Final Project: Analyze

This script is meant to follow the Clean_Data scipt. This tool rasterizes demographic geometry with values corresponding to each demographic parameter. The rasters are clipped to the study extent and a fuzzy membership layer is computed. For the proximity data, a euclidean raster is computed for each feature class, clipped to the study area, and a fuzzy membership layer is computed. FOr the planned landuse data, the data is rasterized then reclassified to isolate business and commercial cells only. The fuzzy membership layers and appropriate layers are then used to perform a fuzzy overlay and weighted and unweighted sum with suitabilty scores.

In [ ]:
```
# calculate demographic percentages:
# percentage of population over 21
arcpy.management.AddField("demographics",
                          "per_over_21",
                          "FLOAT")

# calculate field and compensate for population equal to zero
arcpy.management.CalculateField("metro_tracts",
                                "metro_tracts.per_over_21",
                                "pop_percent(!demographics.pop_over_21!,!demog
raphics.total_pop!)",
                                "PYTHON3",
                                """def pop_percent(pop_21, total_pop):
    if total_pop == 0:
        percent = 0
        return percent
    else:
        percent = pop_21 / total_pop
        return percent""", "TEXT"
                                )
```

Create fuzzy membership layers for each demographic

All factors must be on the same scale to perform fuzzy overlay

In [ ]:
```
# Demographic variables:
metro_tracts_6_ = "metro_tracts"

# Convert tracts to raster with med income values
tracts_income = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\tem
p.gdb\\tracts_income"
arcpy.conversion.PolygonToRaster(in_features=metro_tracts_6_,
                                 value_field="metro_tracts.med_inc_fams_2",
                                 out_rasterdataset=tracts_income,
                                 cell_assignment="CELL_CENTER",
                                 priority_field="NONE",
                                 cellsize="100",
                                 build_rat="BUILD")
# Create fuzzy membership layer for med income
FuzzyMe_medinc = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\Bu
siness_FuzzyLogic.gdb\\FuzzyMe_medinc"
Fuzzy_Membership_4_ = FuzzyMe_medinc
FuzzyMe_medinc = arcpy.sa.FuzzyMembership(in_raster=tracts_income,
                                          fuzzy_function=[["LARGE", 125000, 5
]],
                                          hedge="NONE")
FuzzyMe_medinc.save(Fuzzy_Membership_4_)



# Convert tracts to raster with avg houehold size values
tracts_hhsize = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\tem
p.gdb\\tracts_hhsize"
arcpy.conversion.PolygonToRaster(in_features=metro_tracts_6_,
                                 value_field="metro_tracts.avg_hh_size_2",
                                 out_rasterdataset=tracts_hhsize,
                                 cell_assignment="CELL_CENTER",
                                 priority_field="NONE",
                                 cellsize="100",
                                 build_rat="BUILD")
# Create fuzzy membership layer for avg houehold size
FuzzyMe_hhsize = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\Bu
siness_FuzzyLogic.gdb\\FuzzyMe_hhsize"
Fuzzy_Membership_5_ = FuzzyMe_hhsize
FuzzyMe_hhsize = arcpy.sa.FuzzyMembership(in_raster=tracts_hhsize,
                                          fuzzy_function=[["LARGE", 2, 5]],
                                          hedge="NONE")
FuzzyMe_hhsize.save(Fuzzy_Membership_5_)



# Convert tracts to raster with avg family size values
tracts_famsize = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\te
mp.gdb\\tracts_famsize"
arcpy.conversion.PolygonToRaster(in_features=metro_tracts_6_,
                                 value_field="metro_tracts.avg_fam_size_2",
                                 out_rasterdataset=tracts_famsize,
                                 cell_assignment="CELL_CENTER",
                                 priority_field="NONE",
                                 cellsize="100",
                                 build_rat="BUILD")
```

```
# Create fuzzy membership layer for avg fam size
FuzzyMe_fam = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\Busin
ess_FuzzyLogic.gdb\\FuzzyMe_fam"
Fuzzy_Membership_6_ = FuzzyMe_fam
FuzzyMe_fam = arcpy.sa.FuzzyMembership(in_raster=tracts_famsize,
                                       fuzzy_function=[["LARGE", 2.61500000953
6743, 5]],
                                       hedge="NONE")
FuzzyMe_fam.save(Fuzzy_Membership_6_)



# Convert tracts to raster with percentage of population over 21 values
tracts_per_over_21 = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic
\\Business_FuzzyLogic.gdb\\tracts_per_over_21"
arcpy.conversion.PolygonToRaster(in_features=metro_tracts_6_,
                                 value_field="metro_tracts.per_over_21",
                                 out_rasterdataset=tracts_per_over_21,
                                 cell_assignment="CELL_CENTER",
                                 priority_field="NONE",
                                 cellsize="100",
                                 build_rat="BUILD")
# Create fuzzy membership layer for percentage of population over 21
FuzzyMe_pop21 = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\Bus
iness_FuzzyLogic.gdb\\FuzzyMe_pop21"
Fuzzy_Membership_7_ = FuzzyMe_pop21
FuzzyMe_pop21 = arcpy.sa.FuzzyMembership(in_raster=tracts_per_over_21, fuzzy_f
unction=[["LARGE", 0.5, 5]], hedge="NONE")
FuzzyMe_pop21.save(Fuzzy_Membership_7_)



# Convert tracts to raster with percentage in labor force values
tracts_labor = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\tem
p.gdb\\tracts_labor"
arcpy.conversion.PolygonToRaster(in_features=metro_tracts_6_,
                                 value_field="demographics.labor_force_rate",
                                 out_rasterdataset=tracts_labor,
                                 cell_assignment="CELL_CENTER",
                                 priority_field="NONE",
                                 cellsize="100",
                                 build_rat="BUILD")
# Create fuzzy membership layer for percentage  in labor force
FuzzyMe_labor = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\Bus
iness_FuzzyLogic.gdb\\FuzzyMe_labor"
Fuzzy_Membership_3_ = FuzzyMe_labor
FuzzyMe_labor = arcpy.sa.FuzzyMembership(in_raster=tracts_labor,
                                         fuzzy_function=[["LARGE", 129.5, 5]],
                                         hedge="NONE")
FuzzyMe_labor.save(Fuzzy_Membership_3_)
```

Create fuzzy membership layer for proximity variables and land use

- Proximity to roads, existing costcos, and competing businesses
- Planned land use for business and commercial development

In [ ]:
```
competing_metro = "competing_metro"
planned_LU = "planned_LU"
metro_tracts_7_ = "metro_tracts"
FunctionalClassRoads = "FunctionalClassRoads"
metro_tracts_8_ = "metro_tracts"
metro_tracts = "metro_tracts"
costcos_metro = "costcos_metro"


# Create raster with euclidean distance calculate from each competing business
point
competing_dist = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\te
mp.gdb\\competing_dist"
Euclidean_Distance = competing_dist
Output_direction_raster = ""
Out_back_direction_raster = ""
competing_dist = arcpy.sa.EucDistance(in_source_data=competing_metro,
                                      maximum_distance=None, cell_size="100",
                                      out_direction_raster=Output_direction_ra
ster,
                                      distance_method="GEODESIC",
                                      in_barrier_data="",
                                      out_back_direction_raster=Out_back_direc
tion_raster)
competing_dist.save(Euclidean_Distance)

# Clip raster study extent
competing_stores_clipped = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_Fuzz
yLogic\\temp.gdb\\competing_stores_clipped"
arcpy.management.Clip(in_raster=competing_dist,
                      rectangle="419967.4723 4924223.7912 521254.6998 5029009.
8723",
                      out_raster=competing_stores_clipped,
                      in_template_dataset=metro_tracts,
                      nodata_value="3.4e+38",
                      clipping_geometry="ClippingGeometry",
                      maintain_clipping_extent="NO_MAINTAIN_EXTENT")
competing_stores_clipped = arcpy.Raster(competing_stores_clipped)

# Create fuzzy membership layer for proximity to competing businesses
FuzzyMe_competing = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic
\\Business_FuzzyLogic.gdb\\FuzzyMe_competing"
Fuzzy_Membership_2_ = FuzzyMe_competing
FuzzyMe_competing = arcpy.sa.FuzzyMembership(in_raster=competing_stores_clippe
d,
                                             fuzzy_function=[["LARGE", 1, 1]],
                                             hedge="NONE")
FuzzyMe_competing.save(Fuzzy_Membership_2_)



# Rasterize planned landuse
plannedLU_rast = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\te
mp.gdb\\plannedLU_rast"
with arcpy.EnvManager(snapRaster=r"C:\Users\msong\Desktop\arc2proj\Business_Fu
zzyLogic\Business_FuzzyLogic.gdb\competing_rescale"):
```

```
    arcpy.conversion.FeatureToRaster(in_features=planned_LU,
                                     field="REGIONAL_N",
                                     out_raster=plannedLU_rast,
                                     cell_size="100")
    plannedLU_rast = arcpy.Raster(plannedLU_rast)

# Clip planned landuse to metro counties
plannedLU_rast_Clip = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogi
c\\temp.gdb\\plannedLU_rast_Clip"
arcpy.management.Clip(in_raster=plannedLU_rast,
                      rectangle="419967.4723 4924223.7912 521254.6998 5029009.
8723",
                      out_raster=plannedLU_rast_Clip,
                      in_template_dataset=metro_tracts_7_,
                      nodata_value="65535",
                      clipping_geometry="ClippingGeometry",
                      maintain_clipping_extent="NO_MAINTAIN_EXTENT")
plannedLU_rast_Clip = arcpy.Raster(plannedLU_rast_Clip)

# Reclassify to isolate planned areas for business and commercial
planned_LU_reclass = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic
\\Business_FuzzyLogic.gdb\\planned_LU_reclass"
arcpy.ddd.Reclassify(in_raster=plannedLU_rast_Clip,
                     reclass_field="Value",
                     remap="120 299 0;300 499 1;500 699 0;700 799 1;800 1200
 0",
                     out_raster=planned_LU_reclass,
                     missing_values="NODATA")
planned_LU_reclass = arcpy.Raster(planned_LU_reclass)



# Create raster with euclidean distance calculated from major roads
Roads_dist = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\temp.g
db\\Roads_dist"
Euclidean_Distance_2_ = Roads_dist
Output_direction_raster_2_ = ""
Out_back_direction_raster_2_ = ""
with arcpy.EnvManager(snapRaster="planned_LU_reclass"):
    Roads_dist = arcpy.sa.EucDistance(in_source_data=FunctionalClassRoads,
                                      maximum_distance=None,
                                      cell_size="100",
                                      out_direction_raster=Output_direction_ra
ster_2_,
                                      distance_method="PLANAR",
                                      in_barrier_data="",
                                      out_back_direction_raster=Out_back_direc
tion_raster_2_)
    Roads_dist.save(Euclidean_Distance_2_)

# Clip euclidean distance raster for roads to metro counties
Roads_dist_Clip = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\t
emp.gdb\\Roads_dist_Clip"
arcpy.management.Clip(in_raster=Roads_dist,
                      rectangle="419967.4723 4924223.7912 521254.6998 5029009.
8723",
```

```
                      out_raster=Roads_dist_Clip,
                      in_template_dataset=metro_tracts_8_,
                      nodata_value="3.4e+38",
                      clipping_geometry="ClippingGeometry",
                      maintain_clipping_extent="NO_MAINTAIN_EXTENT")
Roads_dist_Clip = arcpy.Raster(Roads_dist_Clip)

# Create fuzzy membership layer for proximity to roads
FuzzyMe_Roads = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\Bus
iness_FuzzyLogic.gdb\\FuzzyMe_Roads"
Fuzzy_Membership = FuzzyMe_Roads
FuzzyMe_Roads = arcpy.sa.FuzzyMembership(in_raster=Roads_dist_Clip,
                                         fuzzy_function=[["SMALL", 1, 1]],
                                         hedge="NONE")
FuzzyMe_Roads.save(Fuzzy_Membership)



# Create raster with euclidean distance calculated from existing costco locati
ons
existing_costcos_dist = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLo
gic\\temp.gdb\\existing_costcos_dist"
Euclidean_Distance_3_ = existing_costcos_dist
Output_direction_raster_3_ = ""
Out_back_direction_raster_3_ = ""
with arcpy.EnvManager(extent="419967.4723 4924223.7912 521254.6998 5029009.872
3",
                      snapRaster=r"C:\Users\msong\Desktop\arc2proj\Business_Fu
zzyLogic\temp.gdb\plannedLU_rast_Clip"):
    existing_costcos_dist = arcpy.sa.EucDistance(in_source_data=costcos_metro,
                                                 maximum_distance=None,
                                                 cell_size="100",
                                                 out_direction_raster=Output_d
irection_raster_3_,
                                                 distance_method="GEODESIC", i
n_barrier_data="",
                                                 out_back_direction_raster=Out
_back_direction_raster_3_)
    existing_costcos_dist.save(Euclidean_Distance_3_)

# Clip euclidean distance raster for proximity to existing costcos to the metr
o tracts
existing_costcos_dist_Clip = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_Fu
zzyLogic\\Business_FuzzyLogic.gdb\\existing_costcos_dist_Clip"
arcpy.management.Clip(in_raster=existing_costcos_dist,
                      rectangle="419967.4723 4924223.7912 521254.6998 5029009.
8723",
                      out_raster=existing_costcos_dist_Clip,
                      in_template_dataset=metro_tracts_7_, nodata_value="",
                      clipping_geometry="ClippingGeometry",
                      maintain_clipping_extent="NO_MAINTAIN_EXTENT")
existing_costcos_dist_Clip = arcpy.Raster(existing_costcos_dist_Clip)

# Create fuzzy membership layer for proximity to existing costcos
FuzzyMe_costcos = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\B
usiness_FuzzyLogic.gdb\\FuzzyMe_costcos"
Fuzzy_Membership_8_ = FuzzyMe_costcos
```

```
with arcpy.EnvManager(snapRaster=r"C:\Users\msong\Desktop\arc2proj\Business_FuzzyLogic\temp.gdb\plannedLU_rast_Clip"):
    FuzzyMe_costcos = arcpy.sa.FuzzyMembership(in_raster=existing_costcos_dist_Clip,
                                               fuzzy_function=[["LARGE", 1, 1]],
                                               hedge="NONE")
    FuzzyMe_costcos.save(Fuzzy_Membership_8_)
```

Fuzzy Overlay Analysis

In [ ]:
```
FuzzyOv_Fuzz = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLogic\\Business_FuzzyLogic.gdb\\FuzzyOv_Fuzz"
Fuzzy_Overlay = FuzzyOv_Fuzz
FuzzyOv_Fuzz = arcpy.sa.FuzzyOverlay(in_rasters=[FuzzyMe_competing,
                                                FuzzyMe_fam,
                                                FuzzyMe_hhsize,
                                                FuzzyMe_medinc,
                                                FuzzyMe_pop21,
                                                FuzzyMe_Roads,
                                                FuzzyMe_costcos,
                                                FuzzyMe_labor,
                                                planned_LU_reclass],
                                     overlay_type="AND",
                                     gamma=0.9)
FuzzyOv_Fuzz.save(Fuzzy_Overlay)
```

Weighted Sum Analysis

In [ ]:
```
# Equal weight sum
Weighte_Fuzz_noweight = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLo
gic\\Business_FuzzyLogic.gdb\\Weighte_Fuzz_noweight"
Weighted_Sum = Weighte_Fuzz_noweight
Weighte_Fuzz_noweight = arcpy.sa.WeightedSum(in_rasters=[[FuzzyMe_competing,
"VALUE", 1],
                                                         [FuzzyMe_fam, "VALUE"
, 1],
                                                         [FuzzyMe_hhsize, "VAL
UE", 1],
                                                         [FuzzyMe_medinc, "VAL
UE", 1],
                                                         [FuzzyMe_pop21, "VALU
E", 1],
                                                         [FuzzyMe_Roads, "VALU
E", 1],
                                                         [planned_LU_reclass,
"Value", 1],
                                                         [FuzzyMe_costcos, "VA
LUE", 1],
                                                         [FuzzyMe_labor, "VALU
E", 1]])
Weighte_Fuzz_noweight.save(Weighted_Sum)
```

In [ ]:
```
# Unequal weighted sum
Weighte_Fuzz_weighted = "C:\\Users\\msong\\Desktop\\arc2proj\\Business_FuzzyLo
gic\\Business_FuzzyLogic.gdb\\Weighte_Fuzz_weighted"
Weighted_Sum = Weighte_Fuzz_weighted
Weighte_Fuzz_weighted = arcpy.sa.WeightedSum(in_rasters=[[FuzzyMe_competing,
"VALUE", 1],
                                                         [FuzzyMe_fam, "VALUE"
, 2],
                                                         [FuzzyMe_hhsize, "VAL
UE", 2],
                                                         [FuzzyMe_medinc, "VAL
UE", 2],
                                                         [FuzzyMe_pop21, "VALU
E", 2],
                                                         [FuzzyMe_Roads, "VALU
E", 1],
                                                         [planned_LU_reclass,
"Value", 1],
                                                         [FuzzyMe_costcos, "VA
LUE", 3],
                                                         [FuzzyMe_labor, "VALU
E", 2]])
Weighte_Fuzz_weighted.save(Weighted_Sum)
```

In [ ]:
```

```