Report: Detect credit card fraud based on anonymized transaction data

Name: Saurav Kumar Saha
Matriculation Number: 902105

1. Submission of RapidMiner files (ordered in execution sequence)

[a] **Exam-1-Loading.rmp**

[b] **Exam-1-Import.rmp**

[c] **Exam-1-Basic-Preparation.rmp**

[d] **Exam-1-Data-Pre-Processing-Initial.rmp**

[e] **Exam-1-Data-Pre-Processing-Final.rmp**

[f] **Exam-1-Model-Comparison-01-Final.rmp (**Task: Basic Workflow**)**

[g] **Exam-1-Model-Comparison-02-Final.rmp (**Task: Refined Workflow**)**

[h] **Exam-1-Final-Model.rmp**

[i] **Exam-1-Model-Tryout.rmp**

(***) Not important as part of exam task (just tried out)

[j] **Exam-1-Measure-Attribute-Influence.rmp**

2. Documentation

(a) Give a brief introduction to the data

The provided data **Exam-1.csv** is a dataset of anonymized credit card transactions. The dataset has **227,846** transaction records (Examples/Rows) and **31** descriptive variables (Attributes/Columns). The examples are anonymized so there is no descriptive name for each example.

The attributes are – Time, V1 … V28 (28 attributes), Amount and Class

Time values having a range [0 - 172792] might indicate time of the transaction, low value meaning occurring earlier and large value meaning later.

V1-V28 attributes are numeric values (continuous) having different ranges have no meta information provided.

Amount values having range [0 – 25691.160] denotes transaction amount in some currency.

Class values having range [0 - 1] denotes if the transaction was correct (0) or fraudulent (1). 0 has been interpreted as correct by natural occurrences as out of 227,846 records 227,452 (99.827%) have 0 as their Class and only 394 (0.173%) records have 1 as their Class. We except most transaction to be normal or correct.

If we look at distributions (Bell Curve) for the V* attributes along with Class value we can see that for most of them there is a significant difference in the distributions across correct and fraud classes. Although Amount, Time, V13, V15, V24 and V26 on the other hand have more similar distributions across Class which are hard to distinguish.

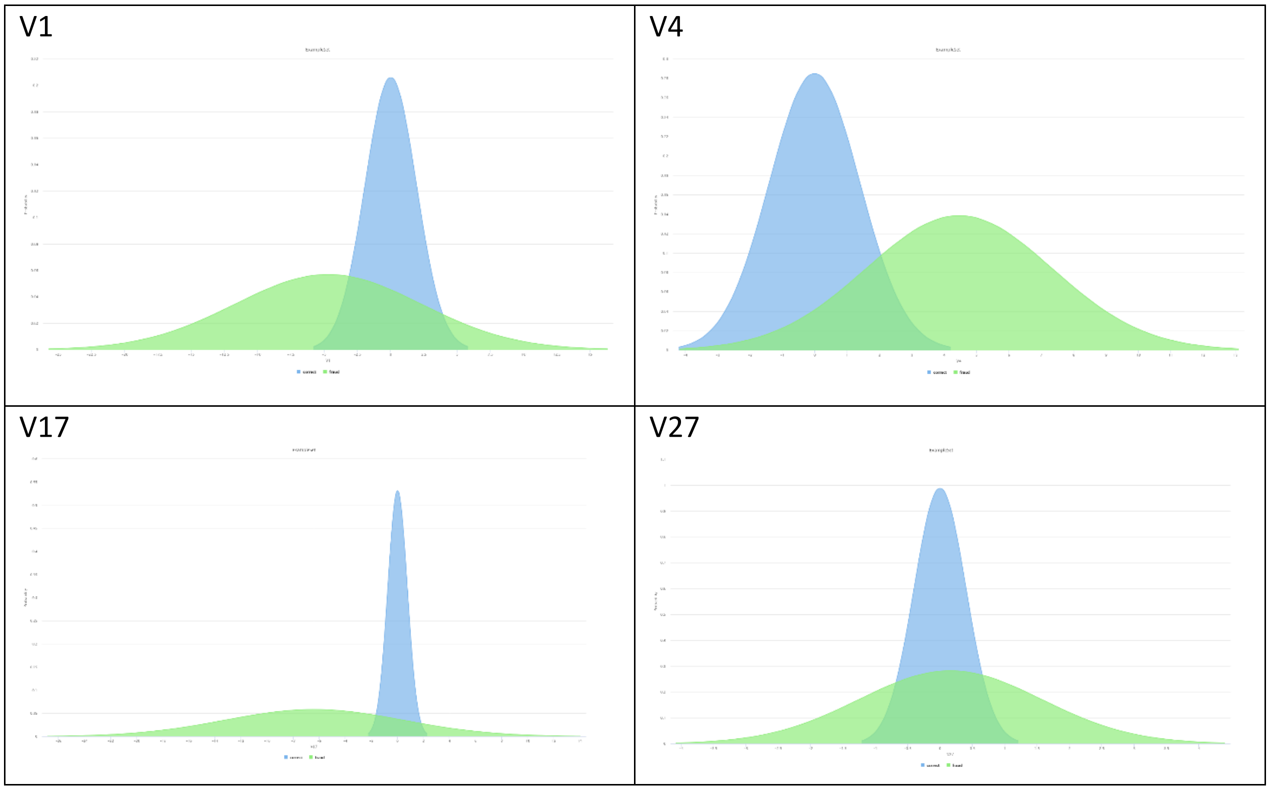


Figure - 1: Distribution of V1, V4, V17, V27 attributes along Class (Blue: correct, Green: fraud)

We can also look at box plots to get an idea how central point and spread measures differs for the attributes.

Time

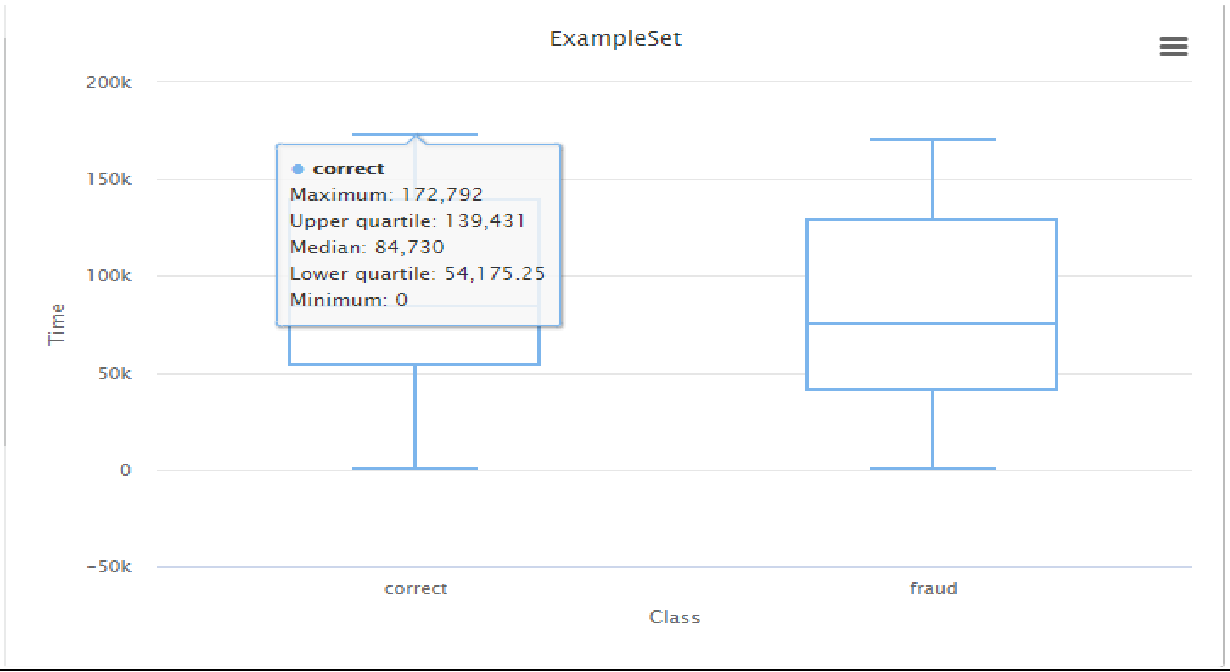


Figure - 2: Box Plot of Time vs Class

No missing value for all examples found.

(b) Give a brief introduction to the machine learning task

This is a supervised machine learning task as we have an outcome variable (Class) and we have to predict it based on the other explanatory variables (Amount, Time, V1-V28) on unseen data. And this is a classification task as our outcome variable is a nominal variable with two distinct classes, correct and fraud. We can use several available classification machine learning models like k-NN, Naive Bayes, Decision Tree, Logistic Regression to fit our data and predict for unseen data in future.

(c) + (d) + (e) + (f) + (g)

1. Data Loading

Used Read CSV and Store operators to load and store the data in RapidMiner data format.

- for Read CSV, used semicolon (;) as separator and first row as names checked
- for Store operator used Exam-1-Data as name for the data object in repository

2. Data Import

- Used Retrieve operator to import previously stored data.
- Used Split Data operator with stratified sampling and 0.7 + 0.3 ratio as partitions for training and test split. Also used 2020 as local random seed.

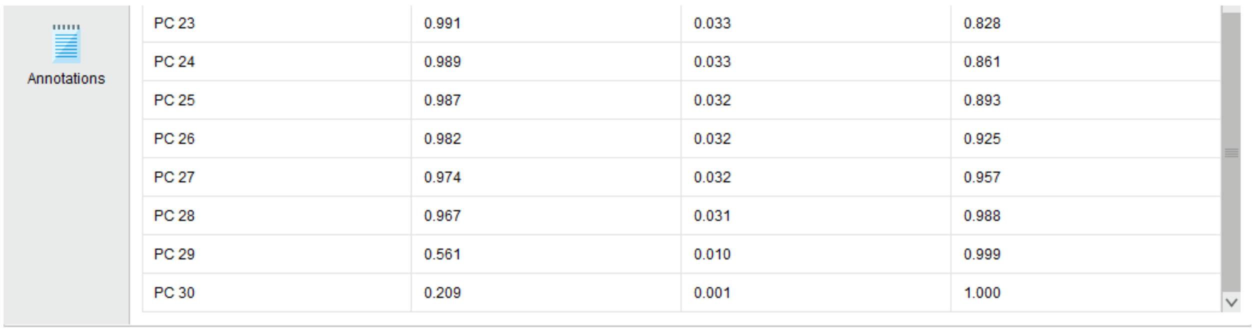
These process splits the unprocessed data in training and test split.

3. Basic Preparation

- Used Set Role operator to mark the Class attribute as label.
- Applied Numerical to Binomial operator to Class attribute with both min and max boundary set to zero to convert zeros as binomial false and ones as binomial true**.**
- Applied Map operator to Class attribute to with value mappings false as correct and true as fraud classes

4. Data Pre-Processing

- Used Z-transformation (centering and scaling)
- applied PCA for principal component analysis with 95% variance retention that ultimately retains 27 PCA dimension out of 30 PCA



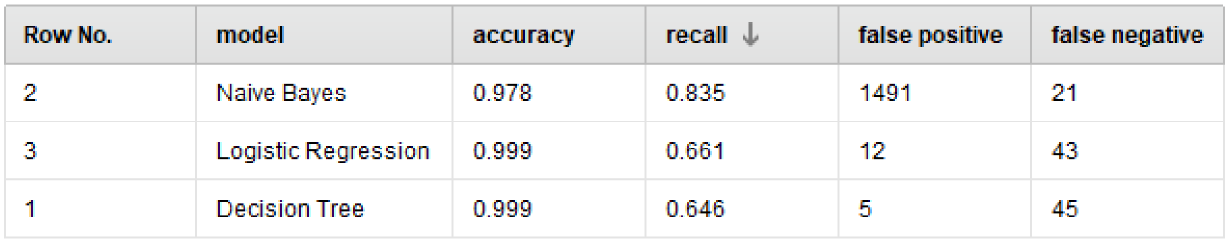
PC 23	0.991	0.033	0.828
PC 24	0.989	0.033	0.861
PC 25	0.987	0.032	0.893
PC 26	0.982	0.032	0.925
PC 27	0.974	0.032	0.957
PC 28	0.967	0.031	0.988
PC 29	0.561	0.010	0.999
PC 30	0.209	0.001	1.000

5. Task: 1 Basic Workflow

- Used previously completed Data Import and Basic Preparation process
- Used 3 classification models Decision Tree, Naive Bayes and Logistic Regression
- Used 10-fold Cross-Validation with stratified sampling and local random seed 2020 for each model

- Decision Tree: criterion=gain ration, maximal depth=20
- Naive Bayes: laplace correction marked
- Logistic Regression: max iterations=0

- Used sub-process per model
- Used a custom sub-process to calculate an aggregated test performance for all models with model name, accuracy, recall, false positive and false negative measures to get and overview of which model is doing well in which perspectives



Row No.	model	accuracy	recall ↓	false positive	false negative
2	Naive Bayes	0.978	0.835	1491	21
3	Logistic Regression	0.999	0.661	12	43
1	Decision Tree	0.999	0.646	5	45

Here is the test performance of all 3 models applied to 30% stratified split ordered by recall (fraud as positive) in descending order.

All the models perform very well in terms of predicting the correct types but in terms of detecting all the fraud cases the models somewhat perform poorly which can be seen from the recall and false negative columns. For Decision Tree, it misclassified almost 45 records as correct which were in fact fraud cases and for Logistic Regression & Naïve Bayes the numbers are 43 and 21 respectively.

There is also a significant measure can be looked upon the false positive numbers which indicate even though they were correct types models predicted them as fraud cases.

6. Task:2 Refined Workflow

- Used Decision Tree, Naive Bayes and Logistic Regression but this time with hyper-parameter optimization after cross validation for each hyper-parameter combination.

- Used Sampling to balance between the correct and fraud class examples. Used following class ratio – fraud = 1.0 and correct = 0.01, tried other ratios like 1.0:0.02, 1.0:0.05, but this ratio worked better for me.

- Used recall as main criterion in Training performance to be considered in hyper-parameter optimization steps

- Used Log, Log Data and Store operator to store the hyper-parameter and recorded performance measures

- Used a 10x (arbitrary) misclassification cost for fraud case to define the Find Threshold and applied to predicted test data by model with confidence values to shift the thresholds by penalizing more for misclassifying the fraud cases.

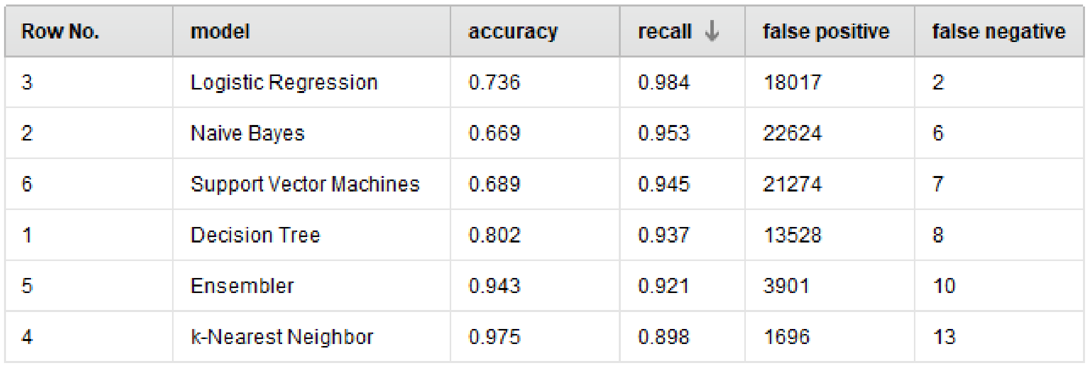
- Tried a k-NN model also.

- Observation: k-NN model performed better after pre-processing indicates that data are better represented in PCA and better stick together in reference to classes due to the natural variances in between them.

- Also used an Ensembler model with optimized Decision Tree (criterion=information gain, maximal_depth-56), Naive Bayes and Logistic Regression (max iterations=3) models using the Log records of optimizations of those models.

- Used an advances model, Support Vector Machines with polynomial kernel type.

- Observation: k-NN and SVM models takes a great amount of time and memory if they are exposed to full training dataset, and to my understanding this is probably because both of these two algorithm uses point-to-point distance calculation which ultimately takes more time and memory.

- 

Row No.	model	accuracy	recall ↓	false positive	false negative
3	Logistic Regression	0.736	0.984	18017	2
2	Naive Bayes	0.669	0.953	22624	6
6	Support Vector Machines	0.689	0.945	21274	7
1	Decision Tree	0.802	0.937	13528	8
5	Ensembler	0.943	0.921	3901	10
4	k-Nearest Neighbor	0.975	0.898	1696	13

- From Task-2 result, we can see Logistic Regression performs best in terms of predicting the fraud cases, it failed to detect only 2 fraud cases. Although the accuracy drops for almost all models with predicting more correct cases as fraud cases which can be seen in false positive column (fraud as positive). This is due to using the 10x misclassification cost for fraud cases and applying the derived threshold to make the models to be more crisp classifiers. Also tried with more cost 50x when the Naive Bayes scored one false negative but obviously more false positives almost 2/3 of the examples.