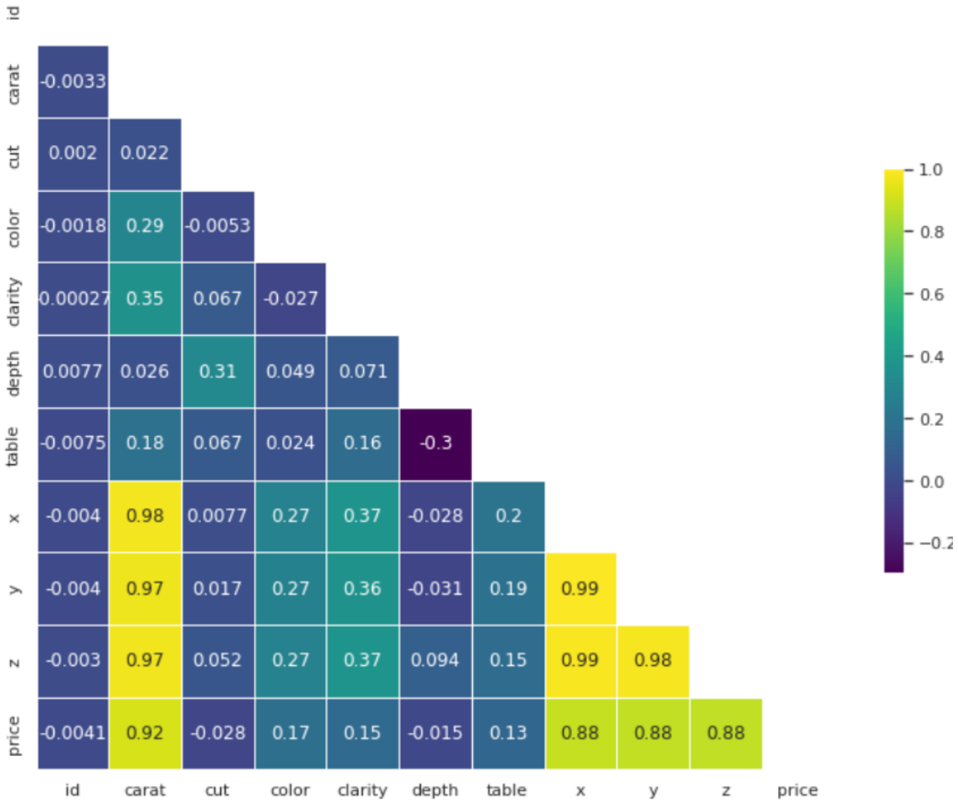
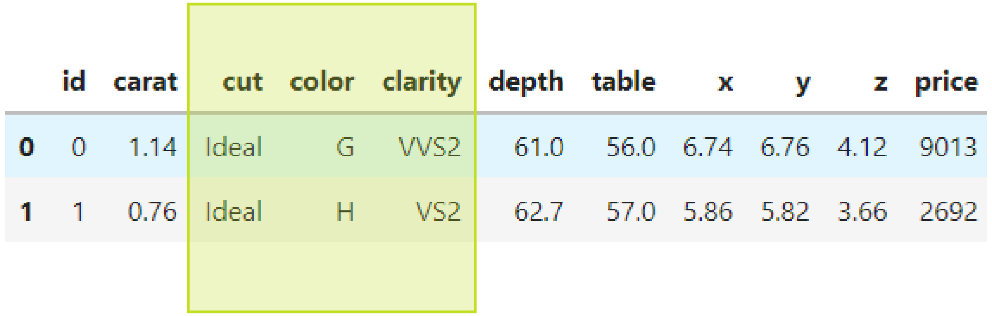
Limpieza

- Combatimos colinealidad
- Convertimos las categóricas en cuantitativas



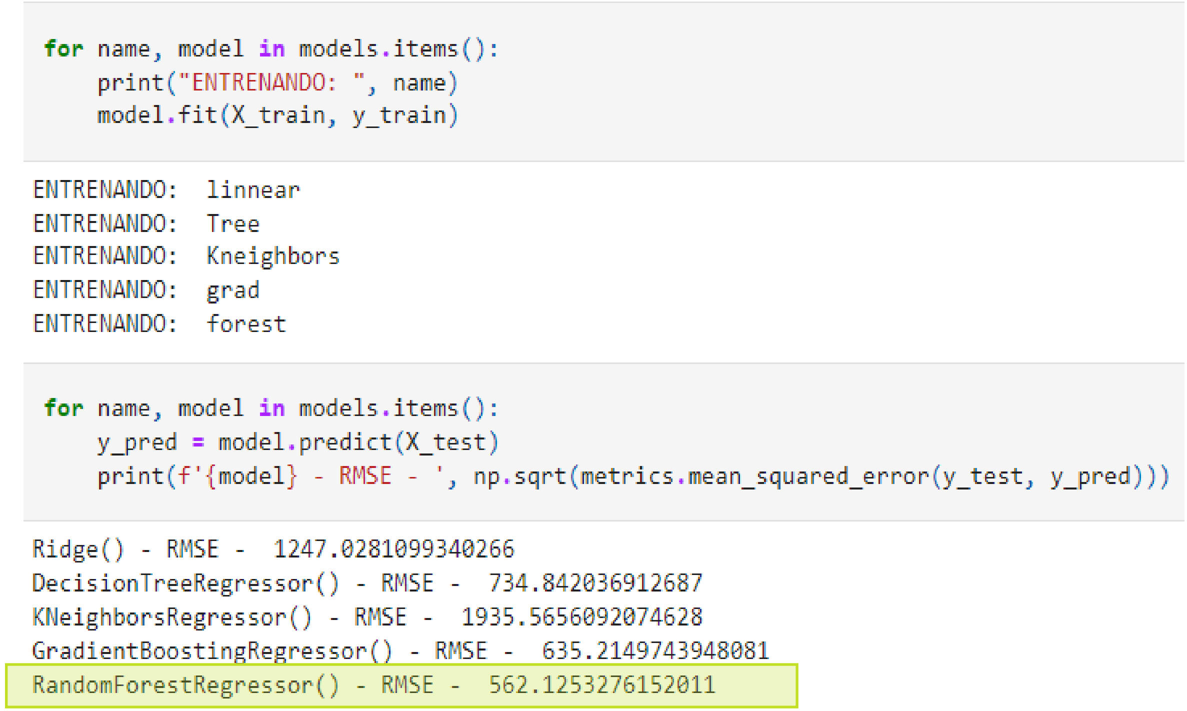
	id	carat	cut	color	clarity	depth	table	x	y	z	price
id											
carat	-0.0033										
cut	0.002	0.022									
color	-0.0018	0.29	-0.0053								
clarity	0.00027	0.35	0.067	-0.027							
depth	0.0077	0.026	0.31	0.049	0.071						
table	-0.0075	0.18	0.067	0.024	0.16	-0.3					
x	-0.004	0.98	0.0077	0.27	0.37	-0.028	0.2				
y	-0.004	0.97	0.017	0.27	0.36	-0.031	0.19	0.99			
z	-0.003	0.97	0.052	0.27	0.37	0.094	0.15	0.99	0.98		
price	-0.0041	0.92	-0.028	0.17	0.15	-0.015	0.13	0.88	0.88	0.88	



	id	carat	cut	color	clarity	depth	table	x	y	z	price
0	0	1.14	Ideal	G	VVS2	61.0	56.0	6.74	6.76	4.12	9013
1	1	0.76	Ideal	H	VS2	62.7	57.0	5.86	5.82	3.66	2692

Predecimos

- Elegimos modelo
- Aplicamos modelo

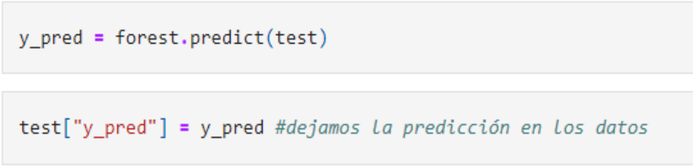


```
for name, model in models.items():
    print("ENTRENANDO: ", name)
    model.fit(X_train, y_train)
```

```
ENTRENANDO:  linnear
ENTRENANDO:  Tree
ENTRENANDO:  Kneighbors
ENTRENANDO:  grad
ENTRENANDO:  forest
```

```
for name, model in models.items():
    y_pred = model.predict(X_test)
    print(f'{model} - RMSE - ', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Ridge() - RMSE -  1247.0281099340266
DecisionTreeRegressor() - RMSE -  734.842036912687
KNeighborsRegressor() - RMSE -  1935.5656092074628
GradientBoostingRegressor() - RMSE -  635.2149743948081
RandomForestRegressor() - RMSE -  562.1253276152011
```



```
y_pred = forest.predict(test)
```

```
test["y_pred"] = y_pred #dejamos la predicción en los datos
```