**Report**

**Final Project**

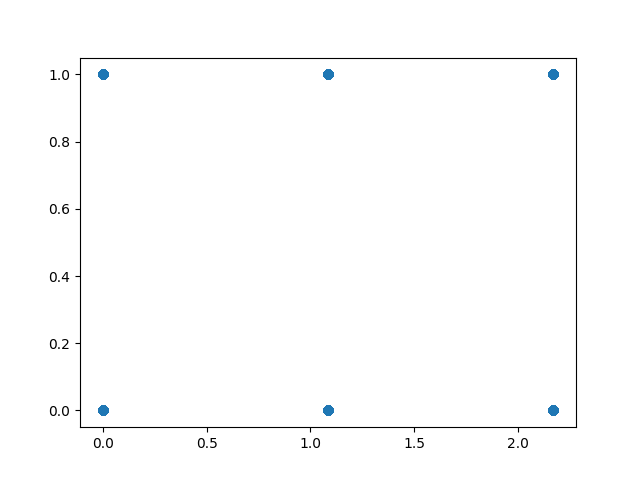
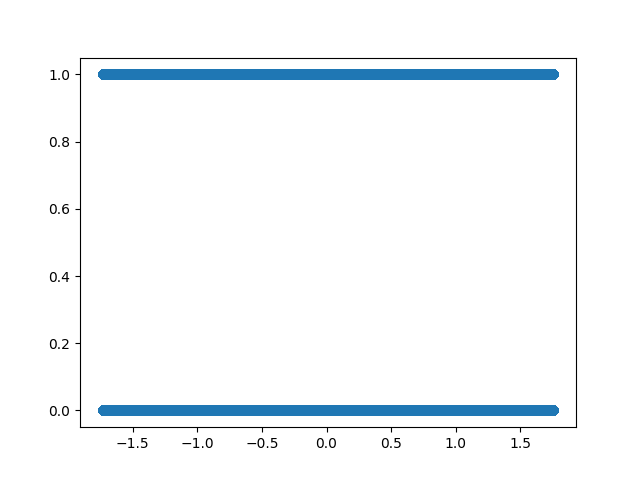
**Machine Learning**

* **Pre-Processing:**

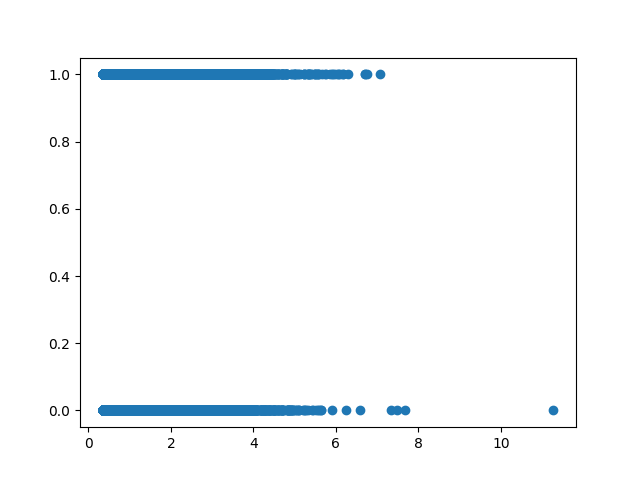
Feat-2, Feat-8, Feat-17, Feat-21 with Target column of train.csv were poorly correlated, so the respective columns were dropped from the train dataset. As we have an excessive number of features and limited computational resources, removing low correlated columns might help in reducing the dimensionality and speeding up training without significant loss in predictive performance. As a result, it increased the model accuracy. Also, there were no categorical entries and missing values in the dataset, after inspecting from train\_df.info (), there was no need for Simple Imputer or One-Hot Encoding.

* **Insights:**

All the Features except Feat-9, Feat-14, Feat-18 and Feat-22 show continuous behavior, i.e., the all the values present in the particular column had output/Target as 0 or 1 in the Target column. But above-mentioned features show different behavior as shown in figure 2.



It can also be seen very clearly that some columns contain values which differ in sign, there value more or less equal, like [-4,4] or [-1.5,1.5] roughly. However, some features contain values from 0 to 8/9/10 or very few values for more that 10 (shown below).



* **Model chosen for the Project**: Gradient Boosting Classifier, these algorithms handle large datasets effectively due to their ability to handle high dimensionality and large numbers of samples. They have implementations that are optimized for speed and memory usage. Also, as the dataset provided in the project is extremely large, there won’t be a problem of over-fitting of data when we use Gradient Boosting Classifier.
* It builds a strong predictive model by combining multiple weak learners, typically decision trees, in a sequential manner. n\_estimators: The number of boosting rounds or trees. learning\_ rate: Step size for each iteration's update. max\_depth: Maximum depth of the individual trees.

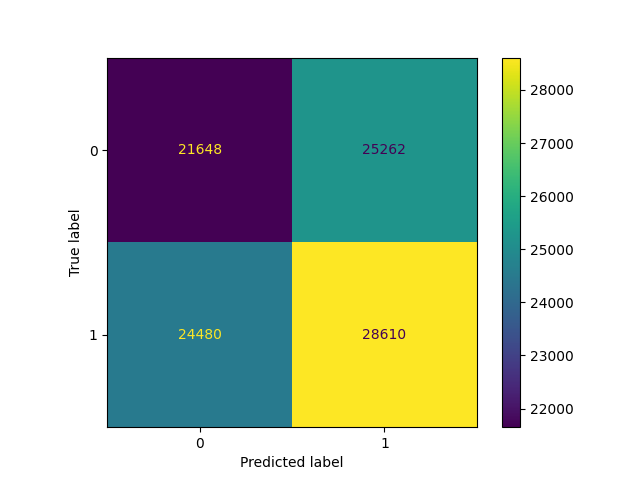
These parameters take quite an amount of time while computation, but in the model, features were reduced and model was tuned with varying the values of parameters for an efficient performance of the model.

A smaller learning rate allows the model to learn slowly and make smaller adjustments, preventing drastic changes that might overfit the training data.

Adding more estimators helps the model learn more complex patterns and relationships in the data. It reduces bias and allows the model to fit the training data better.

* **For Evaluation metrices**, the given problem is a classification problem so Confusion Matrix has been used to calculate number of correctly predicted and incorrectly predicted Target Values.

Confusion Matrix Display is being used to make the matrix more intuitive and insightful.



* **Future Approaches:**

Grid Search CV works better on lower size dataset however we can choose anyone of them,

Randomized Search CV tries to maximize the entropy, the more random selection is there, the more of getting better results. This method is especially useful when the hyperparameter search space is vast and trying out all combinations is computationally expensive.

This model will automatically give the best possible combinations of n\_estimators, max\_depth, learning rate etc.