Project 10: Water Quality Analysis

Phase 4: Water Potability

After many years of research, water quality standards are put in place to ensure the suitability of efficient use of water for a designated purpose. Water quality analysis is to measure the required parameters of water, following standard methods, to check whether they are in accordance with the standard.

Content:

The water quality analysis boarding summary.csv file contains route,trip,stop and week of year from 20140711.

Data source

The data fields in the given file are

TripID Unique identity of trip

RouteID Value representing public transport route

StopID Unique identity of stop

StopName Name of given stop

WeekBeginning Date representing first day of any week

NumberOfBoarding Count of all boarding's occurred at this stop for the named trip over the previous week

External Features

Some Important external data fields calculation

IsHoliday Number of public holidays within that week

DistanceFromCentre Distance measure from the city centre

For Calculating Distance between centre with other bus stops by using Longitude and Latitude we have used the Haversine formula

In [8]:

```
From math import sin, cos, sqrt, atan2, radians

Def calc_dist(lat1,lon1):

   ## approximate radius of earth in km

   R = 6373.0

   Dlon = radians(138.604801) – radians(lon1)
   Dlat = radians(-34.921247) – radians(lat1)

   A = sin(dlat / 2)**2 + cos(radians(lat1)) * cos(radians(-34.921247)) * sin(dlon / 2)**2

   C = 2 * atan2(sqrt(a), sqrt(1 – a))

   Return R * c

In [9]:

Out_geo[‘dist_from_centre’] = out_geo[[‘latitude’,’longitude’]].apply(lambda x: calc_dist(*x), axis=1)

In [10]:

##Fill the missing values with mode

Out_geo[‘type’].fillna(‘street_address’,inplace=True)

Out_geo[‘type’] = out_geo[‘type’].apply(lambda x: str(x).split(‘,’)[-1])

In [11]:

Out_geo[‘type’].unique()

Out[11]:

Array([‘street_address’, ‘transit_station’, ‘premise’, ‘political’,

      ‘school’, ‘route’, ‘intersection’, ‘point_of_interest’,

      ‘subpremise’, ‘real_estate_agency’, ‘university’, ‘travel_agency’,

      ‘restaurant’, ‘supermarket’, ‘store’, ‘post_office’], dtype=object)
```

Adding the details regarding the Public holidays from June 2013 to June 2014

```
In [12]:

“’Holidays—

Out[12]:

“Holidays--\n2013-09-01,Father’s Day\n2013-10-07,Labour day\n2013-12-25,Christmas day\n2013-12-26,Proclamation Day\n2014-01-01,New Year\n2014-01-27,Australia Day\n2014-03-10,March Public Holiday\n2014-04-18,Good Friday\n2014-04-19,Easter Saturday\n2014-04-21,Easter Monday\n2014-04-25,Anzac Day\n2014-06-09,Queen’s Birthday”

In [13]:

Def holiday_label (row):

   If row == datetime.date(2013, 9, 1) :

        Return ‘1’

   If row == datetime.date(2013, 10, 6) :
        Return ‘1’
    If row == datetime.date(2013, 12, 22) :
        Return ‘2’
    If row == datetime.date(2013, 12, 29):
        Return ‘1’
    If row  == datetime.date(2014, 1, 26):
        Return ‘1’
    If row == datetime.date(2014, 3, 9):
        Return ‘1’
    If row == datetime.date(2014, 4, 13) :
        Return ‘2’
    If row == datetime.date(2014, 4, 20):
        Return ‘2’
    If row == datetime.date(2014, 6, 8):
        Return ‘1’
    Return ‘0’
```

In [14]:

```
Data[‘WeekBeginning’] = pd.to_datetime(data[‘WeekBeginning’]).dt.date
```

In [15]:

```
Data[‘holiday_label’] = data[‘WeekBeginning’].apply (lambda row: holiday_label(row))
```

Data Aggregation

Combine the Geolocation,Routes and main input file to get final Output File.

In [16]:

```
Data= pd.merge(data,out_geo,how=’left’,left_on = ‘StopName’,right_on = ‘input_string’)
```

In [17]:

```
Data = pd.merge(data, route, how=’left’, left_on = ‘RouteID’, right_on = ‘route_id’)
```

Columns to keep for further analysis

In [18]:

```
Col = [‘TripID’, ‘RouteID’, ‘StopID’, ‘StopName’,
‘WeekBeginning’,’NumberOfBoardings’,’formatted_address’,
   ‘latitude’, ‘longitude’,’postcode’,’type’,’route_desc’,’dist_from_centre’,’holiday_label’]
```

In [19]:

```
Data = data[col]
```

In [20]:

```
##saving the final dataset

Data.to_csv(‘Weekly_Boarding.csv’,index=False)
```

In [21]:

```
## getting the addresses for geolocation api.

# Address data[‘StopName’].unique()

# sub = pd.DataFrame({‘Address’: Address})

# sub=sub.reindex(columns=[“Address”])

# sub.to_csv(‘addr.csv’)
```

Aggregate the Data According to Weeks and Stop names

NumberOfBoardings_sum Number of Boardings within particular week for each Bus stop

NumberOfBoardings_count Number of times data is recorded within week

NumberOfBoardings_max Maximum number of boarding done at single time within week

In [22]:

```
# st_week_grp1 =
pd.DataFrame(data.groupby([‘StopName’,’WeekBeginning’,’type’]).agg({‘NumberOfBoardings’: [‘sum’,
‘count’]})).reset_index()

Grouped = data.groupby([‘StopName’,’WeekBeginning’,’type’]).agg({‘NumberOfBoardings’: [‘sum’,
‘count’,’max’]})

Grouped.columns = [“_”.join(x) for x in grouped.columns.ravel()]
```

In [23]:

```
St_week_grp = pd.DataFrame(grouped).reset_index()

St_week_grp.shape

St_week_grp.head()
```

Out[23]:

```
(207864, 6)

Out[23]:


StopName

WeekBeginning

Type

NumberOfBoardings_sum

NumberOfBoardings_count

NumberOfBoardings_max


0

1 Anzac Hwy

2013-06-30

Street_address

1003

378

51


1

1 Anzac Hwy

2013-07-07

Street_address

783

360

28


2

1 Anzac Hwy

2013-07-14
```

Street_address

843

343

45

3

1 Anzac Hwy

2013-07-21

Street_address

710

356

28

4

1 Anzac Hwy

2013-07-28

Street_address

898

379

41

Gathering only the Stop Name which having all 54 weeks of Data

In [24]:

```
St_week_grp1 = pd.DataFrame(st_week_grp.groupby(‘StopName’)[‘WeekBeginning’].count()).reset_index()
```

In [25]:

```
Aa=list(st_week_grp1[st_week_grp1[‘WeekBeginning’] == 54][‘StopName’])
```

In [26]:

```
Bb = st_week_grp[st_week_grp[‘StopName’].isin(aa)]
```

In [27]:

```
## save the aggregate data

bb.to_csv(‘st_week_grp.csv’, index=False)
```

Data Exploration

Having Total of 4165 Stops in South Australian Metropolitan Area.

In [28]:

```
Data.nunique()
```

Out[28]:

```
TripID            39282

RouteID              619

StopID             7397

StopName            4165

WeekBeginning         54

NumberOfBoardings     400

Formatted_address    3242

Latitude            3029

Longitude            3008

Postcode              207

Type                16

Route_desc           440

Dist_from_centre     3033

Holiday_label          3

Dtype: int64
```

In [29]:

```
Data.shape

Data.columns

Data.head(3)
```

Out[29]:

```
(10857234, 14)

Out[29]:

Index([‘TripID’, ‘RouteID’, ‘StopID’, ‘StopName’, ‘WeekBeginning’,

       ‘NumberOfBoardings’, ‘formatted_address’, ‘latitude’, ‘longitude’,

       ‘postcode’, ‘type’, ‘route_desc’, ‘dist_from_centre’, ‘holiday_label’],

      Dtype=’object’)

Out[29]:


TripID

RouteID

StopID

StopName

WeekBeginning

NumberOfBoardings

Formatted_address

Latitude

Longitude

Postcode

Type

Route_desc

Dist_from_centre

Holiday_label


0

23631

100

14156

181 Cross Rd

2013-06-30
```

1

181 Cross Rd, Westbourne Park SA 5041, Australia

-34.966656

138.592148

5041

Street_address

Via Woodville Road, Holbrooks Road, Marion Roa…

5.180961

0

1

23631

100

14144

177 Cross Rd

2013-06-30

1

177 Cross Rd, Westbourne Park SA 5041, Australia

-34.966607

138.592301

5041

Street_address

Via Woodville Road, Holbrooks Road, Marion Roa…

5.172525

0

2

23632

100

```
14132

175 Cross Rd

2013-06-30

1

175 Cross Rd, Westbourne Park SA 5041, Australia

-34.966758

138.592715

5041

Street_address

Via Woodville Road, Holbrooks Road, Marion Roa...

5.180709

0
```

In [30]:

```
Data.isnull().sum()
```

Out[30]:

```
TripID                  0
RouteID                  0
StopID                  0
StopName                 0
WeekBeginning            0
NumberOfBoardings         0
Formatted_address       3506
Latitude                 0
Longitude                0
Postcode           425081
Type                    0
Route_desc          2106618
Dist_from_centre         0
```

```
Holiday_label        0
Dtype: int64
```