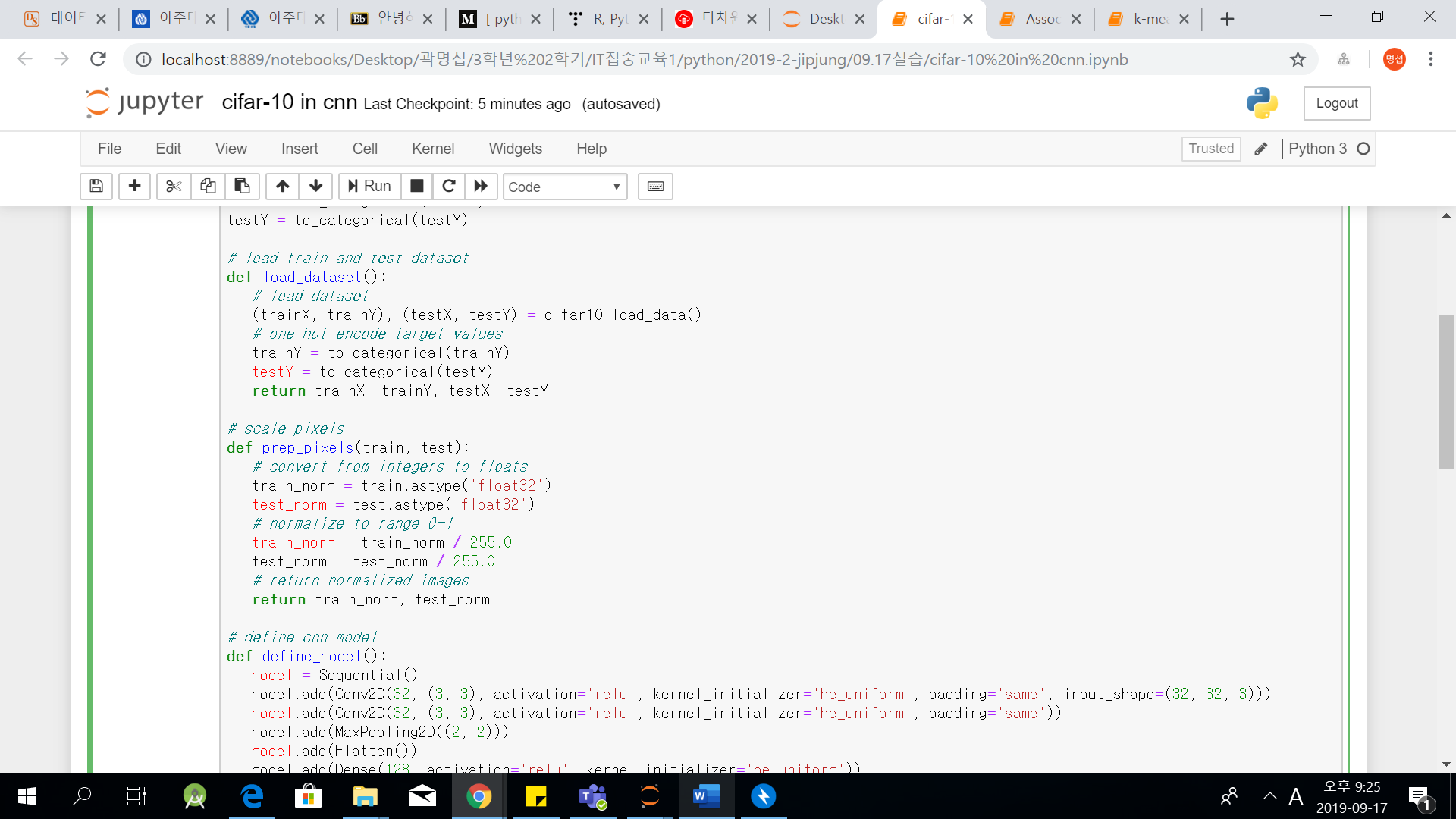
Item pair의 frequency와 support를 계산해준다

Filtering을 하고 그 filtering한 결과를 토대로 pair을 설정해주고 그것의 확률을 구하는 과정인것이다.

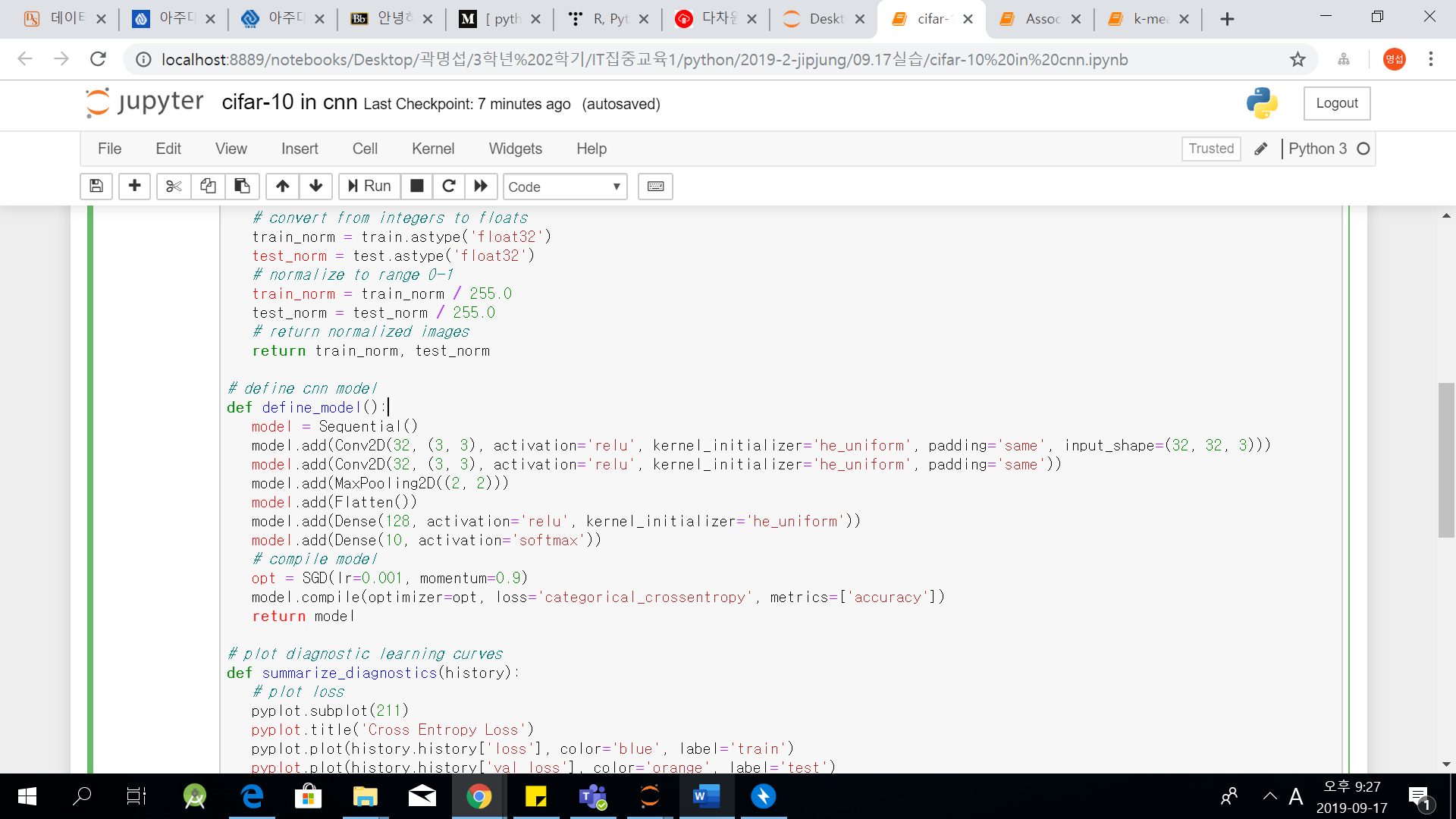
CNN IN CIFAR-10



Dataset을 불러와서 train set과 testset으로 나누어 준다. 저번에 실습하였던 DNN의 CIFAR-10과 큰 차이가 없는것으로 보인다.



Prep\_pixel 함수는 이미지의 각 pixel들을 float 32 자료형으로 바꿔주고 255로 나누어 주어 0과 1사이의 값으로 표준화를 시켜주는 과정이다.



이 과정은 model을 형성하는 과정인데 여기에선 convolution을 2번하고 MaxPool을 1회 한다. 그이후에 flatten까지도 한번 해주고 있다. 그 이후 hidden layer을 하나 또 추가해 준다.

Convolution 같은 경우 맨 처음 파라미터인 32 같은경우 convolution의 필터의 수를 의미하며 (5,5)같은 경우 convolution 커널의 행과 열을 의미한다. activation 함수는 relu로 사용하며 initializer는 he\_uniform을 사용하고 input\_shape는 샘플 수를 제외한 입력 형태를 정의한다. 이렇게 여러 parameter을 쓴 방식들을 modeling한다.

Optimizer는 sgd로 쓰고 있고 learningrate를 0.001, momentum값은 0.9로 주어진다.

Lossfunction은 crossentropy를 사용한다.

<https://medium.com/machine-learning-algorithms-from-scratch/k-means-clustering-from-scratch-in-python-1675d38eee42>

<https://github.com/pavankalyan1997/Machine-learning-without-any-libraries/tree/master/2.Clustering/1.K_Means_Clustering>

<https://www.kaggle.com/datatheque/association-rules-mining-market-basket-analysis>