MAKİNE ÖĞRENMESİ

PROJE RAPORU

Yağmur Özden – 170101046

Mert Burkay Karadeli - 150101037

Elmas Nedir?

Elmas, bilinen en sert maddelerden biridir ve değerli bir taştır. Karbon elementinin bir modifikasyonu grafit, diğeri ise elmastır.

Projenin amacı nedir?

Elmas veri setinde bulunan elmasın 10 farklı özelliğini makine öğrenmesi algoritmalarından birisini kullanarak sahip oldukları özelliklere elmasların satılabileceği/satın alınabileceği fiyatı tahmin etmektir.

Girişler nelerdir?

1) Elmasın karatı
2) Elmasın kesimi
3) Elmasın rengi
4) Elmasın berraklığı
5) Elmasın derinliği
6) Elmasın masası (yüzeyindeki düz fasettir)
7) X eksenindeki büyüklüğü
8) Y eksenindeki büyüklüğü
9) Z eksenindeki büyüklüğü

Çıkışlar nelerdir?

1) Elmasın Fiyatı

Veri Seti nereden alındı?

Goggle'ın kurmuş olduğu kaggle internet sitesinden alındı.

https://www.kaggle.com/shivam2503/diamonds

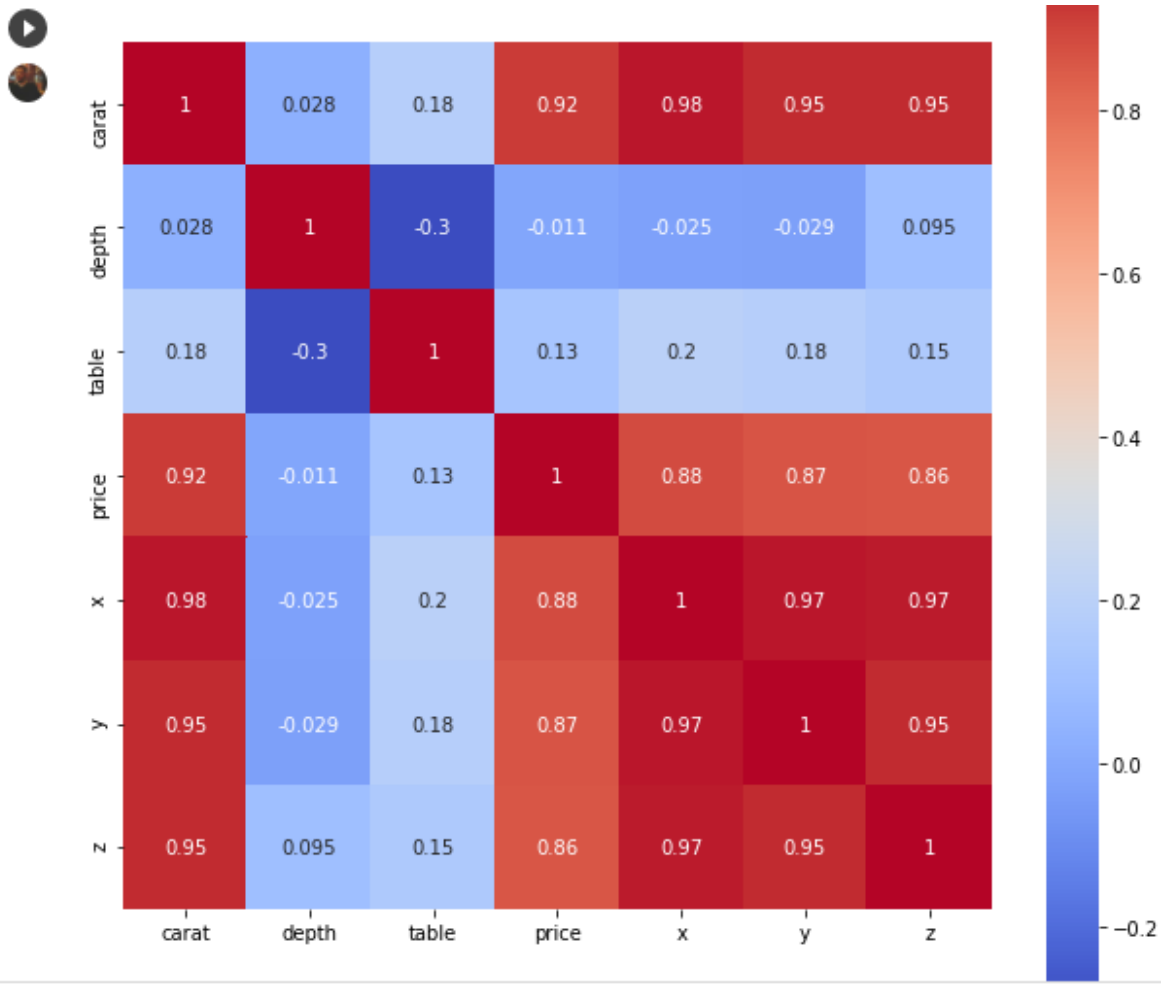
Projenin faydası nedir?

Birden fazla özelliklerinin hesaplanmasıyla elmasların fiyatlarının belirlenmesinde aktif rol oynamaktadır. Bu sayede tam olarak değerini hesaplayıp onun üzerinden al/sat işlemleri daha doğru yapılabilir.

Makine Öğrenmesi Yöntemleri

Projeyi gerçekleştirirken google colaboratory programıyla birlikte Python programlama dili kullanıldı.

Korelasyon Matrisi



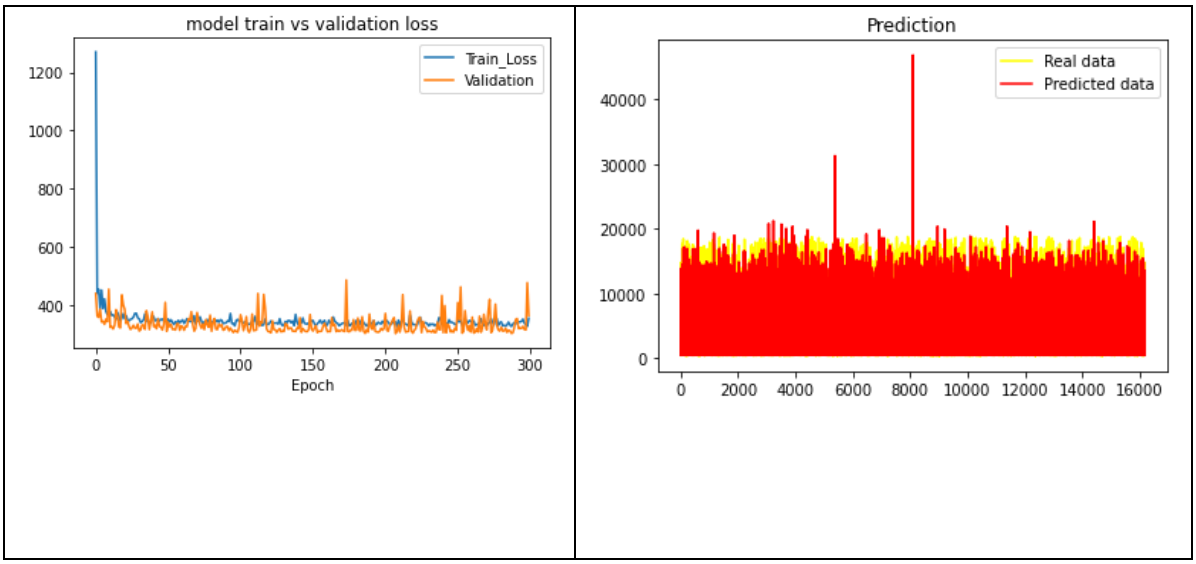
Değişkenler arasındaki korelasyon katsayılarının tablosu yukarıda gözüktüğü gibidir. Bu tabloda bir değişkenin diğer değişkenler ile arasındaki korelasyon görünmektedir.

Yapay Sinir Ağı Modeli

Elimizde negatif bir veri bulunmadığından parametrelerin daha hızlı öğrenilebilmesi adına ReLU kullanılmıştır.

Nöron değerleri fonksiyona sokarak fonksiyonun kuralına göre çıkış değeri bulan algoritmadır.

ReLU Fonksiyonu: Doğrultulmuş lineer birim (rectified linear unit- RELU) doğrusal olmayan bir fonksiyondur. ReLU fonksiyonu negatif girdiler için 0 değerini alırken, x pozitif girdiler için x değerini almaktadır.



Learning Rate: 0,5

Epoch: 300

Batch size: 256

Mean absolute Error: 388,1220062693387

R2 Score: 0,9599946213772175

Mean Squared Error: 643240,8405180847

Secret Layer Number: 2

Explained Variance Score: 0,9627837601115241

Train los tahmin ettirilen veridir. Validation değeri ise tahmin edilen verilerdir. Eğer validation değeri train loss'un üzerinde olursa overfitting olur. Eğer validation değeri train lossun altında kalırsa underfitting olur. Overfitting makinenin verileri ezberlemesi demektir. Yani teste sokulan veriler çoktur makine onları ezberlemiştir, aynı sonuçlar için doğru sonuç verir ancak farklı değerler için yanlışsonuç verir. Underfitting ise makinenin değerleri az öğrenmesi anlamına gelir. Bu durumda ise her sonuç için hata oranı yüksektir.

Burada öğrenme başarıyla gerçekleşmiştir. Gerçek veri ile tahmin edilen veriler arasında büyük uçurumlar oluşmamıştır. Validation değeri ile train loss değerleri arasında büyük farklar yoktur.

Support Vector Machines

Support Vektor Regresyonu iki sınıfı ayıran bir çizgi veya hiper düzlem (çok boyutlu uzayda) bulmaya çalışır. Daha sonra tahmin edilecek sınıflara bağlı olarak yeni noktayı hiper düzlemin pozitif mi yoksa negatif tarafında mı olduğuna göre sınıflandırır.

Elimizdeki veri seti için datasetindeki string değerler integer yapılır.

test_size = 0.45
random_state=101

SVC() kütüphanesinden nesne oluşturulduktan sonra fit() fonksiyonu çağırılır ve alt satırdaki çıktı elde edilir.

SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',max_iter=-1, probability=False, random_state=None, shrinking=True, =0.001, verbose=False)
Daha sonra grid fonksiyonu kullanılarak model eğitilir. Ve alttaki sonuçlar elde edilir.

```
grid_predictions = grid.predict(X_test)
print(confusion_matrix(Y_test,grid_predictions))
```

```
[[ 199  189   78  133  122]
 [ 120  529  409  556  591]
 [  10  220 7841  715  864]
 [  44  363 1147 3434 1270]
 [  67  446 1755 1387 1784]]
```

```
print(classification_report(Y_test,grid_predictions))
```

```
              precision    recall  f1-score   support

        Fair       0.45      0.28      0.34       721
        Good       0.30      0.24      0.27      2205
       Ideal       0.70      0.81      0.75      9650
     Premium       0.55      0.55      0.55      6258
   Very Good       0.39      0.33      0.35      5439

    accuracy                           0.57     24273
   macro avg       0.48      0.44      0.45     24273
weighted avg       0.55      0.57      0.55     24273
```

Naïve Bayes

Naïve Bayes, verilen özellik vektörünün bir etiketle ilişkilendirilme olasılığını türeten Bayes teoremine dayalı bir sınıflandırma yöntemidir. Naïve Bayes, her özellik için saf bir koşullu bağımsızlık varsayımına sahiptir; bu, algoritmanın özelliklerin bağımsız olmasını beklediği anlamına gelir ki bu her zaman böyle değildir.

Öncelikle string değerler integer'a dönüştürülür.

test_size = 0.33
random_state = 72

GaussionNB kütüphanesinden nesne oluşturulur ve kütüphanenin fonksiyonları kullanılır.

accuracy_score= 0.6134486826582777 elde edilir.

Lineer Regression

Lojistik regresyon, belirli bir sınıfa ait bir numunenin olasılığını öğrenen doğrusal bir sınıflandırma yöntemidir. Lojistik regresyon, sınıfları en iyi şekilde ayıran optimal karar sınırını bulmaya çalışır.

String değerler integera dönüştürülür. Giriş değerlerinden oluşan bir liste oluşturulur.

LogisticRegression() kütüphanesinden nesne oluşturulur.

```
lm = LogisticRegression()
lm.fit(X_train,Y_train)
predictions = lm.predict(X_test)
print(classification_report(Y_test,predictions))
```

```
              precision    recall  f1-score   support

        Fair       0.00      0.00      0.00       800
        Good       0.00      0.00      0.00      2453
       Ideal       0.40      0.93      0.56     10705
     Premium       0.36      0.12      0.18      6972
   Very Good       0.00      0.00      0.00      6040

    accuracy                           0.40     26970
   macro avg       0.15      0.21      0.15     26970
weighted avg       0.25      0.40      0.27     26970
```

Decision Trees Reggression

Karar ağacı, bir ağaç yapısı biçiminde regresyon veya sınıflandırma modelleri oluşturur. Bir veri kümesini daha küçük ve daha küçük alt kümelere ayırırken, aynı zamanda ilişkili bir karar ağacı aşamalı olarak geliştirilir. Nihai sonuç, karar düğümleri ve yaprak düğümleri olan bir ağaçtır. Bir karar düğümünün, her biri test edilen öznitelik için değerleri temsil eden iki veya daha fazla dalı vardır. Yaprak düğümü, sayısal hedefle ilgili bir kararı temsil eder. Kök düğüm adı verilen en iyi tahmin ediciye karşılık gelen, bir ağaçtaki en üstteki karar düğümü. Karar ağaçları hem kategorik hem de sayısal verileri işleyebilir.

Decision Tree kütüphanesinden nesne oluşturulur ve fit fonksiyonu çağırılır. Çıktı alttaki gibidir.

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini', _depth=None, max_features=None, max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None, _samples_leaf=1,
min_samples_split=2, _weight_fraction_leaf=0.0,
presort='deprecated',random_state=None, splitter='best')
```

Classification raporu ise aşağıdaki gibidir

```
print(classification_report(Y_test,predictions))
```

	precision	recall	f1-score	support
326	0.00	0.00	0.00	1
334	0.00	0.00	0.00	1
335	0.00	0.00	0.00	1
336	0.00	0.00	0.00	1
337	0.00	0.00	0.00	1
338	0.00	0.00	0.00	1
339	0.00	0.00	0.00	1
340	0.00	0.00	0.00	0
344	0.00	0.00	0.00	0
345	0.00	0.00	0.00	0
351	0.50	1.00	0.67	2
352	0.00	0.00	0.00	1
353	0.50	1.00	0.67	1
357	0.75	1.00	0.86	3
358	0.00	0.00	0.00	1
360	1.00	1.00	1.00	1
362	1.00	1.00	1.00	1
363	1.00	1.00	1.00	2
364	1.00	1.00	1.00	1
367	0.50	1.00	0.67	3
368	0.50	0.33	0.40	3
369	0.00	0.00	0.00	1
373	1.00	0.75	0.86	4
374	1.00	1.00	1.00	2

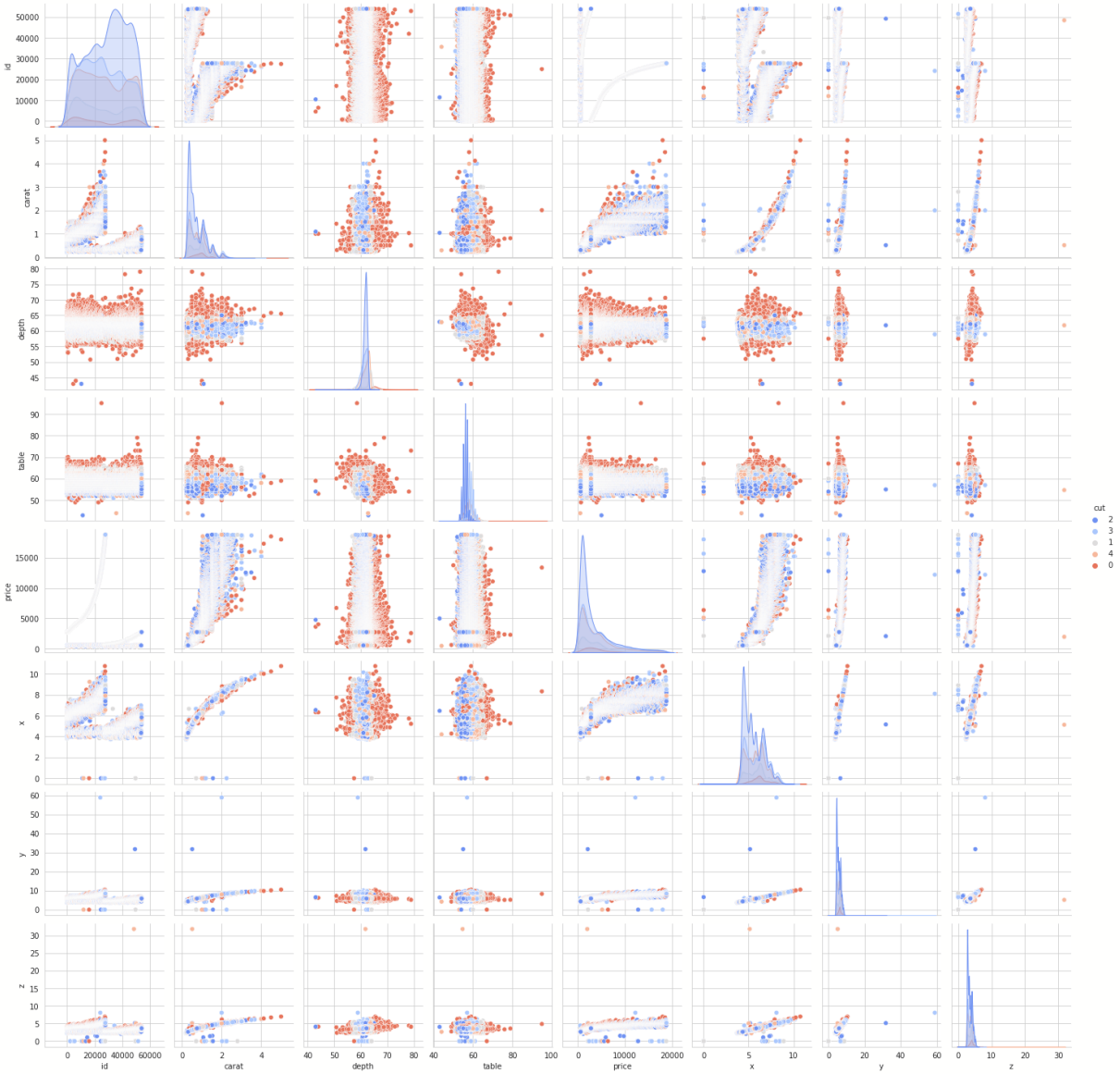
K Nearest Neighbors

KNN regresyonu, sezgisel bir şekilde, aynı mahalledeki gözlemlerin ortalamasını alarak bağımsız değişkenler ile sürekli sonuç arasındaki ilişkiye yaklaşan parametrik olmayan bir yöntemdir.

`LabelEncoder kütüphanesinin nesnenini oluşturulur ve fit fonksiyonu çağırılır.`

```
test_size = 0.2
random_state = 101
```

Sonuç olarak aşağıdaki grafikler elde edilir.



Burada accuary score'u ve classification raporu görülür.dvskmdfksldmff

```
classifier = KNeighborsClassifier(n_neighbors=3)
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
confusion_matrix(Y_test, Y_pred)
accuracy = accuracy_score(Y_test, Y_pred)*100
print('Accuracy of our model is equal ' + str(round(accuracy, 2)) + ' %.')
print(classification_report(Y_test,Y_pred))
```

```
Accuracy of our model is equal 52.93 %.
              precision    recall  f1-score   support

           0       0.40      0.25      0.31       324
           1       0.26      0.28      0.27       973
           2       0.63      0.80      0.70      4273
           3       0.54      0.49      0.51      2790
           4       0.37      0.24      0.29      2428

    accuracy                           0.53     10788
   macro avg       0.44      0.41      0.42     10788
weighted avg       0.51      0.53      0.51     10788
```

Knn algoritmasına göre optimum komşu 1 dir. Ve aşağıdaki grafikten kontrol edilebilir.