**Model Description:**

* Import all the required modules or APIs into python namespace.
* Read the data file by giving the file path as input.
* Using OS module and python PANDAS, read the given csv file as demonstrated in the code. While, reading the csv files define the datatypes of all the variables in the dataset to avoid abnormal behavior afterwards.
* To look at the percentiles, missing values percentage, max and min values of numeric variables use .describe() and .info() methods of pandas data frame.
* Select all the categorical/factor variables by filtering with dtype=’O’, which means object in Python.
* Remove ‘Id’ from the analysis, as it is primary unique identifier and redundant in data analysis.
* Remove ‘monthly\_expenses’ variable from categorical columns list as it is actually a numeric column.
* Check the row-wise and column-wise missing value percentages to check if there exists any.
* Define a function to calculate the count of all the levels or categories in a given categorical variable and apply that function to all the categorical variables.
* To analyze the correlation between categorical variables and target variable loan\_amount, I used **ANOVA** test and based on **F-statistic** and **p-values**, variables are shortlisted.
* Variables having F-statistic greater than 20 and p-value less than 0.05 are chosen.
* Next step is removing outlier data points in the dataset – age.
* Removing the data points where there is no primary\_business and loan\_purpose, as there is a strong relationship between these variables and target variable loan\_amount from the above test.
* Also removing the data points where there is no annual\_income, that is where annual\_income is 0. Because, credit score/reliability of a loan depends on financial support of a citizen and also there is huge percentage of NULL’s in those records.
* Form a correlation matrix of numeric variables selected by dtype=’int32’ and ‘int64’ with target variable loan\_amount.
* **Weak correlation values are observed for all the numeric variables, so we keep all the variables.**
* Next we check the data points where water\_availabity and sanitary\_availability are both NULL. We observed that there is again a huge percentage of NULL’s in other variables in those records. So we remove those data points as well.
* Observed an outlier in occupants\_count variable which is removed from the analysis.
* Next, we start converting the categorical variables to numeric variables by imputing the levels with their corresponding frequency values divided by total number of levels of that categorical variable column – As this is a Regression problem by default, only numeric variables are supported.
* prepare\_data() function prepares the data for algorithm to be applied by concatenating numeric variables, converted categorical variables as input.
* Run\_models() function implements 3 algorithms in total for this model input and loan\_amount as target variable – **Gradient Boosting, Adaptive Boosting and Linear regression.**
* Training sample size is tested with 70, 90 and test sample size with 30, 10.
* Check out the **results()** function for all the model evaluation metrics and results.
* Apart from feature\_importances\_ from sklearn, we also considered numeric and factor variables separately and ran the models again in the hope of finding important predictors – Results are matching with results before (a few of them)
* From the observations that the model results are not quite good, we adopted another method of dealing with categorical variables.
* As loan\_purpose and primary\_business have far too many levels (>70), we opted to check the frequency values of all the levels of the variables. Generally, very less frequent levels are accommodated into one single encoded level for easier analysis in EDA.
* **Post that check, we took each categorical variable and for each categorical variable, for every level – we imputed the level with corresponding mean value of the response (target variable – loan\_amount). For this purpose, we wrote two functions 1. Mean\_response\_impute and 2. Target\_encoding.**
* Finally after running the above functions we have our once again converted data of categorical variables ready and is concatenated with numerical data like before using pre –defined prepare\_data() function.
* Run the models again on newly processed data and results are captured on console screen which in turn are incorporated in **final\_results()** function.
* Important observations are duly noted at the bottom of the code.
* **Additionally, the code is appropriately commented wherever necessary, explaining all the complexities and bottlenecks.**