T.C KAHRAMANMARAŞ SÜTÇÜ İMAM ÜNİVERSİTESİ

MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ

MAKİNE ÖĞRENMESİ

PROJE YAPANLAR

ABDALLAH DAHER ABDİLLE - 18110131516

MAHMUT KIRDI - 19110131309

Proje Konusu

Altın Fiyat Tahmini

Proje Amacı

Altın Fiyatının, Date,SPX,GLD,USO,SLV,EUR/USD öznitelik verileri ile tahmin edilme işlemidir. Böylece Altın alım-satım gibi işlemlerde bir tavsiye ederek yardımcı olur.

Proje İçeriği

Projemizdeki veriler Kaggle'da Veri Seti şeklinde bulunuyor. 6 sütun ve 2290 satırdan oluşmaktadır. Aldığımız veri seti bu şekildedir.

Öznitelikler aşağıda açıklaması bulunmaktadır ;

Date : Tarih

SPX : ABD 500 Endeksi(En büyük ilk 500 şirket)

GLD : Altın Borsa Yatırım Fonu

USO : ABD Petrol Fonu

SLV : Gümüş Borsası

EUR/USD : Euro/Dolar Endeksi

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        from sklearn.model_selection import train_test_split
        from sklearn.ensemble import RandomForestRegressor
        from sklearn import metrics
```

Burada Gerekli Kütüphaneleri import (içeri aktardık) ettik.

1) NumPy, dizilerle çalışmak için kullanılan bir Python kütüphanesidir. Ayrıca doğrusal cebir, fourier dönüşümü ve matrisler alanında çalışmak için de gerekli işlevlere sahiptir. Python'da dizilerin amacına hizmet eden listelerimiz var, ancak işlenmesi yavaştır. NumPy, geleneksel Python listelerinden 50 kata kadar daha hızlı bir dizi nesnesi sağlamayı amaçlamaktadır. NumPy'deki dizi nesnesi (array) ndarray olarak adlandırılır ve ndarray, çalışmayı çok kolaylaştıran birçok destekleyici işlev sağlar. as np demek kısaltma demektir. Numpy'i daha esnek ve basit kullanabilmek için yapılır.

2) Pandas'ın NumPy'daki metotlara benzer metotları vardır. NumPy aynı veri tipleri ile çalışırken Pandas farklı veri tipleri ile de çalışabilir. as pd demek kısaltma demektir. Pandas'ı daha esnek ve basit kullanabilmek için yapılır. Örneğin pd. yazıp tab tuşuna basarsak pandasın içindeki metotları görebilirsiniz.

3) Matplotlib; veri görselleştirmesinde kullandığımız temel python kütüphanesidir. 2 ve 3 boyutlu çizimler yapmamızı sağlar. Matplotlib genelde 2 boyutlu çizimlerde kullanılır.

4) Seaborn, Matplotlib kütüphanesi tabanlı, istatiksel bir Python veri görselleştirme kütüphanesidir. Seaborn kullanıcılara istatiksel görselleştirmeler yapmaları için high-level (yüksek seviyeli) bir arayüz sunar.

5) Yaygın olarak kullanılan açık kaynak kodlu *machine learning* kütüphanesidir. Veri kümesini ikiye bölmek için kullanıyoruz.

6) Random Forest Regresyon, scikit-learn kütüphanesi ensemble modülünün bir sınıfı olarak tanımlanmış. Sınıfımız RandomForestRegressor. Bu sınıftan oluşturacağımız nesneyi, yani regressor, makine yapacak. Modelimizi eğitmek için öncelikle bu sınıftan regressor adında bir nesne oluşturuyoruz. Daha sonra bu nesnenin fit() metoduna X, Y değişkenlerimizi parametre olarak veriyoruz. Böylelikle makinemizi kurmuş oluyoruz.

7) sklearn kütüphanesi ile metrics sınıfı oluşturuyoruz.

```
In [2]: # .csv uzantılı veri setimizi Pandas DataFrame'e yükleme
        gold_data = pd.read_csv('gld_price_data.csv')
```

.csv uzantılı veri setimizi Pandas DataFrame'e yükleme işlemi yapıyoruz. Pandas ile .csv Dosyası Okuma işlemi yapıldı.

```
In [3]: # veri çerçevesindeki ilk 5 satırı yazdır
        gold_data.head()
```

Out[3]:

	Date	SPX	GLD	USO	SLV	EUR/USD
0	1/2/2008	1447.160034	84.860001	78.470001	15.180	1.471692
1	1/3/2008	1447.160034	85.570000	78.370003	15.285	1.474491
2	1/4/2008	1411.630005	85.129997	77.309998	15.167	1.475492
3	1/7/2008	1416.180054	84.769997	75.500000	15.053	1.468299
4	1/8/2008	1390.189941	86.779999	76.059998	15.590	1.557099

Veri çerçevesindeki ilk 5 satıra bakıyoruz.

```
In [4]: # Veri çerçevesinin son 5 satırını yazdır
        gold_data.tail()
```

Out[4]:

	Date	SPX	GLD	USO	SLV	EUR/USD
2285	5/8/2018	2671.919922	124.589996	14.0600	15.5100	1.186789
2286	5/9/2018	2697.790039	124.330002	14.3700	15.5300	1.184722
2287	5/10/2018	2723.070068	125.180000	14.4100	15.7400	1.191753
2288	5/14/2018	2730.129883	124.489998	14.3800	15.5600	1.193118
2289	5/16/2018	2725.780029	122.543800	14.4058	15.4542	1.182033

Veri çerçevesindeki son 5 satıra bakıyoruz.

```
In [5]: # satır ve sütun sayısını alma
        gold_data.shape
```

Out[5]: (2290, 6)

Satır ve sütun sayısını aldık.

```
In [6]: # veriler hakkında bazı temel bilgileri almak
        gold_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2290 entries, 0 to 2289
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Date      2290 non-null   object 
 1   SPX       2290 non-null   float64
 2   GLD       2290 non-null   float64
 3   USO       2290 non-null   float64
 4   SLV       2290 non-null   float64
 5   EUR/USD   2290 non-null   float64
dtypes: float64(5), object(1)
memory usage: 107.5+ KB
```

Veriler hakkında bazı temel bilgileri alma işlemi yaptık.

```
In [7]: # eksik değerlerin sayısını kontrol etme
        gold_data.isnull().sum()
```

```
Out[7]: Date       0
        SPX        0
        GLD        0
        USO        0
        SLV        0
        EUR/USD    0
        dtype: int64
```

Eksik değerlerin sayısını kontrol etme işlemi yaptık.

```
In [8]: # verilerin istatistiksel ölçülerini almak
        gold_data.describe()
```

Out[8]:

	SPX	GLD	USO	SLV	EUR/USD
count	2290.000000	2290.000000	2290.000000	2290.000000	2290.000000
mean	1654.315776	122.732875	31.842221	20.084997	1.283653
std	519.111540	23.283346	19.523517	7.092566	0.131547
min	676.530029	70.000000	7.960000	8.850000	1.039047
25%	1239.874969	109.725000	14.380000	15.570000	1.171313
50%	1551.434998	120.580002	33.869999	17.268500	1.303297
75%	2073.010070	132.840004	37.827501	22.882500	1.369971
max	2872.870117	184.589996	117.480003	47.259998	1.598798

Verilerin istatistiksel ölçülerini alma işlemini yaptık.

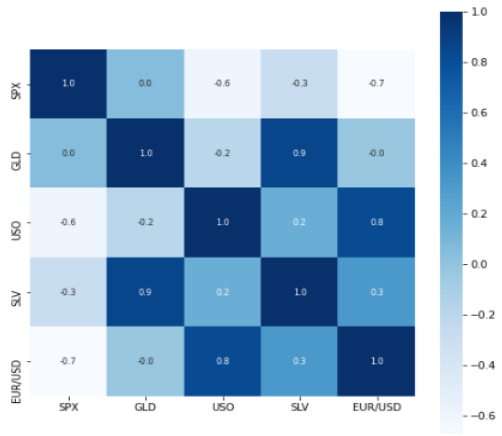
```
In [9]: correlation = gold_data.corr()
```

Korelasyon işlemi yaptık;

1. Pozitif Korelasyon
2. Negatif Korelasyon

Şeklinde 2 tür korelasyon aldık.

```
In [10]: # korelasyonu anlamak için bir ısı haritası oluşturmak
         plt.figure(figsize = (8,8))
         sns.heatmap(correlation, cbar=True, square=True, fmt='.1f',annot=True, annot_kws={'size':8}, cmap='Blues')

Out[10]: <AxesSubplot:>
```

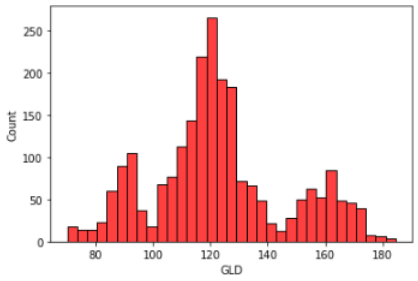


Korelasyonu anlamak için bir ısı haritası oluşturduk.

```
In [11]: # GLD'nin korelasyon değerleri
         print(correlation['GLD'])

SPX        0.049345
GLD        1.000000
USO       -0.186360
SLV        0.866632
EUR/USD   -0.024375
Name: GLD, dtype: float64
```

GLD’nin korelasyon değerlerini yazdırdık.

```
In [35]: # GLD Fiyatının dağılımının kontrol edilmesi
         sns.histplot(gold_data['GLD'],color='red')

Out[35]: <AxesSubplot:xlabel='GLD', ylabel='Count'>
```



GLD Fiyatının dağılımının kontrol edilmesi ve bunun grafik ile gösterimi.

```
In [13]: X = gold_data.drop(['Date','GLD'],axis=1)
         Y = gold_data['GLD']
```

Özellikleri ve Hedefi Bölme işlemi yaptık.

```
In [14]: print(X)
```

```
              SPX        USO      SLV   EUR/USD
0     1447.160034  78.470001  15.1800  1.471692
1     1447.160034  78.370003  15.2850  1.474491
2     1411.630005  77.309998  15.1670  1.475492
3     1416.180054  75.500000  15.0530  1.468299
4     1390.189941  76.059998  15.5900  1.557099
...           ...        ...      ...       ...
2285  2671.919922  14.060000  15.5100  1.186789
2286  2697.790039  14.370000  15.5300  1.184722
2287  2723.070068  14.410000  15.7400  1.191753
2288  2730.129883  14.380000  15.5600  1.193118
2289  2725.780029  14.405800  15.4542  1.182033

[2290 rows x 4 columns]
```

```
In [15]: print(Y)
```

```
0        84.860001
1        85.570000
2        85.129997
3        84.769997
4        86.779999
           ...
2285    124.589996
2286    124.330002
2287    125.180000
2288    124.489998
2289    122.543800
Name: GLD, Length: 2290, dtype: float64
```

```
In [16]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state=2)
```

Verilerimizi Eğitim Verileri ve Test Verilerine Bölme İşlemi gerçekleştirdik.

```
In [17]: regressor = RandomForestRegressor(n_estimators=100)
```

Model Eğitimi : Random Forest Regressor(Rastgele Orman Regrasyonu)

```
In [39]: # modeli eğitme işlemi
         regressor.fit(X_train,Y_train)

Out[39]: RandomForestRegressor()
```

Modeli training etme işlemi (eğitme aşaması) yaptık.

```
In [19]: # Test Verileri üzerinde tahmin işlemi
         test_data_prediction = regressor.predict(X_test)
```

Model Değerlendirmesi işlemi

```
In [19]: # Test Verileri üzerinde tahmin işlemi
         test_data_prediction = regressor.predict(X_test)
```

Test verileri üzerinde tahmin işlemi yaptık.

```
In [20]: print(test_data_prediction)
```

Tahmin işlemini print ile yazdırdık.

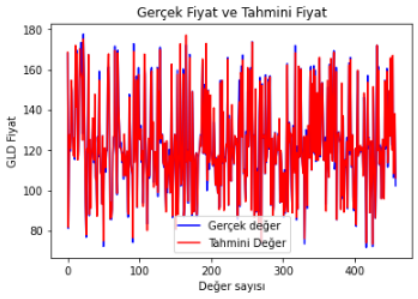
```
In [21]: # R kare hata oranı
         error_score = metrics.r2_score(Y_test, test_data_prediction)
         print("R kare hata oranı : ", error_score)

         R kare hata oranı :  0.9893606307200276
```

R kare hata oranını kontrol ettik.

```
In [22]: Y_test = list(Y_test)
```

Bir Grafikte Gerçek Değerleri ve Tahmini Değerleri Karşılaştırma işlemi yaptık.

```
In [38]: plt.plot(Y_test, color='blue', label = 'Gerçek değer')
         plt.plot(test_data_prediction, color='red', label='Tahmini Değer')
         plt.title('Gerçek Fiyat ve Tahmini Fiyat')
         plt.xlabel('Değer sayısı')
         plt.ylabel('GLD Fiyat')
         plt.legend()
         plt.show()
```



Son olarak da bunu grafik ile gösterdik.