1. Merge datasets
2. Column-wise data variable reduction (part1)
   1. Drop the columns with 85% or more null value

The transaction table has more cases of missing data. 55 of the 394 features have more than 80% missing data, and 113 features have missing data between 70% and 80%. Similar missing data patterns were found among features which have consecutive names. Specifically, the missing data rates of "D6"-"D9" and "D12"-"D14" were all above 87.3122%. The missing data rate of "D6" - "D9" and "D12" - "D14" was above 87.3122%. The missing data rate for "V138" - "V166" is between 86.1227% and 86.1237%. All features from "V323" to "V339" have missing data rate of 86.054967%. This regularity exhibited in the missing data suggests that there may have a strong correlation between these consecutive numerically arranged features, although the data provider does not explain the specific meaning expressed by these encrypted features.

* 1. Remove columns with excessive single value

We remove the features in which more than 90% samples have the same value.

1. Data cleaning
   1. Make feature names uniform in train and test datasets

We found that the feature name starting with ‘id’ is not uniform in train and test datasets. In train data set, we have 'id\_01', 'id\_02', … 'id\_38'. However, in test dataset, we have 'id-01', 'id-02', … 'id-38'.

* 1. clean up the features based on their email domain (“P\_emaildomain” and “R\_emaildomain”)

we need to clean up the features based on their email domain (“P\_emaildomain” and “R\_emaildomain”). Because some of the email domain will separate from the different countries. For example, “yahoo.co.jp”, “yahoo.de”, “yahoo.fr”. What we need here is just the information of email provider. So, all three records above will be converted to “yahoo”.

* 1. clean up the features “DeviceInfo”

“DeviceInfo” record different versions of devices, such as Windows or IOS. We can find information like “Windows 10”, “Windows 7”, “iso 11.2.1”, “iso 11.1.2”, “iso 11.4.1”. However, the versions don’t split different users. What we needed is just the base of the operation system. So, we transfer different versions of device’s record in to one record.

On the other hand, we found that the train and test transaction dates don't overlap, so it would be prudent to use time-based split for validation. Since some of “Diviceinfo” could be for old devices and may be absent from test data, we mark this kind of “Diviceinfo” as “old\_device”.

Chart, histogram

Description automatically generated

* 1. clean up the features “id\_30”

Among the 75 devices recorded in “id\_30”, we can find information like “Windows 10”, “Windows 7”, “iso 11.2.1”, “iso 11.1.2”, “iso 11.4.1”. However, the versions don’t split different users. What we needed is just the base of the operation system. What we can do for data processing is to transfer different versions of device’s record in to one record.

* 1. clean up the features “id\_31”

The Category feature “id\_31” records different versions of browsers, we need to do the same processing to make the record much cleaner. For example, the record “mobile safari 11.0” and “mobile safari generic” should be recorded as “safari” after processing (another feature “DeviceType” has recorded whether the brower is from mobile).

* 1. Transfer “TransactionDT”

The variabele “TransactionDT” in the Transaction dataset is a time delta from a given reference datetime (not an actual timestamp). The minimum value of “TransactionDT” is 86400 which corresponds to the number of seconds in a day (606024 = 86400). Therefore, it is possible the unit of this variable is seconds. The maximum value of “TransactionDT” is 15811131. It is reasonable to conclude that the data spans 6 months (15811131-86400)/86400/30 = 6.066.

To better extract the information in “TransactionDT”, the processing we do includes

* Transfer the unit of 'TransactionDT' from seconds to days
* Add a new column 'Trans\_DayOfWeek' transfer the TransactionDT to the day of week

We hope to explore whether the fraud behavior, in different days of a week, has a tendency or regularity.

* Add a new column 'Trans\_hours' transfer the TransactionDT to the hours of day

We hope to explore whether the fraud behavior, in different hours of a day, has a tendency or regularity.

1. Tackle Class imbalance

Chart, bar chart

Description automatically generated

To reduce the computational stress, we use the strategy of Undersampling.

1. Missing value imputation K-Nearest Neighbors imputation method
   1. Transform categorical features by Label Encoding

One thing to note here is that the KNN Imputer does not recognize text data values. It will generate errors if we do not change these values to numerical values.

* 1. Normalize data by Scikit-Learn’s MinMaxScaler

Another critical point here is that the KNN Imptuer is a distance-based imputation method, and it requires us to normalize our data. Otherwise, the different scales of our data will lead the KNN Imputer to generate biased replacements for the missing values.

* 1. KNN Imputation

we are setting the parameter ‘n\_neighbors’ as 5. So,the missing values will be replaced by the mean value of 5 nearest neighbors measured by Euclidean distance.

1. Column-wise data variable reduction (part2)
   1. Delete the features similar in fraud and non-fraud by violin plot
   2. Delete the features with high correlation (>0.8)
   3. Recursive feature elimination with cross validation and random forest classification

Basically, it uses one of the classification methods (random forest in our example), assign weights to each of features. In this method, we will not only find best features but we also find how many features do we need for best accuracy.

**At this point, we have retained 46 of the best performing features.**

1. Prepare data for modeling
   1. Merge raw datasets
   2. Merge datasets
   3. Select features based on the results of preprocessing
   4. Clean up the features based on their email domain (“P\_emaildomain” and “R\_emaildomain”)
   5. Using Categorical Data with One Hot Encoding

* Common strategies for transforming categorical features include feature aggregation, graph-based transformation, or deep-learning approaches such as feature embeddings.
* one-hot encoding can be applied to nominal variables, in order to improve the performance of the algorithm.
  1. Split the train dataset into train and validation set
  2. Tackling Class imbalance with undersample majority
  3. Normalize the continuous features to make sure they're on the same scale
  4. Missing value imputation (SimpleImputer with mean)
  5. **Decision Tree**

Graphical user interface, text, application

Description automatically generated

* 1. **Random forest with grid\_search**

A picture containing text

Description automatically generated

This result is perhaps overfitting.

Another option:

Text

Description automatically generated with medium confidence