DF 겨울시즌 1조

지하철역 혼잡도 예측

목 차

주제

문제: 기존 경로 추천 앱들은 지하철 혼잡도를 반영하지 못함

해결방안1:서울시 지하철 혼잡도를 ARIMA로 분석, 예측

해결방안2: 예측된 혼잡도를 바탕으로 최적의 역 추천

ARIMA 모델

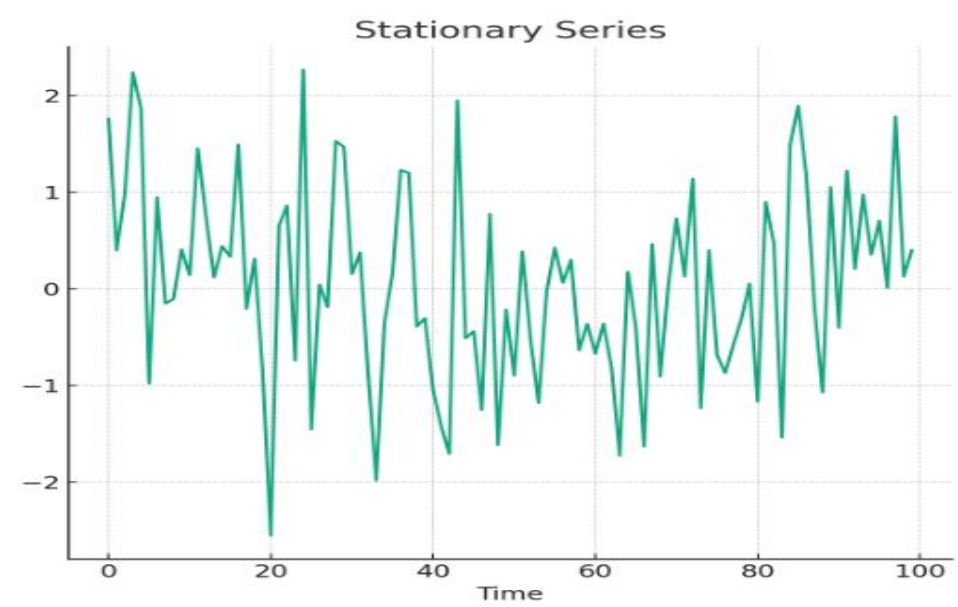
ARIMA 모델은 시계열 분석에 널리 사용되는 모델 (ARIMA(P,D,Q)로 표현됨)

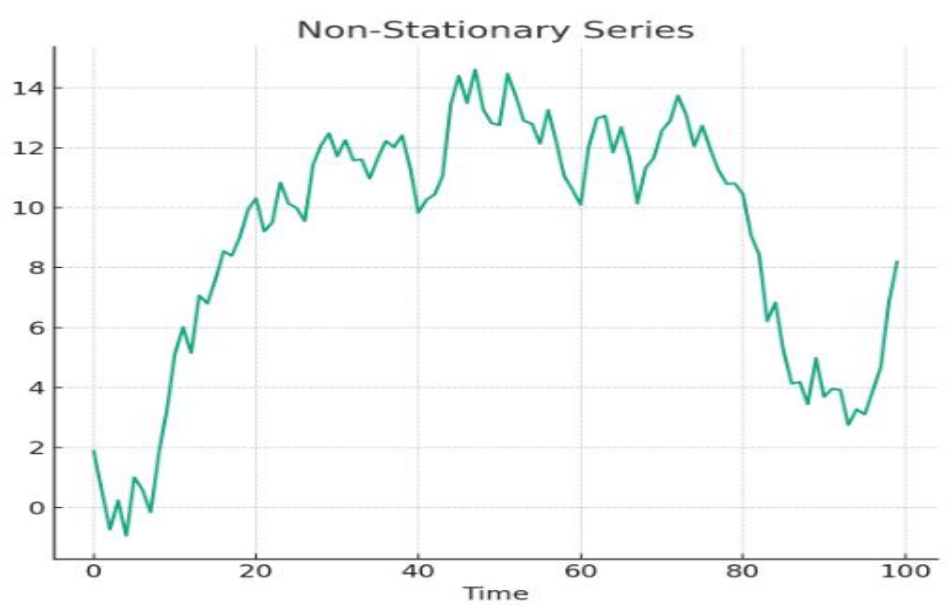
(자기회귀)AR(P), (차분)I(D), (이동 평균)MA(Q) 세가지 요소로 구성됨

자기회귀(AR) :과거 값들의 선형 조합을 사용하여 현재 값이 이전의 값들에 어떻게 의존하는지를 모델링함. 여기서 'p'는 과거 관측값의 수를 나타냄

차분(I) 비정상 시계열 데이터를 정상 시계열 데이터로 변환하는 과정. 데이터에서 일정한 추세를 제거하기 위해 사용되며, 'd'는 차분의 차수를 나타냄

이동평균(MA): 과거 예측 오차들의 선형 조합을 사용하여 현재 값을 예측함. 여기서 'q'는 과거 예측 오차의 수를 나타냄.





ARIMA를 활용해 시계열 데이터의 정상성(stationary)를 확보함

정상성(stationary)이란 시간에 따라 시계열 데이터의 평균이나 분산이 일정함을 의미함

정상 시계열 데이터는 예측 모델링에 있어 예측 가능한 패턴을 더 쉽게 식별 하도록 함

데이터 전처리

```
data=pd.read_csv('/content/drive/MyDrive/서울교통공사_역별 일별 시간대별 승하차인원 정보_20231031.csv',encoding='cp949')
data.head()
 #'수송일자'를 datetime 타입으로 변환하고 '요일' 컬럼 추가
```

	연번	수송일자	호선	역번호	역명	승하차구분	06시 이전	06-07 시간대	07-08 시간대	08-09 시간대	...	15-16 시간대	16-17 시간대	17-18 시간대	18-19 시간대	19-20 시간대	20-21 시간대	21-22 시간대	22-23 시간대	23-24 시간대	24시 이후
0	1	2023-01-01	1호선	150	서울역	승차	215	145	231	594	...	2655	2509	2696	2549	2462	2177	2190	1808	734	7
1	2	2023-01-01	1호선	150	서울역	하차	154	636	595	939	...	2282	2295	2526	1930	1897	1487	991	609	280	46
2	3	2023-01-01	1호선	151	시청	승차	48	73	106	194	...	843	895	959	985	670	630	515	330	146	0
3	4	2023-01-01	1호선	151	시청	하차	64	247	293	463	...	602	575	533	456	285	267	246	154	79	18
4	5	2023-01-01	1호선	152	종각	승차	407	235	158	201	...	1145	1402	1223	1272	911	913	906	602	232	3

5 rows × 26 columns

사용 데이터: 서울교통공사 역별 일별 시간대별 승하차 인원정보
기간:2023년 1월 1일~ 2023년 10월 31일

```python
# '수송일자'를 datetime으로 변환하고 '요일' 컬럼을 추가합니다.
data['수송일자'] = pd.to_datetime(data['수송일자'], format='%Y-%m-%d')
data['요일'] = data['수송일자'].dt.day_name()

# '사당'역 데이터 필터링
sadang_data = data[data['역명'] == '사당']

# 승차와 하차 인원 합계 계산
# 승차 인원 합계
sadang_on_total = sadang_data[sadang_data['승하차구분'] == '승차'].groupby('수송일자').sum().iloc[:, 2:24].sum(axis=1)
# 하차 인원 합계
sadang_off_total = sadang_data[sadang_data['승하차구분'] == '하차'].groupby('수송일자').sum().iloc[:, 2:24].sum(axis=1)

# 혼잡도 계산 (승차 인원 - 하차 인원)
congestion_sadang = sadang_on_total - sadang_off_total

# 혼잡도를 DataFrame으로 변환
congestion_sadang_df = congestion_sadang.reset_index()
congestion_sadang_df.columns = ['수송일자', '혼잡도']
```

전처리1: 분석하고자 하는 역 선정 (ex:사당)

전처리2: 일별 혼잡도(승차-하차)를 계산

congestion_sadang_2023

	수송일자	혼잡도
0	2023-01-01	-434
1	2023-01-02	97
2	2023-01-03	1546
3	2023-01-04	912
4	2023-01-05	138
...	...	...
299	2023-10-27	-2550
300	2023-10-28	-4432
301	2023-10-29	-1134
302	2023-10-30	-749
303	2023-10-31	390

304 rows × 2 columns

7호선 내방역 혼잡도 예측 과정

```
# auto_arima를 사용하여 내방역 데이터에 대한 최적의 모델을 자동으로 찾습니다.
auto_model_naebang = auto_arima(congestion_naebang_2023['혼잡도'], seasonal=True, m=7, trace=True,
                                error_action='ignore', suppress_warnings=True)

# 10월의 혼잡도 예측
forecast_naebang = auto_model_naebang.predict(n_periods=31)
```

pmdarima 의 autoarima 이용

n_periods=31 : 예측하고자 하는 기간 길이 지정

congestion_naebang_2023 : 내방역 혼잡도 값 중 2023년값만 필터링

m=7 : 계절성 패턴이 반복되는 주기 지정. 여기는 일주일 주기 반복 가정

7호선 내방역 혼잡도 예측 과정

```
# 시각화
plt.figure(figsize=(14, 7))
plt.plot(congestion_naebang_2023['수송일자'], congestion_naebang_2023['혼잡도'], label='Actual Congestion - Naebang', color='blue')
plt.plot(forecast_series_naebang.index, forecast_series_naebang, label='Forecasted Congestion - Naebang', color='red', linestyle='--')
plt.title('Congestion Forecast Naebang')
plt.xlabel('Date')
plt.ylabel('Congestion')
plt.legend()
plt.xticks(rotation=45)
plt.show()
```

시각화 도구로 pyplot 사용

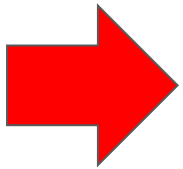
x 축을 날짜, y축을 혼잡도로 정의

실제 데이터를 blue로 지정

예측 데이터를 red로 지정

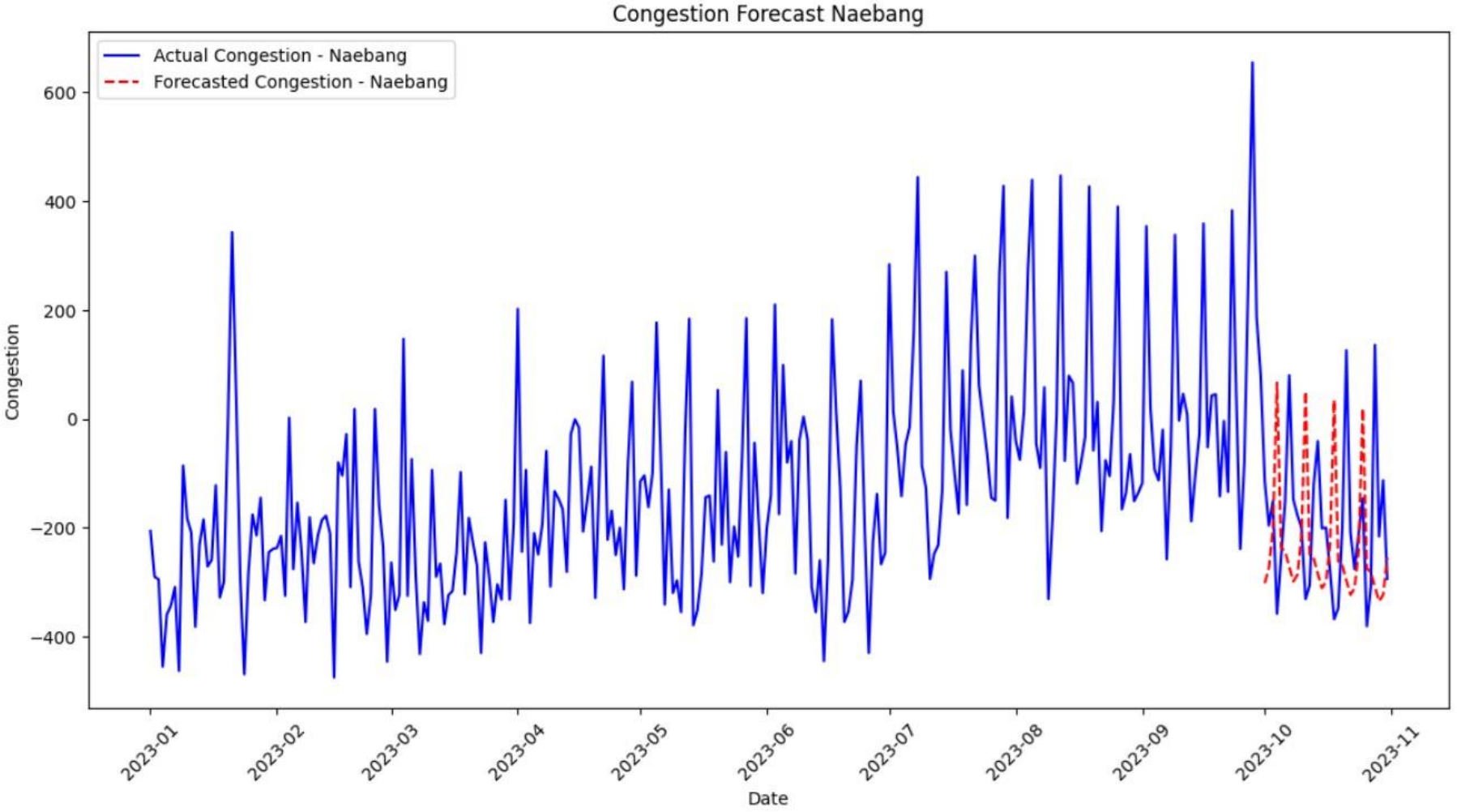
7호선 내방역 혼잡도 예측 과정

```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(1,0,1)[7] intercept   : AIC=inf, Time=1.71 sec
 ARIMA(0,1,0)(0,0,0)[7] intercept   : AIC=4122.839, Time=0.05 sec
 ARIMA(1,1,0)(1,0,0)[7] intercept   : AIC=3988.579, Time=0.73 sec
 ARIMA(0,1,1)(0,0,1)[7] intercept   : AIC=3953.018, Time=0.84 sec
 ARIMA(0,1,0)(0,0,0)[7]             : AIC=4120.840, Time=0.05 sec
 ARIMA(0,1,1)(0,0,0)[7] intercept   : AIC=3997.035, Time=0.29 sec
 ARIMA(0,1,1)(1,0,1)[7] intercept   : AIC=inf, Time=2.55 sec
 ARIMA(0,1,1)(0,0,2)[7] intercept   : AIC=3921.672, Time=1.23 sec
 ARIMA(0,1,1)(1,0,2)[7] intercept   : AIC=inf, Time=2.70 sec
 ARIMA(0,1,0)(0,0,2)[7] intercept   : AIC=4032.063, Time=0.45 sec
 ARIMA(1,1,1)(0,0,2)[7] intercept   : AIC=inf, Time=1.27 sec
 ARIMA(0,1,2)(0,0,2)[7] intercept   : AIC=3904.356, Time=1.65 sec
 ARIMA(0,1,2)(0,0,1)[7] intercept   : AIC=3937.838, Time=0.77 sec
 ARIMA(0,1,2)(1,0,2)[7] intercept   : AIC=inf, Time=3.35 sec
 ARIMA(0,1,2)(1,0,1)[7] intercept   : AIC=inf, Time=3.19 sec
 ARIMA(1,1,2)(0,0,2)[7] intercept   : AIC=inf, Time=1.88 sec
 ARIMA(0,1,3)(0,0,2)[7] intercept   : AIC=inf, Time=1.71 sec
 ARIMA(1,1,3)(0,0,2)[7] intercept   : AIC=3907.131, Time=2.83 sec
 ARIMA(0,1,2)(0,0,2)[7]             : AIC=3902.858, Time=1.05 sec
 ARIMA(0,1,2)(0,0,1)[7]             : AIC=3936.141, Time=0.45 sec
 ARIMA(0,1,2)(1,0,2)[7]             : AIC=3828.434, Time=1.88 sec
 ARIMA(0,1,2)(1,0,1)[7]             : AIC=3826.532, Time=1.50 sec
 ARIMA(0,1,2)(1,0,0)[7]             : AIC=3895.464, Time=1.39 sec
 ARIMA(0,1,2)(2,0,1)[7]             : AIC=inf, Time=1.87 sec
 ARIMA(0,1,2)(0,0,0)[7]             : AIC=3986.257, Time=0.16 sec
 ARIMA(0,1,2)(2,0,0)[7]             : AIC=inf, Time=0.77 sec
 ARIMA(0,1,2)(2,0,2)[7]             : AIC=inf, Time=2.02 sec
 ARIMA(0,1,1)(1,0,1)[7]             : AIC=3839.783, Time=0.80 sec
 ARIMA(1,1,2)(1,0,1)[7]             : AIC=3831.503, Time=1.77 sec
 ARIMA(0,1,3)(1,0,1)[7]             : AIC=3828.016, Time=1.61 sec
 ARIMA(1,1,1)(1,0,1)[7]             : AIC=3826.825, Time=1.13 sec
 ARIMA(1,1,3)(1,0,1)[7]             : AIC=inf, Time=1.10 sec
```



```
Best model:  ARIMA(0,1,2)(1,0,1)[7]
Total fit time: 44.841 seconds
```

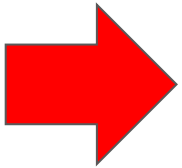
AIC 가 3826.532 로 최소


Congestion Forecast Naebang
Actual Congestion - Naebang
Forecasted Congestion - Naebang
Congestion
600
400
200
0
−200
−400
2023-01
2023-02
2023-03
2023-04
2023-05
2023-06
2023-07
2023-08
2023-09
2023-10
2023-11
Date

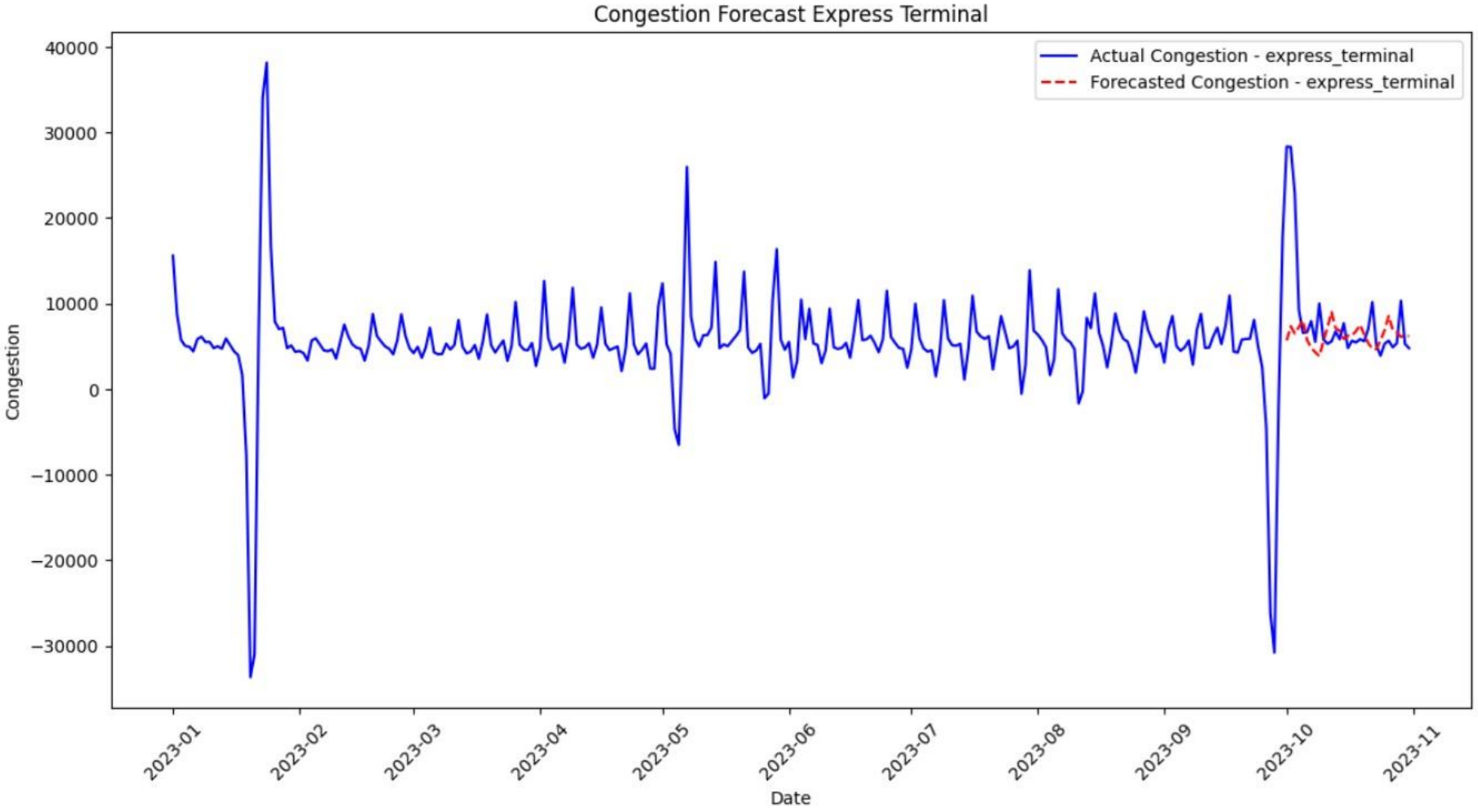
7호선 고속 터미널 역 혼잡도 예측

```
# auto_arima를 사용하여 고속 터미널역 데이터에 대한 최적의 모델을 자동으로 찾습니다.
auto_model_express_terminal = auto_arima(congestion_express_terminal_2023['혼잡도'], seasonal=True, m=7, trace=True,
                                         error_action='ignore', suppress_warnings=True)
```

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(1,0,1)[7] intercept   : AIC=5934.496, Time=1.20 sec
 ARIMA(0,0,0)(0,0,0)[7] intercept   : AIC=6169.937, Time=0.03 sec
 ARIMA(1,0,0)(1,0,0)[7] intercept   : AIC=6029.261, Time=0.40 sec
 ARIMA(0,0,1)(0,0,1)[7] intercept   : AIC=6001.449, Time=0.34 sec
 ARIMA(0,0,0)(0,0,0)[7]             : AIC=6353.424, Time=0.04 sec
 ARIMA(2,0,2)(0,0,1)[7] intercept   : AIC=5943.784, Time=0.82 sec
 ARIMA(2,0,2)(1,0,0)[7] intercept   : AIC=5938.309, Time=0.87 sec
 ARIMA(2,0,2)(2,0,1)[7] intercept   : AIC=5926.722, Time=1.01 sec
 ARIMA(2,0,2)(2,0,0)[7] intercept   : AIC=5924.722, Time=0.93 sec
 ARIMA(1,0,2)(2,0,0)[7] intercept   : AIC=5927.657, Time=0.66 sec
 ARIMA(2,0,1)(2,0,0)[7] intercept   : AIC=5930.408, Time=0.91 sec
 ARIMA(3,0,2)(2,0,0)[7] intercept   : AIC=5921.076, Time=2.02 sec
 ARIMA(3,0,2)(1,0,0)[7] intercept   : AIC=5936.142, Time=1.13 sec
 ARIMA(3,0,2)(2,0,1)[7] intercept   : AIC=5922.903, Time=2.49 sec
 ARIMA(3,0,2)(1,0,1)[7] intercept   : AIC=5931.695, Time=1.09 sec
 ARIMA(3,0,1)(2,0,0)[7] intercept   : AIC=5920.563, Time=2.34 sec
 ARIMA(3,0,1)(1,0,0)[7] intercept   : AIC=5937.910, Time=0.92 sec
 ARIMA(3,0,1)(2,0,1)[7] intercept   : AIC=inf, Time=3.86 sec
 ARIMA(3,0,1)(1,0,1)[7] intercept   : AIC=5929.739, Time=1.00 sec
 ARIMA(3,0,0)(2,0,0)[7] intercept   : AIC=5934.816, Time=1.06 sec
 ARIMA(4,0,1)(2,0,0)[7] intercept   : AIC=5921.894, Time=2.35 sec
 ARIMA(2,0,0)(2,0,0)[7] intercept   : AIC=5933.014, Time=0.69 sec
 ARIMA(4,0,0)(2,0,0)[7] intercept   : AIC=5934.164, Time=1.33 sec
 ARIMA(4,0,2)(2,0,0)[7] intercept   : AIC=5914.829, Time=3.55 sec
 ARIMA(4,0,2)(1,0,0)[7] intercept   : AIC=5928.516, Time=4.33 sec
 ARIMA(4,0,2)(2,0,1)[7] intercept   : AIC=5909.752, Time=2.49 sec
 ARIMA(4,0,2)(1,0,1)[7] intercept   : AIC=5923.442, Time=1.14 sec
 ARIMA(4,0,2)(2,0,2)[7] intercept   : AIC=5907.134, Time=3.85 sec
 ARIMA(4,0,2)(1,0,2)[7] intercept   : AIC=5913.207, Time=4.63 sec
 ARIMA(3,0,2)(2,0,2)[7] intercept   : AIC=5918.568, Time=3.22 sec
 ARIMA(4,0,1)(2,0,2)[7] intercept   : AIC=5920.426, Time=3.28 sec
 ARIMA(5,0,2)(2,0,2)[7] intercept   : AIC=5907.735, Time=5.91 sec
```



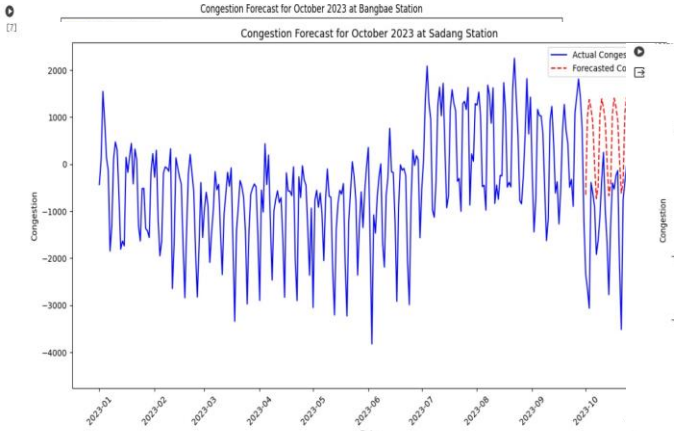
```
Best model:  ARIMA(5,0,3)(2,0,2)[7] intercept
Total fit time: 113.900 seconds
```

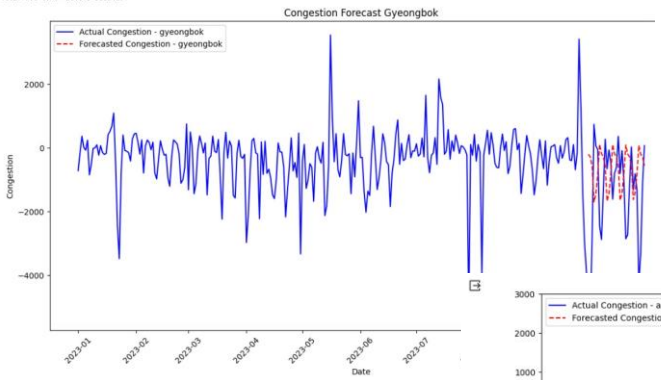

Congestion Forecast Express Terminal
Actual Congestion - express_terminal
Forecasted Congestion - express_terminal
Congestion
Date
40000
30000
20000
10000
0
−10000
−20000
−30000
2023-01
2023-02
2023-03
2023-04
2023-05
2023-06
2023-07
2023-08
2023-09
2023-10
2023-11

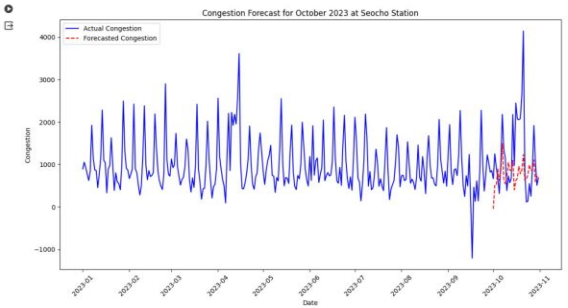
7호선 혼잡도 예측 함수
- compare_stations_and_choose7() -

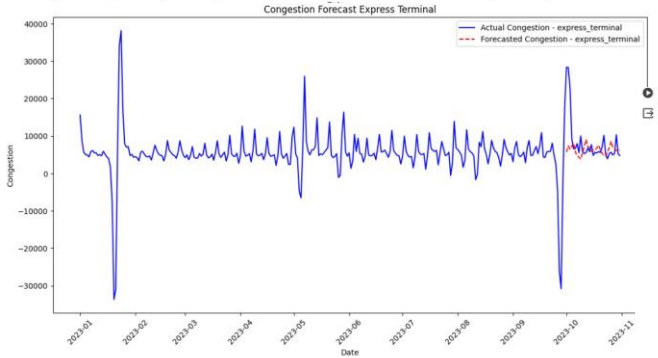
```python
# 역 선택을 위한 함수 정의
def compare_stations_and_choose7(date):
    """
    주어진 날짜에 대해 두 역의 혼잡도를 비교하고, 더 낮은 혼잡도를 가진 역을 추천합니다.

    Parameters:
    - date: 조회하고자 하는 날짜 (예: '2023-10-15')
    """
    # 날짜에 해당하는 혼잡도 조회
    congestion_express = forecast_df_express_terminal[forecast_df_express_terminal['수송일자'] == date]['예측 혼잡도'].values[0]
    congestion_naebang = forecast_df_naebang[forecast_df_naebang['수송일자'] == date]['예측 혼잡도'].values[0]

    print(f"{date} 의 고속터미널역의 혼잡도 : {congestion_express} ")
    print(f"{date}의 내방역의 혼잡도 : {congestion_naebang} ")

    # 두 역의 혼잡도 비교
    if congestion_express < congestion_naebang:
        print(f"{date}에는 고속터미널역이 내방역보다 혼잡도가 낮습니다. 고속터미널역을 추천합니다.")
    elif congestion_express > congestion_naebang:
        print(f"{date}에는 내방역이 고속터미널역보다 혼잡도가 낮습니다. 내방역을 추천합니다.")
    else:
        print(f"{date}에는 두 역의 혼잡도가 같습니다. 어느 역을 선택해도 좋습니다.")
```

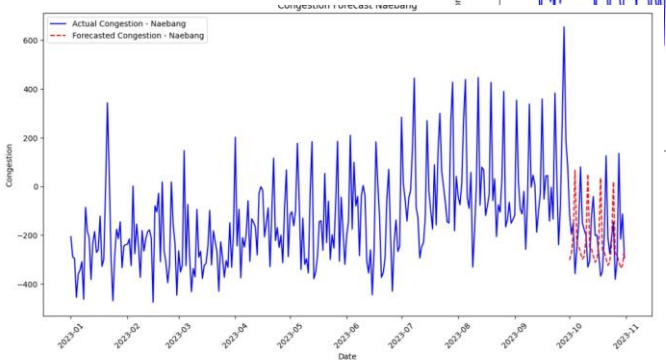
2호선, 3호선 등 주요 역들 혼잡도 예측

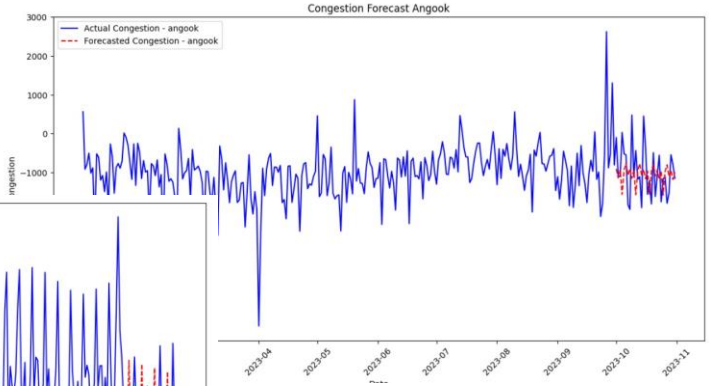












2호선, 3호선 등 주요 역들 혼잡도 예측

```
# 예시 사용
compare_stations_and_choose3('2023-10-13')
```

```
2023-10-13 의 종로3가역의 혼잡도 : 1977.52920129517
2023-10-13의 안국역의 혼잡도 : -771.0033369298524
2023-10-13의 경복궁역의 혼잡도 : -823.4808935561306
2023-10-13에는 경복궁역이(가) 다른 역보다 혼잡도가 낮습니다. 경복궁역을(를) 추천합니다.
```

```
# 예시 사용
compare_stations_and_choose2('2023-10-01')
```

```
2023-10-01 의 서초역의 혼잡도 : -47.30313391678442
2023-10-01 의 방배역의 혼잡도 : -393.77283187170747
2023-10-01 의 사당역의 혼잡도 : -650.8550459790742
2023-10-01에는 사당역이(가) 다른 역보다 혼잡도가 낮습니다. 사당역을(를) 추천합니다.
```

결론

관광지, 핫플레이스, 회사 밀집지역 , 환승역 등 각 장소를 대표 하는 지하철역들 존재

해당 역 혼잡도 == 해당 장소 이용 유동인구

미래 특정 날짜 해당 장소 혼잡도 예측 가능

선제적으로 공적 행사나, 사적 약속이나 계획 융통성 있게 조정 가능