데이터전처리 과정을 설명한 파일입니다.

필요한 패키지 : readxl, raster, rgdal, dplyr, writexl, sp, ggplot2, geosphere, Imap, igraph, ape

수집한 데이터는 전국\_읍면동\_행정코드파일, 전국\_읍면동\_교통/혼잡강도/총인구수/비행가능여부/터미널유무/건물면적/기차역유무, 행정동코드\_법정동코드 연계 자료입니다.

\* map\_list

먼저, 전국\_읍면동\_행정코드파일(shp파일, TL\_SCCO\_EMD.shp)에서 전국을 읍면동으로 나눈 법정동코드를 불러옵니다(데이터명 : map\_list). 시도를 나타내는 법정코드를 새로운 변수(SIDO)를 map\_list에 추가합니다.

\* code\_address, join\_data\_EN

또한 전국\_읍면동\_법정동주소 데이터(읍면동법적코도.xlsx)를 code\_address 데이터명으로 저장합니다. code\_address를 데이터프레임으로 전환해준 뒤 주소단위가 읍면동 이외의 데이터들은 제거합니다. 제거 작업이 끝나면 map\_list와 df\_code\_address 데이터를 join\_data\_EN으로 합칩니다(left\_join).

\* join\_data\_EN 전처리

join\_data\_EN의 결측값을 처리합니다. ( 영문명을 넣거나 주소가 없는 곳은 채워줍니다.) NA값인 행을 찾아서 데이터를 넣어준다. 이때 사용한 코드 방법은 for문을 이용하여 각각 문자열을 공백으로 나눠준 뒤, 제일 위의 읍면동을 data\_name에 저장하고 sido값은 법정코드를 나타내는 EMD\_CD의 앞 2자리를 뽑아서 data\_sido에 저장하여 join\_data\_EN의 SIDO,EMD\_KOR\_NM변수에 넣어준다. 그 후 제대로 각각 조인할 때 데이터가 없는 주소들에 한해서 일일이 수정하여 준다.

\* join\_data\_EN2, duplicated\_data, join\_data\_EN3

공역(비행제한구역) 데이터(데이터 파일 : 공역-법정동.csv)를 join\_data\_EN에 join\_data\_EN\_2 명으로 추가합니다. Join\_data\_EN\_2에서 중복된 데이터를 duplicated\_data에 저장해 줍니다. address\_list.rdata 파일을 불러와 duplicated\_data를 저장합니다 address\_list 데이터는 띄어쓰기 안된 데이터를 띄어쓰기 한 데이터로 변환한 데이터입니다. 띄어쓰기 적용이 제대로 적용 안된 데이터는 일일이 수정하여 저장하였습니다. Join\_data\_EN\_2의 x.1변수의 NA값을 채워 넣습니다. 그 후 필요 없는 데이터(띄어쓰기 안된 주소 변수, 번호)는 제거합니다. 그 후 비행불가능여부 변수의 NA값을 채워줍니다. NA값을 변환하는 방법은 airspace에 join\_data\_EN\_2 데이터와 duplicated\_data 데이터를 inner\_join 공통된 데이터를 저장하여 불필요한 변수 제거한 뒤, airspace에서 중복된 데이터를 제거해주고, join\_data\_EN\_2에도 중복 데이터 제거해준 뒤, join\_data\_EN\_2의 “법정동명” 변수에서 airspace “법정동명” 변수가 포함된 데이터만 골라 비행여부를 대입한다. 이때 기존에 조사하지 못한 읍면동의 비행불가능여부 값이 na값으로 나온다. 이 값은 추후에 채워넣기로 한다. 이때 대입한 데이터를 join\_data\_EN\_3에 저장한다.

\* bus\_join\_data

터미널유무 변수를 join\_data\_EN\_3에 추가한다. 데이터(터미널법정동.csv)를 bus\_station 이름으로 저장한다. bus\_station 변수를 일부 수정한 뒤(join을 해주기 위해 같은 특성으로 만든다.) bus\_station\_copy데이터에 저장한 뒤 join\_data\_EN\_3과 join해준다. 이때 R이 제대로 읽지 못해서 엑셀에 저장했다 다시 불러들인다. bus\_join\_data라는 이름으로 다시 저장한 뒤 중복값을 제거해주고, join하면서 생긴 NA값은 그 읍면동에는 터미널이 없는 값들이라 0으로 변환해준다.

\* trans, trans\_copy

교통혼잡,인구수, 승용차, 버스 데이터가 들어있는 파일(교통관련데이터.xlsx)를 trans 이름명으로 저장한 뒤 일부 데이터를 제거 후 trans\_copy에 저정한다. 이 데이터는 행정주소를 기반으로 수집한 자료여서 join할 때 필요한 새로운 변수를 하나 만들어야 한다. “행정주소”라는 이름의 변수를 만들어서 trans\_copy의 도/광역시/시, 시군구, 읍면동을 합쳐서 행정주소를 데이터를 만든다.

필요없는 변수를 제거하고 순서를 변경해준다.

\* relation\_data, copy\_relation\_data, copy\_2\_realtion\_data

행정주소와 법정주소를 연결해야 한다. 이때 필요한 데이터(행정법정동코드 연계자료.xlsx)를 relation\_data에 저장한다. 필요 없는 변수들을 제거하고 copy\_relation\_data에 저장한다. 이때 열이름을 첫번째 행 데이터 값으로 해준다. 이 데이터는 주소 단위를 “리”로 구분하고 있어서 읍면동을 제외한 모든 주소값들을 제거해준다. 그 후에 trans\_copy에서 “행정주소” 만들었듯이 copy\_relation\_data에도 같은 방식으로 행정주소 변수를 만든다. 법정동명은 8자리코드로 만들어준다.(10자리 코드 중 맨 뒤 2자리는 “리”에 해당되는 값이다.) 행정구역코드 결측값을 제거하기 위해 na값을 찾아보면 1개의 데이터행이 나온다. 이때 주소에 맞춰서 데이터를 넣어주고 전체 결측값들을 제거한 데이터를 copy2\_relation\_data에 저장한다.

\* first\_data\_ver2

지금까지 만든 데이터를 보면 bus\_join\_data, trans\_copy, copy2\_relation\_data를 만들었다. 이 3개의 데이터를 join을 써서 합쳐주면 된다. 먼저 bus\_join\_data와 copy2\_relation\_data를 조인해준다. 두 데이터의 데이터 형태를 맞춰준 뒤 join을 한다. Join한 데이터 중 필요 없는 변수를 제거한 뒤 new\_final\_data에 저장한다. new\_final\_data 와 trans\_copy를 join해준다. 이때 trans\_copy 변수 중 join의 기준값을 쓸 행정주소 값의 공백을 지워준다. 그 후 다시 두 데이터를 join 해준다. 그 뒤 필요한 변수만 추출하여 first\_data\_ver2에 저장해준다. 그 뒤 엑셀파일로 저장해준다.

\* first\_data\_ver2 전처리

저장한 엑셀파일에서 행정주소와 법정동주소를 보고 na값으로 join된 데이터값들 중 일부를 일일이 확인해서 채워준다. 행정주소와 법정동주소가 다르다보니 확인하면서 누락된 값들은 저장해준다. 또한 행정주소와 법정동주소가 같은데도 NA값인 데이터는 trans\_copy값에서 관측이 안된 지역이여서 데이터를 알 수가 없다. 행정주소 지역과 법정동주소 지역이 다른 부분이 있어서 이 부분은 제거해야한다.

\* second\_data, third\_data, third\_no\_na\_data

전처리 해준 데이터를 second\_data에 저장해준다. EMD\_CD를 기준으로 중복된 지역은 제거해주고 third\_data 이름으로 저장한다. 위에서 설명했듯이 아예 관측을 안한 지역의 na값을 제거해야한다. Third\_data의 행정주소와 법정주소가 같은 데이터 중에 승용차값이 na인 값을 no\_reserve\_data\_list에 저장하여 third\_data에서 no\_reserve\_data\_list에 해당하는 데이터들을 지워 third\_propressed\_data에 저장한다. 행정주소와 법정동주소가 같지 않는 지역은 지역이 완전히 같이 않으므로 제거해준다. 제거한 데이터를 third\_no\_na\_data에 저장한다. 그리고 3차가공데이터라는 이름으로 엑셀파일로 저장한다.

3차가공데이터에서 아까 비행불가능한지역 중 na값과 기차역유무 데이터를 추가한다.

---- 이 부분은 범진님이 작성해주세요. ---

\* map, fusion\_data

4차가공데이터를 data 변수에 저장한다. map\_list에 “id” 이름으로 데이터를 새로 추가한다. 데이터의 순서를 지정해주는 데이터이다. 만든 데이터를 map\_list 데이터와 join연산을 하여 “fusion\_data”이름으로 저장한다.

\* mm\_normal

maxmain정규화를 하기 위해 사용자지정 함수를 만든다. “mm\_normal”이라는 이름으로

(x – min(x)) / (max(x) – min(x))의 식을 만든다.

\* korea

spTransform함수를 이용해서 map의 위도,경도 좌표를 원래의 위도,경도로 바꾼다. 그 후 korea라는 데이터명으로 fotify함수를 이용해서 map을 폴리곤형태로 시각화 할 수 있는 데이터로 만들어준다.

\* long\_check\_max, long\_check\_min, lat\_check\_max, lat\_check\_min

위의 각각의 함수는 각 읍면동 중심을 알기 위해서 읍면동 범위의 최소 최대의 위/경도값을 저장해주는 함수이다.

“long\_check\_max” : 각 읍면동의 최대 경도

“long\_check\_min” : 각 읍면동의 최소 경도

“lat\_check\_max” : 각 읍면동의 최대 위도

“lat\_check\_min” : 각 읍면동의 최소 위도

\* long, lat

“long”, “lat” 이라는 함수에 구한 값으로

long에는 위도의 최대, 최소 데이터를 , lat에는 경도의 최대. 최소 데이터를 데이터프레임 형태로 저장한다.

그후 long,lat “long”, “lat”이라는 각각 변수를 만들어서 최대, 최소의 평균값을 저장한다.(apply함수를 사용)

\* location

“location” 이름으로 long의 long변수, lat의 lat변수를 데이터프레임으로 저장한다. “id”라는 변수를 만들어서 위에서부터 0부터 5050까지 대입한다.

\* fiveth\_data

“fiveth\_data”에 mm\_normal, location 데이터를 inner\_join을 사용하여 합쳐준다.

\* terminal, terminal\_dist,

“terminal” 이름으로 전국 터미널 위경도 데이터를 저장한다. 필요 없는 부분은 지워버린다.

Imap 패키지의 gdist를 이용하여 각 읍면동 중심 좌표와 각 터미널 좌표와의 거리를 구한다. 거리 차이를 “terminal\_dist” 이름으로 저장한다. gdist함수를 살펴보면 lon.1, lat.1에는 각 읍면동의 위,경도값을 lon.2, lat.2에는 각 터미널의 좌표 데이터를 입력해준다. 그 중 가장 값이 작은 값을 출력하는 코드를 만들어 각 읍면동별로 가장 거리가 짧은 값을 terminal\_dist에 저장한다.

\* train\_station\_data, train\_station\_dist, airport\_data, airport\_dist

터미널과의 거리 구하는 방식 그대로 train\_station\_data/airport\_data에 기차역 좌표/공항 좌표를 저장하고 같은 방법으로 train\_station\_dist/airport\_dist에 각 읍면동 중심좌표와의 거리가 가장 짧은 값들을 저장한다.

\* distance, sixth\_data

위에서 구한 ternimai\_dist, train\_station\_dist, airport\_dist를 “distance”에 데이터프레임으로 저장한다. 그 후 각각의 열이름을 바꿔준 뒤, 5차가공데이터에 distance 데이터를 “sixth\_data” 이름으로 저장한다.