Implementational Details

This section is further divided into the following subsections :

Files and Folder Description

The following is the directory structure of the repository and a little detail about the purpose of the files.

▼ **Creating_features**

This folder contains code and scripts to the input features for training the ML models. It covers all processes from downloading the complete India polygon to making the features for that year. Note that the files in this folder are hardcoded for the year 2011 but can be used for other years as well by just changing the year. This folder contains the following files and folders.

- **downloadDistrictTiff.js:** This file is used to download the complete India polygon from the Google Earth Engine. On copying and Running this script on the GEE console a 500m .tif file will be stored in your Google drive.
- **split.py:** This file is used to break the complete .tif file into individual districts based on their shape in the 2011_Dist.geojson file. The tiff image

of individual district will be stored in the distTiffFiles500_2011 folder in this directory. The name of the saved file will be in the format of *{district_name}@{ST_CEN_CD}@{CensusCode}.tiff*

- **2011_Dist.geojson**: This file contains information about the shape and census code of individual districts.
- **distTiffFiles500_2011**: On running the split.py script all the district files will be stored in this folder.
- **landsat7_india_500_2011-01-01_2011-12-31.tif**: This is the complete India polygon downloaded from GEE.
- **create_bins.py**: This file is used to perform quantile binning on the each of the bands. It creates the bin boundaries and stores it in the bin_pickle_2011 folder.
- **create_outputs.py**: This performs the ordinal encoding using the bin boundary pickles and hence generates the outputs/features for training models.
- **bin_pickle_2011:** This folders stores the pickle files of bin boundaries of each of the bands.
- **Outputs**: Running the create_outputs.py scripts stores the output features of that year in this folder.

▼ **Output_Features_and_Labels**

This folder contains all the outputs of all the years from 2001 to 2019 obtained from the creating_features directory. It also contains the ground truth for the year 2001 and 2011.

- **{year}_districts_quant.csv**: These files are the out features of that year obtained after performing feature engineering and quantile binning.
- **District - Ground Truth - 2011_2001.csv**: This is the input label file obtained from the census data of 2001 and 2011. It includes the feature columns of ASSETS, FC, CHH, BF, MSW, MSL for both years.

- **FEMP_LIT.csv**: This file contains ground truths of literacy rate and formal employment for the years 2001 and 2011.

▼ **CrossSectional_2001**

It contains the following two files:

- **CrossSectional_2001.ipynb**: It implements the models of SVC and Random forrest which are trained and tested on the year 2001.
- **xgboost_2001.py**: Implements XGClassifier that is trained and tested on the year 2001.

▼ **CrossSectional_2011**

It contains the following two files:

- **CrossSectional_2011.ipynb**: It implements the models of SVC and Random forrest which are trained and tested on the year 2011.
- **xgboost_2011.py**: Implements XGBClassifier that is trained and tested on the year 2011.

▼ **Temporal_Transfer**

This folder checks the accuracies direct forward and backward classifiers using XGBClassifier. It contains the following files.

- **xgboost_2001_to_2011.py**: Trains the model on the year 2001 and tests it on 2011 using census labels.
- **xgboost_2011_to_2001.py**: Trains the model on the year 2011 and tests it on 2001 using census labels.

▼ **TwoStepClassification**

This folder contains the main *Two_step_classification_model.ipynb* file. It takes the input as the feature vectors of the year from 2001 to 2019 and also the ground truths for the years 2001 & 2011. It implements the two-step forward classifier and predicts the labels for the year 2019 and also calculates the ADI.

▼ **Visualisation**

This folder plots the predicted labels of 2019 on maps to rank districts on the basis of their ADI. It contains the following files.

- **plotter.py**: This script is used to add color and features to the original 2011_Dist.geojson files so that when it is plotted on the map it shows the attributes and level of development of each village.
- **2011_Dist.geojson**: This is the modified file made by the plotter script.

▼ **Predictions_2019.csv**

This file contains the predictions of the labels for the year 2019 and also the ADI index for each district. It is generated by the *Two_step_classification_model.ipynb*.

Steps to reproduce the outputs and accuracies

Follow these steps to cover the complete pipeline to predict the labels for 2019 and obtain the accuracies of all the models

- Fork the github repository.
- Copy the **downloadDistrictTiff.js** script on the GEE console and run the process. On completion, it would add the tif file to your google drive. Estimated run time ~ 15-20 Minutes
- After downloading the tif file run the **split.py** program to make tiff files for individuals districts.
- After obtaining the image files for individual districts we will execute the **create_bins.py** file to make and store the bin boundaries in pickle files. The bin boundaries for all the bands will be stored in the bin_pickle_(year) folder.
- Finally, run the **create_features.py** to use the bin boundaries pickle and individual district image to generate ordinally encoded features. The output file will be stored in the output folder
- Note the code in the above steps has path of files and year of execution hardcoded. So you might need to change the parts and year according to you needs.
- At this stage the feature generation is complete and we will move on to training and testing ML models.

- Run the files **CrossSectinal_2001.ipynb** and **CrossSectinal_2001.ipynb** for the cross-sectional accuracies of 2001 and 2011 respectively. They would provide it using SVC and the Random forest model. For XGBoost run the **xgboost_2001.py** and **xgboost_2011.py** files
- For the forward and backward direct temporal analysis use the **xgboost_2001_to_2011.py** and **xgboost_2011_to_2001.py** scripts.
- Final execute the ***Two_step_classification_model.ipynb*** to generate the accuracies of two step classifier (both, basic and improved) and for generating the predictions for 2019.
- Use the **plotter.py** script to generate a modified geojson file so that it can be plotted on geojson.io.

Libraries And Software Used

The following libraries were used in the code base.

- Sklearn (KBinsDiscretizer, SVC, RandomForest,F1score, Train_test_split)
- Pandas
- Numpy
- Rasterio
- tifffile
- json
- XGBoost

To run the code it is advisable to use **Juypter Notebook** on your local machine as it would make it easier to navigate and join file paths. Alternatively, you can also use **Google Collab** Notebook. Apart from that, you can use geojson.io or Jsfiddle for visualizing the outputs.