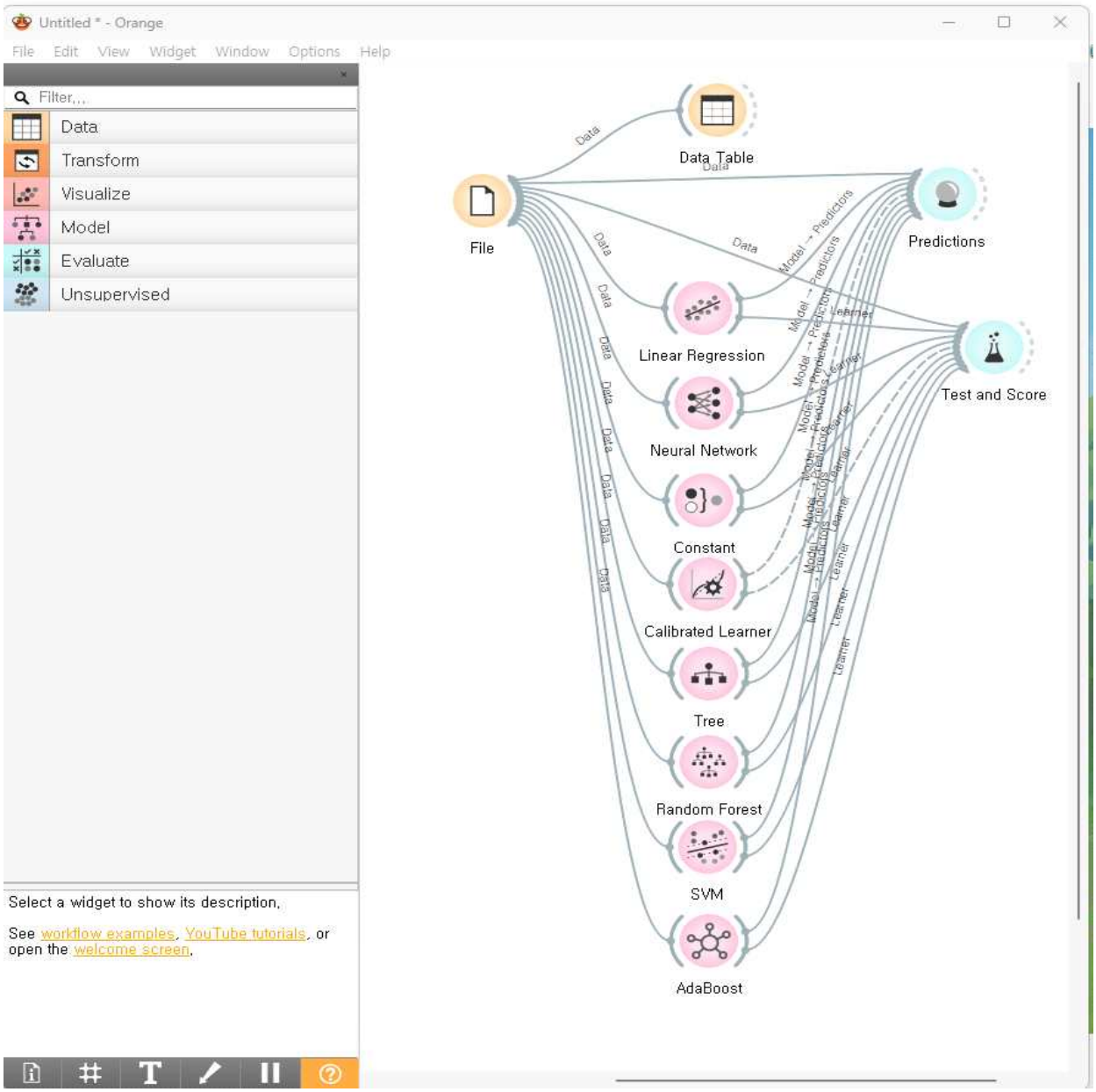
Orange3를 활용한 인공지능 기반 문제해결 실습 평가

22221617 컴퓨터공학과 권체은

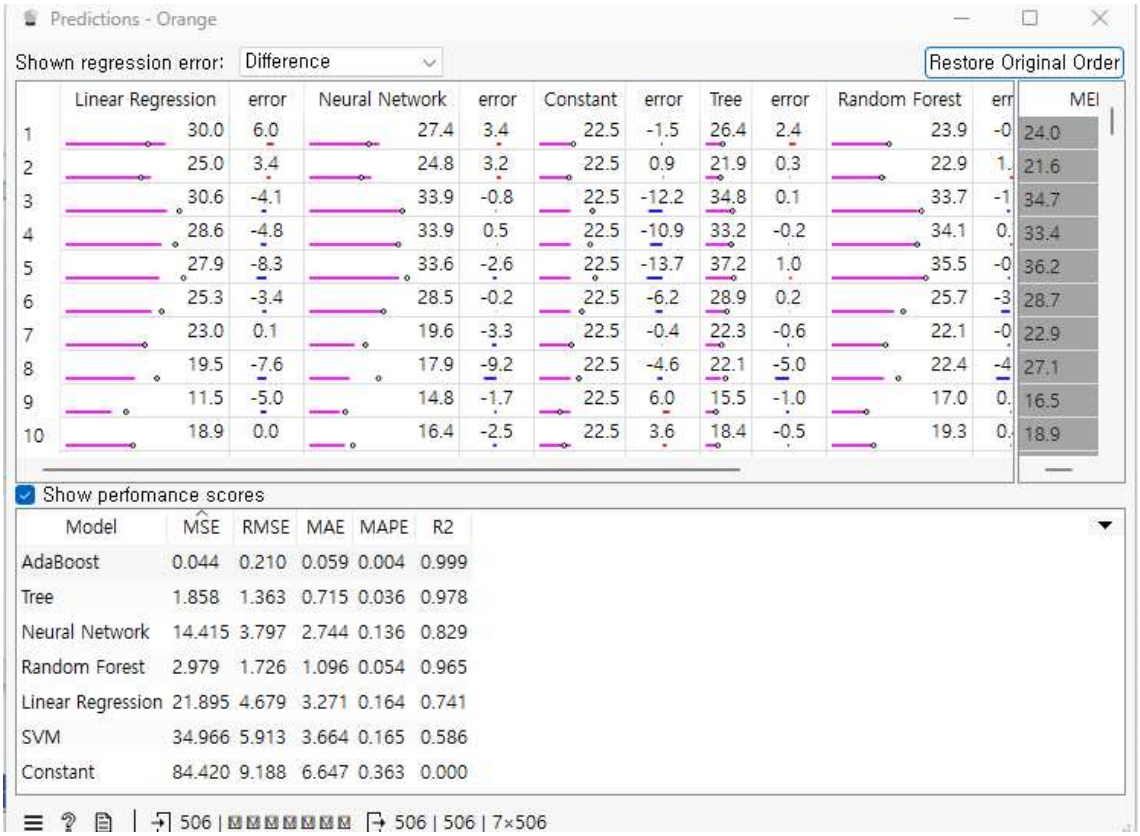
[5주차]

1. 주차 별 위젯 캡처



2. 모델 학습 후 테스트 데이터 결과 사진과 결과 요약

– 100% 학습 데이터로 예측 했을 때 (prdictions 위젯 활용)



	Linear Regression	error	Neural Network	error	Constant	error	Tree	error	Random Forest	err	MEI
1	30.0	6.0	27.4	3.4	22.5	-1.5	26.4	2.4	23.9	-0	24.0
2	25.0	3.4	24.8	3.2	22.5	0.9	21.9	0.3	22.9	1.	21.6
3	30.6	-4.1	33.9	-0.8	22.5	-12.2	34.8	0.1	33.7	-1	34.7
4	28.6	-4.8	33.9	0.5	22.5	-10.9	33.2	-0.2	34.1	0.	33.4
5	27.9	-8.3	33.6	-2.6	22.5	-13.7	37.2	1.0	35.5	-0	36.2
6	25.3	-3.4	28.5	-0.2	22.5	-6.2	28.9	0.2	25.7	-3	28.7
7	23.0	0.1	19.6	-3.3	22.5	-0.4	22.3	-0.6	22.1	-0	22.9
8	19.5	-7.6	17.9	-9.2	22.5	-4.6	22.1	-5.0	22.4	-4	27.1
9	11.5	-5.0	14.8	-1.7	22.5	6.0	15.5	-1.0	17.0	0.	16.5
10	18.9	0.0	16.4	-2.5	22.5	3.6	18.4	-0.5	19.3	0.	18.9

Model	MSE	RMSE	MAE	MAPE	R2
AdaBoost	0.044	0.210	0.059	0.004	0.999
Tree	1.858	1.363	0.715	0.036	0.978
Neural Network	14.415	3.797	2.744	0.136	0.829
Random Forest	2.979	1.726	1.096	0.054	0.965
Linear Regression	21.895	4.679	3.271	0.164	0.741
SVM	34.966	5.913	3.664	0.165	0.586
Constant	84.420	9.188	6.647	0.363	0.000

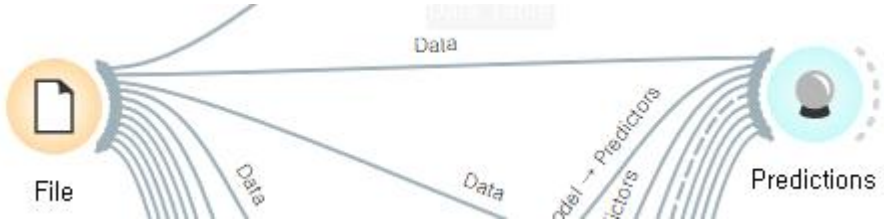
MSE(mean squared error) 버튼을 클릭하면 mean squared error가 작은 순서대로 Sorting 된다. 여기서 mean squared error값이 작은 모델일수록 예측성능이 우수하다. 즉 정답값과 유사하다.

모델을 경쟁시켜 평가 지표를 비교한 결과 AdamBoost –〉 Tree –〉 Neural Network –〉 Random Forest –〉 Linear Regression –〉 SYM –〉 Constant 순으로 정답값과 유사했다.

Ada Boost는 부스팅 알고리즘이 적용된 앙상블 모델, 그리고 초기에 neural network(= solver:최적화 알고리즘, 경사하강법, 성능을 좀 더 개선한 기계는 Adam) 와 linear regression(선형 회귀 모델)을 비교 했을 때, neural network가 mean squared error가 rmse, mae가 작다. 그리고 R2는 1에 가까우면 더 좋은 성능을 내는데 R2값이 Linear Regression 모델보다 훨씬 큰 것이 자료에서 보인다.

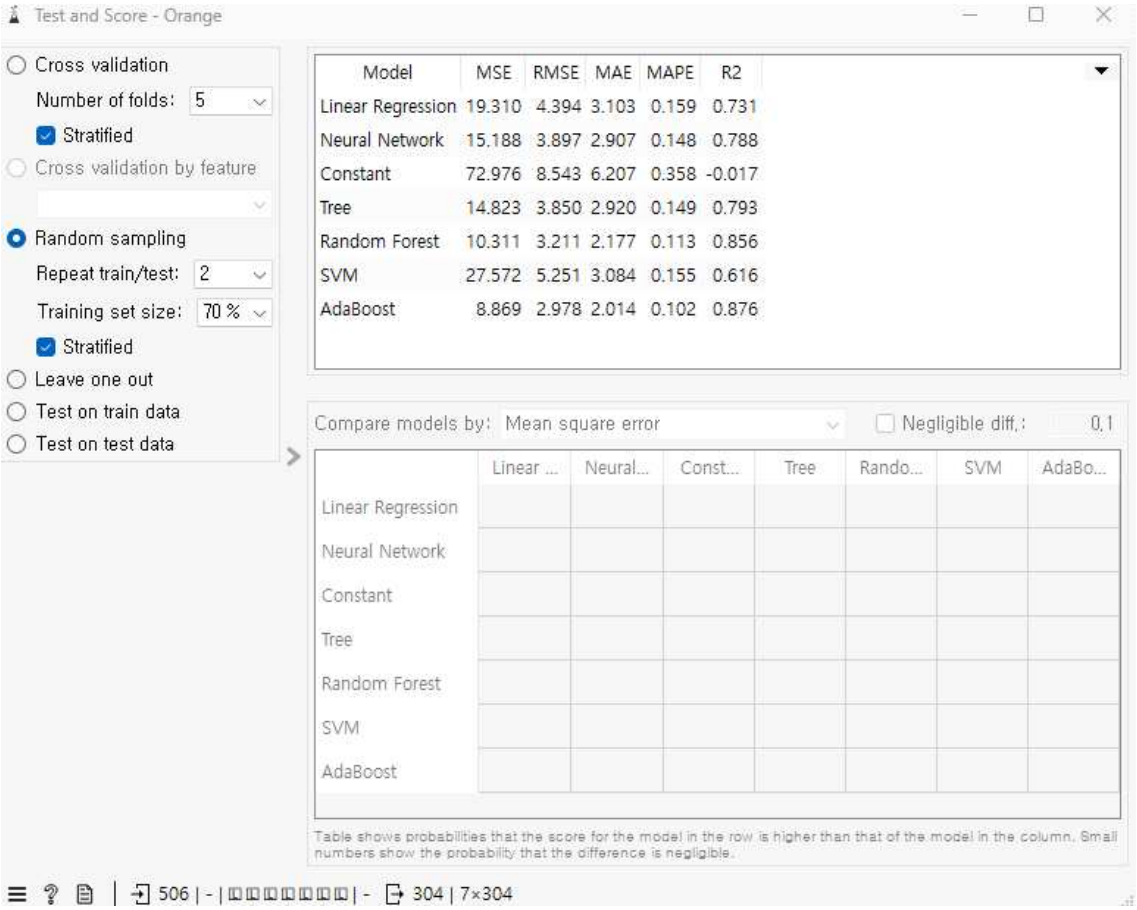
그러므로 MSE 버튼을 눌렀을 때, neural network가 Linear Regression 모델 보다 상단에 위치하는 것을 확인할 수 있다.

하지만 위의 예측이 공정한 것은 아니다. 왜냐하면 학습과 테스트에 같은 데이터를 사용하였기 때문이다.



1. 주차별 위젯 캡처 를 보면 Preditions를 그대로 File에 연결한 모습을 볼 수 있다.

– Test and Score 위젯을 추가



그렇기때문에 Train Data(학습할 때 사용하는 데이터) 와 Test Data(학습이 잘 됐는지 평가하는 데이터)로 나눠서 평가를 한다.

실습을 진행한 과정은 이러하다.

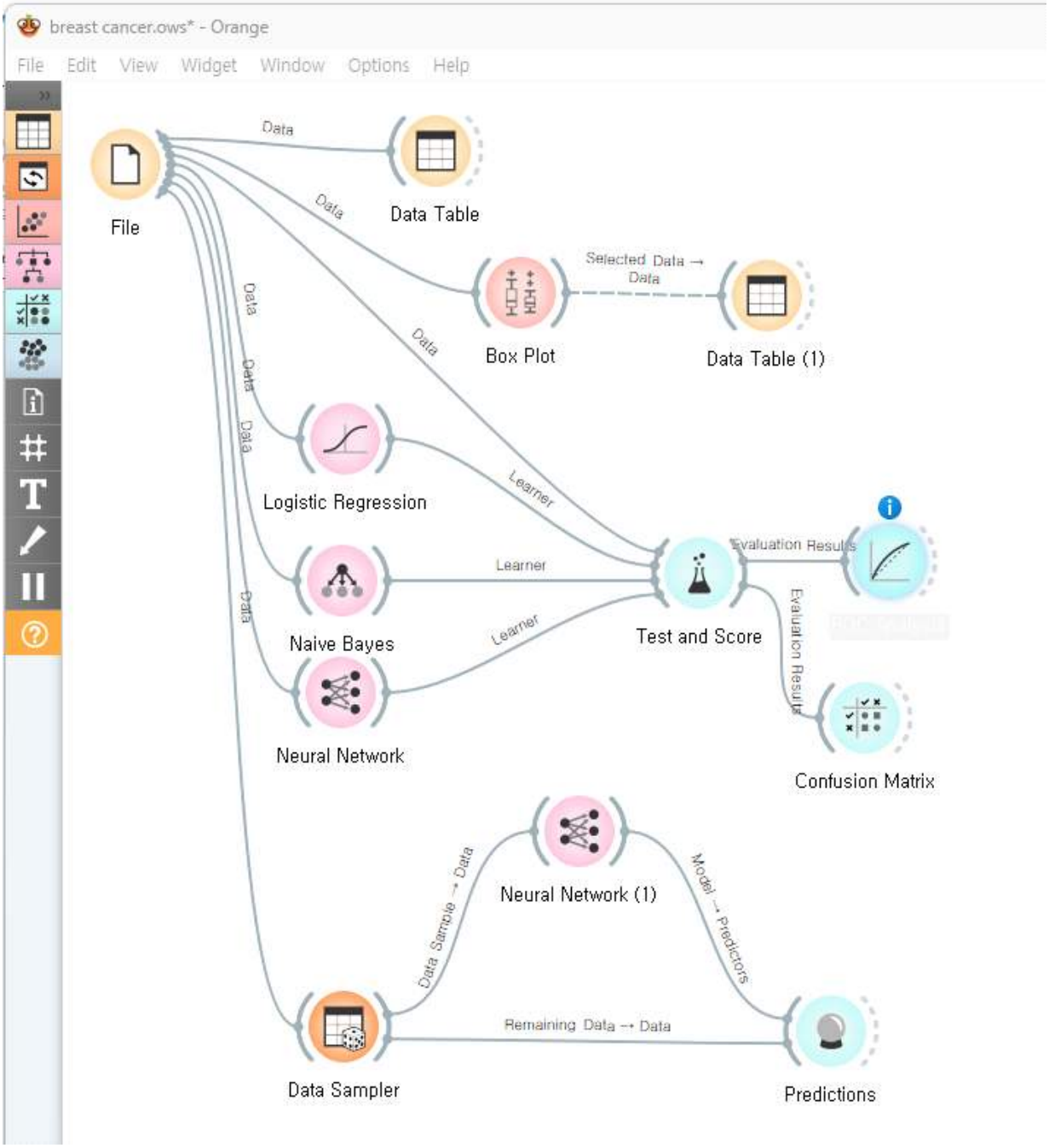
test and score 위젯을 선택 -〉 Train set과 test set으로 나누기 위해서 랜덤 샘플링 옵션을 선택 -〉 training set size를 70%로 조정 (나머지 30%는 테스트 데이터로 쓰이고, 70%는 학습하는데 활용하겠다는 옵션) -〉 repeat train.test = 계속 반복해서 랜덤 샘플하는 걸 임의로 2번으로 줄임.

결과를 보면 100% 학습 데이터를 가지고 테스트 했을 때랑 조금 다르다.

학습 데이터를 100% 테스트하는 데 쓰면 overfitting 문제를 확인 할 수 없어서 학습 성능은 우수하지만 테스트 성능은 떨어지는 일반화 성능 저하 문제가 생기기 때문에 train data 와 test data를 나눠서 좋은 모델을 찾는 평가에 활용해야 한다.

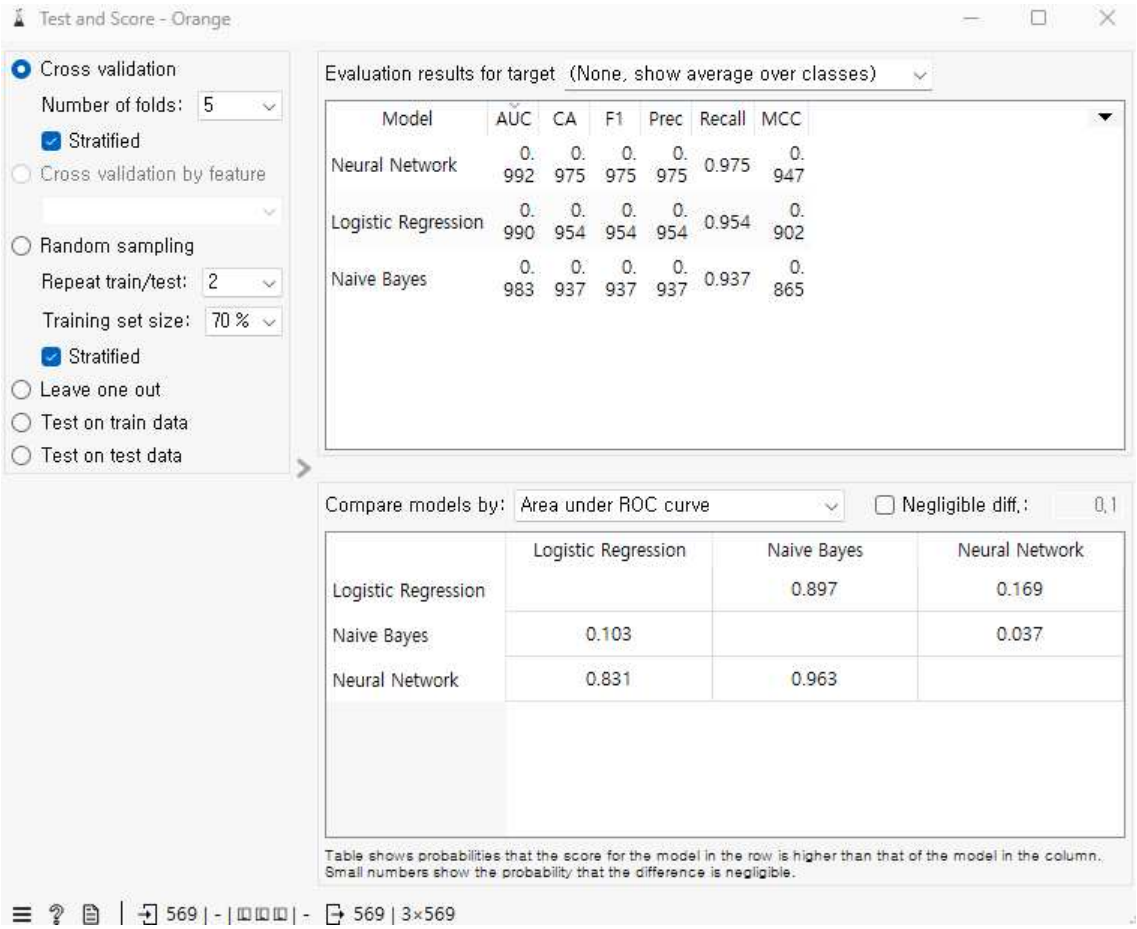
[6주차]

1. 주차 별 위젯 캡처



2. 모델 학습 후 테스트 데이터 결과 사진과 결과 요약

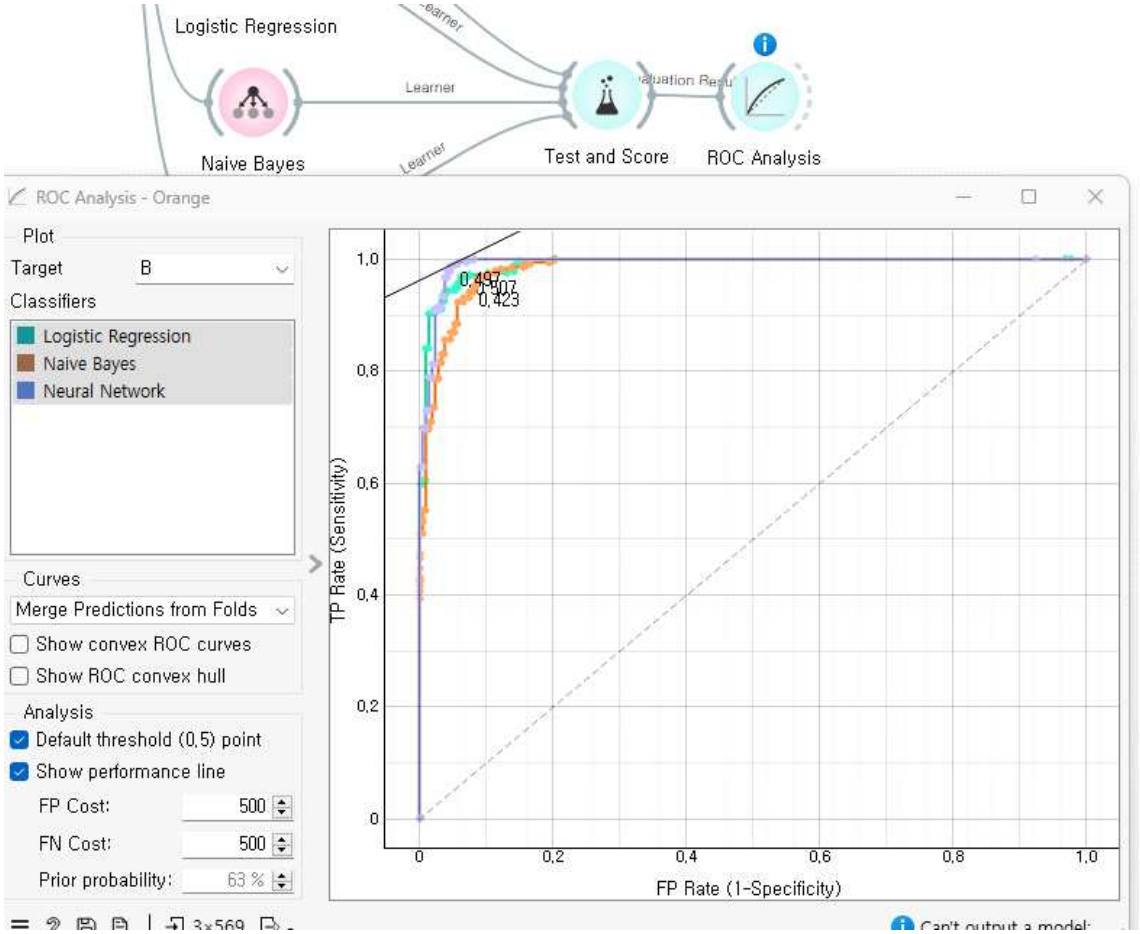
– test and score 위젯에서 cross validation(교차검증)



k번 나누고 각각의 학습 모델의 성능을 비교해서 그 평균값으로 성능을 표시하는 방법. 좌측 상단에 있는 Number of folds의 값을 5으로 하게 된다면 데이터를 5등분 한 뒤, 5분의 4는 훈련 데이터로 사용하고, 남은 1은 검증 데이터로 활용하겠다는 의미이다.

AUC 값이 1에 가까울수록 비교 성능이 좋은 모델이므로, 여기선 neural Network(신경망)가 가장 우수하고, 그 다음으로 Logistic Regression, 그리고 Naive Bayers가 있다.

– ROC Analysis 위젯



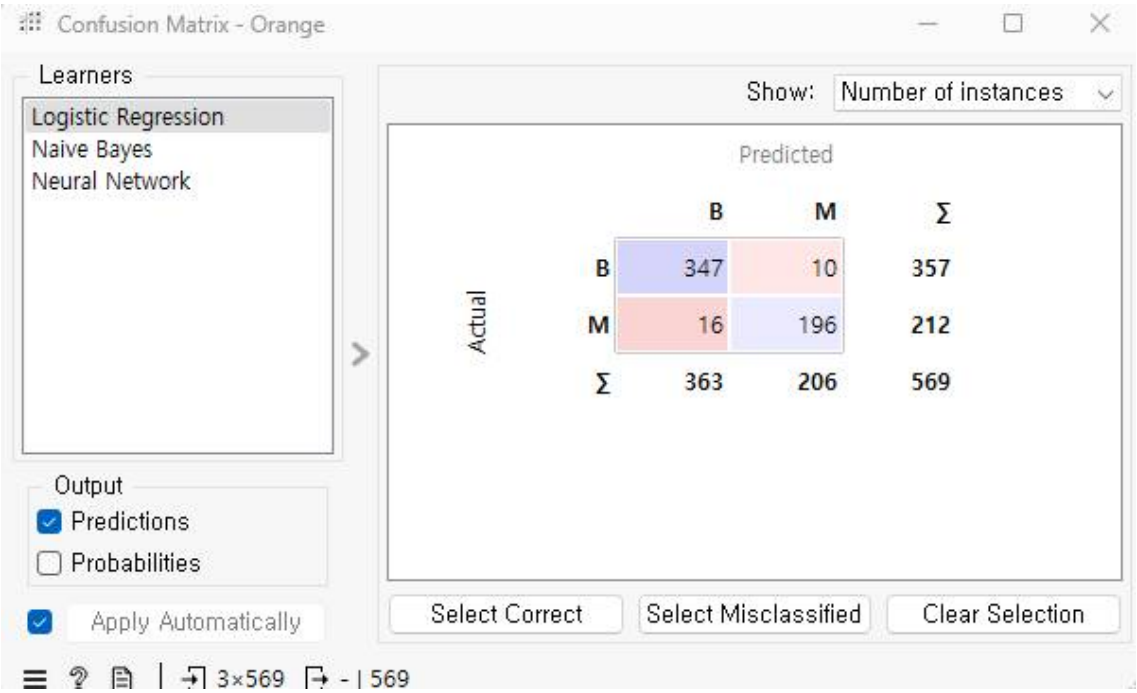
test and score 결과를 ROC Analysis에 연결을 하면, Evaluate result가 ROC 그래프 형태로 생성된다.

녹색 = Logisitic regression 모델
갈색 = Navie Bayes 모델
보라색 = Neural Network 모델

타깃이 benign(양성)일 경우와, malignan(악성)인 경우를 좌측 위 Target 버튼으로 볼 수 있다.

ROC curve에서도 보라색(neural network)가 1에 가까운 것을 볼 수 있다.

– Confusion Matrix (혼동행렬)



위와 마찬가지로 Test and Score에 연결하면, 이러한 값들이 나온다. 좌측 위 모델이 나열되어 있는 것 중, Logistic Regression을 비교하자면, 빨간값들이 틀린 값이다.

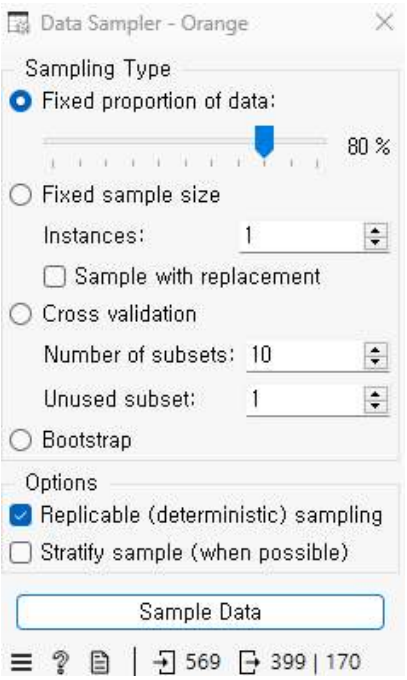
양성 중에 357개가 있었지만 347개를 맞췄지만, 10개의 오차가 있었고,
악성은 212개가 있었지만 196개를 맞추고, 16의 오차가 있었다는 뜻이다.

그래서 비교를 해보면,
neral Network 가 가장
오분류 개수가 작다.

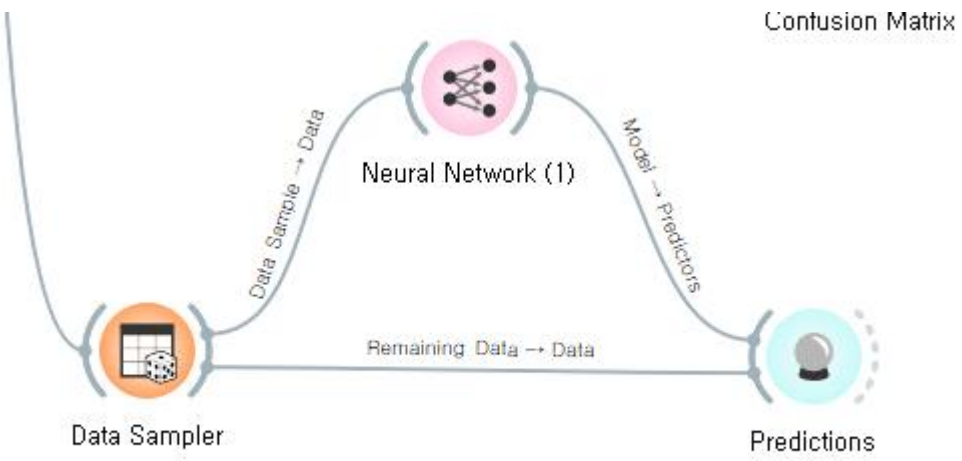
이러한 결과로 인해,
선택한 모델의 분류 성능
결과를 확인 할 수 있다.

– Data Sampler 위젯 추가

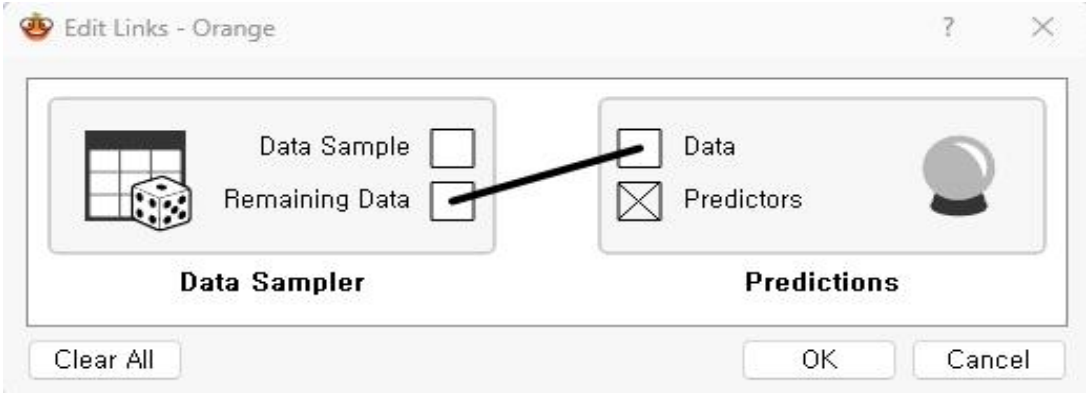
위 자료에서 Neural network 모델이 가장 우수하다는 걸 볼 수 있었다.
좋은 모델로 추론하기 위해서 Data Sampler 위젯을 추가해 훈련 데이터와 테스트 데이터로 분할하는 과정을 거칠 것이다.



Data sampler의 세팅값이다.
fixed proportion of data를 80%로 맞춰준다.
이것은 80%는 훈련 데이터로 쓰고, 20%는 테스트 데이터로 분할 하겠다는 의미이다.

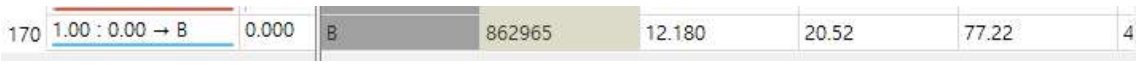


우선 Data Sampler 위젯과 File을 연결한 다음 위 사진처럼 Neural Network와 Predictions 위젯을 배치해준다.



Data Sampler 위젯과 Predictions 사이에 있는 링크를 클릭하면, 초기화면에는 Data sample과 data가 이어진 모양으로 뜨는데, 사진과 같이 바꿔준다.

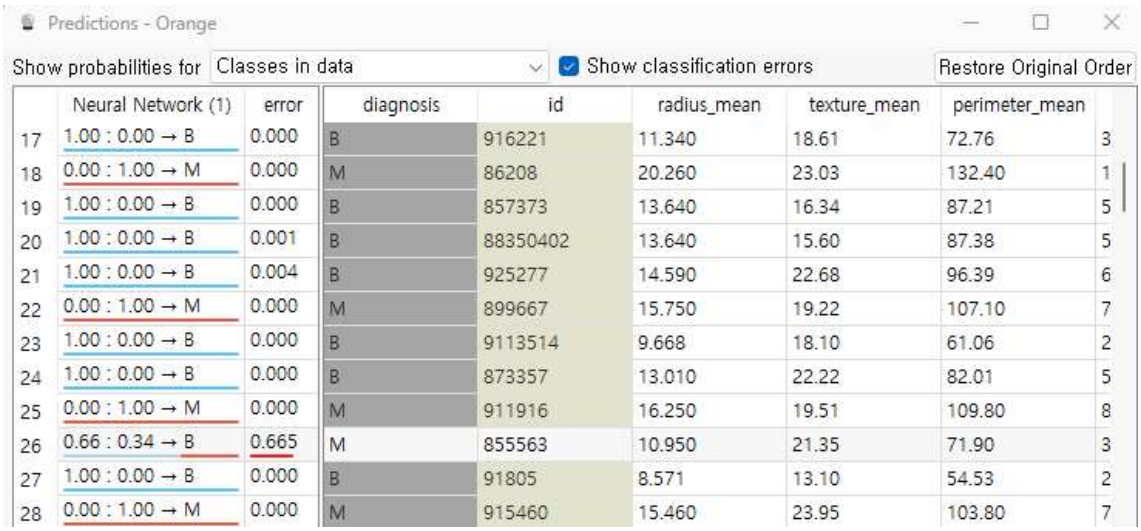
이렇게 하는 이유는 학습할 때 기존에 샘플링된 데이터 샘플링을 활용하고, 추론할때엔 나머지 테스트 데이터 20%를 예측에 활용하는 결과를 낳는다.



170	1.00 : 0.00 → B	0.000	B	862965	12.180	20.52	77.22	4

Predictions을 더블 클릭하면 데이터 개수가 170개가 나오는 것을 볼 수 있다.
569개의 데이터 중 20%, 즉 170개의 테스트 데이터가 나온다.

이 테스트 데이터가 실제로 Diagnosis 진단 결과와 신경망이 예측한 결과가 일치하는 지 확인해보겠다.

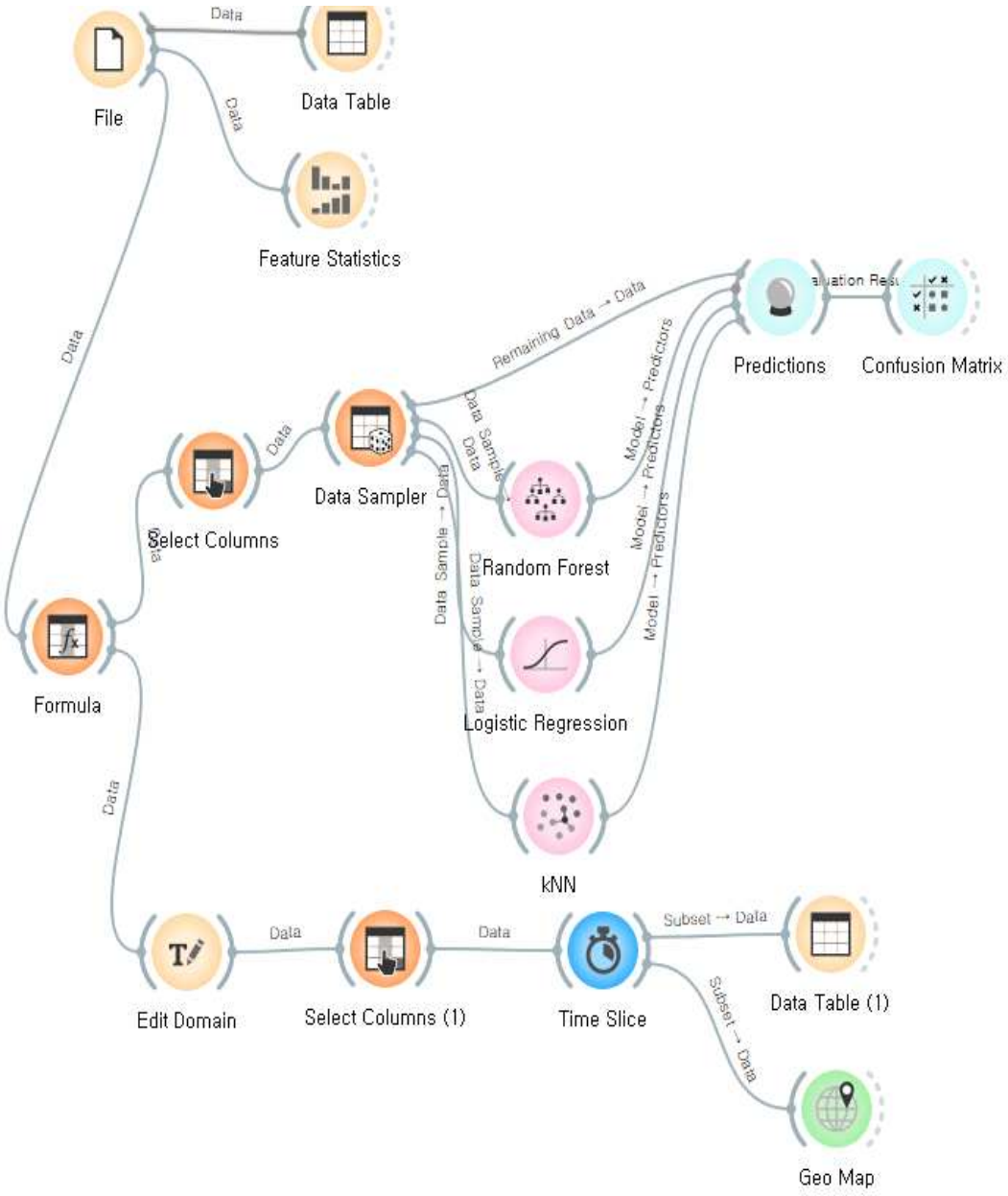


	Neural Network (1)	error	diagnosis	id	radius_mean	texture_mean	perimeter_mean	
17	1.00 : 0.00 → B	0.000	B	916221	11.340	18.61	72.76	3
18	0.00 : 1.00 → M	0.000	M	86208	20.260	23.03	132.40	1
19	1.00 : 0.00 → B	0.000	B	857373	13.640	16.34	87.21	5
20	1.00 : 0.00 → B	0.001	B	88350402	13.640	15.60	87.38	5
21	1.00 : 0.00 → B	0.004	B	925277	14.590	22.68	96.39	6
22	0.00 : 1.00 → M	0.000	M	899667	15.750	19.22	107.10	7
23	1.00 : 0.00 → B	0.000	B	9113514	9.668	18.10	61.06	2
24	1.00 : 0.00 → B	0.000	B	873357	13.010	22.22	82.01	5
25	0.00 : 1.00 → M	0.000	M	911916	16.250	19.51	109.80	8
26	0.66 : 0.34 → B	0.665	M	855563	10.950	21.35	71.90	3
27	1.00 : 0.00 → B	0.000	B	91805	8.571	13.10	54.53	2
28	0.00 : 1.00 → M	0.000	M	915460	15.460	23.95	103.80	7

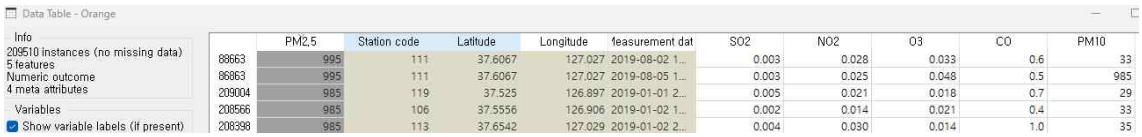
26번 결과에서 의사 선생님은 B(양성)으로 진단했는데 신경망은 M(악성)으로 오분류 한 결과가 있다. 잘못된 추론 결과가 있긴 하지만, 살펴보면 거의 90% 이상 예측에 성공하였다.

[7주차]

1. 주차 별 위젯 캡처



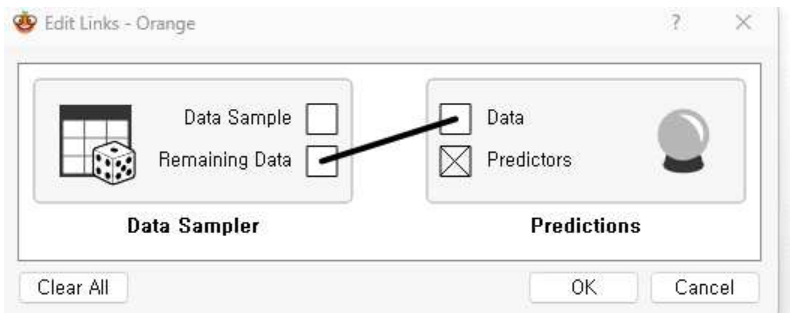
2. 모델 학습 후 테스트 데이터 결과 사진과 결과 요약



우선 PM2.5가 target이기 때문에 Data Table에서 확인해준다.

Feature Statistics는 데이터 특성 통계표를 확인 할 수 있다.

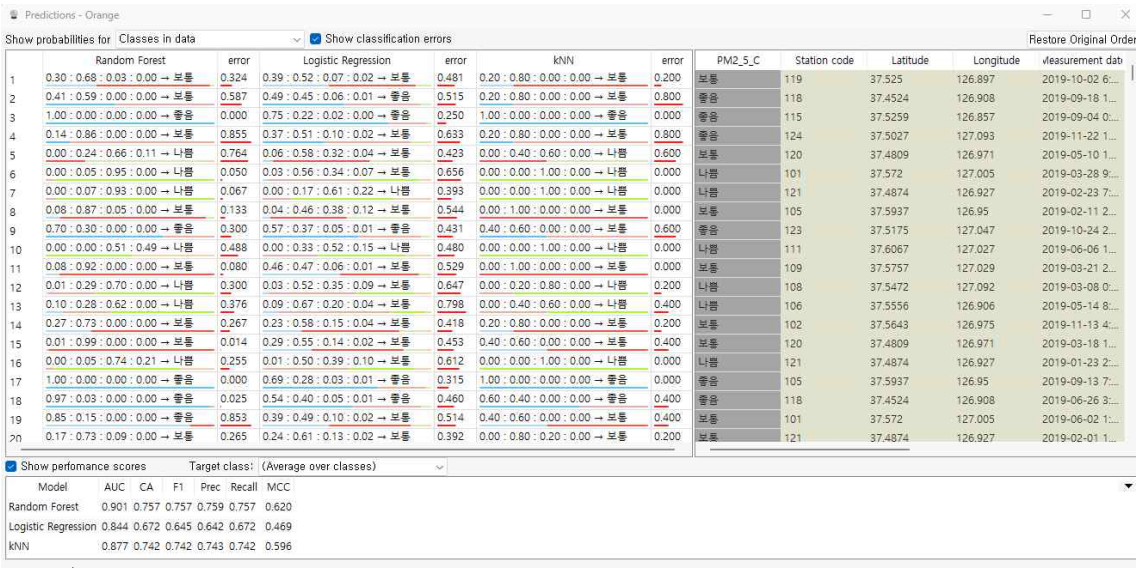
우선 Linear Regression 모델과 Random Forest 모델을 선택 -〉 data sampler와 연결해준다.



여기서 DataSample과 Prediction 사이는 테스트를 위한 것이기 때문에 Remaining Data를 선택하여야 한다.

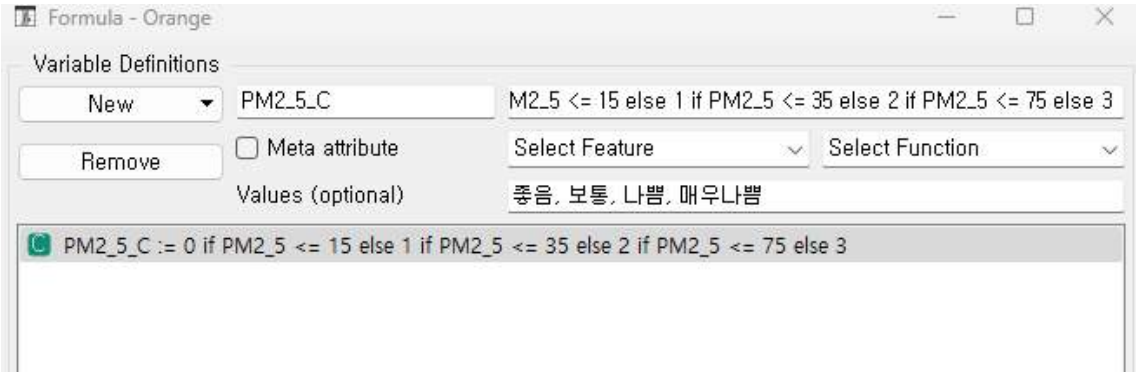
이로써 각 모델들은 80%만 학습을 한다. (data sampler에서 Fixed proportion of data 값을 80%로 줌)

이제 각 Prediction 위젯과 모델을 연결해서 각 모델의 출력을 알아보자.



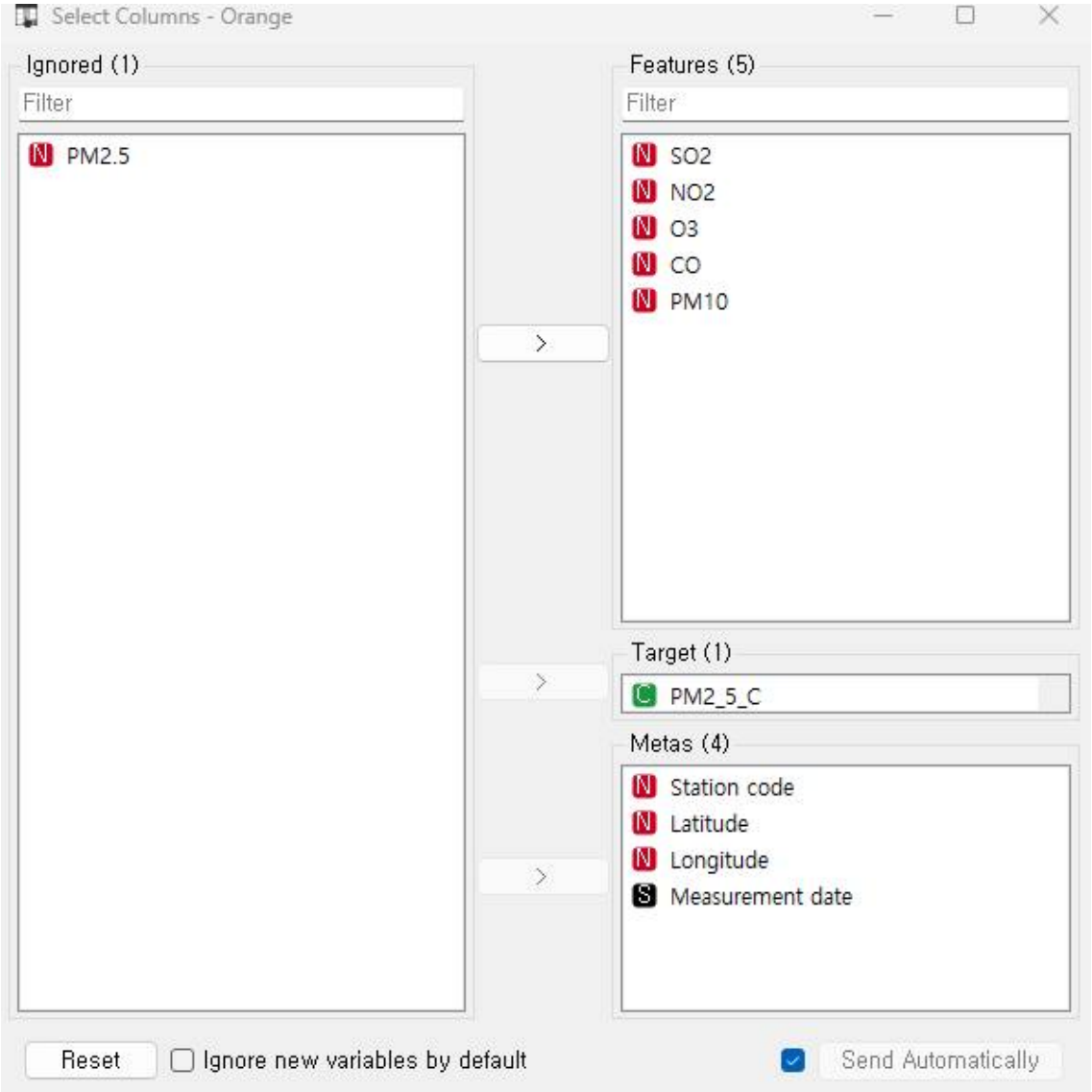
이전에 Liner 모델이 있었을 때는, Liner 모델과 random Forest 두 모델 모두 예측값이 정확하지 않았다.

PM2.5 수치값을 예측하는 회귀 문제에서 -〉 PM2.5의 예보 등급을 예측하는 분류 문제로 문제 수정 = 데이터 속성을 추가해서 새롭게 데이터 추가 =〉 feature constructor 위젯 추가 (지금은 formula로 이름 변경)



formula에서 Variable definitions 섹션 -〉 new (새로운 변수 생성) = Categorical로 데이터 속성 지정 (PM2.5를 분류하려고) -〉 expression에 조건을 정의한다.

-〉 Formula와 Data Sampler 사이에 Select Colums 위젯을 넣는다.



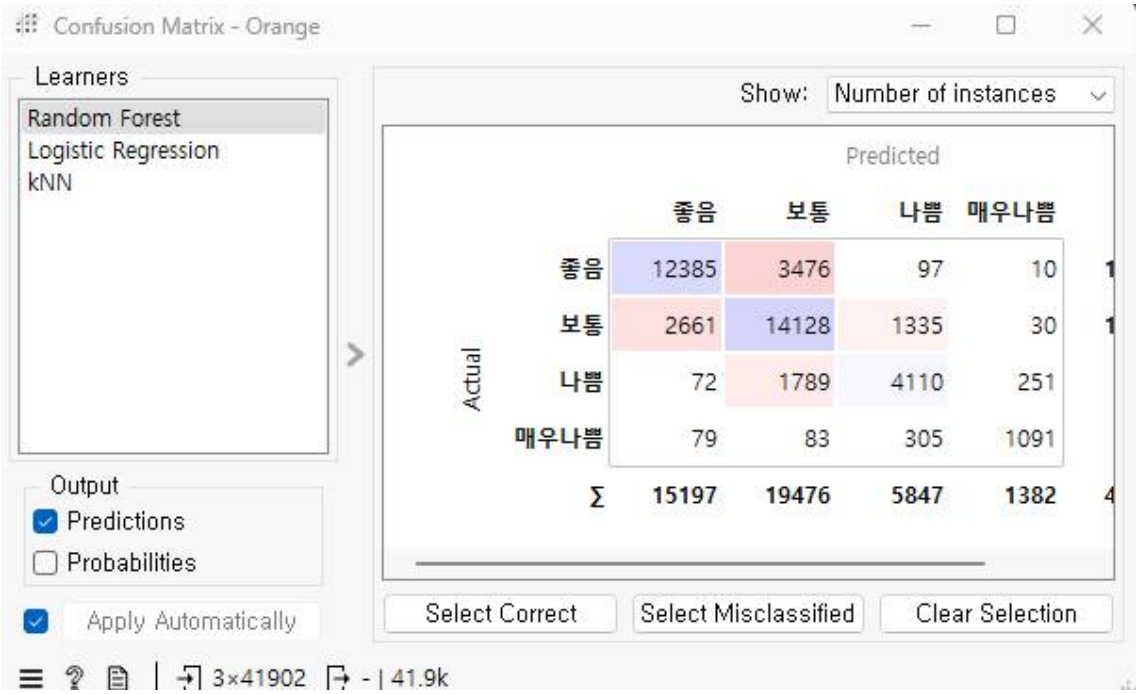
ignored에 PM2.5를 넣고 Target에 아까 Formula에서 만들었던 변수를 범주형 변수로 추가한다.

이랬더니 기존의 Linear 모델은 쓸 수 없어 삭제하였다.

분류 문제이기 때문에 분류 문제의 모델 성능 평가하기 위해서 confusion matrix 위젯을 가져와서 예측 위젯과 연결.

Model	AUC	CA	F1	Prec	Recall	MCC
Random Forest	0.901	0.757	0.757	0.759	0.757	0.620
Logistic Regression	0.844	0.672	0.645	0.642	0.672	0.469
kNN	0.877	0.742	0.742	0.743	0.742	0.596

Predictions를 보니 Random Forest 모델이 제일 우수한 성능을 보여주고 있다.



후에 Confusion Martrix 위젯을 클릭하면 대각성 성분이 맞게 예측한 값, 나머지는 오분류를 가리키고 있다. 이로써 각 모델 별로 혼돈 행렬을 확인 할 수 있다.

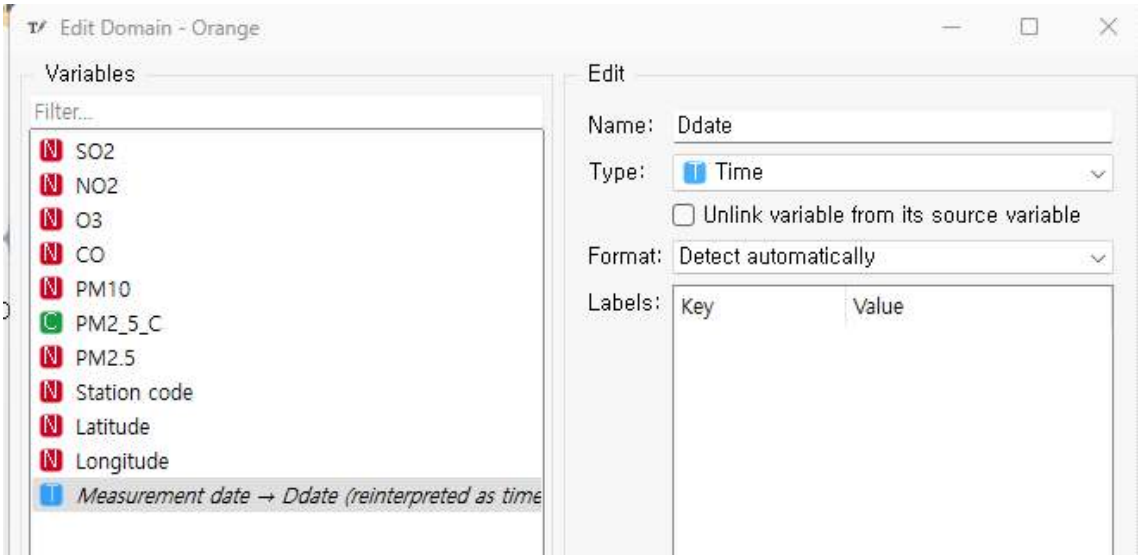
이제 이것을 시각화 해보려고 한다.

options -〉 add-ones =〉 geo, time series 옵션 추가

geo 카테고리 = 위도와 경도를 활용해서 지도로 보는 기능을 제공하는 위젯
time series 카테고리 = 시간의 흐름에 따라 순차적으로 기록된 데이터를 기록하는 위젯

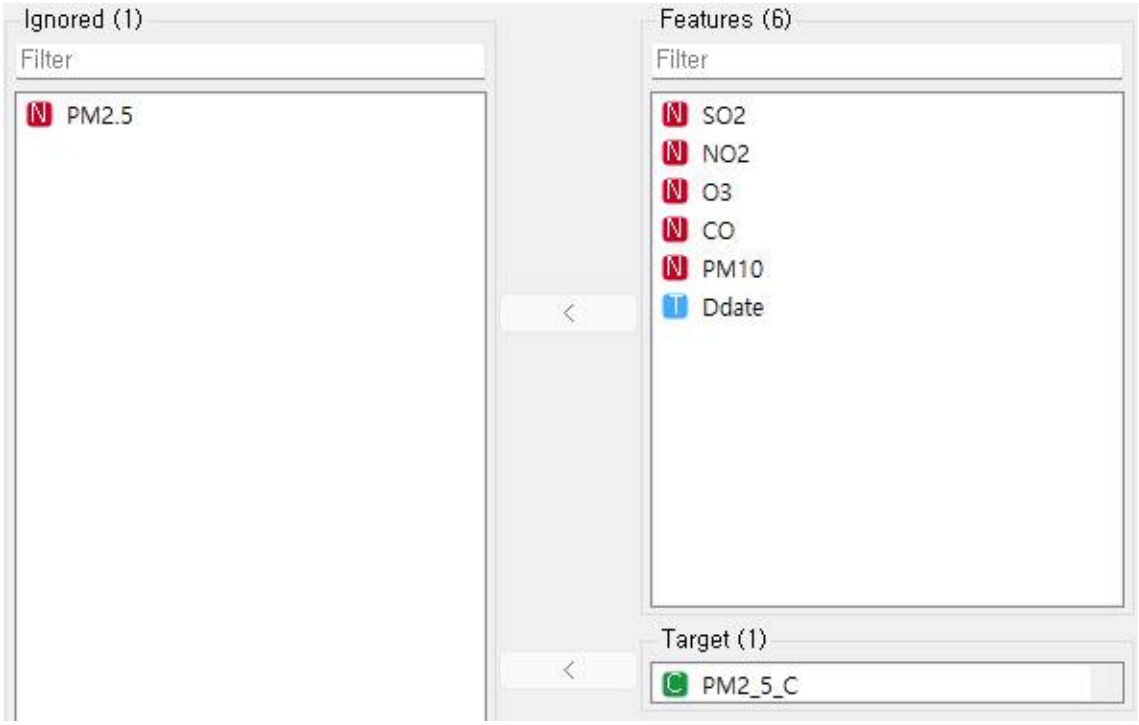
이므로 1시간 단위로 대기 오염 측정 데이터를 분석하고 싶기 때문에, time series의 time slice(시간에 따른 데이터의 변화를 확인하기 위해)를 쓸 것이다.

그 전에 ! 오렌지에서는 특정한 형식 데이터만 시계열 데이터로 자동 인식하기 위해서 형변환을 위해 edit Domain 위젯을 추가할 것이다.



formula와 edit Domain 연결 –〉 Measurement date(시계열 데이터이기 때문에) 선택 –〉 name을 Ddate라고 변경, 타입도 time으로 변경.

그리고 edit domain select columns(1) 하나 더만들어서 Ddate를 feature로 추가.



시간에 따른 데이터 변화를 확인하기 위해 time slice 위젯 활용한다.

=〉 시간 변화에 따라 위젯을 잘라서 확인할 수 있다. –〉 data tabel 위젯이랑 연결 –〉 시간 단위 별로 Ddate가 짤려있다.

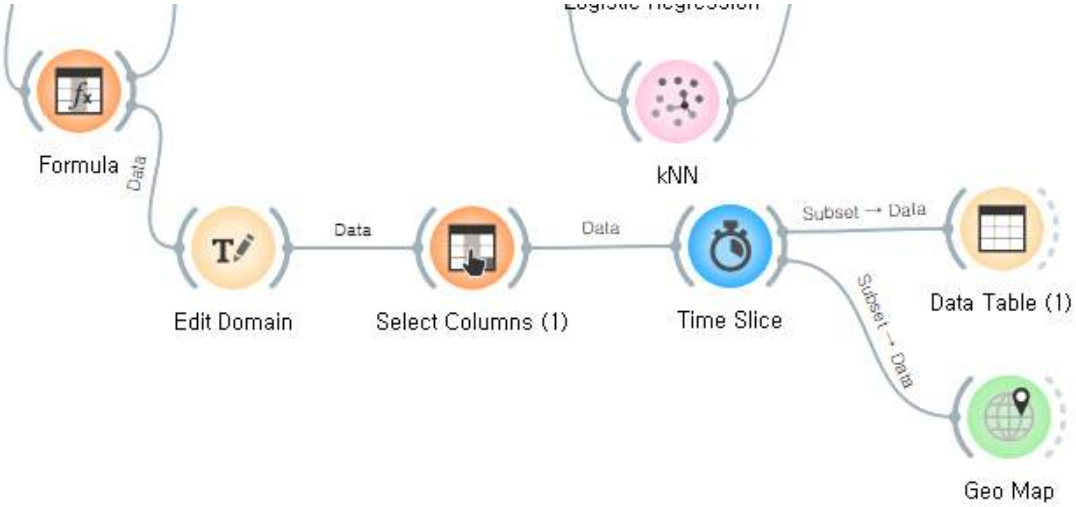

Data table (1) - Orange

Info
5 instances (no missing data)
features
arget with 4 values
meta attributes

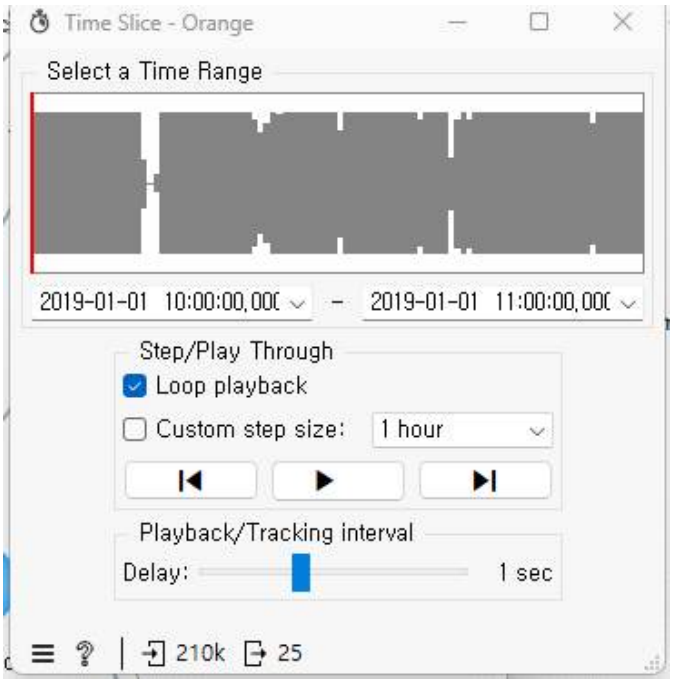
Variables
Show variable labels (if present)
Visualize numeric values
Color by instance classes

		PM2_5_C	Station code	Latitude	Longitude	SO2	NO2	O3	CO	PM10	Ddate
1	보통		125	37.545	127.137	0.005	0.060	0.002	1.0	48	2019-01-01 0...
2	보통		101	37.572	127.005	0.004	0.061	0.002	1.1	37	2019-01-01 0...
3	보통		102	37.5643	126.975	0.003	0.056	0.002	0.8	38	2019-01-01 0...
4	보통		103	37.54	127.005	0.002	0.051	0.002	0.8	34	2019-01-01 0...
5	보통		104	37.6098	126.935	0.004	0.039	0.004	1.0	34	2019-01-01 0...
6	보통		105	37.5937	126.95	0.007	0.048	0.003	1.6	41	2019-01-01 0...
7	보통		106	37.5556	126.906	0.003	0.036	0.002	0.7	52	2019-01-01 0...
8	보통		107	37.5419	127.05	0.003	0.053	0.002	0.9	41	2019-01-01 0...

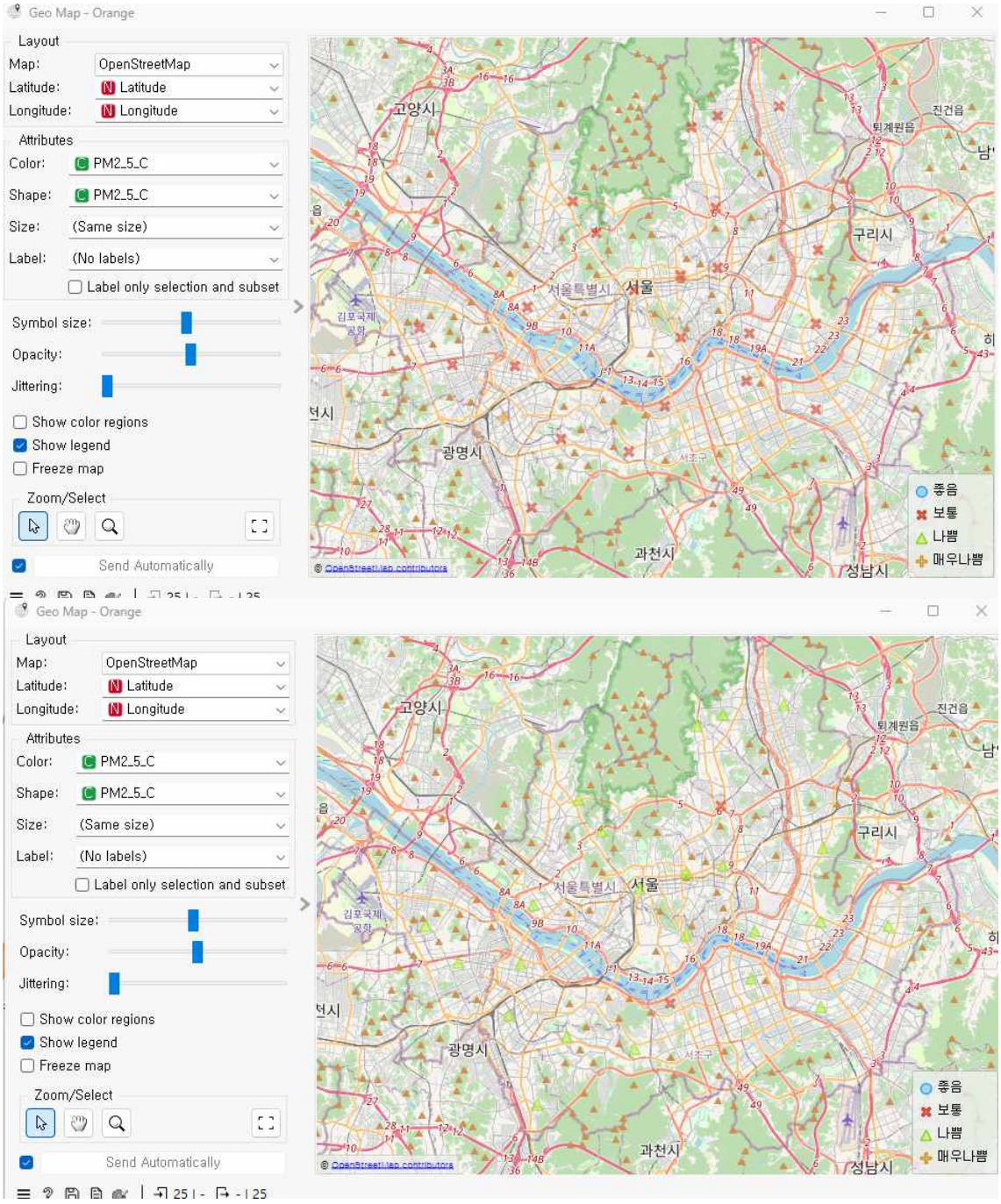
2019년 1월 1일의 미세먼지는 보통이었음을 알 수 있다.



여기서 Time Slice를 클릭하면



이런 화면이 뜨는데 재생 버튼을 누른 뒤, Geo Map(지도에 표시)을 클릭하면, 시간 별로 초미세먼지의 지표(좋음,보통,나쁨,매우나쁨)으로 변화하는 과정을 볼 수 있다.



시간에 따라서 색깔과 아이콘이 달라지는 모습을 확인할 수 있다.