Bangalore Institute of Technology

Department of Computer Science and Engineering

K R Road, V V Puram, Bengaluru-560004

Mini Project Synopsis on

Credit Card Fraud Detection

Submitted as the mini project for the subject
Data Mining and Data Warehousing (18CS641)

Submitted by

Samarth S Hiremath	**1BI20CS150**
Vamshik R	**1BI20CS144**
Rakshith Rajesh N B	**1BI21CS408**

For academic year 2022-23

Under the guidance of
Dr. Bhargavi M S
Associate Professor

Introduction to the problem:

With the increase of people using credit cards in their daily lives, credit card companies should take special care in the security and safety of the customers. According to (Credit card statistics 2021) the number of people using credit cards around the world was 2.8 billion in 2019, in addition 70% of those users own a single card at least.

Reports of Credit card fraud in the US rose by 44.7% from 271,927 in 2019 to 393,207 reports in 2020. There are two kinds of credit card fraud, the first one is by having a credit card account opened under your name by an identity thief, reports of this fraudulent behavior increased 48% from 2019 to 2020. The second type is by an identity thief uses an existing account that you created, and it's usually done by stealing the information of the credit card, reports on this type of fraud increased 9% from 2019 to 2020 (Daly, 2021). Those statistics caught my attention as the numbers are increasing drastically and rapidly throughout the years, which gave me the motive to try to resolve the issue analytically by using different machine learning methods to detect the credit card fraudulent transactions within numerous transactions.

Nowadays as we can see that there is a huge increase online payment, and the payment is mostly done with the help of credit cards. It becomes a big problem for marketing company to overcome with the credit card fraudulent activities. Fraudulent can be done in many ways such as tax return in any other account, taking loans with wrong information etc. Therefore, we need an efficient fraudulent detection model to minimize fraudulent activity and to minimize their losses.

The primary objectives of this project are as follows:

Dataset Selection: We will carefully select diverse datasets from various domains, ensuring they represent real-world scenarios and challenges. These datasets will encompass different dimensions and characteristics, allowing us to evaluate the effectiveness of **Random Forest classifier** across various contexts.

Random Forest Algorithm: Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset**.**" Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

Feature Engineering and Preprocessing: Prior to applying the Random Forest algorithm, we will conduct comprehensive feature engineering and preprocessing tasks to ensure the data's quality and suitability for clustering. This may involve handling missing values, normalizing features, and addressing outliers to enhance the accuracy and robustness of the clustering results.

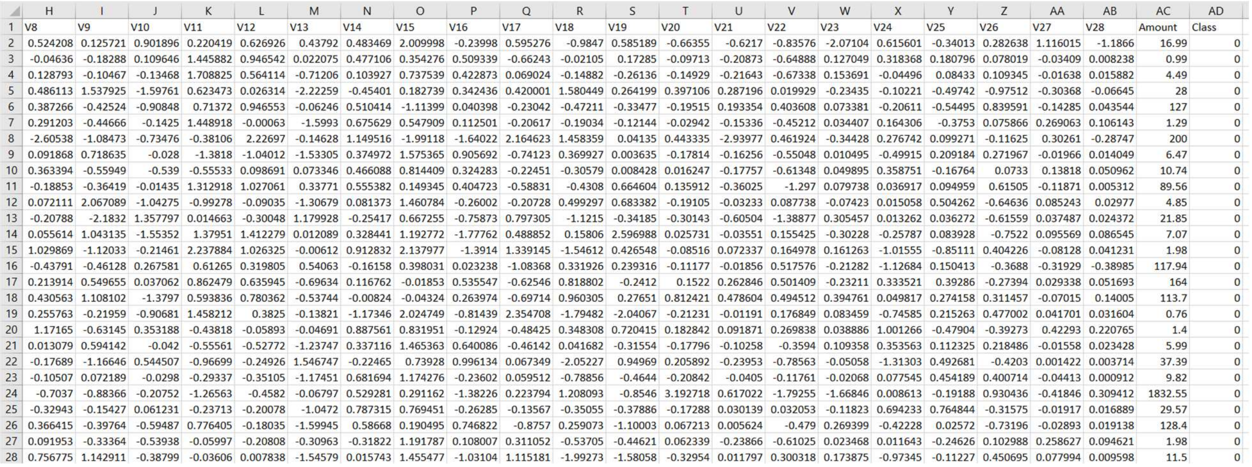
Evaluation and Interpretation: We will evaluate the effectiveness of the Random Forest classifier results using internal evaluation metrics like the silhouette score or external metrics if ground truth labels are available. Additionally, we will interpret the resulting clusters by analyzing their characteristics and visualizing the clusters to gain insights and facilitate decision-making.

By conducting this project, we aim to showcase the potential of the k-means clustering algorithm as a valuable tool for pattern discovery and data segmentation. The outcomes of our analysis will contribute to identifying distinct groups, understanding relationships between data points, and informing decision-making processes across various domains.

Dataset Description:

The dataset was retrieved from an open-source website, Kaggle.com. it contains data of transactions that were made in 2013 by credit card users in Europe, in two days only. The dataset consists of 30 attributes, 11558 rows. 28 attributes are numeric variables that due to confidentiality and privacy of the customers have been transformed using PCA transformation, the two remaining attributes are “Amount” is the amount of each transaction, and the final attribute “Class” which contains binary variables where “1” is a case of fraudulent transaction, and “0” is not as case of fraudulent transaction.

Snapshot



	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD
1	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
2	0.524208	0.125721	0.901896	0.220419	0.626926	0.43792	0.483469	2.009998	-0.23998	0.595276	-0.9847	0.585189	-0.66355	-0.6217	-0.83576	-2.07104	0.615601	-0.34013	0.282638	1.116015	-1.1866	16.99	0
3	-0.04636	-0.18288	0.109646	1.445882	0.946542	0.022075	0.477106	0.354276	0.509339	-0.66243	-0.02105	0.17285	-0.09713	-0.20873	-0.64888	0.127049	0.318368	0.180796	0.078019	-0.03409	0.008238	0.99	0
4	0.128793	-0.10467	-0.13468	1.708825	0.564114	-0.71206	0.103927	0.737539	0.422873	0.069024	-0.14882	-0.26136	-0.14929	-0.21643	-0.67338	0.153691	-0.04496	0.08433	0.109345	-0.01638	0.015882	4.49	0
5	0.486113	1.537925	-1.59761	0.623473	0.026314	-2.22259	-0.45401	0.182739	0.342436	0.420001	1.580449	0.264199	0.397106	0.287196	0.019929	-0.23435	-0.10221	-0.49742	-0.97512	-0.30368	-0.06645	28	0
6	0.387266	-0.42524	-0.90848	0.71372	0.946553	-0.06246	0.510414	-1.11399	0.040398	-0.23042	-0.47211	-0.33477	-0.19515	0.193354	0.403608	0.073381	-0.20611	-0.54495	0.839591	-0.14285	0.043544	127	0
7	0.291203	-0.44666	-0.1425	1.448918	-0.00063	-1.5993	0.675629	0.547909	0.112501	-0.20617	-0.19034	-0.12144	-0.02942	-0.15336	-0.45212	0.034407	0.164306	-0.3753	0.075866	0.269063	0.106143	1.29	0
8	-2.60538	-1.08473	-0.73476	-0.38106	2.22697	-0.14628	1.149516	-1.99118	-1.64022	2.164623	1.458359	0.04135	0.443335	-2.93977	0.461924	-0.34428	0.276742	0.099271	-0.11625	0.30261	-0.28747	200	0
9	0.091868	0.718635	-0.028	-1.3818	-1.04012	-1.53305	0.374972	1.575365	0.905692	-0.74123	0.369927	0.003635	-0.17814	-0.16256	-0.55048	0.010495	-0.49915	0.209184	0.271967	-0.01966	0.014049	6.47	0
10	0.363394	-0.55949	-0.539	-0.55533	0.098691	0.073346	0.466088	0.814409	0.324283	-0.22451	-0.30579	0.008428	0.016247	-0.17757	-0.61348	0.049895	0.358751	-0.16764	0.0733	0.13818	0.050962	10.74	0
11	-0.18853	-0.36419	-0.01435	1.312918	1.027061	0.33771	0.555382	0.149345	0.404723	-0.58831	-0.4308	0.664604	0.135912	-0.36025	-1.297	0.079738	0.036917	0.094959	0.61505	-0.11871	0.005312	89.56	0
12	0.072111	2.067089	-1.04275	-0.99278	-0.09035	-1.30679	0.081373	1.460784	-0.26002	-0.20728	0.499297	0.683382	-0.19105	-0.03233	0.087738	-0.07423	0.015058	0.504262	-0.64636	0.085243	0.02977	4.85	0
13	-0.20788	-2.1832	1.357797	0.014663	-0.30048	1.179928	-0.25417	0.667255	-0.75873	0.797305	-1.1215	-0.34185	-0.30143	-0.60504	-1.38877	0.305457	0.013262	0.036272	-0.61559	0.037487	0.024372	21.85	0
14	0.055614	1.043135	-1.55352	1.37951	1.412279	0.012089	0.328441	1.192772	-1.77762	0.488852	0.15806	2.596988	0.025731	-0.03551	0.155425	-0.30228	-0.25787	0.083928	-0.7522	0.095569	0.086545	7.07	0
15	1.029869	-1.12033	-0.21461	2.237884	1.026325	-0.00612	0.912832	2.137977	-1.3914	1.339145	-1.54612	0.426548	-0.08516	0.072337	0.164978	0.161263	-1.01555	-0.85111	0.404226	-0.08128	0.041231	1.98	0
16	-0.43791	-0.46128	0.267581	0.61265	0.319805	0.54063	-0.16158	0.398031	0.023238	-1.08368	0.331926	0.239316	-0.11177	-0.01856	0.517576	-0.21282	-1.12684	0.150413	-0.3688	-0.31929	-0.38985	117.94	0
17	0.213914	0.549655	0.037062	0.862479	0.635945	-0.69634	0.116762	-0.01853	0.535547	-0.62546	0.818802	-0.2412	0.1522	0.262846	0.501409	-0.23211	0.333521	0.39286	-0.27394	0.029338	0.051693	164	0
18	0.430563	1.108102	-1.3797	0.593836	0.780362	-0.53744	-0.00824	-0.04324	0.263974	-0.69714	0.960305	0.27651	0.812421	0.478604	0.494512	0.394761	0.049817	0.274158	0.311457	-0.07015	0.14005	113.7	0
19	0.255763	-0.21959	-0.90681	1.458212	0.3825	-0.13821	-1.17346	2.024749	-0.81439	2.354708	-1.79482	-2.04067	-0.21231	-0.01191	0.176849	0.083459	-0.74585	0.215263	0.477002	0.041701	0.031604	0.76	0
20	1.17165	-0.63145	0.353188	-0.43818	-0.05893	-0.04691	0.887561	0.831951	-0.12924	-0.48425	0.348308	0.720415	0.182842	0.091871	0.269838	0.038886	1.001266	-0.47904	-0.39273	0.42293	0.220765	1.4	0
21	0.013079	0.594142	-0.042	-0.55561	-0.52772	-1.23747	0.337116	1.465363	0.640086	-0.46142	0.041682	-0.31554	-0.17796	-0.10258	-0.3594	0.109358	0.353563	0.112325	0.218486	-0.01558	0.023428	5.99	0
22	-0.17689	-1.16646	0.544507	-0.96699	-0.24926	1.546747	-0.22465	0.73928	0.996134	0.067349	-2.05227	0.94969	0.205892	-0.23953	-0.78563	-0.05058	-1.31303	0.492681	-0.4203	0.001422	0.003714	37.39	0
23	-0.10507	0.072189	-0.0298	-0.29337	-0.35105	-1.17451	0.681694	1.174276	-0.23602	0.059512	-0.78856	-0.4644	-0.20842	-0.0405	-0.11761	-0.02068	0.077545	0.454189	0.400714	-0.04413	0.000912	9.82	0
24	-0.7037	-0.88366	-0.20752	-1.26563	-0.4582	-0.06797	0.529281	0.291162	-1.38226	0.223794	1.208093	-0.8546	3.192718	0.617022	-1.79255	-1.66846	0.008613	-0.19188	0.930436	-0.41846	0.309412	1832.55	0
25	-0.32943	-0.15427	0.061231	-0.23713	-0.20078	-1.0472	0.787315	0.769451	-0.26285	-0.13567	-0.35055	-0.37886	-0.17288	0.030139	0.032053	-0.11823	0.694233	0.764844	-0.31575	-0.01917	0.016889	29.57	0
26	0.366415	-0.39764	-0.59487	0.776405	-0.18035	-1.59945	0.58668	0.190495	0.746822	-0.8757	0.259073	-1.10003	0.067213	0.005624	-0.479	0.269399	-0.42228	0.02572	-0.73196	-0.02893	0.019138	128.4	0
27	0.091953	-0.33364	-0.53938	-0.05997	-0.20808	-0.30963	-0.31822	1.191787	0.108007	0.311052	-0.53705	-0.44621	0.062339	-0.23866	-0.61025	0.023468	0.011643	-0.24626	0.102988	0.258627	0.094621	1.98	0
28	0.756775	1.142911	-0.38799	-0.03606	0.007838	-1.54579	0.015743	1.455477	-1.03104	1.115181	-1.99273	-1.58058	-0.32954	0.011797	0.300318	0.173875	-0.97345	-0.11227	0.450695	0.077994	0.009598	11.5	0

Preprocessing Techniques to be used:

As there are no NAs nor duplicated variables, the preparation of the dataset was simple the first alteration that was made to be able to open the dataset on Weka program is changing the type of the class attribute from Numeric to Class and identify the class as {1,0} using the program Sublime Text. Another alteration was made on the type as well on the R program to be able to create the model and the visualization.

Classifier technique to be applied:

Random Forest Classifier algorithm: -

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Tools to be used:

1. Google Colaboratory