Proje Adı: ZAMAN SERİLERİ ANALİZİ İLE BİR SONRAKİ GÜN İÇİN KURDAKİ ARTIŞIN VEYA AZALIŞIN OTOMATİK OLARAK SAPTANMASI VEYA BORSA ANALİZİ

LSTM Modeli Kullanarak Hisse Senedi Fiyat Tahmini ;

1). Uygulamada kullandığımız veriseti görüntüsü

	Date	High	Low	Open	Close	VOLUME	Adj Close
2293	07.03.2019	104194	102264	103854	102283	1529152000	102283
2294	08.03.2019	102464	100874	102337	101538	1167492000	101538
2295	11.03.2019	102343	100652	101887	100850	1104289000	100850
2296	12.03.2019	102069	100919	101465	101721	1167495000	101721
2297	13.03.2019	102532	101143	101821	102197	1219604000	102197
2298	14.03.2019	102786	102010	102298	102432	1222111000	102432
2299	15.03.2019	103380	102644	102891	103304	1505157000	103304
2300	18.03.2019	104883	103785	103785	104595	1406049000	104595
2301	19.03.2019	105929	104154	104664	104862	1671553000	104862
2302	20.03.2019	105099	103181	104807	103310	1853840000	103310
2303	21.03.2019	104248	102534	103836	103408	1550236000	103408
2304	22.03.2019	103756	99652.4	103654	99835.3	1785623000	99835.3
2305	25.03.2019	100382	98367.1	100028	99325.9	1626946000	99325.9
2306	26.03.2019	100260	97153.9	99990.5	97378.6	1464774000	97378.6
2307	27.03.2019	97998.4	90523.9	97820.7	91855.1	2007701000	91855.1
2308	28.03.2019	93744.3	90776.8	91181.6	92120.3	2038674000	92120.3
2309	29.03.2019	94247	91904.4	92499	93784.2	1277543000	93784.2
2310	01.04.2019	95117.5	91864.5	93908.4	94101.3	1612414000	94101.3
2311	02.04.2019	95352.2	93132.3	93635.1	93533.1	1244141000	93533.1
2312	03.04.2019	94891.3	94041.5	94158.6	94441.1	1083709000	94441.1
2313	04.04.2019	98336.9	94297.3	94477.9	98336.9	1496492000	98336.9
2314	05.04.2019	99214.1	97422.2	98963.2	98783.4	1449394000	98783.4
2315	08.04.2019	99076.9	96735.6	98978.9	96978.3	1095047000	96978.3
2316	09.04.2019	98748.3	97496.9	97956.7	98135.3	1283386000	98135.3
2317	10.04.2019	98456.1	96912.5	98404.7	97015	1210704000	97015
2318	11.04.2019	97530.9	96021.8	97227.2	96073	1045391000	96073

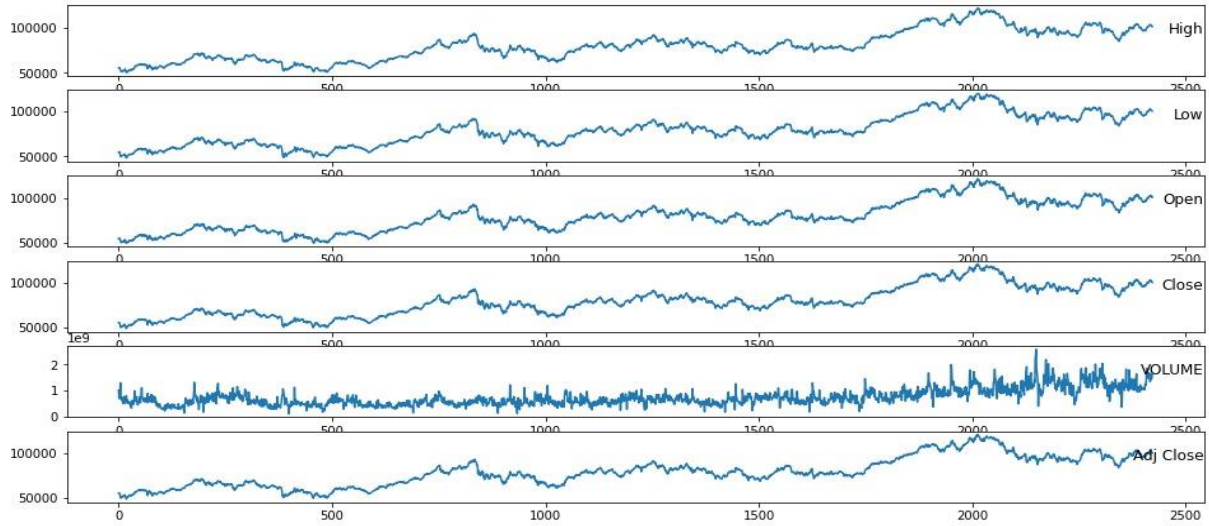
2). Uygulamanın geliştirilmesi için yüklediğimiz kütüphaneler,

```
from pandas import read_csv
from matplotlib import pyplot
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
import pandas as pd
from keras.models import Sequential
from keras.layers.recurrent import LSTM
from keras.layers import Dense, LSTM
from keras.layers import concatenate
import math
import numpy as np
```

3). Verisetinden aldığımız geçmiş tüm sütun verilerin grafiğini çizdirdik, kod bloğu;

```
# load dataset(4lu grafik cizer  1)
dataset = read_csv('XU100.csv', header=0, index_col=0)
data_ = dataset.head()
values = dataset.values
# specify columns to plot
groups = [0, 1, 2, 3, 4, 5]
i = 1
# plot each column
pyplot.figure(figsize=(16,8))
for group in groups:
  pyplot.subplot(len(groups), 1, i)
  pyplot.plot(values[:, group])
  pyplot.title(dataset.columns[group], y=0.5, loc='right')
  i += 1
pyplot.show()
```

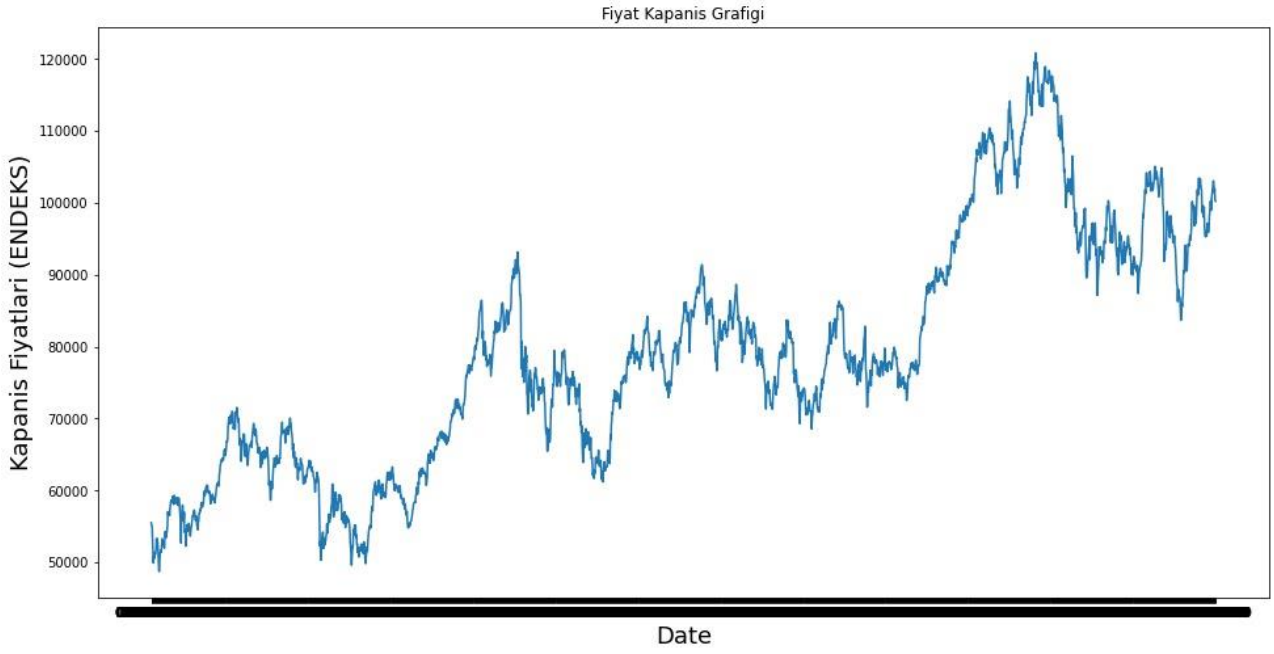
Ekran Çıktısı;



3). Veri setinden aldığımız geçmiş kapanış fiyatlarının grafiğini çizdirdik, kod bloğu;

```
#(kapanisa gore tekli grafik cizer  2)
pyplot.figure(figsize=(16,8))
pyplot.title('Fiyat Kapanis Grafigi')
pyplot.plot(dataset['Close'])
pyplot.xlabel('Date',fontsize=18)
pyplot.ylabel('Kapanis Fiyatlari (ENDEKS)',fontsize=18)
pyplot.show()
```

Ekran Çıktısı;



4).Veri setinde bulunan toplam kapanış gün sayısını belirledik.

Kod bloğu;

```
# sadece kapanis fiyati sutununa gore veribirimi olustur
dataclose = dataset.filter(['Close'])
dataset_close = dataclose.values # numpy arraya donusturur
#egitilen data modeli icin satir sayilarini al
training_data_len = math.ceil( len(dataset_close) * .8)
print(training_data_len)
```

Ekran çıktısı;

1939 (veri setimizde 1939 günü ifade eden satırlar var)

5). İlk günden son güne kadar olan kapanış fiyatlarını 0-1 aralığında ölçeklendirdik.

Kod bloğu; `# veri olceklendir`

```
# temel olarak olcekleme icin kullanilacak minimum ve maksimum degerleri
hesaplar
# bu iki degere dayanarak verileri donusturuyorlar
# degerlerin 0 ile 1 arasinda olacagini soyleyin
# bu bir hucre
scaler = MinMaxScaler(feature_range = (0,1))
scaled_data = scaler.fit_transform(dataset_close)
print(scaled_data)
```

Ekran Çıktısı;

```
[[0.09411966]
 [0.08887224]
 [0.08777746]
 ...
 [0.73767462]
 [0.71560568]
 [0.7141897 ]]
```

(ilk günden son güne kadar olan fiyat hareketlerinin 0-1 aralığına indirgenmiş halidir.)

6). Son 60 günü dikkate almak için son 60 günün veri çerçevesini tasarladık.

Kod bloğu;

```
# yeni bir egitim seti olusturacagiz
# bu hucrede tum egitim verileri setini olusturacagiz
# yapmak istedigimiz sey olcekli egitim veri setini olusturmak
train_data = scaled_data[0:training_data_len,:]#indeks 0 dan veri uzunluguna kadar geri dondurme
X_train = [] # sonraki tum sutunlari X trainine ve Y ye ayirir
Y_train = []
for i in range(60, len(train_data)):#60 aralikta egitim suresinin uzunlugu icin bir dongu olusturalim
    X_train.append(train_data[i-60:i,0])#son 60 degeri X egitim veri setimize ekleyelim x degeri 60 deger icerecek 0 konumundan 59 a kadar
    #ve ardindan Y kare train verisi altina endekslenir
    Y_train.append(train_data[i, 0]) # konumunda stun alacagiz bu yuzden ilk gecis icin ayarlanan 61. degeri 60. pozisyon simdi ne oldugunu gormek icin
    if i<=61 :
        print(" son 60 veri seti :\n{}".format(X_train))
        print(" 61. veri seti : \n {}" .format(Y_train))
```

Ekran çıktısı;

Çerçeveyi açıklayacak olursak sarı okla son 60 günün ilk verisi gösterilmekte. Sarının yanındaki kırımızı ok 2. Günü göstermektedir.

Bir sonraki döngünün gerçekleşmesiyle çerçeve bir adım kayacaktır. 2. Kırımızı ok işaretinin gösterdiği değere bakılacak olursa ilk döngünün 2. Günü değeriyle(ilk kırımızı ok gösterdiği değer) aynıdır. Yeşil oklar ile gösterilen 60 gün sonrası olan 61. Gündür. Alttaki 2 yeşil ok değerleri ise 61 + 62. Günleri göstermektedir.

6). LSTM modeli, girdi değerinin 3 boyutlu olmasını bekler, bu yüzden 2 boyutlu olan verimizi 3 boyutluya çevirdik.

Kod bloğu;

```
# simdi hucrede yeni bir hucre olusturacagiz
# Numpy dizilerini LSTM modelini egitmek icin kullanabiliriz
# X ve Y trainlerini numpy dizinisine donusturulmesi
X_train, Y_train = np.array(X_train), np.array(Y_train)
# bir LSTM agi. girdinin bir sayi biciminde 3 boyutlu olmasini bekler
# train veri seti 2 boyutludrur bu yuzden X tipi shape ile 2 boyutlu satir
sayisini ve sutun sayisini gorebilecegiz
#print(X_train.shape)#satir ve sutun 2 boyut gosterir (1879,60
#LSTM 3 boyutlu istedigi icin 3 boyuta cevirecegiz
X_train = np.reshape(X_train , ( X_train.shape[0], X_train.shape[1], 1 ))
print(X_train.shape)
```

Ekran Çıktısı;

```
(1879, 60, 1)
```

7). Tahmin için LSTM modeli oluşturduk.

Kod bloğu;

```
#LSTM modeli olusturulmasi
# LSTM katmani eklenmesi ve 50 n,ron eklenmesi input shape giris katmani 60
olan zaman adimlarinin ve 1 tekrar olan ozelliklerin sayisinin
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape = (X_train.shape[1],
1)))
#simdi ikinci katmanin ekleyelim 50 nöronu olacak ve donus dizileri...
model.add(LSTM( 50, return_sequences = False ))
#birkac katman daha ekleyecegiz yogun bir katman ekleyecegiz bu katman 25
nörona sahip olacak duzenli n,ron a[ katmani
model.add(Dense(25))
# bir katman daha ekleyecegiz 1 nörona sahip bir katman.
model.add(Dense(1))
# modelin derlenmesi
# modelimize optimize ediciyi vermeliyiz ve sonra kayip fonksiyonu vermeliyiz
model.compile(optimizer = 'adam' , loss='mean_squared_error')#modelin egitimde
ne kadar iyi oldugunu olcmek icin kullanilir
#modelin egitilmesi#islem uzun suruyor epochlu kisim
model.fit(X_train, Y_train, batch_size=1, epochs=1)
#yeni bir hucre olusturacagiz ve bu hucrede test yapacagiz
#1879 dan 2003 dizisine olceklendirilmis degerler iceren yeni bir dizi
olusturalim
#veri kumesinin sonu ve tum sutunlari geri alacagiz
test_data = scaled_data[training_data_len - 60: , :]
#Create data sets X_test and Y_test
X_test = []
Y_test = dataset_close[training_data_len:, :] #Y_testi, modelimizin tahmin
etmesini istedigimiz tum degerler olacaktir
for i in range(60, len(test_data)): # X_test veri setine son 60 degeri
ekleyecegiz
    X_test.append(test_data[i-60:i,0])
#yeni bir hucre olusturalim ve verileri numpy dizisine donusturelim
X_test = np.array(X_test)
# yeni hucre olusturalim ve bu hucrede verileri yeniden sekillendirelim
#verimiz 2 boyutlu LSTM 3 boyutlu istedigi icin 3 boyutluya cevirecegiz
X_test = np.reshape( X_test,(X_test.shape[0], X_test.shape[1], 1 ))#buda zaman
adimi sayisina esit ve daha sonra sahip oldugumuz ozellikler sadece yakin
fiyat iyi yani
#modellerin fiyat ve degerleri tahmin etmesini istemek icin
#X_train testi icin  modeldeki degerlerin tahmini fiyatini al
predictions = model.predict(X_test)
```

```
predictions = scaler.inverse_transform(predictions)# verileri tersine
donusturun boylece tahminler yazilir
# Y_test verilerimizle ayni degerleri icerecek tahminler
```

Ekran Çıktısı;

```
Epoch 1/1
 195/1879 [==>............................] - ETA: 1:26 - loss: 0.0036

Epoch 1/1
 474/1879 [======>.......................] - ETA: 1:22 - loss: 0.0024

Epoch 1/1
 740/1879 [==========>...................] - ETA: 1:09 - loss: 0.0021

Epoch 1/1
 982/1879 [==============>...............] - ETA: 55s - loss: 0.0018

Epoch 1/1
1545/1879 [=======================>......] - ETA: 20s - loss: 0.0014

Epoch 1/1
1879/1879 [==============================] - 101s 54ms/step - loss: 0.0017
```

Eğitilerek %80 'i oranına indirgenmiş verilerin tamamı LSTM modelinde işlendi.

8). Tahminin doğruluk derecesini değerlendirmek için **" kök ortalama kare hatası (RMSE) "** hesapladık.

Kod bloğu;

```
#kok ortalama kare hatasi veya kisa surede RMSE alma
# modelin yaniti  ve RMSE dusuk degerleri ne kadar dogru tahmin ettigini ve
kalintilarin standart sapmasi
# modelinizin ne kadar iyi bir performans gosterdigine dair gercekten bir
fikir edinmek icin
#mse = np.sqrt(np.mean(predictions - Y_test )**2)
#mse = np.sqrt(np.mean(np.power((np.array(Y_test)-np.array(predictions)),2)))
rmse = np.sqrt(((predictions - Y_test) ** 2).mean())
print(rmse)# bu yuzde bes puan, kok ortalama kare hatasi icin sifir degeri
tahminlerin dogru olmasi icin mukemmel oldugu anlamina gelir
print('Test RMSE: %.3f' % rmse)
```

Ekran çıktısı;

```
Test RMSE: 3315.119
```

NOT: RMSE değerini doğru hesaplayamadığımızı düşünüyoruz.Hesaplama için,

1. rmse =np.sqrt(np.mean(((predictions- y_test)**2)))
2. rmse = np.sqrt(np.mean(np.power((np.array(y_test)-np.array(predictions)),2)))
3. rmse = np.sqrt(((predictions - y_test) ** 2).mean())

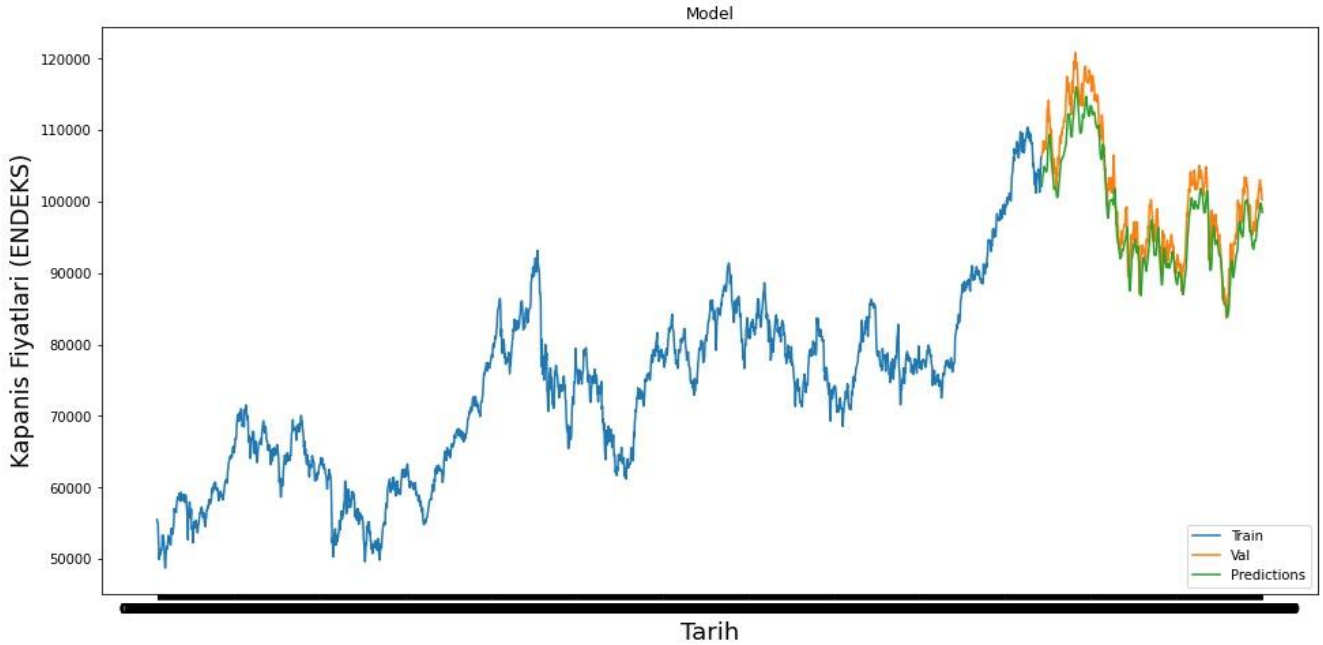
3 Farklı formül denedik ama sonucun yanlış olduğu düşüncesindeyiz. Tahminin doğruluk değerlendirmesini sonuçta gözlemleyebiliyoruz. Ama buradaki oran sebebini bulamadığımız bir nedenden dolayı hatalıdır düşüncesindeyiz.

9). LSTM modelin sonucunu yani tahmini grafiklendirdik.

Kod bloğu;

```
#verileri ciz
train = dataclose[:training_data_len]
valid = dataclose[training_data_len:]
valid['Predictions'] = predictions
#verileri gorsellestirme
plt.figure(figsize=(16,8))
plt.title('Model')
plt.xlabel('Tarih',fontsize=18)
plt.ylabel('Kapanis Fiyatlari (ENDEKS)',fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close','Predictions']])
plt.legend(['Train','Val','Predictions'], loc='lower right')
plt.show()
```

Ekran çıktısı:



Grafikteki;

- Mavi çizgi geçmiş fiyat hareketleri
- Sarı çizği son 60 günde gerçekleşmiş olan gerçek kapanış fiyatları
- Yeşil çizgi LSTM modelin tahmin ettiği kapanış fiyatlarıdır.

LSTM modelin tahmininin doğruluğunu grafikteki turuncu ve yeşil çizgilere bakarak (son 60 gün) değerlendirebiliyoruz.

10). LSTM modelin tahmin sonucunu grafik haricinde liste şeklindede gösterdik.

Kod bloğu;

```
# gercek ve ongorulen fiyat gosterme
print(valid)
```

Ekran çıktısı:

Date	Close	Predictions
16.10.2017	106474.5	107032.312500
17.10.2017	106990.9	107841.671875
18.10.2017	106926.4	108546.242188
19.10.2017	108433.5	108986.476562
20.10.2017	108488.7	109628.437500
...	...	...
16.09.2019	102589.7	104202.328125
17.09.2019	101447.4	104502.085938
18.09.2019	101930.1	104266.226562
19.09.2019	100338.8	104011.625000
20.09.2019	100236.7	103348.781250

Close sütunu gerçek kapanış, Predictions sütunu LSTM tarafından tahmin edilen kapanışlardır. Gerçekleşen ve tahmin edilen değerler arasındaki farkı kolaylıkla gözlemleyebilmekteyiz.

LSTM modeli uygulanmamızda RSME sonucu neden hatalı sonuç verdi bunu araştıracağız ve bizim uygulamamız kapanış fiyat bilgilerini LSTM modele girdi sağlamakta, çok değişkenli modelin tasarlanmasında diğer sütun verilerinin modele girdi sağlanması nasıl olacak bunu araştırıp uygulamayı düşünüyoruz.

kaynaklar;

- https://everythingcomputerscience.com/CSBigData.html
- https://medium.com/@randerson112358/stock-price-prediction-using-python-machine-learning-e82a039ac2bb
- https://www.youtube.com/watch?v=QIUxPv5PJOY