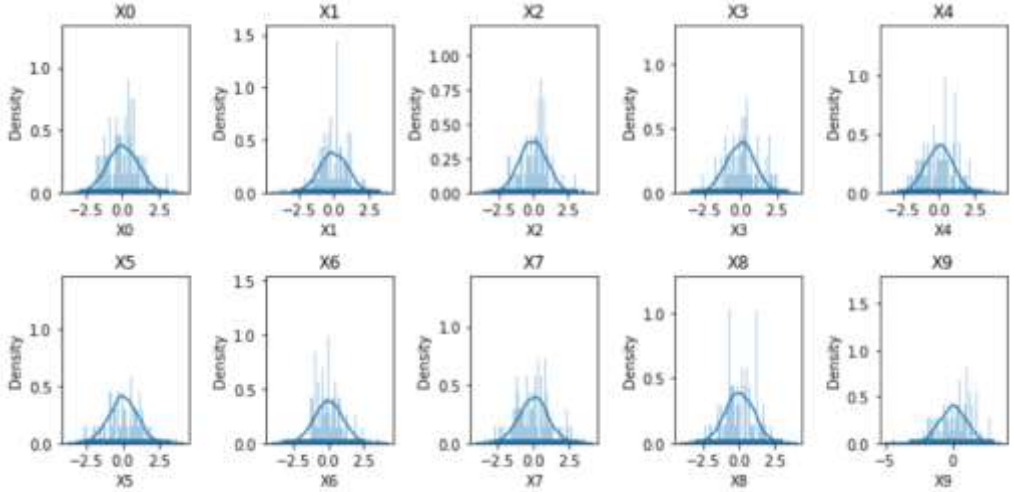
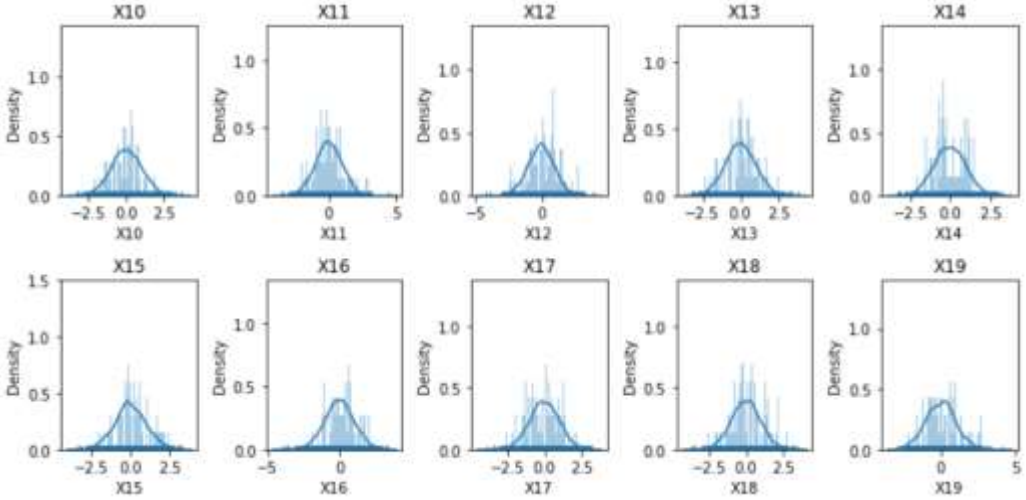
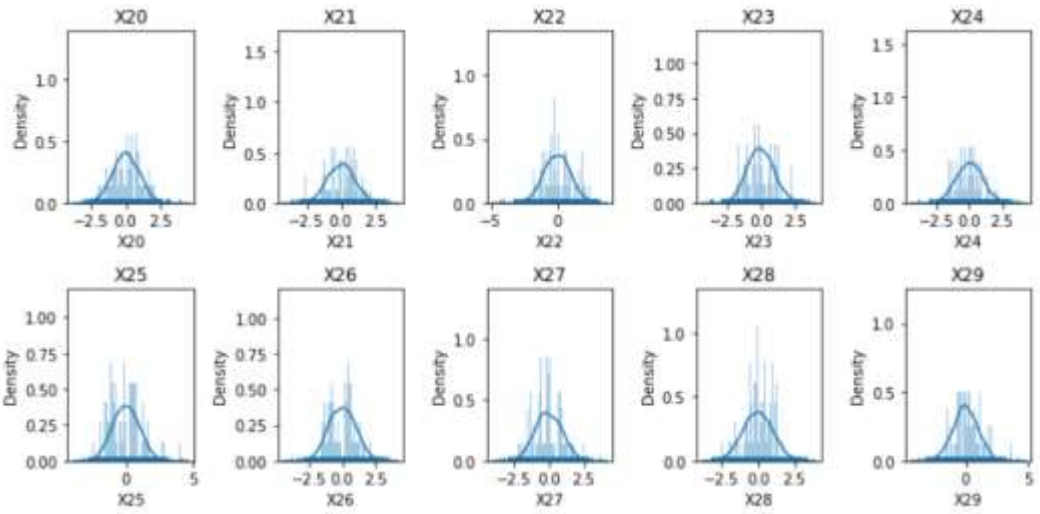
1. Explore the data using visualizations and statistics
    - What are the feature distributions (compute histograms for them)?
      The reason why do we need to check the feature distribution is that many ML models are based on the normal distribution assumption. We need to know what type of distributions we are dealing with. After we draw all features distributions, we can see all features are normal distributed which is pretty good.
        - Columns [0:9] distributions, I set 3000 bins



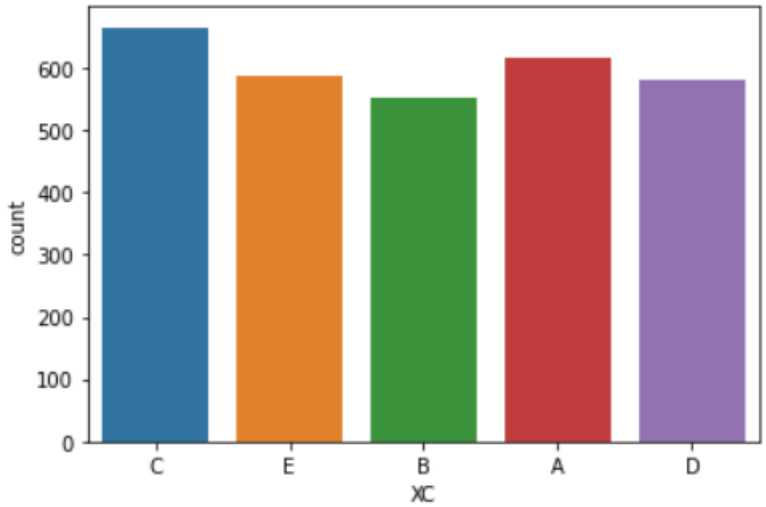
        - Columns [10:19] distributions, I set 3000 bins



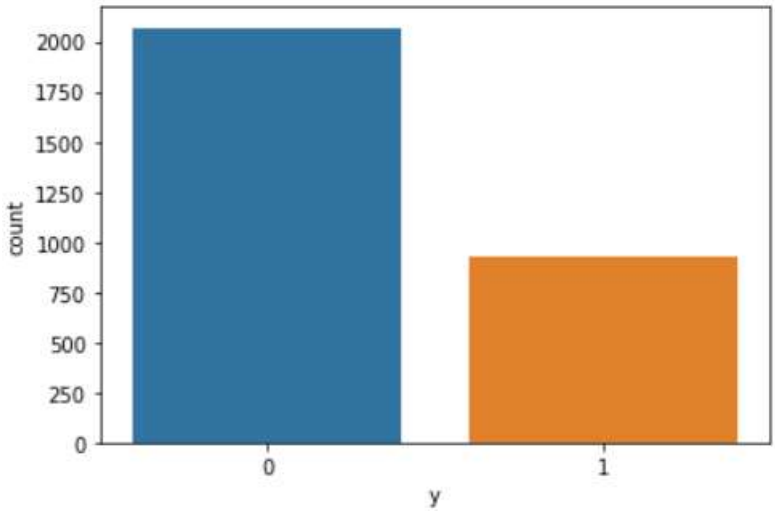
        - Columns [20:29] distributions, I set 3000 bins



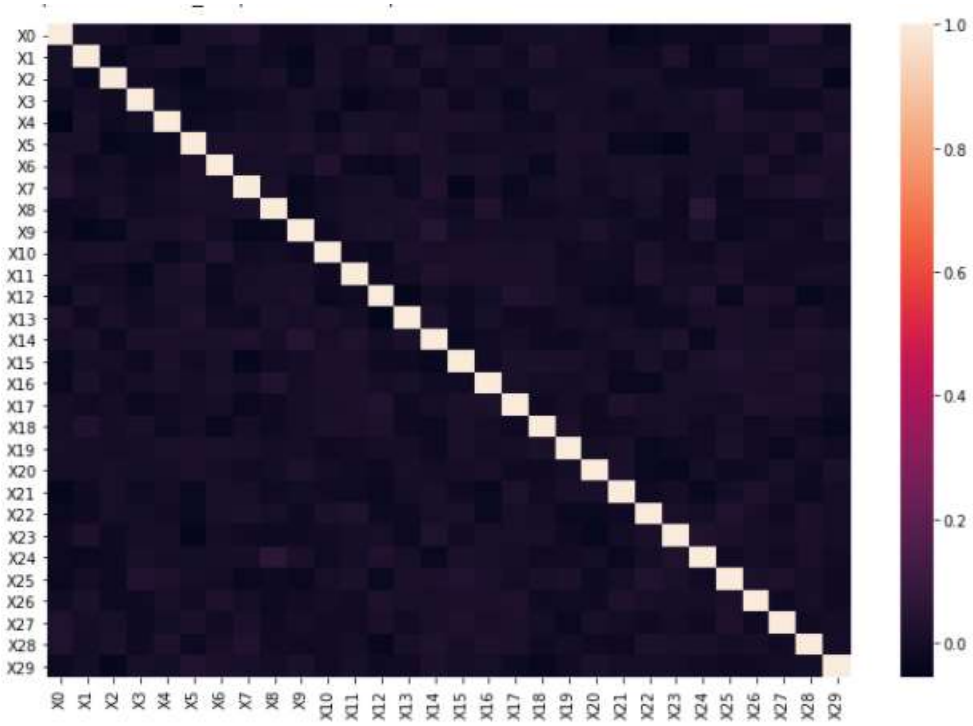
- Distribution of the last column that is the categorical data



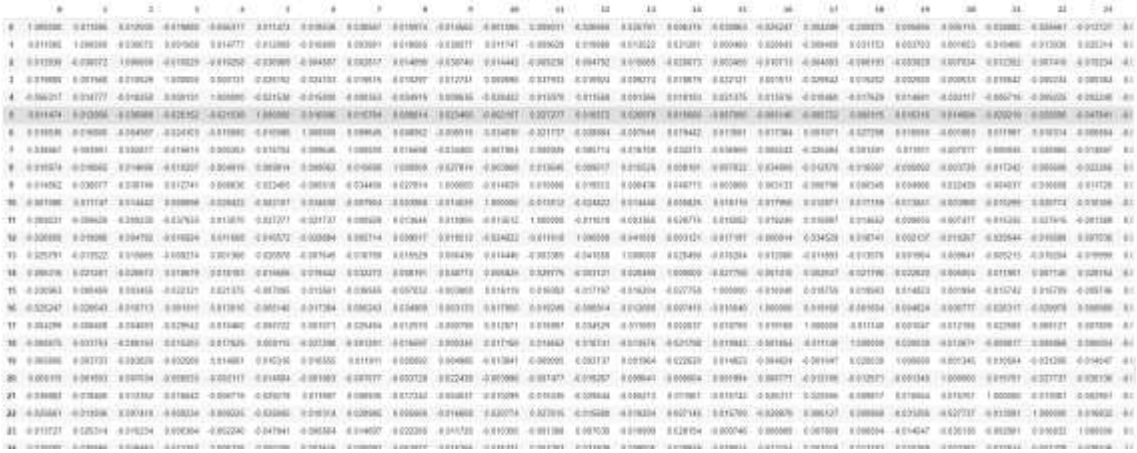
- I think we also need to check the sample distribution. It is quite important too because it affects our predict result. If the data is extremely imbalance, the result will be very bad. Sometimes, we need to think about how to balance the data.



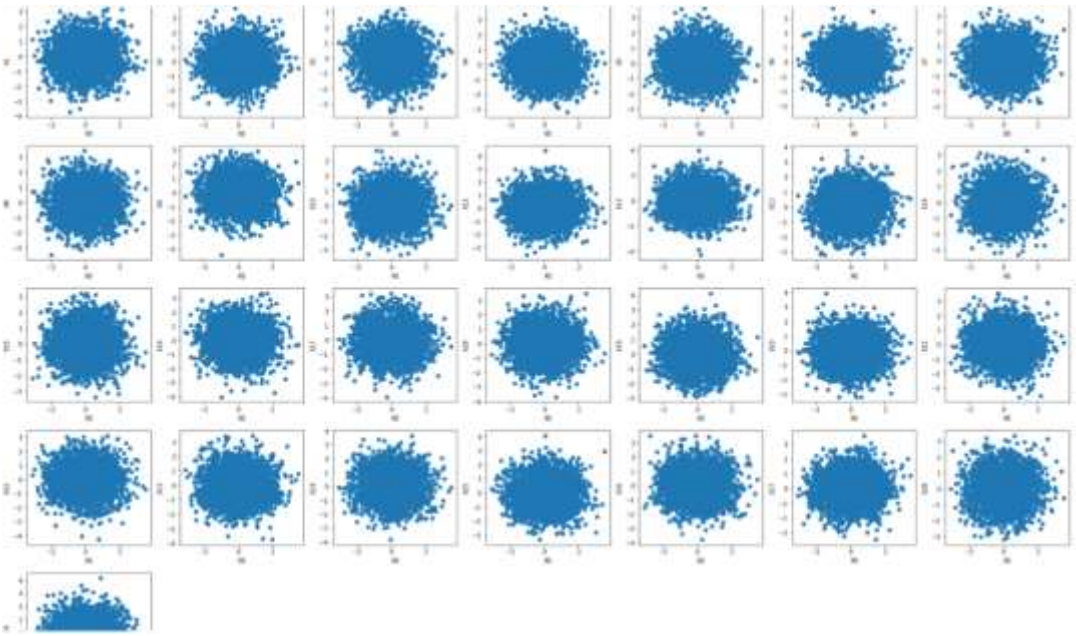
- Is the data correlated?
    - Check Pearson correlation between columns.
      I drew a heat map of every feature to check if they are correlated. We can notice that all features are not correlated at all.



      I also print out their Pearson coefficients

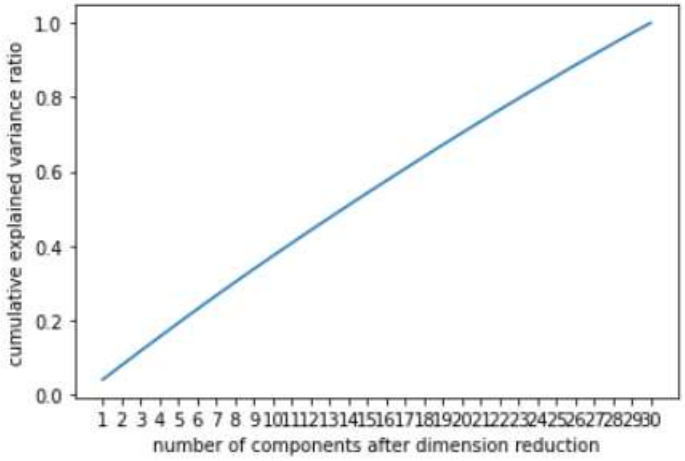


    - Try looking at the scatterplot of the dimensions
      For this part, it has a lot of plots, so here, I only take part of the plots. The graph shown here checks the correlation of first column and the other columns. We can combine the heat map and this graph, we can draw a conclusion that features are not correlated.

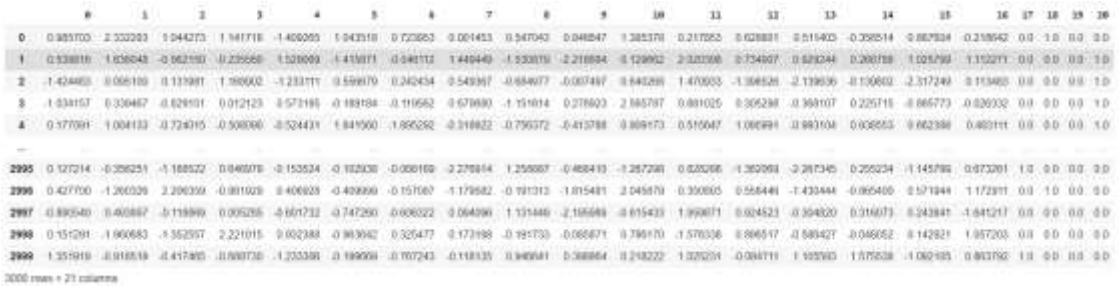


In notebook, we can also run all columns to check any two features are correlated or not.

- Can you remove unnecessary features?
    - Alternatively, can you reduce the dimensionality of the data with PCA? We can check the graph of cumulative explained variance of ratio. We can see that there is no elbow point. So, it is unnecessary to use PCA to reduce the dimensionality but we can check the other method such as lasso or ridge.



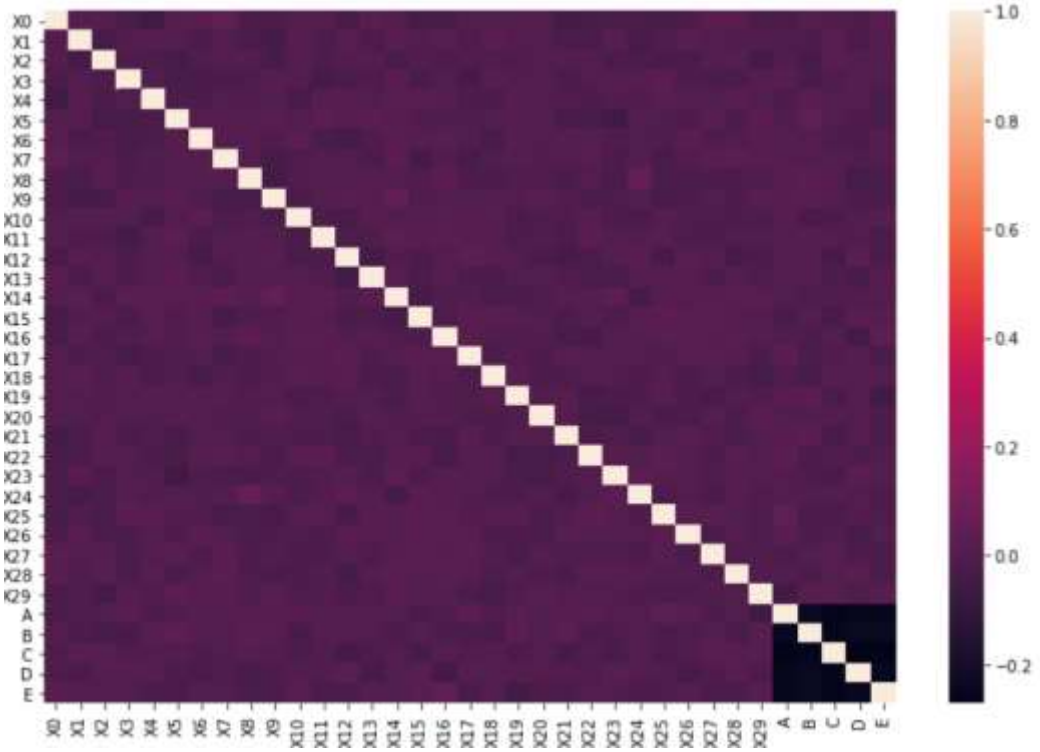
    - When I tried L1 norm that is feature selection, it can actually reduce the dimension. It reduces to only 20 dimensions.



2. Preprocess the data
    - One-hot encode categorical variable

	A	B	C	D	E
0	0.0	0.0	1.0	0.0	0.0
1	0.0	0.0	0.0	0.0	1.0
2	0.0	0.0	0.0	0.0	1.0
3	0.0	0.0	0.0	0.0	1.0
4	0.0	0.0	0.0	0.0	1.0

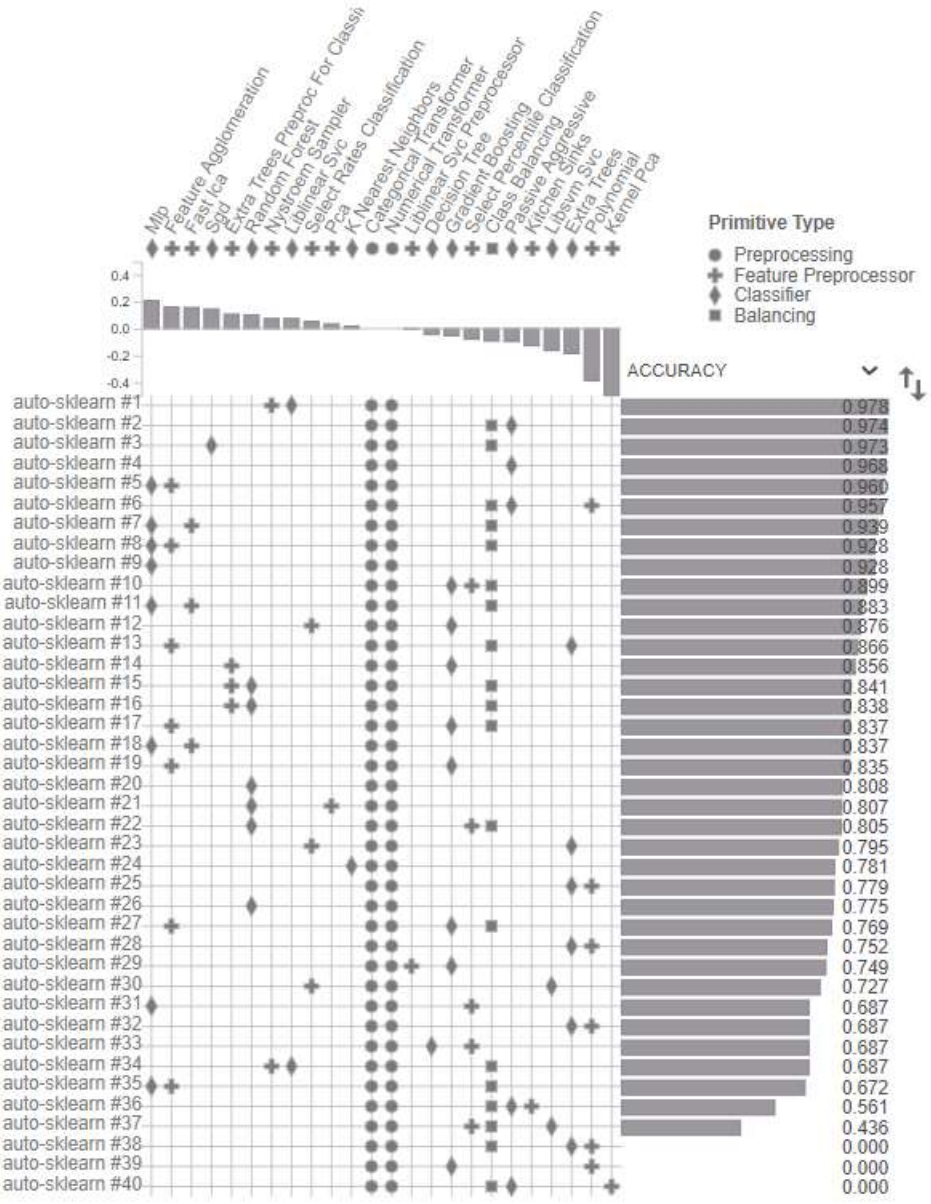
After I added categorical data, we can check the heat map again to see if any features are correlated or not. We can see that they are not correlated.

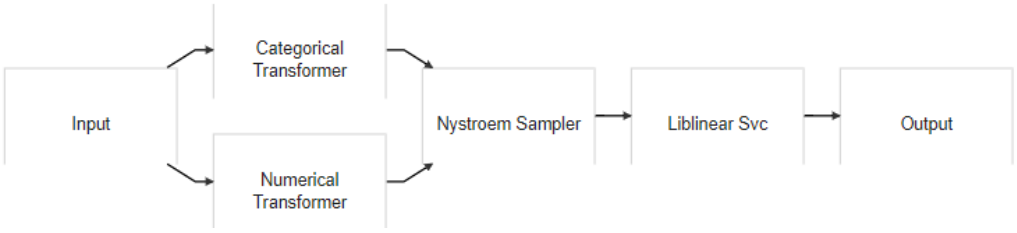


- Check if you need to normalize the data
  We can normalize the data to the same range

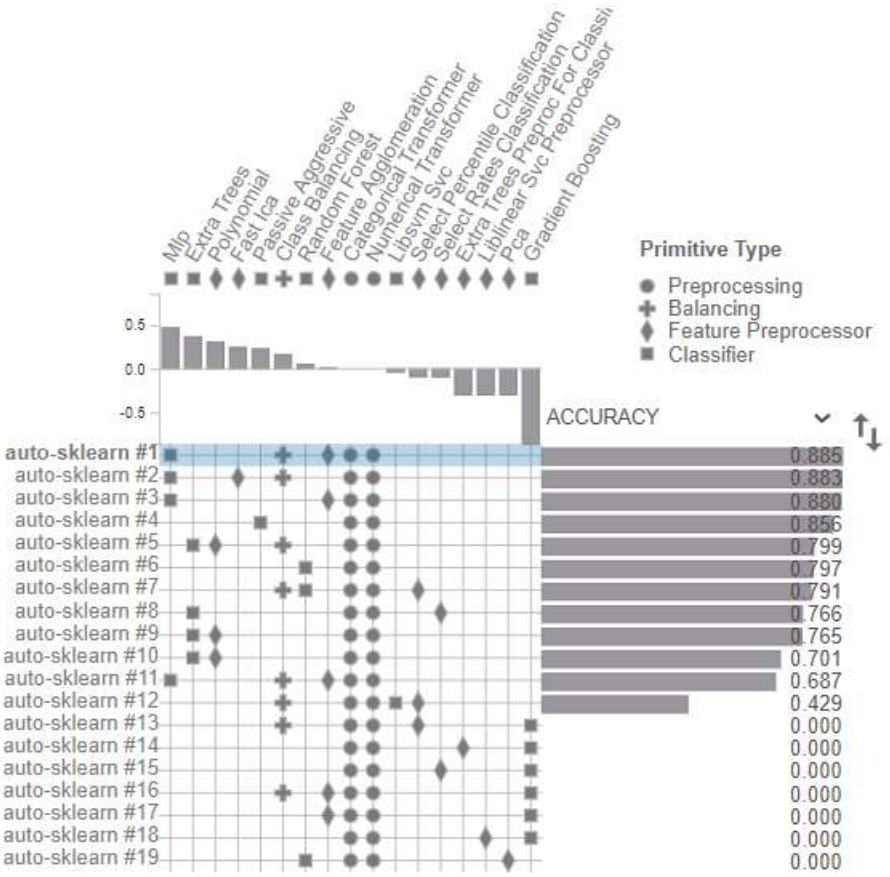
3. Solve the classification problem using Auto-Sklearn. Try different dataset.
    - With / without categorical feature

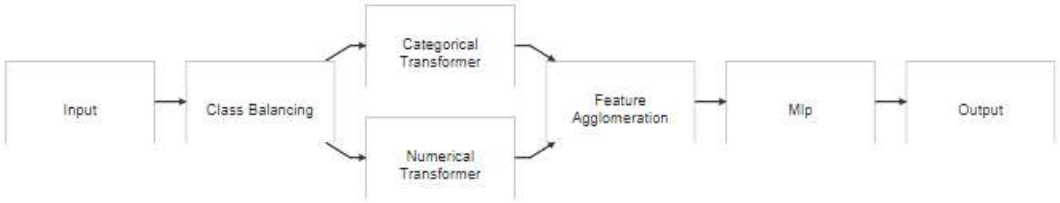
- With categorical data: accuracy is 0.98111112



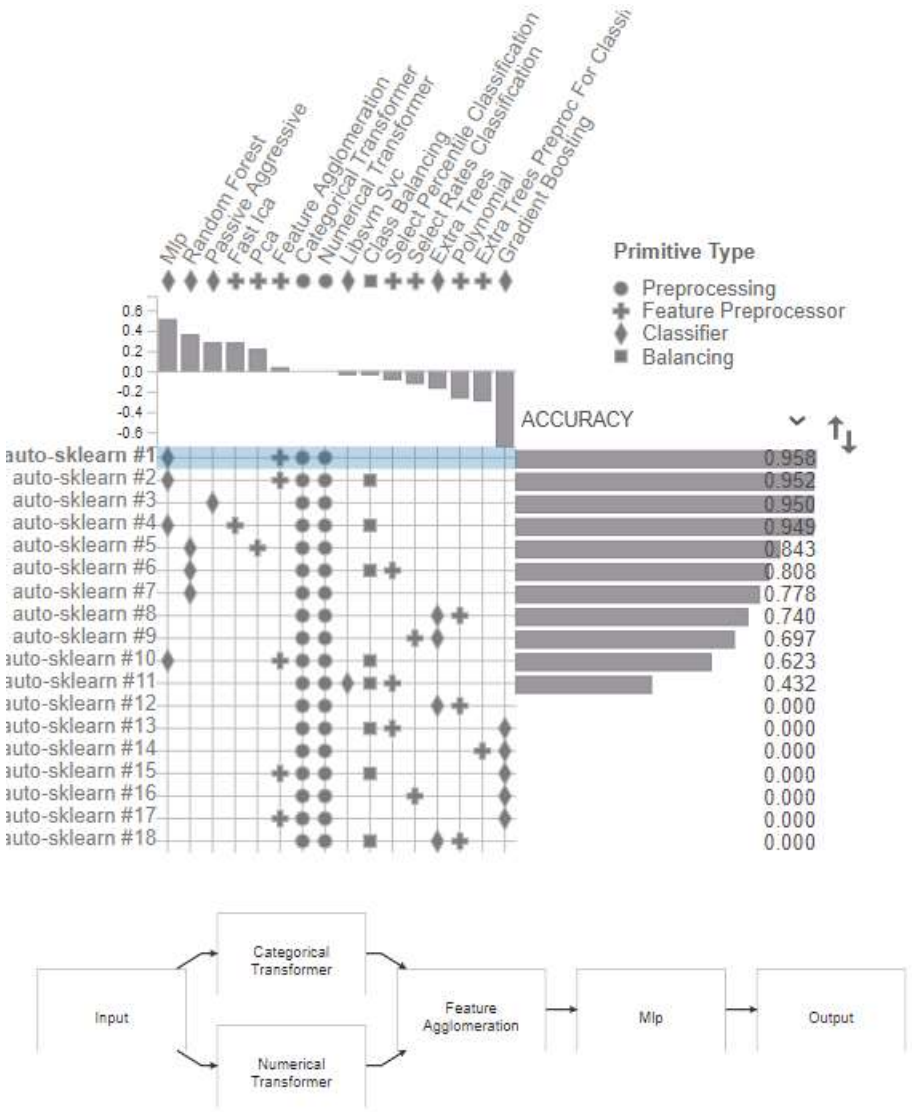


- Without categorical data: accuracy is 0.884444444444445. It is a very bad result without categorical data.



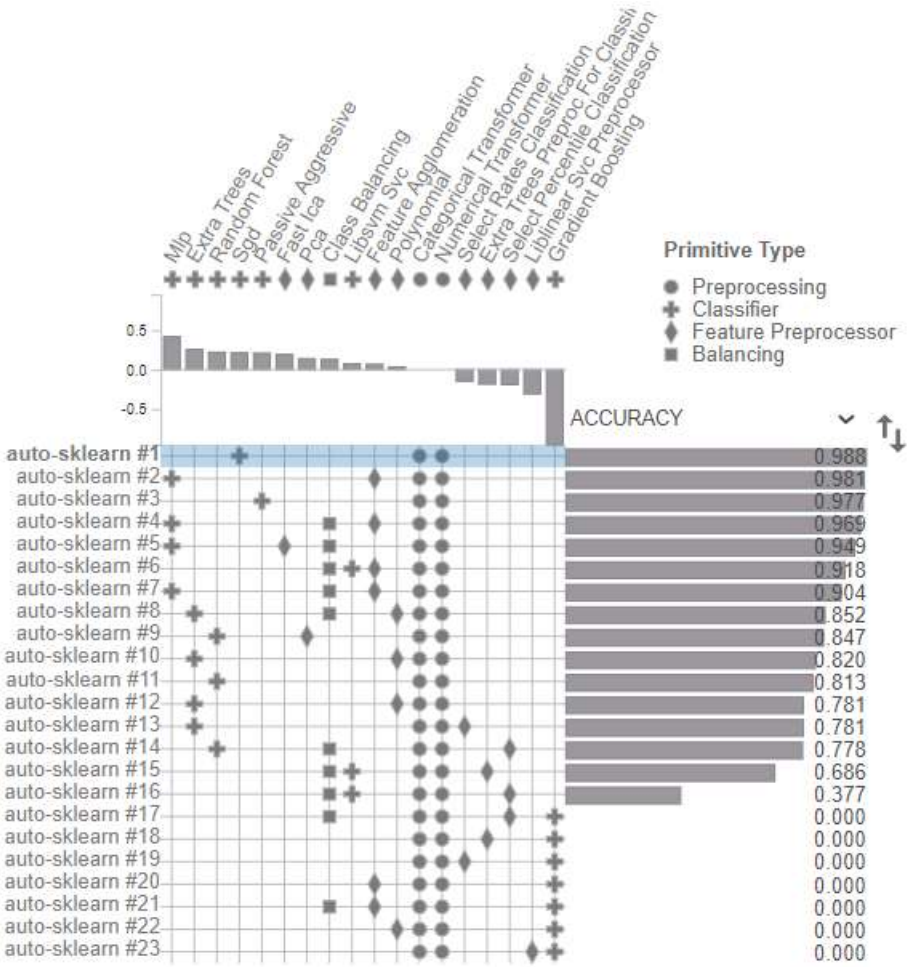


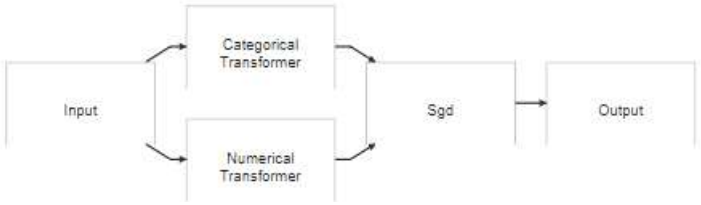
- With / without normalization
  - With normalization: 0.96444444444



  - Without normalization: this dataset is same as the data with categorical data because that data is not normalized either.

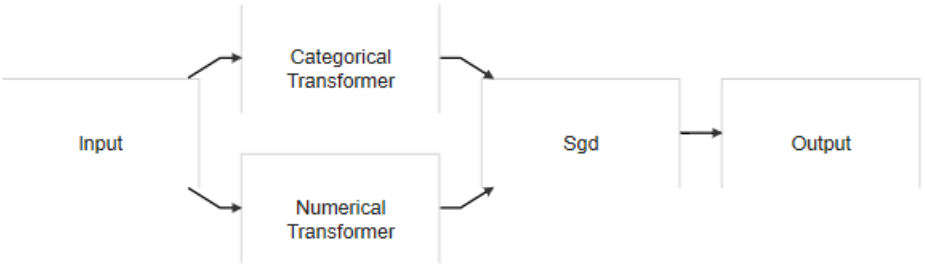
- With / without feature selection / PCA
  - With feature selection: accuracy is 0.98888888888889





- Without feature selection: this dataset is same as the data with categorical data because this data did not do the feature selection.

4. Explore the models using PipelineProfiler. What primitives perform well for this task?
   Primitive name: auto_sklearn.primitives.classifier.sgd.
   Finally, we got the highest accuracy that is 0.9889 with feature selection data. It uses categorical transformer and Numerical transformer to deal with categorical data and numerical data respectively. Then uses the SGD classifier to classify the data. The graph below shows the process of dealing with data.



5. Select the best model out of your experiments (split the data in a 70/30 Training / Validation set)
   In autlML training, we split the data 70/30 training and validation set.
   Finally, we will choose the SGD model to classify the data with parameters:
   Primitive Name: auto_sklearn.primitives.classifier.sgd
   {
   alpha:0.00023,
   Average:True,
   Fit_intercept:True,
   Learning_rate:optimal,
   Loss:log,
   Penalty:l1,
   Tol: 0.00001
   }

   To deal with the categorical data:
   Primitive name: auto_sklearn.primitives.data_preprocessing.categorical_transformer with paramters:
   {
   Categorical_encding: one_hot_encoding,
   Categorical_coalescence: no_coalescense
   }

   To deal with the numerical data:
   Primitive Name: auto_sklearn.primitives.data_preprocessing.numerical_transformer with parameters:
   {
   Imputation_strategy: mean,
   Rescaling: standarize
   }