Assessment Part 3

Analysis of start-up incubators in London by knowledge spillover potential

—— Part 1: preparing data of incubators ——

In this part, the incubators dataset will be read, in order to map the startup incubator locations in London, using the tidyverse package. Subsetting dataframe: The directory includes incubators nationwide, so after reading the csv file we will keep only the incubators within London.

```r
library(tidyverse)

inc_original<- read_csv("C:/Users/User/Desktop/work/casa/0005 GIS/1. week 10 FINAL assignment/output/edit
ed data/incubators.csv", na = "n/a")

inc <- inc_original[grep("London",inc_original$City),]

#drop useless columns
inc=inc[,-c(1,3,5,7:14,16:20,22)]
inc=inc[,-c(4:6)]
```

The following part is the **geolocation** process. The dataset of incubators will be divided into 4 parts, to avoid slow performance.

```r
library(ggmap)

#subset the dataset of incubators in 4 bits
inc1=inc[1:45,]
inc2=inc[46:90,]
inc3=inc[91:136,]
inc4=inc[137:178,]

# inc 3 includes NA input in the addresses.
#These rows will be removed and the complete ones will be kept
inc3=inc3[complete.cases(inc3), ]
```

Geocoding:

```r
# Geocoding Inc1:
# goes through the rows of incubators' dataset and creates a new longitude and latitude field based on th
e input of the original field "Address"
#creates a csv file with the updated addresses, for future use if needed

for(i in 1:nrow(inc1)) {
  result <- tryCatch(geocode(inc1$Address[i], output = "latlona", source = "dsk"),
                     warning = function(w) data.frame(lon = NA, lat = NA, address = NA))
  inc1$lon[i] <- as.numeric(result[1])
  inc1$lat[i] <- as.numeric(result[2])
}
# Write a CSV file with the updated addresses
write.csv(inc1, "geocoded1.csv", row.names=FALSE)


# Same for Inc 2

for(i in 1:nrow(inc2)) {
  result <- tryCatch(geocode(inc2$Address[i], output = "latlona", source = "dsk"),
                     warning = function(w) data.frame(lon = NA, lat = NA, address = NA))
  inc2$lon[i] <- as.numeric(result[1])
  inc2$lat[i] <- as.numeric(result[2])
}

write.csv(inc2, "geocoded2.csv", row.names=FALSE)


# Same for Inc 3


for(i in 1:nrow(inc3)) {
  result <- tryCatch(geocode(inc3$Address[i], output = "latlona", source = "dsk"),
                     warning = function(w) data.frame(lon = NA, lat = NA, address = NA))
  inc3$lon[i] <- as.numeric(result[1])
  inc3$lat[i] <- as.numeric(result[2])
}

write.csv(inc3, "geocoded3.csv", row.names=FALSE)


# Same for Inc 4

for(i in 1:nrow(inc4)) {
  result <- tryCatch(geocode(inc4$Address[i], output = "latlona", source = "dsk"),
                     warning = function(w) data.frame(lon = NA, lat = NA, address = NA))
  inc4$lon[i] <- as.numeric(result[1])
  inc4$lat[i] <- as.numeric(result[2])
}

write.csv(inc4, "geocoded4.csv", row.names=FALSE)




#join the parts back together
inc_geocoded=rbind(inc1,inc2,inc3,inc4)
```

Some addresses were not geolocated due to invalid input. This part keeps only the complete rows and separates the ones with NA values.

```r
na_addresses <- inc_geocoded[is.na(inc_geocoded$lon),]

inc_geocoded=inc_geocoded[complete.cases(inc_geocoded),]
```

The rows below could not be geolocated. They did have an input, but it was either incomplete or invalid. If we wanted to avoid “hard coding” we would remove na_addresses and keep only the dataset that was produced in the above chunk. In this case however, we will just keep from the address input the postcodes and try to geolocate them again.

```r
# assigning a postcode to the addresses with invalid input.
na_addresses[c(1,2),3]="E1W 1UN"
na_addresses[3,3]="N1 7GU"
na_addresses[4,3]="SW1Y 4QU"
na_addresses[5,3]="N1 9AB"
na_addresses[6,3]="SE1 6FE"
na_addresses[7,3]="E15 2GZ"
na_addresses[8,3]="EC2R 8BT"
na_addresses[10,3]="SE1 8WA"
na_addresses[11,3]="BT48 7TG"
na_addresses[12,3]="EC2R 8AE"
na_addresses[13,3]="EC2Y 8DT"
na_addresses[15,3]="SE10 0ER"

#checking that we keep only the complete rows
na_addresses=na_addresses[-c(9,14),]

for(i in 1:nrow(na_addresses)) {
  result <- tryCatch(geocode(na_addresses$Address[i], output = "latlona", source = "dsk"),
                     warning = function(w) data.frame(lon = NA, lat = NA, address = NA))
  na_addresses$lon[i] <- as.numeric(result[1])
  na_addresses$lat[i] <- as.numeric(result[2])
}
# Write a CSV file with the updated addresses
write.csv(na_addresses, "geocoded4.csv", row.names=FALSE)



#adding the updated addresses to the geocoded dataset we created in the beginning, with the rest of the g
eolocated incubators

inc_geocoded=rbind(inc_geocoded,na_addresses)
```

```r
library(sf)

#checking that we keep only the complete rows
inc_geocoded=inc_geocoded[complete.cases(inc_geocoded),]

#create an sf object to work with, for the analysis
inc_SF=st_as_sf(inc_geocoded, coords = c("lon", "lat"), agr = "constant")

library(tmap) # for a quick thematic view of the incubators

#checking CRS system
st_crs(inc_SF)
```

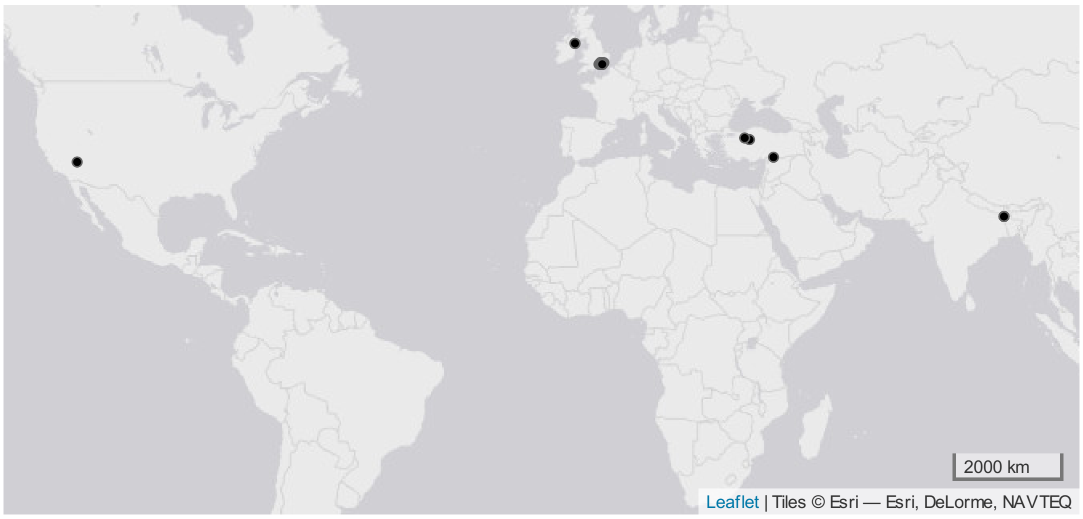
```
## Coordinate Reference System: NA
```

We see that the CRS of incubators is unassigned, so we set it using st_set_crs function and plot the resulting sf object.

```r
inc_SF=inc_SF%>% st_set_crs(4326)
st_crs(inc_SF)
```

```
## Coordinate Reference System:
##   EPSG: 4326
##   proj4string: "+proj=longlat +datum=WGS84 +no_defs"
```

```r
tmap_mode("view")
qtm(inc_SF)
```



From the plot, we see that some of the incubators (possibly due to invalid address input in the original incubators directory) are located outside of UK.

We need to keep only the relevant locations within London. For that, we firstly have to import the London Wards data, so that we can then keep the points within these polygons.

```
#reading shp file of London boundaries
library(rgdal)

LondonWardsSP <- readOGR("C:/Users/User/Desktop/work/casa/0005 GIS/1. week 7 assignment 2/material for pr
actical/LondonWardsBoundaries/LondonWardsNew.shp")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\User\Desktop\work\casa\0005 GIS\1. week 7 assignment 2\material for practical\LondonW
ardsBoundaries\LondonWardsNew.shp", layer: "LondonWardsNew"
## with 649 features
## It has 7 fields
```

```
class(LondonWardsSP)
```

```
## [1] "SpatialPolygonsDataFrame"
## attr(,"package")
## [1] "sp"
```

In order to use functions which are based on overlay or intersection methods, we need to make sure that both objects have the same CRS.

```
#create and SF object of London Wards
LondonWardsSF=st_as_sf(LondonWardsSP)

LondonWardsSF_BNG=st_transform(LondonWardsSF,27700)

LondonWardsSP_BNG=as(LondonWardsSF_BNG,"Spatial")

st_crs(LondonWardsSF_BNG)
```

```
## Coordinate Reference System:
##   EPSG: 27700
##   proj4string: "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000 +ellps=airy +
towgs84=446.448,-125.157,542.06,0.15,0.247,0.842,-20.489 +units=m +no_defs"
```

```
st_crs(inc_SF)
```

```
## Coordinate Reference System:
##   EPSG: 4326
##   proj4string: "+proj=longlat +datum=WGS84 +no_defs"
```

It can be seen that the incubator points and the London Wards have a different CRS, therefore we need to reproject the data into a mutual CRS, in particular the British National Grid.

```
#reprojecting incubators' CRS
inc_SF_BNG=st_transform(inc_SF,27700)

#checking again if CRS match
st_crs(LondonWardsSF_BNG)
```

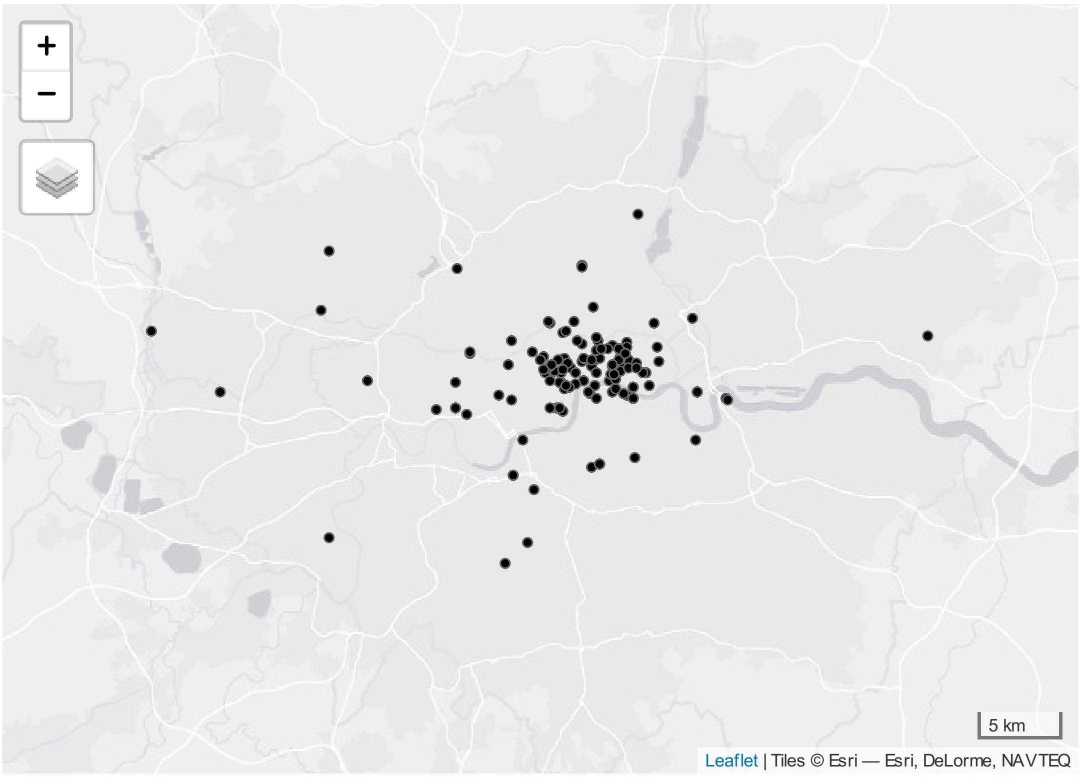
```
## Coordinate Reference System:
##   EPSG: 27700
##   proj4string: "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000 +ellps=airy +
towgs84=446.448,-125.157,542.06,0.15,0.247,0.842,-20.489 +units=m +no_defs"
```

```
st_crs(inc_SF_BNG)
```

```
## Coordinate Reference System:
##   EPSG: 27700
##   proj4string: "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000 +ellps=airy +
towgs84=446.448,-125.157,542.06,0.15,0.247,0.842,-20.489 +units=m +no_defs"
```

So, the **incubators in London** are:

```
inc_SF_BNG=inc_SF_BNG[LondonWardsSF_BNG,]

qtm(inc_SF_BNG)
```



—— **Plotting incubators in London** ——-

In the previous part, we imported the London Wards boundaries. These will be particularly useful for the next part, where the analysis of incubators in certain areas of London will be implemented, since we will need our map to focus on a certain part of London.

However, for the initial visualisation of the incubators in London, where we need a rough idea of the whole, we will need only the London boroughs. So, we will import those as well.

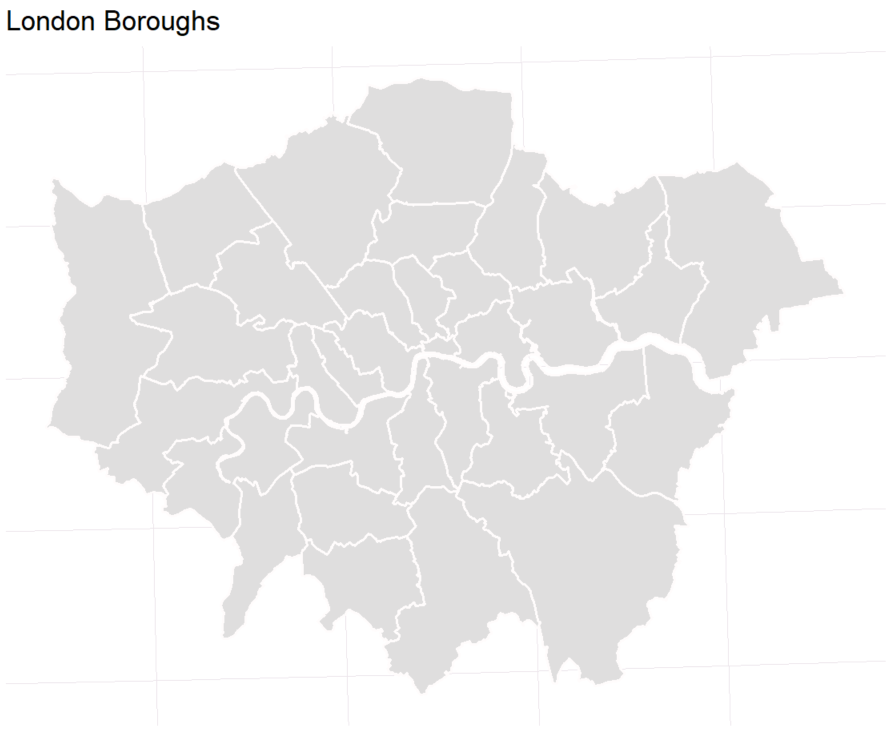
```
library(ggplot2)  #needed for the plot
library(sf)
```

Importing London Boroughs

```
library(rgdal)
library(geojsonio)

EW <- geojson_read("http://geoportal.statistics.gov.uk/datasets/8edafbe3276d4b56aec60991cbddda50_2.geojso
n", what = "sp")

#keeping areas of London
LondonMap <- EW[grep("^E09",EW@data$lad15cd),]

#converting into sf object to work with
LondonMapsf=st_as_sf(LondonMap)


#reprojecting CRS
LondonMap_BNG=st_transform(LondonMapsf,"+init=epsg:27700")

#LondonMap_BNG <- LondonMap_BNG%>% st_set_crs(27700)
```

Plotting London Boroughs

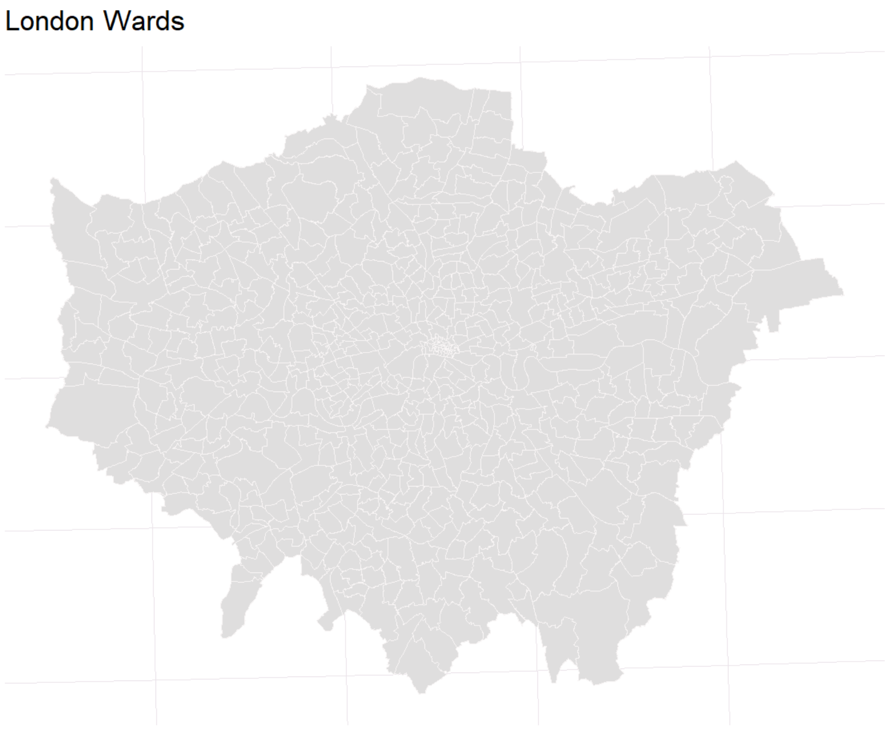
```
ggplot(data=LondonWardsSF_BNG)+geom_sf(mapping = aes(geometry=geometry),data = LondonMap_BNG,fill="#dfded
e",colour="#fffcfc",lwd=0.7)+theme(panel.border = element_blank(),panel.background = element_rect(fill =
'white',colour = "white"),panel.grid.major = element_line(colour = "#ece5eb"),panel.grid.minor = element_
line(colour="#ece5eb"), axis.title=element_text(size=8))+ggtitle("London Boroughs")+labs(x = "", y = "")
+coord_sf()+theme(axis.text = element_blank(), axis.ticks = element_blank())
```



```
ggsave(file="London boroughs update.jpeg", width=7, height=7, dpi=300)
```

Plotting London Wards

```
ggplot(data=LondonWardsSF_BNG)+geom_sf(mapping = aes(geometry=geometry),data = LondonWardsSF_BNG,fill="#d
fdede",colour="#f8f5f5",lwd=0.3)+theme(panel.border = element_blank(),panel.background = element_rect(fil
l = 'white',colour = "white"),panel.grid.major = element_line(colour = "#ece5eb"),panel.grid.minor = elem
ent_line(colour="#ece5eb"), axis.title=element_text(size=8))+ggtitle("London Wards")+labs(x = "", y = "")
+coord_sf()+theme(axis.text = element_blank(), axis.ticks = element_blank())
```

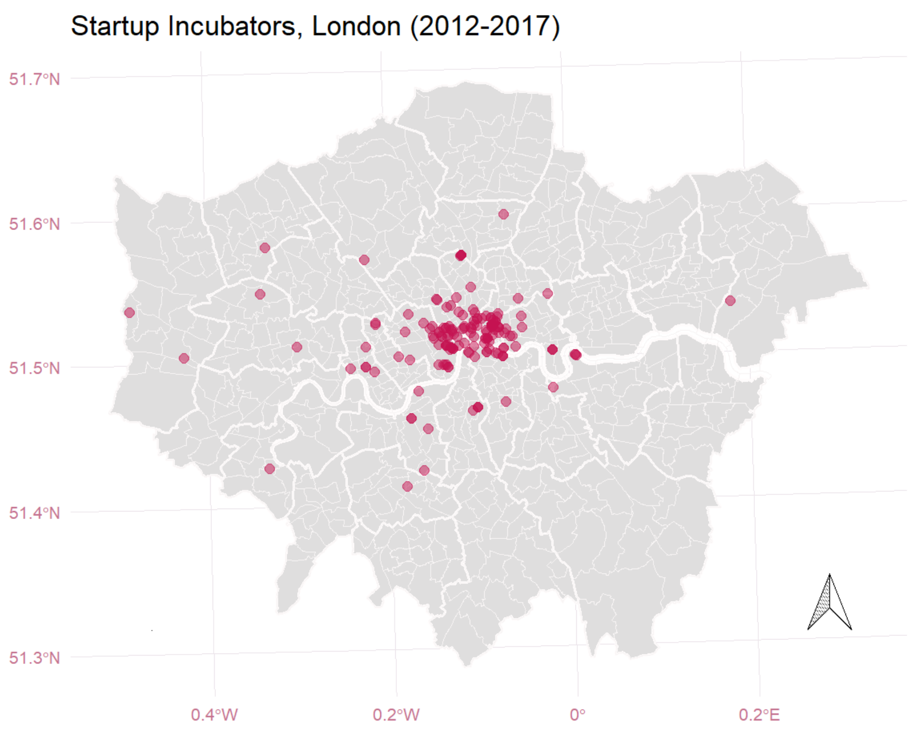


```
ggsave(file="London wards update.jpeg", width=7, height=7, dpi=300)
```

```
#retrieve lon lat from geometry column of incubators
inc_coords <- unlist(st_geometry(inc_SF_BNG)) %>%
  matrix(ncol=2,byrow=TRUE) %>%
  as_tibble() %>%
  setNames(c("lon","lat"))

#keeping relevant columns
inc_plot=inc_SF_BNG[,c(1,2,3)]

inc_plot$lon=inc_coords$lon
inc_plot$lat=inc_coords$lat
```

Final plot of incubators in London

```
library(ggsn)

ggplot(data=LondonWardsSF_BNG)+geom_sf(mapping = aes(geometry=geometry),data = LondonMap_BNG,fill="#dfded
e",colour="#fffcfc",lwd=0.7)+geom_sf(mapping = aes(geometry=geometry),data = LondonWardsSF_BNG,fill=NA,co
lour="#f8f5f5",lwd=0.3)+theme(panel.border = element_blank(),panel.background = element_rect(fill = 'whit
e',colour = "white"),panel.grid.major = element_line(colour = "#ece5eb"),panel.grid.minor = element_line(
colour="#ece5eb"), axis.title=element_text(size=15,face="bold"))+geom_point(data = inc_plot,aes(x=lon, y=
lat), size = 2,colour='#c51251',alpha=5/10)+ggtitle("Startup Incubators, London (2012-2017)")+labs(x = ""
, y = "") +coord_sf()+theme(axis.text = element_text(colour="#c57390",size=8), axis.ticks = element_blank
())+scalebar(data=LondonWardsSF_BNG,location="bottomright",y.min=155850.8, y.max=200933.9, x.min=503568.2
,x.max=561957.5, dist=200, dd2km= TRUE, model='WGS84',st.dist=0.1,st.size = 20,height = 1,anchor = c(x =5
06852.1,y = 158648.0 ))+north(data = NULL, location = "bottomright", scale = 0.1, symbol = 4,y.min=155850
.8, y.max=200933.9, x.min=503568.2,x.max=561957.5, anchor = c(x =561852.1,y = 158648.0 ))
```



```
ggsave(file="incubators update.jpeg", width=7, height=7, dpi=300)
```

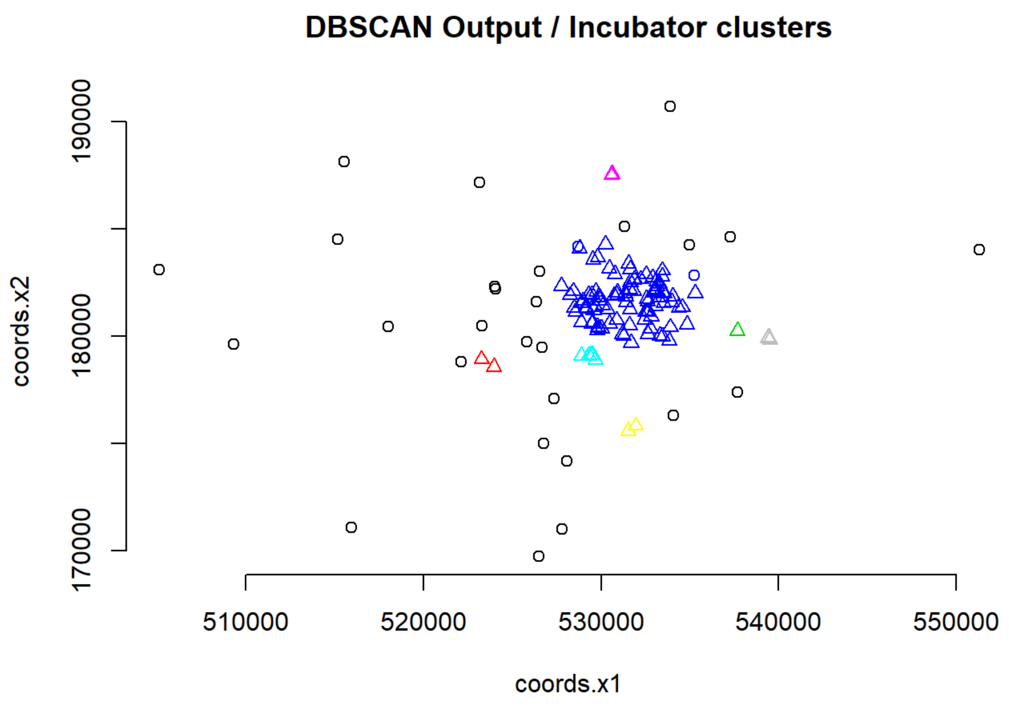
——- **PART 2 Cluster analysis of incubators** ———-

This part will identify clusters, using the **DBSCAN** method. A positive side of the DBSCAN algorithm is that it does not depend on the shape of the cluster (meaning it does not need to be circular) so it will identify any possible clusters, based on the **distance** and **minimum points** parameters that we set.

The cluster with the highest number of incubators will indicate our study area.

```
#additional packages that we will need for future use:

library(raster)
library(fpc)
library(plyr)
library(OpenStreetMap)
```

```
#first extract the points from the spatial points data frame
inc_SP=as(inc_SF_BNG,"Spatial")
inc_points <- data.frame(inc_SP@coords[,1:2])

# DBSCAN method, with minimum points of each cluster=3 and radius=1000m
db <- fpc::dbscan(inc_points, eps = 1000, MinPts = 3)

LondonMap_SP=as(LondonMap_BNG,"Spatial")

#plotting the results

plot(db, inc_points, main = "DBSCAN Output / Incubator clusters", frame = F)
```



```r
# Let's see how many clusters we have and how many points in each cluster
print(db)
```

```
## dbscan Pts=161 MinPts=3 eps=1000
##         0 1 2   3 4 5 6 7
## border 28 0 0   2 0 0 0 0
## seed    0 3 3 107 6 5 4 3
## total  28 3 3 109 6 5 4 3
```

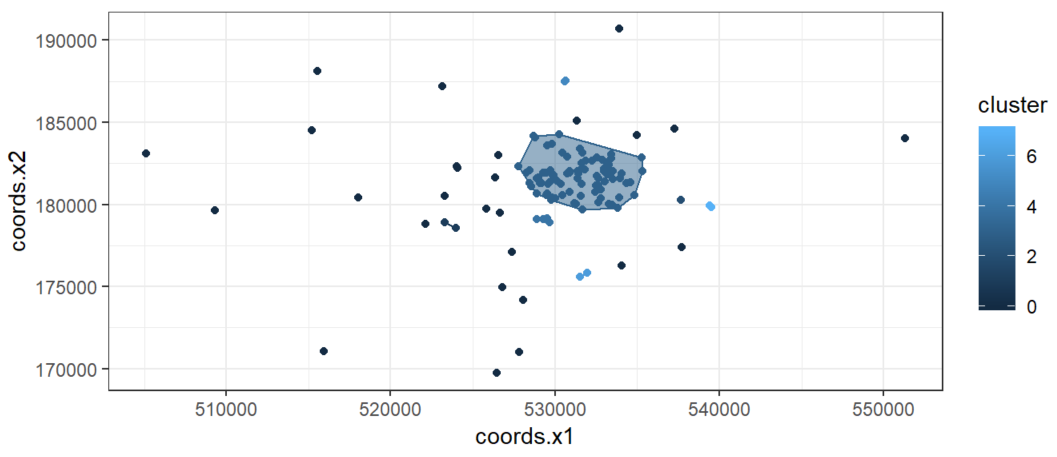
```r
db[["cluster"]]
```

```
##   [1] 1 2 3 3 3 3 3 3 3 3 3 3 0 3 3 3 4 3 0 4 5 5 0 3 3 3 3 3 4 0 3 0 6 3 7
##  [36] 3 3 0 3 4 3 3 3 5 3 3 3 3 3 3 3 5 3 3 3 3 3 0 3 3 3 3 3 3 3 3 6 6 3 0
##  [71] 0 3 3 3 3 3 3 0 0 3 0 4 3 3 3 0 3 3 1 3 3 3 0 3 3 3 0 7 0 3 3 3 0 3 3
## [106] 3 6 3 3 3 3 3 0 0 3 3 3 0 3 3 0 3 3 2 0 3 3 3 0 3 3 3 0 0 3 3 0 0 3 0
## [141] 3 5 3 1 2 3 3 3 3 3 3 4 3 3 3 3 3 3 3 3 7
```

```r
# assign to inc_points a column "cluster"" so that each incubator has an id of the cluster it belongs to
inc_points$cluster <- db$cluster
```

Plotting the clusters of incubators combined with the incubator points

```r
library(tidyverse)

#get the convex hull coordinates from the clusters
chulls <- ddply(inc_points, .(cluster), function(df) df[chull(df$coords.x1, df$coords.x2), ])

# cluster 0 is not an actual cluster, since it includes the outliers. So we keep only the clusters above
0
chulls <- subset(chulls, cluster>=1)

#plotting the clusters and the incubator points
dbplot <- ggplot(data=inc_points, aes(coords.x1,coords.x2, colour=cluster, fill=cluster))

dbplot <- dbplot + geom_point()

dbplot <- dbplot + geom_polygon(data = chulls, aes(x=coords.x1,y=coords.x2, group=cluster), alpha = 0.5)

#customizing visual elements, to be black and white
dbplot + theme_bw() + coord_equal()
```



```
ggsave(file="clusters map update.jpeg", width=7, height=7, dpi=300)
```

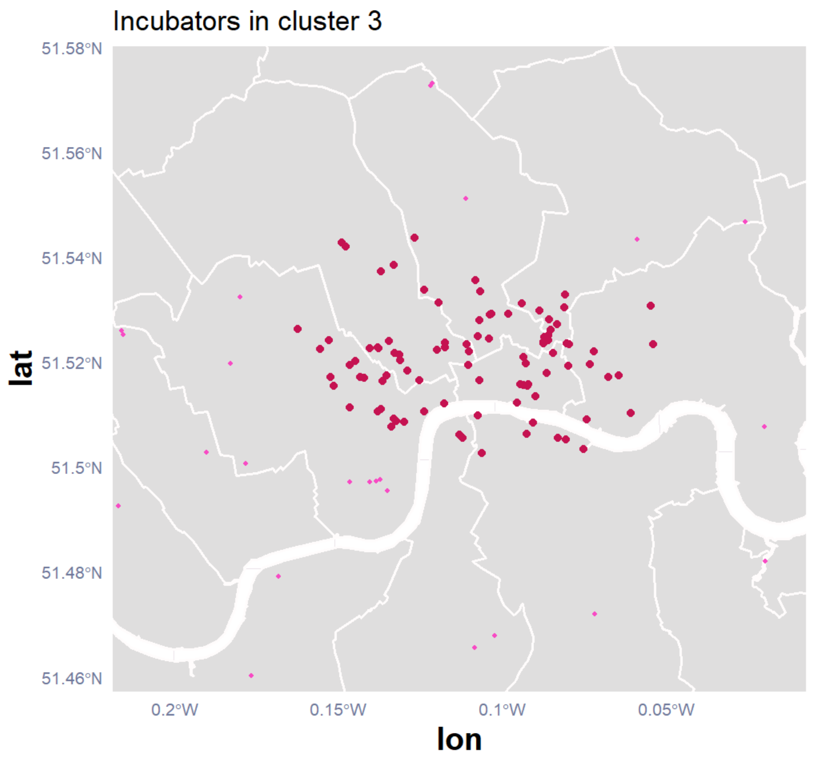
From the table showing the clusters and their respective points(incubator locations) we can see that cluster 3 has the highest number of incubators.

This will be our study area, therefore we need to extract the incubators within cluster 3.

```
#we will add to the sf object "inc_SF_BNG" the column "cluster" which contains the cluster ids for each p
oint

#then we will keep only the rows, which contain value "3" in their 'cluster' field.

inc_SF_BNG$clusters=inc_points$`cluster`

inc_SF_BNG_cluster3=inc_SF_BNG[inc_SF_BNG$clusters == "3", ]
```

Incubators within Cluster 3

```
ggplot()+geom_sf(mapping = aes(geometry=geometry),data = LondonMap_BNG,fill="#dfdede",colour="#fffcfc",lw
d=0.7)+geom_point(data = inc_plot,aes(x=lon, y=lat), size = 0.8,colour='#f84ac4')+geom_sf(aes(size =2), d
ata = inc_SF_BNG_cluster3,lwd=1.5,colour="#c51251")+north(data = NULL, location = "bottomright", scale =
0.1, symbol = 4,y.min=155850.8, y.max=200933.9, x.min=503568.2,x.max=561957.5, anchor = c(x =561852.1,y =
158648.0 ))+theme(axis.text = element_text(colour="#6f789a",size=8), axis.ticks = element_blank())+theme(
panel.border = element_blank(),panel.background = element_rect(fill = 'white',colour = "white"),panel.gri
d.major = element_line(colour = "#ece5eb"),panel.grid.minor = element_line(colour="#ece5eb"), axis.title=
element_text(size=15,face="bold"))+ggtitle("Incubators in cluster 3")+coord_sf()+scale_y_continuous(limit
s = c(174648.0 , 188327.6), expand = c(0, 0))+scale_x_continuous(limits = c(523852.1 , 538535.1), expand
= c(0, 0))
```



```
#note that we need to zoom in for Cluster 3, so we have set new scale_x_continuous and scale_y_continuous
limits, which are smaller than the bounding box values we retrieved in the beginning.


ggsave(file="incubators in cluster 3 update.jpeg", width=6, height=6, dpi=300)
```

—————**Part 2: Social spaces analysis** ————————

In this part we will import the cafe spaces of London and prepare it for the second part of our incubators / social spaces analysis.

```
#Read shp file of cafes

library(rgdal)
cafes_shp <- readOGR("C:/Users/User/Desktop/work/casa/0005 GIS/1. week 10 FINAL assignment/output/edited
data/cafes_osm.shp")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\User\Desktop\work\casa\0005 GIS\1. week 10 FINAL assignment\output\edited data\cafes_o
sm.shp", layer: "cafes_osm"
## with 2890 features
## It has 4 fields
```

```
class(cafes_shp)
```

```
## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"
```

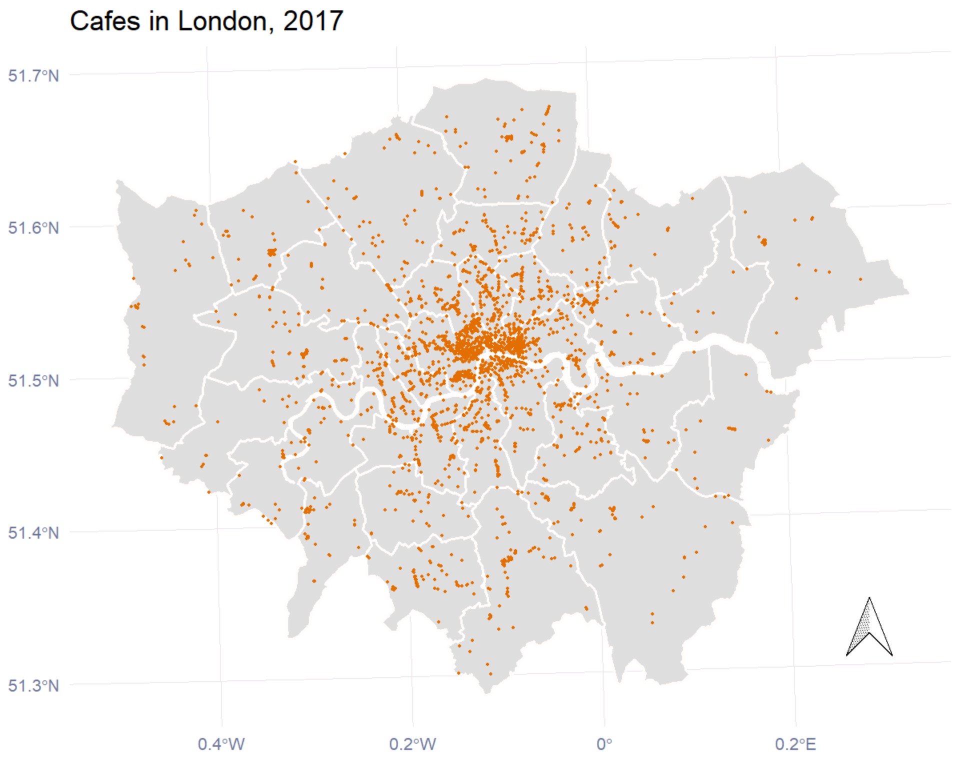
```
#convert sp object to sf
cafes_SF=st_as_sf(cafes_shp, coords = c("lon", "lat"), agr = "constant")



#reproject to the same CRS as the rest of the objects in our analysis
cafes_SF_BNG=st_transform(cafes_SF,27700)
```

Removing columns which contain info that won't be needed

```
cafes_SF_BNG=cafes_SF_BNG[,-c(2,3)]
```

Plot cafes in London

```
ggplot()+geom_sf(mapping = aes(geometry=geometry),data = LondonMap_BNG,fill="#dfdede",colour="#fffcfc",lw
d=0.7)+geom_sf(aes(size =2), data = cafes_SF_BNG,lwd=0.3,colour="#e36e00")+north(data = NULL, location =
"bottomright", scale = 0.1, symbol = 4,y.min=155850.8, y.max=200933.9, x.min=503568.2,x.max=561957.5, anc
hor = c(x =561852.1,y = 158648.0 ))+theme(axis.text = element_text(colour="#6f789a",size=8), axis.ticks =
element_blank())+theme(panel.border = element_blank(),panel.background = element_rect(fill = 'white',colo
ur = "white"),panel.grid.major = element_line(colour = "#ece5eb"),panel.grid.minor = element_line(colour=
"#ece5eb"), axis.title=element_text(size=15,face="bold"))+ggtitle("Cafes in London, 2017")
```



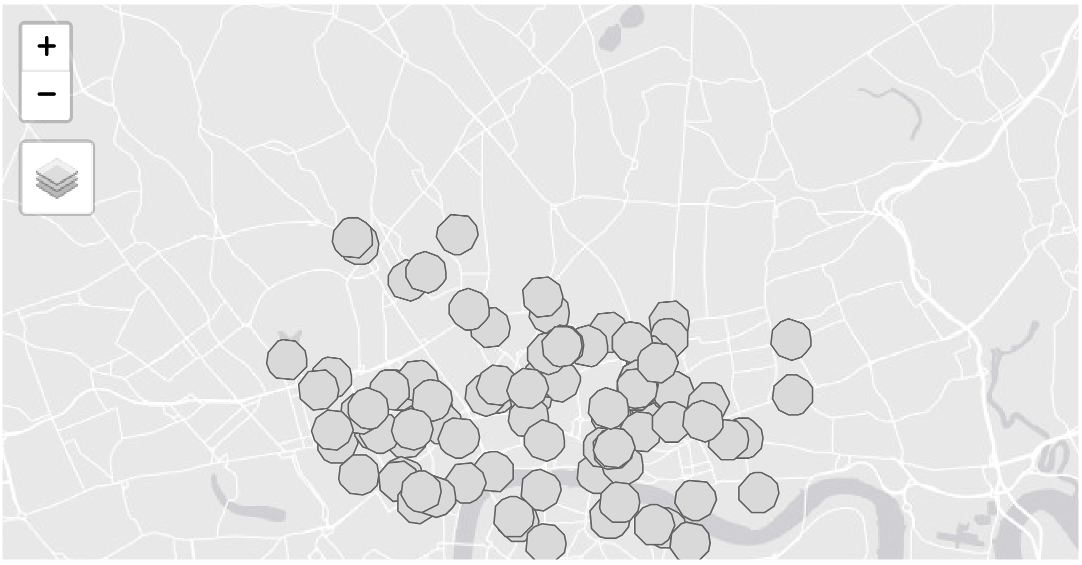
```
ggsave(file="cafes in London.jpeg", width=6, height=6, dpi=300)
```

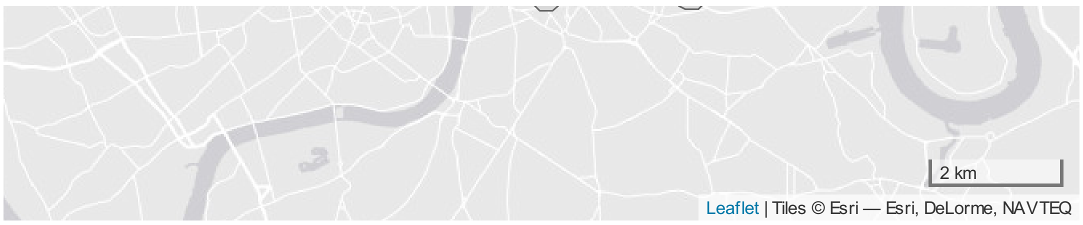
——— **PART 3: cafes - incubators analysis**———

In order to identify the incubators with the highest knowledge exchange potential, we need to calculate the number of cafes that each incubator has within a certain walking distance.

For this part, we will create a buffer around the incubators of 300m which could be translated into a 2-3 min walk from the incubator to the cafe.

```
#create buffer around incubators and plot it in a quick thematic map

inc_cluster3_buff=st_buffer(inc_SF_BNG_cluster3, dist = 300)
inc_cluster3_buff=inc_cluster3_buff[,2]

qtm(inc_cluster3_buff)
```





Counting the number of cafes within each buffer, using the GISTOOLS package.

```
library(GISTools)

inc_buffer_SP=as(inc_cluster3_buff,"Spatial")
cafes_SP=as(cafes_SF_BNG,"Spatial")



#counts the number of cafes and puts the values in a list
x1=poly.counts(cafes_SP, inc_buffer_SP)
class(x1)     #returns a list of values
```

```
## [1] "numeric"
```

```
#create a dataframe out of the list
y1=data.frame(x1)

#rename column in order to merge by col name
y1$'Programme name'=inc_cluster3_buff$`Programme name`

#merge with sf object of incubators
merged = merge(y1, inc_SF_BNG_cluster3, by = "Programme name")

#rename column for our convenience
colnames(merged)[2] <- "no of cafes nearby"
```

```
#put in descending order
merged_sorted <- merged[order(-merged[2]),]  # evala - gia descending

#create simple feature
merged_sorted_SF=st_as_sf(merged_sorted)
```

——— **Part 4 Visualisation: Preparing data for the final map** ————-

In this part, a map will be created showing the locations of incubators within Cluster 3 and their level of potential for knowledge exchange, based on the number of cafes close to them.

The next couple of code bits have to do with retrieving the lon lat data that will be needed for some parts of the ggplot and also for cropping the London Boroughs, so that the map is focused on the part of London where cluster 3 is.

Then, the road network of London is imported, in order to be added in the final map, for a more detailed visualisation.

```
#retrieving lon lat for the geometry column

seal_coords <- unlist(st_geometry(merged_sorted_SF)) %>%
  matrix(ncol=2,byrow=TRUE) %>%
  as_tibble() %>%
  setNames(c("lon","lat"))

final=merged_sorted_SF[,c(1,2,4)]

final$lon=seal_coords$lon
final$lat=seal_coords$lat
```

Preparing a LondonMap object which includes only the areas of London that we are interested in.

```
#setting coordinate system
final_BNG<- final%>% st_set_crs(27700)
London_whole <- LondonMap_BNG%>% st_set_crs(27700)
st_crs(LondonMap_BNG)
```

```
## Coordinate Reference System:
##   EPSG: 27700
##   proj4string: "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000 +ellps=airy +
towgs84=446.448,-125.157,542.06,0.1502,0.247,0.8421,-20.4894 +units=m +no_defs"
```

```
st_crs(final_BNG)
```

```
## Coordinate Reference System:
##   EPSG: 27700
##   proj4string: "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000 +ellps=airy +
towgs84=446.448,-125.157,542.06,0.15,0.247,0.842,-20.489 +units=m +no_defs"
```

In order to set the limits of the plot, we need to include the scale_x and scale_y settings, which require lon and lat values.

For this part, we will use again the st_bbox function

```
library(spatstat)
st_bbox(LondonMap_BNG)
```

```
##     xmin     ymin     xmax     ymax
## 503576.3 155850.7 561958.7 200934.0
```

Reading the road network of London and reprojecting it to the CRS (27700) we've been working on so far.

```
roads <- readOGR("C:/Users/User/Desktop/work/casa/0005 GIS/1. week 10 FINAL assignment/output/original da
ta/cafes osm/gis_osm_roads_free_1.shp")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\User\Desktop\work\casa\0005 GIS\1. week 10 FINAL assignment\output\original data\cafes
osm\gis_osm_roads_free_1.shp", layer: "gis_osm_roads_free_1"
## with 247928 features
## It has 10 fields
## Integer64 fields read as strings:  layer
```

```
roadsSF=st_as_sf(roads)
roadsSF=roadsSF%>% st_set_crs(4326)
roads_SF_BNG=st_transform(roadsSF,27700)

st_crs(roads_SF_BNG)
```

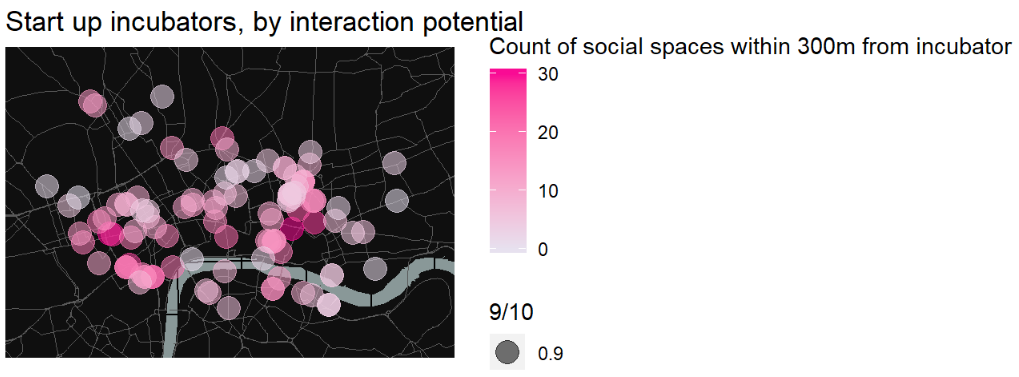
```
## Coordinate Reference System:
##   EPSG: 27700
##   proj4string: "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000 +ellps=airy +
towgs84=446.448,-125.157,542.06,0.15,0.247,0.842,-20.489 +units=m +no_defs"
```

Plotting the road network to see how it looks. We will separate the roads into primary, secondary and trunk, in order to experiment with which ones fit better in the scale of our map.

```
roads_SF_BNG=roads_SF_BNG%>% st_set_crs(27700)
LondonMap_BNG=LondonMap_BNG%>% st_set_crs(27700)

#primary streets
roads_primary=roads_SF_BNG[roads_SF_BNG$fclass==c('primary'),]

roads_primary=roads_primary[LondonMap_BNG,] #krataw mono entos twn wards pou thelw


#secondary streets
roads_secondary=roads_SF_BNG[roads_SF_BNG$fclass==c('secondary'),]

roads_secondary=roads_secondary[LondonMap_BNG,]   #krataw mono entos twn wards pou thelw

#trunk
roads_trunk=roads_SF_BNG[roads_SF_BNG$fclass==c('trunk'),]

roads_trunk=roads_trunk[LondonMap_BNG,]   #krataw mono entos twn wards pou thelw
```

Final plot of the Incubators in Cluster 3, by interaction potential.

```
ggplot()+geom_sf(mapping = aes(geometry=geometry),data = LondonMap_BNG,fill="#0f0f0f",colour=NA)+geom_sf(
mapping = aes(geometry=geometry),data = roads_primary,lwd=0.3)+geom_sf(mapping = aes(geometry=geometry),d
ata = roads_secondary,lwd=0.3)+geom_sf(mapping = aes(geometry=geometry),data = roads_trunk,lwd=0.3)+theme
(panel.border = element_blank(),panel.background = element_rect(fill = '#899898',colour = "#899898"),pane
l.grid.major = element_blank(),panel.grid.minor = element_blank())+geom_point(data = final_BNG,aes(x=lon,
y=lat,colour = final_BNG$`no of cafes nearby`,alpha = 9/10), size = 5)+labs(title = "Start up incubators,
by interaction potential",
       color = "Count of social spaces within 300m from incubator",
       x = "", y = "") +coord_sf()+scale_y_continuous(limits = c(178648.0 , 185327.6), expand = c(0, 0))+
scale_x_continuous(limits = c(526852.1 , 536535.1), expand = c(0, 0))+theme(axis.text = element_blank(),
axis.ticks = element_blank())+ scale_colour_gradient(low = "#e7e1ef", high = "#f90b94")
```


Start up incubators, by interaction potential
Count of social spaces within 300m from incubator
30
20
10
0
9/10
0.9

```
#note north symbol and scalebar could not be placed in correct color for a dark background, had to be add
ed manually

#saving plot
ggsave(file="final map2 update.jpeg", width=14, height=14, dpi=300)
```