Credit Card Fraud Detection using Machine Learning Concepts

Riya Tuteja [1*] ,Muskan [2*] ,Navya [3 *] ,Mr Neeraj tyagi

[1*] Computer Science and Engineering, Panipat Institute of Engineering and technology,

Kurukshetra University, Haryana, India

[2*] Computer Science and Engineering, Panipat Institute of Engineering and technology,

Kurukshetra University, Haryana, India

[3 *] Computer Science and Engineering, Panipat Institute of Engineering and technology,

Kurukshetra University, Haryana, India

4 Professor, Department of Computer Science and Engineering,

Panipat Institute of Engineering and technology, Kurukshetra University, Haryana, India

ABSTRACT-It’s miles critical for credit card organizations to realize see fraudulent credit card income for clients they may be not charged for things they did no longer buy. Such troubles can be handled statistics science and its importance, and Mechanical getting to know, can't be skipped. This the project pursuits to reveal the modelling information set being used gadget getting to know approximately credit Card Fraud Detection. credit score The trouble of finding Card Fraud includes modelling beyond debt card transactions and information of these that look like such fraud. This model is then used to look if it's miles new what's being done is fake or now not. Our goal right here is to find out a hundred% faux jobs while minimizing classes of fraudulent fraud. credit score Card Fraud Detection a wellknown sample separation. in this system, we're centered in analysing and prioritizing data sets and the posting of many confusing locating algorithms in PCA changed credit card processing information.

Keywords— Data Visualization, Data predictions,Support Vector Machines, Random Forest Classification, Auto-encoders.

Introduction

In today's world we are doing many transactions with our credit cards(online or offline) ,though there are some fraud transactions too. So it is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase.There is an ever increasing credit card transactions in the world and very few of them are fraudulent ,but considering this fact too every fraud is cheating to someone because it is someone's hard earned money.

Problem Statement

A transaction contains many features like where it was made ,what amount charged in the transaction ,the time of the transaction and many other things like that which can't be revealed to the public as it will be a privacy breach for the user, and the other problem is searching of the fraud transaction data as only few of the transactions are fraud.

Contents

1. **Data**

The datasets contain transactions made by credit cards in September 2013 by europeans cardholders.

This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

It contains only numeric input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

The dataset can be downloaded from this link.

2. Machine Learning Algorithms

1. K-Nearest Neighbours Algorithm

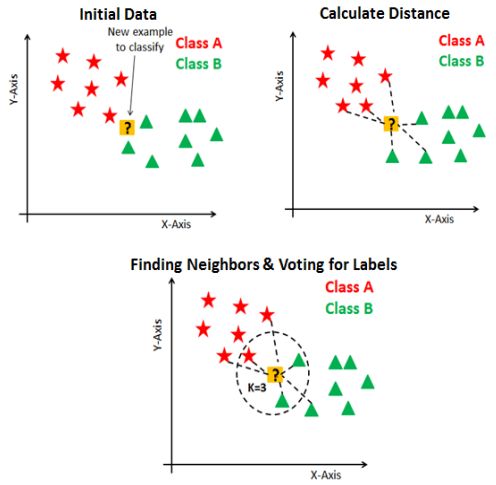
K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions).

Just like almost everything else, KNN works because of the deeply rooted mathematical theories it uses. When implementing KNN, the first step is to transform data points into feature vectors, or their mathematical value. The algorithm then works by finding the distance between the mathematical values of these points. The most common way to find this distance is the Euclidean distance, as shown below.

$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

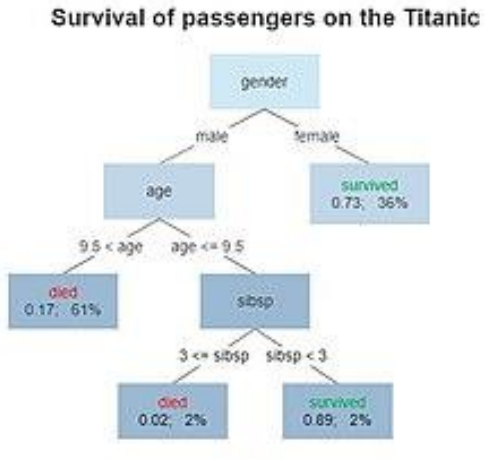
KNN runs this formula to compute the distance between each data point and the test data. It then finds the probability of these points being similar to the test data and classifies it based on which points share the highest probabilities.



2. Decision Tree Classifier

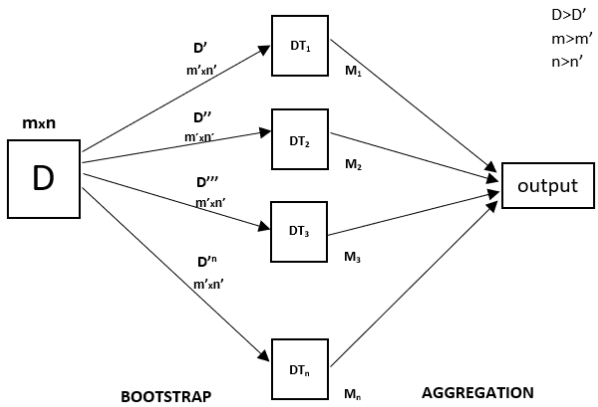
Decision tree build classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision. The

topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.



3. Random Forest

Every decision tree has high variance, but when we combine all of them together in parallel then the resultant variance is low as each decision tree gets perfectly trained on that particular sample data and hence the output doesn't depend on one decision tree but multiple decision trees. In the case of a classification problem, the final output is taken by using the majority voting classifier. In the case of a regression problem, the final output is the mean of all the outputs. This part is Aggregation.



A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as **bagging**. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. This part is called Bootstrap.

4. Naive Bayes(Gaussian)

Given a feature vector *X=(x1,x2,…,xn)* and a class variable *Ck*, Bayes Theorem states that: *P(Ck|X)=P(X|Ck)P(Ck)/(P(X)), for k=1,2,…,K*

We call P(Ck|X) the posterior probability, P(X|Ck) the likelihood, *P(Ck)* the prior probability of a class, and P(X) the prior probability of the predictor. We're interested in calculating the posterior probability from the likelihood and prior probabilities.

Using the chain rule, the likelihood P(X|Ck) can be decomposed as:

$$P(X \mid C_k) = P(x_1, \ldots, x_n \mid C_k) = P(x_1 \mid x_2, \ldots x_n, C_k) P(x_2 \mid x_3, \ldots x_n, C_k) \cdots P(x_{n-1} \mid x_n, C_k) P(x_n \mid C_k)$$

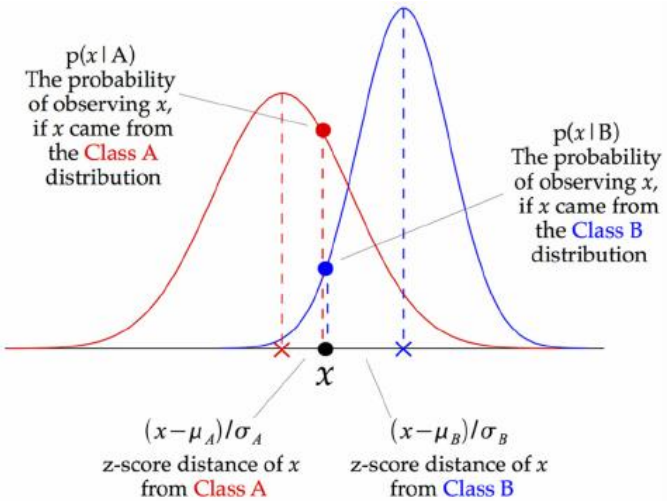
In terms of machine learning, we mean to say that the features provided to us are independent and do not affect each other, and this does not happen in real life. The features depend on the occurrence or value of another, which is simply ignored by the Naive Bayes classifier and is hence given the term, "**NAIVE".**

Thus, by conditional independence, we have:

$$P(y|x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)$$

Gaussian Naive Bayes:

Used in classification, and it assumes that features follow a normal distribution.



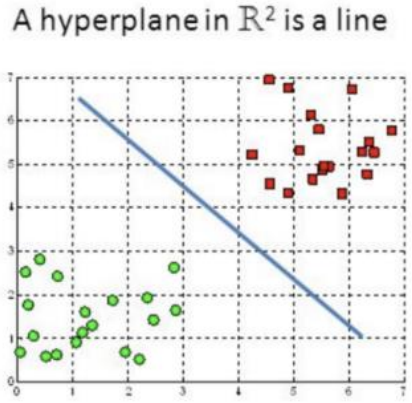
5. Support Vector Machine(Classifier)

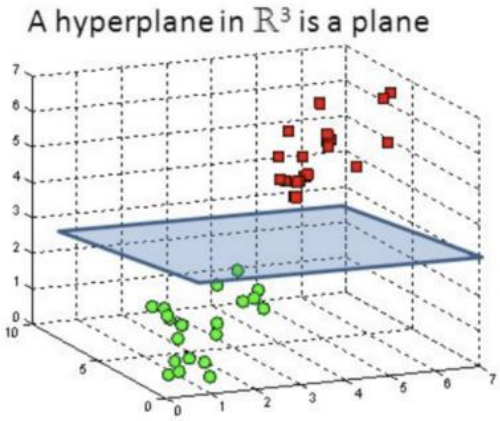
A support vector machine (SVM) is a supervised machine learning algorithm that can be used for both classification and regression tasks. In SVM, we plot data points as points in an n-dimensional space (n being the number of features you have) with the value of each feature being the value of a particular coordinate.

The classification into respective categories is done by finding the optimal **hyperplane** that differentiates the two classes in the best possible manner.

A hyperplane is a generalization of a plane:

1. in two dimensions, it's a line.
2. in three dimensions, it's a plane.
3. in more dimensions, you can call it a hyperplane.

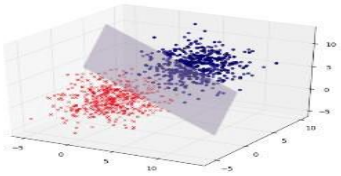




This equation is derived from two-dimensional vectors.

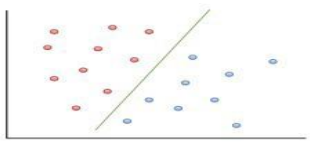
$\mathbf{w}^T\mathbf{x} = 0$

Hyperplane



$y = ax + b$

Line



The hypothesis function h is defined as:

$$h(x_i) = \begin{cases} +1 & if\ w \cdot x + b \geq 0 \\ -1 & if\ w \cdot x + b < 0 \end{cases}$$

The point above or on the hyperplane will be classified as class +1, and the point below the hyperplane will be classified as class -1.

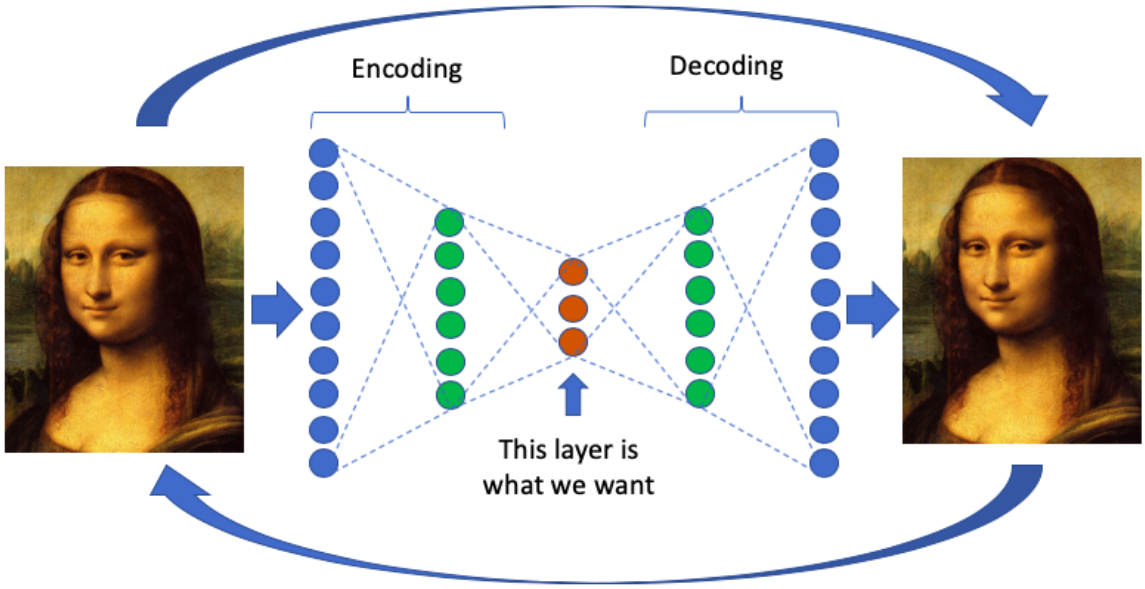
Computing the (soft-margin) SVM classifier amounts to minimizing an expression of the form.

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(w \cdot x_i - b)\right)\right] + \lambda\|w\|^2.$$

6. Auto-encoders for anomaly detection

An auto-encoder is a special type of neural network that copies the input values to the output values as shown in Figure below. It does not require the target variable like the conventional Y, thus it is categorized as unsupervised learning. You may ask why we train the model if the output values are set to equal to the input values. Indeed, we are not so much interested in the output layer. We are interested in the hidden core layer. If the number of neurons in the hidden layers is less than that of the input layers, the hidden layers will extract the essential information of the input values. This condition forces the hidden layers to learn the most patterns of the data and ignore the "noises". So in an auto-encoder model, the hidden layers must have fewer dimensions than those of the input or output layers. If the number of neurons in the hidden layers is more than those of the input layers, the neural network will be given too much capacity to learn the data. In an extreme case, it could just simply copy the input to the output values, including noises, without extracting any essential information.

Figure below also shows the encoding and decoding process. The encoding process compresses the input values to get to the core layer. The decoding process reconstructs the information to produce the outcome. The decoding process mirrors the encoding process in the number of hidden layers and neurones. Most practitioners just adopt this symmetry.



What we will do is try to regenerate the transaction without knowing whether the transaction is fraudulent or not, then we measure the reconstruction error for a transaction, and on the basis of that we will predict the type of transaction.

3. Measuring the accuracy

We will measure the accuracy of a model by the number of fraudulent transactions predicted as genuine.

Results

Though we have increased the cases for false negatives (i.e. the case where genuine transaction is considered as fraud) but the reduction in case of fraud transaction matters more. So we can consider it as the best model till now. This research is all about studying credit card fraud-detection models based on different machine learning classification algorithms. The goal is to be in this training and testing. To find out the best way to process the dataset and the best machine learning classification algorithm for the dataset of this credit card transaction. So to achieve this, we chose five different classifiers, respectively. Between them, ten different combinations of algorithms and sampling methods were used to evaluate their predicted performance as a way to get better results for credit card fraud detection. Finally, we cross-validated the technique applied to all the individual classifiers to obtain more accurate results.

References:

1. You can visit this link for the Github link for the code and all the pictures related to the project.
2. You can visit this link for the google drive link for the code ,colab file and all the pictures related to the project, just make a few changes to the notebook like change the mounting addresses to get and make files(images , data files etc.)
3. Referred to wikipedia and official website (http://scikit-learn.org/stable/modules/manifold.html) for the code part.
4. M. Kanchana, V. Chadda, and H. Jain, ―Credit card fraud detection,‖ Int. J. Adv. Sci. Technol., vol. 29, no. 6, pp. 2201–2215, 2020, doi: 10.17148/ijarcce.2016.5109.
5. A. RB and S. K. KR, ―Credit Card Fraud Detection Using Artificial Neural Network,‖ Glob. Transitions Proc., pp. 0–8, 2021, doi: 10.1016/j.gltp.2021.01.006.
6. R. R. Popat and J. Chaudhary, ―A Survey on Credit Card Fraud Detection Using Machine Learning,‖ Proc. 2nd Int. Conf. Trends Electron. Informatics, ICOEI 2018, vol. 25, no. 01, pp. 1120–1125, 2018, doi: 10.1109/ICOEI.2018.8553963.
7. O. Adepoju, J. Wosowei, S. Lawte, and H. Jaiman, ―Comparative Evaluation of Credit Card Fraud Detection Using Machine Learning Techniques,‖ 2019 Glob. Conf. Adv. Technol. GCAT 2019, pp. 1–6, 2019, doi: 10.1109/GCAT47503.2019.8978372.
8. M. Deepa and D. Akila, ―Survey Paper for Credit Card Fraud Detection Using Data Mining Techniques,‖ Int. J. Innov. Res. Appl. Sci. Eng., vol. 3, no. 6, p. 483, 2019, doi: 10.29027/ijirase.v3.i6.2019.483-489.
9. P. Save, P. Tiwarekar, K. N., and N. Mahyavanshi, ―A Novel Idea for Credit Card Fraud Detection using Decision Tree,‖ Int. J. Comput. Appl., vol. 161, no. 13, pp. 6–9, 2017, doi: 10.5120/ijca2017913413.
10. J. Vimala Devi and K. S. Kavitha, ―Fraud Detection in Credit Card Transactions by using Classification Algorithms,‖ Int. Conf. Curr. Trends Comput. Electr. Electron. Commun. CTCEEC 2017, pp. 125–131, 2018, doi: 10.1109/CTCEEC.2017.8455091.
11. R. R. Popat and J. Chaudhary, ―A Survey on Credit Card Fraud Detection Using Machine Learning,‖ Proc. 2nd Int. Conf. Trends Electron. Informatics, ICOEI 2018, no. Icoei,
12. pp. 1120–1125, 2018, doi: 10.1109/ICOEI.2018.8553963.
13. S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang, and C. Jiang, ―Random forest for credit card fraud detection,‖ ICNSC 2018 - 15th IEEE Int. Conf. Networking, Sens. Control, pp. 1–6, 2018, doi: 10.1109/ICNSC.2018.8361343.

14. S. Mittal and S. Tyagi, ―Performance evaluation of machine learning algorithms for credit card fraud detection,‖ Proc. 9th Int. Conf. Cloud Comput. Data Sci. Eng. Conflu. 2019, pp. 320–324, 2019, doi: 10.1109/CONFLUENCE.2019.8776925.
15. X. Yu, X. Li, Y. Dong, and R. Zheng, ―A Deep Neural Network Algorithm for Detecting Credit Card Fraud,‖ Proc. - 2020 Int. Conf. Big Data, Artif. Intell. Internet Things Eng.
16. ICBAIE 2020, pp. 181–183, 2020, doi: 10.1109/ICBAIE49996.2020.00045.
17. S. Bagga, A. Goyal, N. Gupta, and A. Goyal, ―Credit Card Fraud Detection using Pipeling and Ensemble Learning,‖ Procedia Comput. Sci., vol. 173, pp. 104–112, 2020, doi: 10.1016 j.procs.2020.06.014.
18. R. San Miguel Carrasco and M.-A. Sicilia-Urban, ―Evaluation of Deep Neural Networks for Reduction of Credit Card Fraud Alerts,‖ IEEE Access, vol. 8, pp. 186421–186432, 2020, doi: 10.1109/access.2020.3026222.