1) Patterns and anomalies (in order of the notebook scripts):
- one of the samples contained ambiguous data about the real label, so it was deleted, and the final amount of samples is 3922.
- the final label distribution is Clean – 86%, Exploit – 9% and Financial Scam – 5%
- about 24% of the data was verified by a human expert, particularly almost all Scams, 2/3 of Exploits and a 1/5th of Clean samples.
- almost all addresses are encountered only once, but some of them – twice and 3 of them are labeled as Exploit twice, so the first step of the final algorithm would label activity of these accounts as Exploit.
- There are different 7 different Chain IDs and 5 different data sources with different proportion of labels in it, so we need to make sure to divide data for train and test set such that they are represented equally.
- Also regarding these columns we may notice that label Exploit is coming from data source "Public Historical data", or that about half of the label Scam is coming from the chain 56 or\ and data source "Manual Validation". So if we can increase mining these resources, we can improve class balance of our dataset.
- There are 20 broken bytecodes (0x), but I don't see it as a problem since their opcodes seem ok.
- OPCODE: there are 58k+ unique code words in all dataset, if we filter the occurrence to quarter of the dataset: 980 occurrences only 139 code words left. EDA didn't reveal any significant pattern of class separation, so we need to use complex models.

2) Training data preprocessing:
I would use ether
- one hot encoding of the 139 code words + category of 'other'. So every code word is represented as a vector of 0s and one 1 of the size 140. Or even frequency of 300 code words, which gives vector of size 220.
- or train a word2vec model using the corpus of all the code samples we have ONLY IF we need to reduce the size of word vectors, but I think OneHot is better, and the size shouldn't be an issue here.

3) Models and approaches:
- I would divide the data as 85% train, 10% test and 5% validation sets – proportionally over data source, chain id and of course labels.
- I would train an RNN, specifically – LSTM or Attention based RNN
- I would start with a binary classification of Clean over Malicious, and go on to separate Malicious class into Exploit and Scam if the classification model is good, otherwise I try 3 classes at once
- I would construct a multi-label loss function with weights for each class, for example $L=-1/w_i \sum y_i \log(p_i)$ over $i=1,2,3$ where w_i is proportion of this class, y_i – real label and p_i – predicted probability of this class.

4) Justification:
It seems that opcode is a predictive feature for this task and since it is a code I want to use a model that captures importance of sequence. I want to use weighted loss because the classes are severely imbalanced and I don't think that removing samples from a small dataset is a good idea, or augment data is a good idea since fraudster find their way to deceive a system, and we want to identify the real pattern of malicious activity.