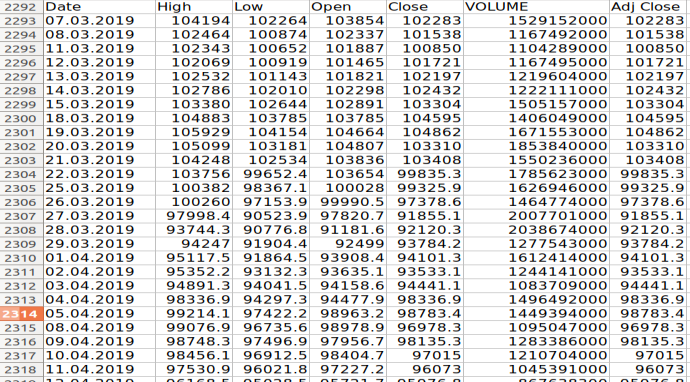
**Proje Adı:** ZAMAN SERİLERİ ANALİZİ İLE BİR SONRAKİ GÜN İÇİN KURDAKİ ARTIŞIN VEYA AZALIŞIN OTOMATİK OLARAK SAPTANMASI VEYA BORSA ANALİZİ

LSTM Modeli Kullanarak Hisse Senedi Fiyat Tahmini ;

1). Uygulamada kullandığımız veriseti görüntüsü



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)

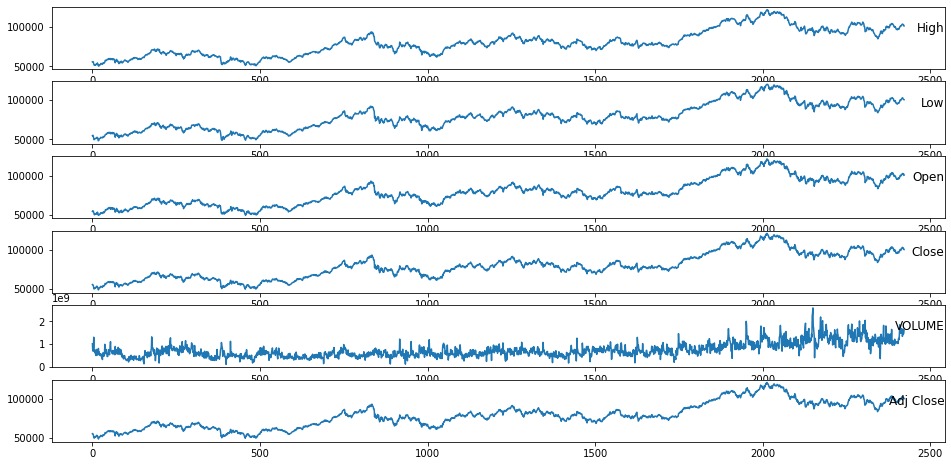
  pyplot.plot(values[:, group])

  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')

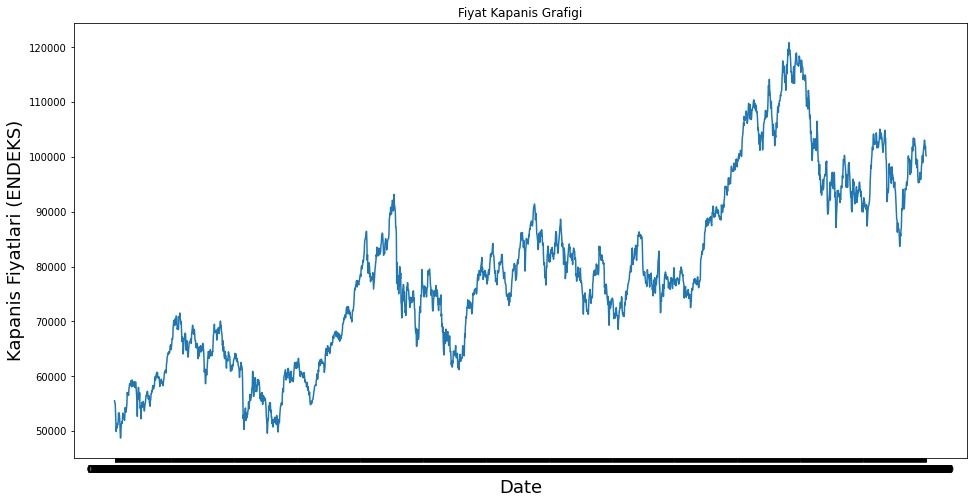
pyplot.plot(dataset['Close'])

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;

 ( 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

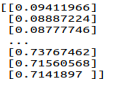
# bu bir hucre

scaler = MinMaxScaler(feature\_range = (0,1))

scaled\_data = scaler.fit\_transform(dataset\_close)

print(scaled\_data)

Ekran Çıktısı;

 ( 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

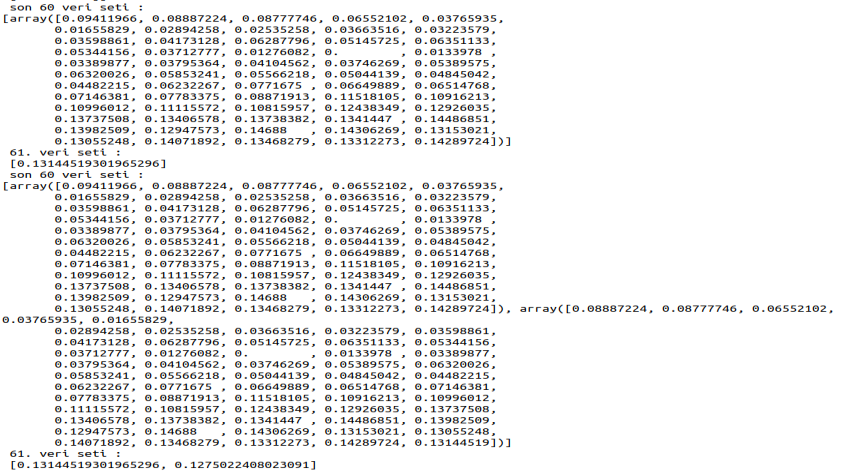
Y\_train.append(train\_data[i, 0]) # konumunda stun alacagiz bu yuzden ilk gecis icin ayarlanan 61. degeri 60. pozisyon simdi ne oldugunu gormek icin

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;



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;













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;



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)

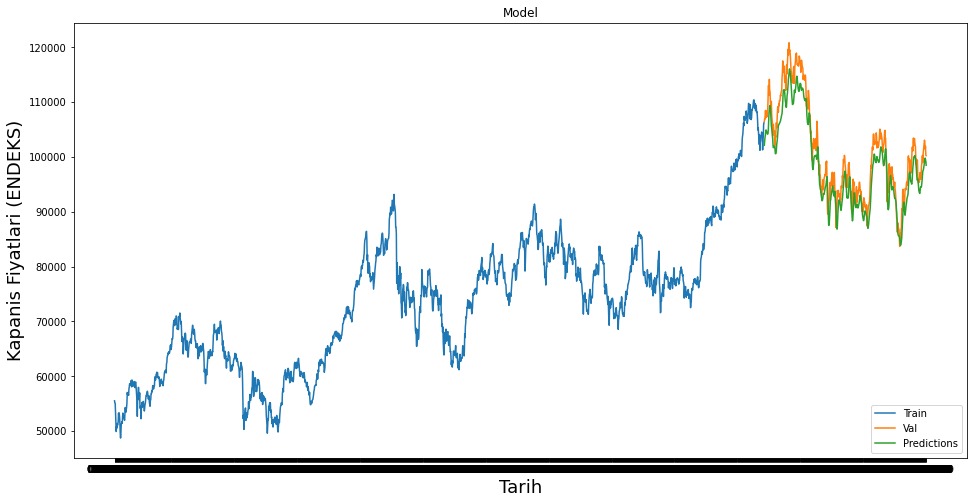
plt.plot(train['Close'])

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.

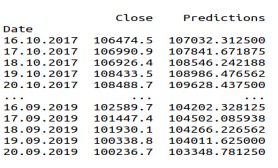
10). LSTM modelin tahmin sonucunu grafik haricinde liste şeklindede gösterdik.

Kod bloğu;

# gercek ve ongorulen fiyat gosterme

print(valid)

Ekran çıktısı:



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>