# Sourcing external distances

*Don Li*

*12/06/2020*

## Join external path data

Sourcing path data from Azure Maps and OSRM. The source code for the functions needed is contained in `external_routing.R`.

```
library( data.table )
library( osrmr )
source( "G:/azure_hackathon/data/Don2/distance_functions.R" )

# Make a data frame to attach the OSRM distances
trip_summary = all_data[ , {
    trip_time = difftime( date_[.N], date_[1], units = "s" )
    trip_time = as.numeric(trip_time)
    list(
        lat1 = rawlat[1], lng1 = rawlng[1],
        lat2 = rawlat[.N], lng2 = rawlng[.N],
        known_dist = sum( H_dist ),
        known_time = trip_time
    )
}, by = "trj_id" ]

trip_summary[ , c("OSMR_dist") := {
    cat( "############\n" )
    status = "Fail"
    attempt = 0
    while( status == "Fail" ){
        attempt = attempt + 1
        message = paste0( "Request: ", trj_id,
            ". Attempt: ", attempt )
        cat( message, "\n" )

        route_info = osmr_routing( lat1, lng1, lat2, lng2, localhost = FALSE )

        if ( ! "try-error" %in% class(route_info) ){
            status = "Success"
        }
        message = paste0( "Status: ", status, "\n" )
        cat( message, "\n" )
    }
    n = which( trj_id == trip_summary$trj_id )
    message = paste0( "Trip ", n, " out of ", length(trip_summary$trj_id) )
    cat( message, "\n" )

    route_geom = route_info$routes[[1]]$geometry
    route_path = decode_geom( route_geom, precision = 5 )
    trip_dist = sum( haversine( route_path,"lat"], route_path,"lng"] ) )
    list( OSMR_dist = trip_dist )
```

```
}, by = "trj_id" ]

save( trip_summary,
    file = "G:/azure_hackathon/datasets2/external_paths/Azure_maps.RData" )
```

```
azure_key = readLines( "../data/keys_and_stuff/azure_maps.txt" )

trip_summary[ is.na(azure_dist), c("azure_dist", "azure_ETA") := {
    cat( trj_id, "\n" )
    route = try({
        azure_route( lat1, lng1, lat2, lng2, azure_key )
    })
    if ( "try-error" %in% class( route ) ){
        lat1 = round( lat1, 4 )
        lng1 = round( lng1, 4 )
        lat2 = round( lat2, 4 )
        lng2 = round( lng2, 4 )
        route = try({
            azure_route( lat1, lng1, lat2, lng2, azure_key )
        })
    }
    if ( "try-error" %in% class( route ) ){
        lat1 = round( lat1, 3 )
        lng1 = round( lng1, 3 )
        lat2 = round( lat2, 2 )
        lng2 = round( lng2, 2 )
        route = try({
            azure_route( lat1, lng1, lat2, lng2, azure_key )
        })
    }
    azure_dist = route$metadata$lengthInMeters/1000
    list( azure_dist = azure_dist )
}, by = "trj_id" ]

save( trip_summary,
    file = "G:/azure_hackathon/datasets2/external_paths/OSRM.RData" )
```

Join the datasets

```
load( "G:/azure_hackathon/datasets2/external_paths/Azure_maps.RData" )
azure_summary = trip_summary
load( "G:/azure_hackathon/datasets2/external_paths/OSRM.RData" )
osrm_summary = trip_summary

external_distance_summary = azure_summary
external_distance_summary[ osrm_summary, OSMR_dist := {
    i.OSMR_dist
}, on = "trj_id" ]

save( external_distance_summary,
    file = "G:/azure_hackathon/datasets2/external_paths/external_dist.RData" )
```