QGIS 3.4 Orfeo ToolBox(OTB) 머신러닝 기반 드론 영상분석

2019. 1.14.
국립공원공단 유병혁

QGIS 3.4 Orfeo ToolBox(OTB) - 머신러닝 기반 드론 영상분석

QGIS 3.4 Orfeo ToolBox(OTB) - (1) 연동 설정하기
http://blog.daum.net/geoscience/1284

QGIS 3.4 Orfeo ToolBox(OTB) - (2) 파이썬에서 OTB 호출하기
http://blog.daum.net/geoscience/1322

QGIS 3.4 Orfeo ToolBox(OTB) - (3) 무인기 영상 세그먼테이션(분할)
http://blog.daum.net/geoscience/1324

QGIS 3.4 Orfeo ToolBox(OTB) - (4) 훈련데이터 만들기
http://blog.daum.net/geoscience/1325

QGIS 3.4 Orfeo ToolBox(OTB) - (5) 머신러닝 분류기
http://blog.daum.net/geoscience/1326

QGIS 3.4 Orfeo ToolBox(OTB) - (6) 가시광선 식생지수 계산하기(BandMath)
http://blog.daum.net/geoscience/1330

QGIS 3.4 Orfeo ToolBox(OTB) - (7) 파이썬 콘솔에서 레이어 추가하기
http://blog.daum.net/geoscience/1318

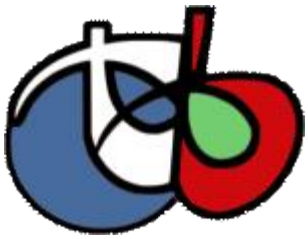
QGIS 3.4 Orfeo ToolBox(OTB) - (8) 가시광선 식생지수 계산하기(파이썬 콘솔)
http://blog.daum.net/geoscience/1331

QGIS 3.4 Orfeo ToolBox(OTB) - (9) 머신러닝 분류기에 피처 추가하기
http://blog.daum.net/geoscience/1329

QGIS 3.4 Orfeo ToolBox(OTB) - (1) 연동 설정하기

안녕하세요? 이번 글은 QGIS 3.4에서 Orfeo Toolbox (OTB) 6.6을 연동하는 방법을 정리해 보겠습니다.

Orfeo ToolBox(오르페오 툴박스)는 최신 원격탐사를 위한 오픈소스 프로젝트입니다.
테라바이트 규모에서 고해상도 광학, 다중분광, 라이다 영상들을 처리할 수 있습니다.



Orfeo ToolBox 공식 홈페이지: https://www.orfeo-toolbox.org/

일단, QGIS와는 별개의 프로젝트이므로 공식 홈페이지에서 OTB를 다운로드 합니다.
OTB 다운로드: https://www.orfeo-toolbox.org/download/

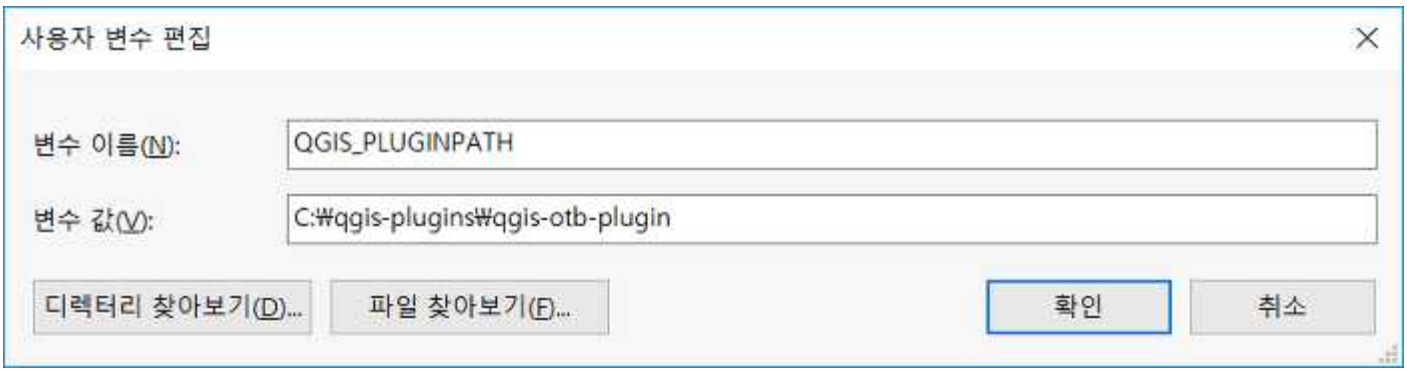
OTB는 zip파일로 제공되며 저는 C:₩OTB-6.6.0-Win64 폴더에 압축 해제하였습니다.
이제 QGIS에서 이 위치가 인식되어 서로 연동되면 되겠죠?!

연동을 위해 QGIS Orfeo ToolBox (OTB) 플러그인을 설치합니다.
qgis-otb-plugin | https://gitlab.orfeo-toolbox.org/orfeotoolbox/qgis-otb-plugin

OTB 플러그인은 현재 수동 설치를 참조하지만 초기 테스트 후에 qgis 처리 코어로 이동될 예정입니다.
이 플러그인을 통해 QGIS에서 모자이크, 정사보정, 영상분류, 영상융합, 심지어 텐서플로 기반 딥러닝과 같은 다양한 응용 프로그램에 접근할 수 있습니다.

일단 깃랩에 등록된 qgis-otb-plugin 저장소를 PC에 복사합니다.
저는 공식 깃랩에서 안내한 c:₩qgis-plugins₩qgis-otb-plugin 를 그대로 따랐습니다.

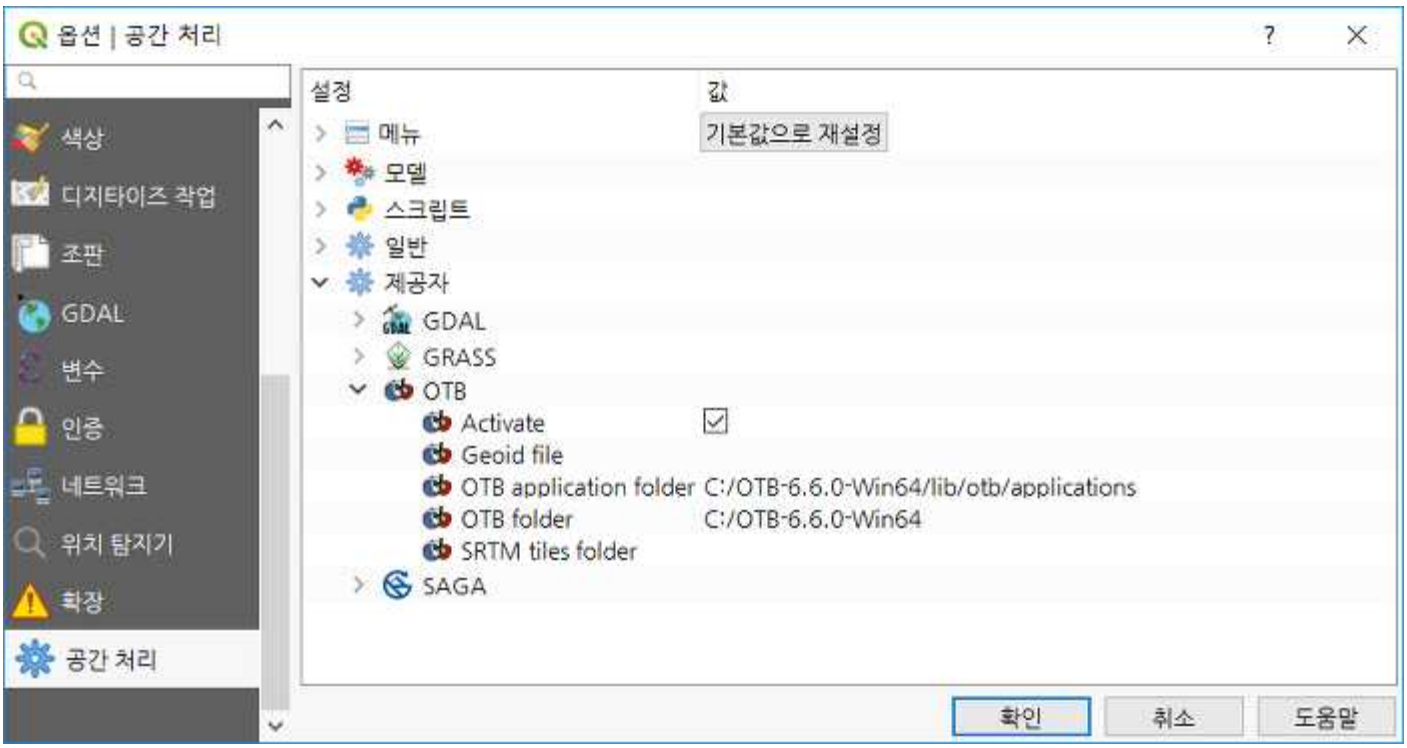
계정의 환경 변수 편집에서 아래와 같이 QGIS_PLUGINPATH 사용자 변수를 생성합니다.



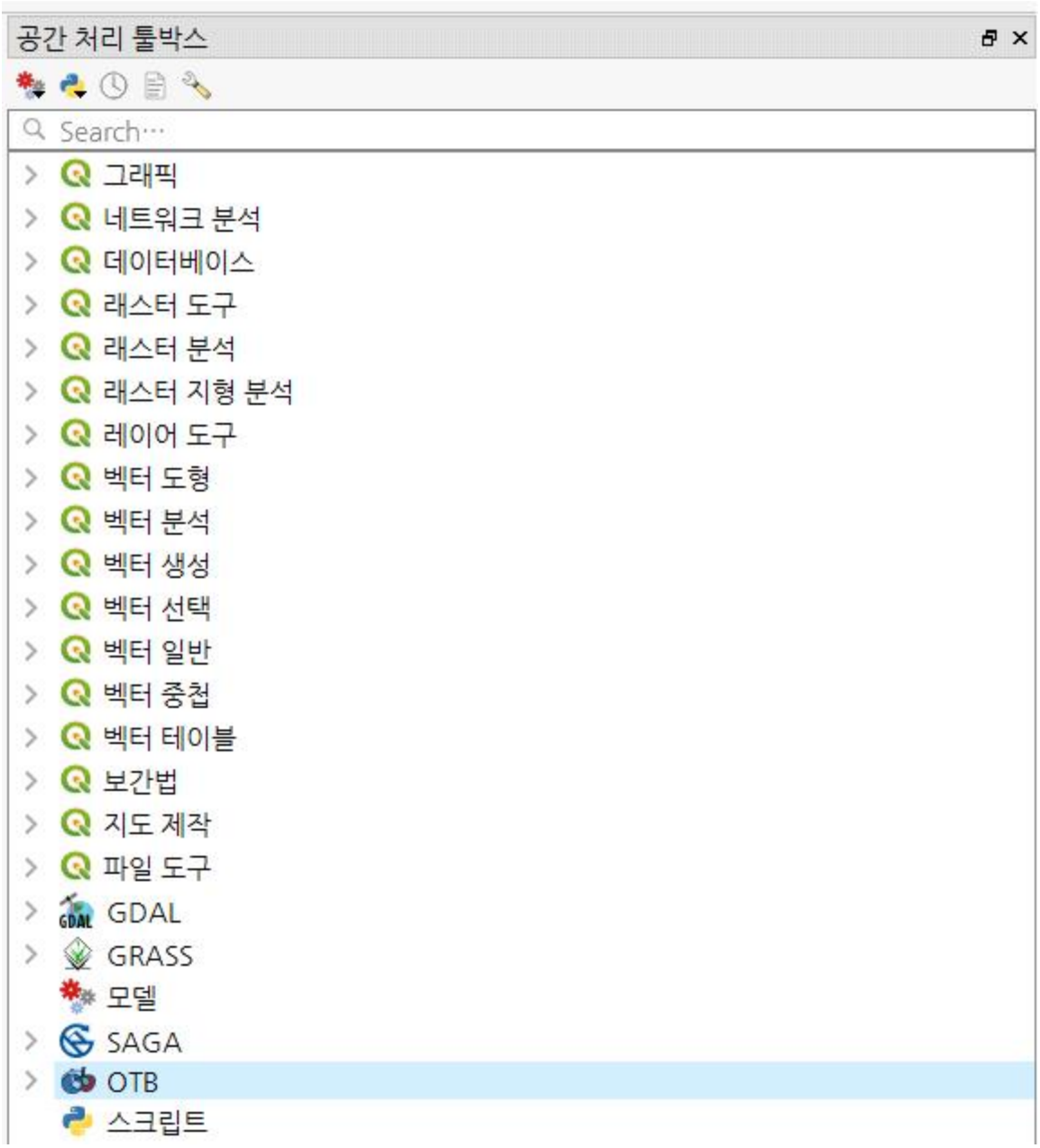
자, 이제 QGIS 3.4를 실행하고 상단 메뉴에서 '플러그인 >플러그인 관리 및 설치'를 선택합니다.

플러그인 창에서 '설치됨'을 선택하고 'Orfeo ToolBox (OTB)' 플러그인을 아래와 같이 체크합니다.

이제 상단 메뉴에서 '설정 >옵션'을 선택하고 좌측 '공간 처리' 탭을 선택합니다.
Activate를 체크하고 OTB application folder와 OTB folder를 아래와 같이 지정합니다.



공간 처리 툴박스에 OTB가 추가되었습니다. 간단하죠?!



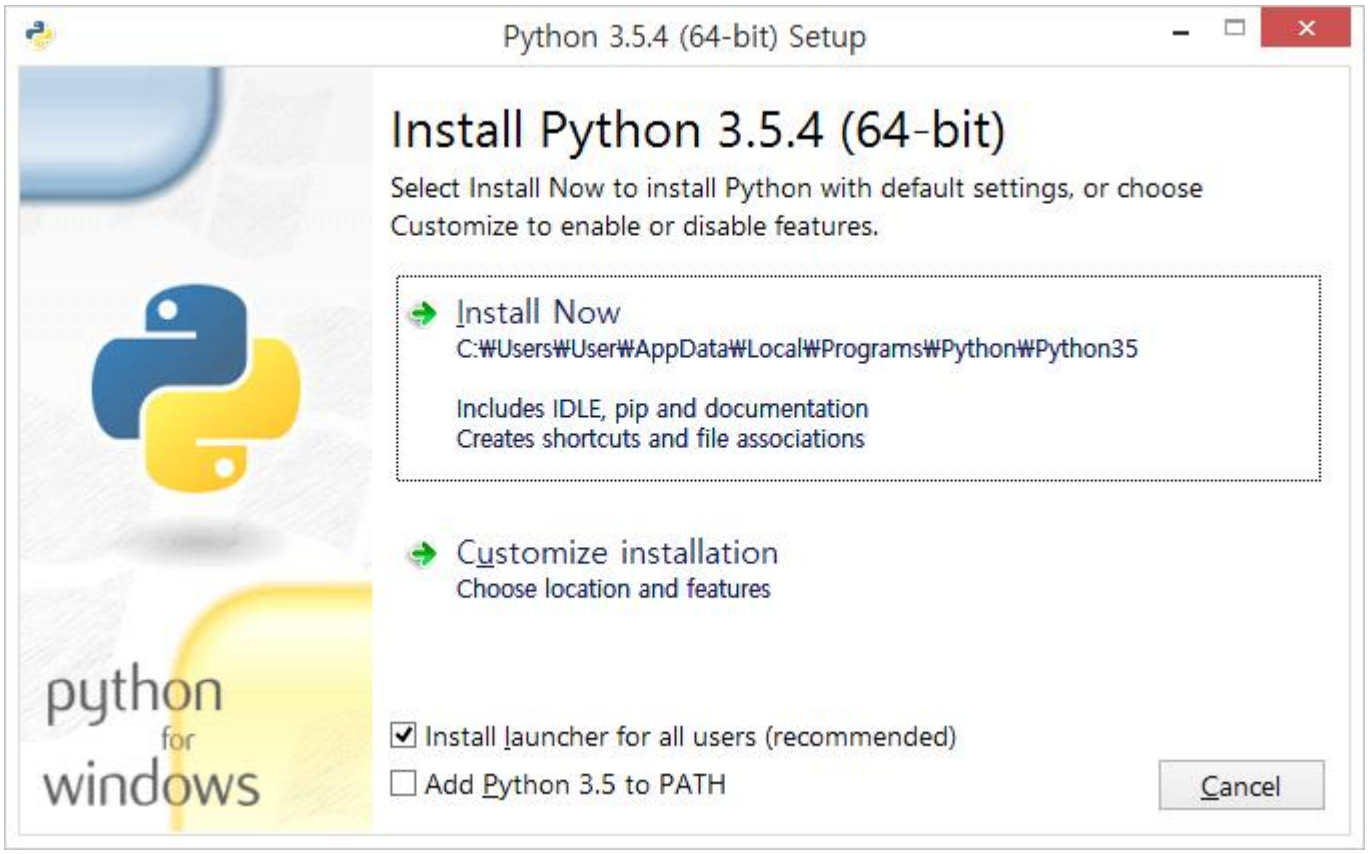
자, 이제 OTB의 최신 원격탐사 기능들을 QGIS에서 바로 이용하실 수 있습니다.

QGIS 3.4 Orfeo ToolBox(OTB) - (2) 파이썬3에서 OTB 호출하기

안녕하세요? 이번 글은 파이썬 3에서 Orfeo ToolBox(줄여서 OTB)를 호출하는 방법을 정리해 보겠습니다.

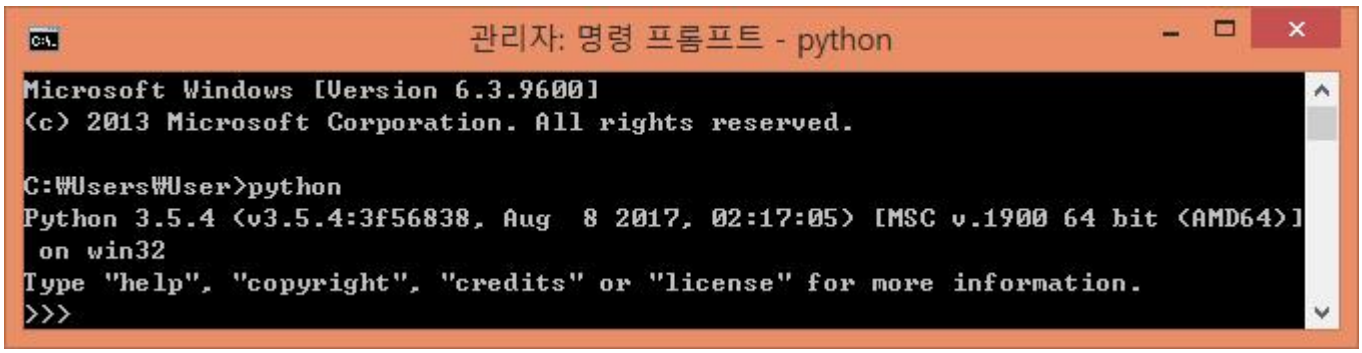
OTB는 파이썬 2.7과 3.5를 위한 바인딩을 제공하는데요, 여기선 파이썬 3.5에서 OTB를 설정하겠습니다.
Python bindings | https://www.orfeo-toolbox.org/CookBook/Installation.html#python-bindings

파이썬 3.5는 '파이썬 공식 홈페이지(https://www.python.org/)'에서 내 PC용 파일을 받아주시면 됩니다.



설치하실 때 'Add Python 3.5 to PATH'를 클릭해주시기 바랍니다.

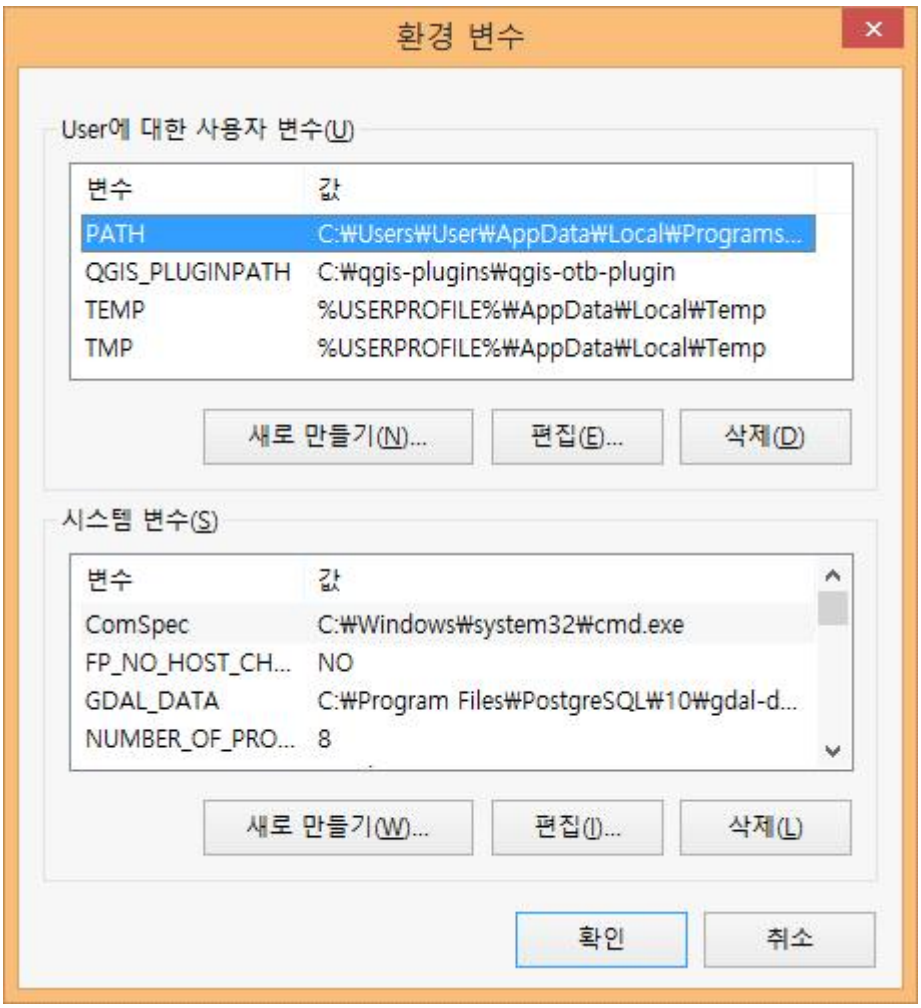
이제 명령 프롬프트에서 python을 클릭하시면 아래와 같이 Python 3.5가 실행됩니다.



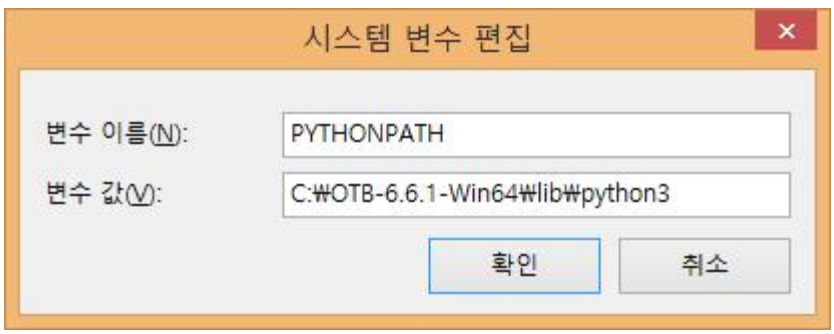
OTB 바인딩 설정을 위해서는 넘파이(NumPy)를 설치해주셔야 합니다.

```
python -m pip install --upgrade pip
python -m pip install --upgrade numpy
```

이제 '환경 변수 >시스템 변수'를 설정해 주시면 되겠는데요,

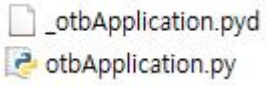


먼저 PYTHONPATH를 다음과 같이 설정합니다:
변수 이름: PYTHONPATH
변수 값: C:₩OTB-6.6.1-Win64₩lib₩python3
※ OTB 경로는 설정해주신 위치에 맞게 수정하시기 바랍니다.



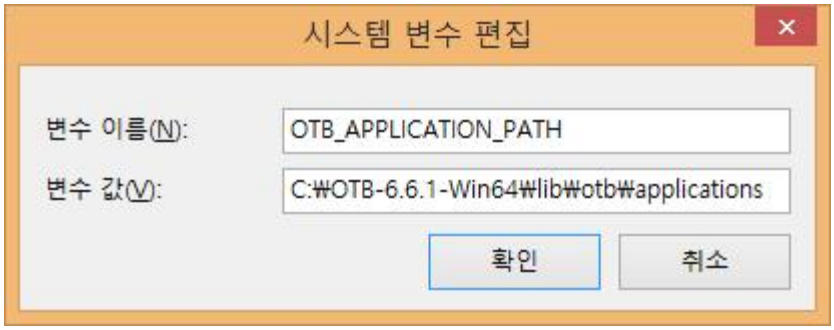
lib 폴더에는 python과 python3 폴더가 있습니다. 각각 Python 2.7과 Python 3.5용 폴더로 보시면 되겠습니다.

python3 폴더에는 아래 파일들이 위치하고 있습니다.



다음으로는 시스템 변수 PATH에 OTB bin 폴더 위치를 추가합니다.
변수 이름: PATH
변수 값: C:₩OTB-6.6.1-Win64₩bin

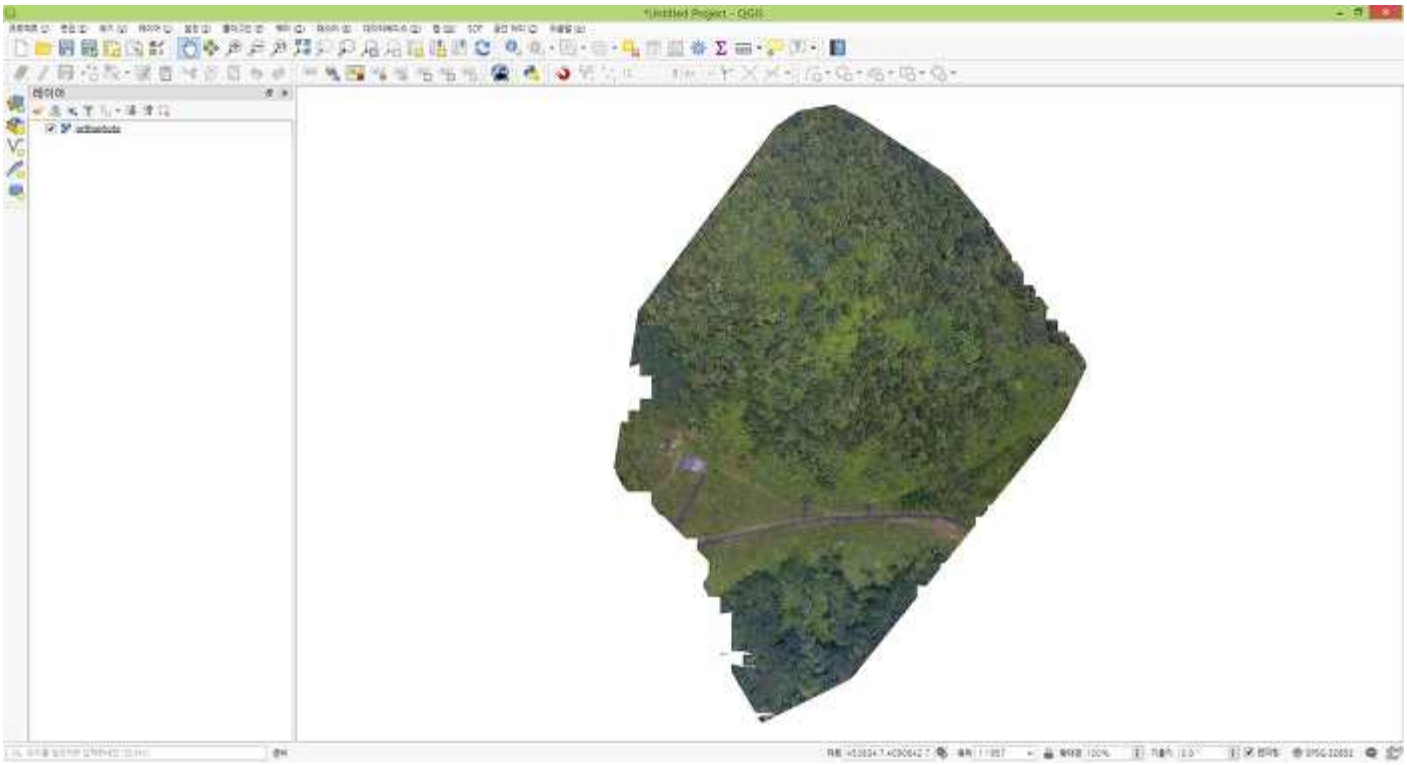
끝으로 아래와 같이 시스템 변수를 추가합니다. 이제 모든 설정이 끝났습니다!
변수 이름: OTB_APPLICATION_PATH
변수 값: C:₩OTB-6.6.1-Win64₩lib₩otb₩applications



IDLE에서 다음 호출이 정상 동작하면 이제 OTB 바인딩이 된 상태입니다.

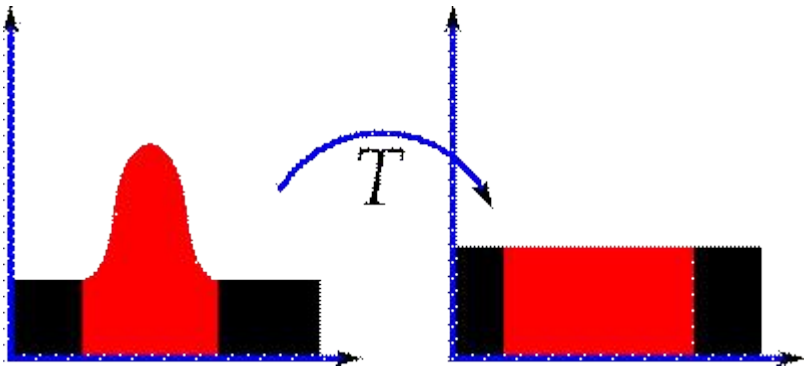
```
1import otbApplication
```

그럼, 간단한 예제를 통해 파이썬 3에서 OTB 기능을 호출해 보도록 할까요?!
소백산국립공원 주목군락 일원의 무인기 정사영상을 실습용 래스터로 사용하겠습니다.



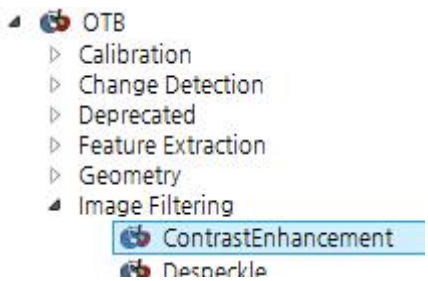
위 래스터에 OTB가 제공하는 대비 향상(Contrast Enhancement) 기법을 적용해 보겠습니다.
ContrastEnhancement - Contrast Enhancement
https://www.orfeo-toolbox.org/CookBook/Applications/app_ContrastEnhancement.html

대비 향상 기법은 히스토그램 균일화(histogram equalization) 알고리즘을 적용한 것입니다.
히스토그램 균일화는 이미지의 히스토그램이 특정 영역에 집중되어 있는 분포를, 골고루 분포하도록 하는 작업을 말합니다.

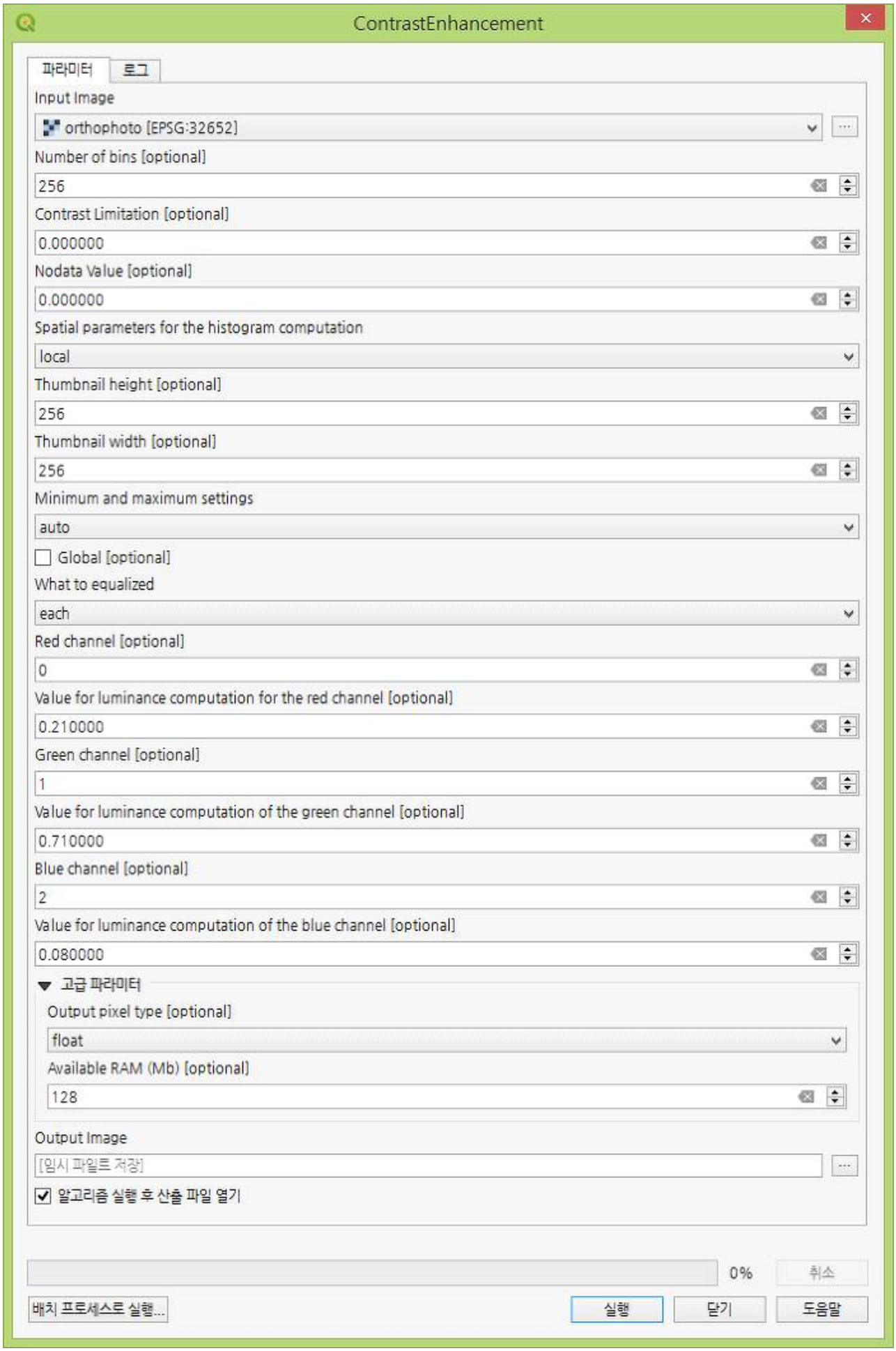


https://opencv-python.readthedocs.io/en/latest/doc/20.imageHistogramEqualization/imageHistogramEqualization.html

해당 기능은 'OTB >Image Filtering >ContrastEnhancement'를 통해 실행하실 수 있습니다.



실행 창은 아래와 같은데요, 이것을 Python 3.5에서 OTB를 호출하여 구동해 보고자 합니다.

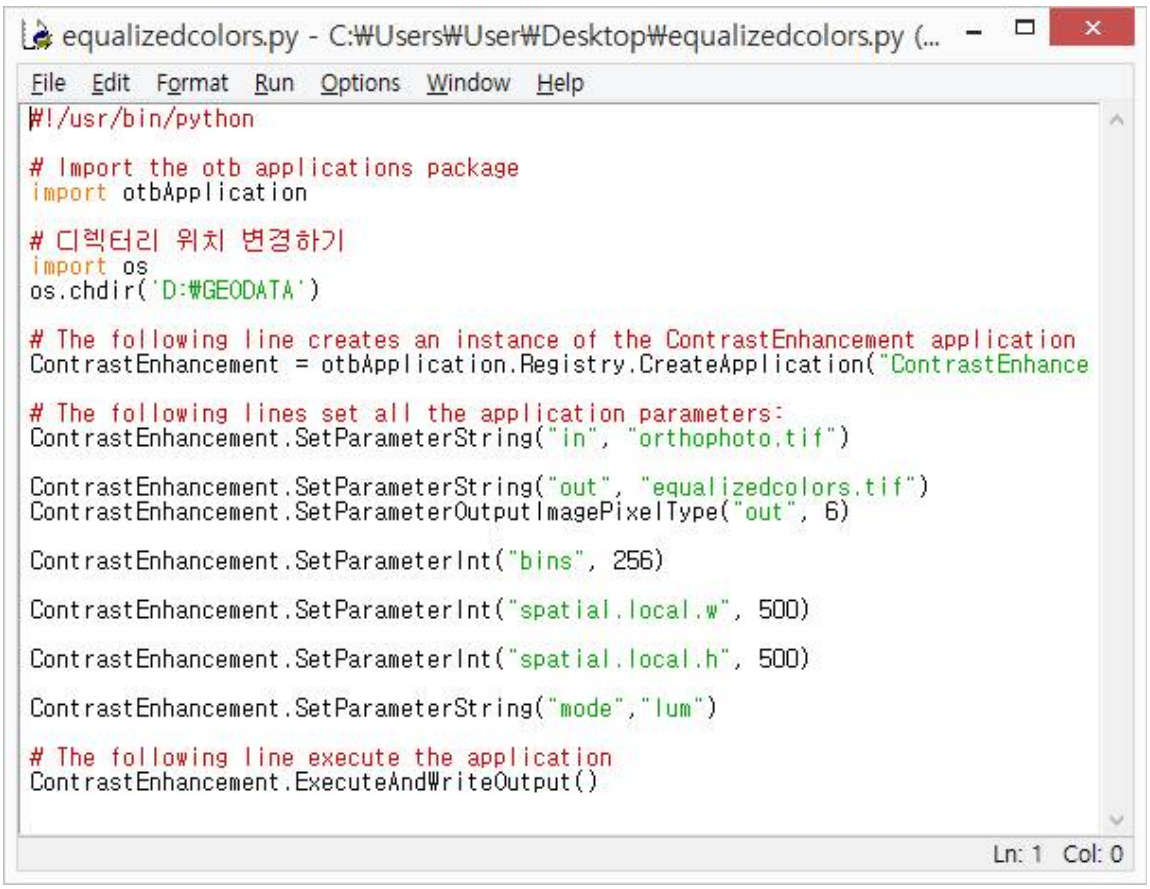


해당 코드는 직접 작성하실 필요가 없습니다. 각각 기능에 대한 예제 코드를 OTB CookBook에서 제공합니다.
Welcome to Orfeo ToolBox! | https://www.orfeo-toolbox.org/CookBook/

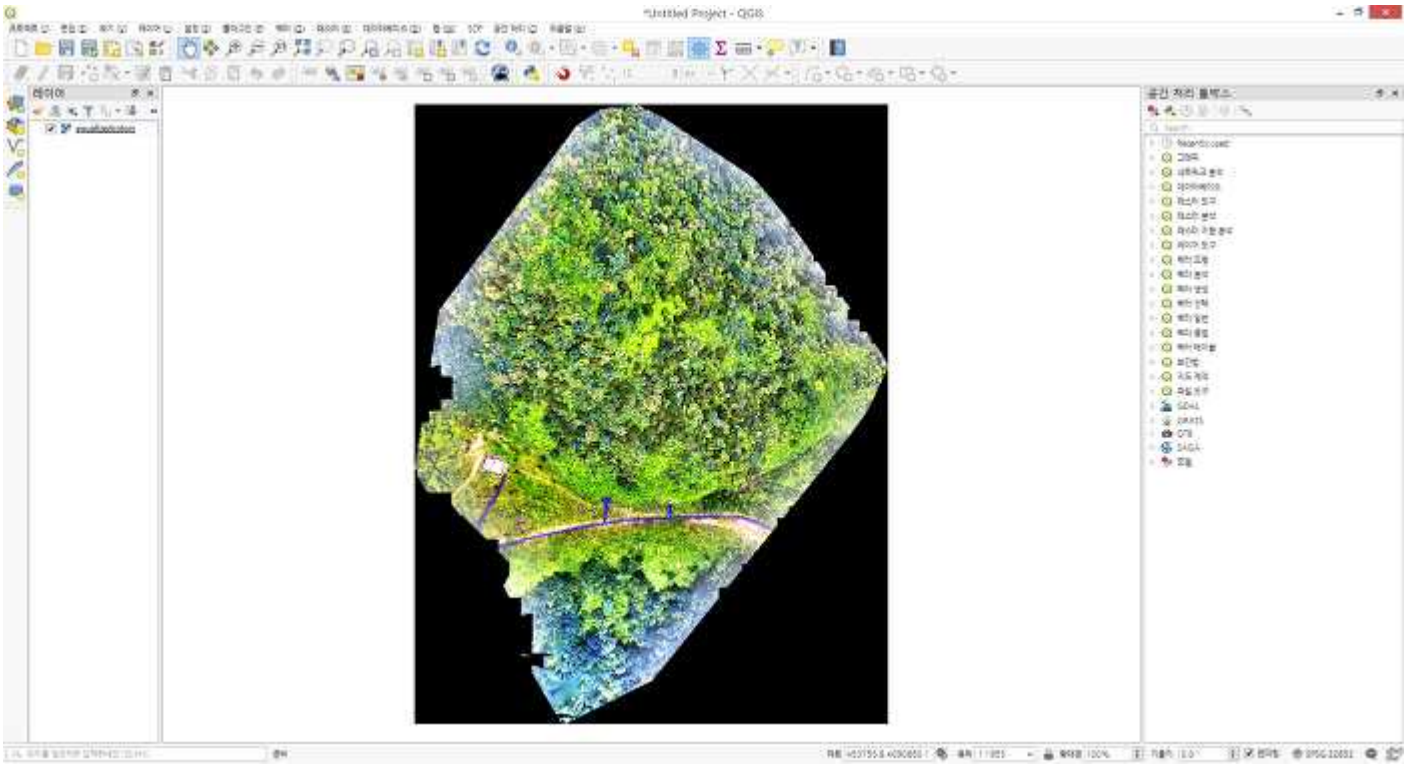
```
#!/usr/bin/python

# Import the otb applications package
import otbApplication

# The following line creates an instance of the ContrastEnhancement application
ContrastEnhancement = otbApplication.Registry.CreateApplication("ContrastEnhancement")

# The following lines set all the application parameters:
ContrastEnhancement.SetParameterString("in", "colours.tif")

ContrastEnhancement.SetParameterString("out", "equalizedcolors.tif")
ContrastEnhancement.SetParameterOutputImagePixelType("out", 6)

ContrastEnhancement.SetParameterInt("bins", 256)

ContrastEnhancement.SetParameterInt("spatial.local.w", 500)

ContrastEnhancement.SetParameterInt("spatial.local.h", 500)

ContrastEnhancement.SetParameterString("mode","lum")

# The following line execute the application
ContrastEnhancement.ExecuteAndWriteOutput()
```

해당 코드를 갈무리해서 아래와 같이 파이썬 파일을 생성합니다.
변경한 부분은 디렉터리 위치 변경과 입력 파일명이 전부입니다.

코드를 보시면 히스토그램 빈의 개수는 256개이고, 지역 히스토그램 계산을 위한
너비, 높이는 각각 500, 500, 모드는 휘도(luminanace)인 것을 확인하실 수 있습니다.

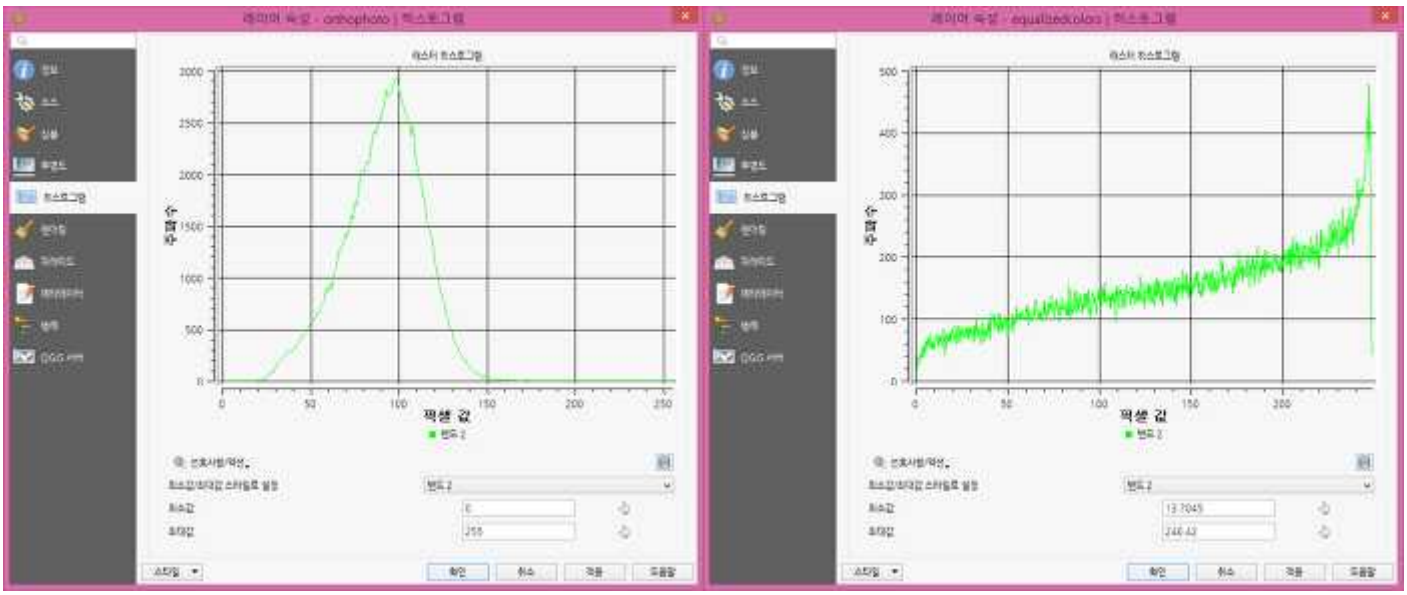


예제 코드를 약간 수정하여 아래와 같이 대비 향상이 처리되었습니다. 간단하죠?!



참고로, QGIS에서 대비 향상 전후 영상의 히스토그램을 비교해보실 수 있습니다.

※ 히스토그램은 해당 레이어 속성에서 '투명도 >NODATA 값'을 0으로 설정(하고 확인 클릭)한 후, '최소값/최대값 스타일로 설정 >밴드 2'로 선택하고 '선호사항/액션 >선택 밴드 보이기'를 클릭하시면 됩니다.



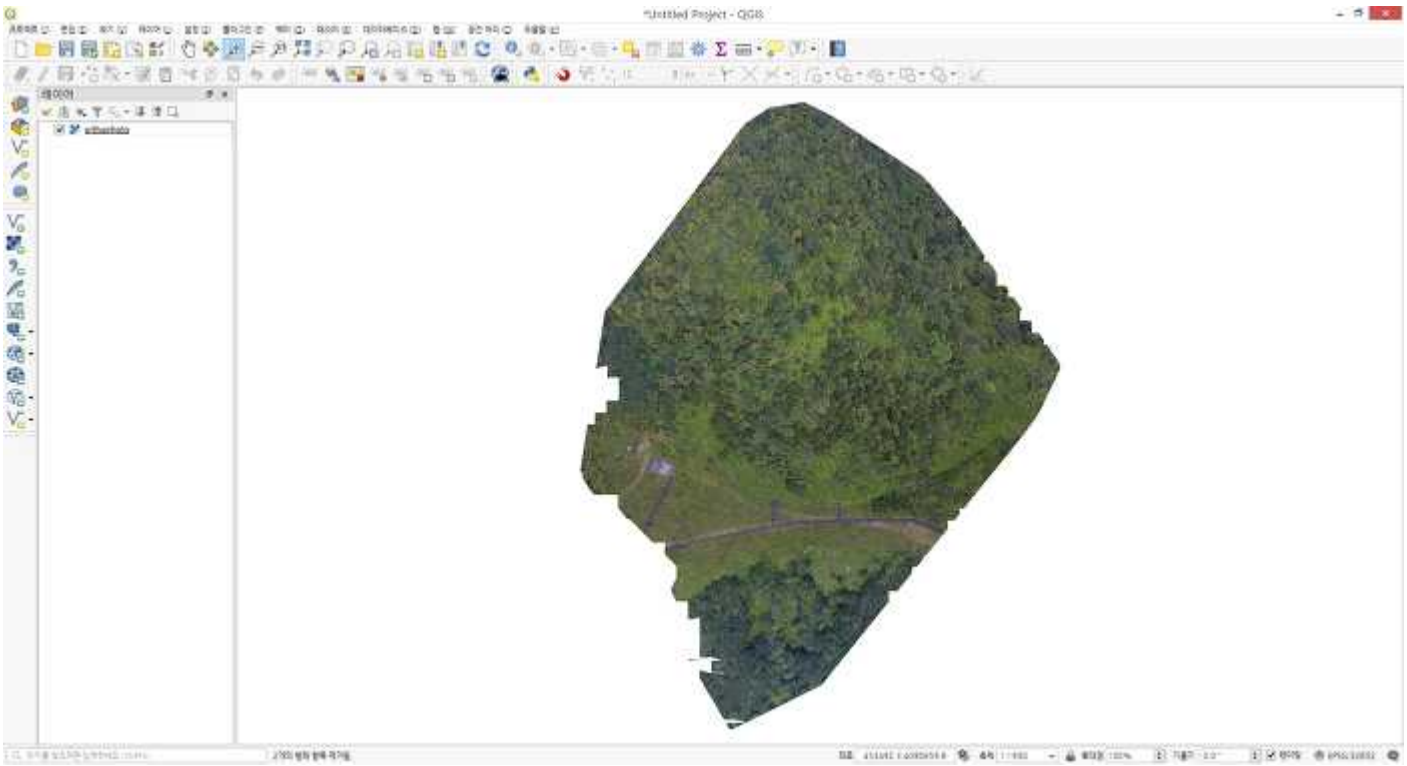
QGIS 3.4 Orfeo ToolBox(OTB) - (3) 무인기 영상 세그먼테이션(분할)

안녕하세요? 이번 글은 QGIS 3.4 Orfeo ToolBox를 통해 무인기 영상을 세그먼테이션하는 방법을 정리해 보겠습니다.
영상 세그먼테이션은 객체기반 영상분석(Object-Based Image Analyis; OBIA)을 수행할 때 그 첫 번째 단계라고 볼 수 있습니다.

무인기 영상과 같이 수 cm 공간해상도를 제공하는 공간정보는 각각의 화소에 기반한 분류보다는 화소값들을 일정 기준으로 그루핑한 객체에 기반하여 처리하는 것이 유리할 수 있겠죠?!
그럼, 영상 세그먼테이션을 어떻게 처리하는지 한 번 살펴보겠습니다.

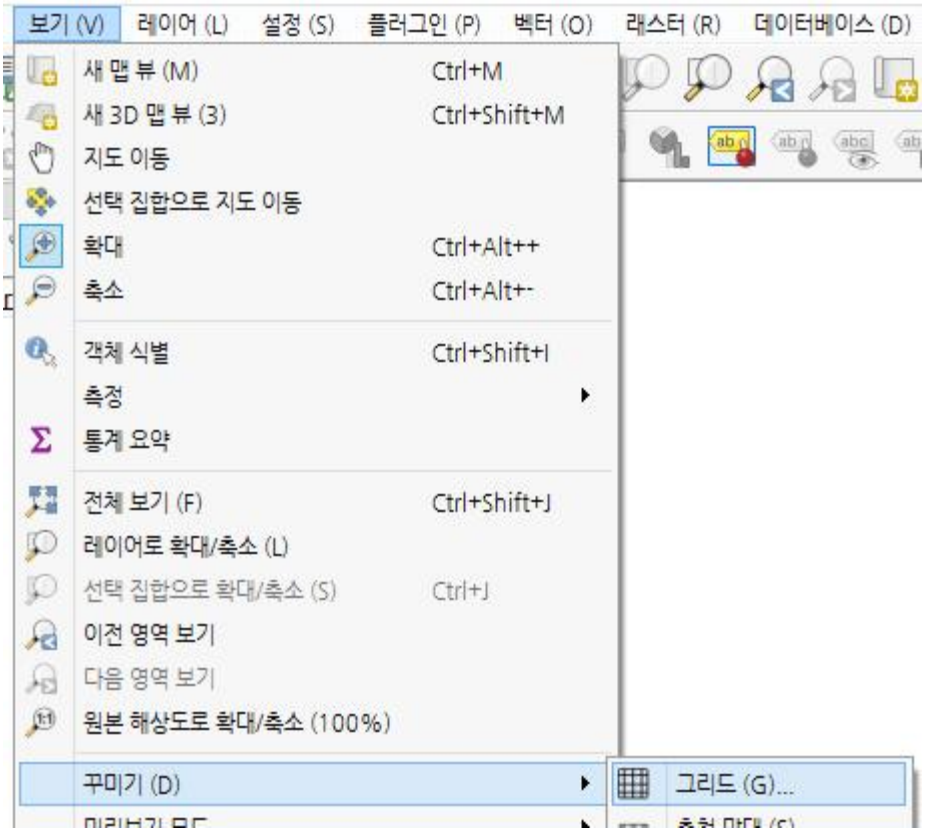
이미지 출처: Image Classification Techniques in Remote Sensing
https://gisgeography.com/image-classification-techniques-remote-sensing/

먼저 실습을 위해 소백산국립공원 주목군락 일원을 촬영한 무인기 영상을 레이어 추가했습니다.

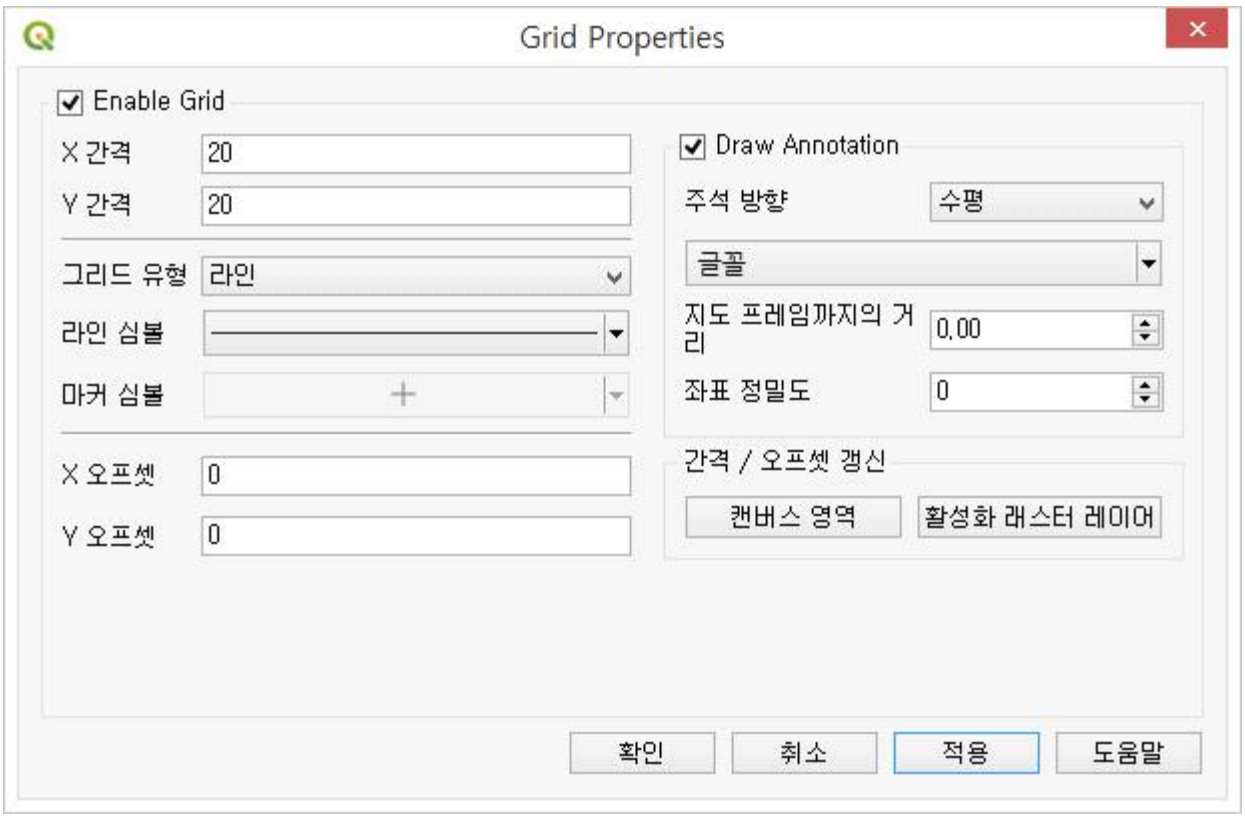


전체 데이터를 처리하는데는 일정 시간이 소요되기 때문에, 여기서는 일부 지역을 잘라내서 사용하겠습니다.

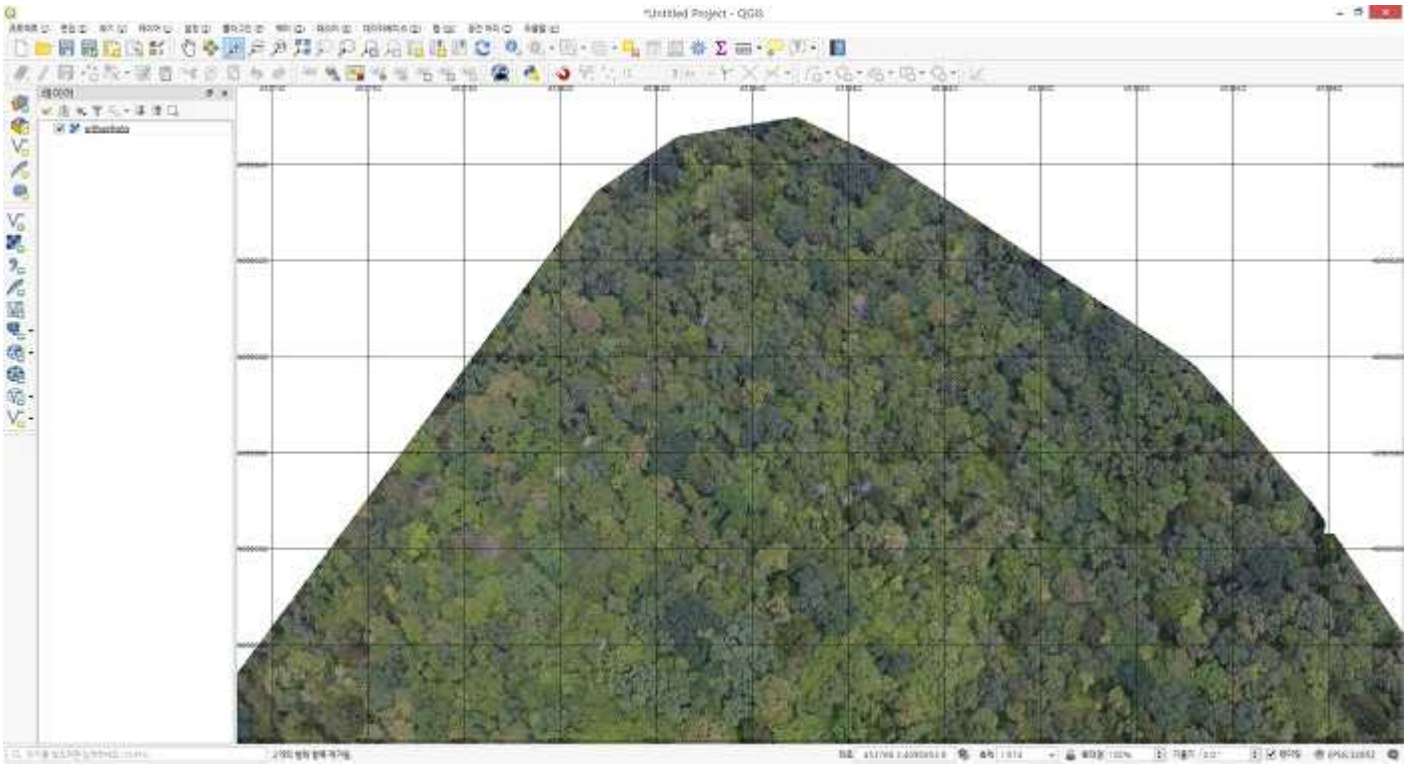
원하는 영역을 깔끔하게 추출하기 위해 상단 메뉴에서 '보기 >꾸미기 >그리드'를 실행합니다.



'Grid Properties' 창이 실행되는데요, Enable Grid를 체크하고 XY 간격을 지정해 줍니다. 여기서는 각각 20m로 지정했습니다. Draw Annotation을 체크하신 후 원하는 글꼴(폰트, 글자크기)과 좌표 정밀도(소수점이하 자릿수)를 적당하게 지정하였습니다.

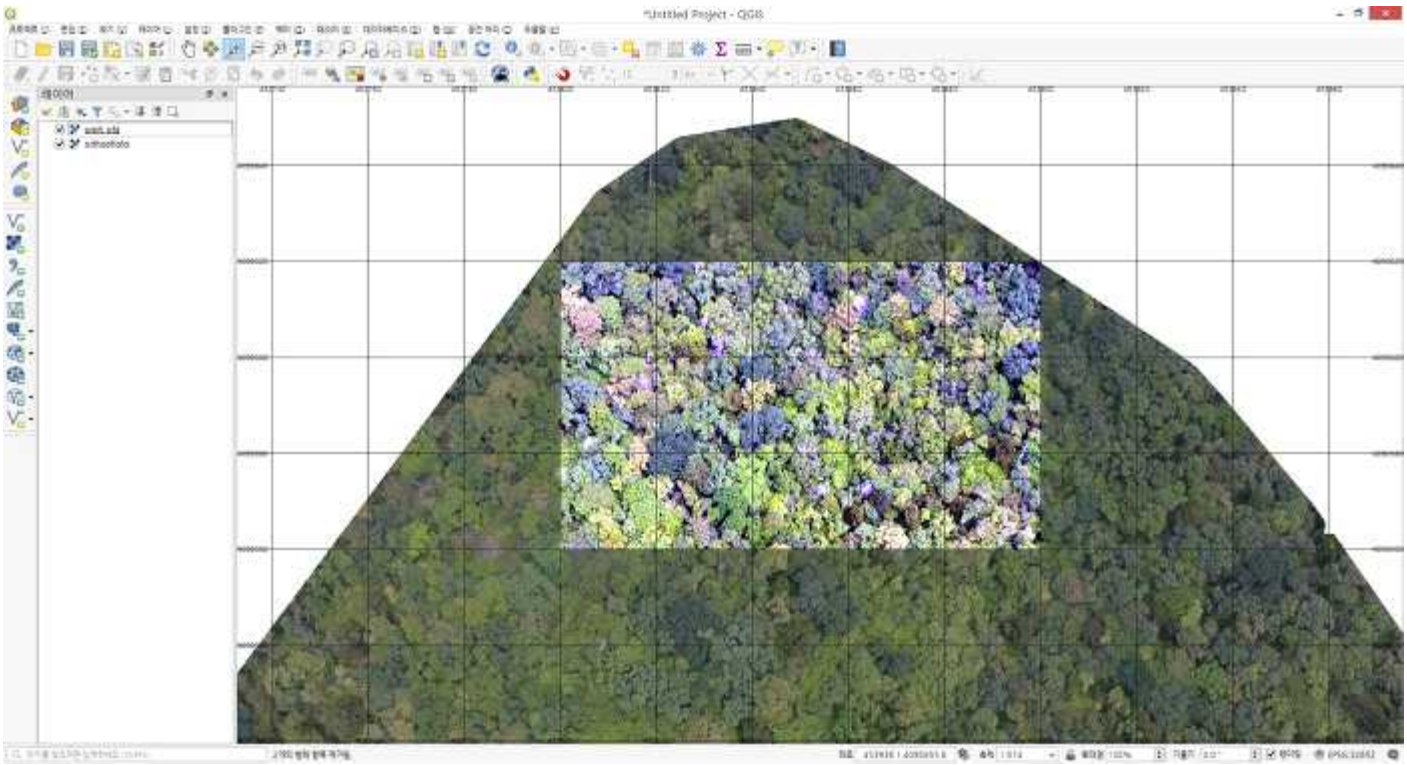


아래와 같이 그리드가 표시되었는데요, 이 그리드를 참고하여 적정한 영역을 잘라내겠습니다.

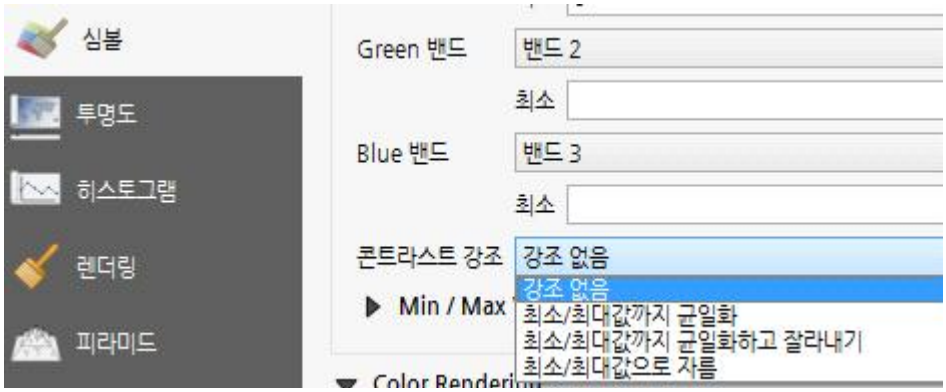


OTB에서 범위를 통해 영상을 잘라내는 방법은 아래 글을 참조하시면 됩니다.
QGIS 3.4 Orfeo ToolBox(OTB): SRTM 타일 다운로드 및 처리 | http://blog.daum.net/geoscience/1323

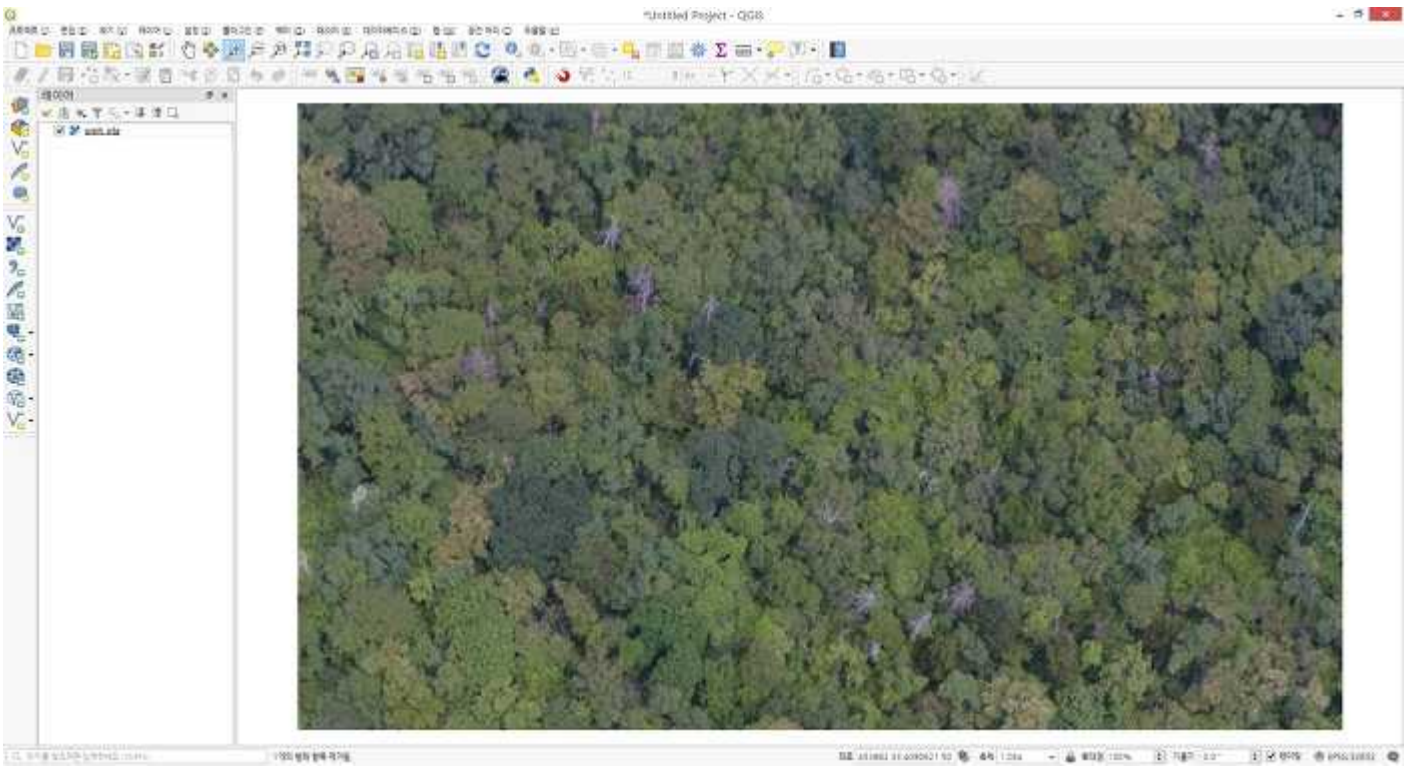
저는 아래와 같이 100x60m 영역을 잘라냈습니다.



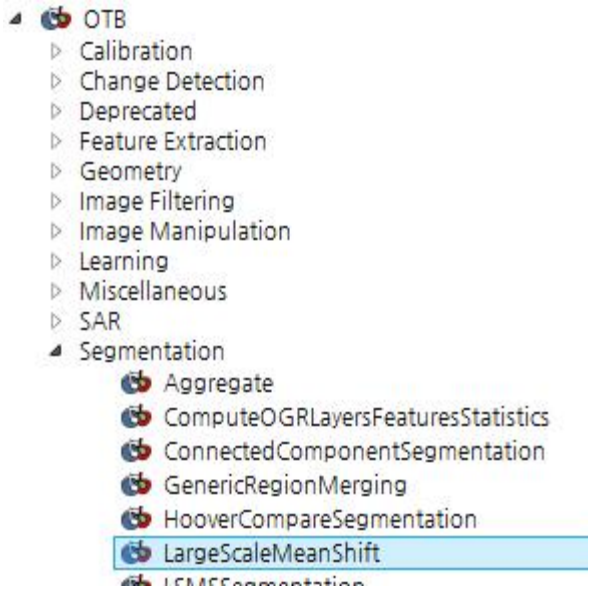
콘트라스트 강조로 인해 잘려진 영상의 색과 색 대비가 커졌을 텐데요,
원치 않으시면 레이어 속성에서 '심볼 >콘트라스트' 강조에서 '강조 없음'을 선택해 주시면 됩니다



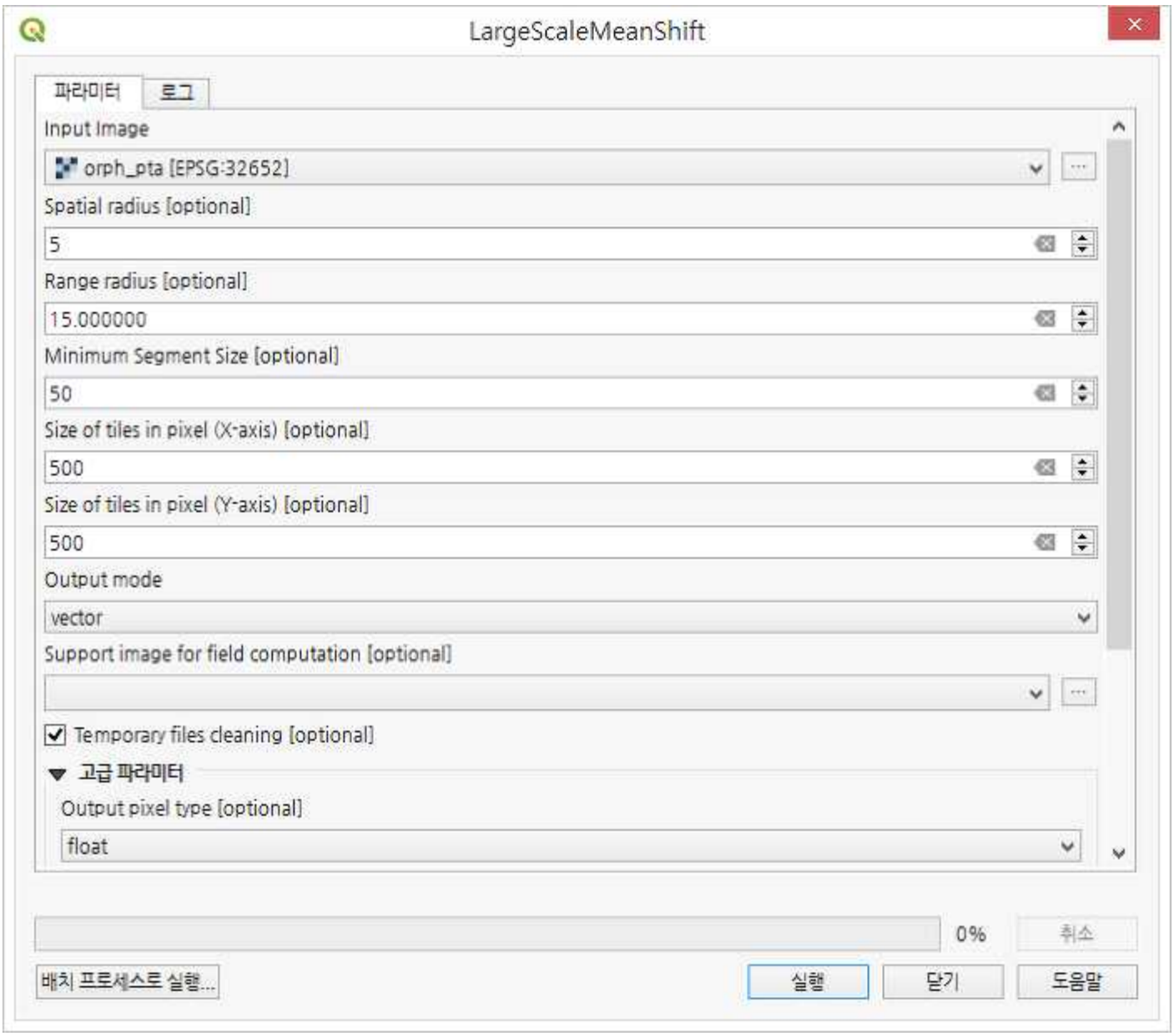
자, 이제 이렇게 잘려진 영상에 세그먼테이션을 적용해 보겠습니다.



OTB에서는 대용량 공간정보에 민 시프트 알고리즘을 적용하실 수 있는데요, 'OTB >Segmentation >LargeScaleMeanShift'를 실행하시면 됩니다.
MeanShift 알고리즘에 관해서는 다크프로그래머 님의 다음 글을 추천 드립니다: [영상추적#1] Mean Shift 추적 | http://darkpgmr.tistory.com/64



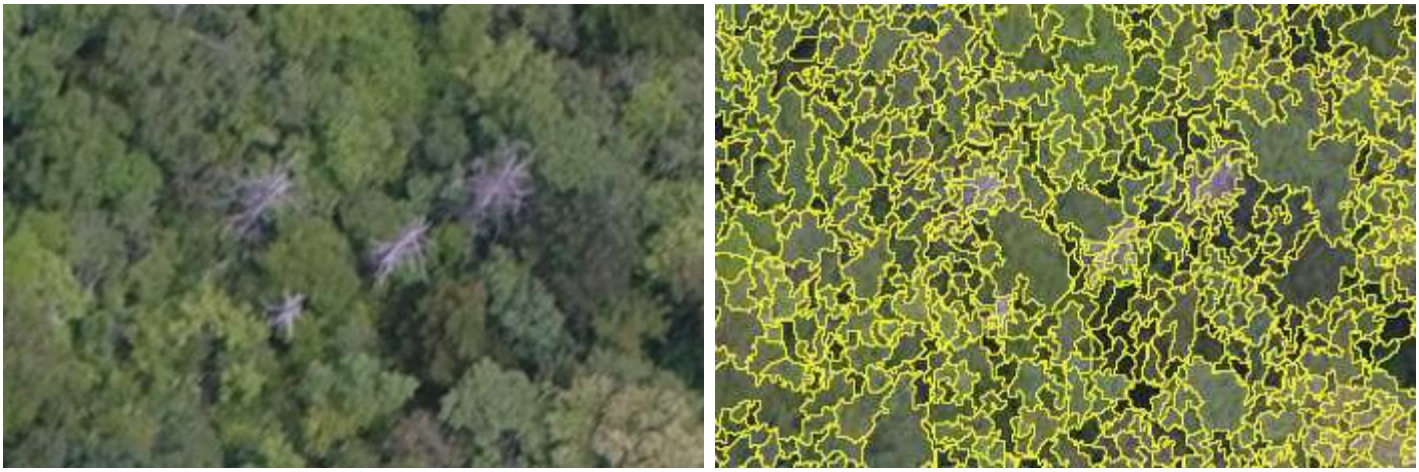
LargeScaleMeanShift 실행 창은 아래와 같습니다. 일단 Input Image에 잘려진 무인기 영상을 선택해 봅니다.



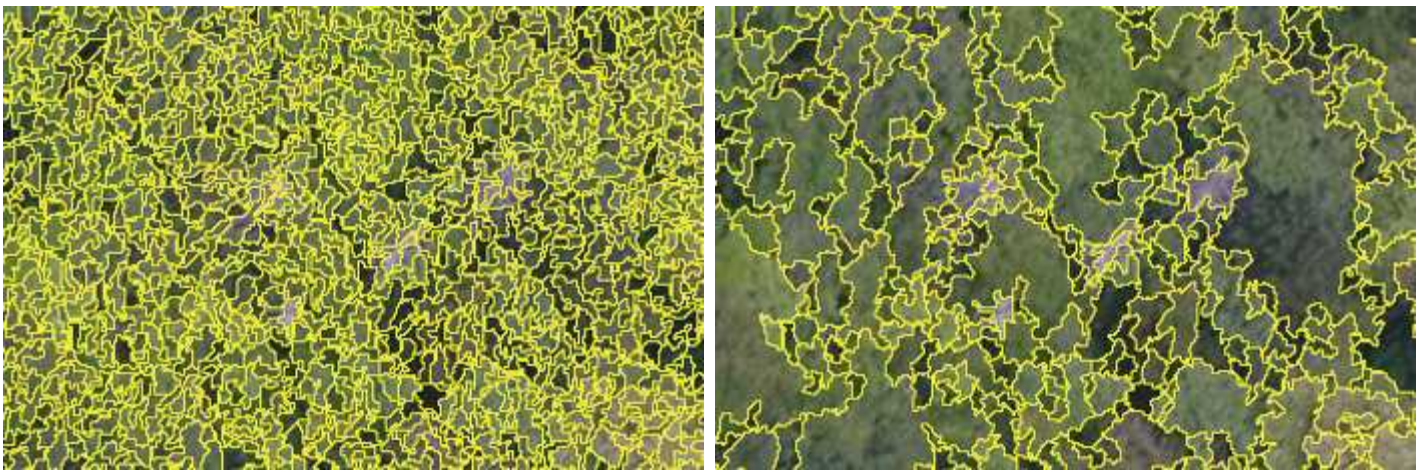
각각의 매개변수를 간략히 살펴볼까요?! 먼저 눈의 띄는 변수로 Spatial radius와 Range radius가 있습니다.
Spatial radius는 평균을 계산할 공간 인접지역의 반경을 정의합니다. 이 값이 커지면 결과는 더 평활해지나 더 많은 시간이 소요됩니다.

Range radius는 분광 신호 유클리드 거리(방사측정 단위로)의 임계값에 해당합니다. 이 값이 클수록 엣지 보존이 적어지고 값이 작을수록 노이즈 평활화가 적어집니다. 변수는 처리 시간에 영향을 미치지 않습니다.

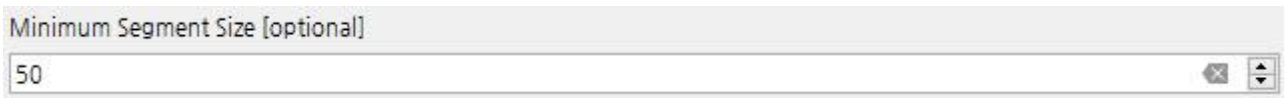
아래 세그먼테이션은 range radius를 15로 적용한 결과입니다.



동일 영상에 대해 range radius를 5로 적용한 결과는 좌측, 25로 적용한 결과는 우측과 같습니다. 매개변수 적용에 따라 차이가 있죠?!

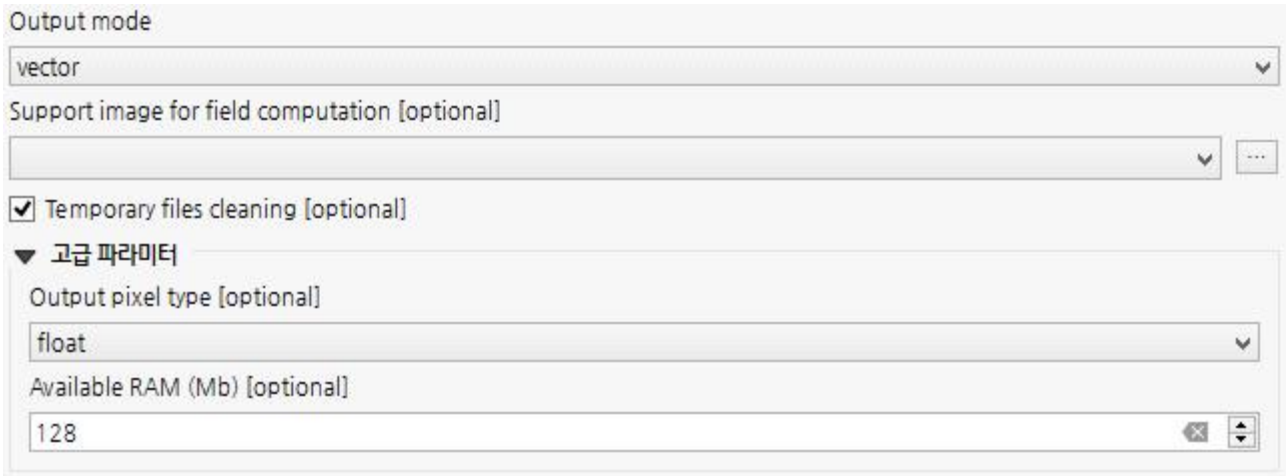


Minimum Segment Size는 분할 후에 각각의 세그먼트 크기가 이 기준보다 작은 경우 가장 가까운 분광신호를 가진 세그먼트에 병합합니다.

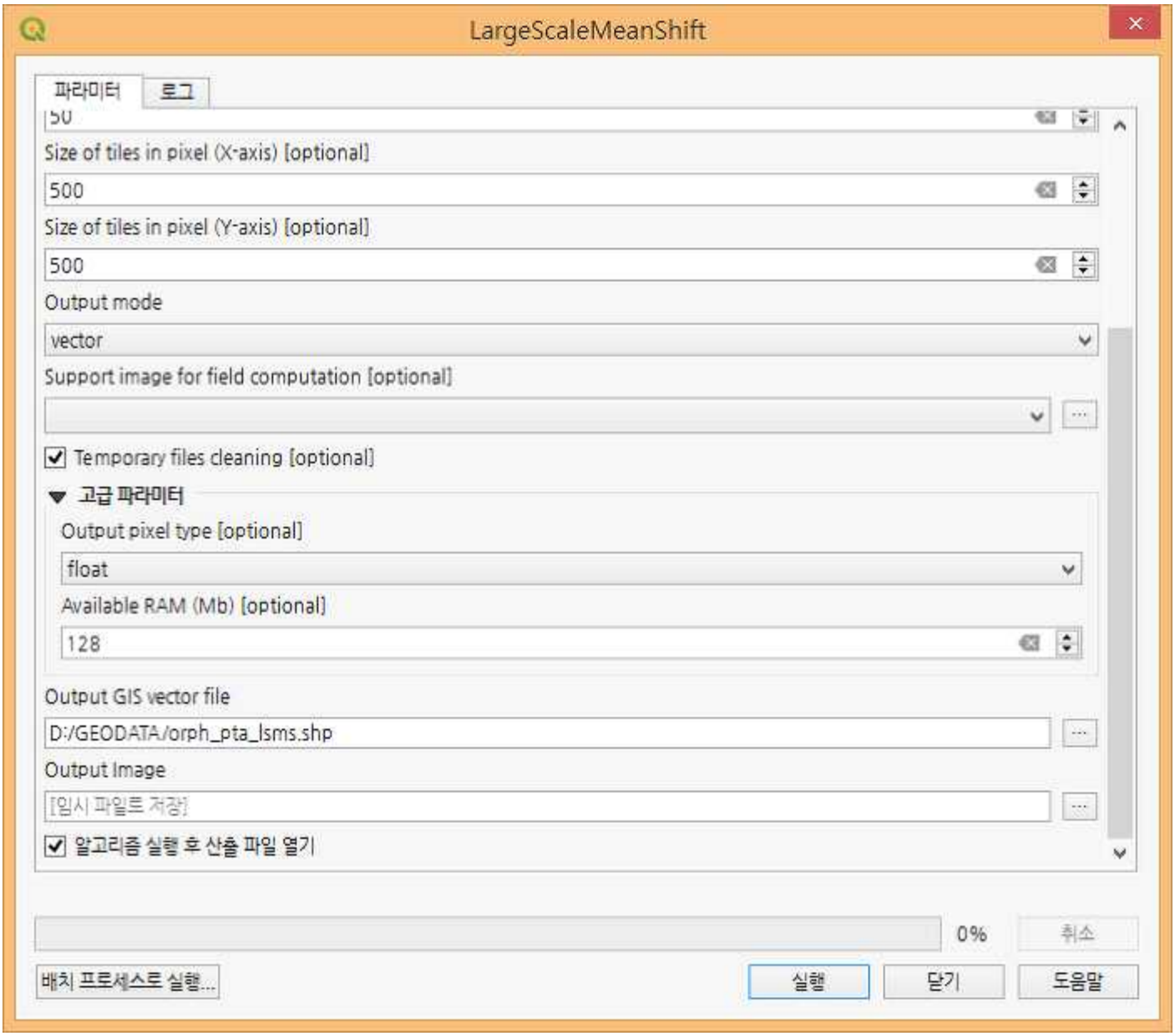


아래 값은 화소 단위의 타일 크기(X축, Y축)을 설정합니다.

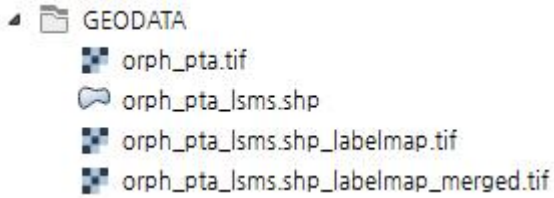
출력 모드는 vector와 raster가 있습니다.
출력 모드가 vector인 경우에는 Support image for field computation이 활성화됩니다.
이때 레이어를 선택하지 않으면 입력이미지의 각 밴드별 평균, 표준편차가 계산됩니다.



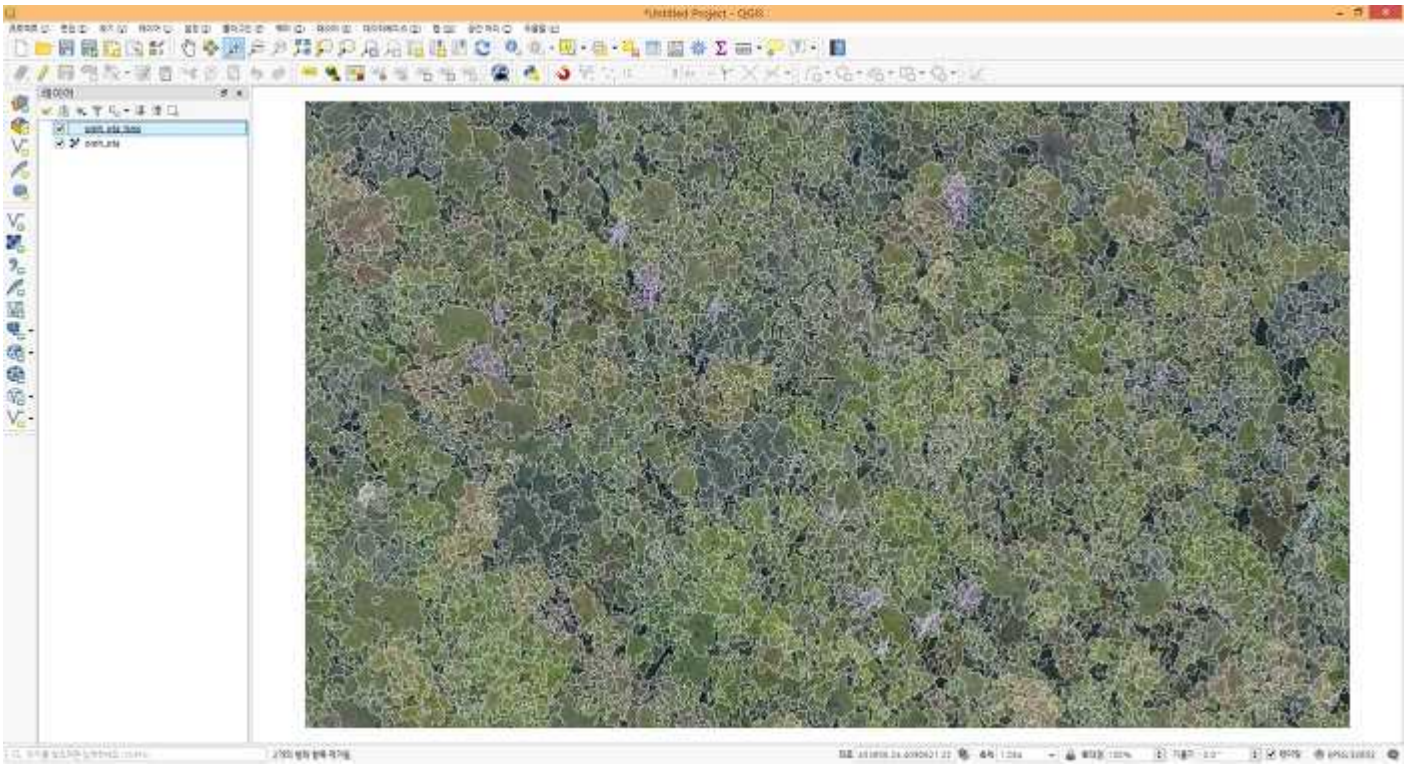
자, 이제 결과를 확인해볼까요?!



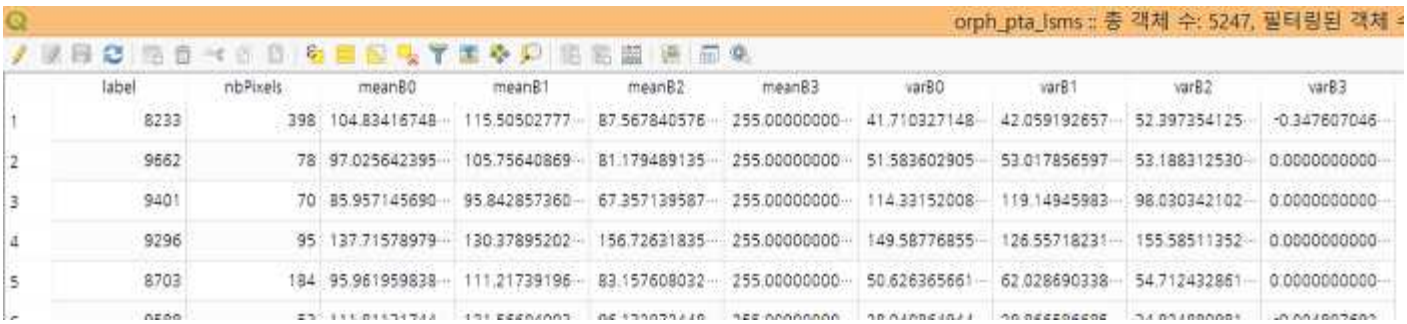
아래와 같이 3개 파일이 생성되었습니다.



먼저, 무인기 영상에 세그먼테이션 벡터를 중첩한 화면입니다.

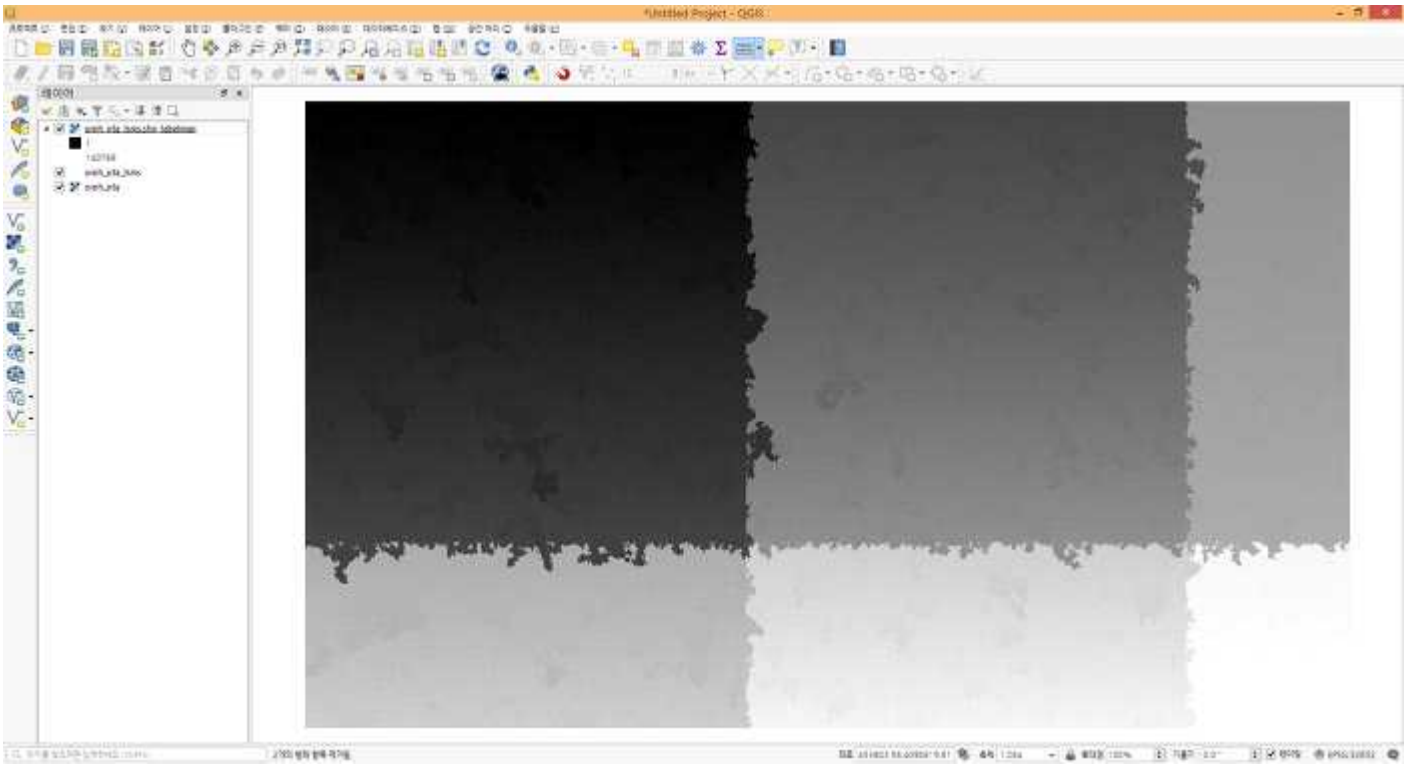


세그먼테이션 속성에는 라벨, 화소개수, RGB 밴드별 평균과 표준편차 값이 자동 추가됩니다.

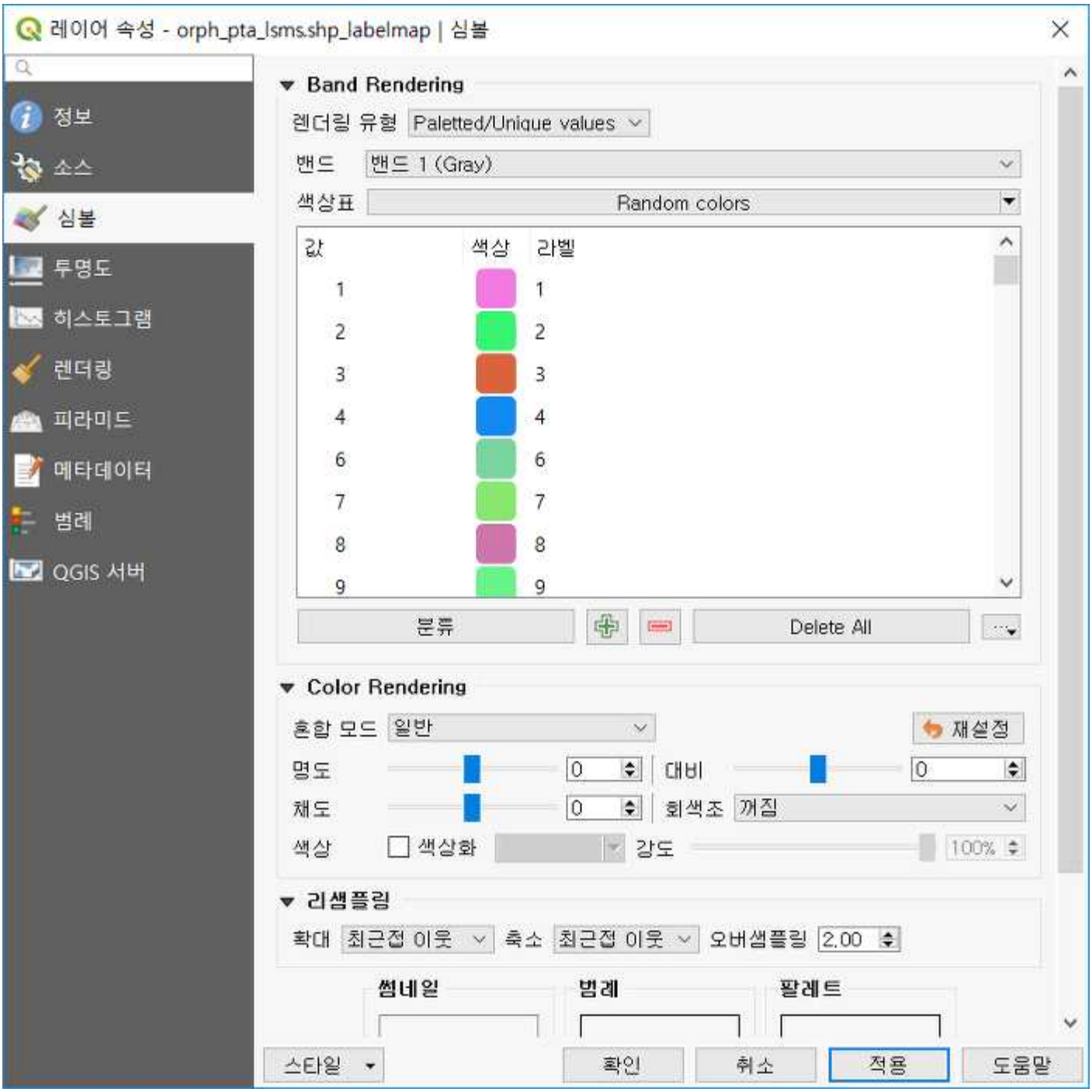


	label	nbPixels	meanB0	meanB1	meanB2	meanB3	varB0	varB1	varB2	varB3
1	8233	398	104.83416748...	115.50502777...	87.567840576...	255.00000000...	41.710327148...	42.059192657...	52.397354125...	-0.347607046...
2	9662	78	97.025642395...	105.75640869...	81.179489135...	255.00000000...	51.583602905...	53.017856597...	53.188312530...	0.0000000000...
3	9401	70	85.957145690...	95.842857360...	67.357139587...	255.00000000...	114.33152008...	119.14945983...	98.030342102...	0.0000000000...
4	9296	95	137.71578979...	130.37895202...	156.72631835...	255.00000000...	149.58776855...	126.55718231...	155.58511352...	0.0000000000...
5	8703	184	95.961959838...	111.21739196...	83.157608032...	255.00000000...	50.626365661...	62.028690338...	54.712432861...	0.0000000000...

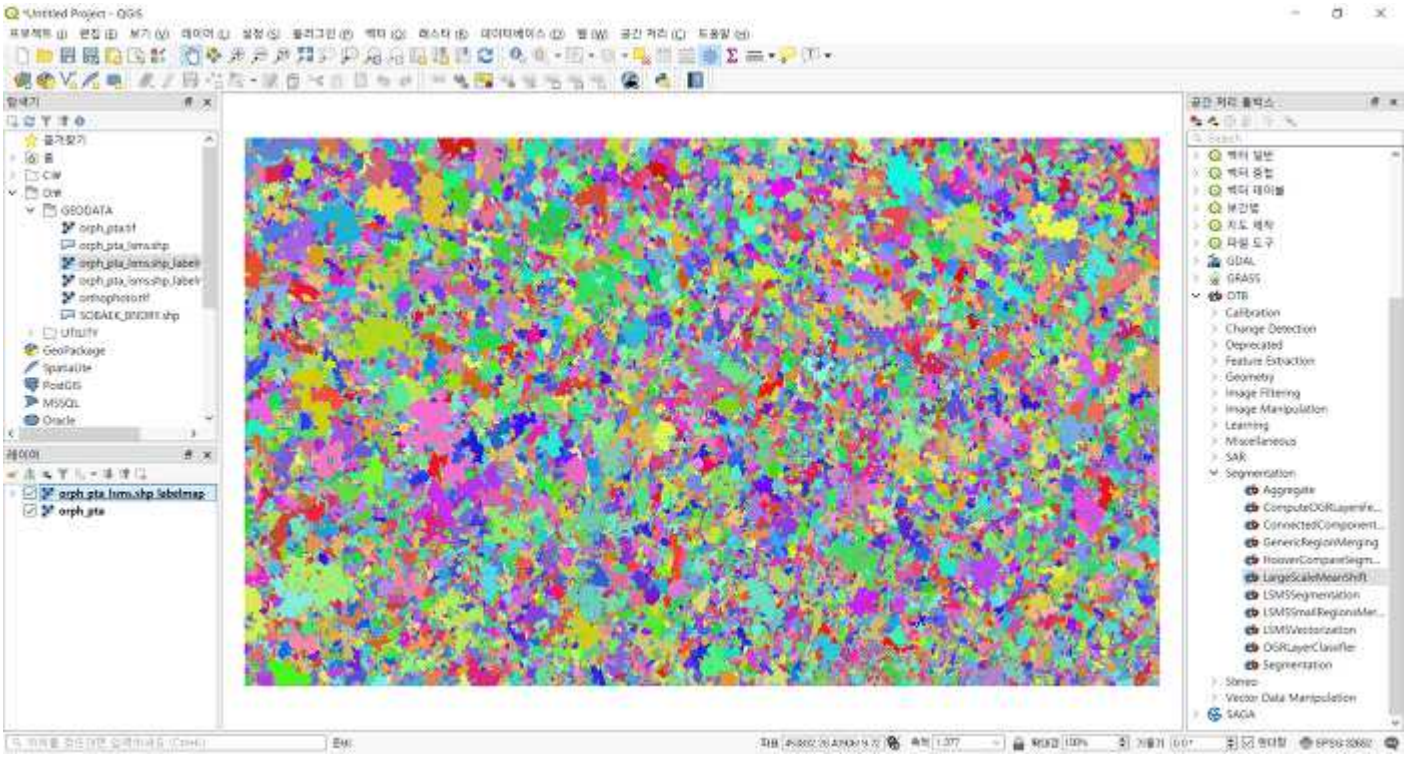
labelmap은 민시프트 세그먼테이션을 적용한 래스터 영상입니다.



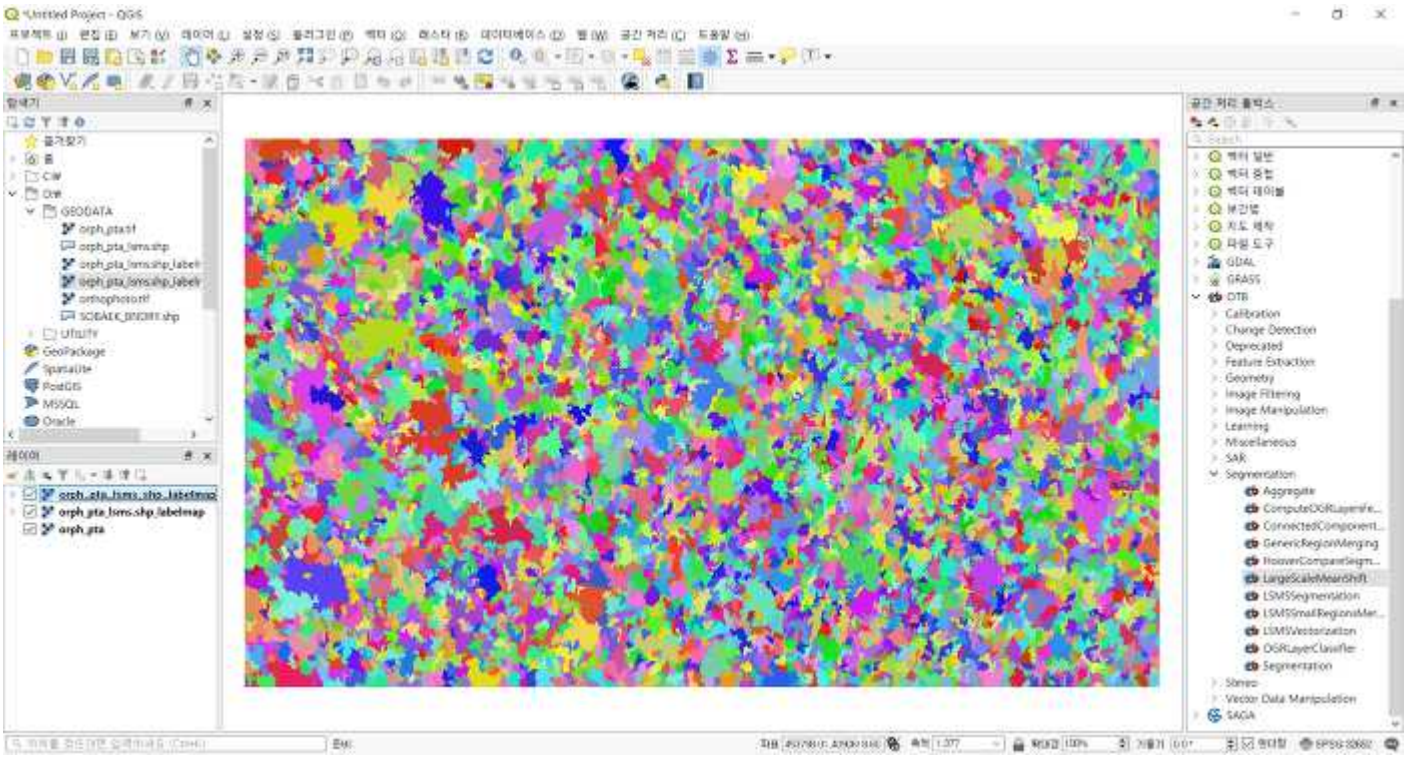
세그먼트를 확연히 구분하기 위해 '레이어 속성 >심볼'에서 렌더링 유형을 'Paletted/Unique values'로 변경하고 색상표 Random colors를 적용합니다.



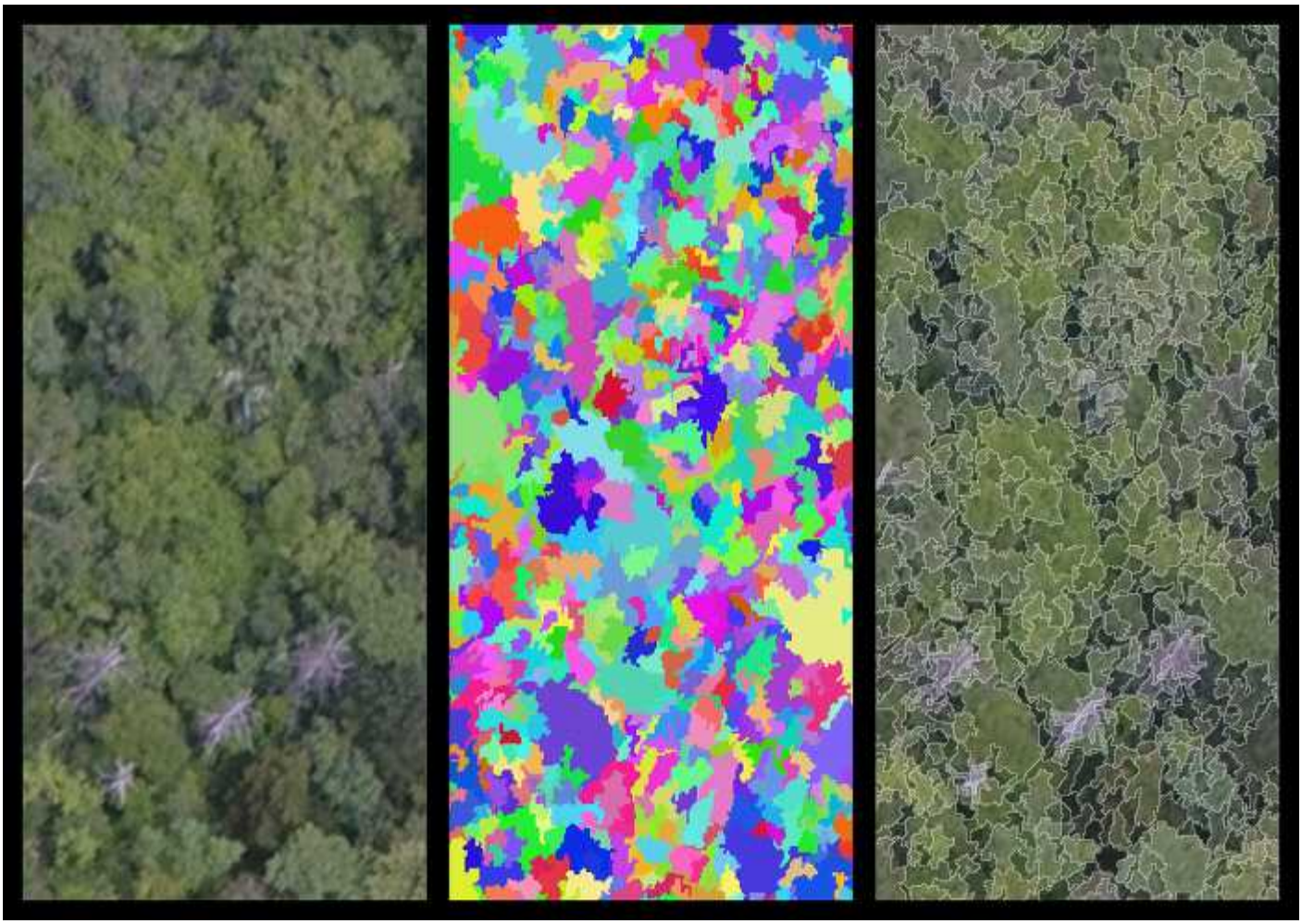
아래와 같이 각각의 세그먼트가 서로 다른 색상으로 표현됩니다.



Minimum Segment Size 설정에 따라 50보다 작은 세그먼트들을 인근에 병합한 영상입니다.



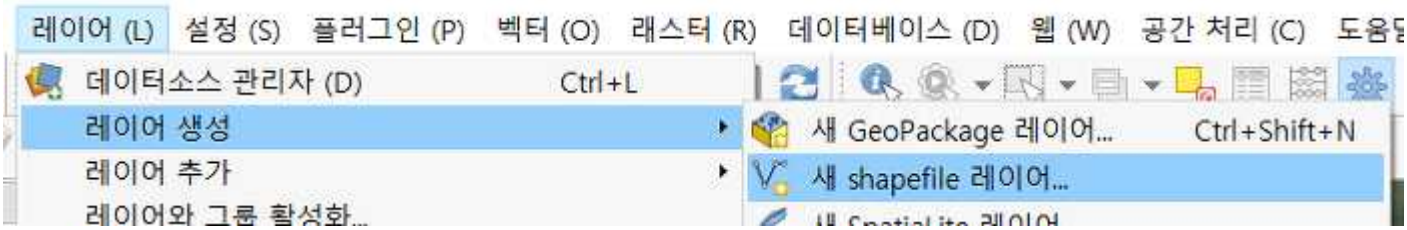
동일 지역에 대하여 무인기 정사영상, 세그먼테이션 래스터, 세그먼테이션 벡터와 정사영상 중첩 결과를 비교해 봤습니다. 크게 어렵지 않죠?!



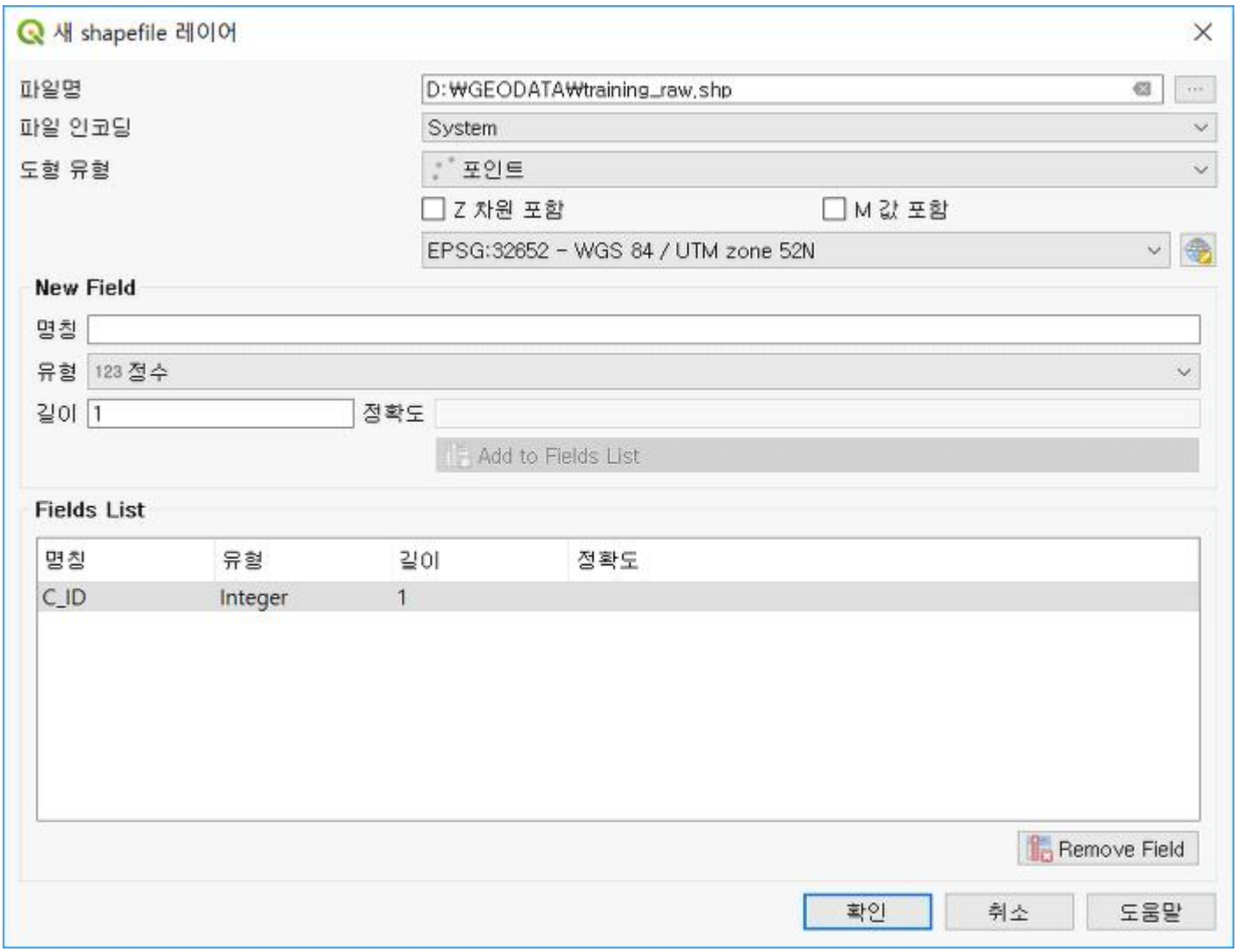
QGIS 3.4 Orfeo ToolBox(OTB) - (4) 훈련데이터 만들기

안녕하세요? 이번 글에서는 훈련 데이터를 만드는 과정을 정리해 보겠습니다.

먼저, 폴리곤 파일을 하나 생성하겠습니다. 화면 상단에서 '레이어 >레이어 생성 >새 Shapefile 레이어'를 클릭합니다.

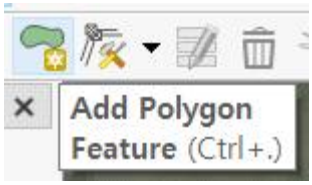


아래와 같이 training_raw.shp 파일을 하나 생성하고 C_ID라는 정수형 필드를 하나 추가했습니다.

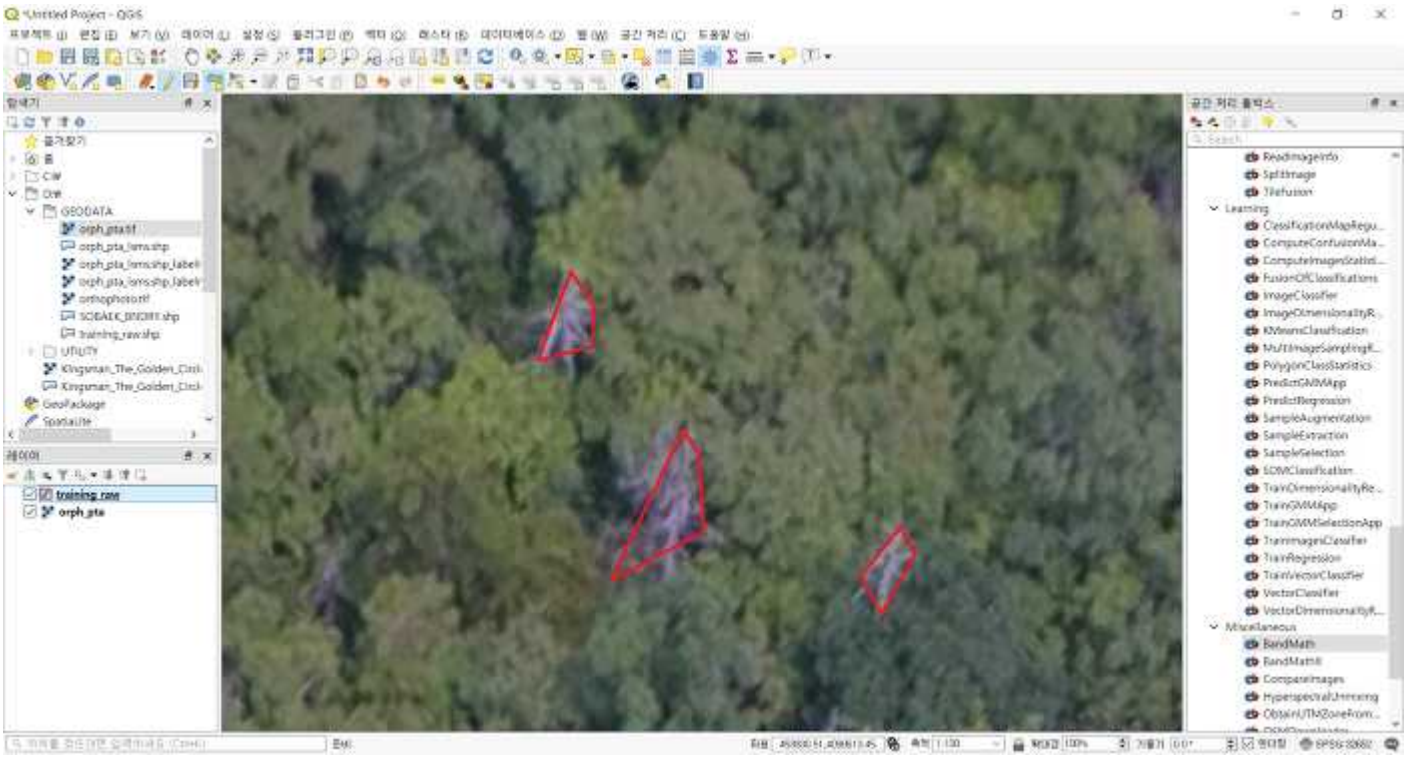


디지타이즈 작업 툴바에서 '편집 모드 켜고끄기' 버튼을 클릭하면 현재 선택된 레이어를 편집할 수 있습니다.

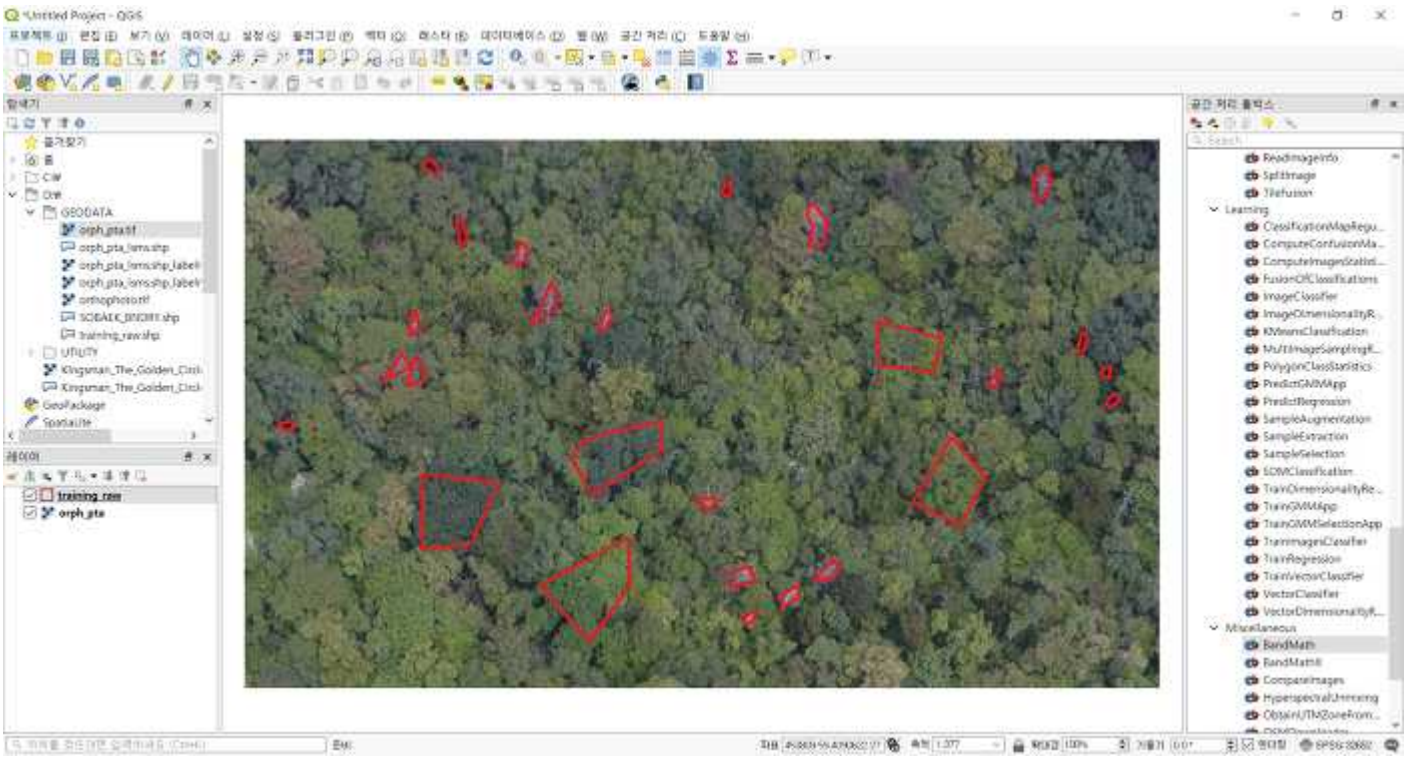
Add Polygon Feature를 클릭하고,



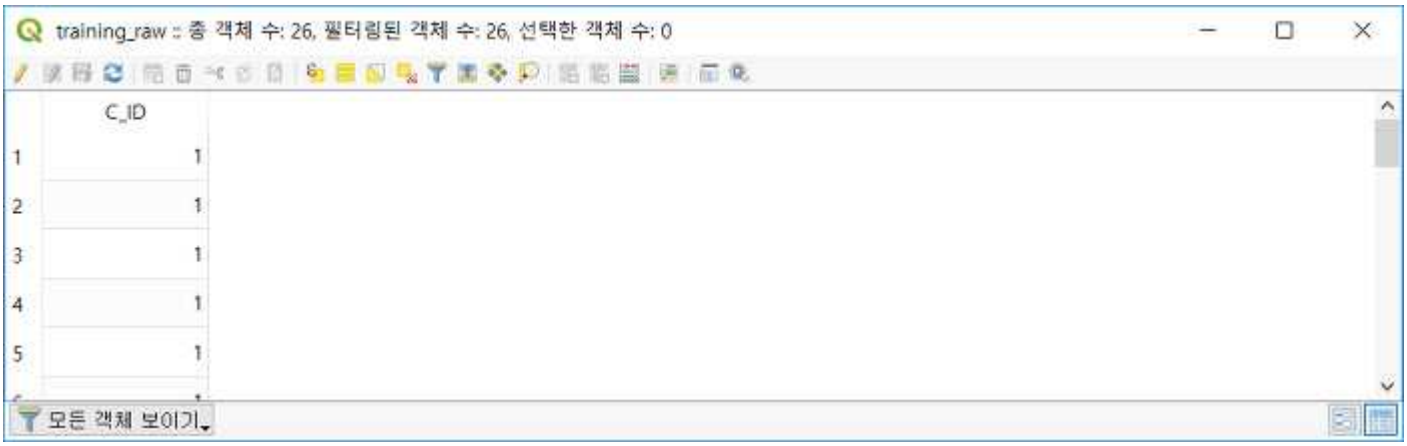
화면상에서 훈련데이터로 사용하고자 하는 영역을 지정해 줍니다. C_ID는 1: Dead tree, 2: Live tree, 3: Shadow로 정의하겠습니다.



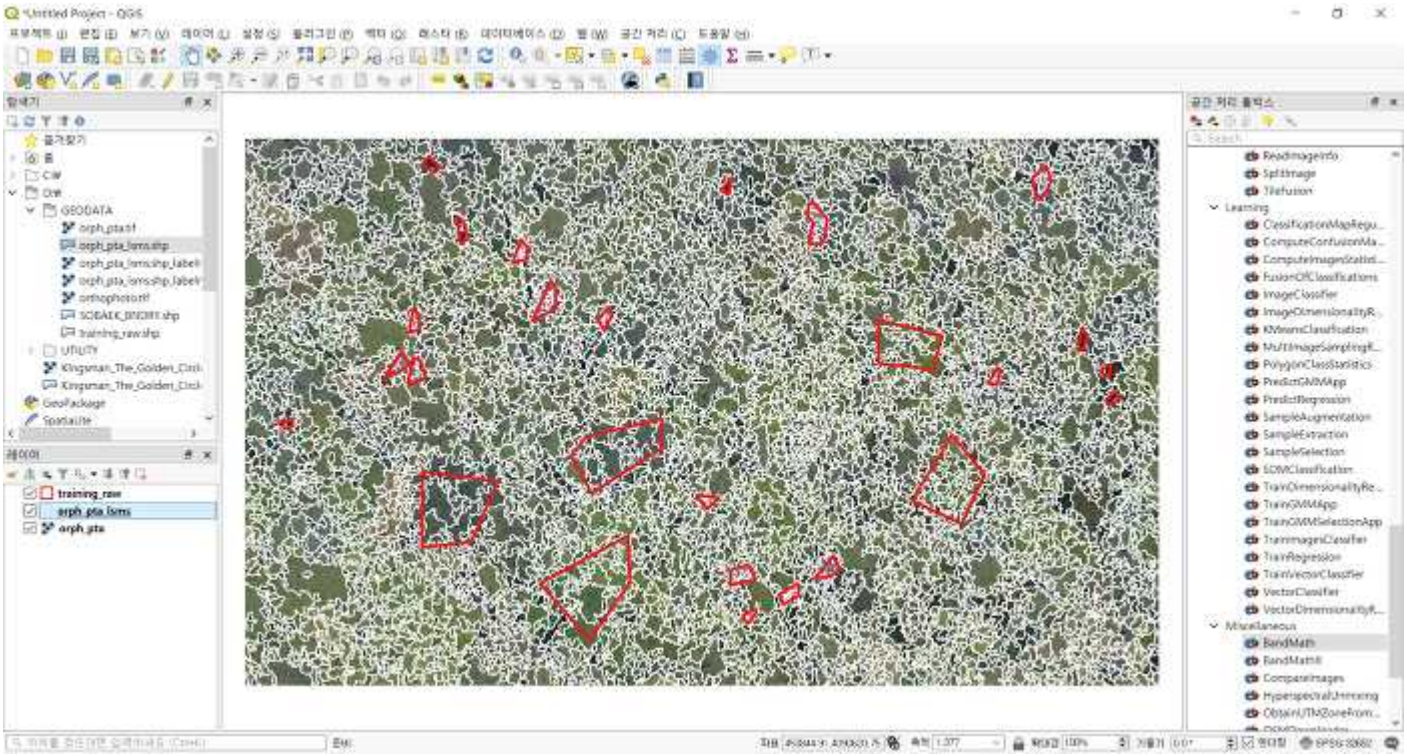
아래와 같이 분석가의 판단 하에 훈련데이터 영역을 지정해 줍니다.



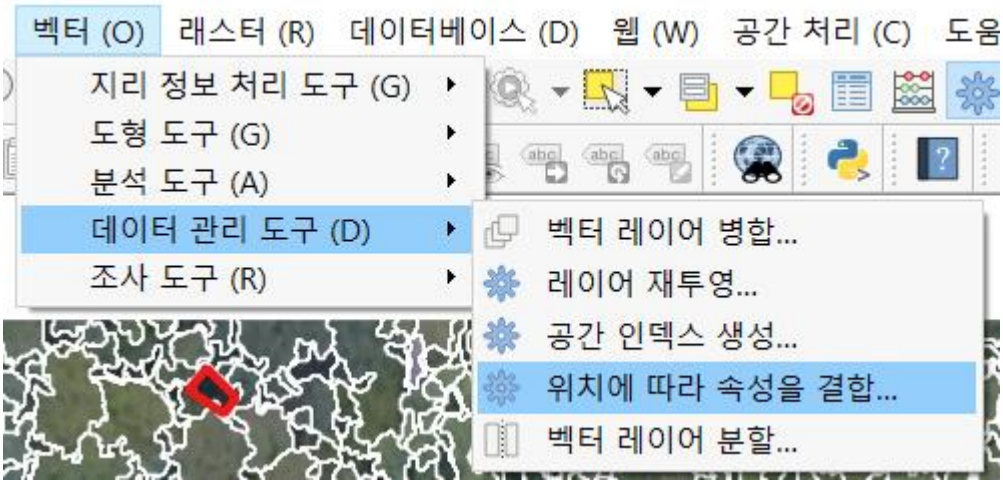
해당 데이터의 속성을 보시면 C_ID에 정의된 값이 들어있는 상태를 확인하실 수 있습니다.



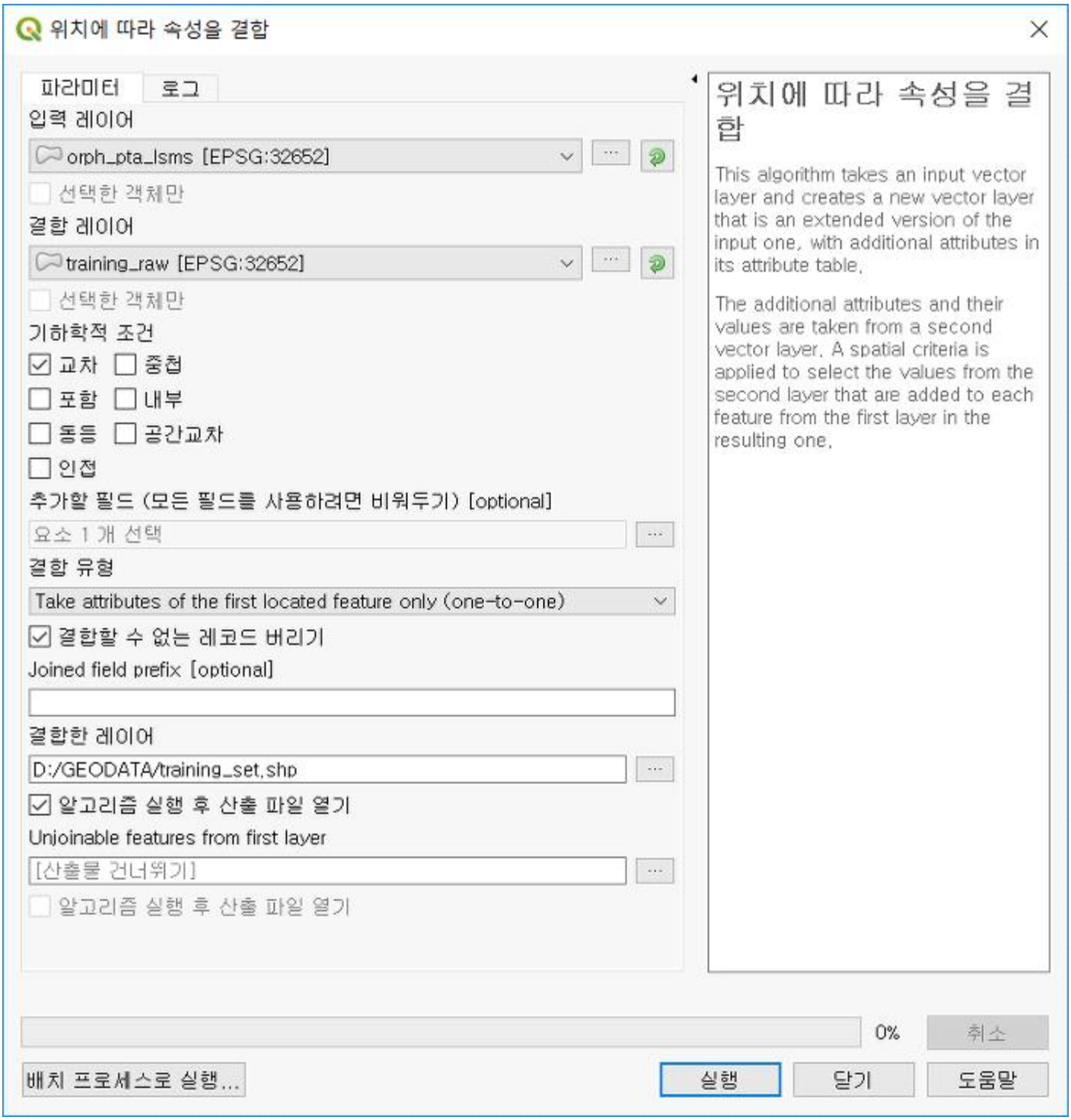
앞서 처리한 세그먼테이션 폴리곤을 레이어 추가합니다. 이제 훈련데이터 영역에 해당하는 세그먼트를 선택하고 해당 C_ID 값을 이식해 주면 되겠습니다.



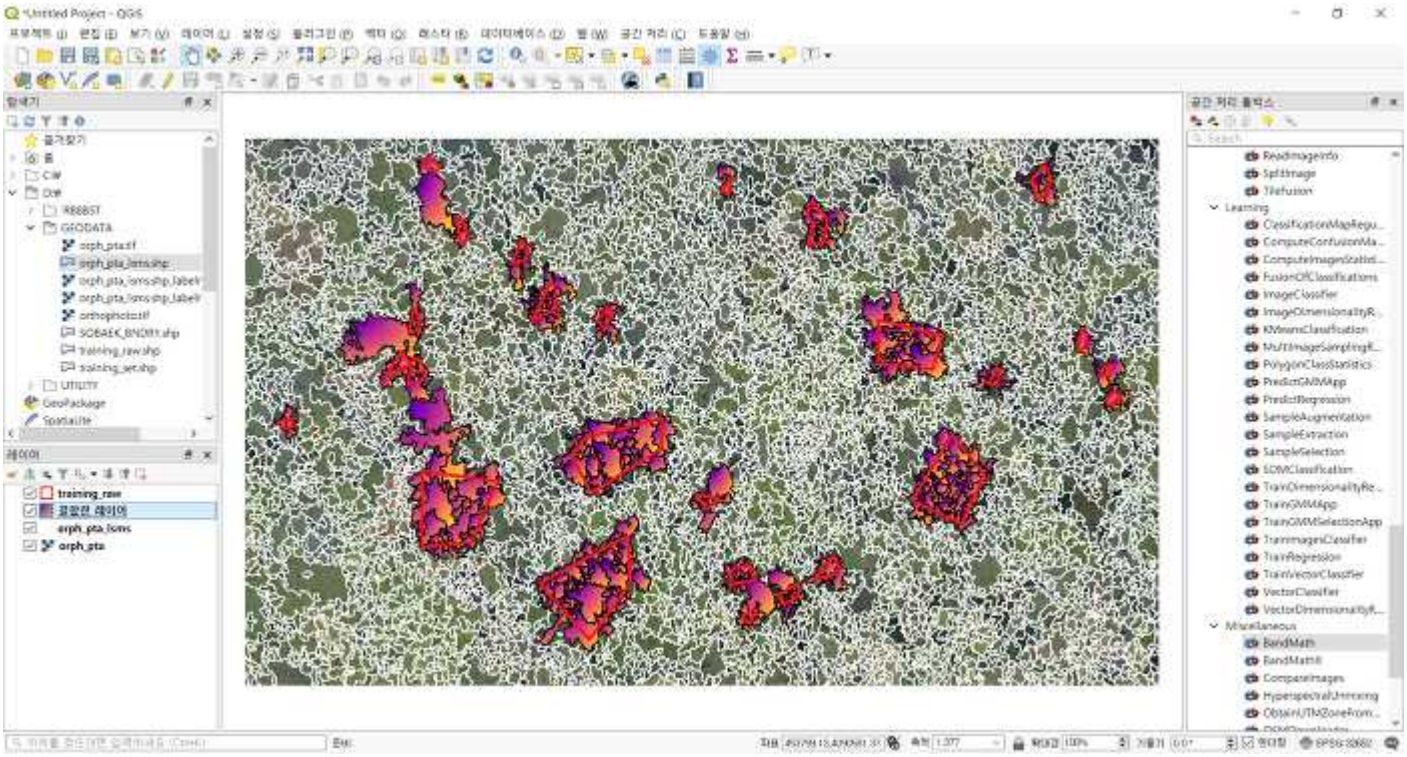
'벡터 >데이터 관리 도구 >위치에 따라 속성을 결합'을 클릭합니다.



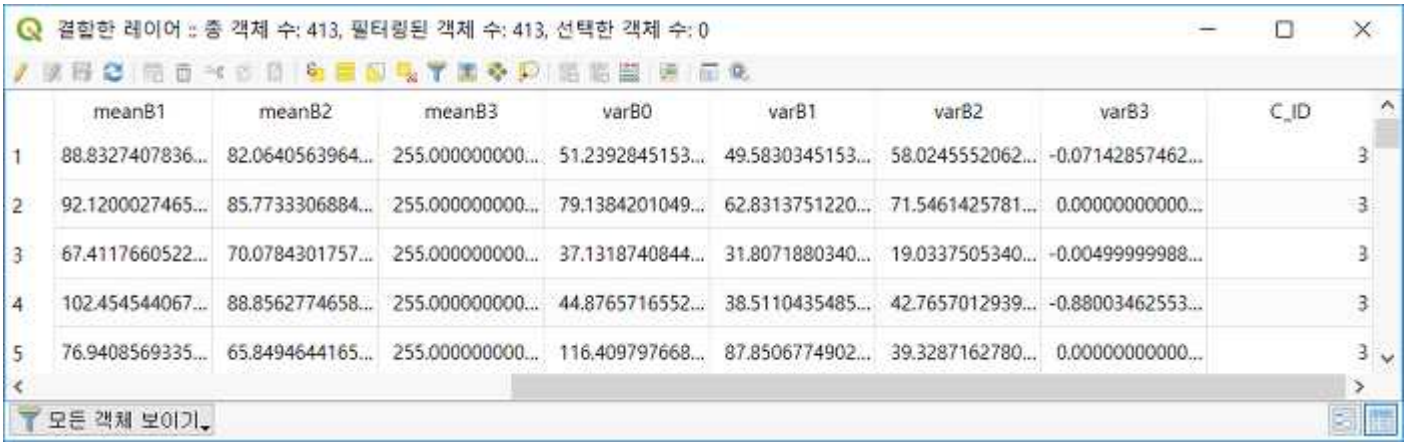
입력 레이어는 세그먼테이션, 결합 레이어는 훈련데이터 영역을 지정합니다.
기하학적 조건, 추가할 필드, 결합 유형 등을 지정해주고 훈련용 세그먼트를 추출합니다.



자, 아래와 같이 훈련데이터가 생성되었습니다. 속성 테이블을 한 번 열어볼까요?!

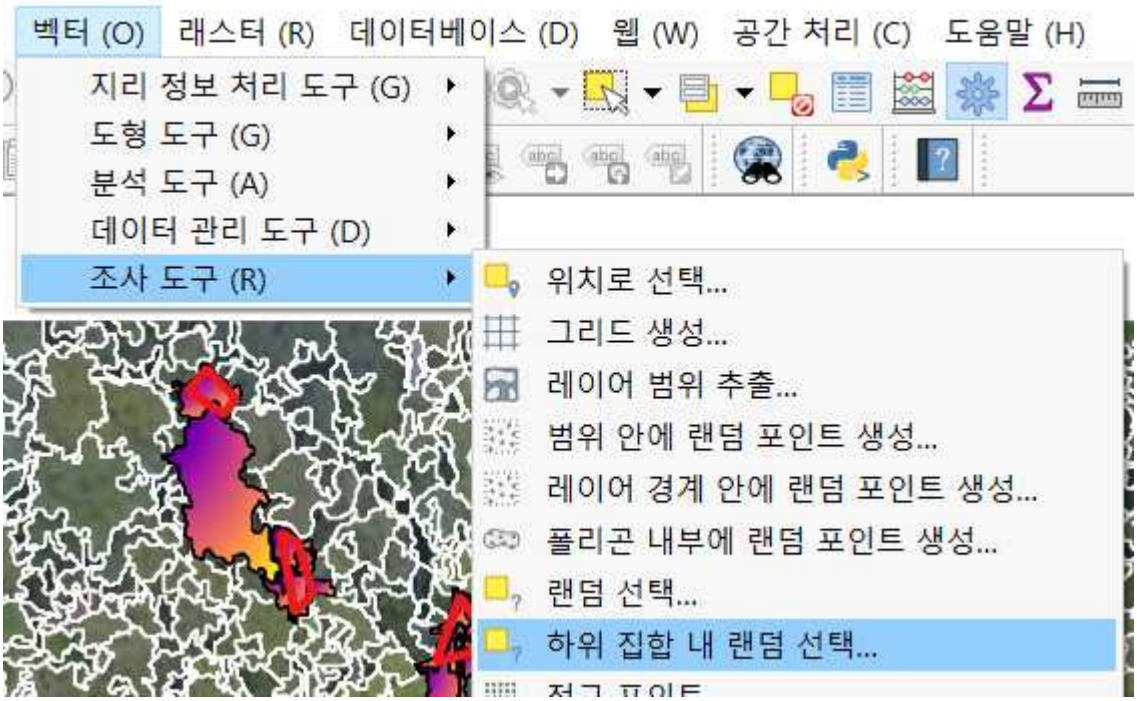


속성 테이블에는 기존 필드값에 C_ID 필드가 추가된 것을 확인하실 수 있습니다.

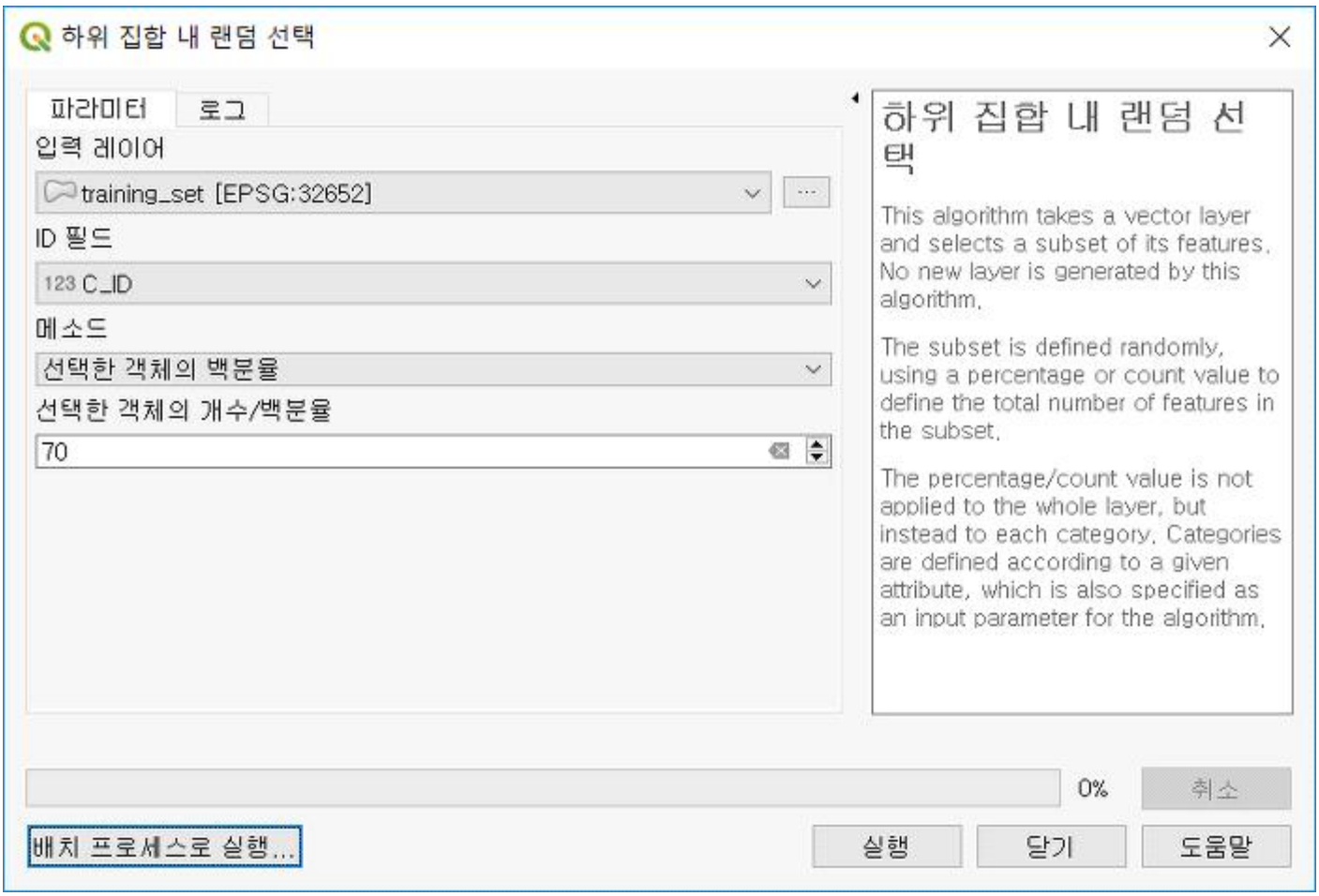


	meanB1	meanB2	meanB3	varB0	varB1	varB2	varB3	C_ID
1	88.8327407836...	82.0640563964...	255.000000000...	51.2392845153...	49.5830345153...	58.0245552062...	-0.07142857462...	3
2	92.1200027465...	85.7733306884...	255.000000000...	79.1384201049...	62.8313751220...	71.5461425781...	0.00000000000...	3
3	67.4117660522...	70.0784301757...	255.000000000...	37.1318740844...	31.8071880340...	19.0337505340...	-0.00499999988...	3
4	102.454544067...	88.8562774658...	255.000000000...	44.8765716552...	38.5110435485...	42.7657012939...	-0.88003462553...	3
5	76.9408569335...	65.8494644165...	255.000000000...	116.409797668...	87.8506774902...	39.3287162780...	0.00000000000...	3

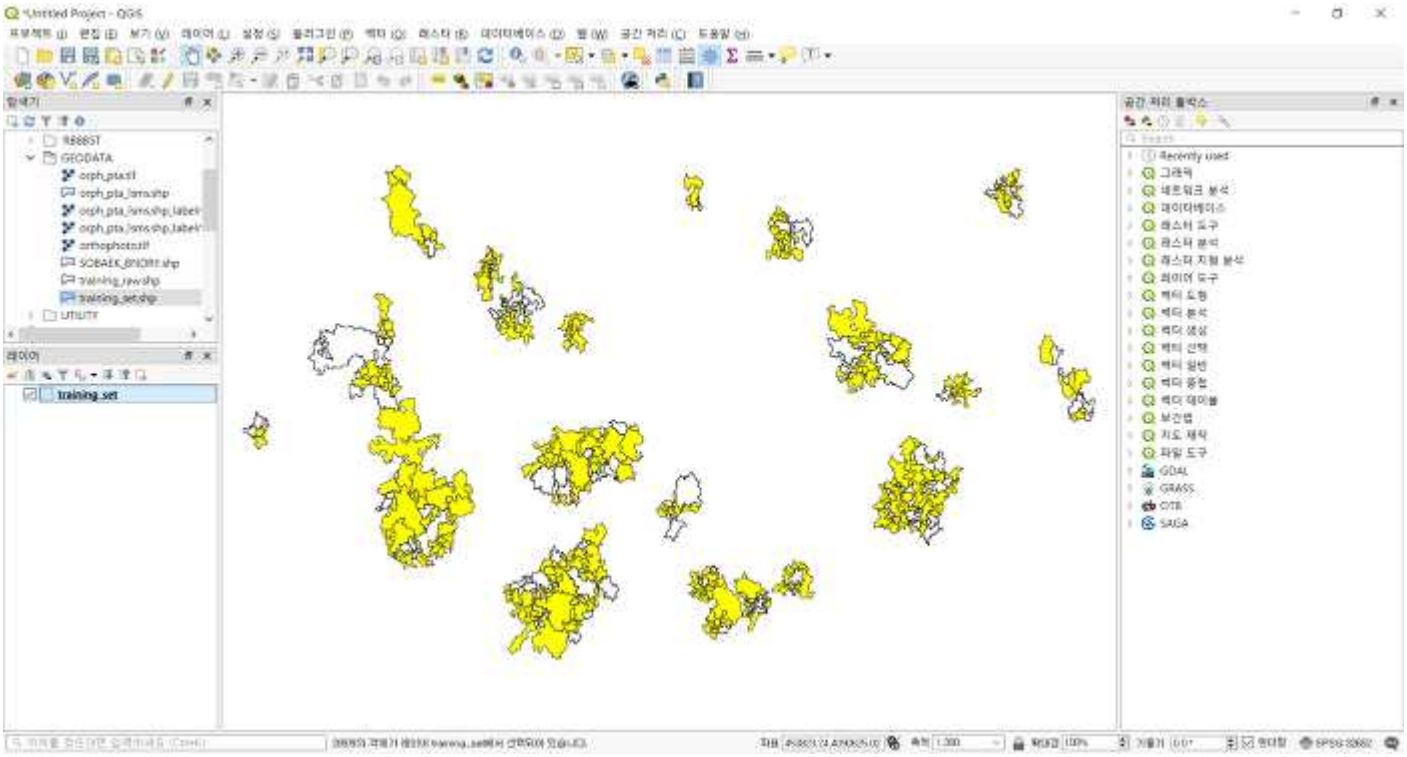
이제 훈련데이터를 학습과 검증 목적으로 분할하겠습니다. '벡터 >조사 도구 >하위 집합 내 랜덤 선택'을 클릭합니다.



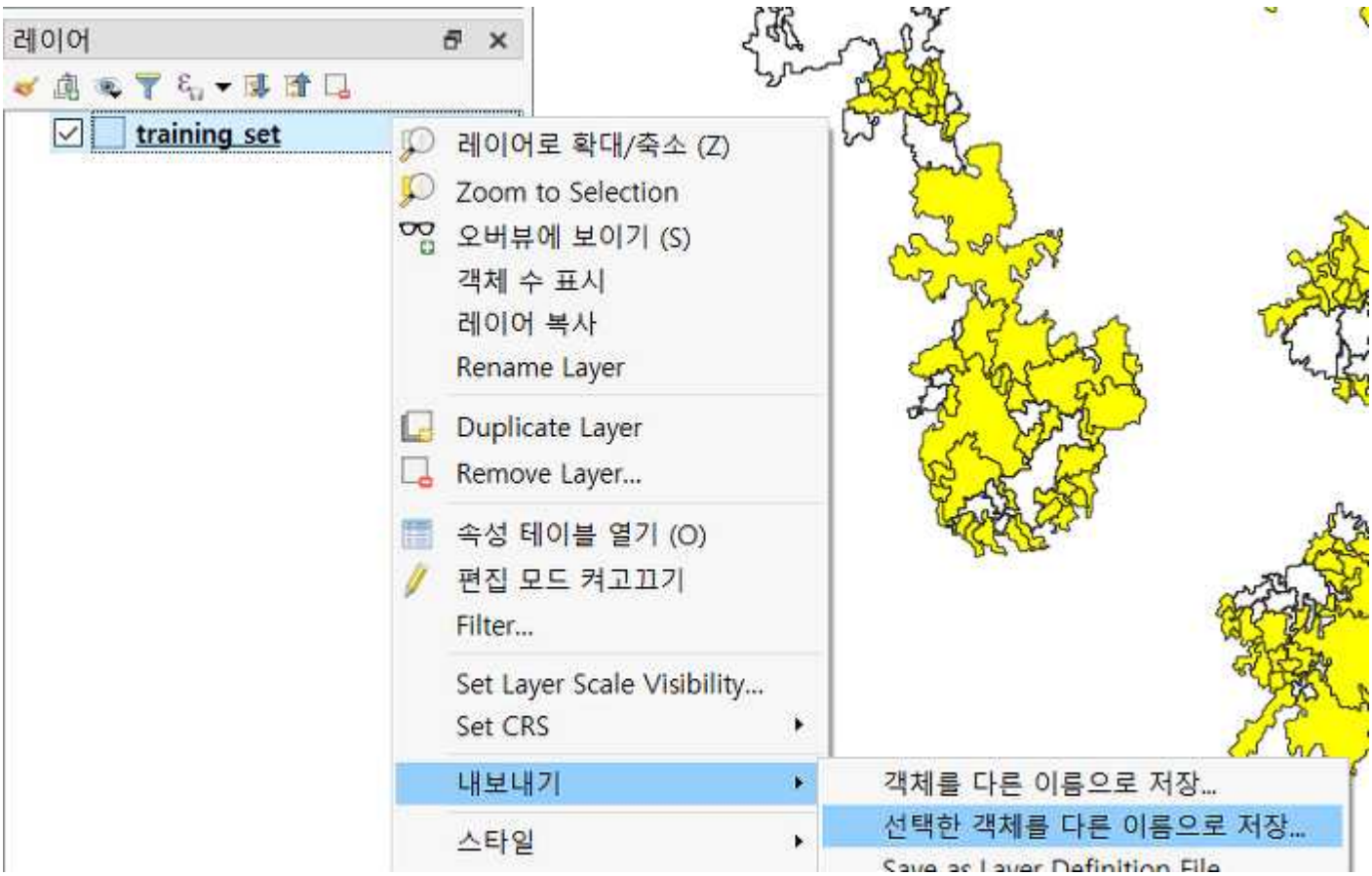
하위 집합 내 랜덤 선택 창에서 ID 필드는 C_ID, 메소드는 선택한 객체의 백분율, 선택한 객체의 개수/백분율은 70을 선택합니다.



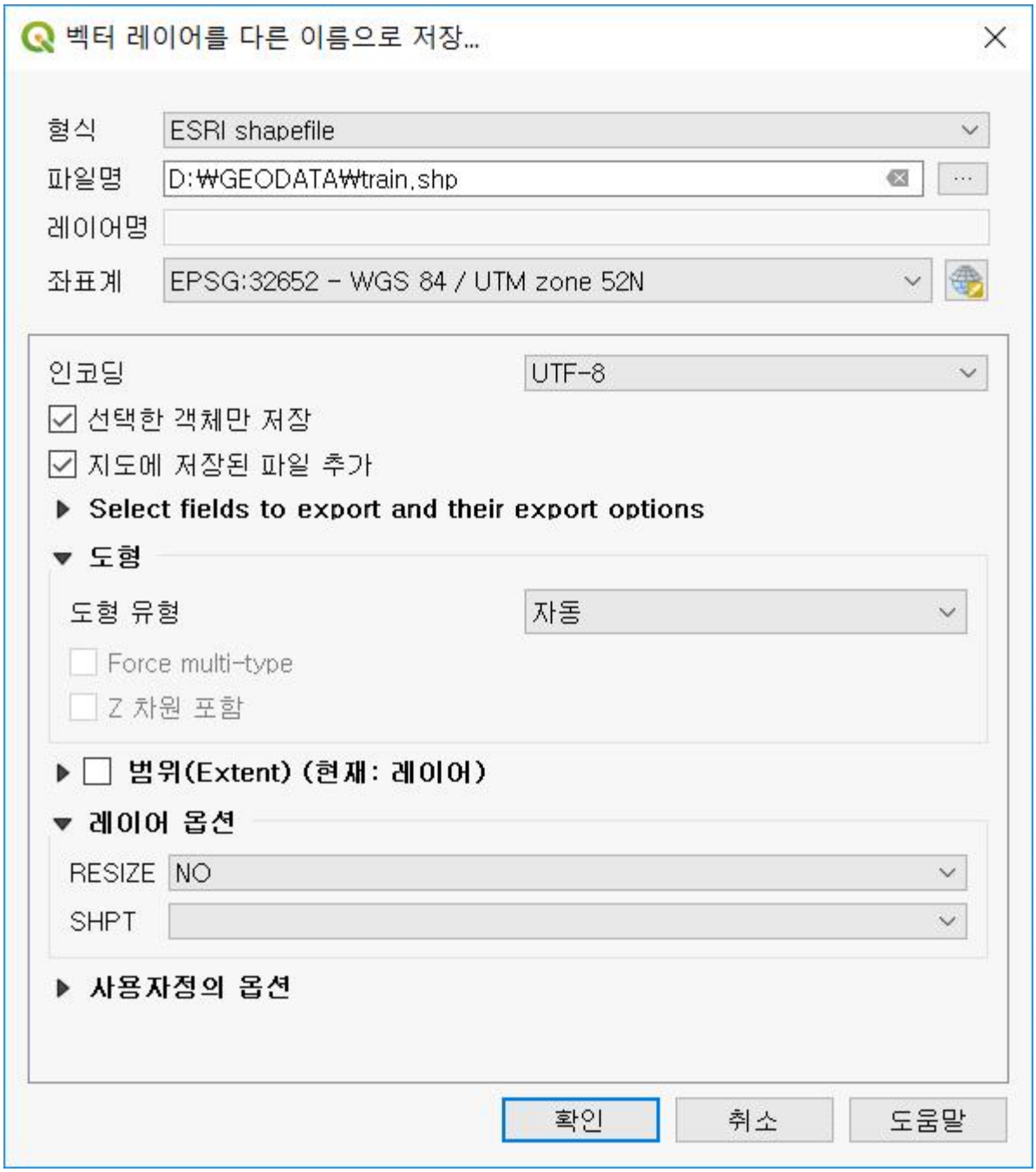
아래와 같이 C_ID 필드를 기준으로 전체 데이터의 70%에 해당하는 세그먼트가 임의 선택됩니다.



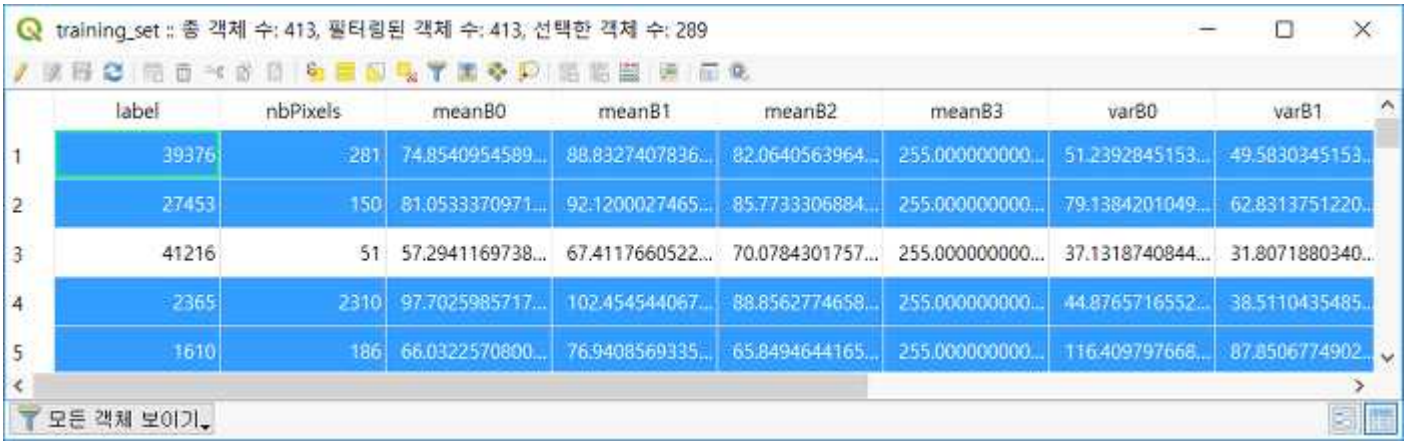
훈련 데이터 명을 우클릭하고 '내보내기 >선택한 객체를 다른 이름으로 저장'을 선택합니다.



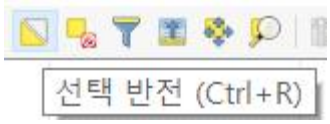
선택된 객체를 train.shp으로 저장하겠습니다.



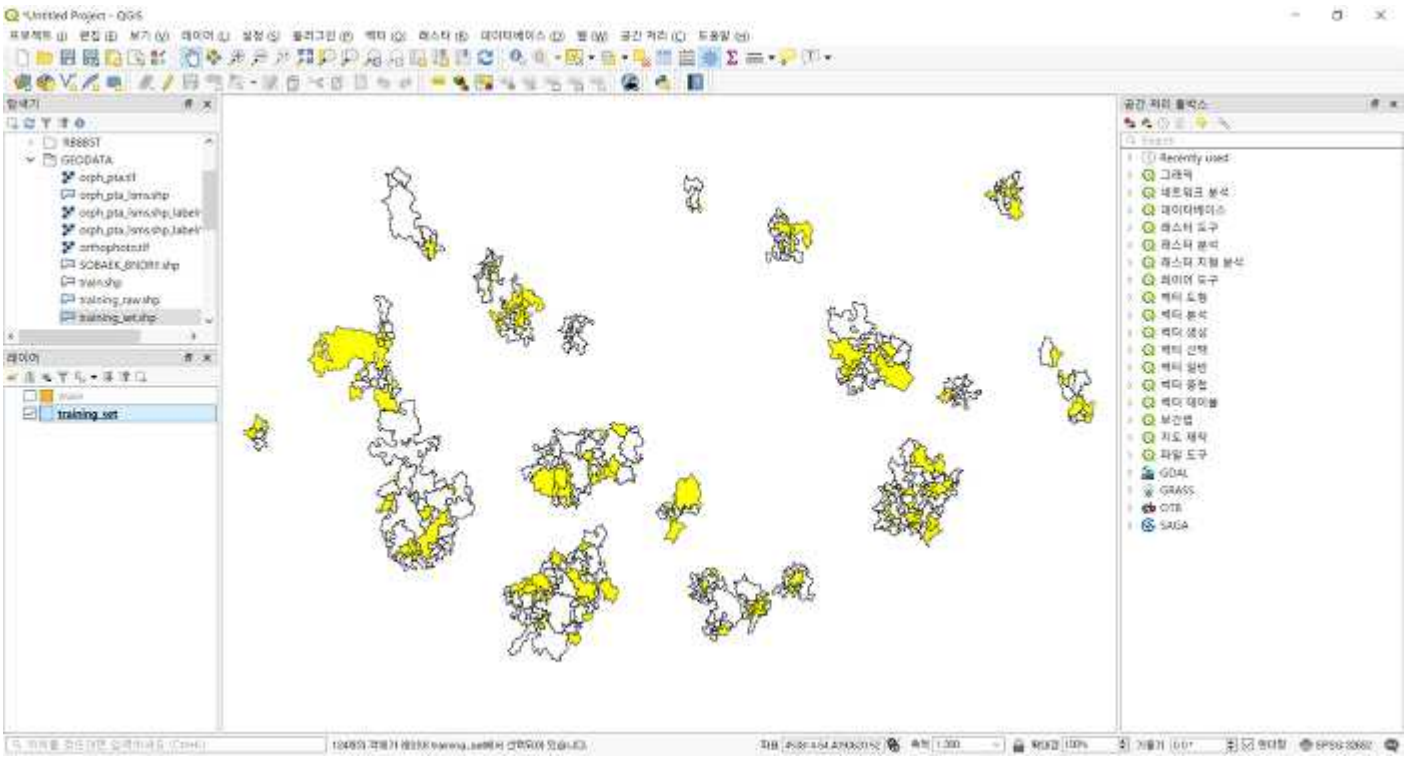
훈련 데이터의 속성에 들어가서 선택되지 않은 객체를 선택합니다.



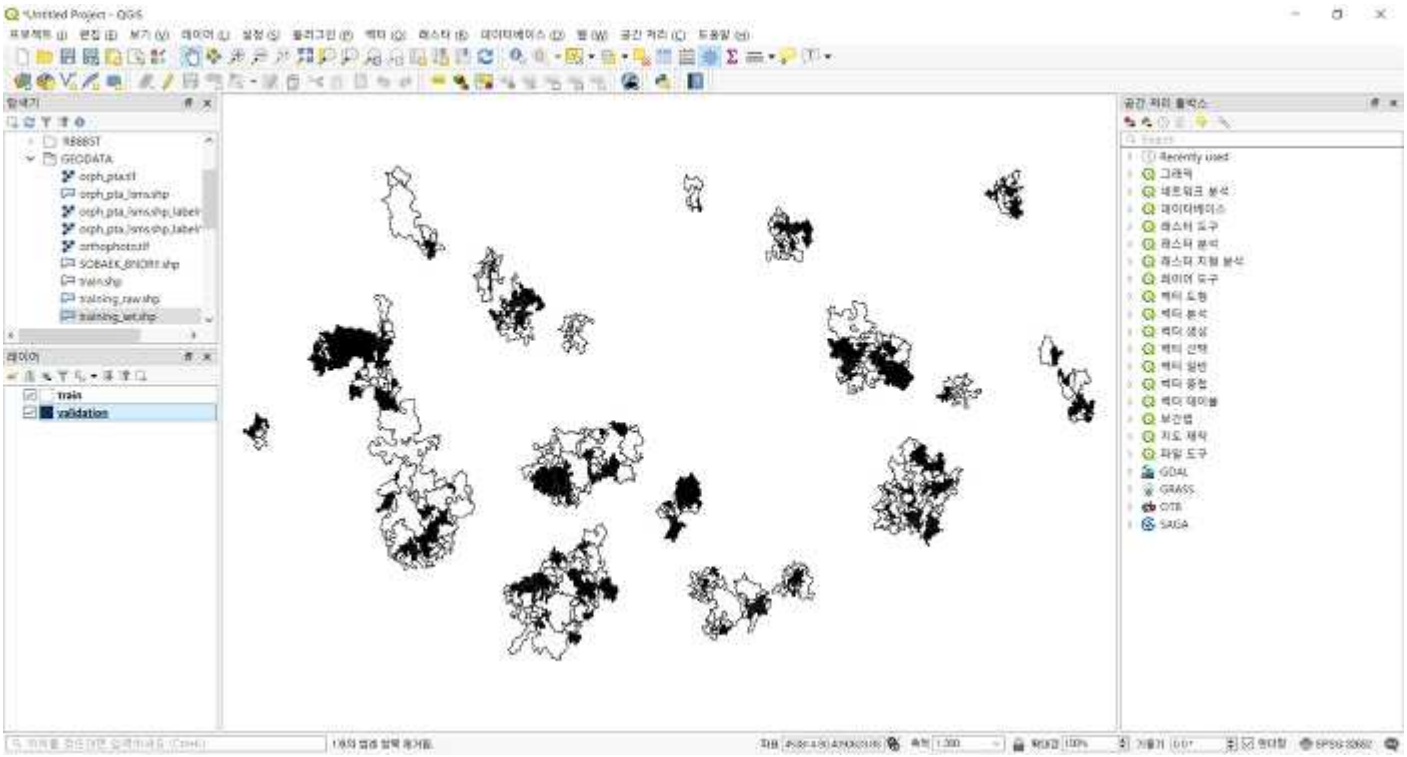
속성 테이블 상단에 '선택 반전' 버튼을 클릭하시면 됩니다.



자, 반대로 세그먼트가 선택됐죠?! 이걸 validation.shp으로 저장해줍니다.



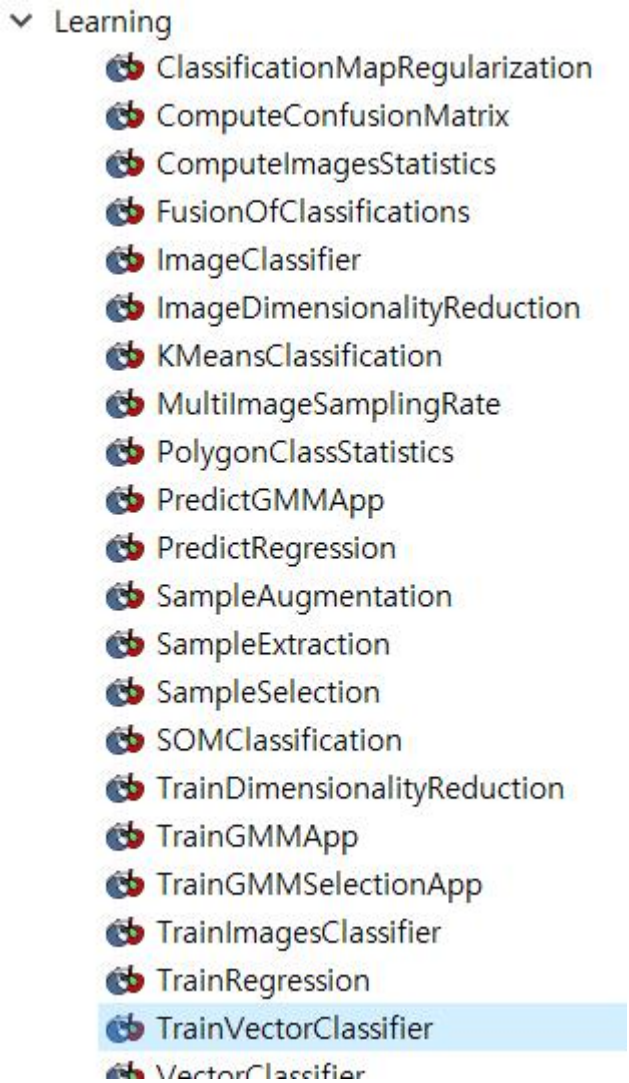
이제 분류기 훈련데이터를 위한 학습용, 검증용 데이터가 확보되었습니다.



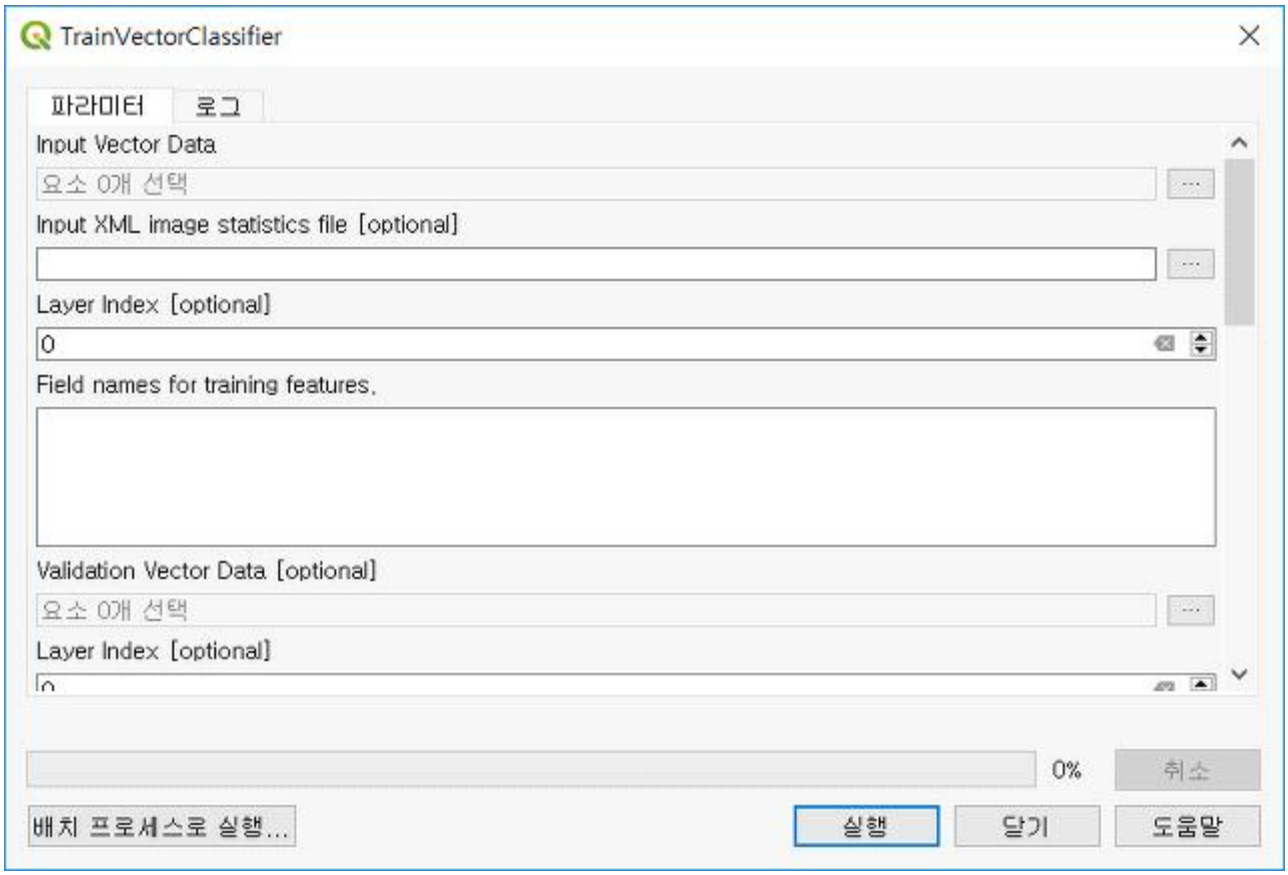
QGIS 3.4 Orfeo ToolBox(OTB) - (5) 머신러닝 분류기

안녕하세요? 훈련 데이터를 기반으로 분류기를 직접 다뤄 보겠습니다.

'Learning >TrainVectorClassifier'를 실행합니다.



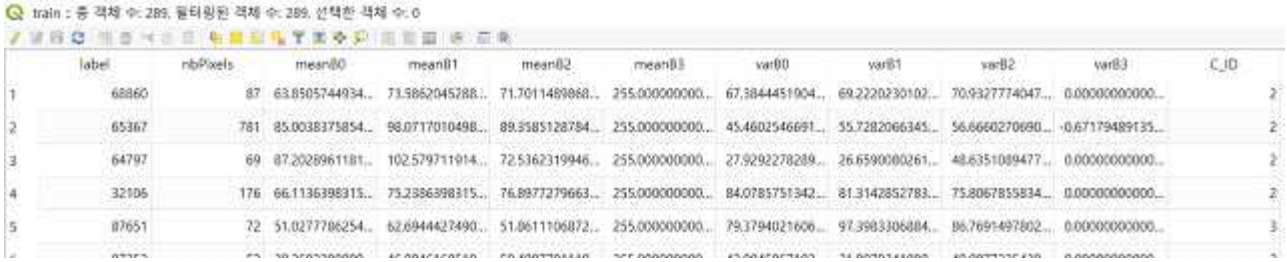
TrainVectorClassifier 실행 창은 아래와 같습니다. 어떤 파라미터가 있는지 살펴볼까요?!



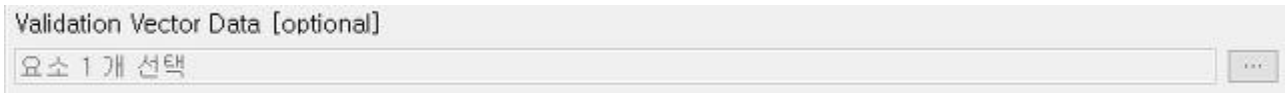
Input Vector Data는 앞서 제작한 train.shp(훈련용)을 지정해 줍니다.

Field names for training features에는 train.shp 필드값 중 훈련에 사용할 필드명을 적어주시면 됩니다.

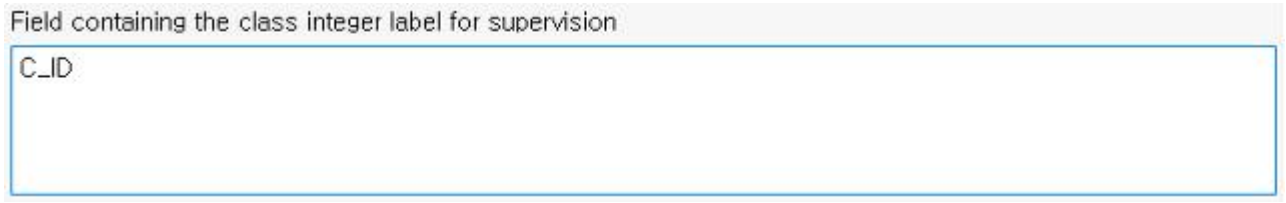
참고로, train.shp의 속성 테이블은 아래와 같습니다.



Validation Vector Data는 validation.shp(검증용) 데이터를 지정합니다.

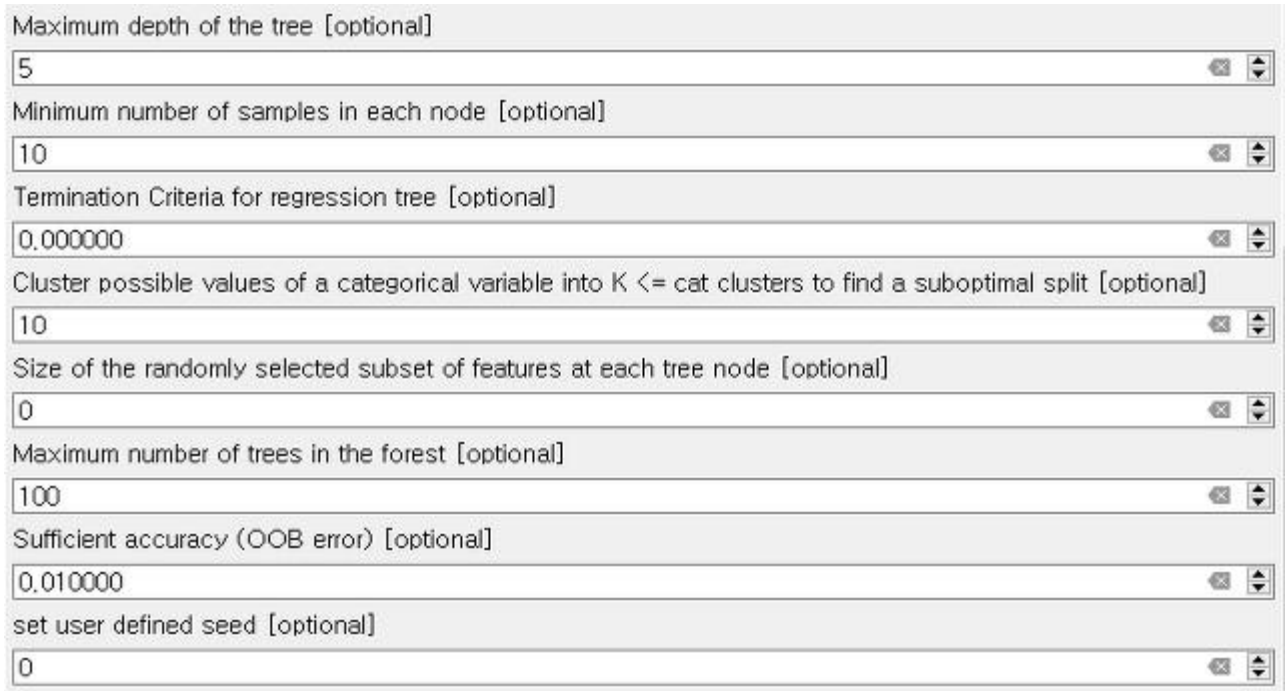


Field containing the class integer label for supervision에는 C_ID, 즉 분류 ID가 저장된 필드명을 입력해 줍니다.



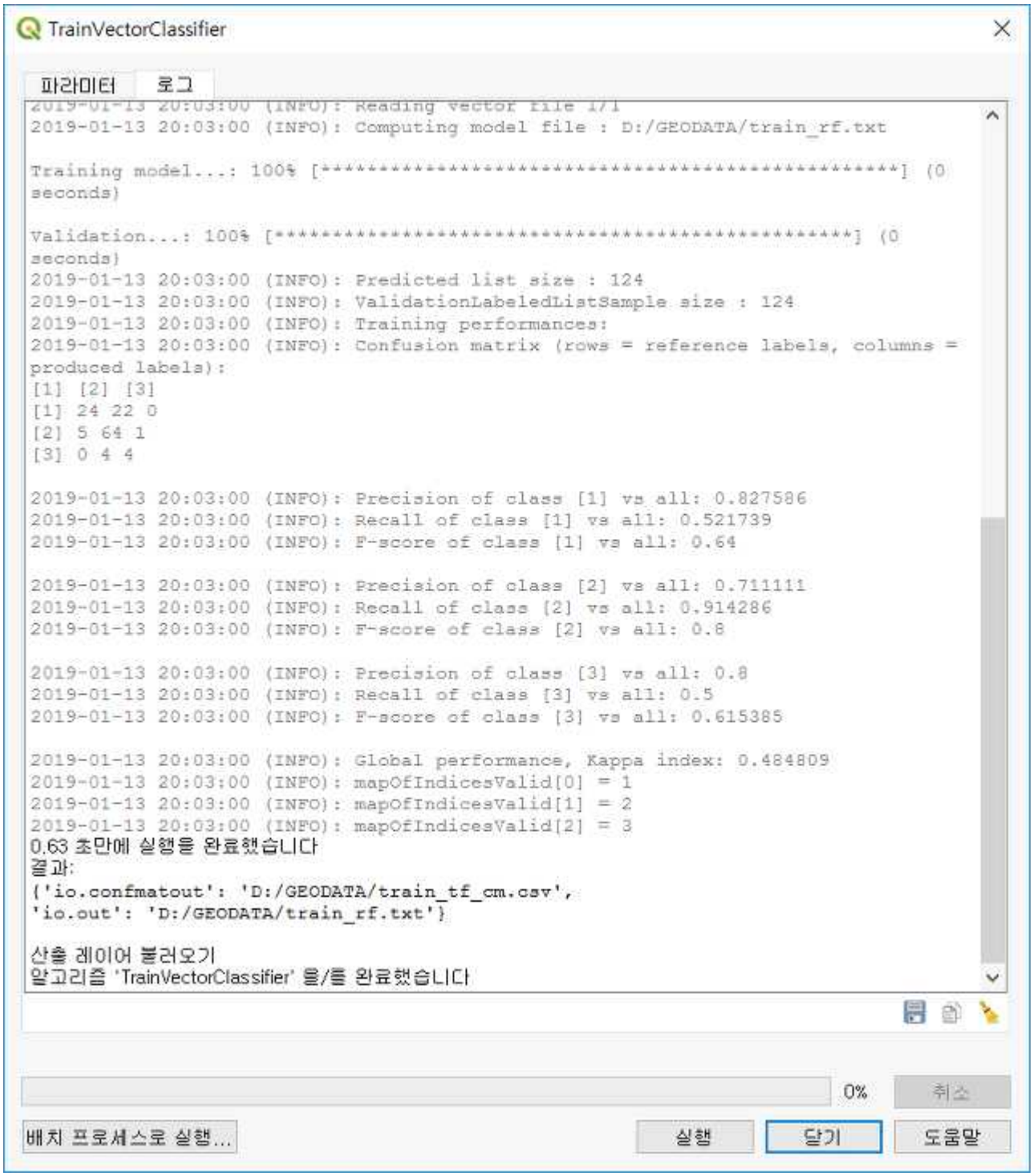
Classifier to use the training에서는 분류기를 선택해주시면 되는데요, 아래와 같이 다양한 분류기들이 제공되고 있습니다. 저는 최근 공부하고 있는 랜덤 포레스트(random forest: rf) 분류기를 선택해 주겠습니다.

선택된 분류기에 따라 그 분류기에 해당하는 매개변수 설정이 안내됩니다. rf에 대한 설정값들은 아래와 같습니다.

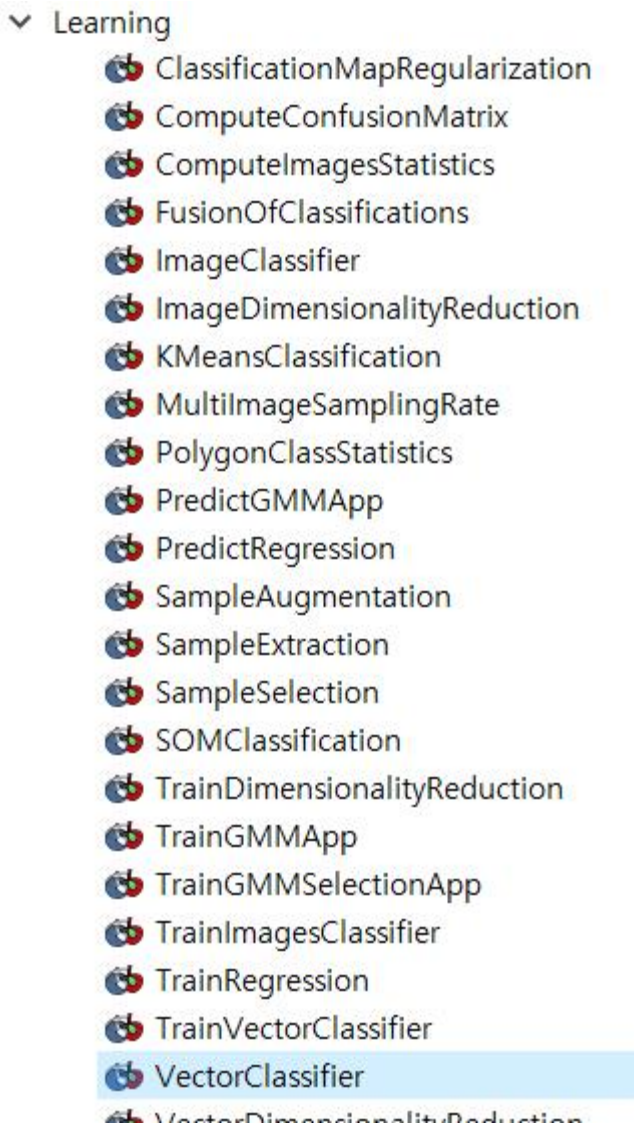


Output model은 분류기에 훈련 데이터를 통해 학습시킨 모델 파일로써, 확장자는 txt를 쓰시면 됩니다.
Output confusion matrix(오차행렬) or contingency table(분할표)는 검증 데이터를 통해 확인한 결과입니다.

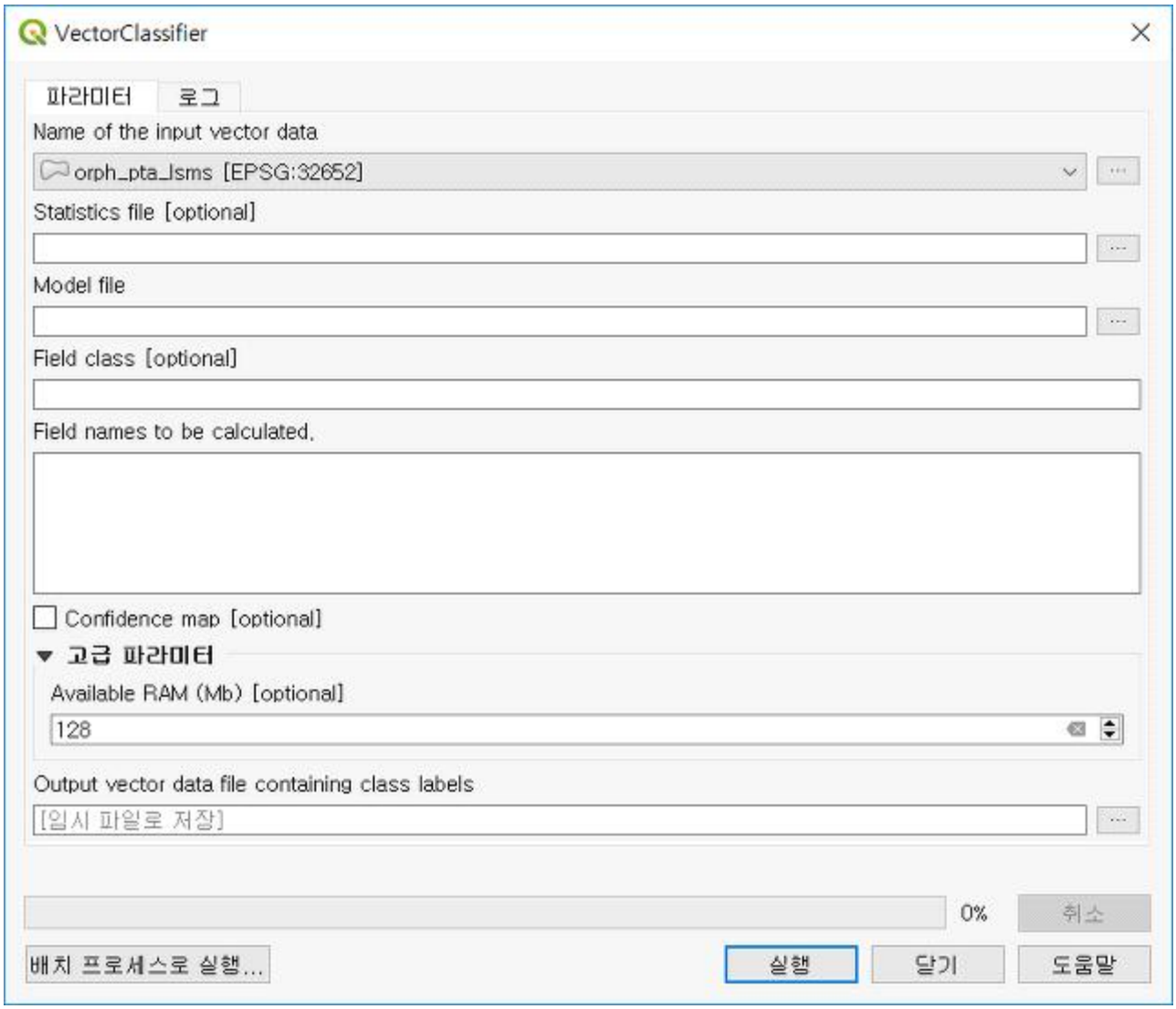
아래와 같이 훈련이 종료되었습니다. 결과를 전체 세그먼테이션에 적용해볼까요?!



'Learning >VectorClassifier'를 실행합니다.



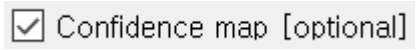
VectorClassifier 창은 아래와 같습니다. 일단 Name of the Input vector data에 세그먼테이션 파일을 지정하고,



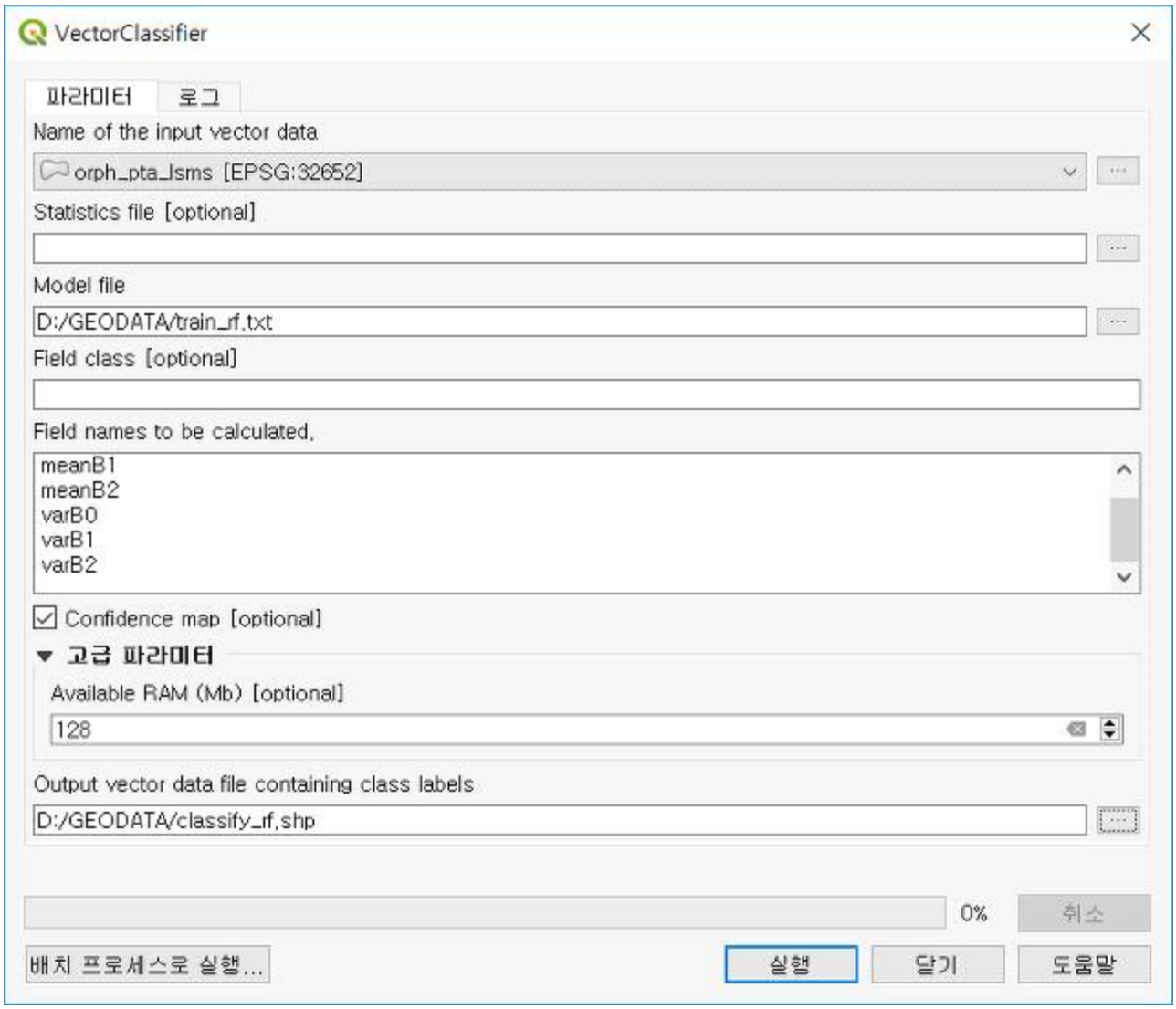
Model file에 앞서 훈련시킨 모델 파일을 지정합니다.

Field names to be calculated는 앞서 훈련 단계와 동일하게 필드값을 지정해주시면 됩니다.

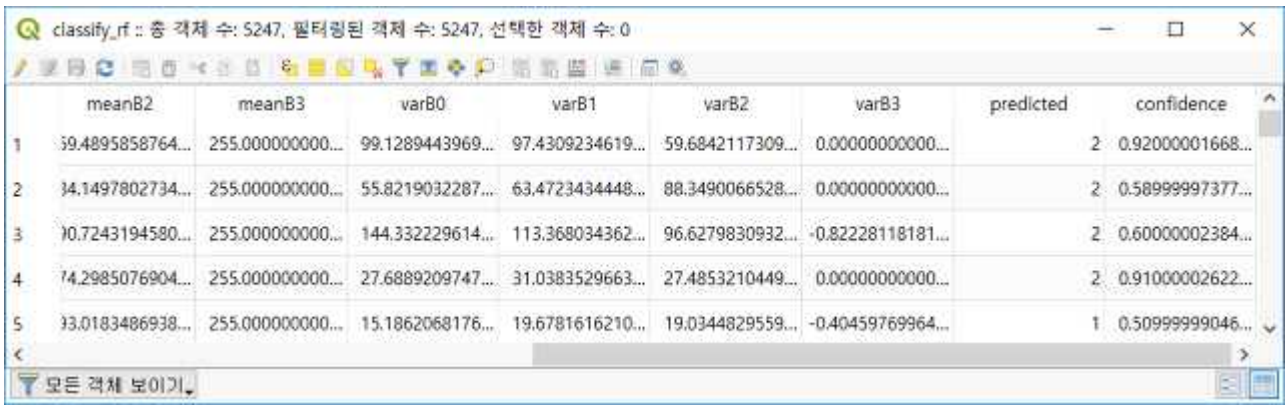
Confidence map은 체크해두시는 편이 좋은데요, 이 값의 의미는 선택한 분류기마다 차이가 있습니다. 랜덤 포레스트(rf) 분류기에서는 다수를 점한 클래스의 투표 비율을 나타냅니다.



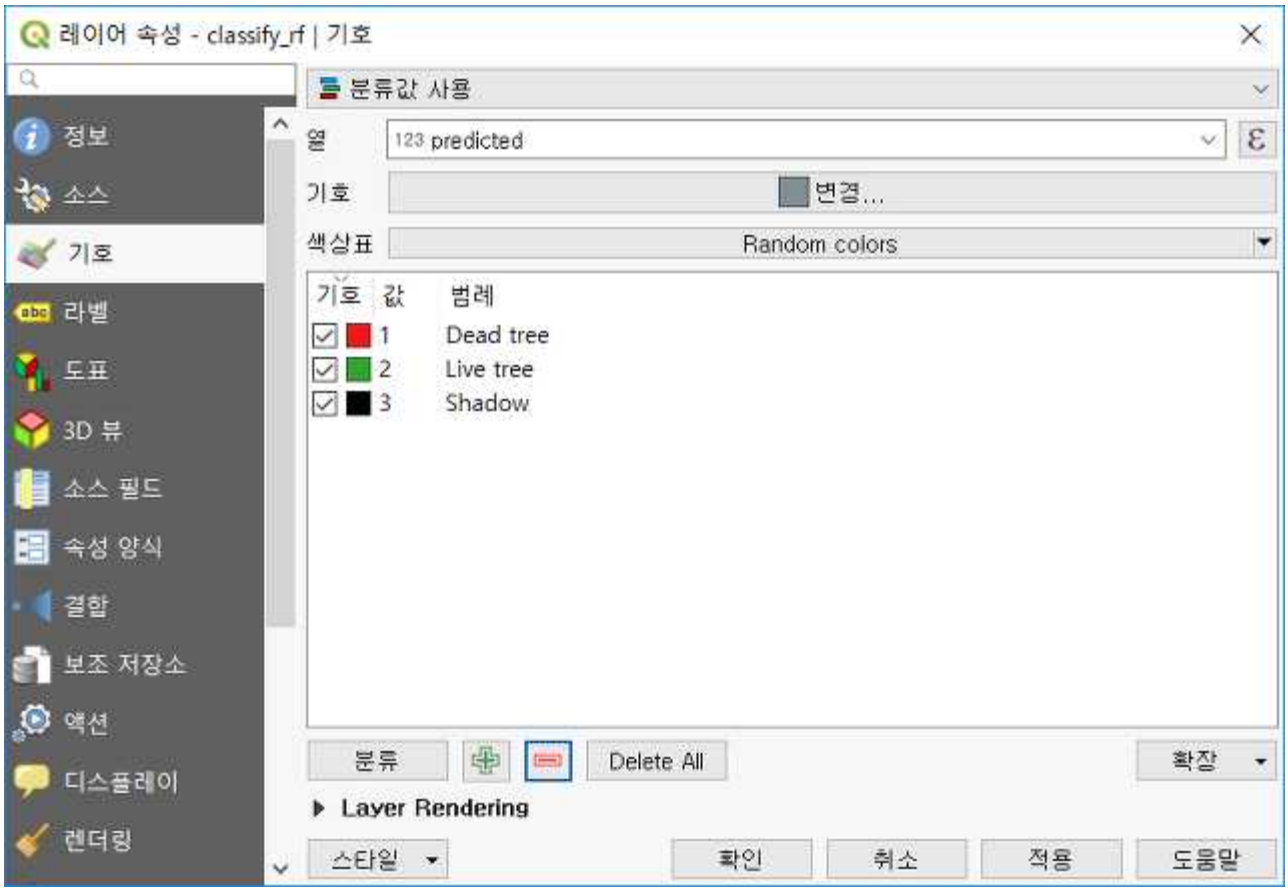
이제 분류기 결과를 추출해볼까요?!



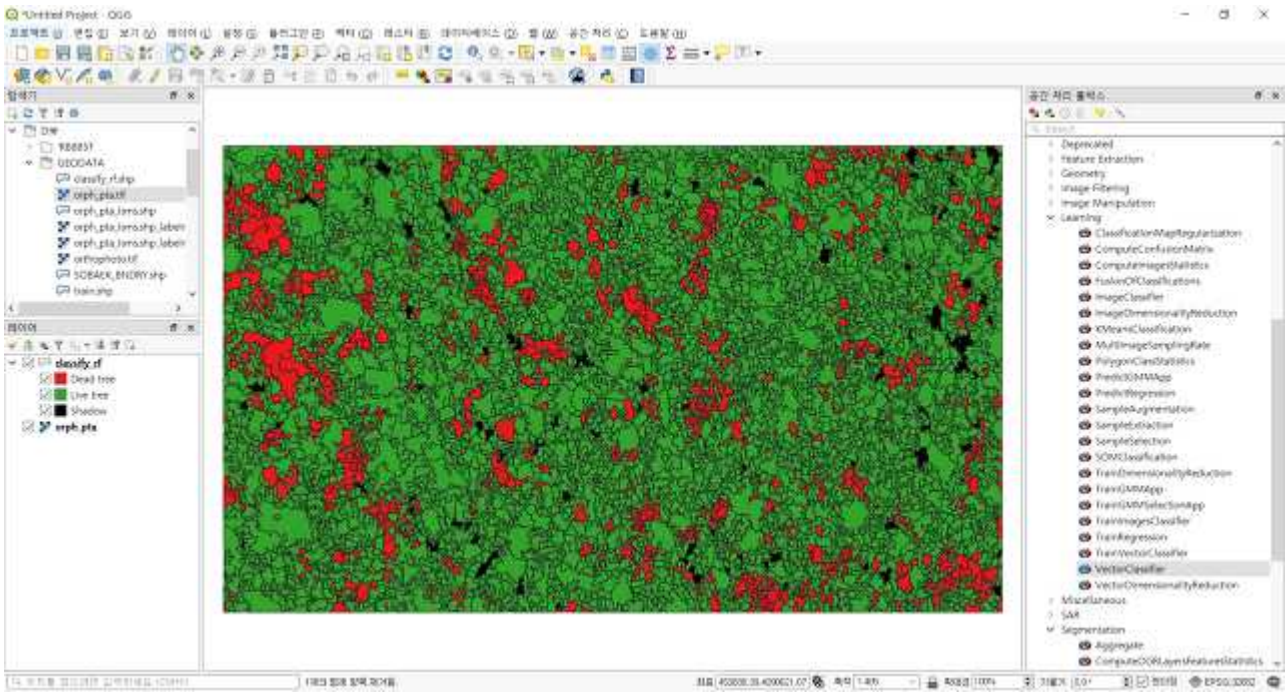
결과 파일의 속성 테이블을 열어보시면 predicted와 confidence 필드값이 추가된 것을 보실 수 있습니다.



레이어 속성에서 '기호 >분류값 사용'을 선택하고 각 값에 범례를 적용해 보겠습니다.



결과는 아래와 같습니다. 썩 좋지 않은데요, 이후에 이 분류기를 개선하는 방법을 정리해 보겠습니다.



QGIS 3.4 Orfeo ToolBox(OTB) - (6) 가시광선 식생지수 계산하기 (BandMath)

안녕하세요? 이번 글은 OTB BandMath 기능을 통해 무인기 영상의 식생지수를 계산해 보겠습니다.

현재 널리 쓰이고 있는 저가 UAV는 거의 대부분 일반적인 RGB 카메라를 탑재하고 있는데요,
전세계 70%를 점유하고 있는 중국 DJI 드론도 대부분 sRGB 스타일의 카메라를 탑재하고 있습니다.
sRGB는 1996년에 HP와 Microsoft과 협력하여 만든 표준 RGB 색 공간(standard RGB color space)입니다.

R, G, B는 각 원색의 색 좌표(chromatic coordinate)로 정의될 수 있는데요,
아래 그림은 r좌표와 g좌표로 정의된 색도분포표(chromaticity diagram)입니다.
1931년 국제조명위원회(CIE)가 제정한 것으로, x축은 적색 좌표, y축은 녹색 좌표입니다.

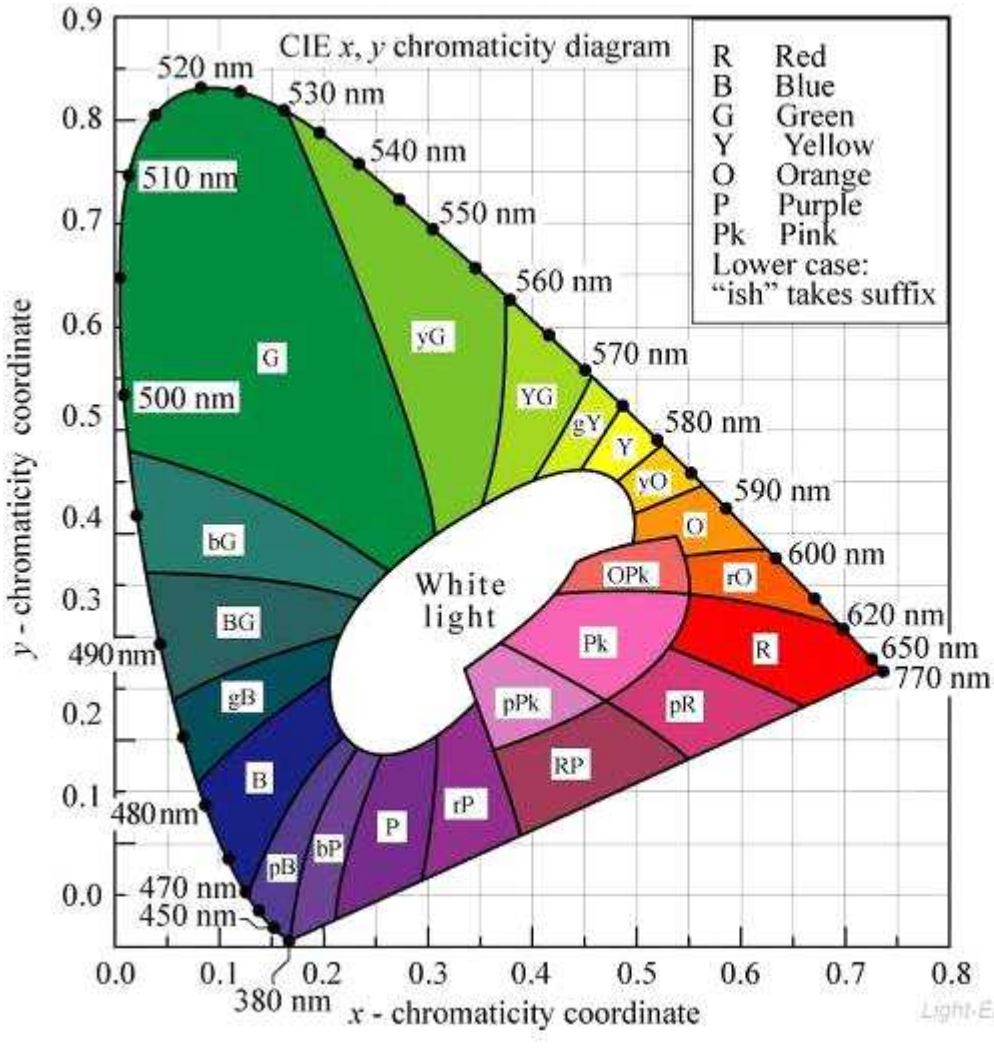


Fig. 17.3. 1931 CIE chromaticity diagram with areas attributed to distinct colors (adopted from Gage *et al.*, 1977).

이미지 출처: https://ecse.rpi.edu/~schubert/Light-Emitting-Diodes-dot-org/chap17/F17-03%20Chromaticity%20diagram%20(Gage).jpg

보통 다중분광 위성영상의 경우 RGB 밴드와 NIR 밴드를 함께 취득하여 NDVI와 같은 식생지수를 계산했었는데요,
UAV 기반 RGB영상은 공간해상도는 매우 우수한 반면, 분광해상도는 원격탐사 목적에 제한적이라고 볼 수 있습니다.

최근에 UAV에 탑재 가능한 적외선 카메라들 상당수는 기존 카메라를 NIR 필터로 개조한 형태입니다.

이미지 출처: http://diydrones.com/profiles/blogs/diy-camera-filter-swap-on-a-canon-powershot

이런 경우에는 natural-color composite와 false-color composite 영상을 동시에 취득할 수가 없는데요,
여기서는 RGB 영상만으로 식생지수를 계산하고, 이것을 OTB BandMath에서 처리하는 방법을 정리해 보겠습니다.

먼저, RGB 영상의 식생지수에 대해 알아보겠습니다. 여기서는 아래 지수 중 NGRDI를 계산해 보겠습니다.

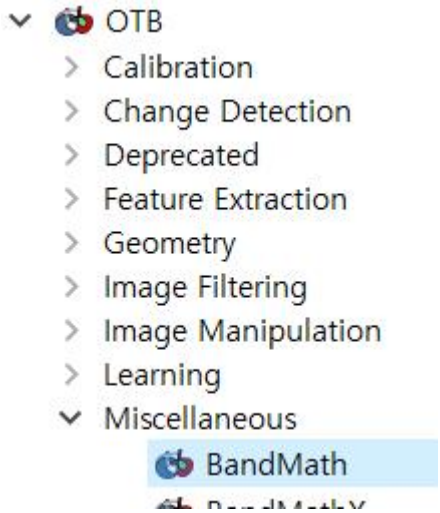
Table 3. Spectral vegetation indices evaluated in this study.

Index	Name	Formula	References	Camera
NExG	Normalized Excess green index	(2*G − R − B)/(G + R + B)	[43,45]	MS, RGB
NGRDI	Normalized green-red difference index	(G − R)/(G + R)	[21,44]	MS, RGB
GNDVI	Green normalized difference vegetation index	(NIR − G)/(NIR + G)	[46]	MS, CIR
ENDVI	Enhanced normalized difference vegetation index	(NIR + G − 2*B)/(NIR + G + 2*B)	(www.maxmax.com)	MS, CIR
$CI_{red\ edge}$	Red edge chlorophyll index	NIR/RE-1	[47]	MS
DATT	DATT	(NIR − RE)/(NIR − R)	[48]	MS

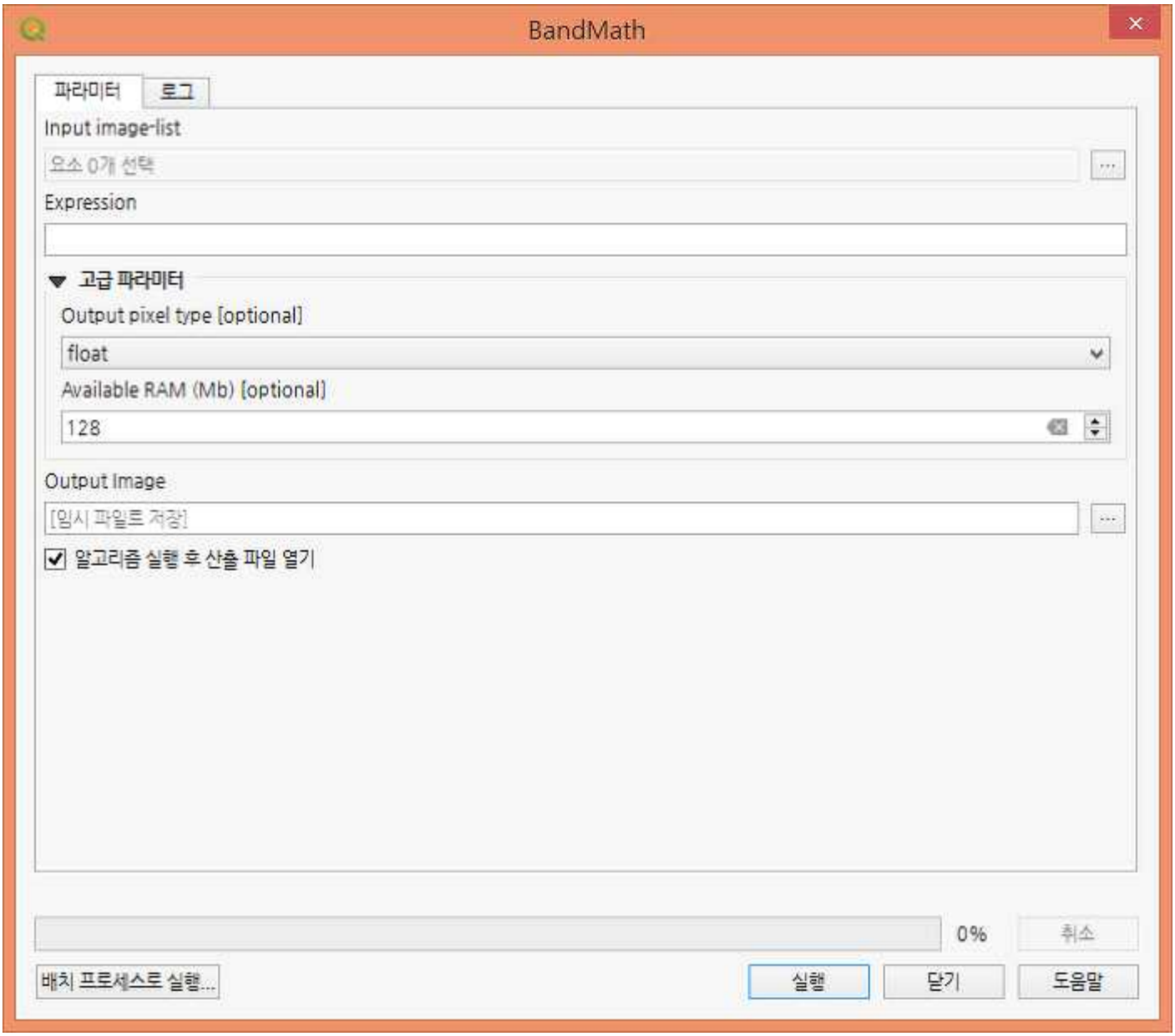
Note: For RGB and CIR images, NIR, R, G and B are the radiometric normalized pixel values of the NIR, red, green and blue band, respectively. For multispectral images, NIR, RE, R, G and B are the reflectance values of the near infrared, red edge, red, green and blue bands, respectively.

이미지 출처: https://www.mdpi.com/2072-4292/10/6/824

'OTB >Miscellaneous >BandMath'를 실행합니다.



BandMath 실행 창은 아래와 같습니다.

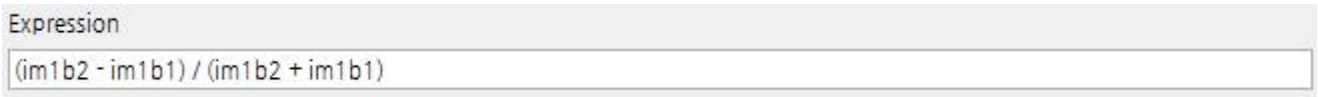


Input image-list에 무인기 영상을 선택합니다.

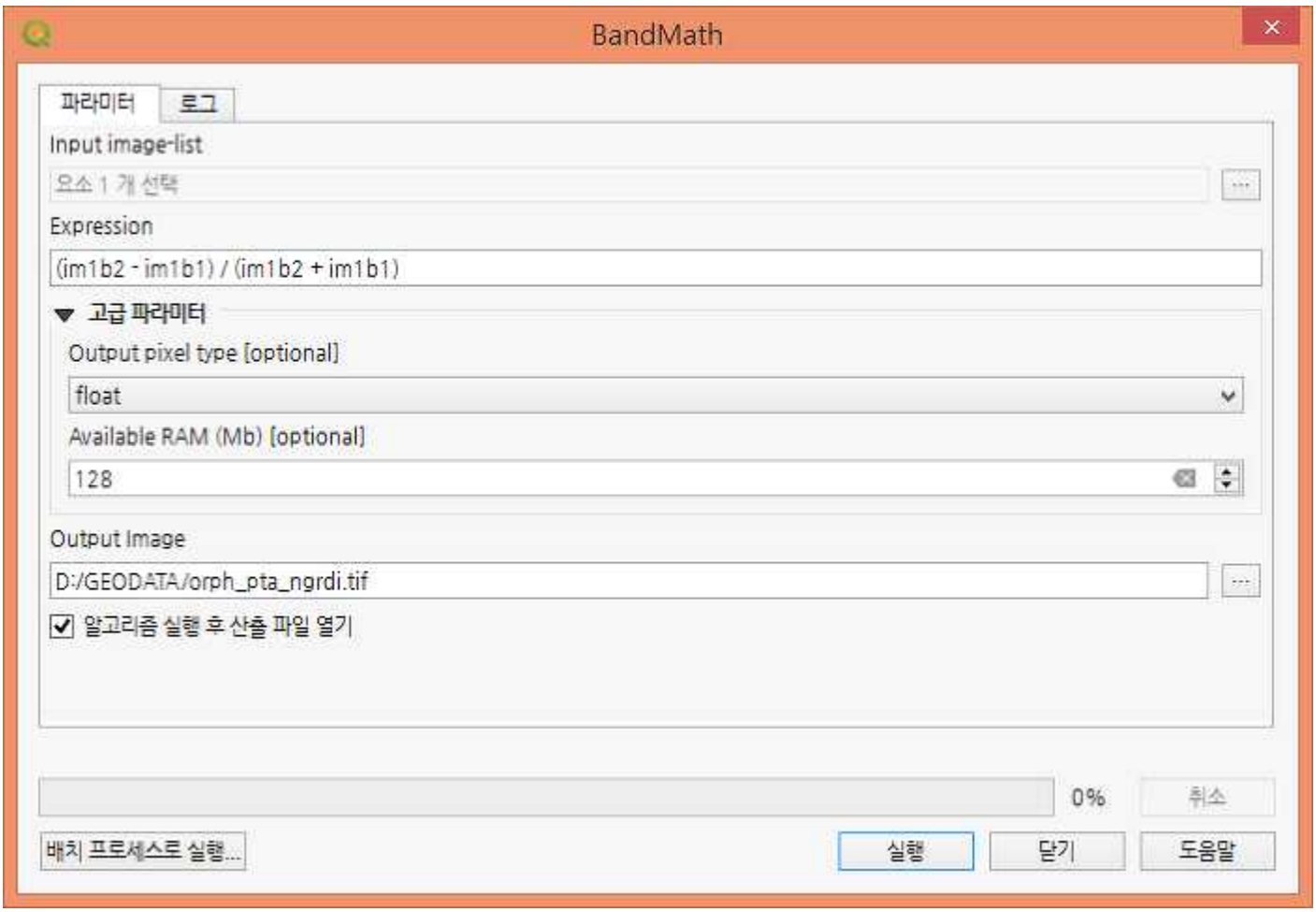
NGRDI 지수의 식은 아래와 같습니다. 여기서 G, R에 해당되는 밴드를 지정해 주면 되겠습니다.

OTB에서는 muParser 수학 표현식을 사용합니다.
muparser - Fast Math Parser Library | https://github.com/beltoforion/muparser

Input image-list에 등록된 첫 번째 이미지의 1번, 2번 밴드가 각각 R, G에 해당됩니다.
이에 수식은 아래와 같이 작성될 수 있습니다(예: im1b2는 첫번째 영상 2번 밴드의 화소값).



이제 BandMath를 실행해 보겠습니다.

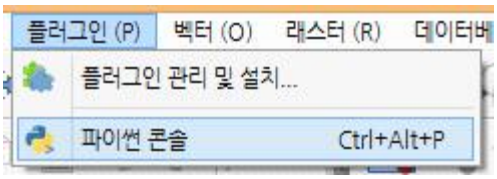


아래 이미지에서 좌측은 무인기 원본영상, 우측은 NGRDI 처리영상입니다. 식생과 비식생을 대비시키는데 도움이 되겠죠?!

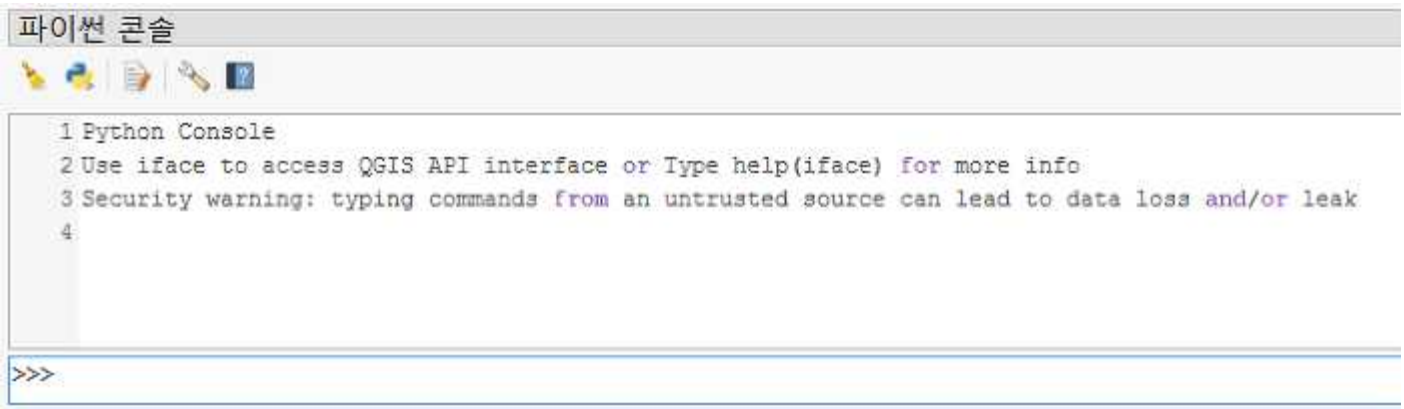
QGIS 3.4 Orfeo ToolBox(OTB) - (7) 파이썬 콘솔에서 레이어 추가하기

안녕하세요? 이번 글은 QGIS 3.4 파이썬 콘솔에서 레이어를 추가해 보겠습니다.
PyQGIS(Python과 QGIS의 조합)의 첫 걸음으로 볼 수 있겠죠?!

먼저, '플러그인 >파이썬 콘솔'을 실행합니다.



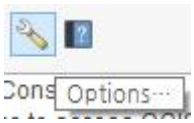
QGIS 하단에 아래와 같이 '파이썬 콘솔' 창이 실행됩니다.



파이썬 IDE에 익숙하신 분들은 '편집기 보이기' 창을 클릭하시면,

다음과 같이 좌측은 콘솔, 우측은 편집기 상태로 화면이 설정됩니다.

'Options' 버튼을 누르시면 '파이썬 콘솔 설정' 창이 실행되는데요,



화면 좌측과 같이 '콘솔', '편집기' 각각 설정이 분리되어 있습니다.

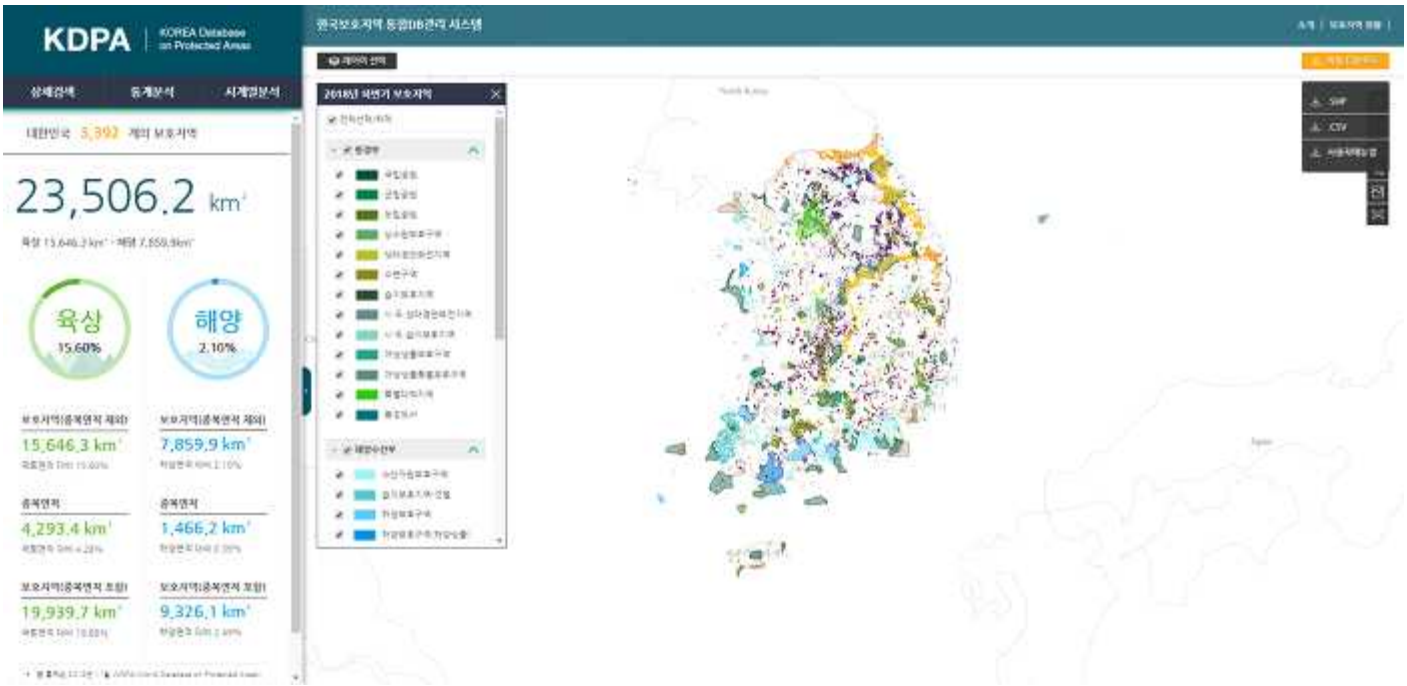
예를 들어, 콘솔과 편집기의 글꼴을 변경하고자 하시면, 각각의 설정을 변경해주셔야 합니다.
여기서는 글꼴을 Arial로 변경해 보겠습니다.

아래와 같이 콘솔과 편집기의 기존 글꼴이 Arial로 변경되었습니다.

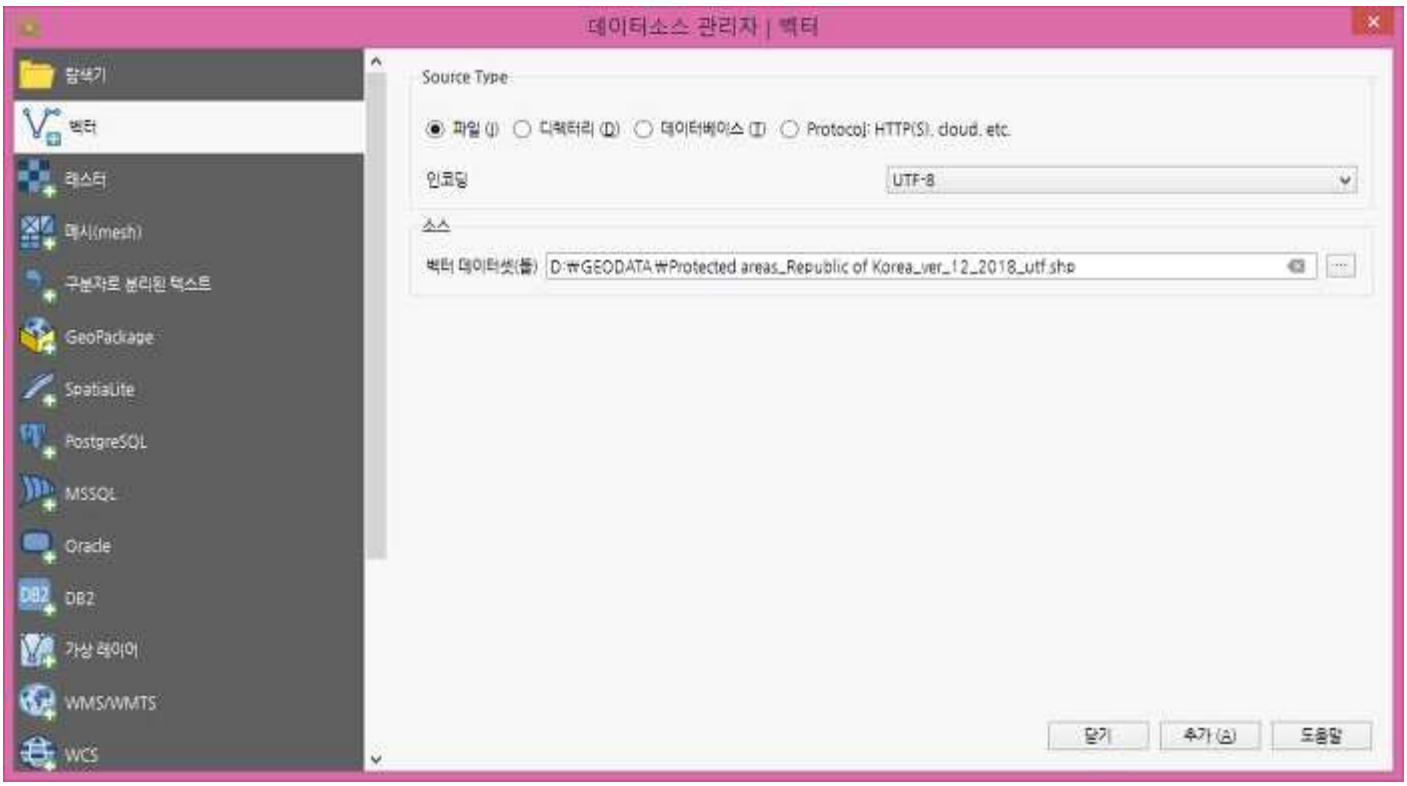
편집기에서 '저장' 버튼을 누르고,

파이썬 코드를 저장할 파일을 설정하도록 하겠습니다. 여기서는 'qgis3_addlayers.py'로 정의하였습니다.

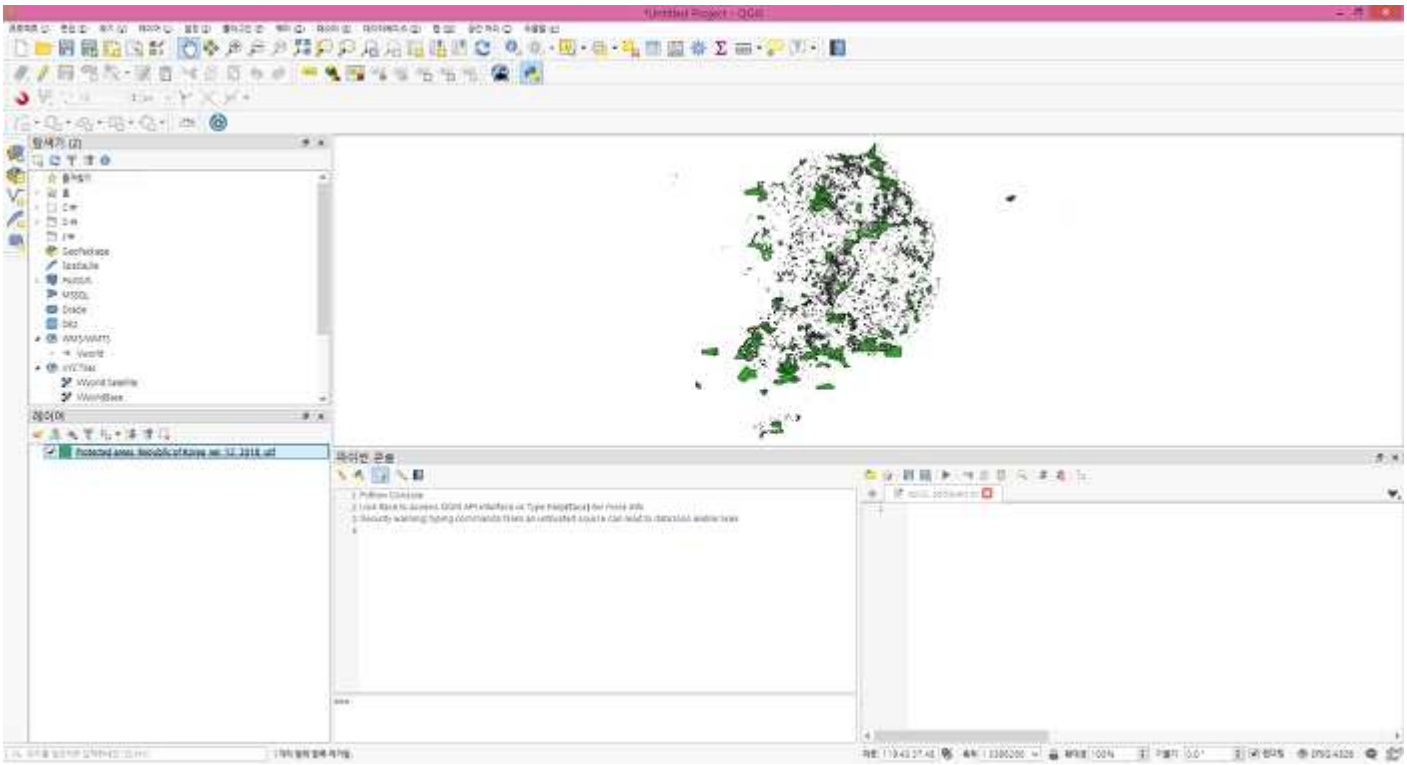
자, 벡터 레이어를 추가해볼까요?! 실습을 위한 소스는 KDPA 데이터를 사용하겠습니다.
KDPA(한국보호지역 통합DB관리 시스템)에서 국립공원 SHP 파일 다운로드 | http://blog.daum.net/geoscience/1239



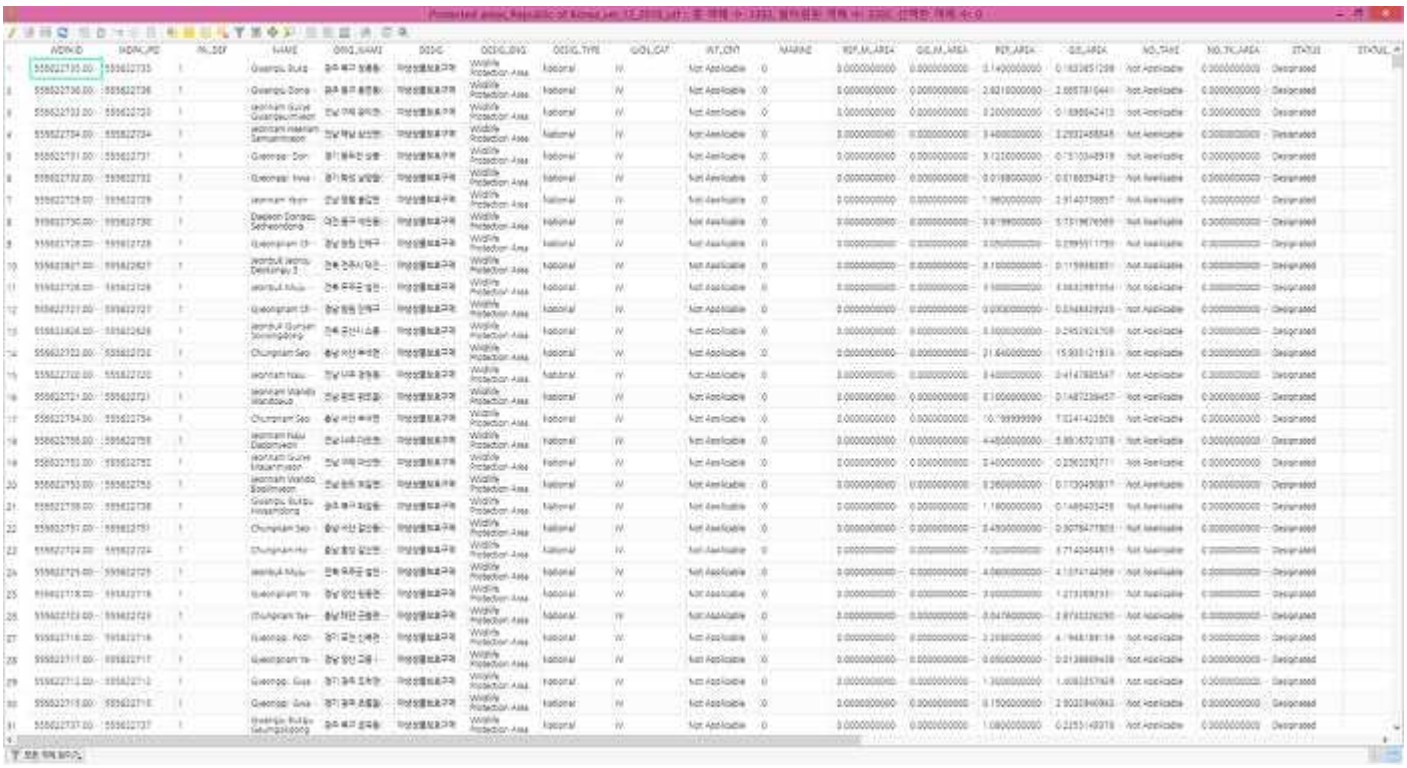
KDPA 데이터를 다운로드 받고 데이터소스 관리자를 통해 레이어를 추가해 봅니다.
※ KDPA 데이터는 인코딩을 UTF-8로 설정해주셔야 합니다.



아래와 같이 레이어가 추가되었는데요, 여기서 일부 피처만 선택해서 사용하겠습니다.

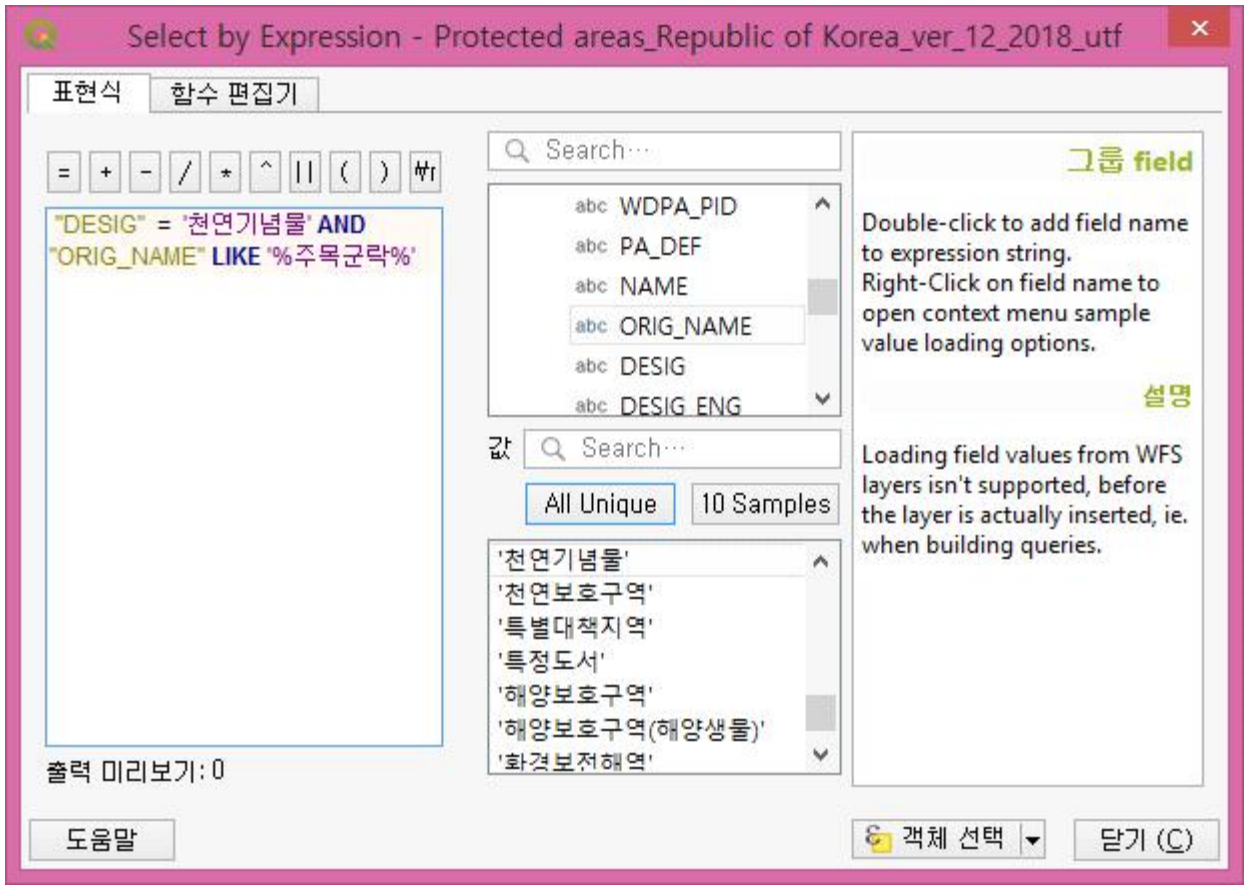


아래와 같은 속성 테이블에서,

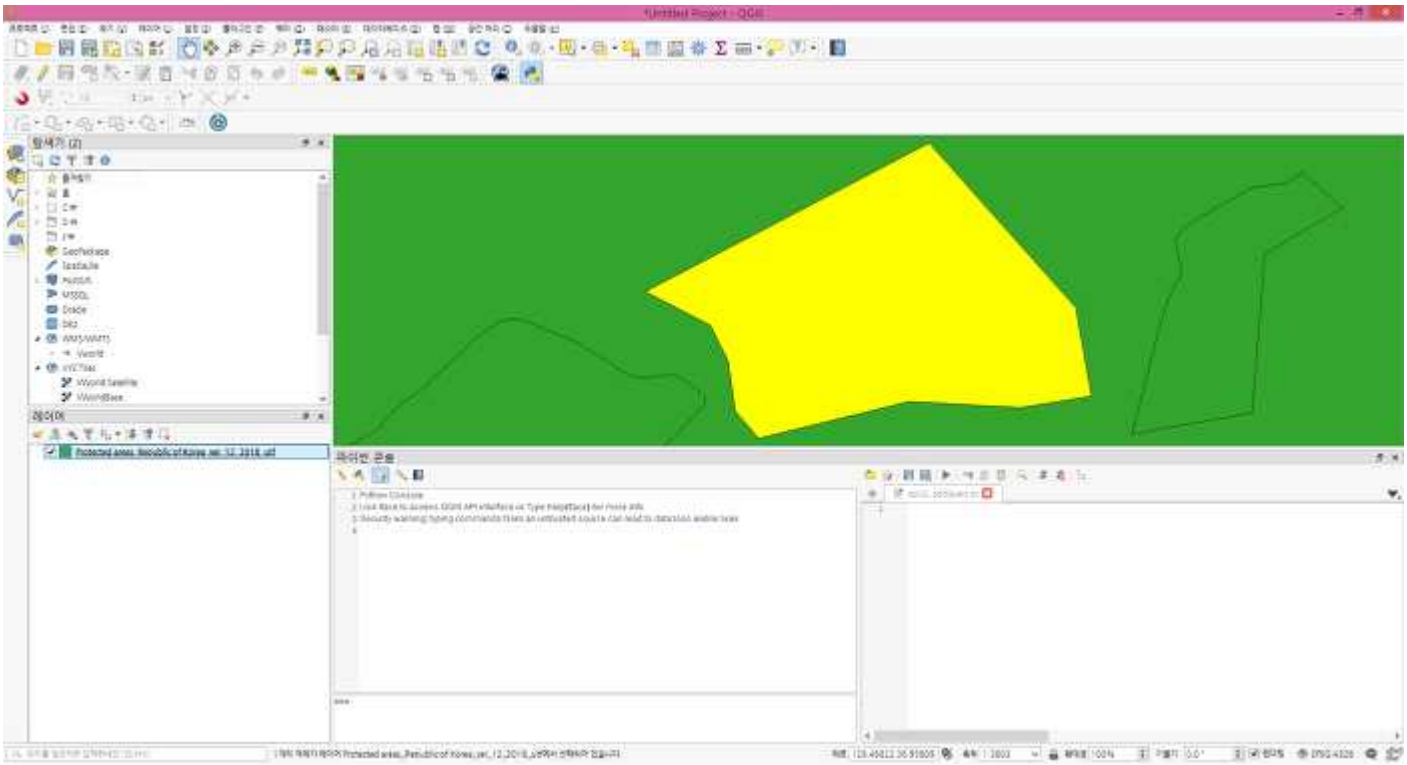


'표현식을 이용해 객체 선택'을 클릭하고,

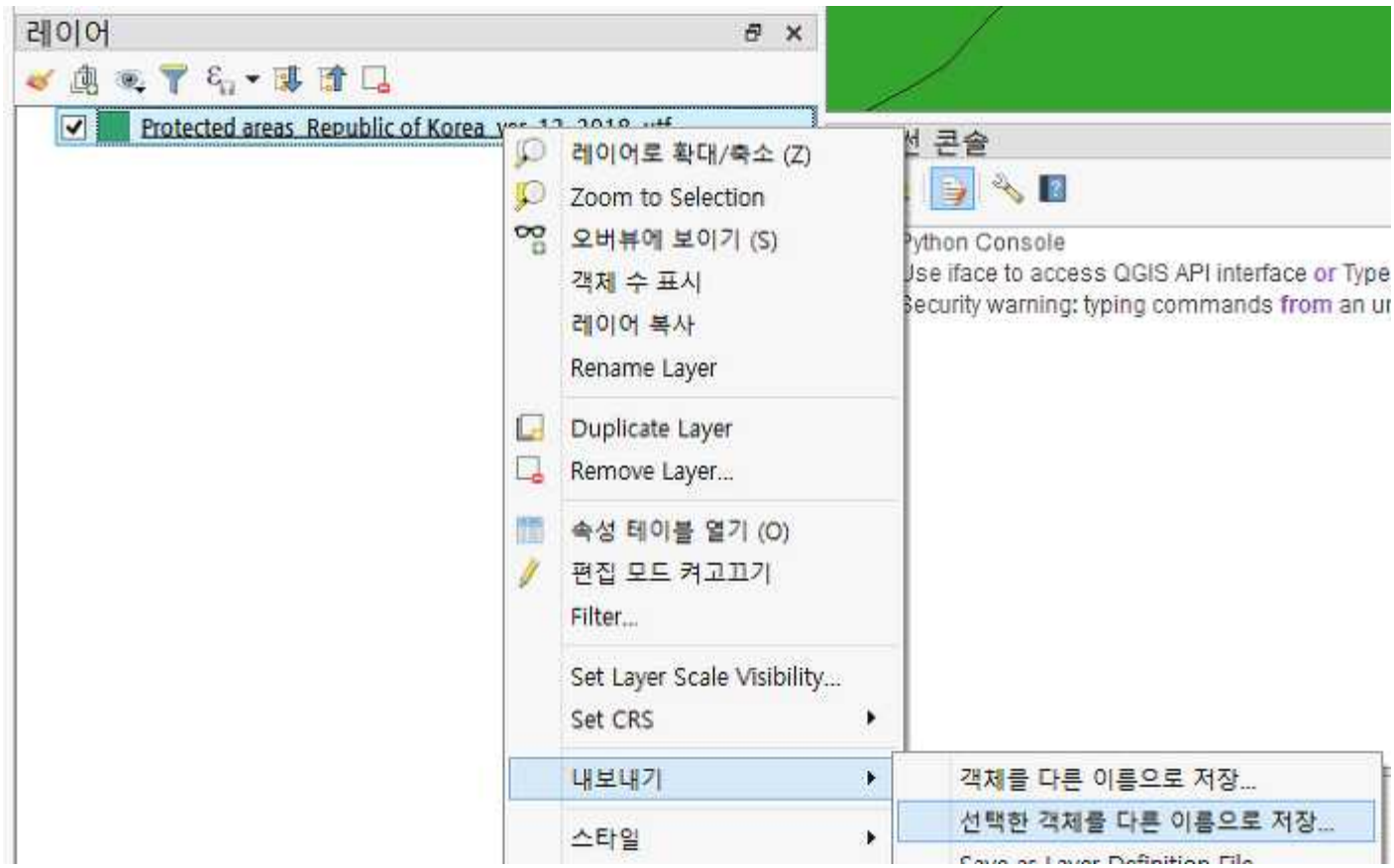
아래 표현식을 통해 소백산 주목군락(천연기념물 제244호) 피처를 선택합니다.



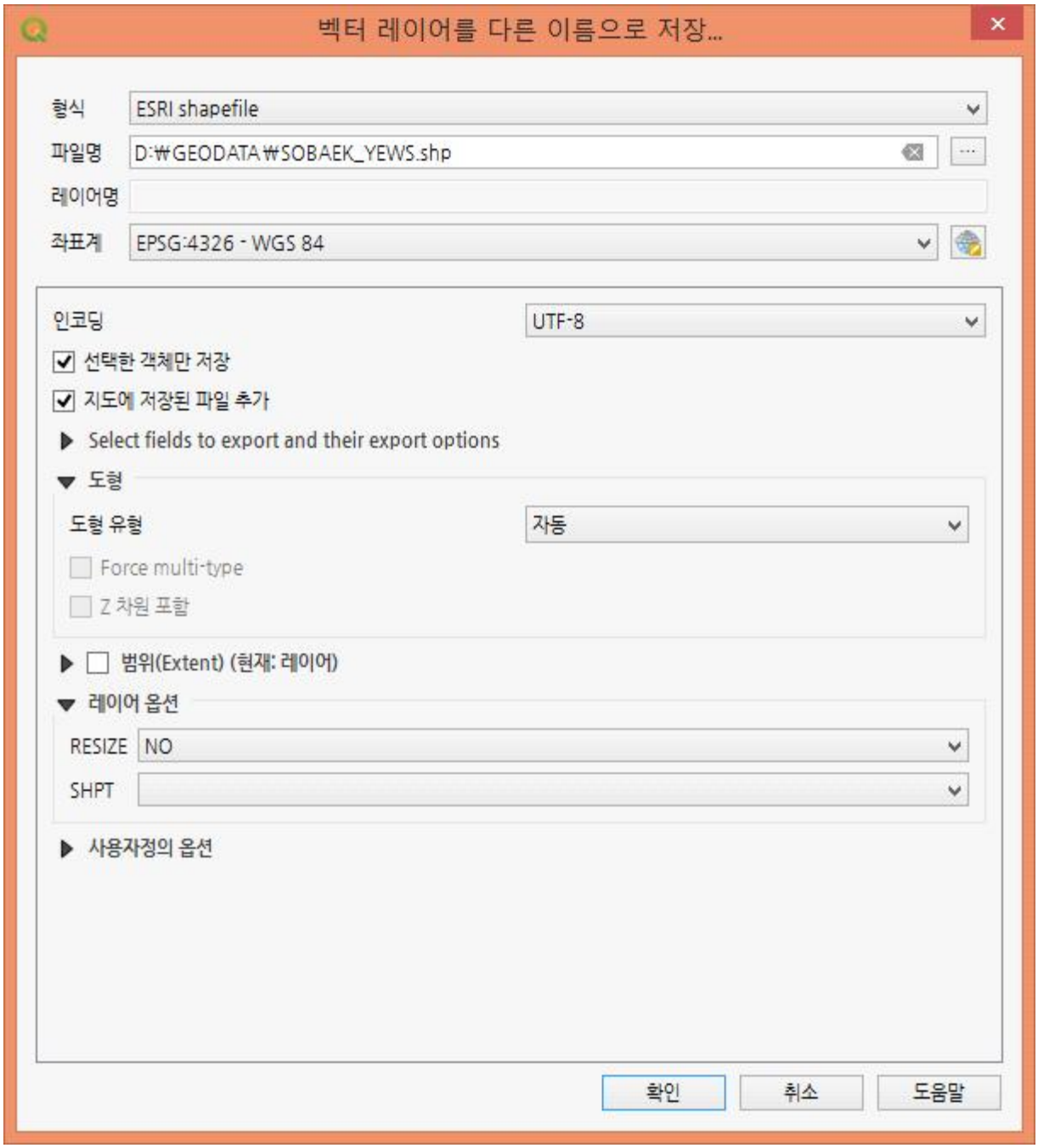
아래 피처가 해당되는데요,



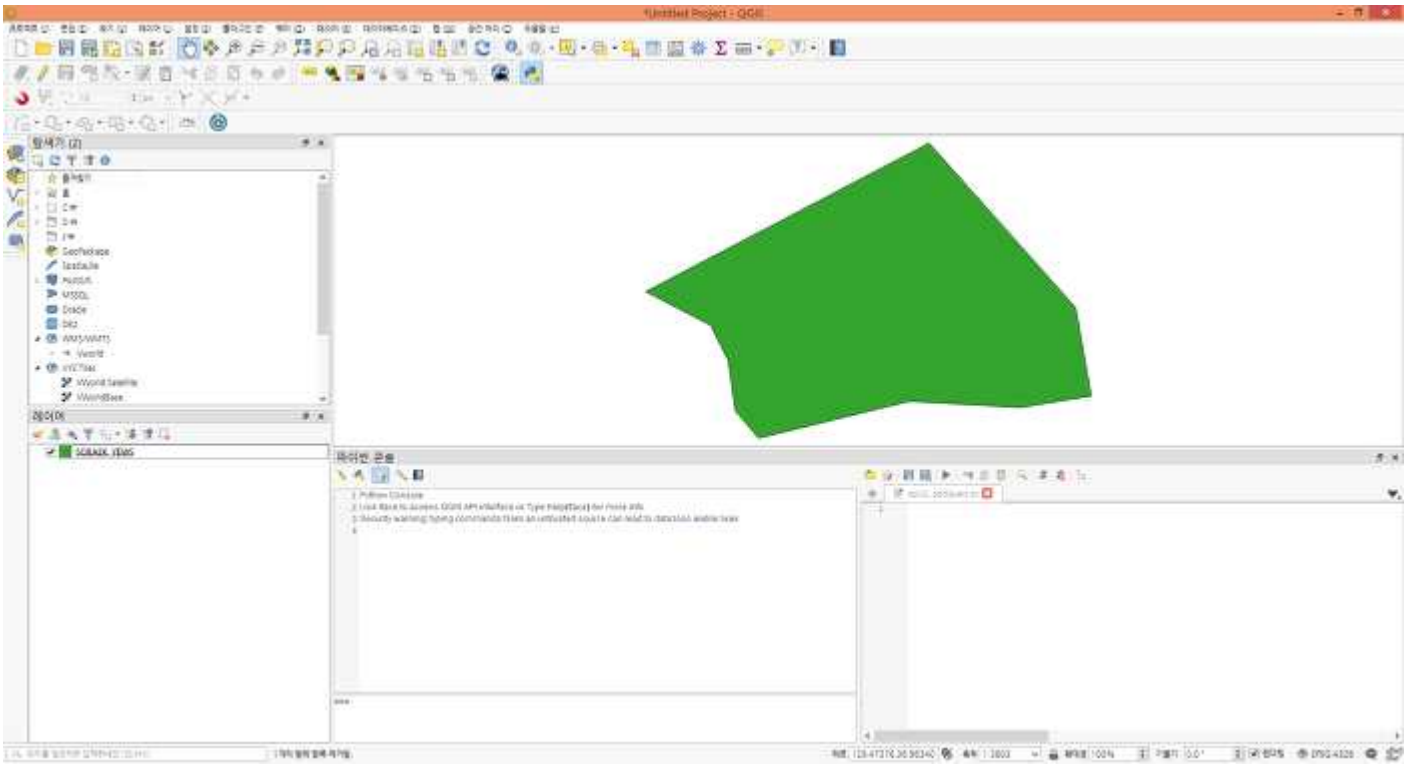
레이어 명을 우클릭하고 '내보내기 >선택한 객체를 다른 이름으로 저장'을 클릭해서,



SOBAEK_YEWS.shp이라는 이름으로 저장해 보겠습니다.



자, 이제 한 개 피처로 구성된 벡터 레이어를 생성했습니다.



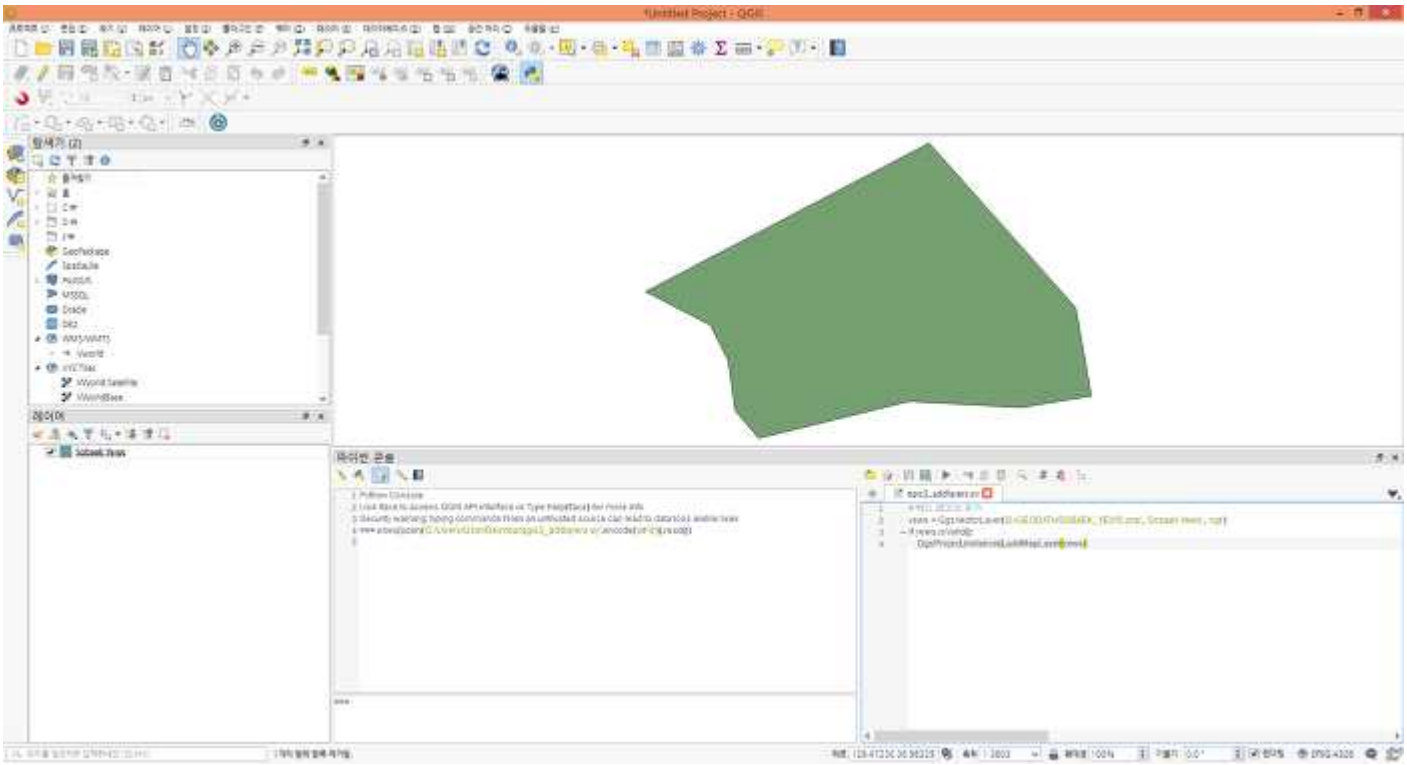
자, 이제 본격적으로 파이썬 콘솔을 사용해 보겠습니다. QGIS 3.4 파이썬 콘솔은 시작할 때 qgis.core와 qgis.gui 모듈을 자동으로 불러옵니다. 따라서 QGIS API를 바로 호출하실 수 있습니다.

```
1 # 벡터 레이어 추가
2 yews = QgsVectorLayer('D:/GEODATA/SOBAEK_YEWS.shp', 'Sobaek Yews', 'ogr')
3 if yews.isValid():
4     QgsProject.instance().addMapLayer(yews)
```

작성한 스크립트 실행은 아래 아이콘을 클릭하면 됩니다.

아래와 같이 벡터 레이어가 추가되었습니다 아래처럼 코드 한 줄로 간결하게 표현할 수도 있습니다.

```
1yews = iface.addVectorLayer('D:/GEODATA/SOBAEK_YEWS.shp', 'Sobaek Yews', 'ogr')
```



QGIS는 벡터 레이어를 불러올때 Single symbol과 임의의 색상을 이용하여 표현합니다.
만약 고정된 레이어 색상을 적용하고자 하시면, 아래와 같이 코드를 추가하시면 됩니다.

```
# 기호 설정
renderer = yews.renderer()
symbol = renderer.symbol()
symbol.setColor(QColor('#33a02c'))
```

QColor 객체는 색상 명칭으로도 생성하실 수 있는데요, 색상 명칭은 아래와 같이 확인 후 적용하실 수 있습니다.

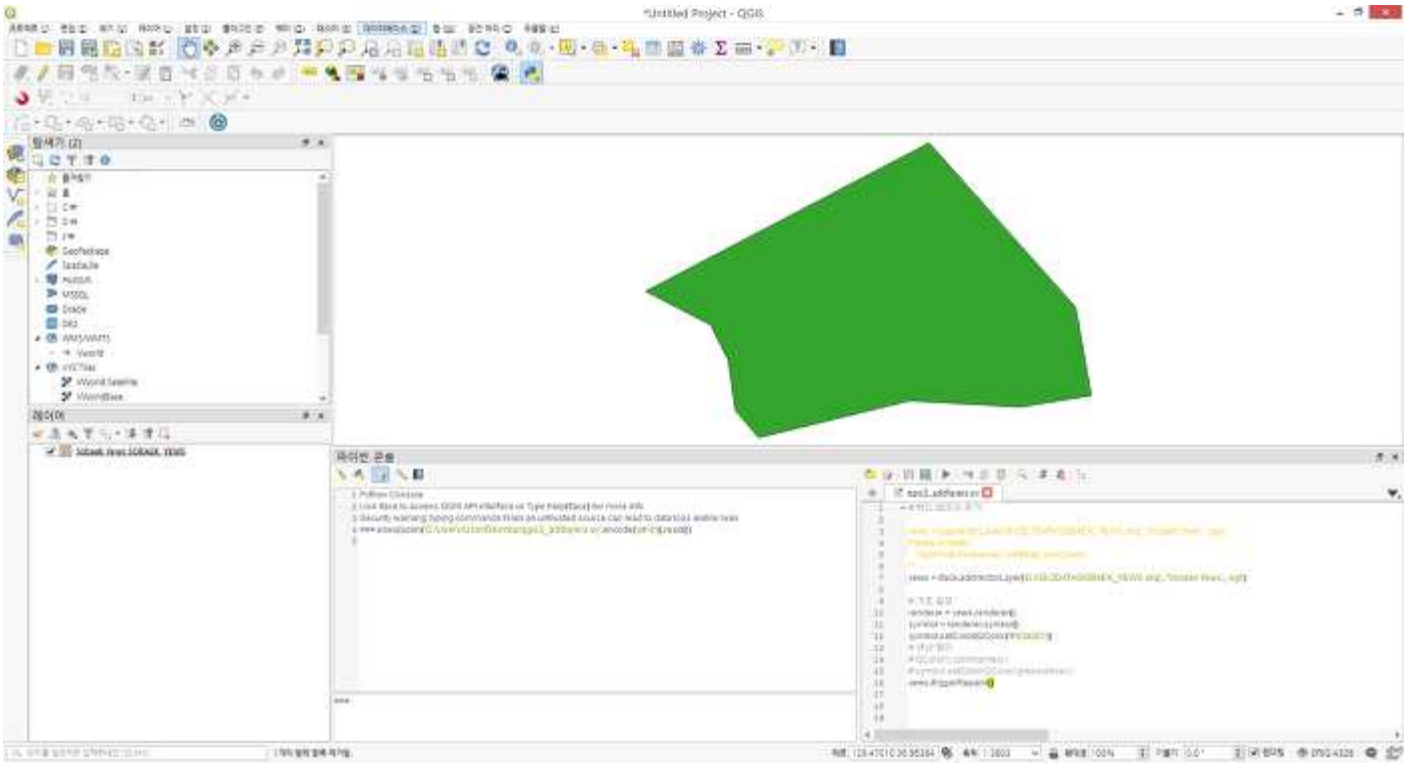
```
# 색상 확인
QColor().colorNames()
symbol.setColor(QColor('greenyellow'))
```

파이썬 콘솔

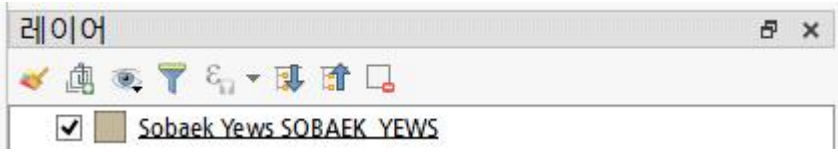
```
24 >>> QColor().colorNames()
25 ['aliceblue', 'antiquewhite', 'aqua', 'aquamarine', 'azure', 'beige', 'bisque', 'black', 'blanchedalmond', 'blue', 'blueviolet', 'brown', 'burlywood', 'cadetblue', 'chartreuse', 'chocolate', 'coral', 'cornflowerblue', 'cornsilk', 'crimson', 'cyan', 'darkblue', 'darkcyan', 'darkgoldenrod', 'darkgray', 'darkgreen', 'darkgrey', 'darkkhaki', 'darkmagenta', 'darkolivegreen', 'darkorange', 'darkorchid', 'darkred', 'darksalmon', 'darkseagreen', 'darkslateblue', 'darkslategray', 'darkslategrey', 'darkturquoise', 'darkviolet', 'deeppink', 'deepskyblue', 'dimgray', 'dimgrey', 'dodgerblue', 'firebrick', 'floralwhite', 'forestgreen', 'fuchsia', 'gainsboro', 'ghostwhite', 'gold', 'goldenrod', 'gray', 'green', 'greenyellow', 'grey', 'honeydew', 'hotpink', 'indianred', 'indigo', 'ivory', 'khaki', 'lavender', 'lavenderblush', 'lawngreen', 'lemonchiffon', 'lightblue', 'lightcoral', 'lightcyan', 'lightgoldenrodyellow', 'lightgray', 'lightgreen', 'lightgrey', 'lightpink', 'lightsalmon', 'lightseagreen', 'lightskyblue', 'lightslategray', 'lightslategrey', 'lightsteelblue', 'lightyellow', 'lime', 'limegreen', 'linen', 'magenta', 'maroon', 'mediumaquamarine', 'mediumblue', 'mediumorchid', 'mediumpurple', 'mediumseagreen', 'mediumslateblue', 'mediumspringgreen', 'mediumturquoise', 'mediumvioletred', 'midnightblue', 'mintcream', 'mistyrose', 'moccasin', 'navajowhite', 'navy', 'oldlace', 'olive', 'olivedrab', 'orange', 'orangered', 'orchid', 'palegoldenrod', 'palegreen', 'paleturquoise', 'palevioletred', 'papayawhip', 'peachpuff', 'peru', 'pink', 'plum', 'powderblue', 'purple', 'red', 'rosybrown', 'royalblue', 'saddlebrown', 'salmon', 'sandybrown', 'seagreen', 'seashell', 'sienna', 'silver', 'skyblue', 'slateblue', 'slategray', 'slategrey', 'snow', 'springgreen', 'steelblue', 'tan', 'teal', 'thistle', 'tomato', 'transparent', 'turquoise', 'violet', 'wheat', 'white', 'whitesmoke', 'yellow', 'yellowgreen']
26
```

이제 아래 코드까지 추가하시면 이제 지정한 색상으로 레이어가 표현됩니다. 간단하죠?!

```
1yews.triggerRepaint()
```

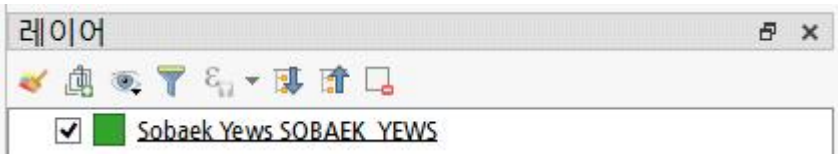


그런데 레이어 색상은 변경되었지만 범례는 여전히 임의 색상이 적용되고 있습니다. 이건 고쳐야겠죠?!



아래와 같이 코드를 추가해주시면 되겠습니다.

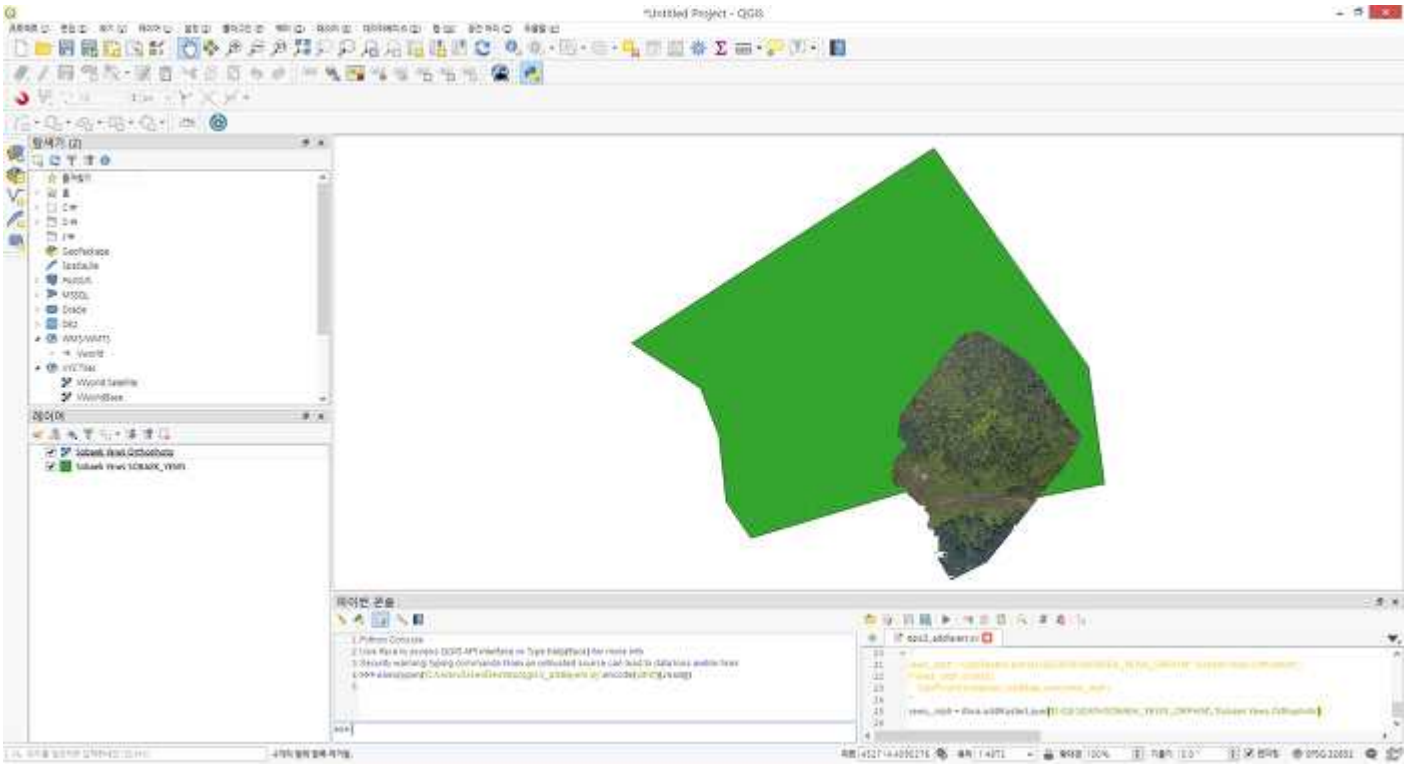
```
1iface.layerTreeView().refreshLayerSymbology(yews.id())
```



이번에는 레스터 레이어를 추가해 보겠습니다. 방식은 벡터 레이어와 추가와 거의 동일합니다.
여기서는 해당 지역을 촬영한 드론 정사영상(OpenDroneMap/WebODM으로 처리)을 사용하였습니다.

```
# 래스터 레이어 추가
'''
yews_orph = QgsRasterLayer('D:/GEODATA/SOBAEK_YEWS_ORPH.tif', 'Sobaek Yews Orthophoto')
if yews_orph.isValid():
    QgsProject.instance().addMapLayer(yews_orph)
'''
yews_orph = iface.addRasterLayer('D:/GEODATA/SOBAEK_YEWS_ORPH.tif', 'Sobaek Yews Orthophoto')
```

이상으로 파이썬 콘솔에서 벡터/래스터 레이어를 추가해 봤습니다.

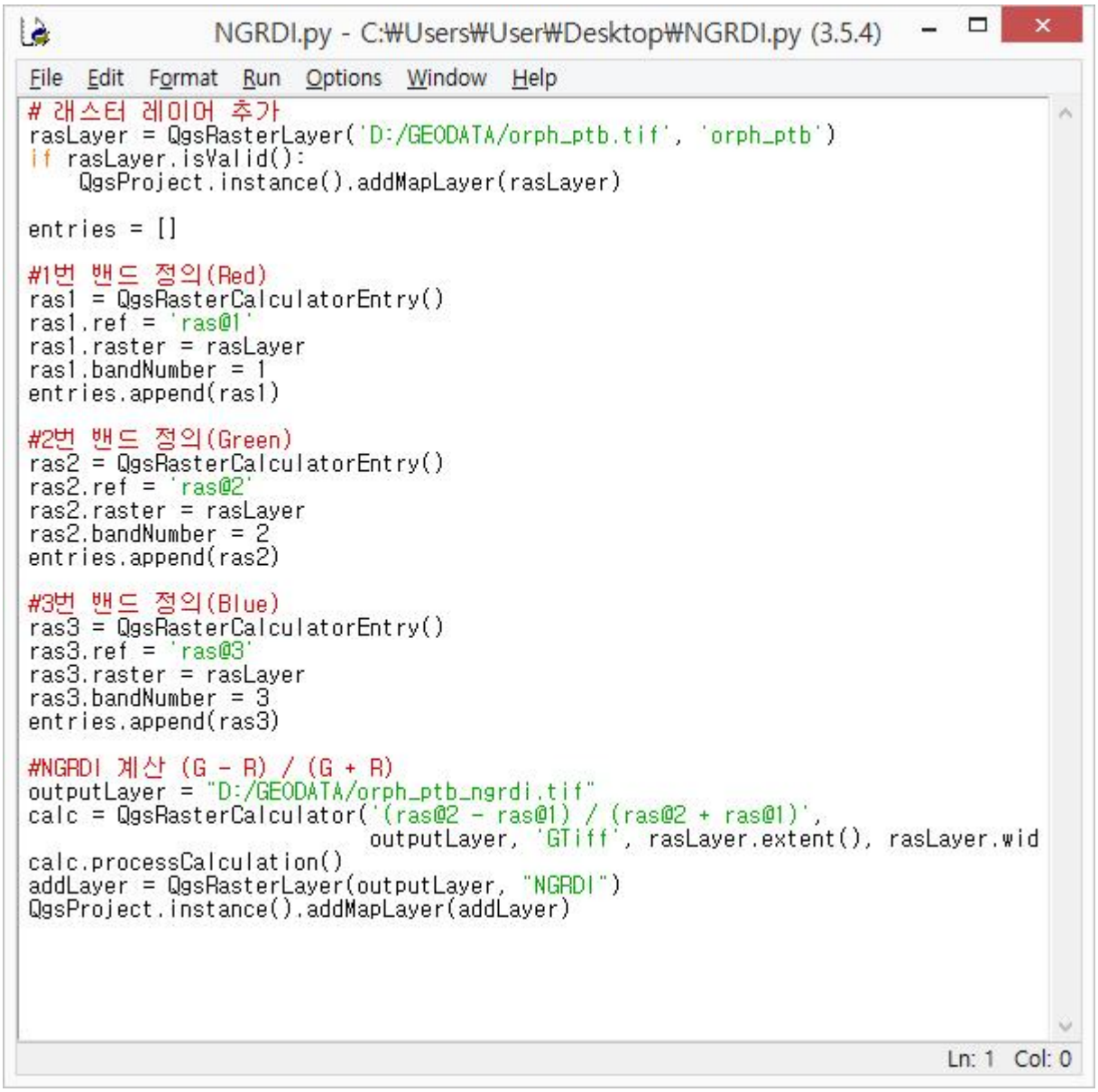


QGIS 3.4 Orfeo ToolBox(OTB) - (8) 가시광선 식생지수 계산하기 (파이썬 콘솔)

이번 글은 파이썬 콘솔에서 무인기 영상의 식생지수를 계산해 보도록 하겠습니다.
앞서 OTB BandMath 도구를 사용했던 부분을 파이썬으로 구현했다고 보시면 됩니다.

QGIS 3.4 파이썬 활용과 관련해서는 아래 글을 참조하시기 바랍니다.
PyQGIS Developer Cookbook | https://docs.qgis.org/testing/en/docs/pyqgis_developer_cookbook/index.html

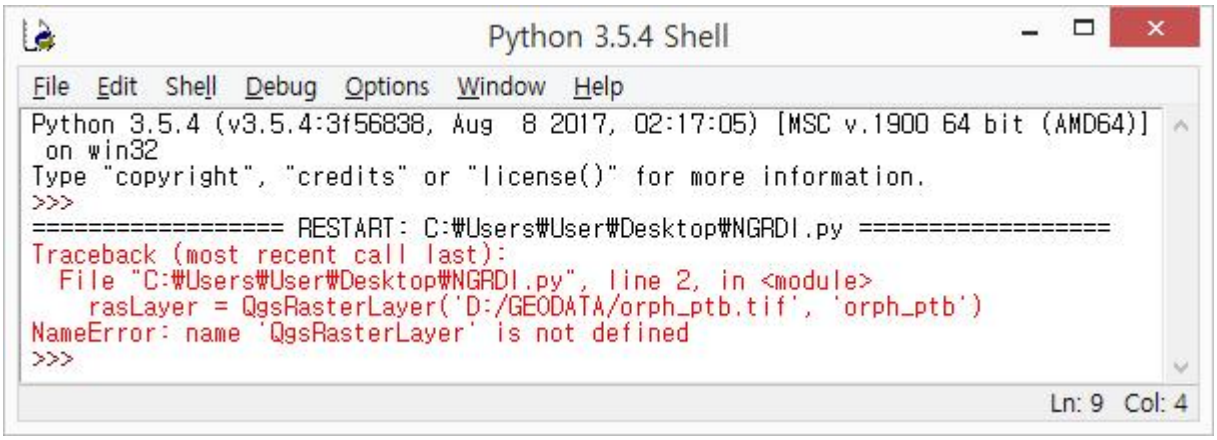
저는 식생지수의 일종인 NDGRI를 계산하기 위한 파이썬 코드를 NGRDI.py로 작성했습니다.



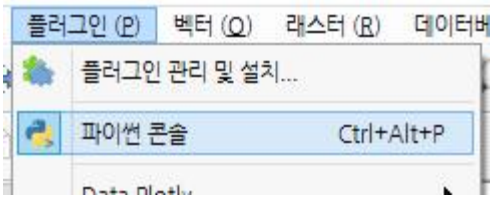
원본 파이썬 코드는 아래와 같습니다.

```
# 래스터 레이어 추가
rasLayer = QgsRasterLayer('D:/GEODATA/orph_pta.tif', 'orph_pta')
if rasLayer.isValid():
    QgsProject.instance().addMapLayer(rasLayer)

entries = []

#1번 밴드 정의(Red)
ras1 = QgsRasterCalculatorEntry()
ras1.ref = 'ras@1'
ras1.raster = rasLayer
ras1.bandNumber = 1
entries.append(ras1)

#2번 밴드 정의(Green)
ras2 = QgsRasterCalculatorEntry()
ras2.ref = 'ras@2'
ras2.raster = rasLayer
ras2.bandNumber = 2
entries.append(ras2)

#3번 밴드 정의(Blue)
ras3 = QgsRasterCalculatorEntry()
ras3.ref = 'ras@3'
ras3.raster = rasLayer
ras3.bandNumber = 3
entries.append(ras3)

#NGRDI 계산 (G - R) / (G + R)
outputLayer = "D:/GEODATA/orph_pta_ngrdi.tif"
calc = QgsRasterCalculator('(ras@2 - ras@1) / (ras@2 + ras@1)',
                           outputLayer, 'GTiff', rasLayer.extent(), rasLayer.width(), rasLayer.height(), e
ntries)
calc.processCalculation()
addLayer = QgsRasterLayer(outputLayer, "NGRDI")
QgsProject.instance().addMapLayer(addLayer)

```

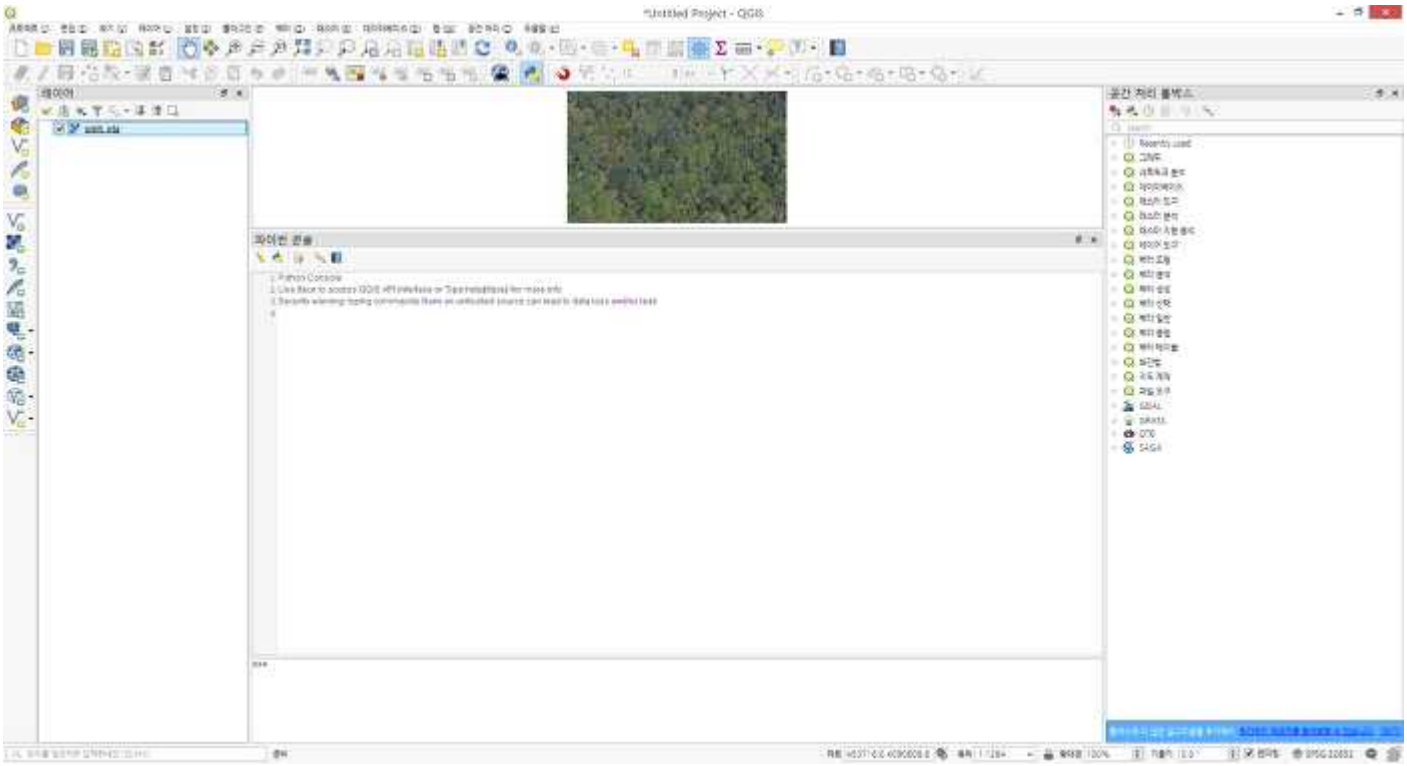
IDLE에서 해당 코드를 실행하면 당연히 오류가 나겠죠?!. QGIS 3.4 파이썬 콘솔에서 동작시켜 보겠습니다.



QGIS 3.4를 실행하고 '플러그인 >파이썬 콘솔'을 실행합니다.

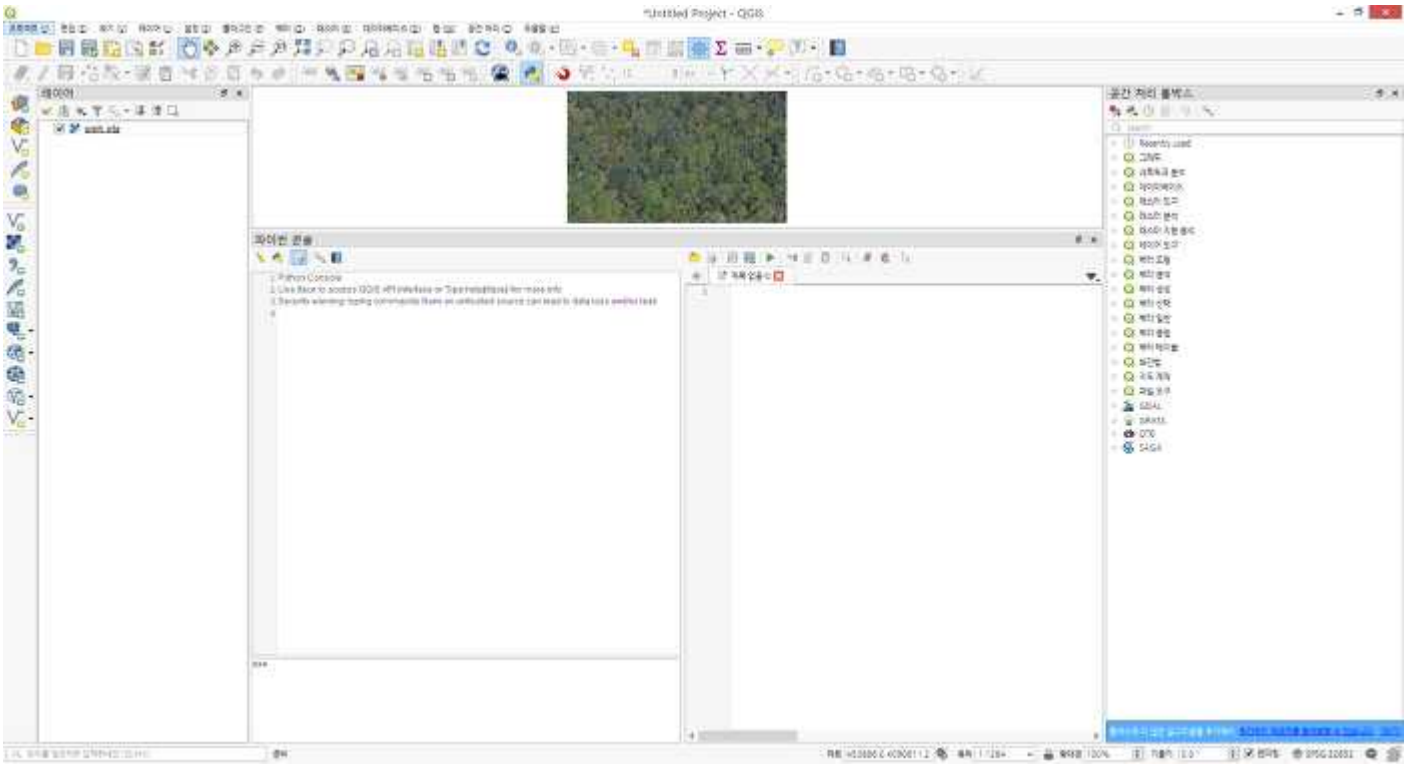


아래와 같이 파이썬 콘솔 창이 실행되는데요,



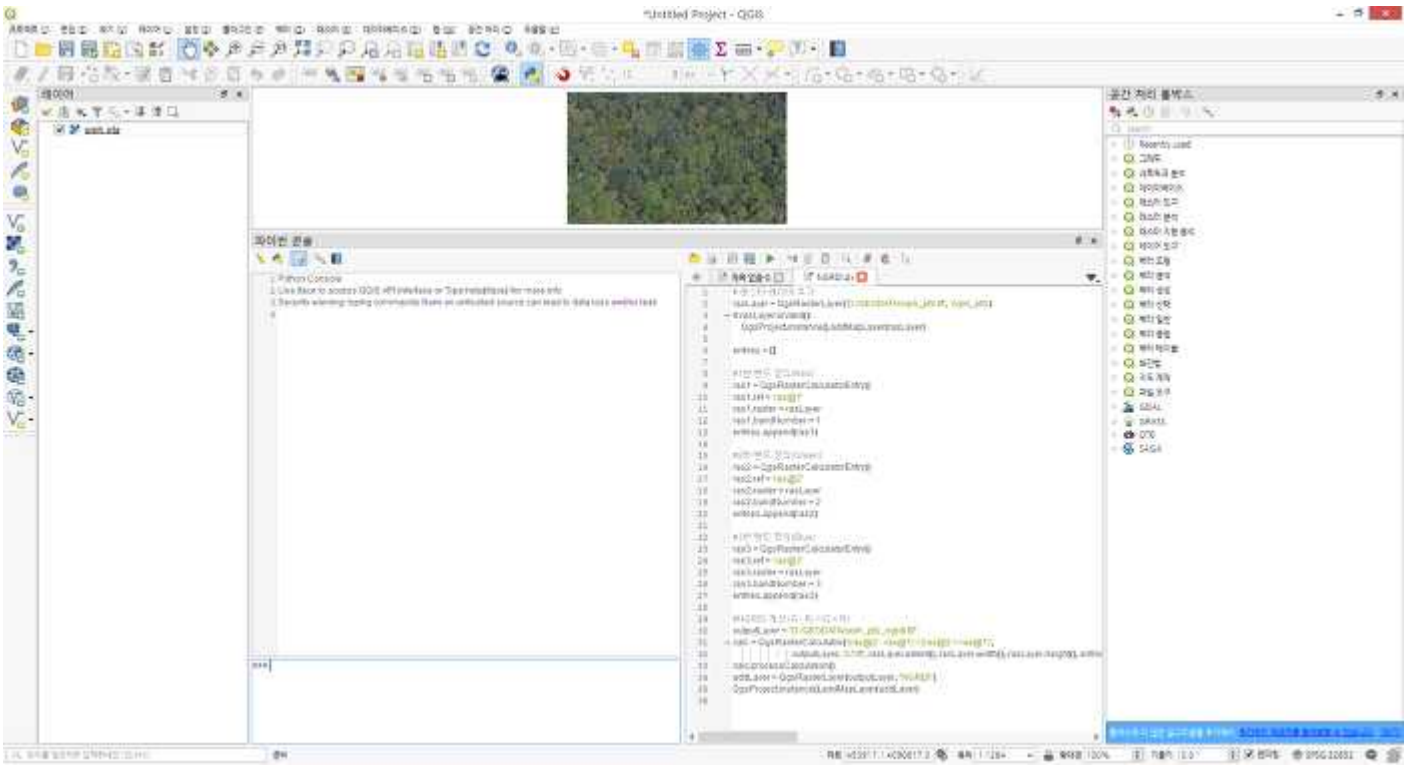
파이썬 콘솔 창의 상단 메뉴에서 '편집기 보이기' 버튼을 클릭하면,

아래와 같이 화면이 분할됩니다.



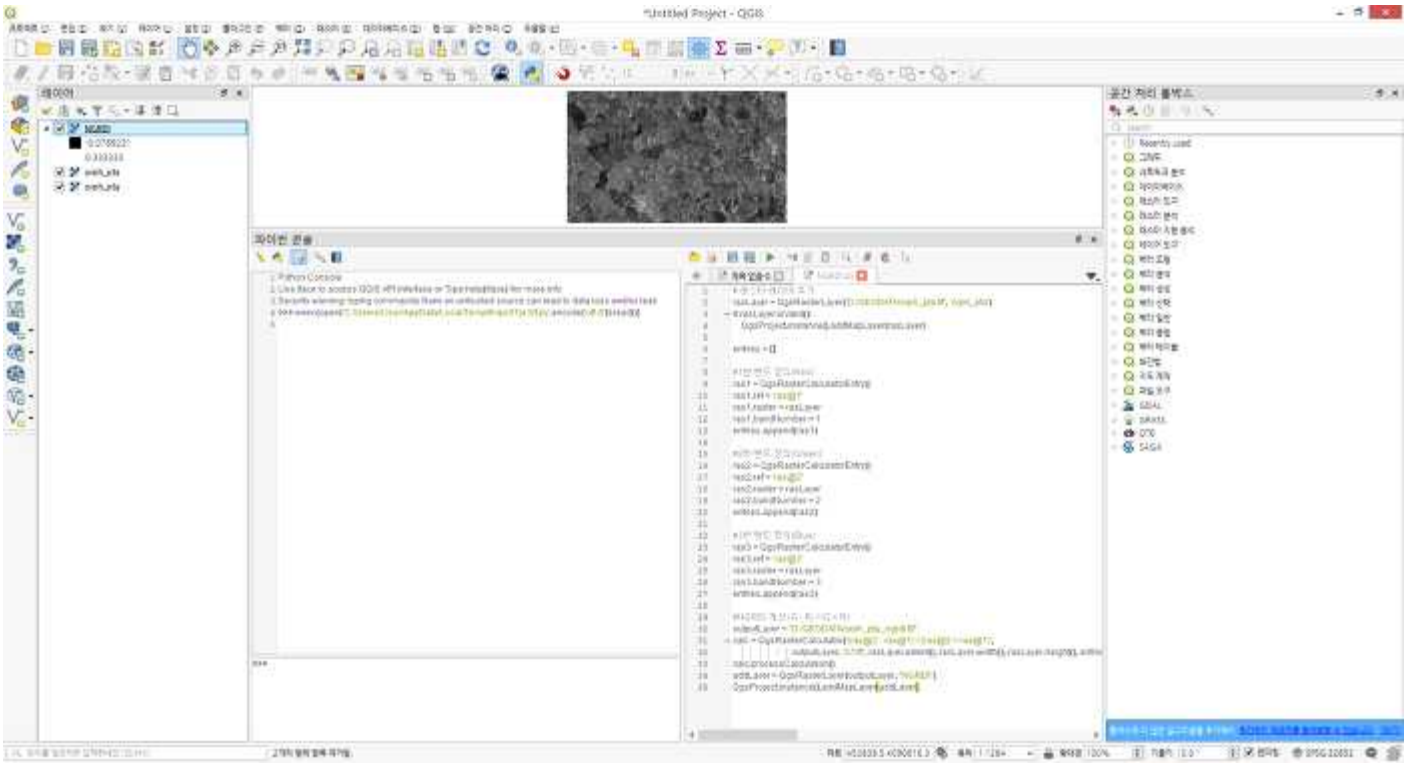
화면 우측에서 '스크립트 열기'를 클릭하고 앞서 작성한 NGRDI.py 파일을 추가해 줍니다.

코드가 추가되었습니다.



이제 '스크립트 실행' 버튼을 클릭해볼까요?!

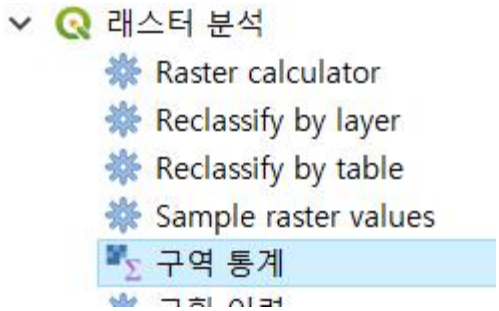
아래와 같이 NGRDI 레이어가 자동 계산되어 추가되었습니다.



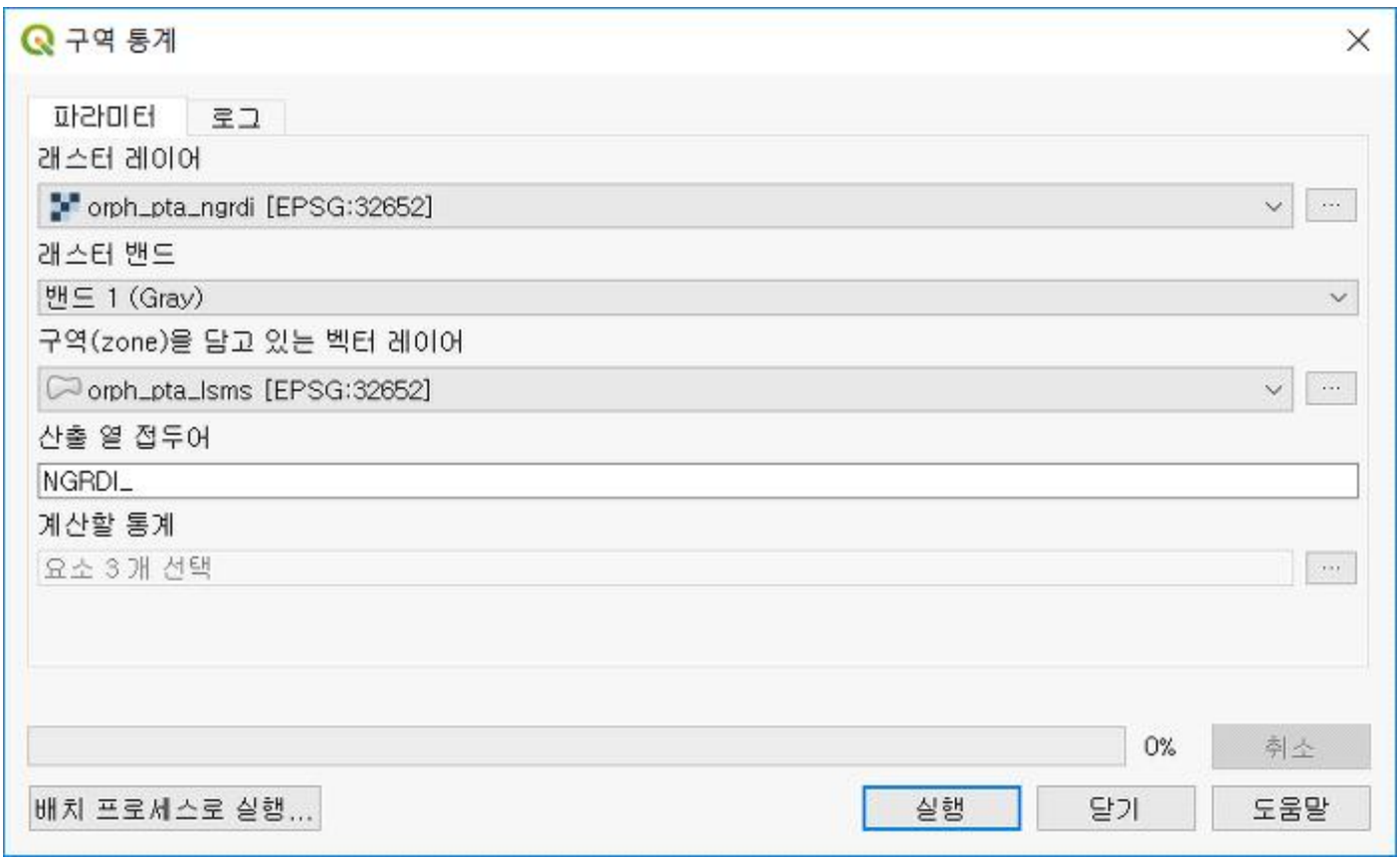
QGIS 3.4 Orfeo ToolBox(OTB) - (9) 머신러닝 분류기에 피처 추가하기

이번 글은 앞서 만든 분류기 모델을 개선하기 위해 피처를 추가해 보도록 하겠습니다.
식생지수의 일종인 NGRDI 값의 세그먼트별 통계치(평균, 표준편차)를 사용하고자 합니다.

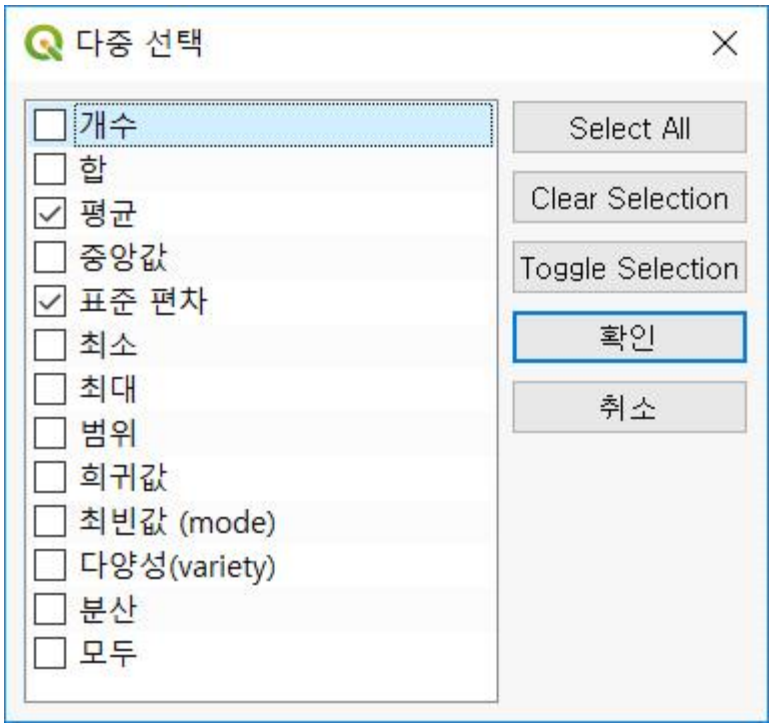
'래스터 분석 >구역 통계'를 실행합니다.



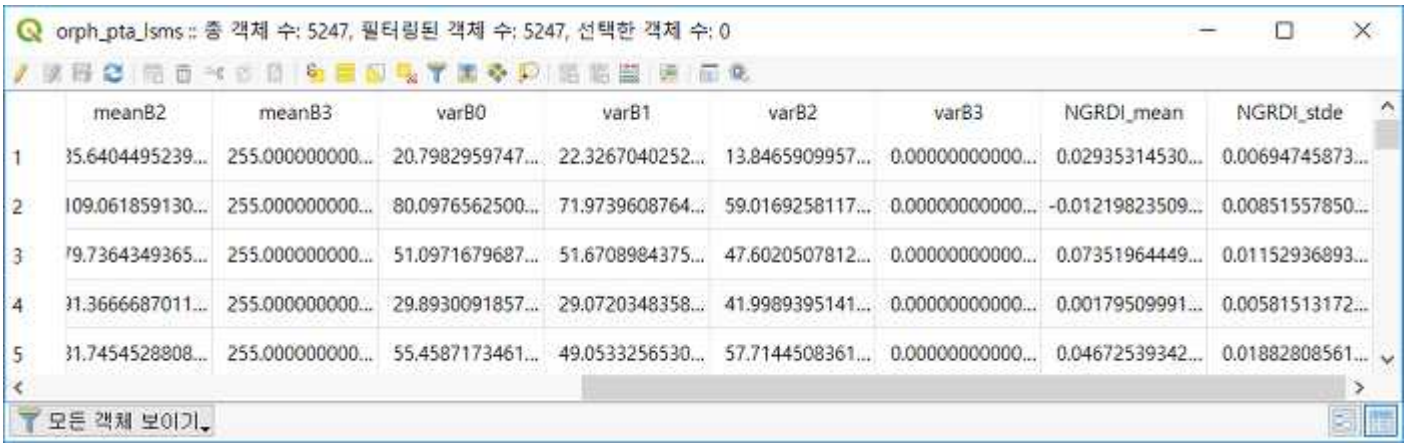
구역 통계 창은 아래와 같은데요, 래스터 레이어는 NGRDI를, 구역(zone)을 담고 있는 벡터 레이어는 세그먼트를 지정합니다. 산출 열 접두어는 생성될 통계 값 필드명에 적용될 접두어로 여기서는 'NGRDI_'를 입력했습니다.



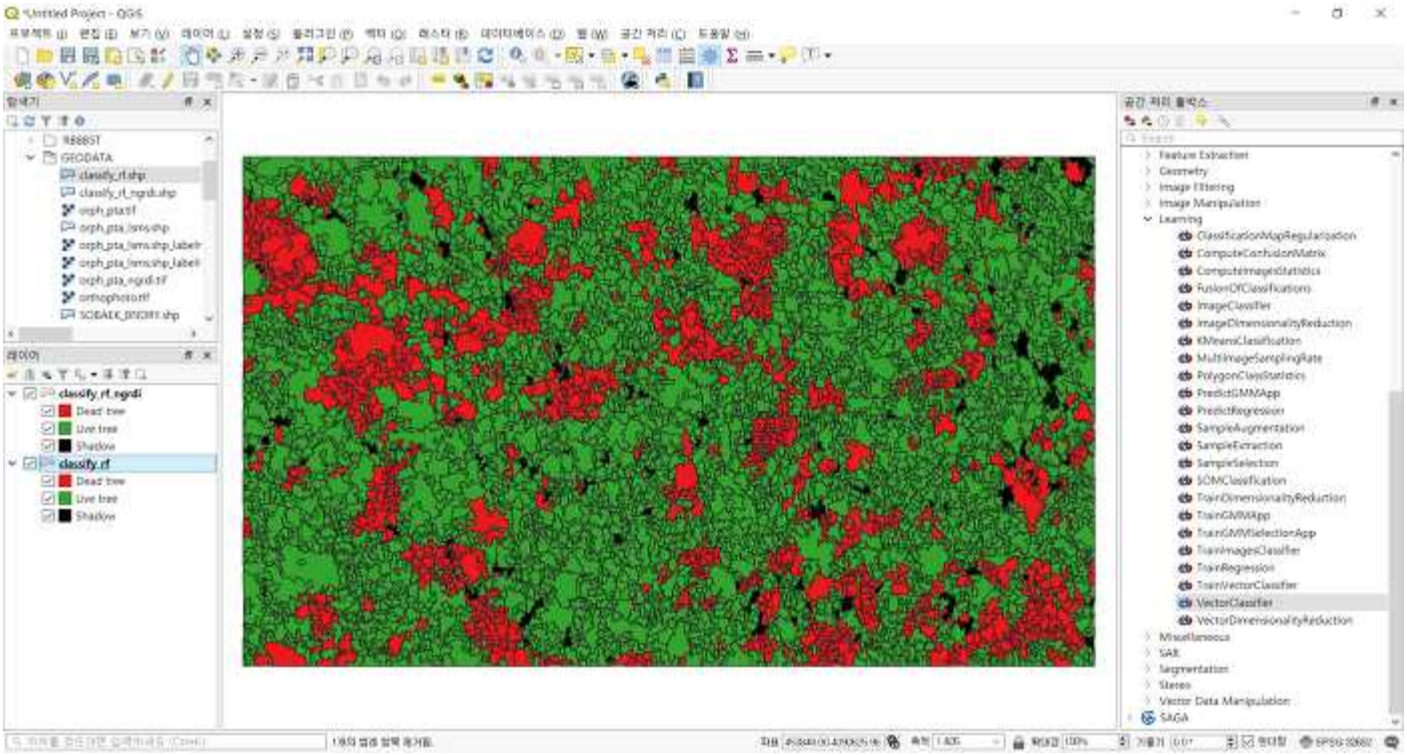
계산할 통계를 클릭하면 다중 선택 창이 실행되는데요, 여기서는 평균, 표준 편차를 선택하겠습니다.



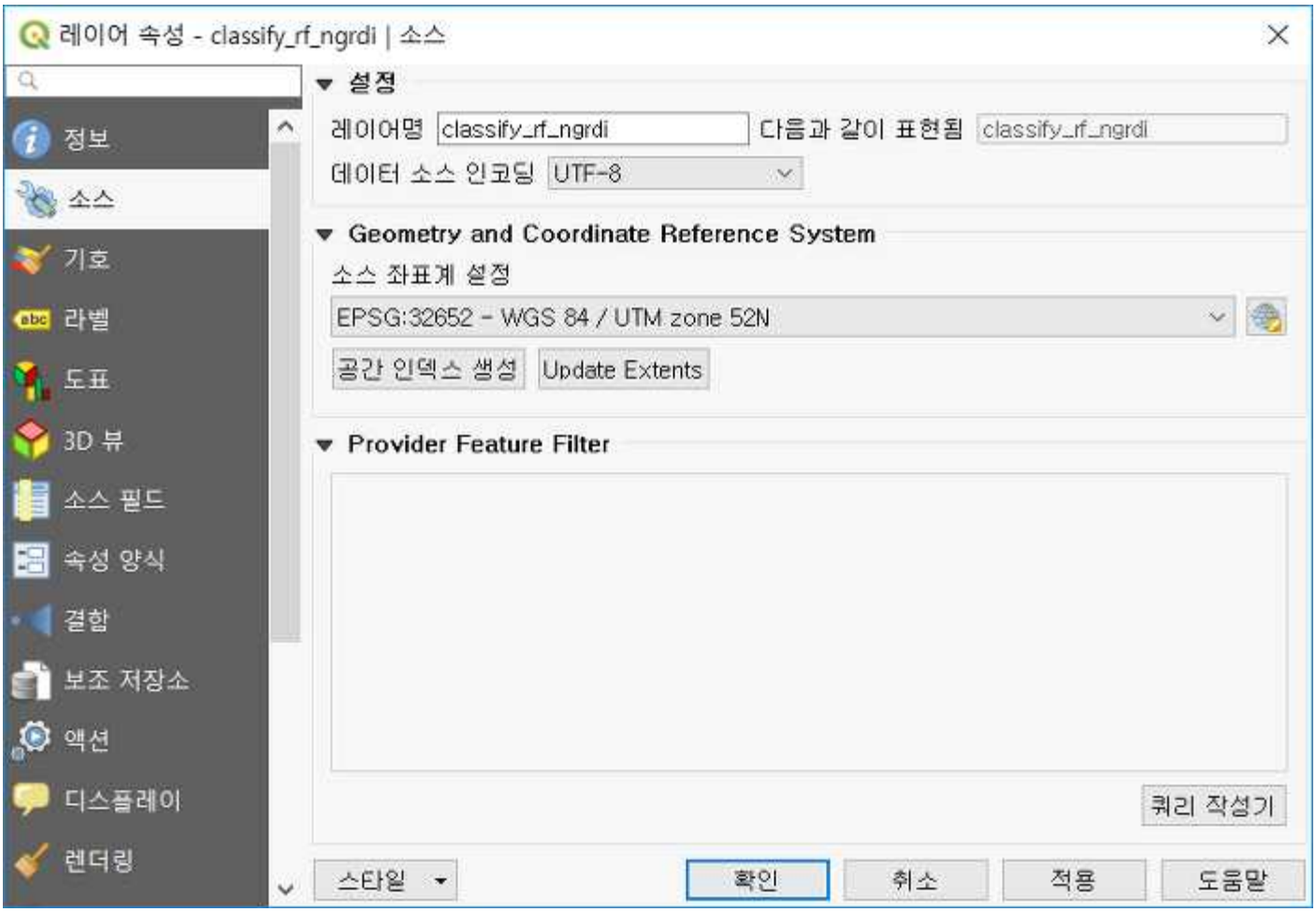
이제 세그먼테이션 속성 테이블에 NGRDI_mean, NGRDI_stde가 추가되었습니다. 분류기 모델을 다시 훈련시켜 보겠습니다.



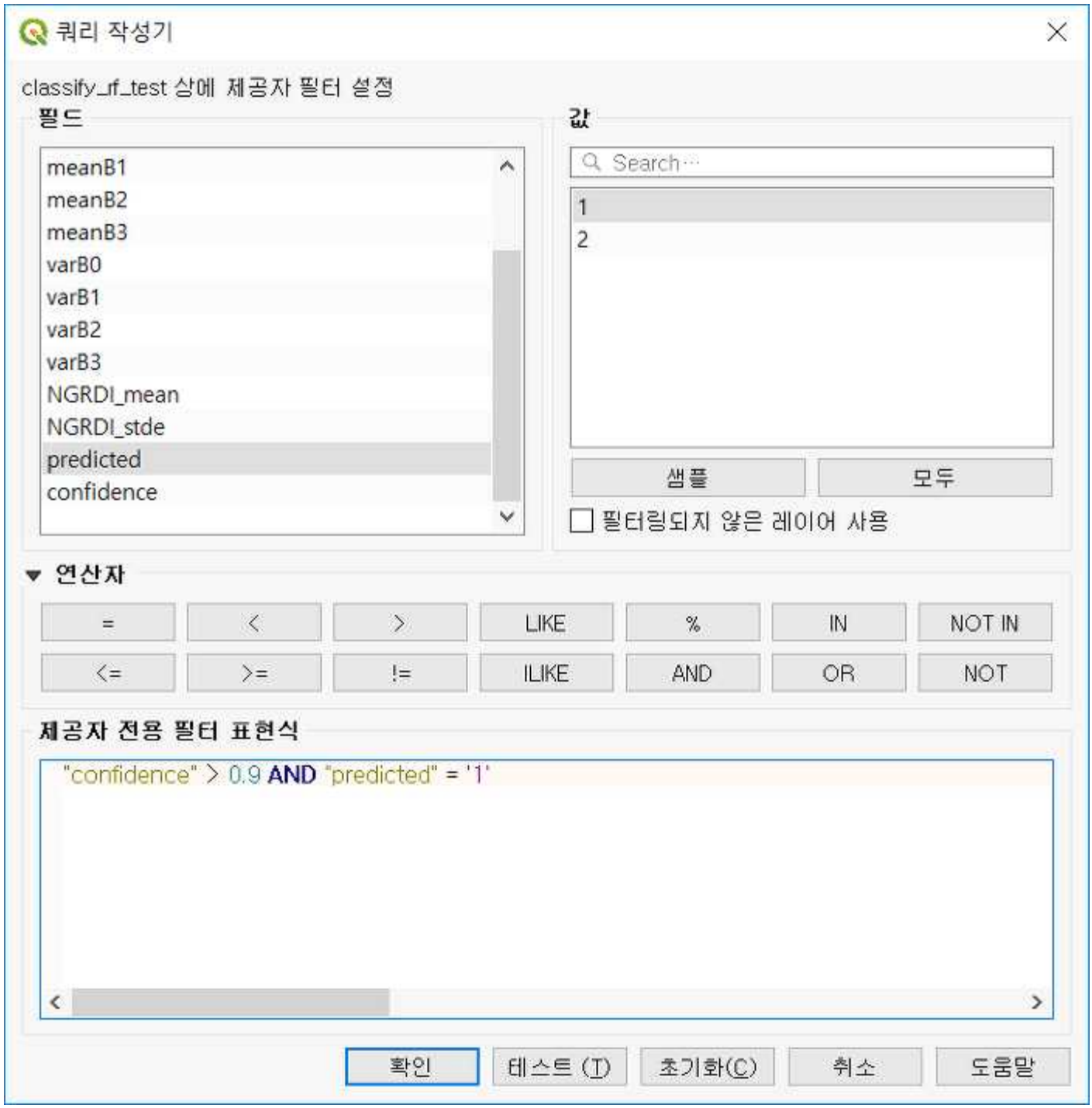
새로 훈련시킨 모델로 분류한 결과입니다.
Confidence map 필드값을 통해 투표 비율이 높은 dead tree 피처만 선택해 보겠습니다.



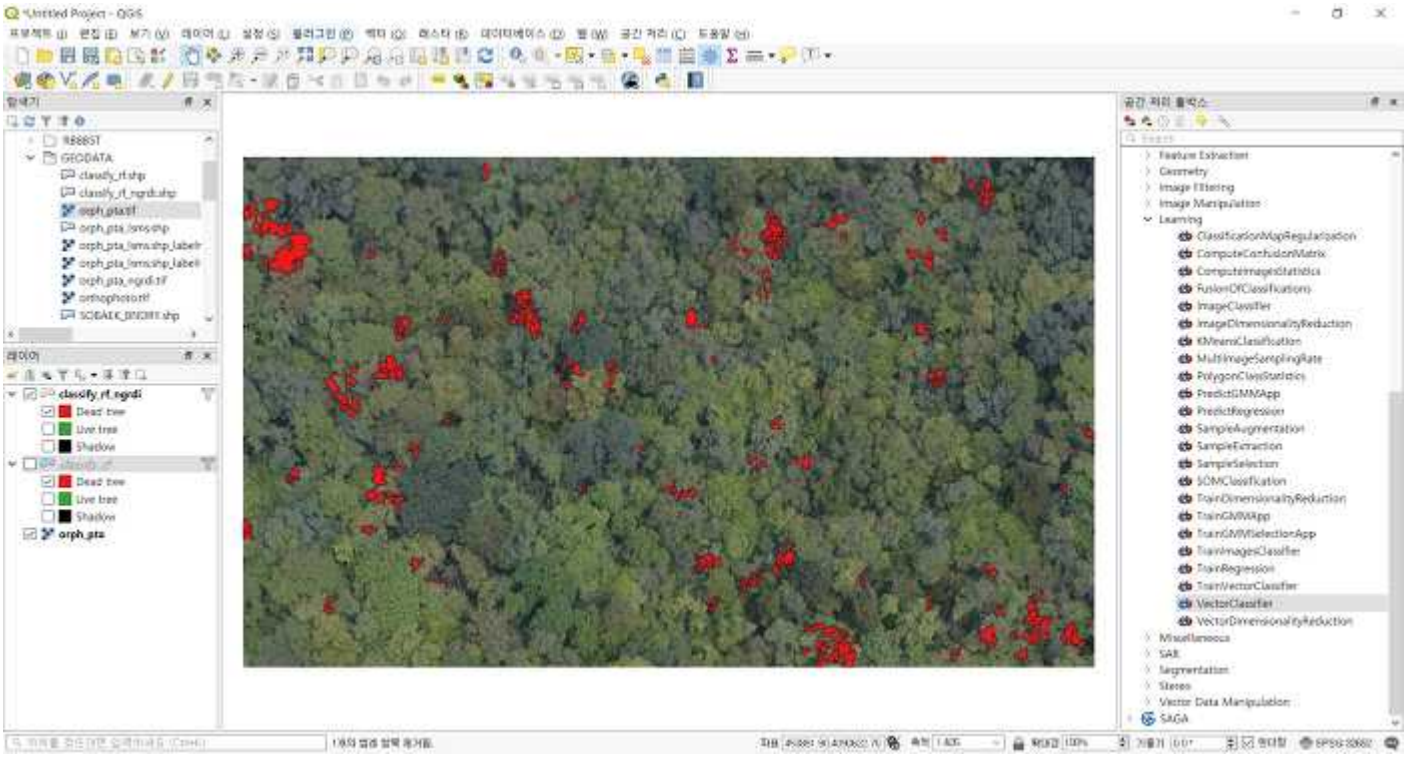
분류결과 레이어 속성에서 '소스 >Provider Feature Filter'에서 '쿼리 작성기'를 클릭합니다.



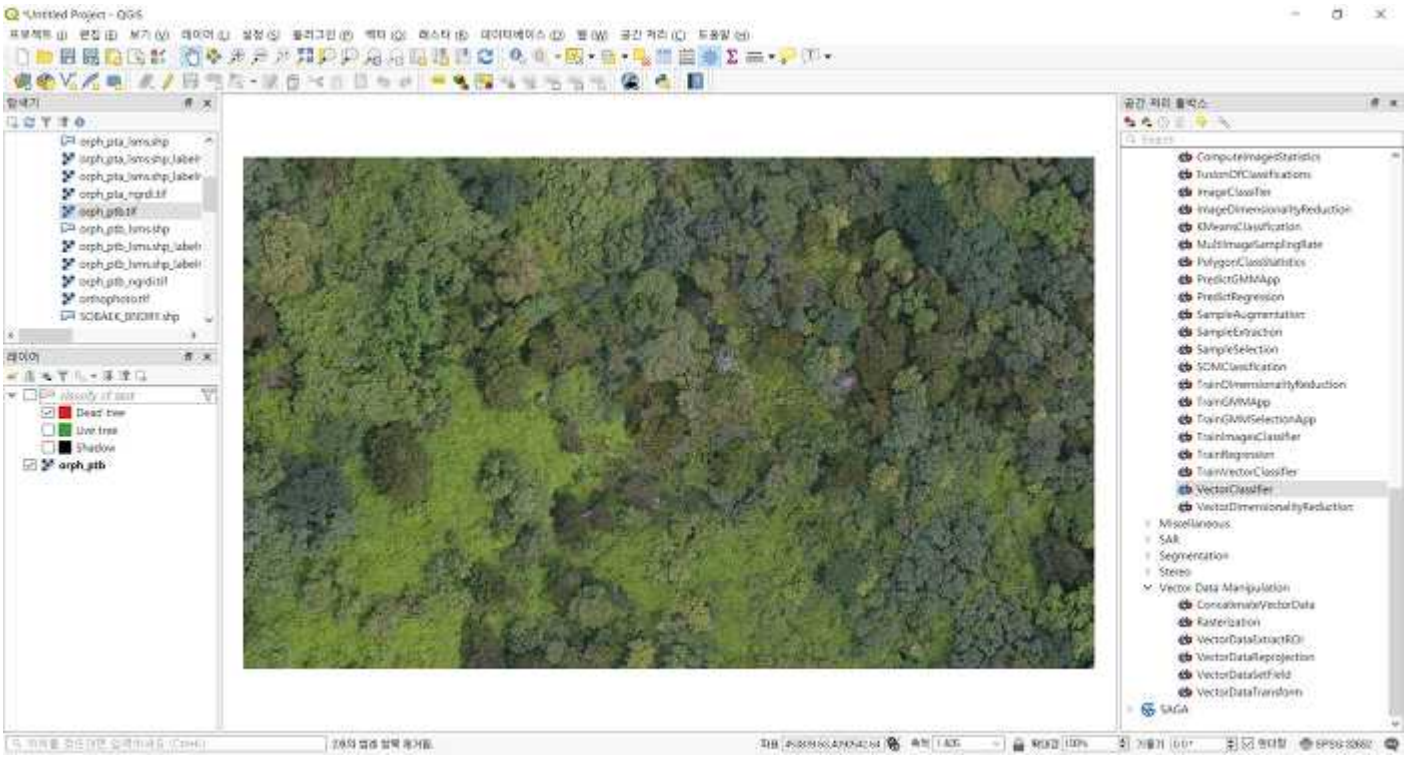
아래와 같이 Confidence가 0.9 이상이면서 predicted가 1(dead tree)인 피처를 선택해 보겠습니다.



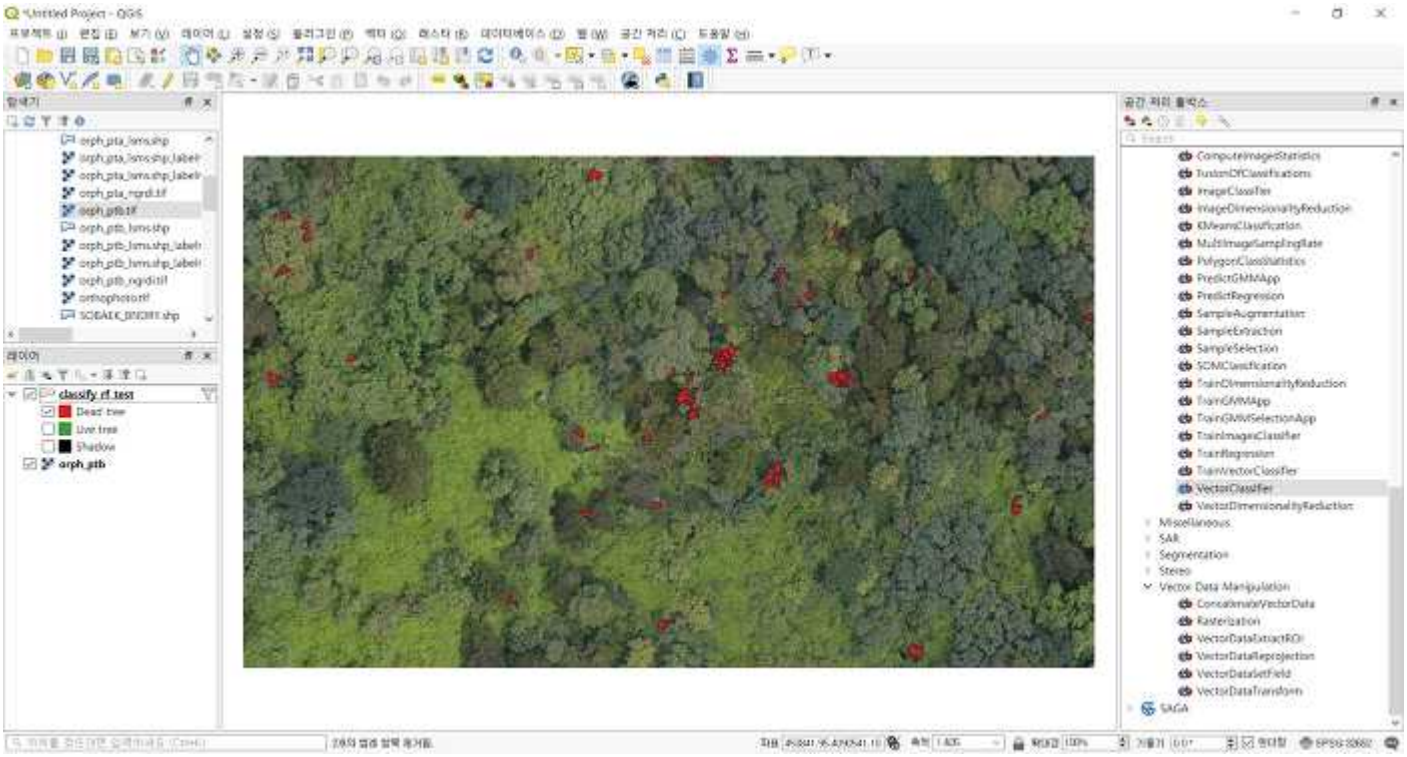
아래와 같이 피처가 선택되었습니다.



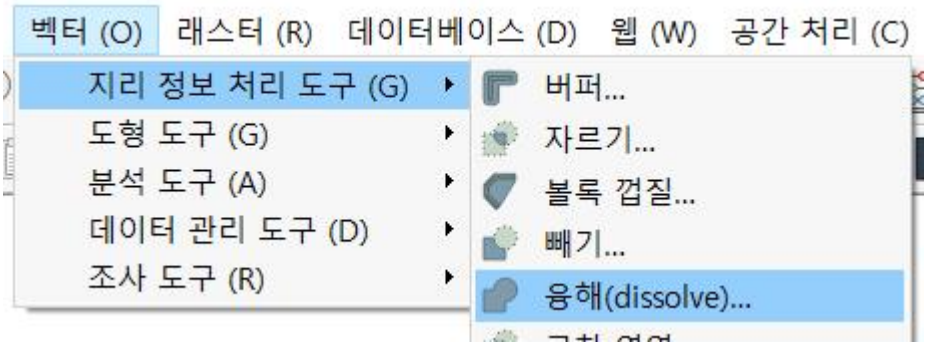
위 영상은 훈련 데이터로 사용되었던 영역이므로, 무인기 영상에서 다른 지역을 테스트해 보겠습니다.



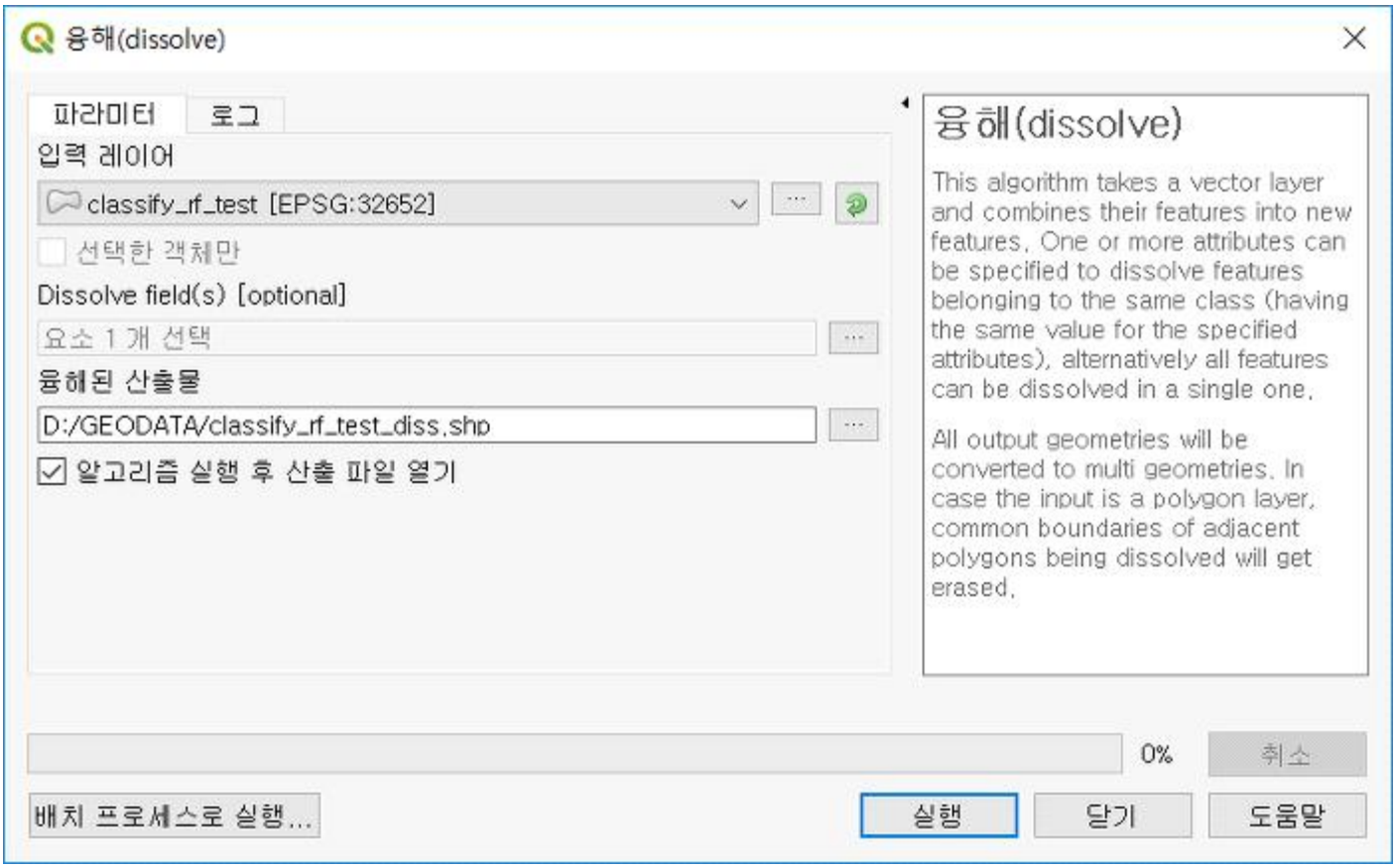
결과는 아래와 같습니다.



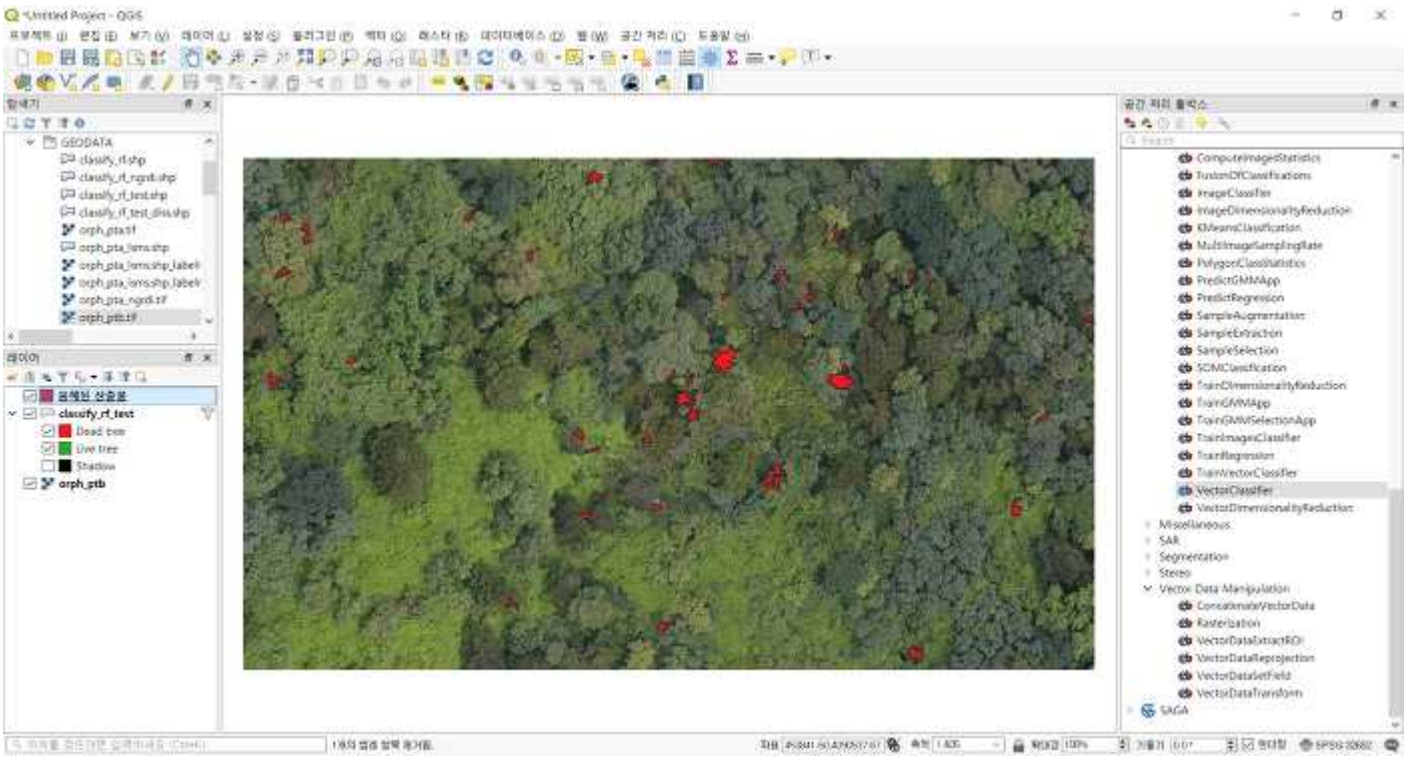
세그먼트는 '벡터 >지리 정보 처리 도구 >융해(dissolve)'를 통해 인접한 피처들을 병합하실 수 있습니다.



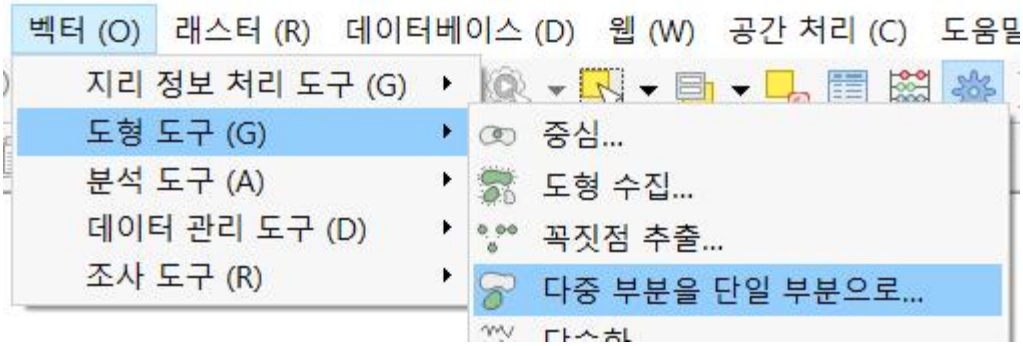
predicted 필드값을 기준으로 융해(dissolve)를 적용해 보겠습니다.



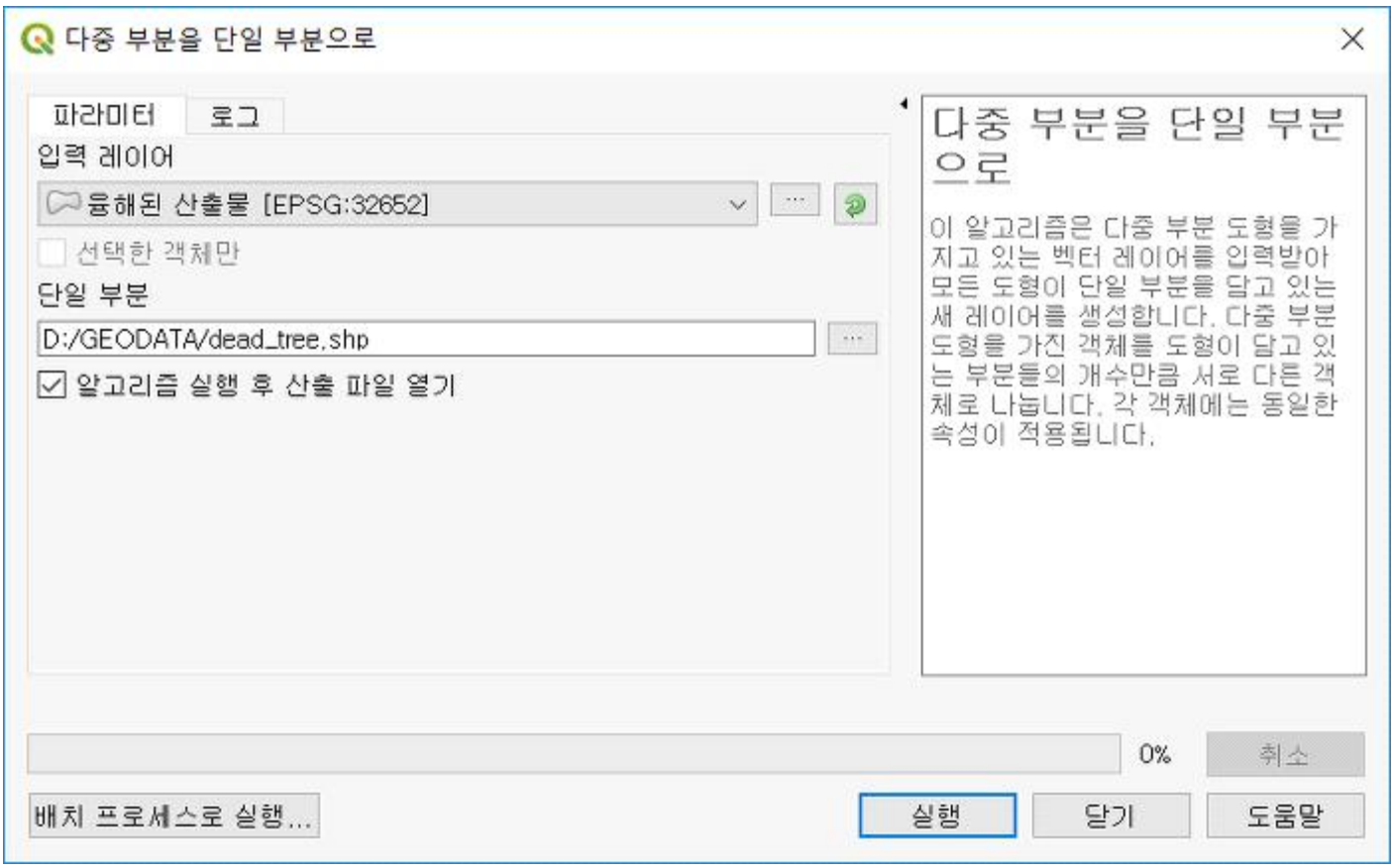
아래와 같이 인접한 피처가 병합되었습니다.



위 파일은 다중 영역 도형으로 구성되어 있습니다. 단일 부분으로 나누시려면 '벡터 >도형 도구 >다중 부분을 단일 부분으로'를 실행하시면 됩니다.



이렇게 되면 각각의 피처로 재분할됩니다. 면적, 둘레길이와 같이 피처를 추가 선별할 기준을 추가할 수 있겠죠?!



QGIS 3.4 Orfeo ToolBox(OTB)
머신러닝 기반 드론 영상분석

발행일
2019. 1.14.

작성자
국립공원공단 유병혁