**#2**

**Data Visualization**

**1. 문제정의**

**주제 : 와인 품질(Quality) 분류 예측**

와인의 품질(Quality)를 분류할 수 있는 기준은 다양하게 존재한다. 가령, 도수와 밀도, 산도 뿐만 아니라 휘발성산, 시트르산, 이산화황 등을 고려한다. 또한, white 혹은 red 와인 등 와인의 타입 또한 고려하여 품질을 정한다. 즉, 다시 말해 와인의 품질은 주관적인 요소와 객관적인 요소의 조합으로 정해진다. 와인의 품질은 개인의 입맛과 취향에 따라 다를 수 있지만, 전문가들은 일반적으로 품질을 평가하는 일련의 기준과 특징을 고려한다. 이를테면, 와인의 아로마, 맛, 몸체, 균형, 알코올 함량, 산도, 탄닌질 등이 해당된다.

와인의 품질을 평가하는 것은 예측이 어려운 작업이다. 와인은 수많은 요인에 의해 영향을 받으며, 작물, 토양, 기후, 포도 품종, 양조 기술, 보관 조건 등 여러 가지 요소가 와인의 특성에 영향을 미치기 때문이다. 이러한 요소들은 매번 와인이 생산될 때마다 다르게 작용할 수 있으므로 완벽하게 예측하기는 어렵다.

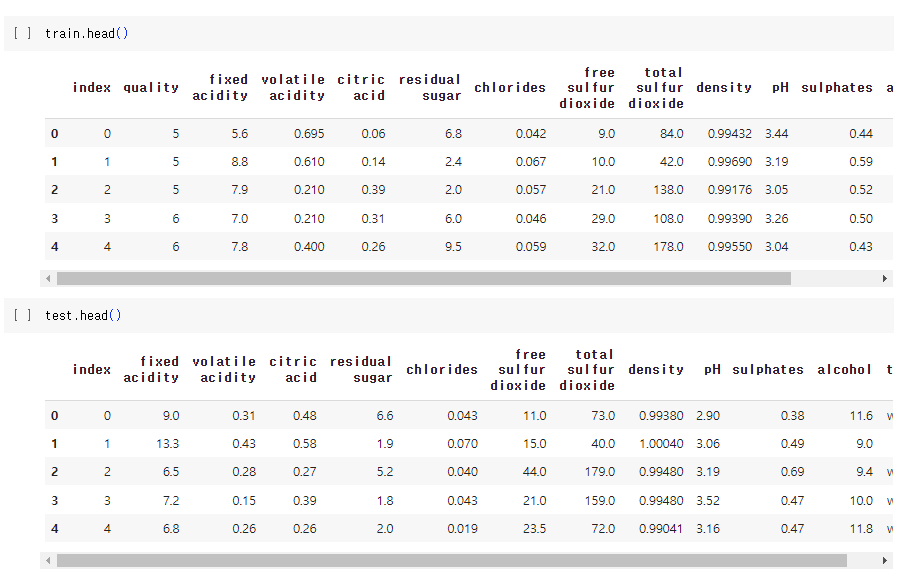
하지만 전문가들은 자신들의 경험과 지식을 바탕으로 와인의 품질을 예측하고 평가하는데 도움을 줄 수 있다. 와인 평가사, 소믈리에, 와인 크리틱 등은 와인을 평가하고 평가 지표를 사용하여 와인의 품질을 측정한다. 또한, 와인 경시 대회와 평가회에서 전문 심사원들이 와인을 평가하고 순위를 매기는 경우도 있다.

하지만, 이러한 인간의 주관적인 견해가 들어간 것이 아닌, 와인의 객관적인 요소를 가지고 예측 후, 품질을 정확하게 정할 순 없을까 생각하게 되었다. 설령, 객관적인 요소가 모두의 만족을 가져다 줄 순 없더라도, 이전의 데이터로 예측된 결과는 그 품질을 보증할 수 있는 객관적인 지표가 될 수 있을 것이다. 따라서 와인의 품질(Quality)을 조금 더 객관적인 지표로 정하고자 과거의 데이터를 가지고 이를 예측해 보기로 하였다.

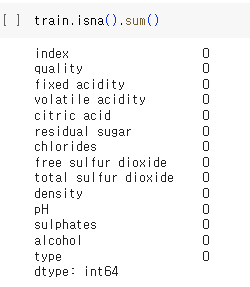
**2. 데이터 탐색 및 전처리**

데이터는 Dacon에서 구하였다. [ <https://dacon.io/competitions/open/235610/overview/description> ]

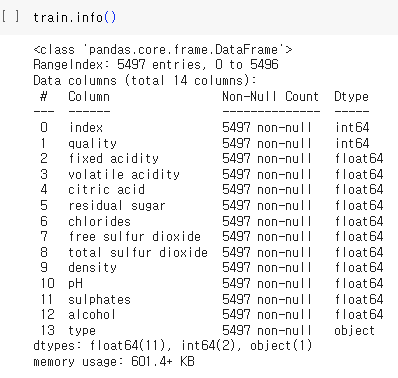
사용 데이터는 ⓐ wine\_train.csv , ⓑ wine\_test.csv 이다.



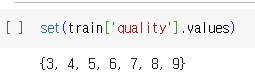
test 데이터 내에 존재하는 wine들의 quality를 예측하는 것이므로, wine\_train.csv에는 quality변수가 있지만, wine\_test에는 quality 변수가 없음을 확인하였다.



결측치가 존재하면 모델을 학습하는데 걸림돌이 될 수 있으므로, NA값을 확인한다. 결측치가 없음을 확인하였다.

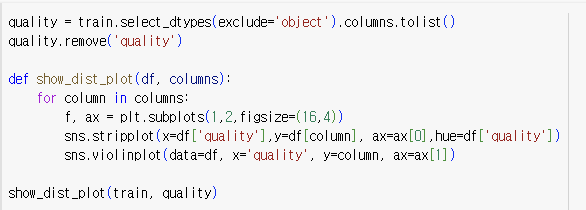


train데이터의 정보를 확인하였다. Type 변수가 object인 것을 제외하면 모든 변수는 수치형 (int, float) 데이터이다. 하지만 대부분의 모델에서는 수치형 데이터만을 입력으로 받을 수 있다. 따라서 type의 값을 수치형으로 바꾸어야 함을 인지하였다.



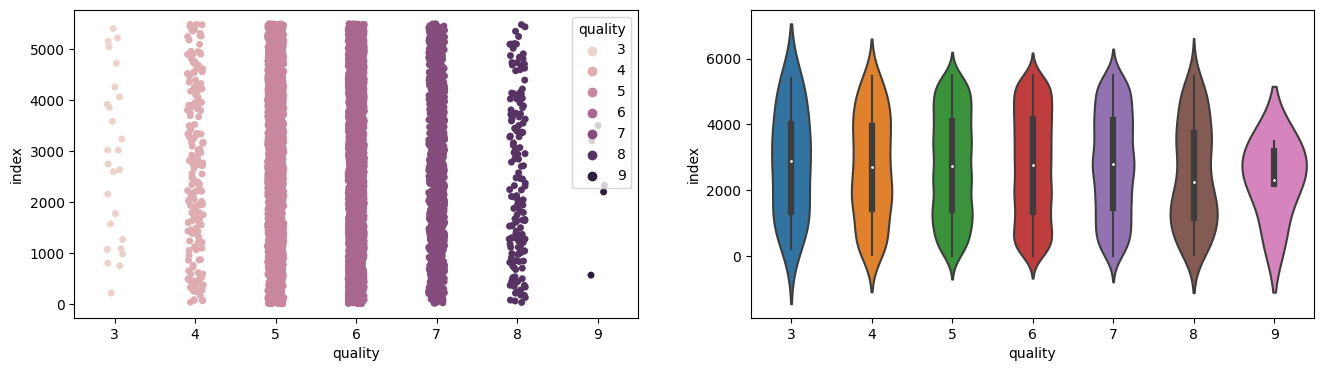
집중적으로 예측하고자 하는 것 (target변수)은 quality이다. 따라서 wine\_train의 quality에는 어떠한 값들이 들어 있는지 확인하였다. 그 결과 3점부터 9점까지 품질 점수가 측정되었다는 것을 알게 되었다.

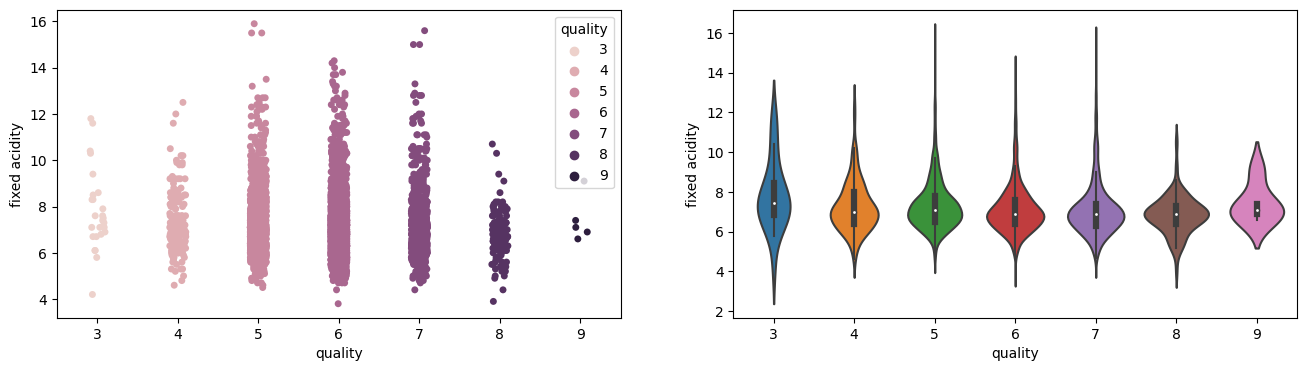
이후, 각 변수들과 quality와의 관계를 파악해야 하므로

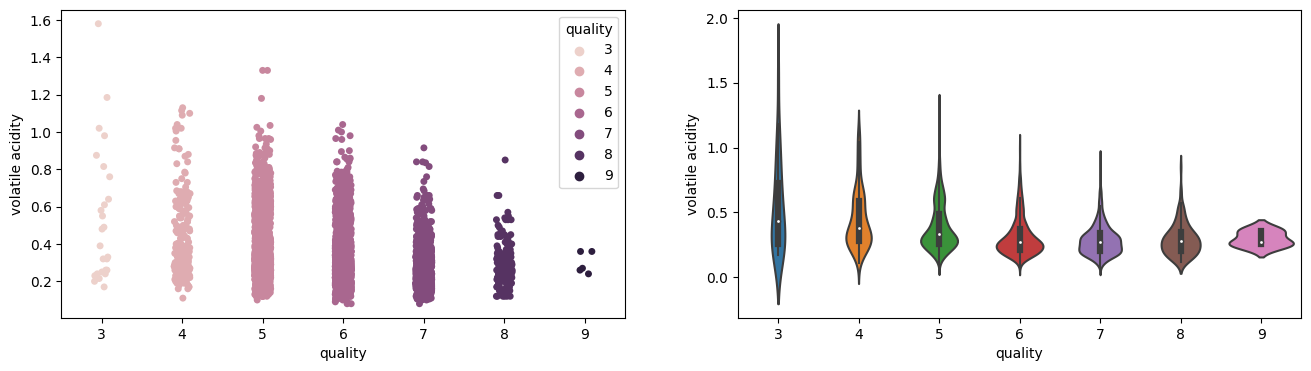


위의 코드를 통해, 각 변수들과 quality 변수와의 관계를 시각적으로 파악하고자 하였다.

위의 코드를 실행하고,

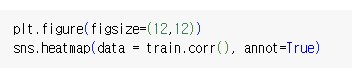




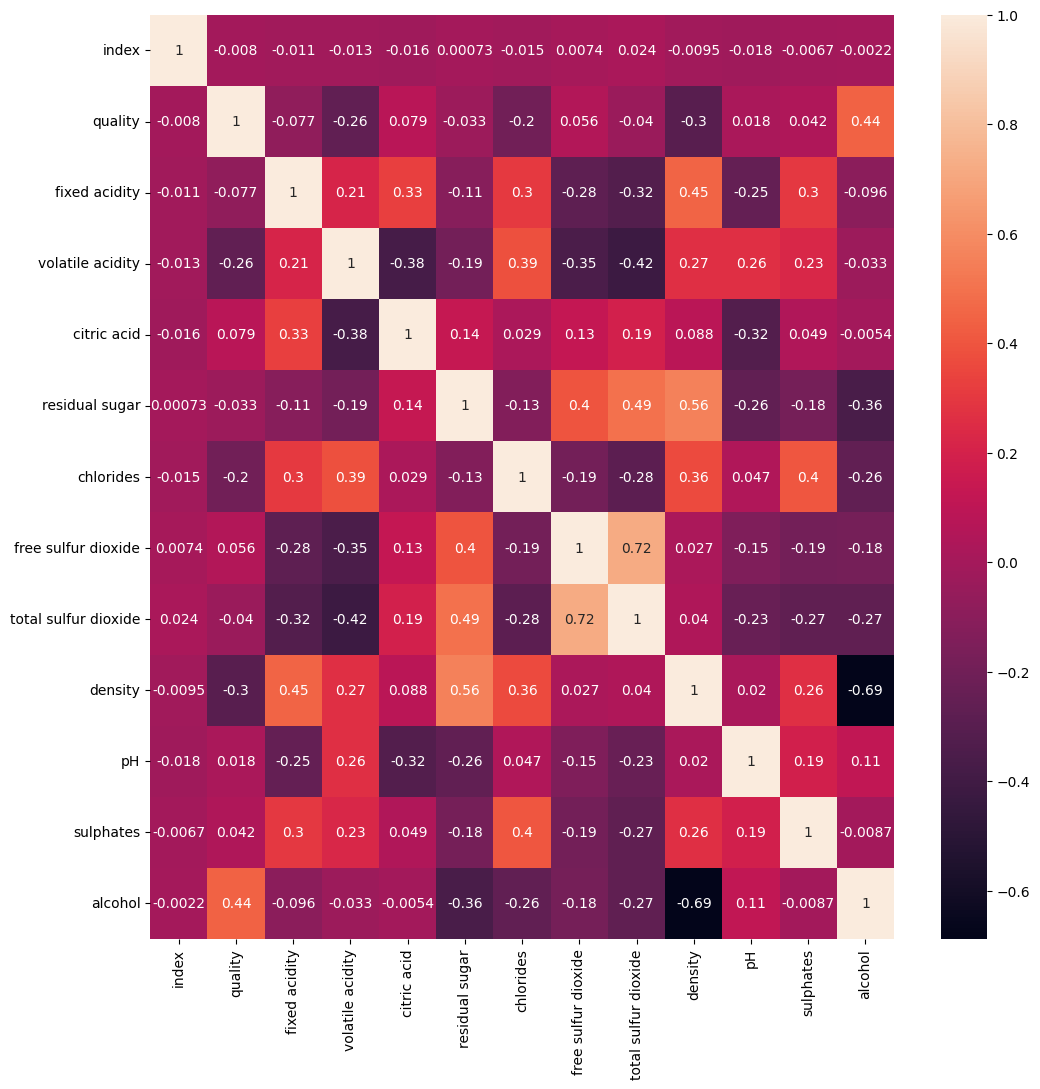


등등.. 의 관계를 알게 되었다.

하지만, 이렇게 하니 각 변수들의 범위(scale)이 달라서 파악하기 어려웠다. 따라서 각 변수별로 quality의 분포가 어떻게 되는지 시각화하여 확인하기 위해 heatmap으로 상관관계를 파악하였다.



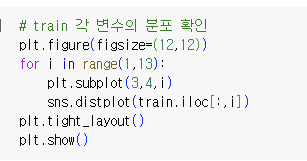
그 결과,



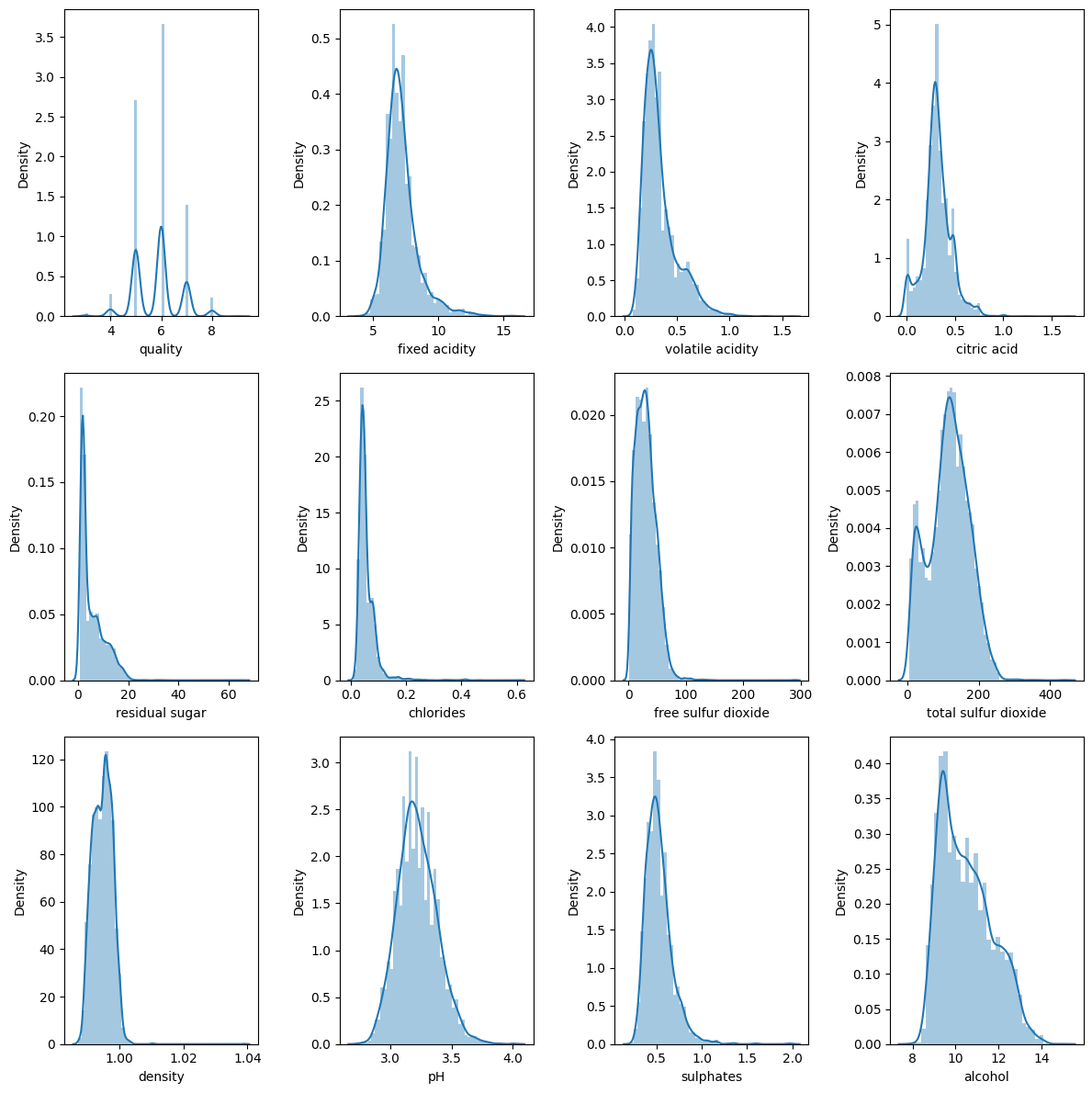
위와 같은 결과를 알 수 있었다. 즉, train data이 변수간 상관관계를 확인 할 수 있었다.

Quality 변수(target 변수)와 다른 변수 간의 상관관계가 존재하는 변수가 거의 없음을 확인 할 수 있었다. 따라서 선형회귀는 적합하지 않은 방법임을 깨달았다.

위의 heatmap 외에도



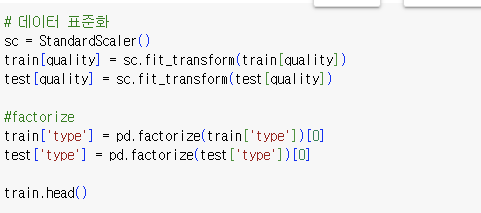
위의 코드를 사용하여 변수 간의 관계를 파악하였다.



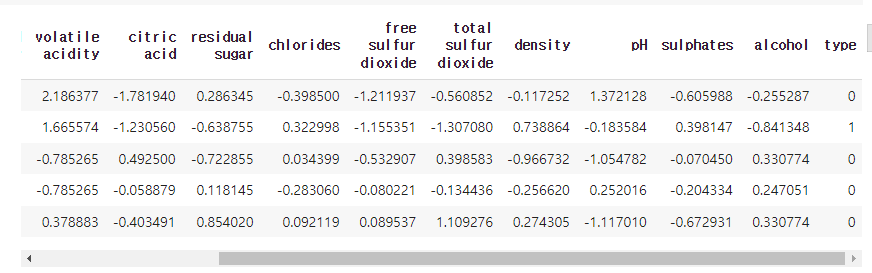
즉 데이터를 탐색하며,

ⓐ Type의 object형을 수치형으로 바꿔야하며, ⓑ 변수들의 값의 범위(scale)을 바꿔야 함을 깨달았다.

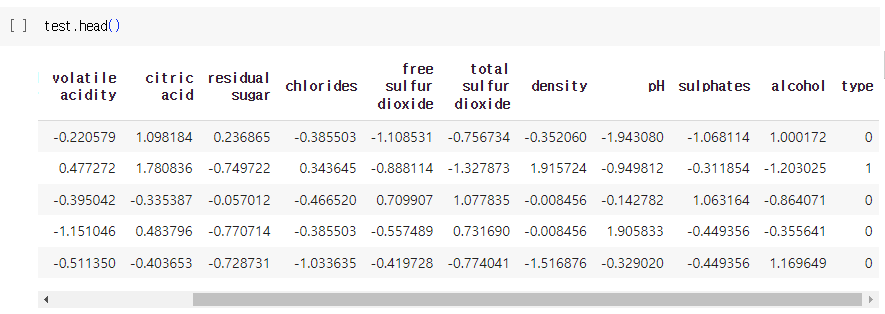
따라서, ⓐ Type을 0과 1로 변환하고, ⓑstandScaler로 표준화 작업을 진행하는 것을 결정하였다.



위의 코드 결과,



위와 같이 각 변수의 값을 표준화 하여 맞추었고, type 값을 white : 0, red : 1로 mapping하여 변환하였다.



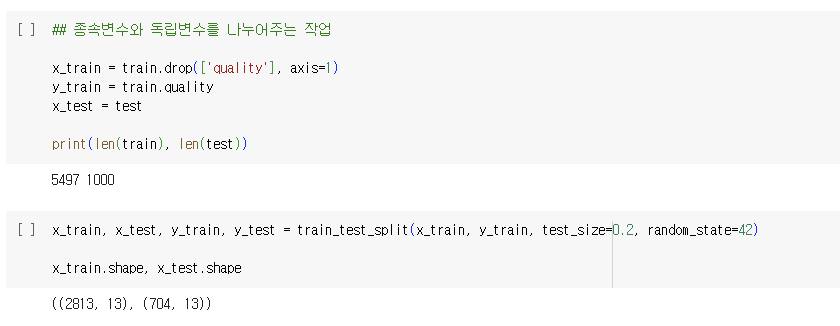
예측해야 하는 것은 test data이기 때문에 wine\_test도 알맞게 변환하였다.

**3. 예측 분석 (회귀 예측 / 분류예측 모두 가능)**

① RandomForest ② SVM ③ 로지스틱 회귀 모델 ④ Decision Tree ⑤ Knn 모델

총 5가지를 사용하여 각각을 예측하였다.

본격적인 예측을 진행하기 전에

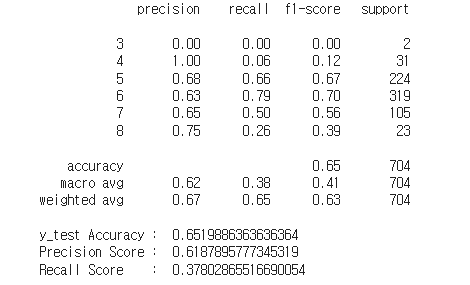


x\_train, x\_test, y\_train, y\_test을 나누는 작업을 진행하였고,

test\_size는 0.2로 즉, 20%는 test, random\_state는 42로 잡고 예측을 진행하였다.

① RandomForest

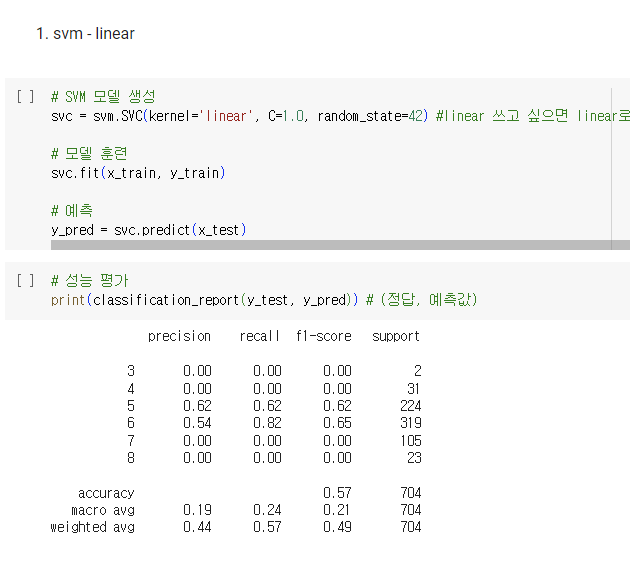




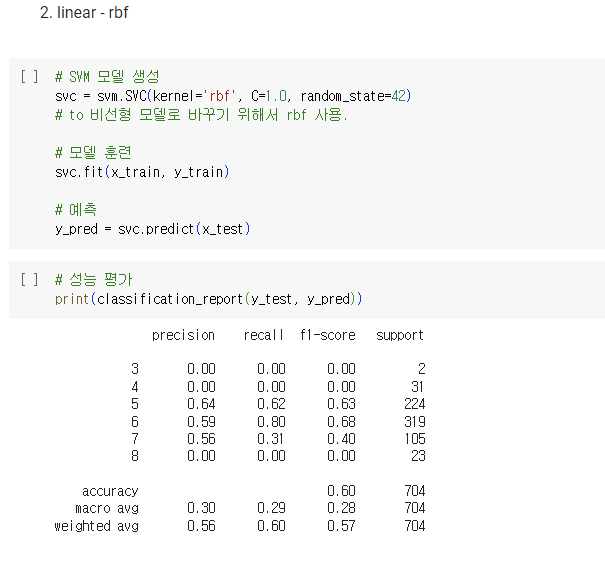
RandomForest를 사용한 결과 y\_test의 accuracy가 65%, precision이 61%, Recall이 37%의 결과가 유도되었다.

② SVM

Svm은 linear과 rbf두가지를 각각 사용하였다.

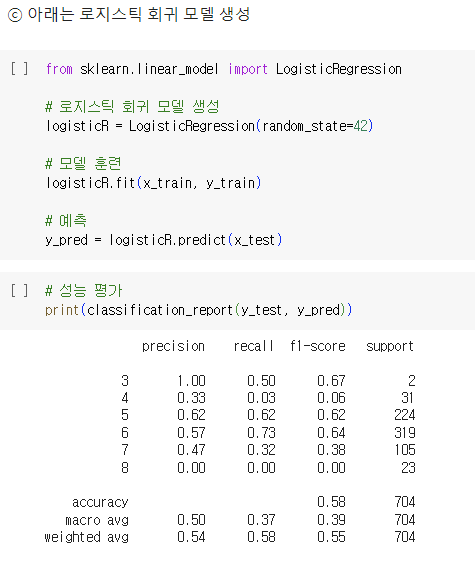


SVM의 linear의 결과 Accuracy는 0.57의 결과가 유도되었다.



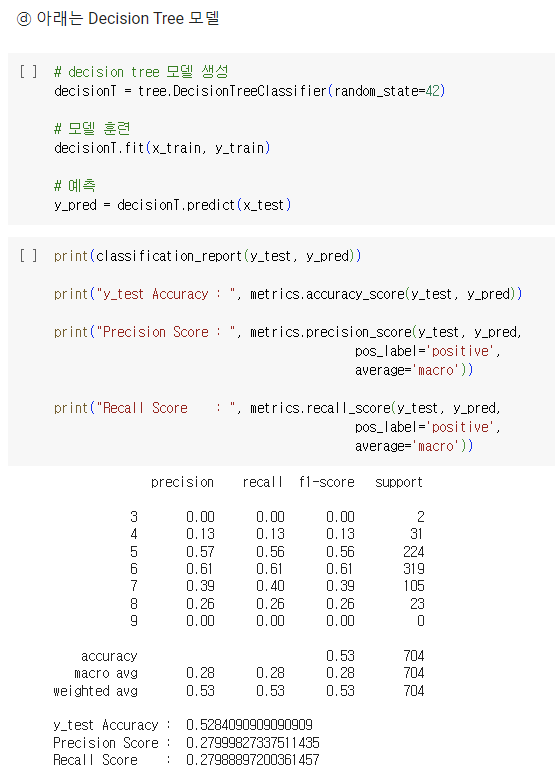
비선형 모델인 rbf로 변경하자, 정확성이 60%으로 높아졌다.

③ 로지스틱 회귀 모델



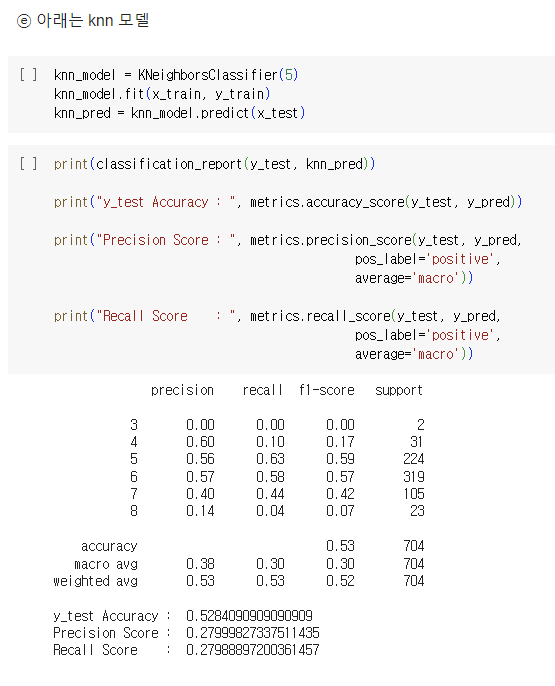
로지스틱 회귀 결과 정확성이 0.58의 결과가 유도되었다.

④ Decision Tree



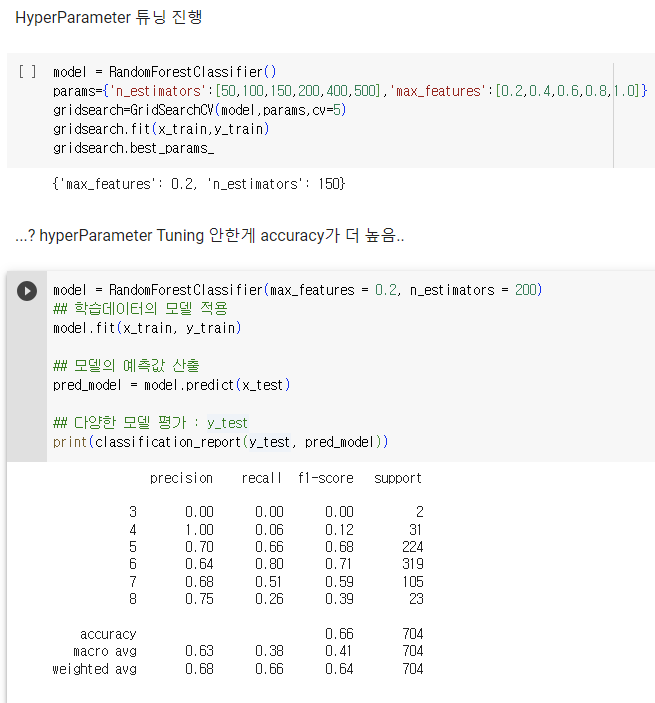
Decision Tree의 결과 Accuracy는 53%, Precision은 27%, Recall은 27%의 결과가 유도되었다.

⑤ Knn 모델



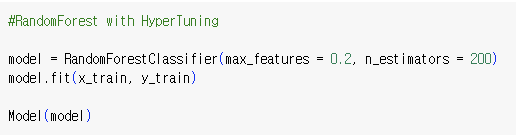
Knn의 모델을 사용한 결과 Accuracy는 52%, precision은 27%, Recall은 27%의 결과가 유도되었다.

따라서 ⓐ~ⓔ 까지 비교한 결과, RandomForest를 사용한 model 예측 성능이 가장 높은 것을 알았다. 따라서 이의 성능을 좀 더 높이고자 HyperParameter Tuning을 진행하고자 하였다.

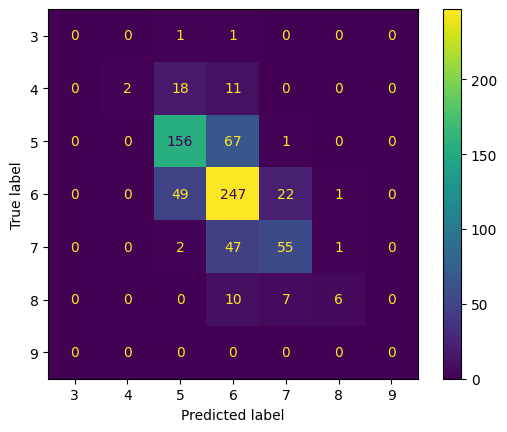


따라서 기존의 RandomForest의 n\_estimators = 500을 max\_features = 0.2, n\_estimators = 200으로 변경하였고, 정확도가 0.01 향상한 것을 알 수 있었다.

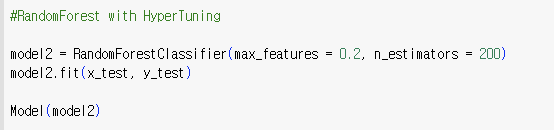
결과적으로, 최종 결과는 HyperParameter Tuning을 진행한 RandomForest 모델을 사용한 것이 가장 최적의 모델임을 알았고, 예측 결과는 Accuracy 0.66, Precision은 0.61, Recall은 0.37로 마감하였다.

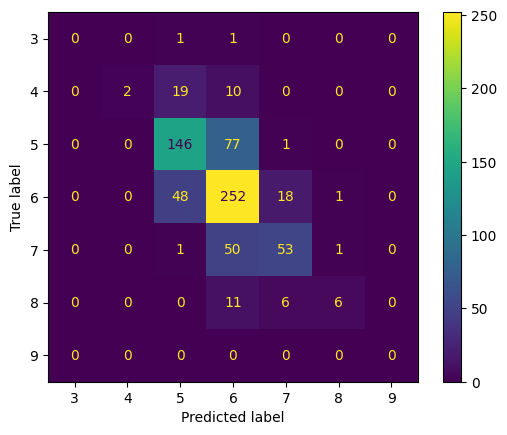


Train 데이터를 가지고, 예측한 것이 얼마나 맞았는지 labeling된 것을 시각화하여 표현하였다



또한, test 데이터 또한 같은 시각화를 진행하였고,





위와 같은 결과를 확인할 수 있었다.

**4. 테스트 데이터로 예측한 결과 군집화 => K-means 알고리즘 적용 : 실루엣 계수를 이용한 적절한 K 선정 및 PCA로 군집 결과 시각화**

와인 품질을 예측하고자 하는 test Data의 실루엣 분석 점수를 추정하였다.

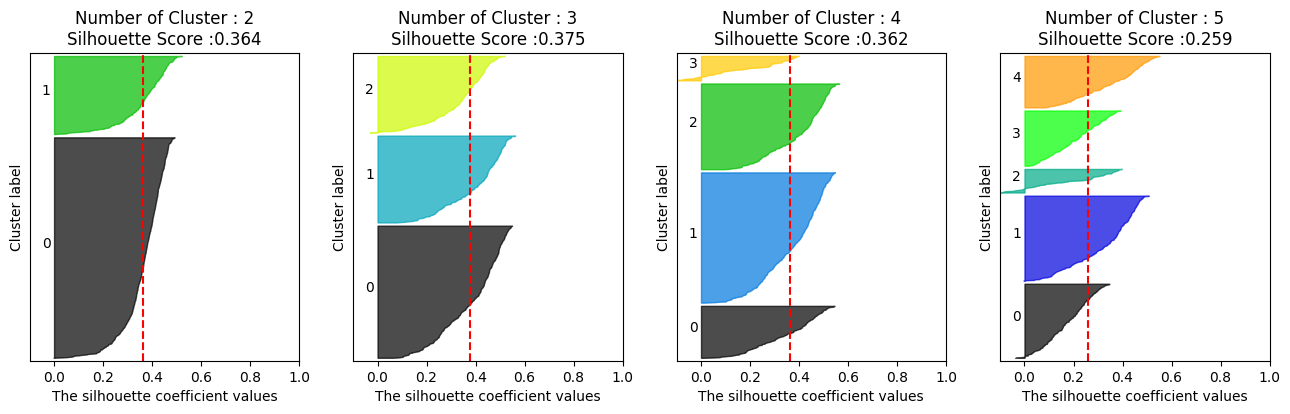


그 결과 0.375점을 얻을 수 있었다.



실루엣 계수를 면적으로 시각화 하기 위해 위와 같은 코드를 작성하였고,





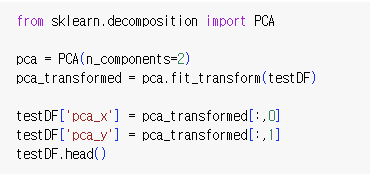
그 결과 위와 같은 시각화 된 실루엣 점수를 확인할 수 있었다.

이 함수 결과 clustering을 3으로 두는 것이 가장 좋다는 것을 알게 되었다. 각각의 그림의 면적이 가장 비슷하며, 다른 함수들에 비해 silhouette coefficient values의 값도 비슷하거나, 높기 때문이다.

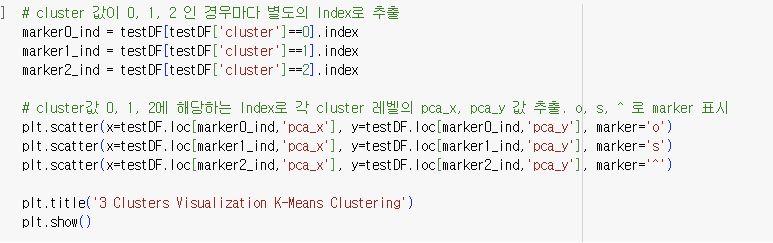
위의 그래프를 해석한 결과, 앞에서 작성한 코드 결과 중 <<#모든 데이터의 평균 실루엣 계수값을 구함. : 결과적으로 0.375점을 얻음>> 이 잘 진행되고 있다는 것을 다시 한번 확인 할 수 있었다.

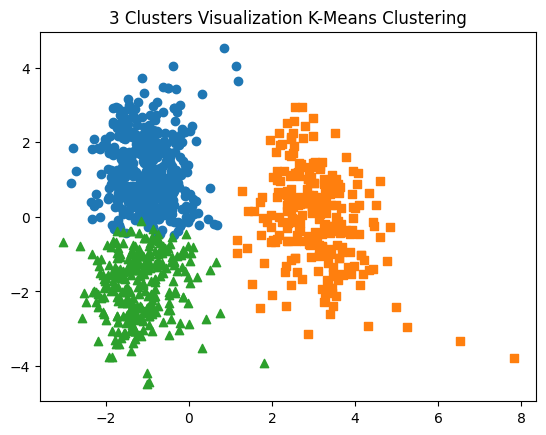
마지막으로, K-means clustering을 진행하기 위해





다음과 같은 pca\_transform을 진행하고,

이를 시각화 하기 위해, 위와 같은 코드를 작성하였다. 그 결과,



위와 같은 3clusters visualization K-means clustering 결과가 나오게 되었다.

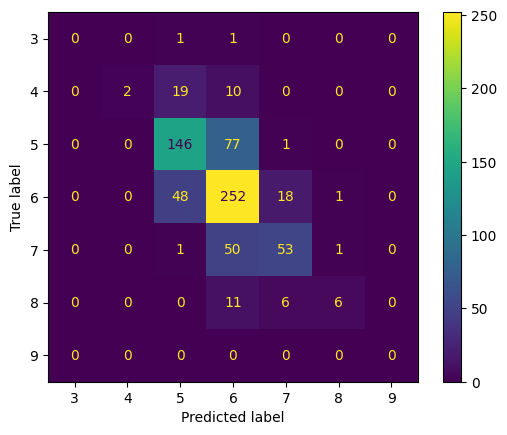
파란색과 초록색의 경우 겹치는 영역도 약간은 존재하긴 하지만, 전체적으로 각각의 군집은 군집끼리 뭉치고, 군집 외의 거리도 어느정도 확보한 결과를 볼 수 있었다.

**5. 결과 해석**

결과적으로, 와인의 품질을 예측하기 위해서는 RandomForest가 최선의 modeling 방법인 것을, 알게 되었다.

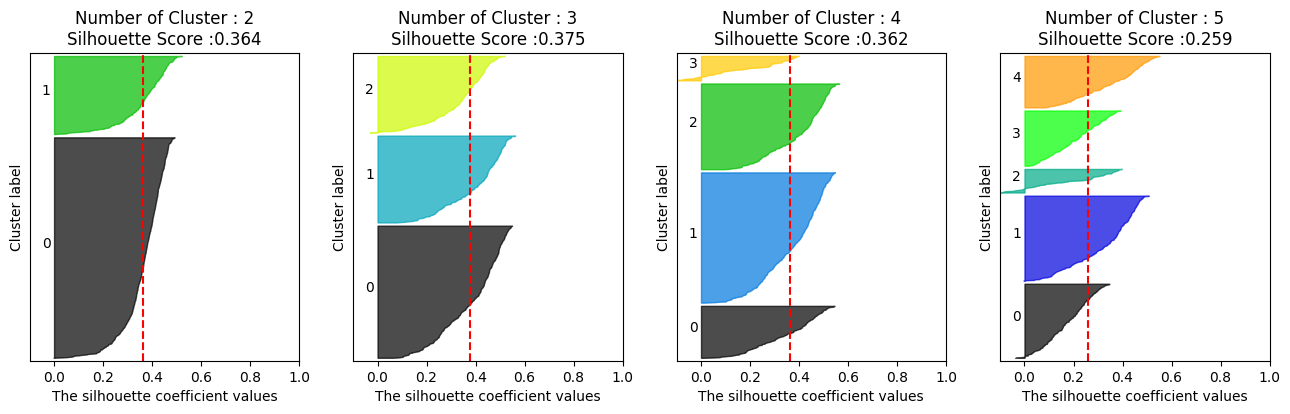
HyperTuning Parameter을 진행한 RandomForest 모델의 예측 결과, Accuracy 0.66, Precision : 0.61, Recall : 0.37이다.

위의 model을 적용 후, test에 적용해 보니,

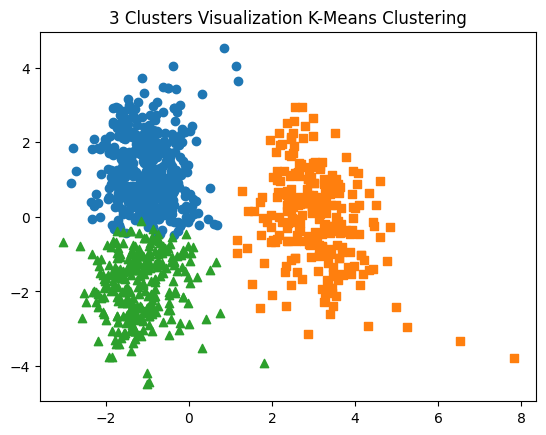


다음과 같은 True Label 대비 Predicted Label의 결과를 알 수 있었고, 비교적 높게 결과를 예측 하였음을 알 수 있었다.

이러한 테스트 데이터로 예측한 결과를 군집화 하기 위해 K-means 알고리즘을 적용해보니,



실루엣 계수가 3일 때, 가장 좋은 결과를 나타냄을 알게 되었고



3 Clusters Visualization K-Means Clustering을 한 결과 다음과 같다.

위의 예측을 한 결과, 예측의 Accuracy가 낮아서 아쉬운 결과를 보였다. Grid search과정에서 n-estimators의 변수의 크기가 작아도 좋은 결과를 내는 것을 확인하였다. 아마 모델이 train 데이터에 과적합 되었을 것이라 예측된다. 해당 부분을 보완하기 위해, 튜닝에 참여 하지 않은 Hypter Parameter들을 Grid Search에 참여하게 해서 최적의 값을 찾아 보고자 한다.