Homework 6 report

이름: 유성민

학과: 소프트웨어

학년: 3

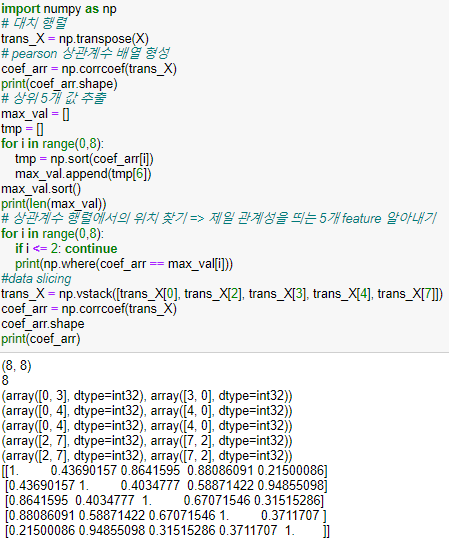
학번: 201520908

1. **Coefficient interpretation (logistic regression)**

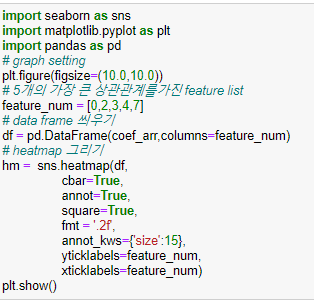
주어진 데이터의 모든 feature에 대해서 correlation coefficient array를 구하고 이 중 correlation coefficient 가 높은 5개의 feature를 이용해 feature간의 correlation을 **heatmap으로 그린다.** [5]

1-1) 개요

선형관계를 잘 형성해주기 위해서는 표준화가 필요합니다. 따라서 z-score를 이용하여 데이터표준화를 진행하고, numpy package의 corrcoef 함수를 이용하여 선형 관계 지수를 나타내는 Pearson correlation coefficient를 보고 상위 5개의 값을 추출합니다.

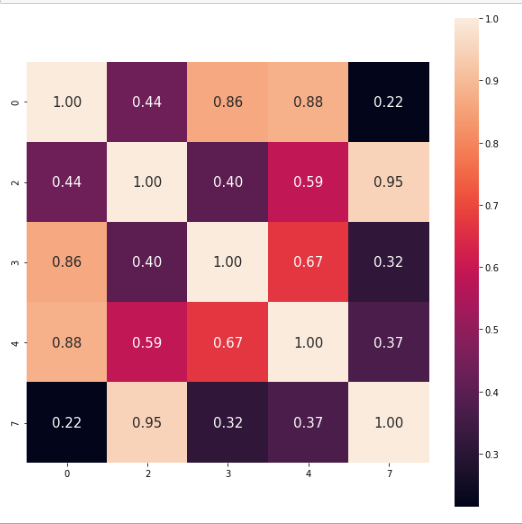
2-2) 코드

보다 편한 계산을 위해 전치행렬로 바꾸고, corrcoef함수를 사용하여 상관계수를 도출, for문에서 각각의 feature에 대해 정렬을 하고 리스트의 제일 뒤에 최대값이 위치하므로 따로 max\_val리스트에 담고 해당 상관계수의 위치를 확인하여 상위 5개의 상관계수를 가졌던 feature의 pair를 확인합니다. 확인 후 해당 데이터만 따로 뽑아서 데이터를 생성합니다. 그리고는 heatmap을 그리기 위해 상관계수를 담은 리스트를 갱신합니다.

결과를 보면 0,3 / 0,4 / 2,7의 결과가 가장 높았고 서로 상관관계가 제일 높은 5개를 뽑으면 0, 2, 3, 4, 7이라고 할 수 있습니다.

heatmap 작성코드

1-3) 결과

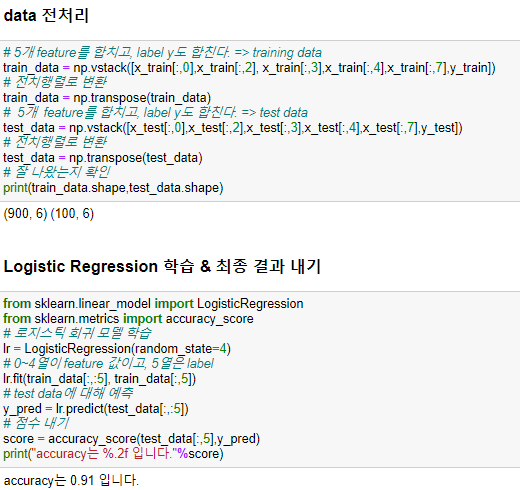
2번과 7번 feature가 가장 높은 상관관계인 0.95를 가졌고 0번과 4번, 0번과 3번이 그 뒤를 이었습니다.

위에서 구한 correlation coefficient array를 바탕으로 multiple **logistic** regression 모델에서 사용할 feature를 정하고 90%의 데이터를 학습데이터로 사용하여 regression 모델 학습 후 10% 테스트 데이터에 대한 **test acc를 출력**한다. [5]

1-1) 개요

제일 좋은 결과를 보였던 5개의 feature로 이루어진 training data 및 test data가 필요합니다 따라서 data 전처리를 필요로 하게 됩니다. 그리고 Logistic Regression을 이용하여 학습시키고 accuracy를 출력합니다

1-2) 코드

데이터를 vstack을 이용하여 수직으로 합치고 전치행렬을 사용하였습니다. 모양을 보니 잘 나온 것을 확인할 수 있었습니다. 총 1000개의 sample이 9:1로 분할 되었고, 5개의 feature + y 값 총 6개입니다. 로지스틱 회귀를 트레이닝 데이터로 학습시키고 테스트 데이터로 테스트한 결과 정확도가 0.91로 상당히 높은 것을 볼 수 있었습니다. 따로 hyper parameter를 조절하지 않았음에도 서로 상관계수가 큰 데이터끼리 학습이 되니 이러한 높은 결과가 나왔다고 생각합니다.

1. **Variable selection (linear regression)**

Linear regression 모델의 variable 선택을 통한 모델을 비교하기 위해서 수업시간에 배운 Adjusted 을 기반으로 forward selection을 수행한 후 **test 를 출력하고** x축은 선택한 variable의 개수, y축은 테스트 데이터의 Adjusted 과 학습데이터의 Adjusted 를 각각 다른 색으로 **그래프를 이용해 출력한다**.[10]

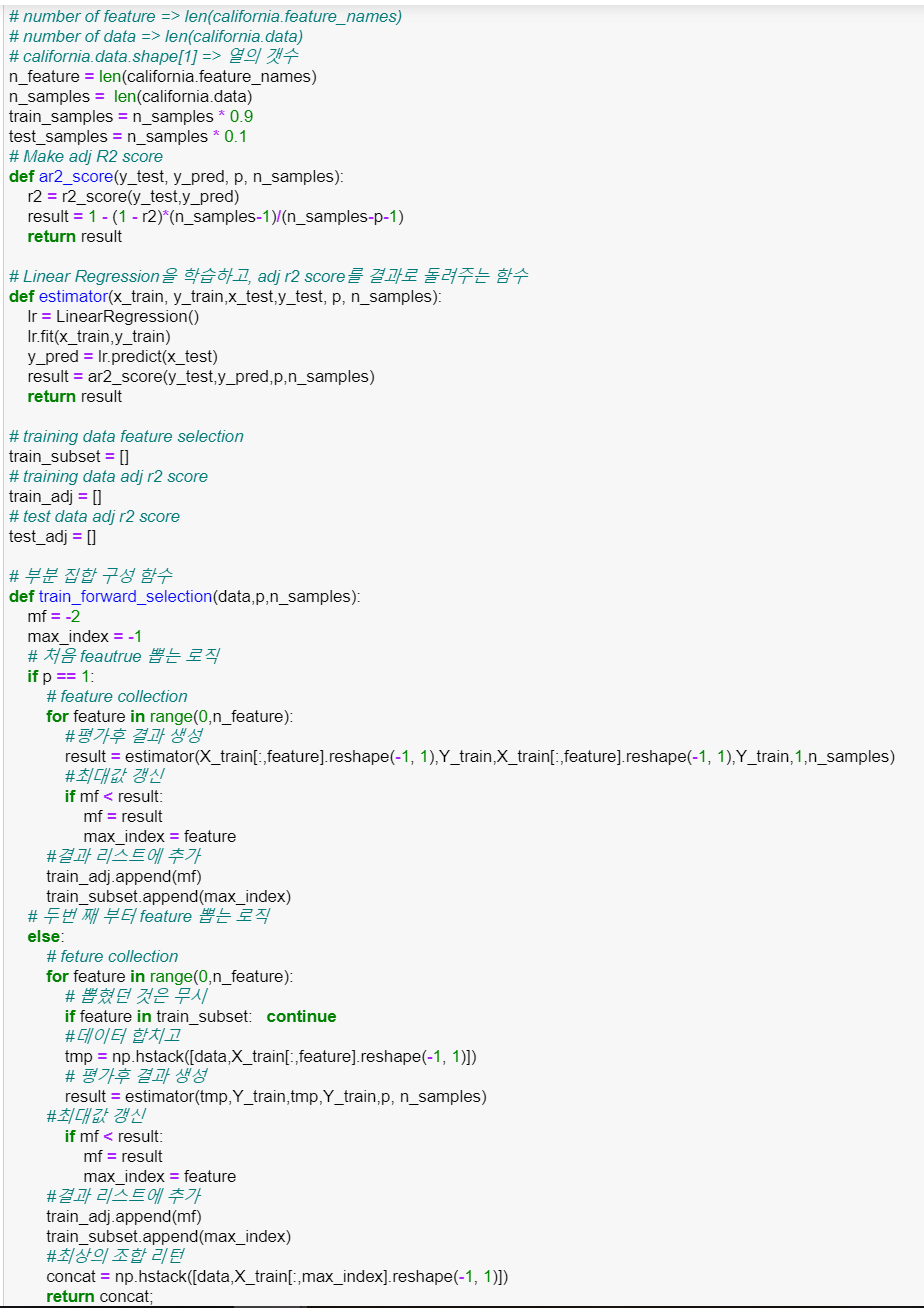
* + - forward selection 을 이용하여 variable을 4개 까지 선택한다. (for 문 이용)
    - sklearn의 r2\_score 함수에 기반하여 Adjusted 를 만든다.

2-1) 개요

adj r2 score = 1 - (1 - r2\_score)\*(n\_-1)/(n-p-1) 공식을 사용하여 함수를 작성하고, 각 feature를 1개 고르고 8개중 가장 좋은 것 선택, 첫 번째 골라진 것 제외 7개 중 두 번째 feature 선택, 첫 번째 고른 것과 두 번째 고른 것 합치고 2개를 제외한 6개 중 세 번째 것을 뽑고, 이전 단계에서 합쳐 놓은 데이터와 세번째 것을 합칩니다. 마지막으로 5개중 하나를 뽑고 이전 단계에서 합쳐 놓은 데이터와 네번째 것을 합치고, 뽑는 기준은 adj r2 score가 높은 것입니다. 그리고 각각의 단계에서 train data로 학습과 평가를 진행, 하고 feature가 각 단계에서 선택 될 때마다 test data를 이용하여 adj r2 score를 구하고 모든 단계가 종료되면, 그래프로 나타냅니다.

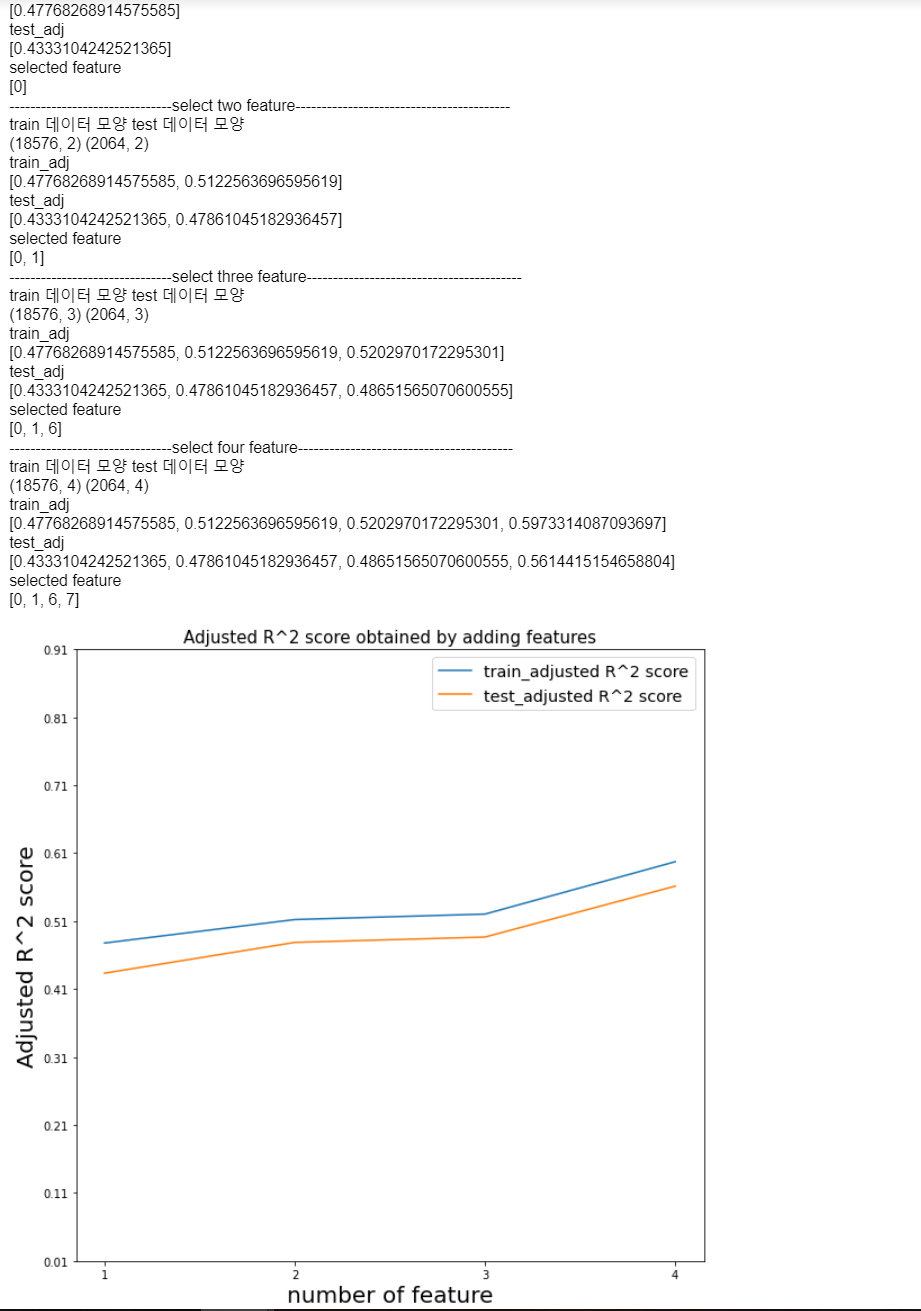
2-2) 코드

ar2\_score함수는 r2\_score를 기반으로 adj r2를 구하고 리턴, estimator함수는 각각 인자에서 데이터를 받아 linear regression을 진행하고 adj r2로 평가후 그 결과를 리턴하는 함수, train\_forward\_selection함수는 train data를 기반으로 feature를 1개씩 추가해가며, estimator함수에 집어넣고 서로 다른 부분집합을 평가 후, 가장 큰 결과를 남기는 함수입니다. 로직을 보면 아무 것도 존재하지 않고, 첫 번째 feature를 선택하는 로직과 두번째 선택부터 네 번째 선택까지 하는 로직으로 나뉘었는데 이유는 두 번째 선택부터는 데이터가 합쳐 져야하는 부분이 생기기 때문입니다. (1번째 선택, 2번째 선택(7가지)) 해당 pair를 생성 후 estimator에 집어넣기에 나눴습니다. 그리고 뽑힘 중복 방지를 위해서 train\_subset이라는 리스트를 두어서 해당 리스트에 있는 경우 건너뛰도록 설계했습니다. 마지막은 최대값 갱신 코드와 뽑힌 feature의 index를 train\_subset에 저장하고, 그래프 출력을 위해 결과로 나온 train데이터의 adj r2 score를 train\_adj 리스트에 집어넣는 로직으로 마무리되고, 테스트 데이터로 평가를 진행해야 하기에 이를 위해 필요한 제일 좋았던 pair로 이루어진 train data를 리턴하고 마무리합니다.





1-3) 결과

feature가 뽑힐수록 성능이 좋아지는 것을 볼 수가 있었습니다. 이는 overfit하지 않고, 점점 underfit에서 벗어나는 모양새입니다. 확실히 adj r2 score는 feature가 증가할수록 패널티를 받으니 눈에 띄게 증가하지는 않고 서서히 증가하는 것을 알 수 있습니다.

**Regularized linear model (linear regression)**

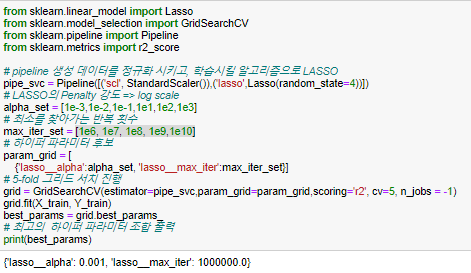
Regularization 방법 중 하나인 lasso regression을 이용하여 5-fold cross validation을 통해 최적 하이퍼파라미터 탐색 후, 최종 모델의 test 를 출력한다. [10]

* + Scikit learn의 라이브러리를 기반하여 수행한다.

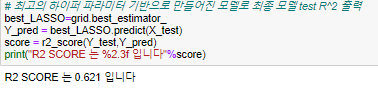
3-1) 개요

평가할 하이퍼 파라미터로는 l2 penalty의 강도인 alpha와 최저 점을 찾아가는 반복횟수 max\_iter를 선정하였고, alpha는 [1e-3,1e-2,1e-1,1e1,1e2,1e3]가 후보, max\_iter는 [1e6, 1e7, 1e8, 1e9,1e10]를 후보로 선정하였습니다. 해당 parameter 조합에 대한 성능을 보기위해 grid search를 진행하였고, 평가 기준은 r2 score입니다.

3-2) 코드

최고의 파라미터 조합으로는 alpha 0.001, max\_iter는 백만이 나왔습니다. 반복 횟수가 너무 많아도 안되고, 정규화 강도는 작은 것이 좋았습니다.

3-3) 평가 및 결과

r2 score는 0.621로 모델이 어느 정도 형성이 되었다고 말 할 수 있습니다. 1에 가까울 수록 좋지만 또 너무 1에 가까우면 overfit의 가능성이 있어 적당한 것이 좋습니다.