**Phase 3: Project Design Phase**

**Topic- Proposed Solution and Solution Architecture**

**1. Feeding the input data**

Every AI or ML system needs data to get started. In this scenario, it will be transaction data such as:

* transaction value
* product SKU
* type of credit card
* etc.

But we’ll also add data relating to how the customers connect to the site:

* IP data
* device type
* VPN, proxy or Tor usage
* etc.

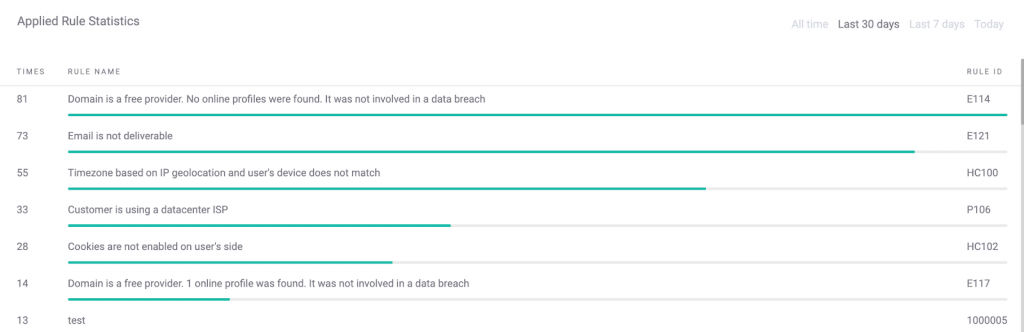
Note that the more data you have to start with, the more accurate your results will be. This is particularly important if your fraud prevention software does not allow custom fields, as you could be missing out on crucial information.

**2. Generating the rules**

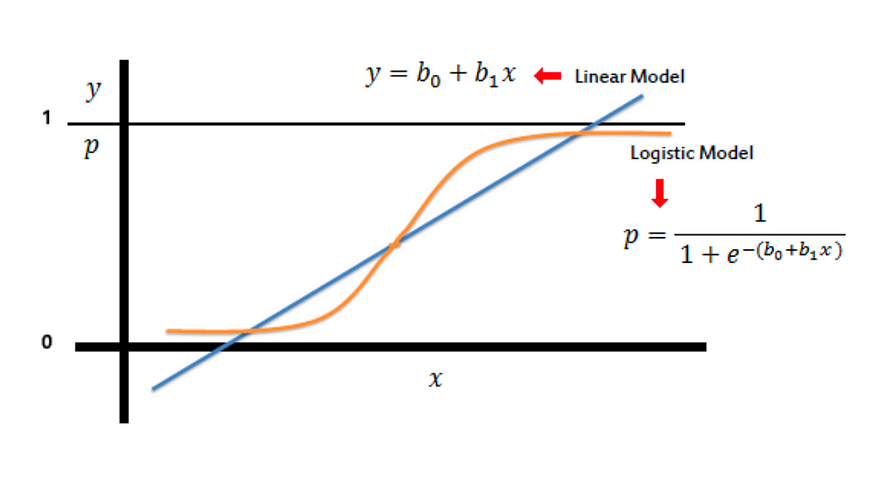
SEON’s machine learning can generate two main types of rules:

* **Single parameter rules,** also known as heuristic rules: an example of a single parameter rule would be: block if the IP is X.
* **Complex rules:**including multiple parameters.

Each listed rule shows an accuracy score. You can adjust accuracy thresholds to tighten or loosen triggering conditions.



Note that the rule names are extremely descriptive, allowing you to understand why it was generated at a glance. You can clearly see how all the rules are designed to understand how the customer logged in could affect the transaction value lost to fraud.



**3. Reviewing and activating the rule**

SEON allows you to filter the rules by any data point, including its type and predicted accuracy. The accuracy part is particularly useful, and it is calculated using a complex confusion matrix.

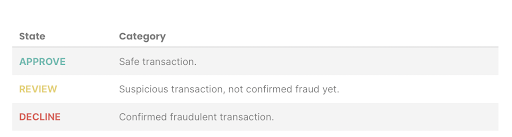
By default, machine-learning suggestions are switched off. You can quickly enable them using the ON/OFF switch. It’s also possible to manually create and adjust thresholds for the rule to be triggered.

**4. Training the algorithm**

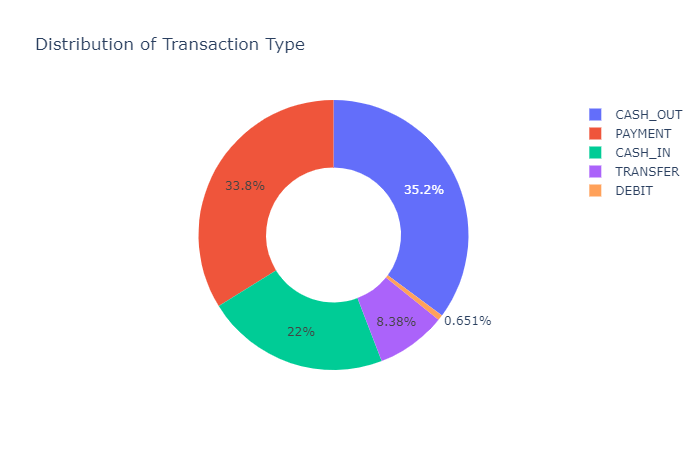
Providing feedback data is the key to refining rules and getting better accuracy. With SEON, there are two ways to provide feedback and label the actions:

* **Via the GUI:**a simple, visually-friendly way to mark actions
* **Using the Label API:**you can mark actions programmatically via API calls

However you do it, the actions should be marked as either APPROVED, REVIEWED, or DECLINED.



The algorithms retrain themselves every day based on the last 180 days worth of data. You can access them at any time in your backend and scoring engine (where you manage all the risk rules).

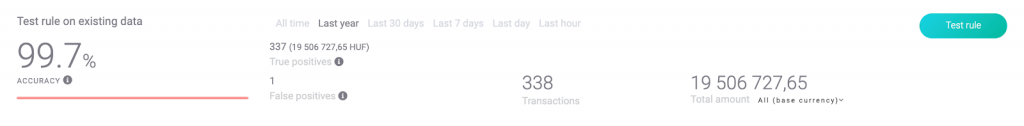


**5. Testing rules on historical data**

Good fraud prevention software should allow you to revisit past cases to see if the rules would have helped. This is done in a sandbox environment, where you can switch the rules on and off and witness their accuracy in person.

Running a test will create a confusion matrix based on previous transactions over the selected time frame and highlights the estimated accuracy rate of the rule.

In the field of machine learning a confusion matrix, or error matrix, is a table layout that allows visualization of the performance of an algorithm. This allows you to calculate accuracy over a specific date range – selectable from the last hour through to the last year.



In the right hands, this gives fraud managers complete control over their risk strategy, allowing them not only to reduce but also to monitor, test, and tweak results at will.

Due to huge amount of data models for Support Vector Machine and Random Forest were unable to compile, even on Google Collab. Further work can be done by under sampling of data by 50:50, that would reduce data size even more and as a result SVM and Random Forest results can be compiled accurately.