수강 과목 : 기계학습
담당 교수 : 양희정 교수
과제 제출자 : 이우람 (학번: 202376602)

케글 데이터 출처 :
https://www.kaggle.com/datasets/shivam2503/diamonds

[1] 다이아몬드 데이터 구조(변수 설명)

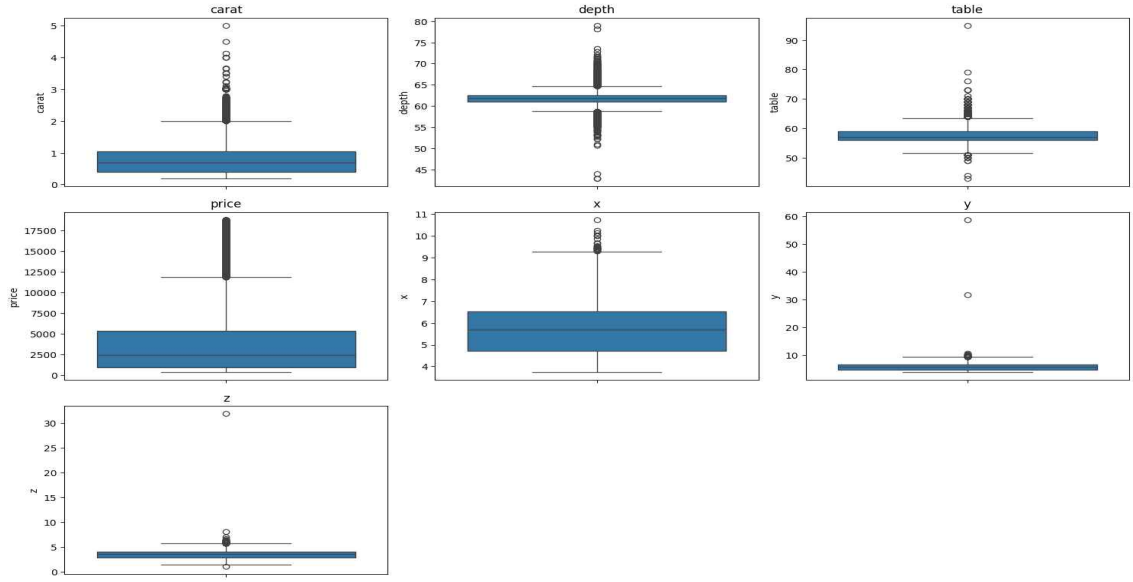
변수	의미	기타
carat	무게	ex) 3캐럿짜리 다이아몬드, 수치형 데이터
cut	세공의 질	다이아몬드의 단면을 어떻게 커팅하였는지에 대한 상태값을 분류, 계층이 있는 범주형(factor), "Fair", "Good", "Very Good", "Premium", "Ideal"로 나눔
color	색깔	총 7개의 컬러로 표기된 범주형 데이터, J (worst) to D (best)
clarity	투명도	범주형 데이터, I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best)
depth	깊이	수치형 데이터
table	넓은 폭 대비 꼭대기의 넓이	수치형 데이터
price	가격(미국달러, ($326--$18,823)	수치형 데이터
x	길이(mm)(0--10.74	수치형 데이터
y	넓이(mm)(0--58.9)	수치형 데이터
z	깊이(mm)(0--31.8)	수치형 데이터

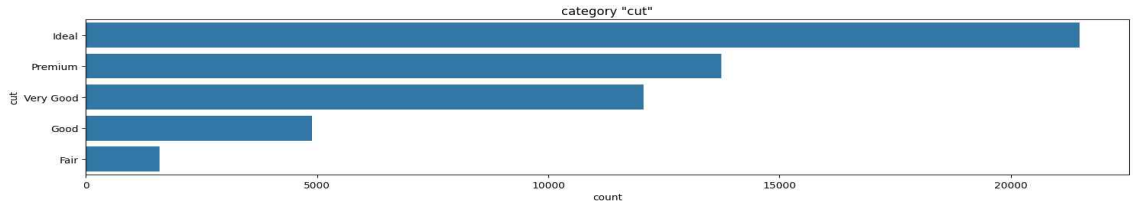
[2] 데이터 수 (53940개)

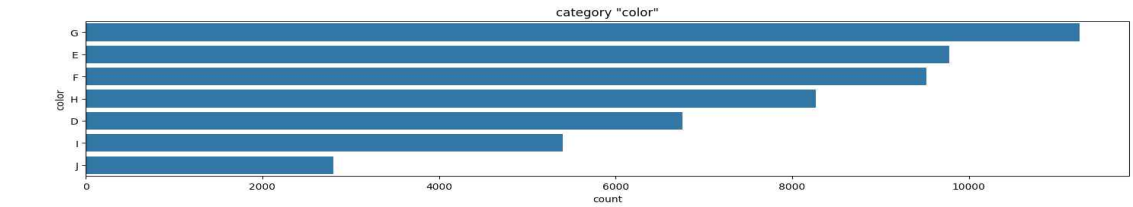
	Unnamed: 0	carat	cut	color	clarity	depth	table	price	x	y	z
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...	...	...
53935	53936	0.72	Ideal	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	53937	0.72	Good	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	53938	0.70	Very Good	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	53939	0.86	Premium	H	SI2	61.0	58.0	2757	6.15	6.12	3.74
53939	53940	0.75	Ideal	D	SI2	62.2	55.0	2757	5.83	5.87	3.64

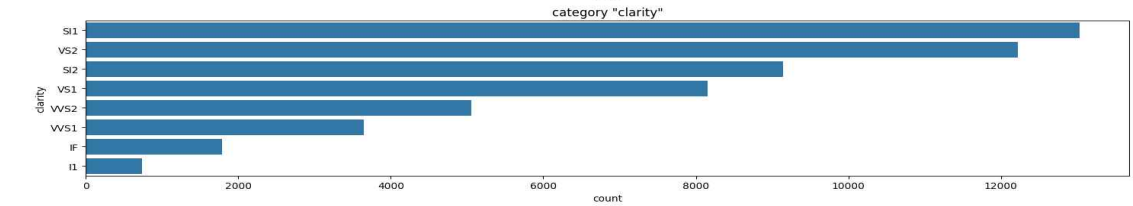
53940 rows × 11 columns

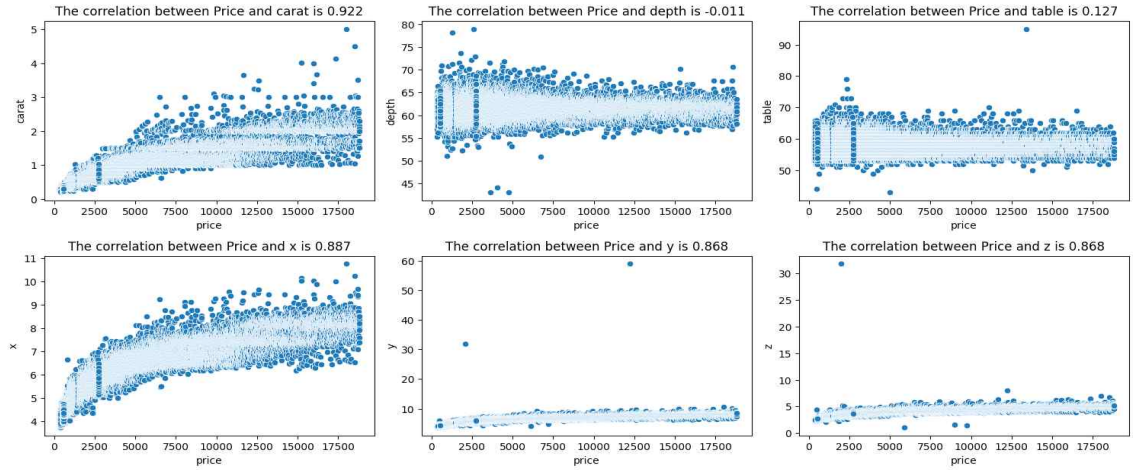
[3] 탐색적 데이터 분석(시각화)



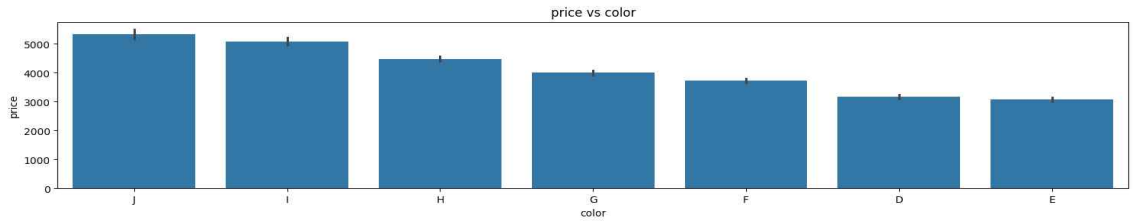


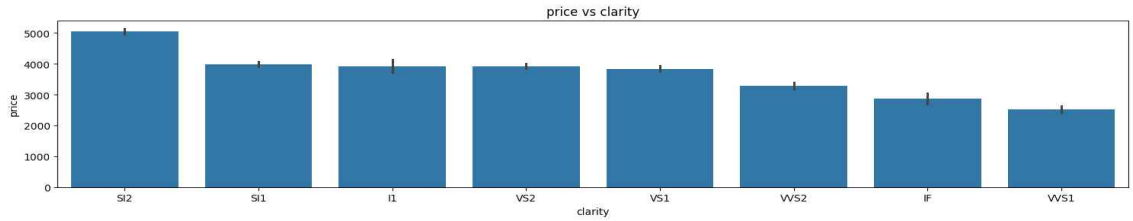


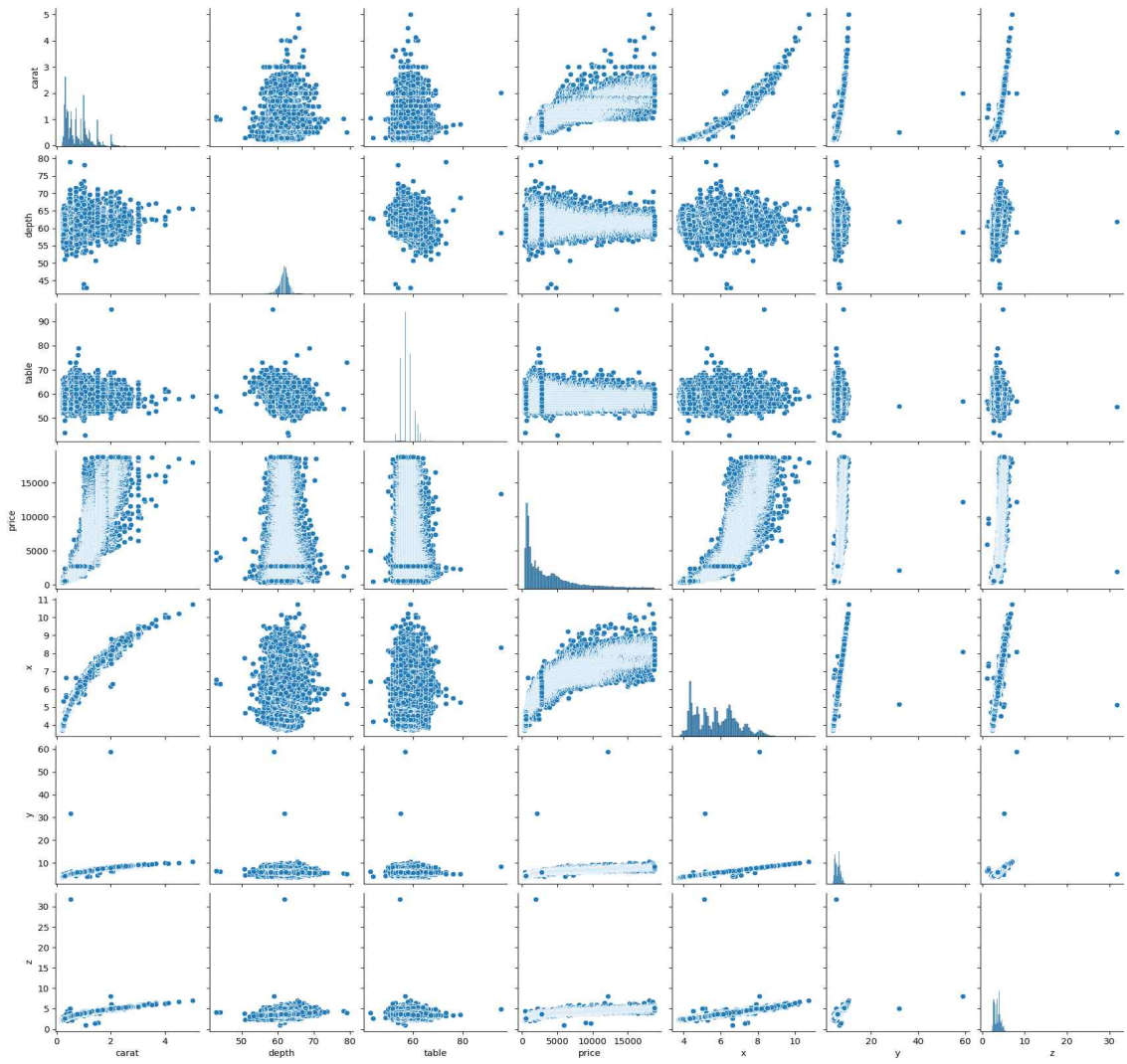





price vs cut
price
cut
Premium
Fair
Very Good
Good
Ideal


price vs color
price
color
J
I
H
G
F
D
E


price vs clarity
price
clarity
SI2
SI1
I1
VS2
VS1
VVS2
IF
VVS1


carat
depth
table
price
x
y
z

[4] 탐색적 데이터 분석(전처리)

1) 첨도와 왜도 확인

- 왜도(skew)와 첨도(kurtosis)를 통해 정규성을 검정한다.
- West 등(1995)의 연구에 따르면 정규분포 기준은 |왜도| < 2, |첨도| < 7 이면 정규분포에서 크게 벗어나지 않아 정규성을 충족한다고 볼 수 있다.
- Shapiro 검정에 비해 느슨한 기준이기 때문에 많은 논문에서 많이 인용되고 있다.

- 전처리 전 수치형 데이터별 첨도와 왜도 결과값

	첨도	왜도
carat	1.2457811079895178	1.1132176108381227
depth	5.419006676383654	-0.11371107011617339
table	2.774996519742369	0.7920686095882784
price	2.1792544447759763	1.6182203665466373
x	-0.7043849287510566	0.39690785208661533
y	92.29538299947419	2.470200340127712
z	48.03544208520775	1.5893087544943625

- 전처리 후 수치형 데이터별 첨도와 왜도 결과값

	첨도	왜도
carat	0.7366524118507214	1.0545281842352914
depth	-0.005512015344716126	-0.24785727145633726
table	-0.2104258549073479	0.3627769301846416
price	2.170606352489278	1.6179350598193931
x	-0.7665620176817569	0.4037317323389572
y	-0.776394884191538	0.39782182975139435
z	-0.7817634324461884	0.39373817295025787

- 전처리 후 데이터 구조

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	4	5	1	61.5	55.0	326	3.95	3.98	2.43
1	0.21	3	5	2	59.8	61.0	326	3.89	3.84	2.31
3	0.29	3	1	3	62.4	58.0	334	4.20	4.23	2.63
4	0.31	1	0	1	63.3	58.0	335	4.34	4.35	2.75
5	0.24	2	0	5	62.8	57.0	336	3.94	3.96	2.48

- 스케일링과 훈련검증용 분할 후 X_train 데이터

	carat	cut	color	clarity	depth	table	x	y	z
count	3.559000e+04	3.559000e+04	3.559000e+04	3.559000e+04	3.559000e+04	3.559000e+04	3.559000e+04	3.559000e+04	3.559000e+04
mean	1.357598e-17	-9.662902e-17	-1.397527e-16	1.200875e-16	4.981186e-16	-2.671273e-16	-1.253782e-16	3.214312e-16	8.385163e-17
std	1.000014e+00	1.000014e+00	1.000014e+00	1.000014e+00	1.000014e+00	1.000014e+00	1.000014e+00	1.000014e+00	1.000014e+00
min	-1.259400e+00	-2.955154e+00	-2.010204e+00	-1.881940e+00	-2.682154e+00	-2.608800e+00	-1.773034e+00	-1.832641e+00	-3.077112e+00
25%	-8.512384e-01	-9.910298e-01	-8.306941e-01	-6.670143e-01	-6.196234e-01	-6.393461e-01	-9.104560e-01	-9.105526e-01	-9.094147e-01
50%	-1.852907e-01	-8.967972e-03	-2.409394e-01	-5.955150e-02	9.777852e-02	-1.469827e-01	-2.990787e-02	-2.462463e-02	-2.196798e-02
75%	5.451037e-01	9.730939e-01	9.385700e-01	5.479113e-01	6.358300e-01	8.377440e-01	7.338329e-01	7.347422e-01	7.345440e-01
max	4.863023e+00	9.730939e-01	1.528325e+00	2.370300e+00	2.608685e+00	3.053379e+00	3.195774e+00	3.202685e+00	3.149563e+00

- 스케일링과 훈련검증용 분할 후 X_test 데이터

	carat	cut	color	clarity	depth	table	x	y	z
count	15253.000000	15253.000000	15253.000000	15253.000000	15253.000000	15253.000000	15253.000000	15253.000000	15253.000000
mean	-0.004784	0.016915	0.007675	-0.008216	-0.026560	0.001181	-0.005318	-0.005306	-0.007813
std	1.003368	0.983908	1.004255	0.993154	0.988382	0.988529	1.001929	1.001555	1.001045
min	-1.259400	-2.955154	-2.010204	-1.881940	-2.682154	-2.805745	-1.764049	-1.805521	-2.131472
25%	-0.851238	-0.991030	-0.830694	-0.667014	-0.619623	-0.639346	-0.919441	-0.910553	-0.923963
50%	-0.185291	-0.008968	-0.240939	-0.059551	0.008103	-0.146983	-0.038893	-0.033665	-0.036516
75%	0.545104	0.973094	0.938570	0.547911	0.635830	0.837744	0.742818	0.734742	0.734544
max	4.325969	0.973094	1.528325	2.370300	2.608685	2.954907	2.989114	2.958602	3.033176

[5] 모델선정(RandomForestRegressor-R2score 설명력 98%)

	Model	R2 Score	Mean Squared Error	Root Mean Squared Error	Mean Absolute Error	Mean Absolute Percentage Error
0	LinearRegression()	0.917902	1.306392e+06	1142.974878	769.619653	0.426232
1	Ridge()	0.917900	1.306425e+06	1142.989529	769.614486	0.426240
2	Lasso()	0.917654	1.310341e+06	1144.701192	769.175932	0.425478
3	ElasticNet()	0.844915	2.467808e+06	1570.925971	1061.311759	0.437912
4	SGDRegressor()	0.917421	1.314049e+06	1146.319698	773.842664	0.428945
5	AdaBoostRegressor()	0.921825	1.243969e+06	1115.333783	888.102031	0.511218
6	GradientBoostingRegressor()	0.976956	3.666922e+05	605.551181	329.949621	0.104946
7	RandomForestRegressor()	0.981799	2.896218e+05	538.165179	261.472172	0.063237
8	DecisionTreeRegressor()	0.967391	5.188933e+05	720.342512	346.193765	0.082760

[6] RandomizedSearchCV를 활용한 하이퍼파라미터 선정

```
from sklearn.model_selection import RandomizedSearchCV
rf = RandomForestRegressor()
rf_random = RandomizedSearchCV(estimator=rf,
                               param_distributions=random_grid,
                               cv = 3)

rf_random.fit(X_train,y_train)
```

```
▸ RandomizedSearchCV
▸ best_estimator_: RandomForestRegressor
    ▸ RandomForestRegressor
```

```
# 최적 파라미터 결과
rf_random.best_params_
```

```
{'n_estimators': 136,
 'min_samples_split': 5,
 'min_samples_leaf': 4,
 'max_depth': 50}
```

```
# 최적 파라미터 적용
rf = RandomForestRegressor(n_estimators=136,
                           min_samples_split=5,
                           min_samples_leaf=4,
                           max_depth=50)
rf.fit(X_train,y_train)
y_pred = rf.predict(X_test)
r2_score(y_test,y_pred)
```

```
0.9814360316113937
```

[7] 실제값과 예측값 비교 시각화