Assignment:

1) Make a function that takes as input the data without NULLs and other parameters as follows:
   a. List of numerical features
   b. List of ordinal categorical features
   c. List of Nominal categorical features
   d. Clamping-outliers threshold (1.5 or 2…etc.)
   e. Correlation-threshold for dropping one feature of 2 highly correlated feature
   f. Dropping-column threshold (if N% of the column has the same value, then drop the column; make N a hyper-parameter to be set by the user)
   g. Alpha of Shapiro-Wilk test
   h. Training dataframe (without NULLs) as input

   And the function automatically makes all of this:
   a) Drops any column which has 80% of values with the same value (for all variables)
   b) Drops one of two highly correlated features (the one dropped is the one which is less correlated to the response variable) (for all numerical variables)
   c) Clamps the outliers (for all numerical variables)
   d) Log-transforms any highly skewed variable (for all numerical variables)
   e) Performs the Shapiro-Wilk test with alpha = 0.01 to decide which variables follow gaussian distribution and which of them do not follow a gaussian distribution (for all numerical variables)
   f) Min-max scale the non-gaussian features, and standardize the gaussian features (for all numerical variables)
   g) OHE the nominal features, and label encodes the ordinal features (for categorical variables)
   h) Merges the pre-processed categorical and numerical data in one dataframe and returns it to the user
2) Implement the Linear-regression using the SVD and pseudo-inverse **but with bias term** (the y-intercept term) **by adding a column of ones in the data matrix X,** then compare the predictions with the Sklearn linear-regression model (with y-intercept term)
3) Implement the mini-batch gradient descent from scratch and you will be asked how your code works (some students will be picked to explain parts from their code, so please don't copy from the internet without understanding); then compare it with the sklearn.linear_model.SGDRegressor output with the same learning rate.