**차원에 따른PCA &LDA performance**

**-using ORL database**

인공지능융합학과(석사과정)

학번: 2021123173

이름: 송 우 석

1. **Introduction**
   1. Curse of dimensionality(차원의 저주):

데이터의 차원이 높아질수록 알고리즘의 실행이 아주 까다로워지는 일을 말한다. Feature의 개수가 늘어나면서 classifier의 Complexity(복잡도)를 증가시키기 때문이다.

또한 Overfitting(과적합)과도 관련이 있다. 복잡도가 높아지면서 training 데이터 세트에 대한 error는 줄어들지만, test 데이터 세트에 대한 error는 커질 수 있다.

Curse of dimensionality를 해결하기 위해선 ‘Assuming conditional independence’, ‘Assuming equal covariance for all classes’ 등의 Heuristic solution이 있고 또한 더 좋은 방법인 ‘Incorporating prior knowledge’ , ‘Reducing dimensionality’ 등의 방법도 있다.

* 1. Dimensionality reduction(차원축소):

Curse of dimensionality를 해결하기 위해 차원을 축소시키는 방법으로 크게 ’feature selection’ 과 ‘feature extraction’이 있다.

Feature extraction은 보통 선형변환에 의해 이뤄지는데 밑의 그림1은 linear feature extraction을 보여준다. y는 x(feature)의 정사영으로서 w가 얼마나 x의 feature을 잘 요약하는가를 결정한다.

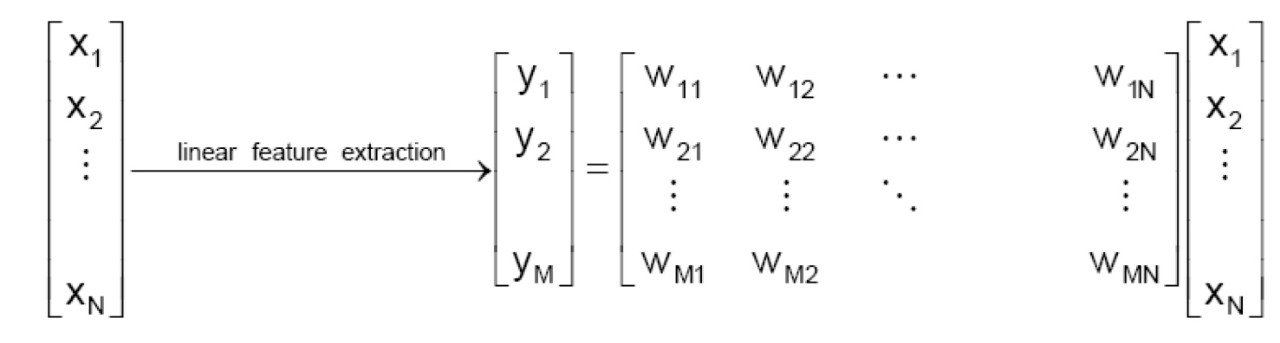


그림 1

* 1. Feature extraction(특성 추출):

특성 추출은 objective function에 따라서 2가지로 나뉜다. 신호자체의 성분을 인식하는 ‘signal representation’과 분류를 위한 ‘Classification’ 있는데 예를 들어 PCA와 LDA가 대표적인 Feature extraction 방법에 속한다.

* + 1. PCA (Principal Component Analysis):

주성분 분석은 고차원의 데이터를 저차원의 데이터로 환원시키는 기법을 말한다. 이 때 서로 연관 가능성이 있는 고차원 공간의 표본들을 선형 연관성이 없는 저차원 공간(주성분)의 표본으로 변환하기 위해 직교변환을 사용한다. 데이터를 한개의 축으로 정사영시켰을 때 그 분산이 가장 커지는 축을 첫 번째 주성분, 두 번째로 커지는 축을 두 번째 주성분으로 놓이도록 새로운 좌표계로 데이터를 선형변환 한다. 이와 같이 표본의 차이를 가장 잘 나타내는 성분들로 분해함으로써 데이터 분석에 여러가지 이점을 제공한다. 이 변환은 첫째 주성분이 가장 큰 분산을 가지고, 이후의 주성분들은 이전의 주성분들과 직교한다는 제약 아래에 가장 큰 분산을 갖고 있다는 식으로 정의되어있다. 중요한 성분들은 공분산행렬의 고유 벡터이기 때문에 직교하게 된다. [2]

밑의 그림 2는 PCA 수식을 손으로 정리해 놓은 그림이다.

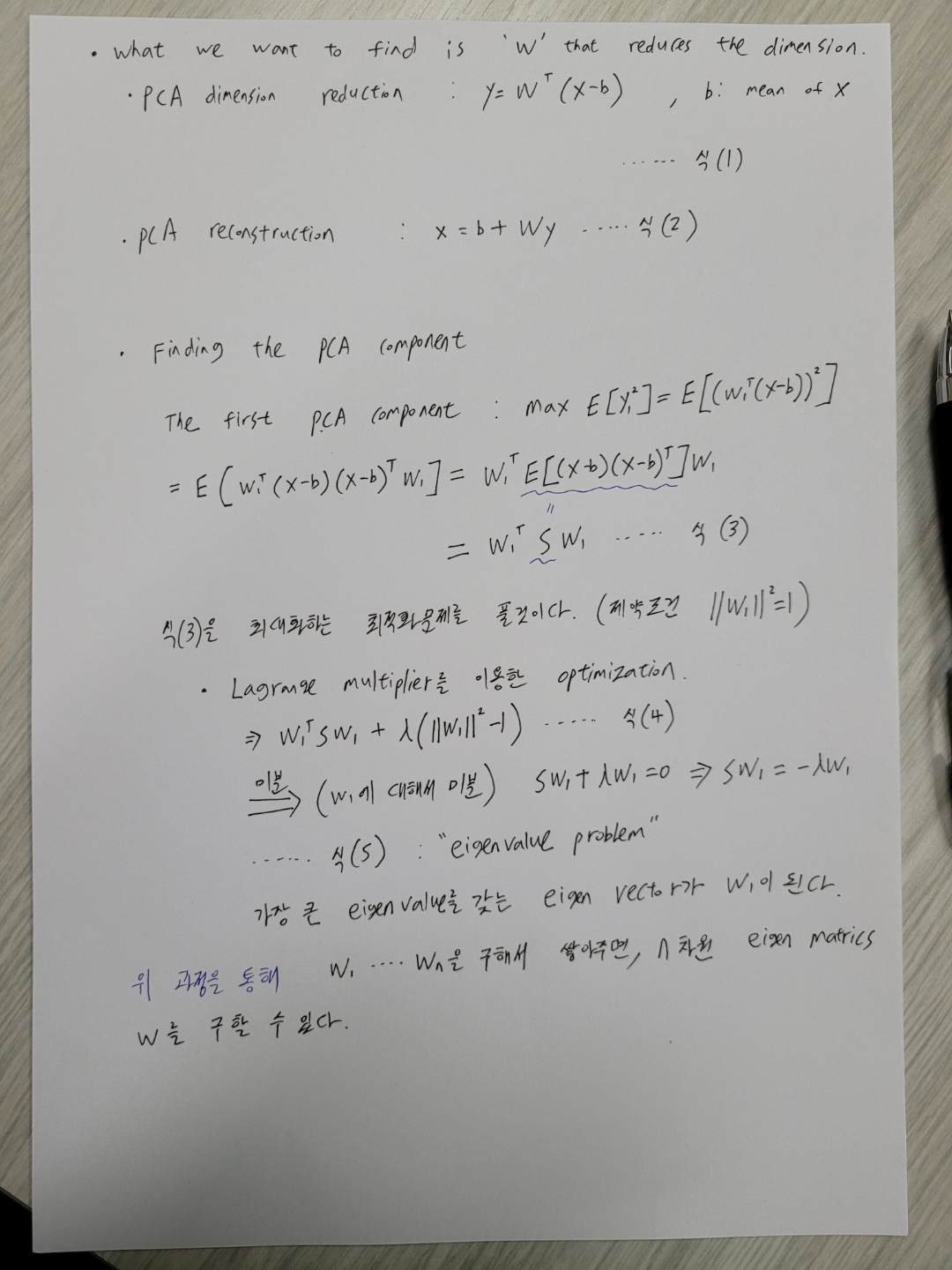


그림 2

1.3.2 LDA(Linear discriminant analysis):

Lda는 데이터 분포를 학습해 결정경계를 만들어 데이터를 분류하는 모델이다. 데이터를 특정 한 축에 사영한 후에 두 범주를 잘 구분할 수 있는 직선을 찾는 걸 목표로 한다. 사영 후 두 범주의 중심(평균)이 서로 멀도록, 그 분산이 작도록 하는 직선을 찾는 것이 목표이다. [3]

밑의 그림 3는 LDA 수식을 손으로 정리해 놓은 그림이다.

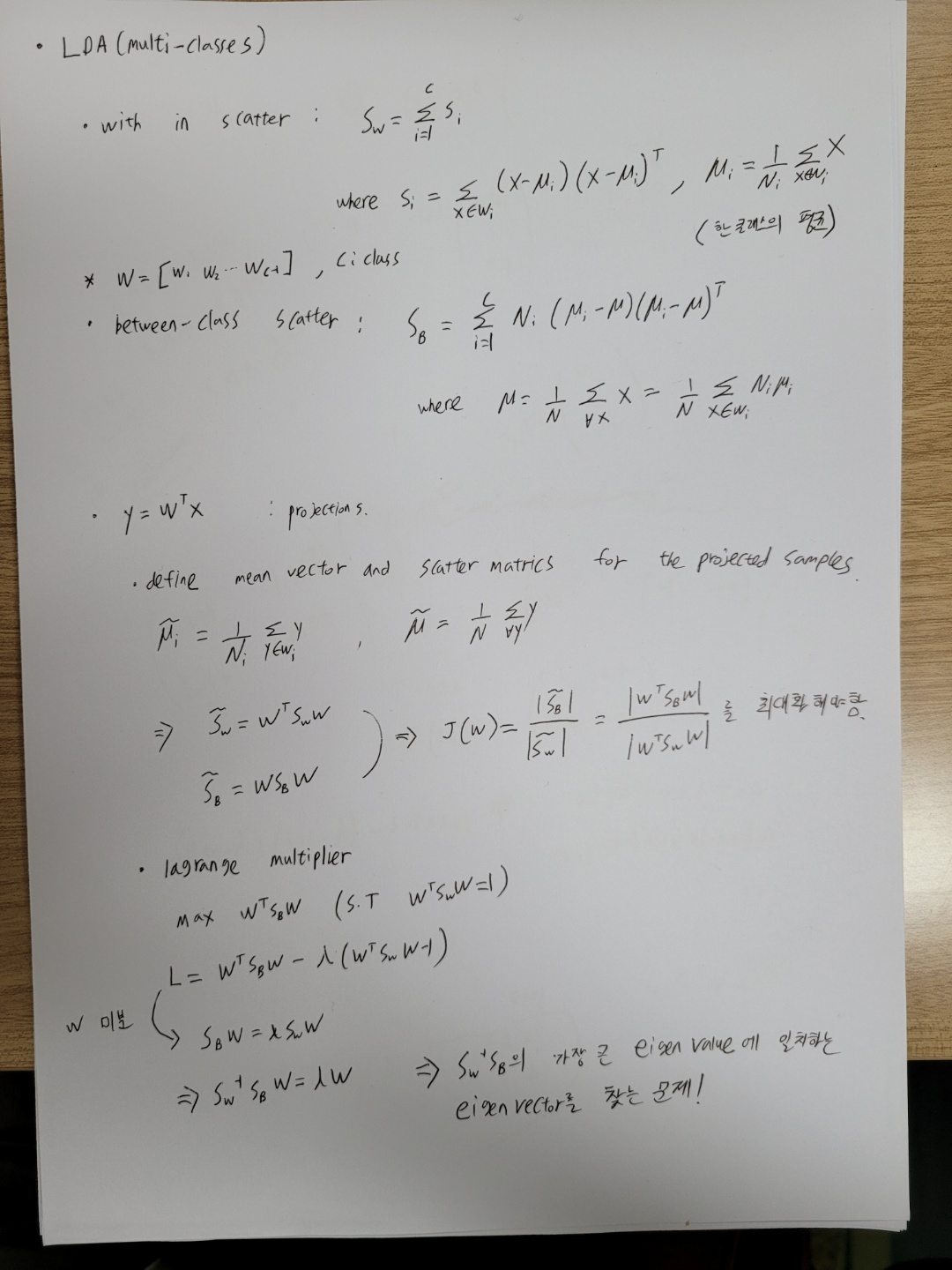


그림 3

1. **Method**

*2.1 데이터 업로드, 전처리*

(1)Orl database(사람 40명 \* 각 사람당 10장 =400장의 사진 pixcel값)를 다운로드 받고 Google의 Colab(파이썬)을 이용해서 데이터를 업로드 했다.

(2) Training data와 test data를 사람 수를 기준으로 나눴다. (36명/4명)

*2.2 PCA*

(1) 차원 축소를 통해 얻은 W 행렬 (eigen matrics)를 2차원 그래프에 찍어서 eigenface를 얻는다.

(2) Dimension reconstruction을 통해 이미지를 복원해본다.

(3) Test 데이터의 이미지를 3차원 PCA 공간에 찍어본다.

(4) Test 데이터(4명 \*10장)를 gallery(4명\*7장)와 query(4명 \*3장) 이미지로 나눈다.

(5) Test 이미지를 차원 축소시키고, 각각의 query 이미지에 대해서 가장 가까운 gallery 이미지 7개를 찾았다. (7-nearest neighbor)

(6) 축소시키는 차원 d= 3,4,10,50,100,1000으로 바꿔가면서 실험을 했고 그에 대한 정확도를 표로 뽑아냈다. (10-fold cross validation 사용)

*2.3 LDA*

(1) 차원축소를 통해 fisher-face를 얻는다

(2) Test 데이터의 이미지를 3차원 LDA 공간에 찍어본다.

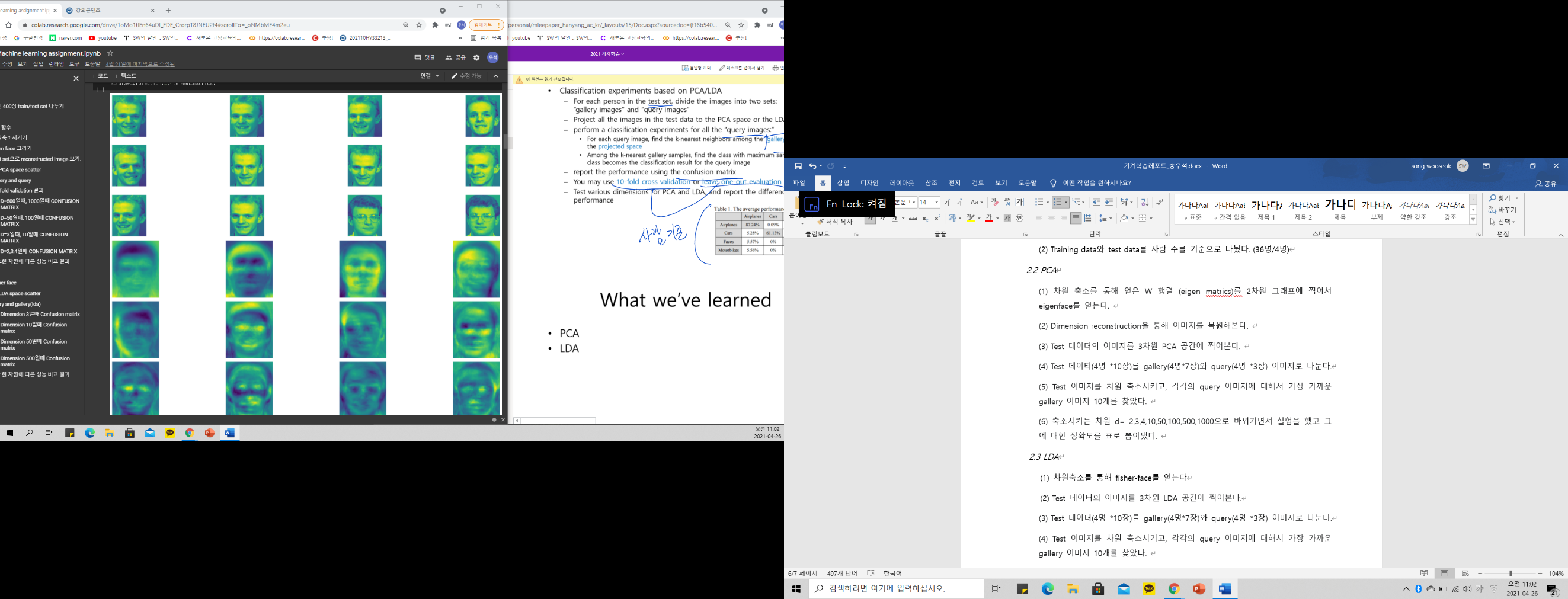
(3) Test 데이터(4명 \*10장)를 gallery(4명\*7장)와 query(4명 \*3장) 이미지로 나눈다.

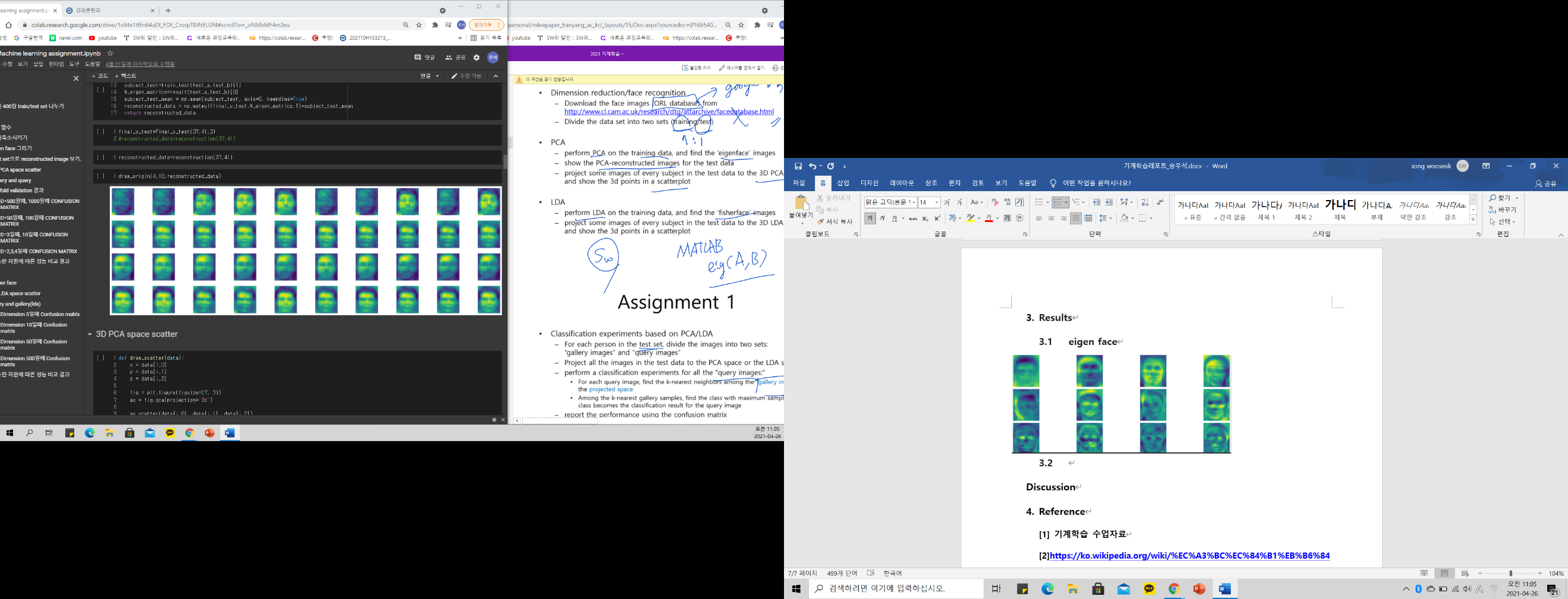
(4) Test 이미지를 차원 축소시키고, 각각의 query 이미지에 대해서 가장 가까운 gallery 이미지 7개를 찾았다. (7-nearest neighbor)

(5) 축소시키는 차원 d= 3,10,50,100으로 바꿔가면서 실험을 했고 그에 대한 정확도를 표로 뽑아냈다. (10-fold cross validation 사용)

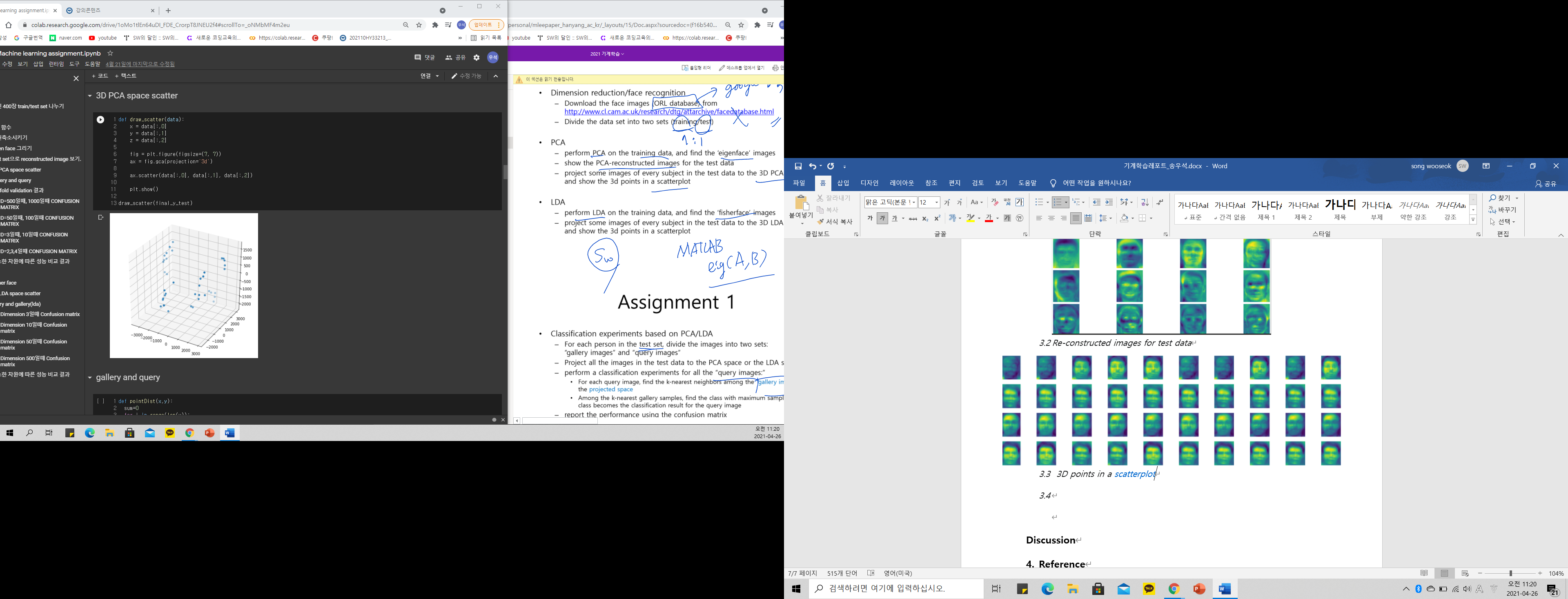
1. **Results**

**3.1PCA**

* 1. *.1 Eigen face images for training data*

*3.1.2**Re-constructed images for test data*

* + 1. *3D points in a scatterplot*



*3.1.4 Confusion matrix for PCA (depends on dimension)*

*Ex) d=3일 때*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *Subject1* | *Subject2* | *Subject3* | *Subject4* |
| *Subject1* | 92.86 | 3.33 | 1.42 | 2.38 |
| *Subject2* | 0 | 96.19 | 2.38 | 1.43 |
| *Subject3* | 0 | 3.33 | 90.00 | 6.67 |
| *Subject4* | 1.90 | 5.23 | 7.62 | 85.24 |

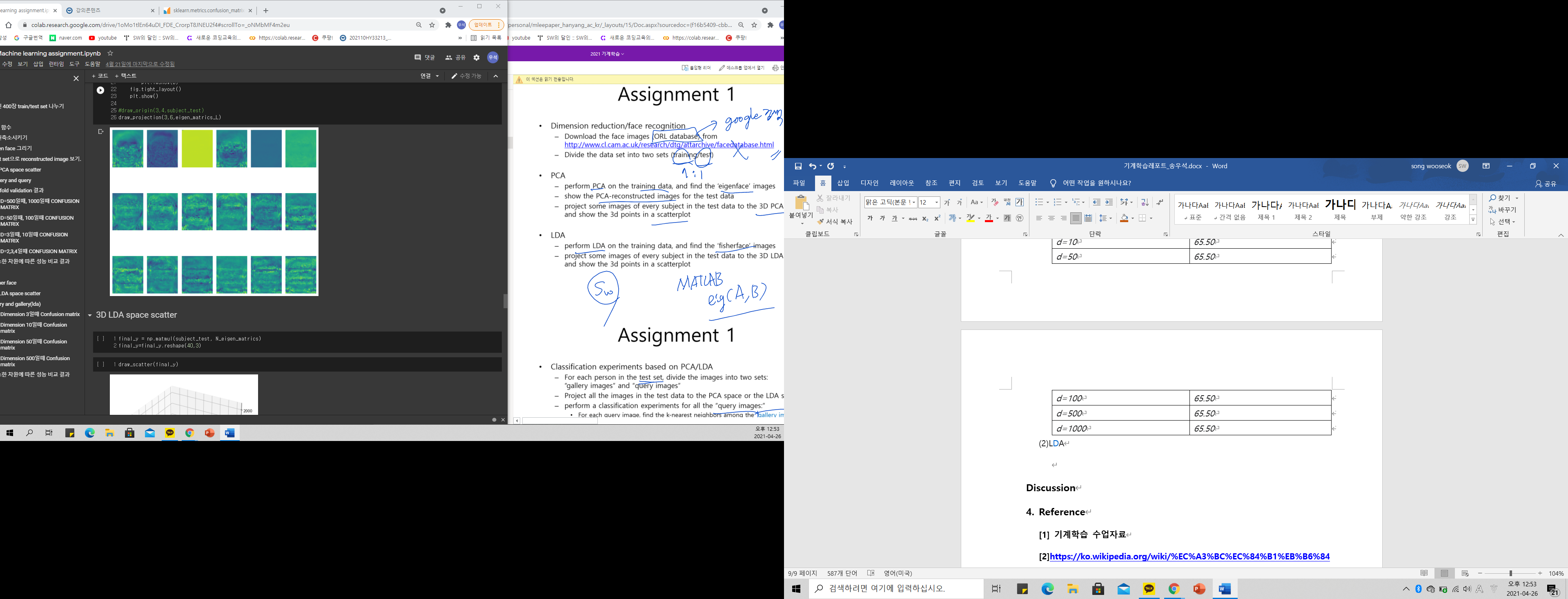
정확도 : (92.86+ 96.19+ 90.00+ 85.24) / 4 =91.07

이런 식으로 d=3,4,10,50,100,500,1000 일 때의 정확도를 구했다.

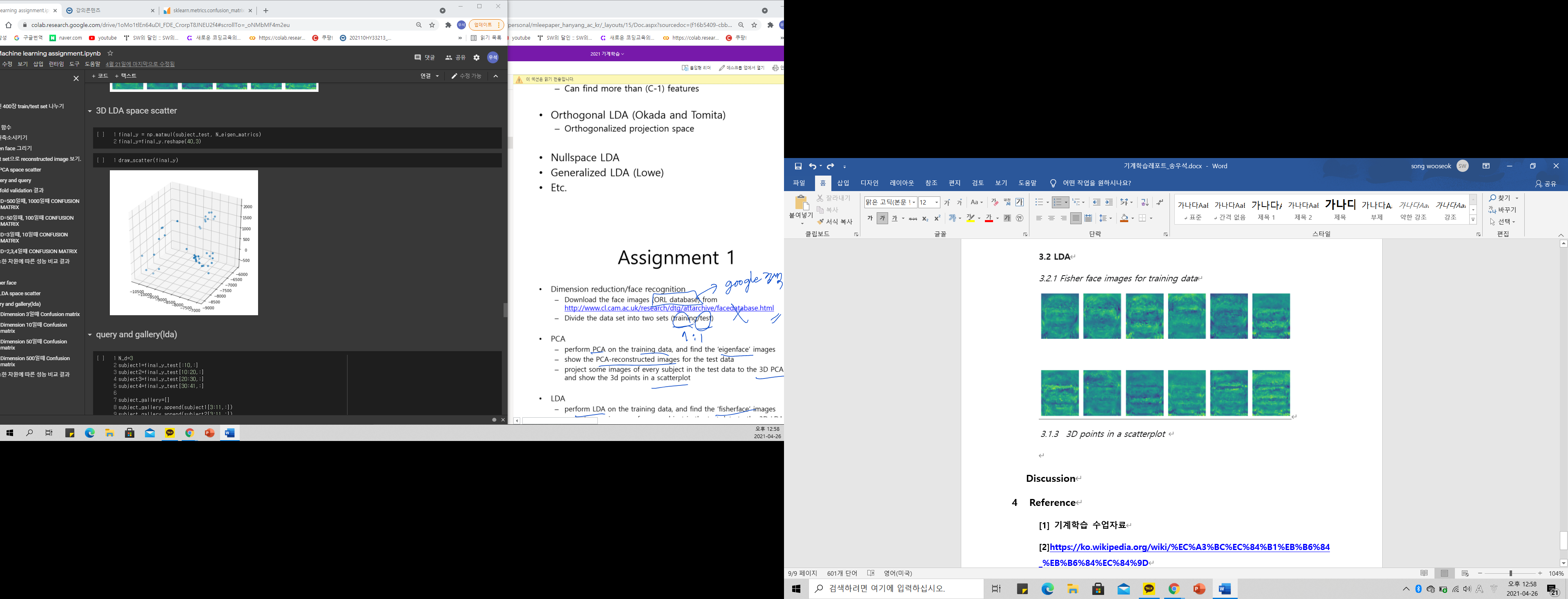
|  |  |
| --- | --- |
|  | *PCA\_accuracy* |
| *d=3* | *91.07* |
| *d=4* | *91.31* |
| *d=10* | *87.50* |
| *d=50* | *88.58* |
| *d=100* | *88.58* |
| *d=1000* | *88.58* |

**3.2 LDA**

*3.2.1 Fisher face images for training data*



* + 1. *3D points in a scatterplot*



*3.2.3 Confusion matrix for LDA (depends on dimension)*

|  |  |
| --- | --- |
|  | *LDA\_accuracy* |
| *d=3* | *73.57* |
| *d=10* | *86.07* |
| *d=50* | *88.70* |
| *d=100* | *88.70* |

1. **Discussion**

Curse of dimensionality를 해결하기 위해 Feature extraction 방법 2가지를 실험해보았다. 마지막에 performance를 측정할 때 10-fold cross validation 사용했는데, 최종 정확도는 10번의 정확도의 평균을 내서 구했다.

사진 한 장의 픽셀은 112\*92 로서 10304 dimension이었다. 10304 dimension을 d차원으로 축소시킨다고 했을 때, Pca를 이용한 차원 축소방법은 d가 커질수 query 이미지가 자신의 class에 맞는 gallery 이미지를 neighbor로 잘 골라 낼 것이라고 예상했다. 즉 실험의 정확도가 점점 올라갈 것이라고 예상했다. 하지만 PCA 방법은 분류보다는 신호성분을 인식하는 방법이므로 차원과 정확도가 상관이 없는 것처럼 보였다.

Lda는 분류를 위한 차원축소 방법으로 알고 있었기에 query/gallery 실험을 했을 때 정확도가 더 좋게 나올 것이라고 예상했다. LDA을 이용한 차원축소 실험의 정확도는 d가 커질수록 올라갔고, d=50일 때 이상에서는 88.70으로 머물로 있는 것을 확인했다. PCA보다는 정확도가 0.12 정도 차이났고 예상보다 큰 차이는 없었지만 정확도가 향상된 것을 확인할 수 있었다.

파이썬을 처음 다뤄보기 때문에 과제를 하면서 어려움이 있었지만 과제 수행중 코딩 실력이 많이 는 것 같고, 까다로웠던 부분은 한 scatter matrics에 대해서Eigen vector를 구하는데에 시간이 다소 소요된다는 점이었다. 다른 여러가지 차원 축소 실험 후 성능을 비교해보면 앞으로 연구하는데 도움이 될 것 같다. 실험에 대한 코드는 본 레포트와 별개로 따로 제출하였으며, “PCA”, “LDA1”,”LDA2” 총 세 개의 파이썬 파일로 제출하였다.

1. **Reference**

[1] 기계학습 수업자료

[2]<https://ko.wikipedia.org/wiki/%EC%A3%BC%EC%84%B1%EB%B6%84_%EB%B6%84%EC%84%9D>

[3] https://ratsgo.github.io/machine%20learning/2017/03/21/LDA/