Fraud detection with machine learning

Matteo Turla 966763

1 Introduction

The goal of a data-driven credit card fraud detection system is to detect, given a transaction and its characteristics, if it is fraudulent or genuine. Usually this system are composed of two layers of security: the first layer estimates a risk score for each transaction using machine learning algorithms, the second layer raises an alert if the risk score is high. A human investigator is then responsible of determining the true origin of the transaction. If the system raises to many alerts then the investigator will be unable to check all of them, if the system raises to few alerts then there is the possibility to miss some fraudulent transactions.

The problem of detecting fraud can be seen as a binary classification problem, starting from a dataset of historical transactions it is required to extract a set of features used to classify each payment. All of this must be done in real-time.

There are three main groups of feature that characterize a credit card transaction:

- Transaction features: account number, transaction amount, transaction date time, terminal number, location and category information.
- Account features: account number, card and contract information.
- Customer information: type of customer, personal information.

The most basic dataset required to perform data-driven frauds detection is composed of a set of historical transactions and a binary label: 1 if it is a fraud, 0 otherwise. In the following are summarised the most important challenges in fraud detection:

- Class imbalance: the percentage of fraudulent transactions is around 1 percent.
- Concept drift: fraud patterns change over time, usually fraud detection system are continuously trained using the most recent transactions history or it is used online learning.
- Sequential modelling: each customer generates a stream of transactions over time, it is required to model this stream of data in order to extract important features, usually rolling window techniques are used.

2 Dataset and exploration analysis

The dataset used in this paper is composed of a history of labaled transactions from 2018-05-01 to 2018-10-01, each transaction is characterized by six features:

- transaction id
- transaction date and time
- customer id
- terminal id
- transaction amount
- fraud label

In total there are 1.466.093 transactions, 12.979 fraudulent transactions (0.89 %) and 1.453.114 genuine transactions (99.11 %). The daily number of genuine and fraudulent transactions follow a stationary process, there are on average 9600 daily transactions of which 85 are fraudulent. The number of transactions grouped by hour follow a Gaussian distribution centered at 12.00, while the distribution of the number of frauds have two important peaks at 01.00 a.m. and at 04.00 a.m. as shown in figure 1, The same explorations is done based on the week day.

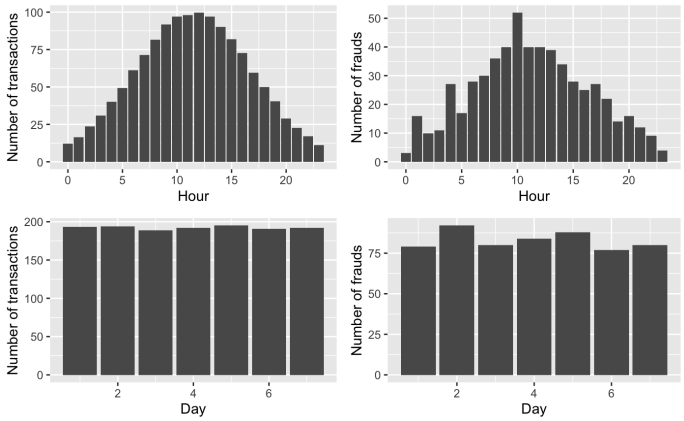
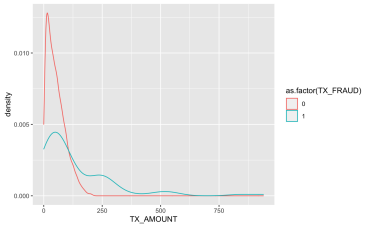
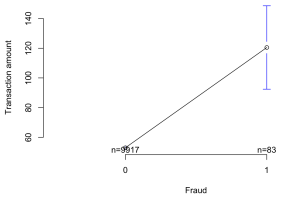


Figure 1: Distribution of transactions grouped by hour (top) and group by day (bottom)

The transaction amount feature has most of its probability mass around 55$, with a very long right tail, as shown in figure 2. Transactions with a very large amount are more likely to be frauds.



(a) Transaction amount distribution



(b) Transaction amount mean

Figure 2: Transaction amount exploration

3 Features engineering

Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models. The design of these features is usually problem dependent and require expert knowledge.

From the current dataset, three main group of features are extracted: date-time related features, customer spending behaviour features and terminal features.

The date-time variable is transformed as follow:

- the first feature define if the transaction occurs during the weekend or not
- the second feature define if the transaction occurs during the day or during the night

The customer type transformation consist in creating features that characterize the customer behaviour spending. In order to define this type of features it is used the Recency, Frequency, Monetary Value framework defined by Van Vlasselaer et al. 2015. It consists in keeping track, for each customer, the average spending amount and number of transactions for different time windows in the past: 1, 7 and 30 days windows.

The terminal type of transformation consists in assigning a risk score to each terminal id based on past frauds, using a delayed time windows. The risk score assesses the exposure of a given terminal ID to fraudulent transactions. It is important to have a delayed period to account for the fact that the true label is known after a certain period of investigation or after a customer compliant, thus we can not use, for a given terminal id, its most recent transactions because the labels are still not available.

The risk score is given by the total number of fraud transactions over the total number of transactions in the last n days shifted by the delay period. n define the size of the time window: 1 day for short term window, 7 days for medium term window and 30 days for long term window. The delay period is usually set at 7 days.

The transformed dataset is then composed of 15 features, figure 3 shows the correlation plot.

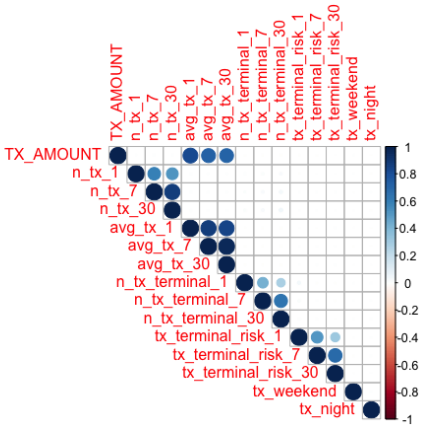


Figure 3: Correlation of the extracted features

4 Metrics

Standard measures for classification problem, as the accuracy, are not well suited for frauds detection due to class unbalanced problem. A dummy classifier which classify all transactions as genuine, achieves 99 % accuracy, but it is useless. The goal of fraud detection system is to maximize the detection of frauds while rising the minimum possible numbers of false alarms.

Fraud detection is a cost-sensitive problem, classify a genuine transaction as fraudulent has a lower business cost than classify a fraudulent transaction as genuine. Missing a fraudulent transaction cause the loss of the amount, loss of future amounts due to the fact that the card is compromised and lower reputation in the company. Instead rising a false alarm cause the blocking of the payment until the investigator check what is happened.

In the following it is discussed some metrics and their implication on fraud detection system, using as reference the paper of Tharwat 2020.

The metrics used in a fraud detection system can be grouped into: threshold based, threshold free and top-k metrics. Threshold based metrics depends on a specific probability threshold, it must be tune for each different model, thus it is difficult to compare models using this type of metrics. The advantage is that they are very simple to interpret and can be used to do a cost sensitive analysis.

Threshold free metrics try to summarise threshold based metrics with a single number, the most used ones are AUC ROC and Average precision.

A fraud detection system has the goal to optimize both precision and recall: maximize the detection of fraudulent transactions and minimize the number of false alarms.

4.1 Threshold based metrics

The fraud detection system returns for each transaction a probability score, the higher its value, the higher is the probability that the transaction is fraudulent. In order to use a threshold based metrics it is required to set a threshold t such that: if $p(x) >= t$ then x is a fraudulent transaction.

The most important threshold based metrics in fraud detection problem are Precision and Recall. Recall is the proportion of true fraudulent transactions in the dataset which are classified as fraudulent, also called detection rate. Precision is the proportion of fraudulent predictions that are indeed frauds or the percentage of correct fraudulent prediction.

A system with high recall and low precision return a lot of alerts, but most of them are incorrect, indeed many false positive. A system with High precision and low recall return few alert and most of them are correct, but it will miss a lot of fraudulent transactions.

4.2 Threshold free metrics

They evaluate the performance of a classifier over all possible thresholds and then provide a number as a summary. Usually this metrics are based on computing the area under the curve (AUC) of a particular curve, for example the Receiving Operating Characteristic Curve (AUC-ROC) or the Precision-Recall curve (Average precision).

The ROC curve compare the true positive rate with the false positive rate at different threshold. it assesses the proportion of detected frauds versus the number of false alarm. The Precision-Recall curve plot the precision against the recall for all different thresholds. The main goal is to put in evidence classifiers that achieve both high precision and high recall, the average precision is a summary of the curve, higher is better. A classifier which always classify transactions as fraudulent achieve an average precision given by the proportion of fraud in the dataset. The main problem of this metric is that it is very difficult to interpret from an operational point of view.

4.3 top-k metrics

This metrics are used to asses the quality of the classifier from an operational point of view. it assumes that the investigator can only check k alerts during a given day. Precision at k is the proportion of the k most suspicious alert that are indeed frauds. a top-k precision of 30% means that 30 out of 100 most suspicious alerts where indeed corrects. Note that the highest value can be lower than 1 if k is higher than the number of fraudulent transactions for a given day. Card precision at k measure the precision at k in terms of cards rather than transactions, the idea is that multiple transactions from the same card should be counted as single alert since the investigator can check all the recent transactions. These metrics are computed daily and aggregated over time by taking the mean.

5 Cross validation for streaming data

To compare different models, tune their hyper parameters and estimate the generalization error, it is required to split the dataset into different sets, in machine learning is generally used cross validation to tune the hyper parameters and an holdout test to estimate the performance of the best model.

When dealing with temporal data it is required to take into account the ordering of the data to avoid temporal information leakage into the trained model and simulate a real-case scenario. The general rule is to test the model on unseen data coming in time after the training data, and between this two sets there must be a delay period to take into account the fact that fraud labels are not immediately available. All the models reported in the following section are trained and tuned using prequential validation and tested on an hold out test coming later in time, a simple visual explanation of the splitting strategy is given by figure 4, each block correspond to one week of data.

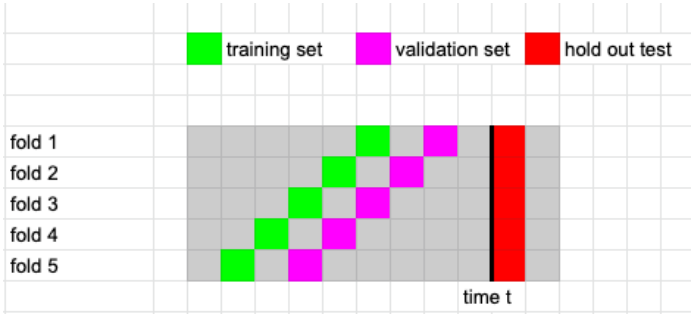


Figure 4: Splitting strategy

6 Supervised techniques

In the current section it is summarised the performance of 4 different models, tuned and tested using the prequential validation strategy. The models are reported in table 1. Table 2 shows the performance of the tuned models, including a dummy classifier which always return 0.5 as probability of being a fraud. As it can be seen, all model are clearly better than the dummy classifier and no one is overfitting or underfitting.

6.1 SVM vs XGboost performance

Both model achieve a p@100 of 63%, this means that out of 100 alerts, 63 were indeed fraudulent transactions, note that the average number of daily fraudulent transactions is 90.

The main difference between SVM and xgboost is that SVM has an higher average precision, thus fixed a certain precision level SVM has an higher recall

Model	hyper-p	Value
Random Forest	n. columns sub-sampled	$[5, 10, 15*]$
XGboost	max depth	$[3*, 6, 9, 12]$
	lambda	$[3, 5*]$
	learning rate	$[0.01, 0.05, 0.1*]$
SVM radial kernel	C	$[1, 10*, 100, 1000]$
	gamma	$[0.001*, 0.0001]$
3-layer NN	learning rate	$[0.001, 0.0001*]$

Table 1: Model and hyper-parameters grid. * is the best found value

Model	set	pk	card p@k	avarage precion	AUC ROC
Dummy classifier	train	0.015	0.007	0.009	0.5
	test	0.008	0.015	0.009	0.5
Random Forest	train	0.601	0.562	0.687	0.856
	test	0.57	0.527	0.649	0.837
XGboost	train	0.724	0.651	0.814	0.996
	test	0.631	0.571	0.662	0.903
SVM radial kernel	train	0.65	0.584	0.719	0.897
	test	0.625	0.561	0.700	0.892
3-layer NN	train	0.58	0.58	0.72	0.91
	test	0.59	0.58	0.66	0.86

Table 2: Model and hyper-parameters grid. * is the best found value

with respect to XGBoost. Also XGboost is more prone to overfit and require the tuning of an higher number of hyper-parameters, thus SVM is preferable.

7 Unsupervised technique for anomaly detection

Anomaly detection is a a set of techniques used to identify data points that deviates from a dataset's normal behaviour. In this case, fraud transaction can be seen as anomaly or outlier of the genuine transactions distribution. Unsupervised clustering methods can be used to cluster the high-dimensional space of the input feature and then compute, for each point, the distance with the nearest cluster. If the distance is high this means that the point is far away from all the other points and thus it can be an anomaly or an outlier. In the following it is analyzed two different type of clustering: k-means and Gaussian mixture model.

The following procedure has been applied to both K-mean and Gaussian mixture model:

- Split training set into genuine and fraudulent transactions

- cluster the genuine training set by optimizing the number of cluster
- Compute the distance metrics for each point in the genuine training set
- decide a distance threshold
- evaluate the model on the test set

This clustering methods are also compared with an unsupervised deep learning techniques used to identify anomaly.

In order to evaluate the model it has been used an hold out dataset split. Train and test set are separated by a delay period.

7.1 Data transformation

K-means and Gaussian mixture model works really well if the input space is an Euclidean space and all the features have the same scale, thus categorical features, even if encoded, should be discarded. Note that even if categorical features can be encoded into a numerical representation, it does not mean that the encoding has a meaningful representation in the Euclidean space.

Before clustering it has been performed a PCA reduction, in particular it has been selected only 3 components, the reason is that distances becomes meaningless as the number of dimension increase as discussed by Aggarwal, Hinneburg, and Keim 2001.

7.2 K-means

K-means clustering is an iterative data partitioning algorithm that assigns to each data point one and only one cluster, in particular, among the k possible clusters, it is assigned the one which minimize the euclidean distance between the data point and the centroid. The number of clusters k has been selected using the Elbow method.

7.3 Gaussian Mixture Model

Gaussian mixture model is a model based clustering algorithm, it assume that our data is distributed according to a linear combination of different multivariate Gaussian distribution. The goal is to fit K Gaussian multivariate distributions in order to maximize the likelihood of our observations:

$$P(X \mid \pi, \mu, \sigma) = \prod_{n=1}^{N} [\sum_{k=1}^{K} \pi_k N(x_n \mid u\mu_k, \Sigma_k)] \tag{1}$$

Where π_k is a real number, u_k is the mean vector of the k^{th} Normal distribution and Σ_k is the corresponding covariance matrix. Usually to optimize this model it is used the Expected maximization algorithm.

To detect outliers is then computed the density of a given data point (equation 2), if this quantity is low, it means that the point is an outlier for our linear combination of Gaussian distribution, and so it is a fraudulent transaction.

$$p(x_i \mid \pi, \mu, \Sigma) = \sum_{k=1}^{K} \pi_k N(x_n \mid u\mu_k, \Sigma_k) \tag{2}$$

7.4 Autoencoder

An autoencoder is a neural network that is trained to learn a latent representation of the training data distribution. The autoencoder architecture is compose of two parts:

- Encoder: it maps the input to a lower dimensional latent space (also called latent code)
- Decoder: it maps the latent space back to the input space

Figure 5, shows the general architecture of an autoencoder.

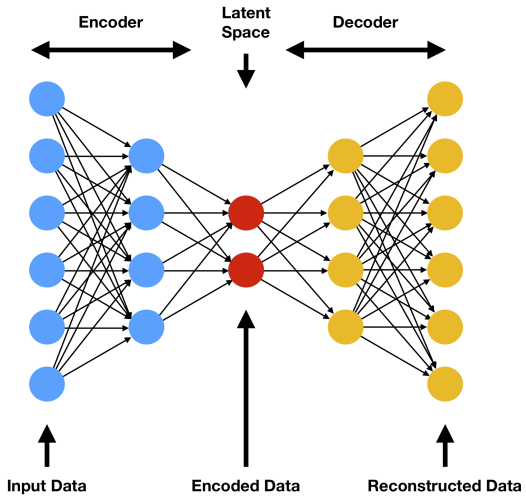


Figure 5: Autoencoder architecture

Because the latent space has a lower dimension than the input space the autoencoder must learn from the data what are the most important information.

For example given an image, the autoencoder first encodes the image into a lower dimensional latent representation, then decodes the latent representation back to an image. The goal of an autoencoder is to minimize the reconstruction loss between the original image and the reconstructed one.

In order to train this type of neural network we do not need any label, this technique is also called self-supervised learning. The goal is to minimize the reconstruction loss (MSE, MAE in general):

$$Loss(x) = norm(Decode(Encode(x)) - x) \tag{3}$$

To train the network it is only used non-fraudulent transaction, in this way the network will learn to reconstruct only the inputs coming from the same data distribution, while all other inputs will be reconstructed very poorly and the reconstruction loss will be very high. Points with very high reconstruction loss are points coming from a different training distribution, in this case they are transaction with a very high risk score.

7.5 Results

Table 3 report the performance of the fitted model on the hold out test set. the performance of this model are worse than the performance of the supervised method, however the clustering models are trained using only 1000 compared to the 67000 observations used by the supervised method. If the dataset present problem of concept drift over time, then this method would be preferable, because it is needed less observations to extract useful pattern from the data.

A concept drift problem is related to the fact that fraudulent patterns may change over time, for example a new fraud technique has been discovered or old techniques become obsolete. The opportunity of obtaining good result using a small number of observations, let the company adapt rapidly to new scenarios.

The best performing method is the autoencoder, but it requires a lot of training data.

Model	pk	average precision
Dummy classifier	0.008	0.009
K-means	0.30	0.23
Gaussian mixture model	0.45	0.37
Autoencoder	0.52	0.50

Table 3: Unsupervised techniques performance

8 Conclusion

In this paper it has been discussed the main problem of fraud detection system and how to approach them with machine learning, in particular it is discussed the right metrics and validation strategy to use with a very unbalanced dataset ordered by time. Supervised methods are stronger than the unsupervised ones, but they require a lot of training data, this could be a problem in scenario with time drift concept. In the end it is discussed how different unsupervised algorithms can be used to detect anomaly.

References

[AHK01] Charu C Aggarwal, Alexander Hinneburg, and Daniel A Keim. "On the surprising behavior of distance metrics in high dimensional space". In: *International conference on database theory*. Springer. 2001, pp. 420–434.

[Tha20] Alaa Tharwat. "Classification assessment methods". In: *Applied Computing and Informatics* (2020).

[Van+15] Véronique Van Vlasselaer et al. "APATE: A novel approach for automated credit card transaction fraud detection using network-based extensions". In: *Decision Support Systems* 75 (2015), pp. 38–48. ISSN: 0167-9236. DOI: https://doi.org/10.1016/j.dss.2015.04.013. URL: https://www.sciencedirect.com/science/article/pii/S0167923615000846.