Timeseries Analysis

-final project-

201811530 통계학과 임도현

Content Page

시계열분석 Term Project

201811530 통계학과 임도현

1. Goal of Project

이번 연구의 주제는 주어진 부산시의 기온데이터를 통해 평균기온을 예측하는 모형을 개발하는 것이다. 이로서 부산시의 평균기온이 알고 싶을 때 주어진 모형을 사용하여 예측할 수 있을 것이다.
이때, 고전적인 시계열 분석 모형의 성능과 딥러닝 모형의 성능을 비교하는 것에 초점을 맞추고 분석을 진행한다.

2. Data Description

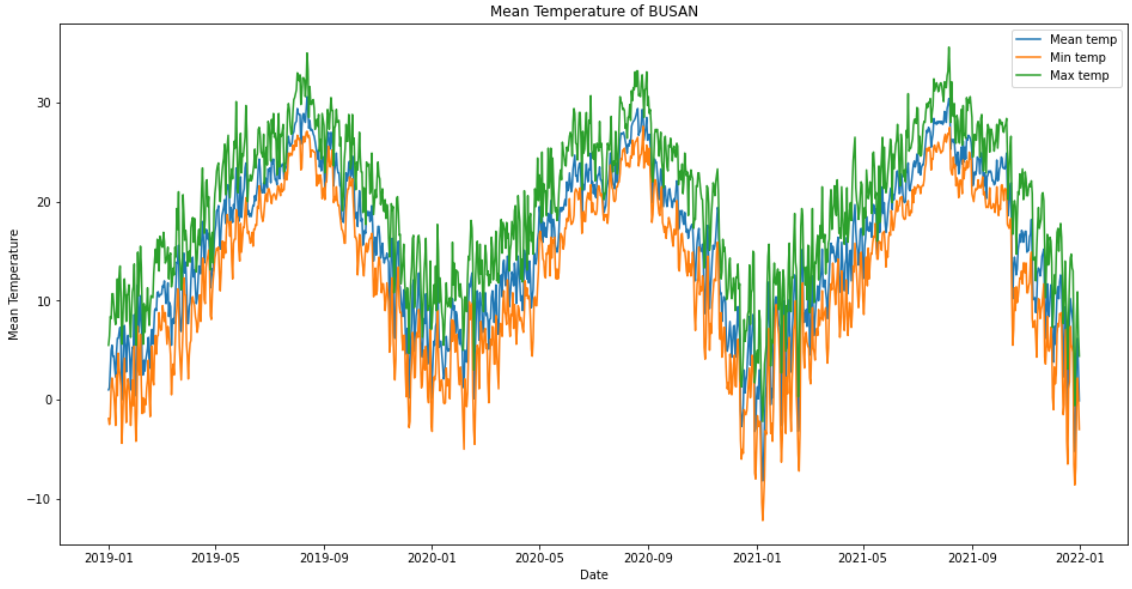
주어진 데이터는 2019년 1월 1일부터 2021년 12월 31일까지 부산광역시의 평균기온(℃), 최저기온(℃), 최고기온(℃)을 담고 있는 데이터이다. 따라서 총 1096개의 행과 4개의 열로 구성되어 있다.

원본 데이터 샘플 예시)

	일시	평균기온(°C)	최저기온(°C)	최고기온(°C)
0	2019-01-01	1.0	-1.9	5.5
1	2019-01-02	1.2	-2.5	6.4
2	2019-01-03	3.0	-1.7	8.4
3	2019-01-04	5.0	1.4	8.2
4	2019-01-05	5.5	2.2	10.7

1096 rows * 4 columns

<날짜에 따른 평균기온, 최저기온, 최고기온의 시각화>



3. Modeling

부산시 평균 기온 예측 모형개발에 사용될 모델들은 다음과 같다.

- ARIMA : 고전적인 '시계열' 예측 모형
- LSTM : 딥러닝을 활용한 '시계열' 예측 모형

위 두 가지 모형을 사용하게 된 이유는 첫째로 고전적인 시계열 예측모형과 딥러닝 모형이 예측결과가 어떤 차이를 보여줄지가 궁금했기 때문이다. 둘째로는 딥러닝으로 시계열 데이터를 예측하는 데에는 보편적으로 LSTM 모형이 자주 사용되어 왔기 때문이다.

위의 모형들을 데이터에 알맞게 훈련시킨 뒤 평가지표를 가지고 가장 최적의 모형을 식별한다.

우선 현재의 데이터를 Train/Valid/Test dataset으로 8:1:1 비율로 분할하자. 데이터의 개수가 적기 때문에 Valid/Test dataset은 적은 비율을 차지하게 된다.

<Train/Valid/Test dataset 분할 결과>

```
Train shape : (877, 4)
Valid shape : (109, 4)
Test shape : (110, 4)
```

평가지표로는 RMSE(Root Mean Squared Error)와 MAPE(Mean Absolute Percentage Error)를 채택한다.

- RMSE(Root Mean Squared Error) : 지표 자체가 직관적이고 단순하다.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

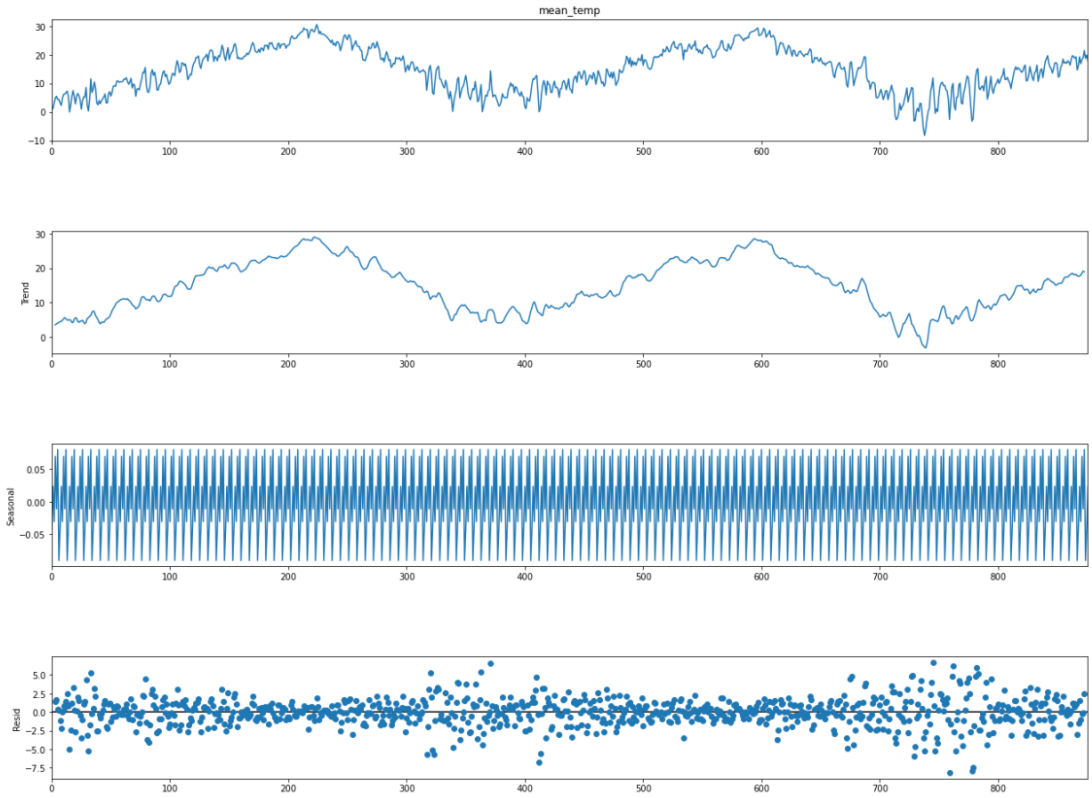
- MAPE(Mean Absolute Percentage Error) : 변수의 스케일의 영향을 받지 않는다.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

RMSE는 단순하지만 스케일의 영향을 받고, MAPE는 변수의 스케일의 영향을 받지 않으므로 상호간에 보완이 가능하다. 따라서 두 지표를 채택한다.

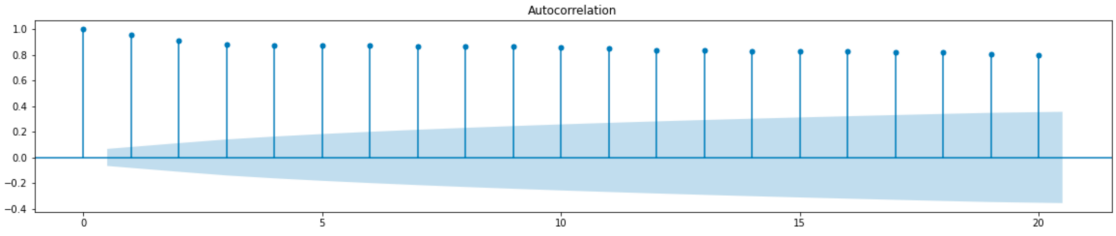
- ARIMA

<계절성분해 시각화>



그래프를 확인해 보았을 때 특정 패턴이 존재한다. 기온 데이터라 어쩌면 직관적인 결과라 할 수 있다. 데이터의 패턴은 정상성에 의심을 야기하므로 이를 판단하기 위해 ACF 그래프를 그려보자.

<ACF 그래프>



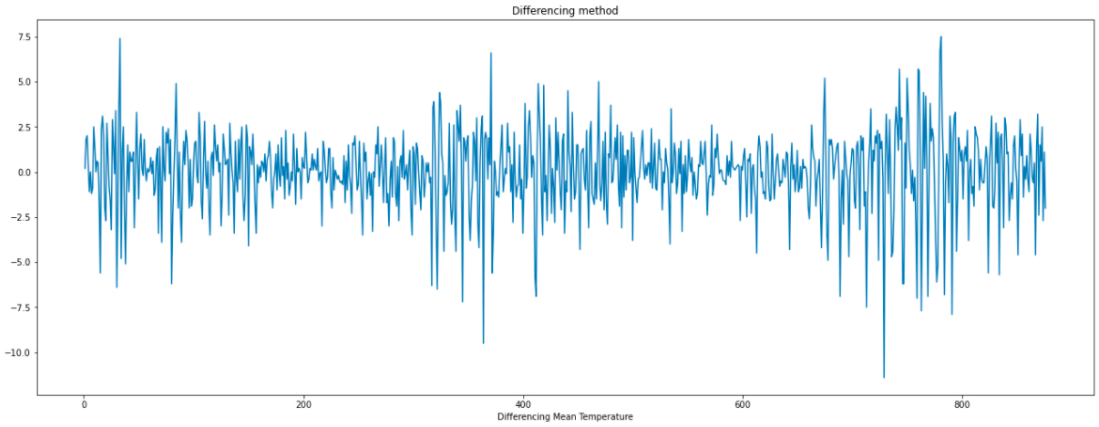
ACF의 값이 아주 천천히 작아지는 것을 알 수 있다. 거의 작아지지 않는다고도 해석할 수 있다. 즉 정상성을 만족하지 않는다.
정량적인 판단 또한 필요하므로 정상성을 판단하는 단위근 검정인 ADF(Augmented Dickey-Fuller) Test를 실시해보자.

<ADF 검정 결과>

```
ADF Statistic: -2.049500458248717
p-value: 0.2652991514719423
Critical Values:
        1%: -3.438
        5%: -2.865
        10%: -2.569
```

p-value가 0.05를 초과하므로 귀무가설인 정상성만족을 기각한다. 즉 해당 데이터는 정상성을 만족하지 않는다. 따라서 1차 차분을 시행해보자.

<1차 차분 결과 시각화>



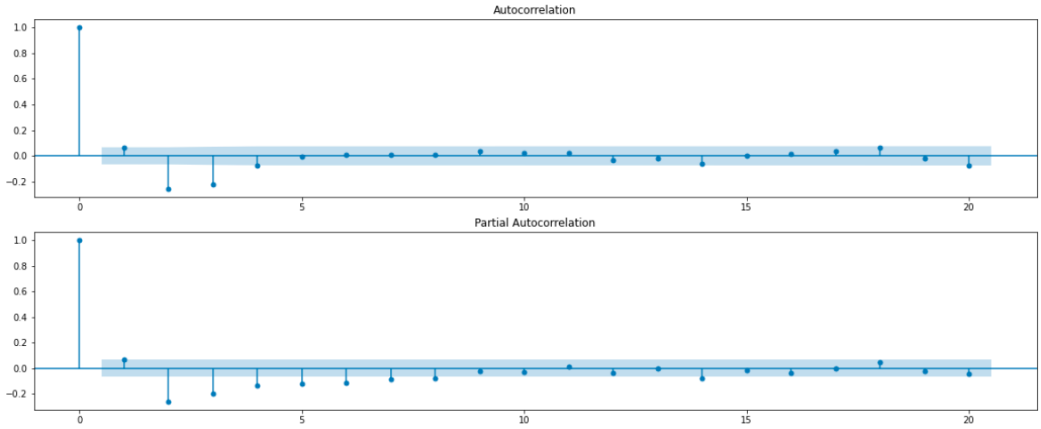
일정한 패턴이 확인되지 않고 나름대로 정상성을 만족하는 듯 보인다. ADF 검정 결과를 확인해보자.

<1차 차분 후 ADF 검정 결과>

```
ADF Statistic: -15.71722298453241
p-value: 1.3372921182191863e-28
Critical Values:
        1%: -3.438
        5%: -2.865
        10%: -2.569
```

p-value가 매우 작으므로 귀무가설을 기각한다. 즉 1차 차분한 데이터는 정상성을 만족한다. 이제 ARIMA 모형의 차수 p와 q를 결정하자.

<1차 차분 후 ACF/PACF 시각화 결과>



ACF와 PACF 모두 금방 0에 수렴하는데, 2~3번째 lag 이후 0에 수렴한다. 따라서 ARIMA(1,1,1), ARIMA(2,1,1), ARIMA(1,1,2), ARIMA(2,1,2)등의 모델을 Validation set에 시도해보자.

<ARIMA모형의 Validset에 대한 성능지표 결과>

	RMSE	MAPE
(1, 1, 1)	6.815	24.478
(2, 1, 1)	6.619	23.568
(1, 1, 2)	6.788	24.323
(2, 1, 2)	6.602	23.489

평가지표에서 차수(2,1,2)가 가장 좋은 성능을 보였으므로 ARIMA(2,1,2)모형을 채택한다.

- LSTM

LSTM 모형에서는 최저기온과 최고기온 데이터를 Feature로 하여 평균기온을 예측해보자.
최저기온과 최고기온의 크기가 서로 다르므로 적절한 훈련을 위해 MinMaxScailing을 통해 0~1 사이의 값으로 스케일링 한다. 또한 딥러닝을 위한 PyTorch Framework 사용을 위해 Numpy 배열을 Tensor 형태로 바꿔준다.

<기존 데이터의 Tensor변환 결과>

```
Training Shape : torch.Size([877, 1, 2]) torch.Size([877, 1])
Validation Shape : torch.Size([109, 1, 2]) torch.Size([109, 1])
Testing Shape : torch.Size([110, 1, 2]) torch.Size([110, 1])
```

또한 LSTM 구조를 정의하는 Class를 구현한 후, Learning rate와 Epoch를 변경해가며 Validation set에 대한 성능지표의 변화를 관찰해보자.

<LSTM 구조>

```
class LSTM1(nn.Module):
  def __init__(self, num_classes, input_size, hidden_size, num_layers, seq_length):
    super(LSTM1, self).__init__()
    self.num_classes = num_classes #number of classes
    self.num_layers = num_layers #number of layers
    self.input_size = input_size #input size
    self.hidden_size = hidden_size #hidden state
    self.seq_length = seq_length #sequence length

    self.lstm = nn.LSTM(input_size=input_size, hidden_size=hidden_size,
                        num_layers=num_layers, batch_first=True) #lstm
    self.fc_1 =  nn.Linear(hidden_size, 128) #fully connected 1
    self.fc = nn.Linear(128, num_classes) #fully connected last layer

    self.relu = nn.ReLU()

  def forward(self,x):
    h_0 = Variable(torch.zeros(self.num_layers, x.size(0), self.hidden_size)).to(device) #hidden state
    c_0 = Variable(torch.zeros(self.num_layers, x.size(0), self.hidden_size)).to(device) #internal state
    # Propagate input through LSTM

    output, (hn, cn) = self.lstm(x, (h_0, c_0)) #lstm with input, hidden, and internal state

    hn = hn.view(-1, self.hidden_size) #reshaping the data for Dense layer next
    out = self.relu(hn)
    out = self.fc_1(out) #first Dense
    out = self.relu(out) #relu
    out = self.fc(out) #Final Output

    return out
```

<Parameter(learning rate/epoch)에 따른 LSTM의 Validaset에 대한 성능지표 결과>

	RMSE	MAPE
LSTM(0.01 / 1000)	8.187	31.262
LSTM(0.01 / 1000)	8.187	31.262
LSTM(0.001 / 1000)	3.489	13.121
LSTM(0.0001 / 1000)	0.392	1.333
LSTM(0.01 / 2000)	0.375	1.294
LSTM(0.001 / 2000)	0.370	1.274
LSTM(0.0001 / 2000)	0.364	1.257
LSTM(0.01 / 5000)	0.349	1.201
LSTM(0.001 / 5000)	0.346	1.183
LSTM(0.0001 / 5000)	0.344	1.177
LSTM(0.01 / 10000)	0.345	1.177
LSTM(0.001 / 10000)	0.345	1.175
LSTM(0.0001 / 10000)	0.343	1.167

Learning rate가 낮을수록, Epoch이 많이 진행될수록 성능이 좋아지는 모습을 알 수 있다. 특히 learning rate가 크고(0.01) Epoch가 적은(1000) 모델은 ARIMA(2,1,2)모형보다 성능이 좋지 않은 것을 알 수 있다.

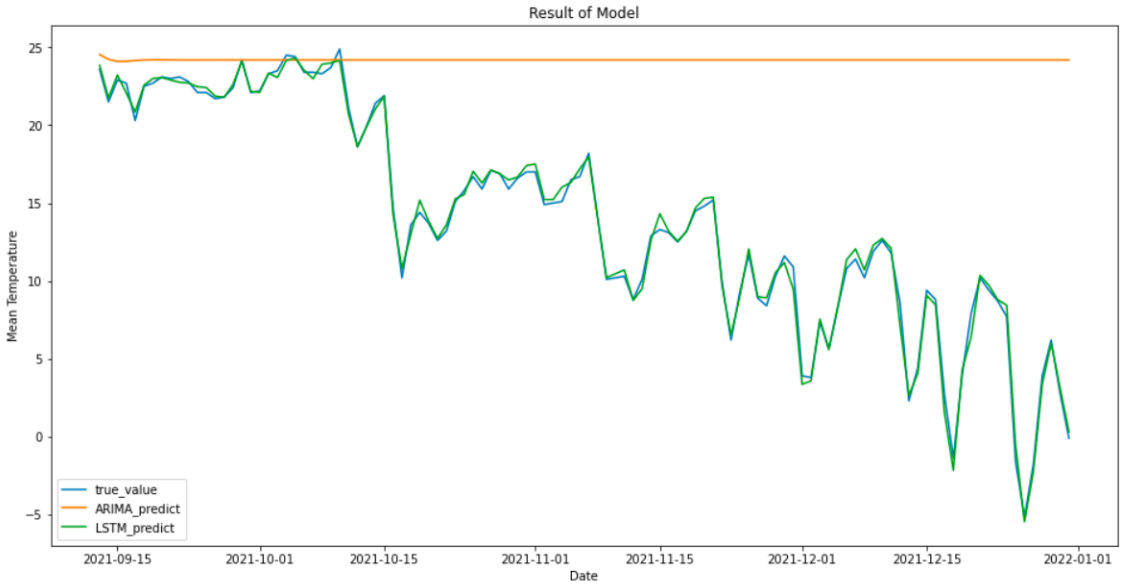
가장 성능이 좋은 모델은 learning_rate가 0.0001이고 epoch가 10000인 LSTM 모형이다. 따라서 이 모델을 채택하기로 한다.
이제 TestDatset에 우리가 채택한 모델 두 가지를 적용하고 성능을 비교해보자.

<ARIMA(2,1,2)모형과 LSTM(lr=0.0001, epoch=10000)모형의 Testset에 대한 성능지표결과>

	RMSE	MAPE
ARIMA(2,1,2)	12.399	385.436
LSTM(lr=0.0001, epoch=10000)	0.461	8.363

성능의 차이가 꽤나 심하게 차이가 난다. 실제 기온예측 값들을 시각화해보자.

<모형들의 평균기온 예측값 시각화>



4. Conclusion

ARIMA 모형의 경우 기온 예측의 값이 'stuck' 상태이다. 일정 값 이상에서 전혀 변화가 일어나지 않는 모습을 보여주고 있으며, 반면에 LSTM 모형의 경우 평균기온 예측이 매우 잘 된 상황이다.

따라서 우리는 부산시의 평균기온 예측 모형을 LSTM 모형으로 선정해야 함을 알 수 있다. 이는 시각화 뿐만 아니라 기존의 성능지표의 결과로도 합리적인 선택이다.

5. Retrospect

LSTM 모형의 경우에는 최저기온과 최고기온을 지표로 사용하여 학습을 진행하였다. 때문에 사실 학습이 잘되는 것이 당연한 사실이었고, ARIMA와 결과 비교를 하는 것이 불합리적이라는 생각이 들었다.

또한 ARIMA모형에 대한 깊은 이해가 조금 더 필요하겠다는 생각이 들었으며, 왜 'ARIMA모형의 예측값'들이 섭씨 24도 언저리에서 계속 머물러있는지에 대한 원인을 알아볼 필요가 있다.

분석을 진행하며 개략적인 시계열데이터 예측과정에 대한 프로세스를 배울 수 있어 보람찼고 부족한 부분을 잘 알아차릴 수 있어 보람찬 프로젝트였다.