

بررسی سری زمانی فروشگاه آنلاین

داده ها

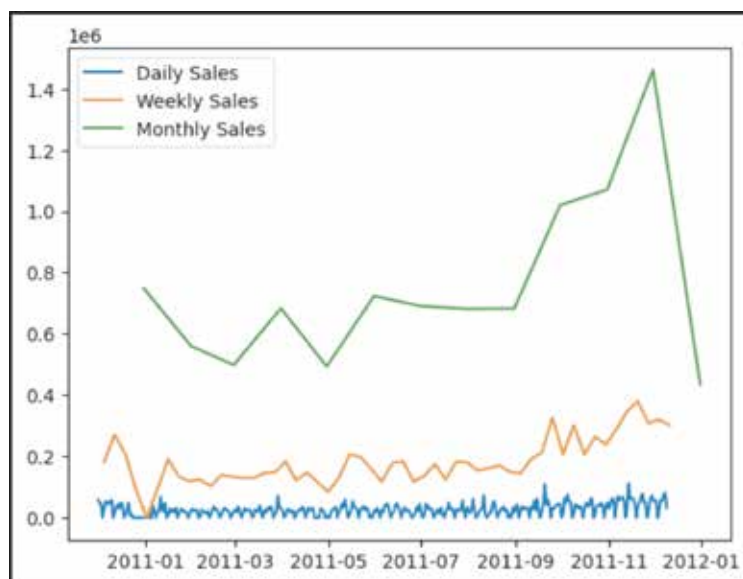
InvoiceDate	InvoiceNo	StockCode	Description	Quantity	UnitPrice	CustomerID	Country
2010-12-01 08:26:00	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2.55	17850.0	United Kingdom
2010-12-01 08:26:00	536365	71053	WHITE METAL LANTERN	6	3.39	17850.0	United Kingdom
2010-12-01 08:26:00	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2.75	17850.0	United Kingdom
2010-12-01 08:26:00	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	3.39	17850.0	United Kingdom
2010-12-01 08:26:00	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	3.39	17850.0	United Kingdom

InvoiceDate	InvoiceNo	StockCode	Description	Quantity	CustomerID	Country	UnitPrice
زمان خرید	شماره خرید	کد محصول	توضیحات	مقدار	مشتری	کشور	قیمت واحد

ساختن یک ستون جدید با ضرب دو متغیر مقدار و قیمت واحد

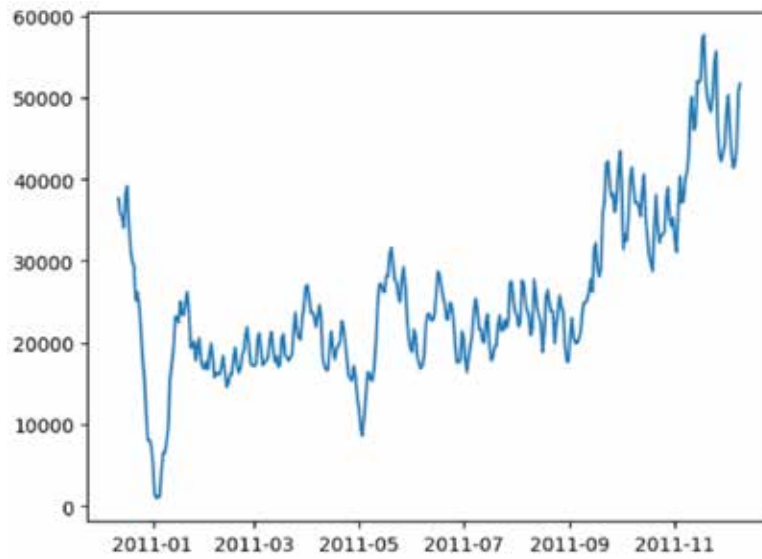
```
df['Price'] = df['Quantity'] * df['UnitPrice']
df.head()
```

در این مدل سعی کردم که داده ها را به شاخص های زمانی متفاوتی تقسیم کنم و با استفاده از آن مدل پیشنهاد بدم، در اولین گام داده های روزانه به نظر noise خیلی زیادی داشتند و تصمیم بر این گرفتم که داده ها را به صورت هفتگی بررسی کنم و چون داده های یک سال را داشتم فقط حدود ۵۲ سطر داشتم و مدل هایی که می ساختم قدرت پیش بینی خوبی نداشت به دلیل کمبود داده. با انجام میانگین متحرک، تونستم که داده های روزانه را بهتر کنم، مدل هایی که پیشنهاد داده شد برای مدل های هفتگی هم در آخر قرار میدم.

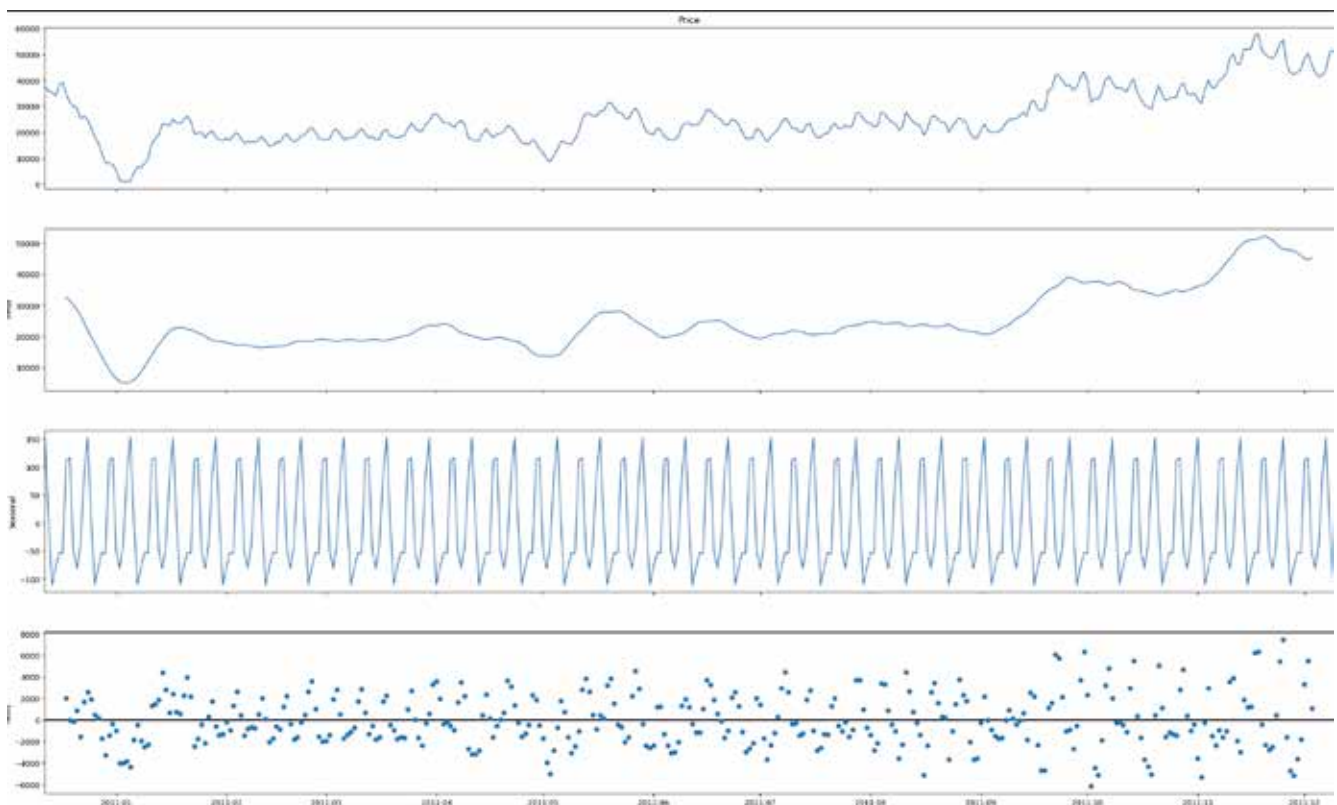
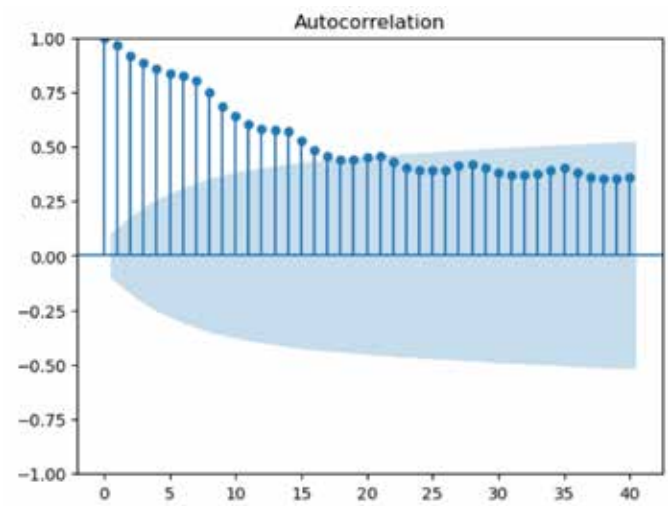
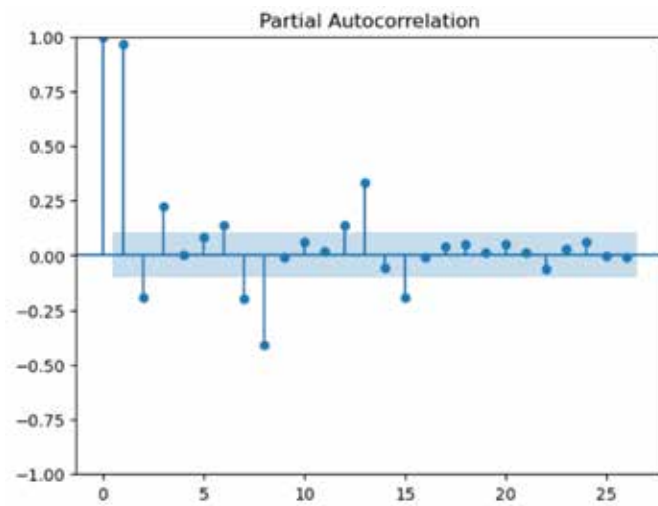


در اینجا نمودارهایی ماهانه (سبز)،
هفتگی (نارنجی) و روزانه (آبی) نشون داده شده

نمودارهای مربوط به داده های روزانه



نمودار سری زمانی



با گرفتن آزمون دیکی-فولر و نموداری که در صفحه قبل بود مطمئن شدیم که داده های ما مانا نیستند، به این جهت برای ثبات واریانس آن لگاریتم میگیریم و برای ثبات میانگین آن، داده ها را تفاضلی میکنیم.

```
result = adfuller(daily_rolling)
print(f'P_value : {result[1]}')
if result[1] > 0.05:
    print('تست نشان میدهد که داده های مانا نیستند')
else:
    print('داده ها مانا هستند')
```

P_value : 0.9102068703433419

تست نشان میدهد که داده های مانا نیستند

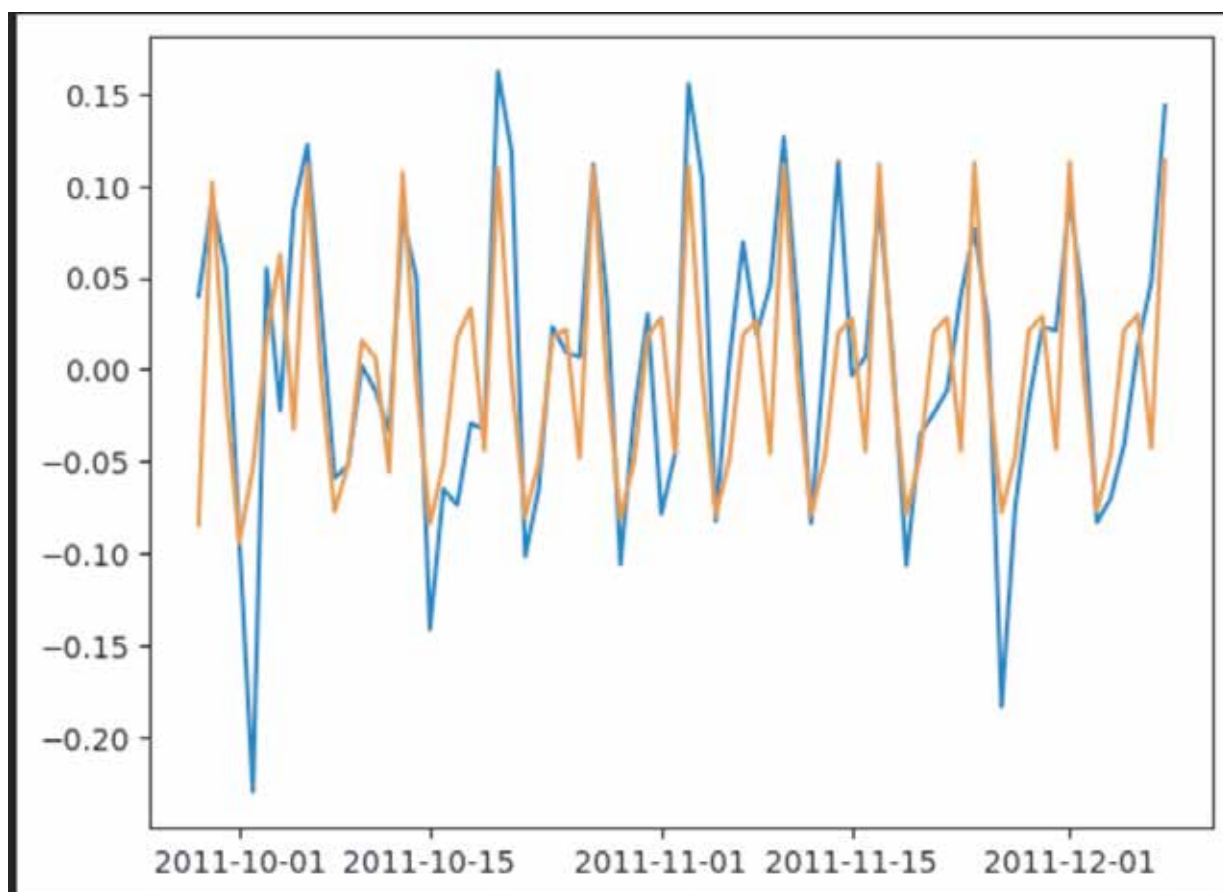


دوباره تست دیکی-فولر گرفته شد داده ها مانا اعلام شدند.

یکی از مدل های آرمای پیشنهاد شده به این نحوه بود
با $order = (2, 0, 1)$ و پارامترهای فصلی $(1, 1, 0, 7)$.

```
model = ARIMA(order=(2,0,1) , seasonal_order=(1,1,0,7)).fit(train_d)

prediction_daily = model.predict(int(len(df_d) * 0.2))
prediction_daily = prediction_daily.dropna()
```



نمودار داده های **تست (اصلی)** و داده های **پیش بینی** شده

```
MAE : 0.04225011394655314
MSE : 0.0031346843858728003
RMSE : 0.05598825221305627
MAPE : 1.50%
```

که امروزه های این مدل به این شکل بود.

```

param_grid = {
    'p': [0, 1, 2],
    'd': [0],
    'q': [0, 1, 2],
    'P': [0, 1],
    'D': [0],
    'Q': [0, 1],
    'm': [7]
}
grid = ParameterGrid(param_grid)

rmse_list = []

for i in grid:
    model_cv = ARIMA(order=(i['p'], i['d'], i['q']), seasonal_order=(i['P'], i['D'], i['Q'], i['m']))
    CV = model_selection.RollingForecastCV(h=4, step=1, initial=int(0.4 * len(train_d)))
    cv_score = model_selection.cross_val_score(model_cv,
                                                y=train_d,
                                                scoring='mean_squared_error',
                                                cv=CV, error_score=1000000)

    rmse = np.sqrt(np.average(cv_score))
    rmse_list.append(rmse)

```

با استفاده از این قطعه کد توانستیم بهترین پارامترهای ممکن که برای مدل ARIMA مناسب هست را انتخاب کنیم عملکرد این کد این هست که یک ترکیبی از پارامترها را در مدل میزازه و ارور اونها را با MSE می سنجه و در یک list قرار میده و کمترین مقدار MSE به ما مدل مورد انتظار را میده

```

best_params = grid[np.argmin(rmse_list)]
best_params

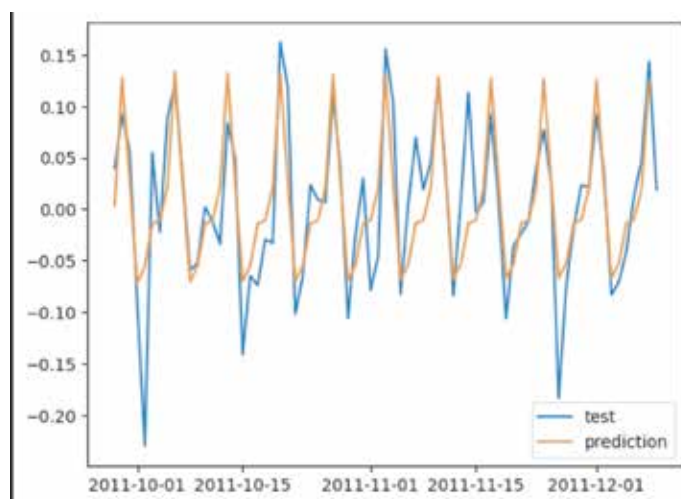
{'q': 0, 'p': 1, 'm': 7, 'd': 0, 'Q': 1, 'P': 1, 'D': 0}

model_tuned_d = ARIMA(order=(1,0,0), seasonal_order=(1,0,1,7)).fit(train_d)

prediction_daily_tuned = model_tuned_d.predict(int(len(df_d) * 0.2))
prediction_daily_tuned = prediction_daily_tuned.dropna()

```

با توجه به قطعه کد قبلی بهترین پارامترها برای مدل کردن داده های ما $p=1$, $d=0$, $q=0$ است و در پارامترهای فصلی $P=1$, $D=0$, $Q=1$ و $m=7$ بود که بعد از مدل کردن و fit کردن روی داده های train توانستیم مدل را بسازیم و بعد از آن روی داده های تست که ۲۰ درصد کل داده ها را تشکیل می داد آن را پیش بینی کردیم



نمودار داده های تست (اصلی) و داده های پیش بینی شده

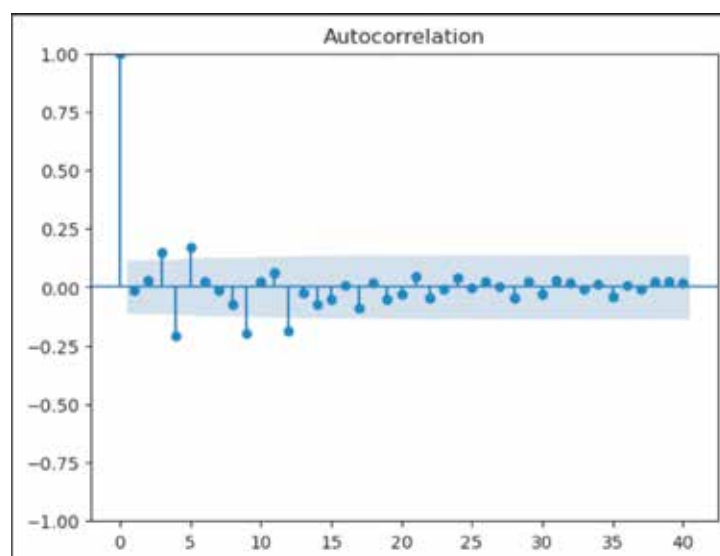
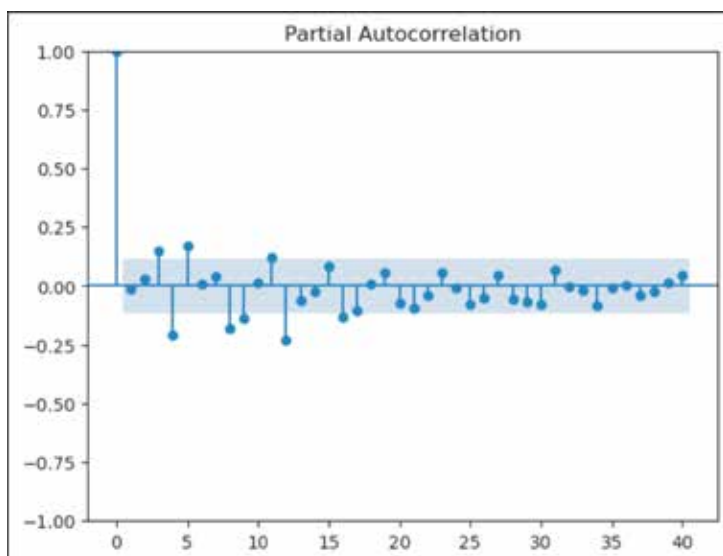
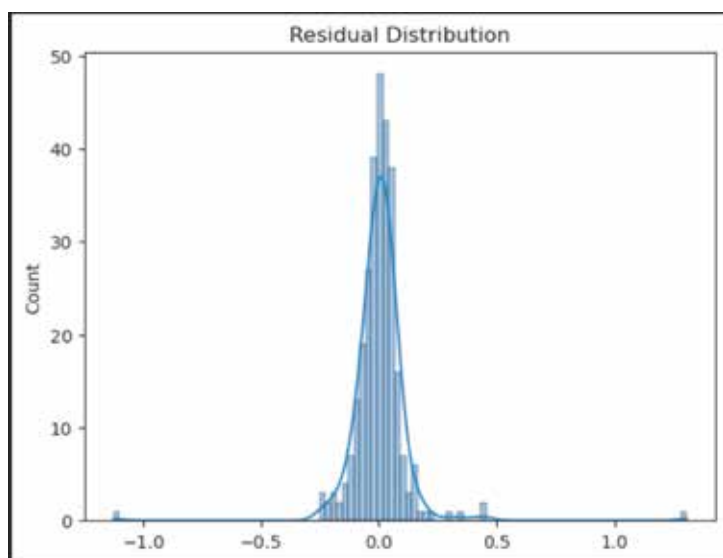
```

MAE : 0.03388731269674192
MSE : 0.0021288387273140507
RMSE : 0.04613934034329111
MAPE : 1.05%

```

که ارور های این مدل نسبت به قبلی پیشرفت کرد.

تحلیل باقی مانده ها



طبق تحلیل باقی مانده مدل، میتوان نتیجه گرفت که هنوز فرصت برای بهتر شدن مدل وجود دارد و مدل به صورت کامل تمام اطلاعات نهفته در داده را استخراج نکرده.