CAPSTONE PROJECT:

THE BATTLE OF NEIGHBOURHOODS

INDEX

I. INTRODUCTION/BUSINESS PROBLEM

This analysis explores the most suitable locations for a business personnel to invest or open a Finger millet (locally known as 'Ragi') processing unit in the state of Tamilnadu in India. Tamil Nadu is the tenth largest Indian state by area and the sixth largest by population and comprise of 37 districts. The economy of Tamil Nadu is the second-largest state economy in India. Tamil Nadu has historically been an agricultural state and is a leading producer of agricultural products in India. The Cauvery delta region is known as the Rice Bowl of Tamil Nadu.

Scenario and Objective:

Business personnel who wants to invest or open a Finger millet (locally known as 'Ragi') processing unit as a side business. The objective is to find the best locations that has good amount of irrigated area of high-yield Finger millet, good amenities/facilities and connectivity to other cities.

Methodology:

1. Extract the data of Finger millet irrigated area for each of the districts in TamilNadu from https://tn.data.gov.in.
2. Using Foursquare API we will get all venues for each of the districts in Tamil Nadu
3. Exploratory analysis and Data Visualization
4. Clustering the districts using K-Means
5. Compare the districts to find the best Places for setting up of Finger millet processing unit
6. Inference and relevant conclusions

II. DATA COLLECTION

For this project we need the following data:

1. The data that contains Finger millet irrigated area for each of the districts in TamilNadu is extracted,
   - Data Source: https://tn.data.gov.in
   - Description: This data set contains the required information. And we will use this data set to explore the Finger millet irrigated area. The geographical coordinates are appended to this data using the geocoder.
2. All the venues for each of the distric are extracted using Foursquare API
   - Data Source: Foursquare API
   - Description: By using this API we will get all the venues in each of the districts in Tamil Nadu.

III. EXPLORATORY DATA ANALYSIS

The raw data of 'Finger millet irrigated area' in 32 districts of Tamilnadu. There are 11 columns including data regarding irrigated, un-irrigated for high, local yield etc., A few entries of the table is provided below,

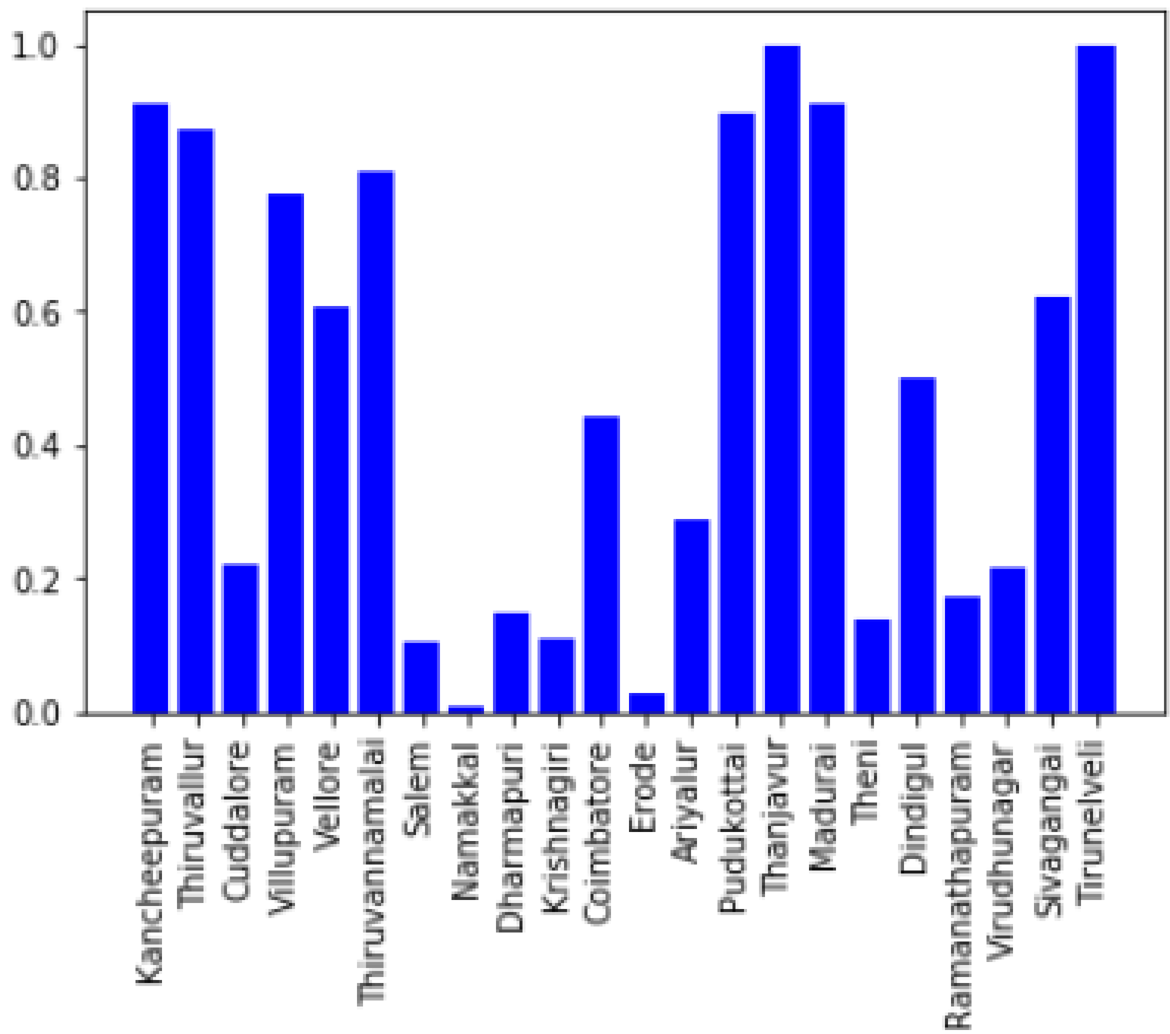
	Sl.No	District	Irrigated Area for High Yielding variety of Ragi (in ha)	Un-Irrigated Area for High Yielding variety of Ragi (in ha)	Total Area for High Yielding variety of Ragi (in ha)	Irrigated Area for Local Yielding variety of Ragi (in ha)	Un-Irrigated Area for Local Yielding variety of Ragi (in ha)	Total Area for Local Yielding variety of Ragi (in ha)	Total irrigated Area for Ragi (in ha)	Total Un irrigated Area for Ragi (in ha)	Total Area for Ragi (in ha)
0	1	Kancheepuram	300	22	322	5	3	8	305	25	330
1	2	Thiruvallur	218	17	235	7	8	15	225	25	250
2	3	Cuddalore	35	124	159	0	0	0	35	124	159
3	4	Villupuram	953	247	1200	10	17	27	963	264	1227
4	5	Vellore	2733	1783	4516	3	0	3	2736	1783	4519

Since the objective is determine the districts with high proportion of irrigated area for high yielding variety of ragi and decently equipped with ameneties/facilities and has good connectivity.

	District	Irrigated Area for High Yielding variety of Ragi (in ha)	Total Area for Ragi (in ha)	proportion
0	Kancheepuram	300	330	0.909091
1	Thiruvallur	218	250	0.872000
2	Cuddalore	35	159	0.220126
3	Villupuram	953	1227	0.776691
4	Vellore	2733	4519	0.604780

To further the analysis, those districts with proportion of at least 0.005 of Ragi (high yield) irrigated area were filtered.

A barplot of those filtered districts, depicting the proportion of Ragi (high yield) irrigated area,



It can be observed from the plot, that the districts 'Tirunelveli' and 'Thanjavur' has higher proportion of Irrigated Area for High Yielding variety of Ragi.

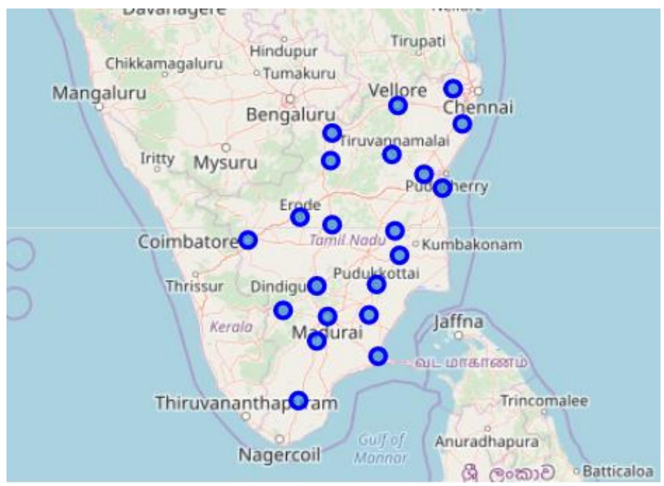
Further, to determine the feasible location for the investment- amenities, facilities and the connectivity need to be assessed.

IV. DATA PREPARATION AND PROCESSING

We add the geographical coordinates to these 22 districts and plot them on the map,

```
geolocator = Nominatim(user_agent="district")
df1['District_coordinates']=
df1['District'].apply(geolocator.geocode).apply(lambda x: (x.latitude,
x.longitude))
df1[['Latitude', 'Longitude']] = df1['District_coordinates'].apply(pd.Series)
df1.drop(columns='District_coordinates')
```

	District	proportion	Latitude	Longitude
0	Kancheepuram	0.909091	12.637438	80.045914
1	Thiruvallur	0.872000	13.139436	79.907304
2	Cuddalore	0.220126	11.742694	79.750306
3	Villupuram	0.776691	11.939829	79.494564
4	Vellore	0.604780	12.907175	79.130970
5	Thiruvannamalai	0.808680	12.208554	79.037947
6	Salem	0.106943	44.939157	-123.033121
7	Namakkal	0.009747	11.219169	78.167870
8	Dharmapuri	0.149781	12.134799	78.158986
9	Krishnagiri	0.110740	12.513614	78.174025
10	Coimbatore	0.444444	11.001812	76.962842
11	Erode	0.025228	11.330648	77.727652
12	Ariyalur	0.289474	11.135771	79.072320
13	Pudukottai	0.894737	10.375158	78.816734
14	Thanjavur	1.000000	10.786027	79.138150
15	Madurai	0.913043	9.926115	78.114098
16	Theni	0.139535	10.010814	77.481010
17	Dindigul	0.500000	10.365541	77.969585
18	Ramanathapuram	0.171053	9.365235	78.834319
19	Virudhunagar	0.216216	9.582240	77.953683
20	Sivagangai	0.619048	9.950117	78.696000
21	Tirunelveli	1.000000	8.729526	77.685235



We get the nearby venues using FourSquare and append to our data. And the sample data looks like,

```
CLIENT_ID = '    ' # Foursquare ID
CLIENT_SECRET = '    ' # Foursquare Secret
VERSION = '20191008' # Foursquare API version

print('Your credentials:')
print('CLIENT_ID: ' + CLIENT_ID)
```

```
print('CLIENT_SECRET:' + CLIENT_SECRET)


def getNearbyVenues(names, latitudes, longitudes, radius=4000, LIMIT=100):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)

        # create the API request URL
        url =
'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&
v={}&ll={},{}&radius={}&limit={}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']

        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item
in venue_list])
    nearby_venues.columns = ['District',
                  'District Latitude',
                  'District Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
location_venues =
getNearbyVenues(names=df1['District'],latitudes=df1['Latitude'],longitudes=df
1['Longitude'])
```

	District	District Latitude	District Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Thiruvallur	13.139436	79.907304	Thiruvallur Railway Station	13.116644	79.914239	Train Station
1	Thiruvallur	13.139436	79.907304	Sennetta Hotel	13.111584	79.914530	Hotel
2	Thiruvallur	13.139436	79.907304	Kamal Salon	13.111209	79.897965	Cosmetics Shop
3	Thiruvallur	13.139436	79.907304	Putlur Railway Station	13.123250	79.936865	Train Station
4	Thiruvallur	13.139436	79.907304	Hotel Sri Sai Vinayaga	13.109010	79.921330	Hotel
5	Thiruvallur	13.139436	79.907304	Tirupachur	13.139416	79.872124	Historic Site
6	Cuddalore	11.742694	79.750306	A2B Restaurant	11.750600	79.760761	Vegetarian / Vegan Restaurant
7	Cuddalore	11.742694	79.750306	Cuddalore Bus Stand	11.746475	79.755533	Bus Station
8	Cuddalore	11.742694	79.750306	Adayar Ananda Bhavan	11.750629	79.760783	Vegetarian / Vegan Restaurant
9	Cuddalore	11.742694	79.750306	MORE. Super Market	11.761534	79.753802	Department Store

The processing of this data include two steps which would be used later for modeling

1. Grouping the venues

```
# one hot encoding
venues_onehot = pd.get_dummies(location_venues[['Venue Category']],
prefix="", prefix_sep="")

# add street column back to dataframe
venues_onehot['District'] = location_venues['District']

# move street column to the first column
fixed_columns = [venues_onehot.columns[-1]] +
list(venues_onehot.columns[:-1])

#fixed_columns
venues_onehot = venues_onehot[fixed_columns]

venues_onehot.head()
TamilNadu_grouped = venues_onehot.groupby('District').mean().reset_index()
TamilNadu_grouped
```

	District	ATM	Accessories Store	Airport	American Restaurant	Arcade	Asian Restaurant	BBQ Joint	Bakery	Bank	...	Steakhouse	Sushi Restaurant	Theater	Theme Park Ride / Attraction	Thrift / Vintage Store	Train Station
0	Ariyalur	0.0	0.00	0.0	0.0	0.0	0.00	0.00	0.00	0.0	...	0.00	0.0	0.0	0.0	0.0	0.000000
1	Coimbatore	0.0	0.01	0.0	0.0	0.0	0.07	0.01	0.02	0.0	...	0.01	0.0	0.0	0.0	0.0	0.010000
2	Cuddalore	0.0	0.00	0.0	0.0	0.0	0.00	0.00	0.00	0.0	...	0.00	0.0	0.0	0.0	0.0	0.000000
3	Dharmapuri	0.0	0.25	0.0	0.0	0.0	0.00	0.00	0.00	0.0	...	0.00	0.0	0.0	0.0	0.0	0.250000
4	Dindigul	0.0	0.00	0.0	0.0	0.0	0.00	0.00	0.00	0.0	...	0.00	0.0	0.0	0.0	0.0	0.142857

2. Finding the frequency of venues and the top venues with each district.

```
# Define a function to return the most common venues/facilities nearby
real estate investments
```

```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]


num_top_venues = 10
indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['District']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1,
indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))
# create a new dataframe
venues_sorted = pd.DataFrame(columns=columns)
venues_sorted['District'] = TamilNadu_grouped['District']

for ind in np.arange(TamilNadu_grouped.shape[0]):
    venues_sorted.iloc[ind, 1:] =
return_most_common_venues(TamilNadu_grouped.iloc[ind, :], num_top_venues)
venues_sorted.head()
```

	District	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue
0	Ariyalur	South Indian Restaurant	Platform	Indian Restaurant	Currency Exchange	Farmers Market	Cosmetics Shop	Costume Shop	Cupcake Shop	Department Store
1	Coimbatore	Indian Restaurant	Clothing Store	Asian Restaurant	Pizza Place	Ice Cream Shop	Multiplex	Hotel	Fast Food Restaurant	Café
2	Cuddalore	Vegetarian / Vegan Restaurant	Bus Station	Department Store	Women's Store	Farmers Market	Cosmetics Shop	Costume Shop	Cupcake Shop	Currency Exchange
3	Dharmapuri	South Indian Restaurant	Accessories Store	Train Station	Coffee Shop	Fast Food Restaurant	Cosmetics Shop	Costume Shop	Cupcake Shop	Currency Exchange
4	Dindigul	Multicuisine Indian Restaurant	Train Station	Indian Restaurant	Multiplex	Restaurant	Bus Station	Women's Store	Diner	Costume Shop

V. MODELING

Using the data of Venue frequency, we cluster the districts using Kmeans similarity.

```
from sklearn.cluster import KMeans
# set number of clusters
kclusters = 5
TamilNadu_grouped_clustering = TamilNadu_grouped.drop('District', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters,
random_state=0).fit(TamilNadu_grouped_clustering)

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]

array([4, 3, 0, 3, 3, 3, 4, 1, 4, 3], dtype=int32)

# add clustering labels
venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)
TamilNadu_merged = df1
merged = pd.merge(TamilNadu_merged, venues_sorted, on='District')
merged.head()
```

	District	proportion	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue
0	Thiruvallur	0.872000	13.139436	79.907304	3	Hotel	Train Station	Cosmetics Shop	Historic Site	Airport	Fast Food Restaurant	Costume Shop
1	Cuddalore	0.220126	11.742694	79.750306	0	Vegetarian / Vegan Restaurant	Bus Station	Department Store	Women's Store	Farmers Market	Cosmetics Shop	Costume Shop
2	Villupuram	0.776691	11.939829	79.494564	1	Costume Shop	Indian Restaurant	Light Rail Station	Flea Market	Women's Store	Fast Food Restaurant	Cupcake Shop
3	Vellore	0.604780	12.907175	79.130970	4	Indian Restaurant	Hotel	Bank	Historic Site	Department Store	Chinese Restaurant	Market
4	Thiruvannamalai	0.808680	12.208554	79.037947	1	Vegetarian / Vegan Restaurant	Indian Restaurant	Café	Resort	Mountain	Women's Store	Electronics Store

The clusters plotted on map as below,

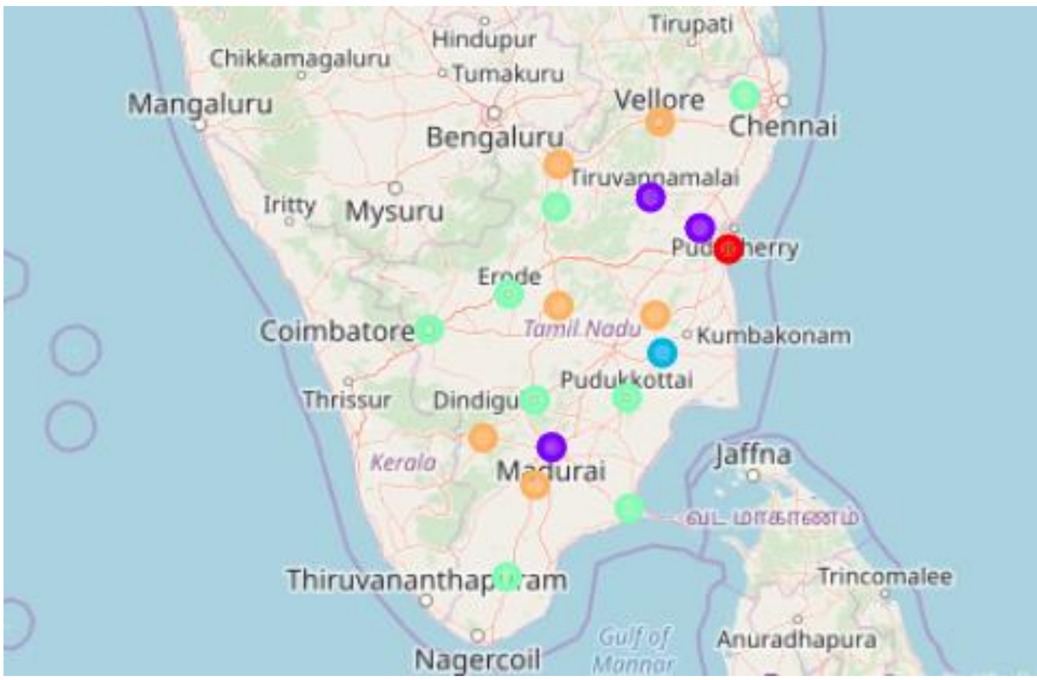
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=7)

# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(merged['Latitude'], merged['Longitude'],
merged['District'], merged['Cluster Labels']):
```

```
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster),
parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```



VI. DISCUSSION/CONCLUSION

This is also appended to the data with venues listed in order of frequency.

```
merged.loc[merged['Cluster Labels'] == 3, merged.columns[[0] + list(range(5, merged.shape[1]))]]
```

]:

	District	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Thiruvallur	Hotel	Train Station	Cosmetics Shop	Historic Site	Airport	Fast Food Restaurant	Costume Shop	Cupcake Shop	Currency Exchange	Department Store
5	Salem	Coffee Shop	American Restaurant	Bar	Pizza Place	Park	Mexican Restaurant	Brewery	Breakfast Spot	Movie Theater	Pub
7	Dharmapuri	South Indian Restaurant	Accessories Store	Train Station	Coffee Shop	Fast Food Restaurant	Cosmetics Shop	Costume Shop	Cupcake Shop	Currency Exchange	Department Store
9	Coimbatore	Indian Restaurant	Clothing Store	Asian Restaurant	Pizza Place	Ice Cream Shop	Multiplex	Hotel	Fast Food Restaurant	Café	Restaurant
10	Erode	Ice Cream Shop	Pizza Place	Food & Drink Shop	Clothing Store	Restaurant	Farmers Market	Food Court	Fried Chicken Joint	Grocery Store	Cosmetics Shop
12	Pudukottai	Movie Theater	ATM	Bus Station	Indian Restaurant	Motorcycle Shop	Coffee Shop	Asian Restaurant	Cupcake Shop	Department Store	Dessert Shop
16	Dindigul	Multicuisine Indian Restaurant	Train Station	Indian Restaurant	Multiplex	Restaurant	Bus Station	Women's Store	Diner	Costume Shop	Cupcake Shop
17	Ramanathapuram	Department Store	Hotel	Shopping Mall	Bus Station	Fast Food Restaurant	Historic Site	Seafood Restaurant	ATM	Train Station	Sushi Restaurant
19	Tirunelveli	Hotel	Indian Restaurant	Train Station	Clothing Store	Bakery	Dessert Shop	Fast Food Restaurant	Costume Shop	Cupcake Shop	Currency Exchange

Considering all the clusters, we find the cluster'3' is relatively equipped with amenities. And considering the irrigated area for Ragi, 'Tirunelveli' could be a good choice to invest or open a Finger Millet processing unit as a side business.