17/5/2019
----Connect R with Postgres
Data Path:
C:/Users/Sharvil T/Desktop/Jitendra-Sir
Mapping/4may/21_april_db_as_on_4may.backup

C:/Users/Sharvil T/Desktop/Mapping/backup_town/akola_t

C:/Users/Sharvil T/Desktop/Mapping/backup_town/akola_t.sql
C:/Users/Sharvil T/Desktop/Mapping/backup_village/akola_v.sql

Work to be done
1)Dump backup files using psql Terminal

Use StackBuilder and include POSTGIS to include spatial data for viewing
USE Psql or command prompt as you Please
Useful Commands
/? for all psql commands
/l all databases
/c connect to a database
/i execute sql file[for dumps]

2)Make a connection between R and Postgres

Extract all with Akola
(Our main focus is to obtain prox(Proximity related columsn as they pertain to amenities within the village)

```
--select table name with reference to akola
select table_name
from information_schema.tables
where table_type='BASE TABLE'
and table_schema='public'
and table_name like 'akola%'
;
--Select column names from akola with proximity-------
select column_name
from information_schema.columns
where table_name ='akola'
and column_name like '%prox';
```

To Compute Nearest Distance of village from a set of villages we make use of KNN

##KNN To Compare Distance between Two Villages

```
with demo as(
select village_name,census_2011code,centroid from akola
where village_name='mahamadpur')
select ST_Distance(a.centroid,d.centroid) from akola a,demo d where
a.village_name='mahaldoli';
```

Answer:"0.64395711934398" Which is in Degrees

---Distance of mahaldoli from mahamadpur

```
with demo as(
select village_name,census_2011code,centroid from akola
where village_name='mahamadpur')
select ST_Distance_Sphere(a.centroid,d.centroid) from akola a,demo d where
a.village_name='mahaldoli';
```

---Answer:70966.92297752 in metres

```
select ((ST_Area(geom)*power(0.3048,2))::double precision)as area from akola
limit 11;
---Ans:Area in square feet


--Distance of village Mahamadpur from all villages in Square Metres
with demo as(
select village_name,centroid from akola where village_name='mahamadpur'
group by village_name,centroid )
select a.village_name,ST_Distance_Sphere(a.centroid,d.centroid)as distance_sqm
from akola a,demo d
group by a.village_name,a.centroid,d.centroid
order by a.centroid<->d.centroid;

<-> means "distance between box centers"
<#> means "distance between box edges"


with demo as(
select village_name,centroid from akola where village_name like'm%'
group by village_name,centroid )
select a.village_name,d.village_name,
(ST_Distance_Sphere(a.centroid,d.centroid)*0.001) as distance_Km from akola
a,demo d
where ST_DistanceSphere(a.centroid,d.centroid)<>0
group by a.village_name,a.centroid,d.centroid,d.village_name
order by a.centroid<->d.centroid limit 15;

---Alternate Query
with demo as(
select village_name,centroid from akola where village_name like 'd%'
group by village_name,centroid )
select (select d.village_name from demo d group by d.village_name
             having count(1)='1'
               order by d.village_name
             limit 1)as Start_Village
,a.village_name as Dest_Village,
(ST_Distance_Sphere(d.centroid,a.centroid)*0.001) as distance_Km
from demo d,akola a
where ST_DistanceSphere(d.centroid,a.centroid)<>0
order by d.centroid<->a.centroid limit 50;

-- Prob Statement : If you have to find K nearest neighour of all villages in
the list then what to do
-- Ans :Make use of ST_Distance_Sphere and order by <-> distance operator

---K Nearest neighour of evry village starting with d

with demo as(
select village_name,centroid from akola where village_name like 'd%'
group by village_name,centroid)
select distinct on (d.village_name) d.village_name as Start_Village
,a.village_name as Dest_Village,
(ST_DistanceSphere(d.centroid,a.centroid)*0.001) as distance_Km
from demo d,akola a
where ST_DistanceSphere(d.centroid,a.centroid)<>0 and STD
order by d.village_name,d.centroid<->a.centroid ;

select count(village_name) from akola where village_name like 'd%';

--K Nearest Neighour of Every Village

with demo as(
select village_name,centroid from akola
```

```
group by village_name,centroid)
select distinct on (d.village_name) d.village_name as Start_Village,d.centroid
as Centroid_start,
a.village_name as Dest_Village,a.centroid  Centroid_End,
(ST_Distance_Sphere(d.centroid,a.centroid)*0.001) as distance_Km
from demo d,akola a
where ST_DistanceSphere(d.centroid,a.centroid)<>0
order by d.village_name,d.centroid<->a.centroid limit 50;

select * from akola limit 11;
--Prob:How will you fetch latitude and longitude from Geom Lines and polygons
--Ans:Firstly let us undertsand that since they are lines you will have to find
centroid of these \
-- lines/polygons to fetch a point and tehn convert tht point

select ST_X(ST_Centroid(geom))as Latitude,ST_Y(ST_Centroid(geom)) from akola
where village_name ='adsul';
--Find hanging Queries
SELECT * FROM pg_stat_activity WHERE state = 'active';
--Kill a Query
select pg_terminate_backend(10332);
--K Nearest Neighour with latitude and longitude Respectively

with demo as(
select village_name,centroid from akola
group by village_name,centroid)
select distinct on (d.village_name) d.village_name as Start_Village,
ST_X(d.centroid)as Latitude_start,ST_Y(d.centroid)as Longitude_start,
a.village_name as Dest_Village,
ST_X(a.centroid)as Latitude_Dest,ST_Y(a.centroid)as Longitude_Dest,
(ST_Distance_Sphere(d.centroid,a.centroid)*0.001) as distance_Km
from demo d,akola a
where ST_DistanceSphere(d.centroid,a.centroid)<>0
order by d.village_name,d.centroid<->a.centroid limit 50;


--Area of Villages from Smallest to Largest
select village_name,census_2011code,((ST_Area(geom)*power(0.3048,2))::double
precision)as area_Sqm from akola
order by area_Sqm asc limit 11;

#####Establish Connection between R and Postgres
drv<-dbDriver("PostgreSQL");
conn<-dbConnect(drv,dbname=dbname,host=host,port=port,
                user=user,password=password);
##To Check Connection has been established


####
frame<-dbGetQuery(conn,statement = paste("select
village_name,near_pre_vil_town_name,
(near_pre_facilty_prox),geom,centroid from akola where near_pre_facilty_prox!
='NULL' group by 1,2,3,4,5  order by 3;
"))
Store Query in a frame

Make connection
install.packages("devtools");
require(devtools)
devtools::install_github("dkahle/ggmap", ref = "tidyup")
---Now we ahve to get api from google cloud platform to view our co-ordiantes on
google map
register_google(key="AIzaSyCbChBi43l_5bI1JRNtOOPuEXXiwQdry8k");
```

```
# Using Leflets as the google API  wont work
library(sp)
rm(df)
library(leaflet)
df1$
mp1<-leaflet(df1) %>%addProviderTiles(providers$Stamen.Toner)%>%
addMarkers(~Longitude,~Latitude,popup =Knn_Mapping$start_village[1:50])%>%
addTiles()

# Convert Knn into start and destination types
df<-as.data.frame(Knn_Mapping)
str(df)
colnames(df)<-c("Start-
Village","Longitude","Latitude","Dest_Village","Longitude_Dest","Latitude_Dest",
"Distance")
mp1<-leaflet(df) %>%addProviderTiles(providers$OpenStreetMap)%>%
addMarkers(~Longitude,~Latitude,popup =Knn_Mapping$start_village)%>% addTiles()
mp1
# Importing icon image
Icon <- makeIcon(
  iconUrl = "",
  iconWidth = 2*31*215/230, iconHeight = 2*31,
  iconAnchorX = 2*31*215/230/2, iconAnchorY = 2*16
)
# We need to Combine multiple column values into one.For uniquely grouping them
later
# Soln:Use Unlist

frame1<-data.frame(df$Longitude_Start,df$Longitude_Dest)
frame1<-data.frame(Longitude<-unlist(frame1,use.names = FALSE))
frame2<-data.frame(df$Latitude_Start,df$Latitude_Dest)
frame2<-data.frame(Latitude<-unlist(frame2,use.names = FALSE))

frame3<-data.frame(df$`Start-Village,df$Dest_Village)
frame3<-data.frame(Latitude<-unlist(frame3,use.names = FALSE))
combine_frame<-as.data.frame(cbind(frame1,frame2,frame3))
str(combine_frame)
colnames(combine_frame)<-c("Longitude","Latitude","Villages","Point")
##Now we have to intorduce a column that uniquely categorizes start amd dest
accordingly by keeping an offset of 893
combine_frame[1:15,]

combine_frame$Point[1:893]<-c("Start")
combine_frame$Point[894:1786]<-c("Dest")

# Remove a column in R
combine_frame<-combine_frame[,-4]
#View Structure of a Data frame in R
str(combine_frame)
# Convert column to factor
combine_frame$Point<-as.factor(combine_frame$Point)
# Now lets a make a map
mp2<-leaflet(combine_frame) %>%addProviderTiles(providers$OpenStreetMap)%>%
addMarkers(~Longitude,~Latitude,popup =~Villages )%>% addTiles()
install.packages("geosphere");
library(geosphere)
#
# This is anotehr way without using unlist or gather from tidyr
# # Now our aim is to draw polylines for which we will need geosphere
# rm(m)
# m<-leaflet(data=df)%>%addTiles
# for (i in 1:nrow(df))
```

```
# m<-m%>%addPolylines(lat=c(df[i,]$Latitude_Start,df[i,]$Latitude_Dest),
#                     lng=c(df[i,]$Longitude_Start,df[i,]$Longitude_Dest))
#
#
# m<-m%>
%addMarkers(~c(Longitude_Start,Longitude_Dest),~c(Latitude_Start,Latitude_Dest)
#               ,popup =~`c(Start-Village,Dest_Village))
#
# m

rm(m1)

####Polylines connects two or more points for a given point
###Used For loop where 1st latlong will form a line with 864th which is its
closest knn neighour

m1<-leaflet(data=combine_frame)%>%addTiles
for (i in 1:893){
  j<-i+893
  m1<-m1%>%addPolylines(lat=c(combine_frame[i,]$Latitude,combine_frame[j,]
$Latitude),
                     lng=c(combine_frame[i,]$Longitude,combine_frame[j,]
$Longitude))
}

###Red Color for Starting points
####Green color for Destination points
new<-("red")[combine_frame$Point[1:893]]
new1<- ("green")[combine_frame$Point[894:1786]]

icons <- awesomeIcons(
  icon = 'ios-close',
  iconColor = 'black',
  library = 'ion',
  markerColor =c(new,new1)
)
--This will plot the points on the lines
---addAwesomeMarkers will allowe you to add icons
m1<-m1%>%addAwesomeMarkers(lat=~Latitude,lng= ~Longitude,popup = ~Villages,icon
= icons)

m1
```