

Fraud detection at self-checkouts in retail

Laporan Data Mining



Disusun oleh:

Muhammad Naufal Syawali Akbar (1301164488)

Kelas:

Mata Kuliah Pilihan Data Mining

IF41-GAB01

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

BANDUNG

2019

Link video presentasi:

<https://youtu.be/-SpTkWn-kzQ>

A. Latar Belakang Masalah

Dalam Data Mining Cup (DMC) 2019 topik yang dilombakan adalah “*Fraud detection at self-checkouts in retail*”, artinya pada lomba ini menggunakan dataset terkait dengan transaksi alat tulis yang dibayar secara otomatis atau menggunakan metode “self-checkout”. Dimana pelanggan melakukan scan terhadap produk secara langsung dan membayar produk di tempat kasir pembayaran. Customer dapat menggunakan smartphone untuk melakukan scan atau toko memberikan fasilitas *mobile scanner*.

Dengan adanya sistem automasi ini memang membantu pembeli agar lebih cepat dan tidak mengantri di depan kasir pembayaran. Namun dengan cara ini bisa saja menimbulkan kecurangan diantara pembeli terhadap penjual, oleh karena itu tugas dari dmc 2019 ini adalah mengekspos apakah terjadi tindak *Fraudsters* (penipuan) atau tidak, dari data *purchases* (pembelian).

Untuk melihat atau memprediksi apakah terjadi *fraudster* disini penulis menggunakan teknik prediksi klasifikasi *K Nearest Neighbor (KNN)* dan *Random Forest* sebagai pembanding untuk mendapatkan performansi prediksi terbaik, penulis menyarankan teknik berikut karena dataset yang diberikan oleh dmc 2019 memiliki data latih cukup untuk membuat model semaksimal mungkin, dan untuk mengekstraksi fitur dari dataset penulis akan menggunakan *Principle Component Analysis (PCA)* karena perlu dilakukannya penyederhanaan dimensi dalam ekstraksi fitur yang diperlukan.

B. Tujuan

Tujuan dari tugas analisis data mining ini adalah untuk memprediksi terjadinya *fraudting* (penipuan) terkait dengan pembeli terhadap penjual peralatan alat tulis. Pada data testing yang diberikan oleh dmc 2019 perlu melakukan prediksi model yang nantinya disimpan di data testing tersebut, dengan membuat model terlebih dahulu dengan menggunakan data training.

C. Deskripsi Dataset

Dataset yang diberikan oleh dmc 2019 dibagi menjadi dua kategori yaitu data train data data test dimana masing – masing jumlah data yaitu 1800 untuk data training dan 49000 untuk data testing. Pada data train terdapat 10 kolom fitur dengan tipe data berupa integer dan float, sedangkan untuk data testing terdapat 9 kolom fitur saja. Fitur – fitur tersebut diantara lain adalah.

- Trust level: nilai indeks kepercayaan pelanggan secara individual
- Total scan time: nilai total waktu dalam detik dimana produk awal dan akhir di scan
- Grand total: hasil akhir dari produk – produk yang di scan
- Line item void: jumlah scan yang kosong
- Scans without registration: jumlah percobaan terhadap scanner tanpa melakukan scan apapun
- Quantify modification: jumlah modifikasi dari produk yang di scan
- Scanned line items: rata – rata jumlah produk yang di scan perdetik

- Value per second: rata -rata value atau nilai dari produk yang di scan
- Line item per position: rata – rata jumlah item per total nilai dari semua data yang di scan dan tidak dilakukan pembatalan barang
- Fraud: label kelas terjadinya *fraud* atau penipuan.

D. Praproses

Pada Langkah ini, praproses dilakukan terhadap data training dan data testing untuk membuat dataset menjadi berkualitas, adapun teknik praproses yang dilakukan adalah dengan memisahkan dataset dari simbol vertikal “|” pada setiap baris datanya, setelah dipisahkan maka akan melakukan pengecekan apakah ada value atau nilai null atau “NaN”. Namun setelah dilakukan pengecekan tersebut ternyata tidak ada nilai yang berisikan NaN. Langkah selanjutnya dalam praproses adalah menormalisasi dataset agar dapat diproses oleh ekstraksi fitur, pada gambar 1 (a) merupakan dataset mentah dan belum dilakukan praproses apapun sedangkan gambar 1 (b) adalah dataset telah melalui langkah praproses, dan untuk gambar 1 (c) adalah bukti dataset tidak memiliki nilai null atau “NaN”.

trustLevel	totalScanTimeInSeconds	grandTotal	lineItemVoids	scansWithoutRegistration	quantityModifications	...
5	1054	54.7	7	0	3	0.0275142314990512
3	108	27.36	5	2	4	0.12962962962963
3	1516	62.16	3	10	5	0.00857519788918206
6	1791	92.31	8	4	4	0.0161920714684534
5	430	81.53	3	7	2	0.0627906976744186
1	770	11.09	11	5	2	0.0337662337662338
3	294	55.63	2	7	1	0.0374149659863946

Gambar 1 (a) Dataset mentah dalam format csv

df_train.head()						
	trustLevel	totalScanTimeInSeconds	grandTotal	lineItemVoids	scansWithoutRegistration	quantityModifications
0	5	1054	54.70	7	0	3
1	3	108	27.36	5	2	4
2	3	1516	62.16	3	10	5
3	6	1791	92.31	8	4	4
4	5	430	81.53	3	7	2
df_test.head()						
	trustLevel	totalScanTimeInSeconds	grandTotal	lineItemVoids	scansWithoutRegistration	quantityModifications
0	4	467	88.48	4	8	4
1	3	1004	58.99	7	6	1
2	1	162	14.00	4	5	4
3	5	532	84.79	9	3	4
4	5	890	42.16	4	0	0

Gambar 1 (b) Dataset yang sudah melalui praproses

```
df_train.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1879 entries, 0 to 1878
Data columns (total 10 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   trustLevel                            1879 non-null   int64
1   totalScanTimeInSeconds                1879 non-null   int64
2   grandTotal                            1879 non-null   float64
3   lineItemVoids                         1879 non-null   int64
4   scansWithoutRegistration              1879 non-null   int64
5   quantityModifications                1879 non-null   int64
6   scannedLineItemsPerSecond            1879 non-null   float64
7   valuePerSecond                       1879 non-null   float64
8   lineItemVoidsPerPosition              1879 non-null   float64
9   fraud                                1879 non-null   int64
dtypes: float64(4), int64(6)
memory usage: 146.9 KB
```

Gambar 1 (c) dataset tidak memiliki nilai null atau “NaN”

E. Analisis pemilihan Metode & Algoritma

Pada tugas data mining ini, penulis menggunakan beberapa metode untuk melakukan fitur ekstraksi, dan klasifikasi model diantara lain adalah:

1. Principle Components Analysis (PCA)

PCA merupakan salah satu metode untuk melakukan ekstraksi fitur dalam bidang data mining, cara kinerja metode PCA adalah dengan mengurangi dimensi pada fitur yang dipilih, input PCA merupakan data original dari dataset. PCA mencari nilai kombinasi dari fitur inputan lalu meringkasnya, sehingga mengurangi data dimensi aslinya. PCA mampu memaksimalkan nilai varian dan meminimalisir kesalahan dalam rekonstruksi dimensi data dengan melihat jarak diantara satu fitur dengan yang lain.

2. K Nearest Neighbor (KNN)

K Nearest Neighbors (KNN) digunakan untuk melakukan klasifikasi, KNN merupakan algoritma Machine Learning yang dijuluki sebagai algoritma malas atau Lazy Learning, karena KNN merupakan algoritma paling sederhana. KNN menggunakan objek dengan jarak yang terdekat sebagai tetangga dalam ruang fitur untuk membuat model prediksi atau model latih.

3. Random Forest

Random forest merupakan klasifikasi yang cocok dengan dataset dalam jumlah banyak, random forest membuat sebuah model dengan decision tree, di analogikan dengan pohon – pohon di dalam sebuah hutan, dimana masing – masing pohon mengambil sebuah keputusan. Random forest mengambil keputusan dari semua pohon dan mengumpulkannya yang akhirnya mendapatkan hasil klasifikasi.

F. Analisis Parameter dan Eksperimen

Pada tahap ini akan membahas terkait dengan parameter yang digunakan dan hasil dari eksperimen yang telah dilakukan. Setelah praproses langkah selanjutnya adalah memilih fitur dari dataset training yang digunakan. Parameter fitur yang terpilih adalah trust Level hingga line item per position seperti

pada gambar 2 (a). Setelah fitur dari dataset training terpilih selanjutnya adalah membagi dataset training menjadi data latih (train) dan data validasi dengan parameter “test-size = 0.25”. Artinya data train sebanyak 75% dari dataset training dan 25% menjadi milik data validasi, untuk membuat model lebih konsisten maka data train haruslah memiliki jumlah yang lebih banyak, seperti pada gambar 2(b).

```

Feature Extract PCA

from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler

model_features = ['trustLevel', 'totalScanTimeInSeconds', 'grandTotal',
                  'lineItemVoids', 'scansWithoutRegistration', 'quantityModifications',
                  'scannedLineItemsPerSecond', 'valuePerSecond', 'lineItemVoidsPerPosition']

[ ] df_x = df_train.loc[:, model_features].values
    df_y = df_train.loc[:, ['fraud']].values
    dtest_x = df_test.loc[:, model_features].values

```

Gambar 2 (a): Pemilihan fitur dari dataset training.

```

Splitting Data into Training and Validation

[ ] from sklearn.model_selection import train_test_split
    X_train, X_val, y_train, y_val = train_test_split(x, df_y, test_size=0.25, random_state=42)

    print("Jumlah Data Train: ", len(X_train))
    print("Jumlah Data Train: ", len(X_val))

Jumlah Data Train: 1409
Jumlah Data Train: 470

```

Gambar 2 (b): Pembagian dataset training menjadi data train dan data validasi

Selanjutnya masing – masing tipe data dilakukan ekstraksi fitur menggunakan PCA untuk mencari nilai variances terbaik dari fitur yang telah dipilih, PCA akan mereduksi 9 fitur yang telah dipilih sebelumnya menjadi hanya 2 dimensi (2-D) data saja, tujuannya adalah membuat data menjadi lebih mudah untuk di proses pada tahap klasifikasi dan mempersingkat waktu kompleksiti. Hasil ekstraksi fitur PCA dapat dilihat pada gambar 3 (a), dan gambar 3 (b) adalah gambar persebaran nilai fraud dari hasil ekstraksi fitur PCA.

```

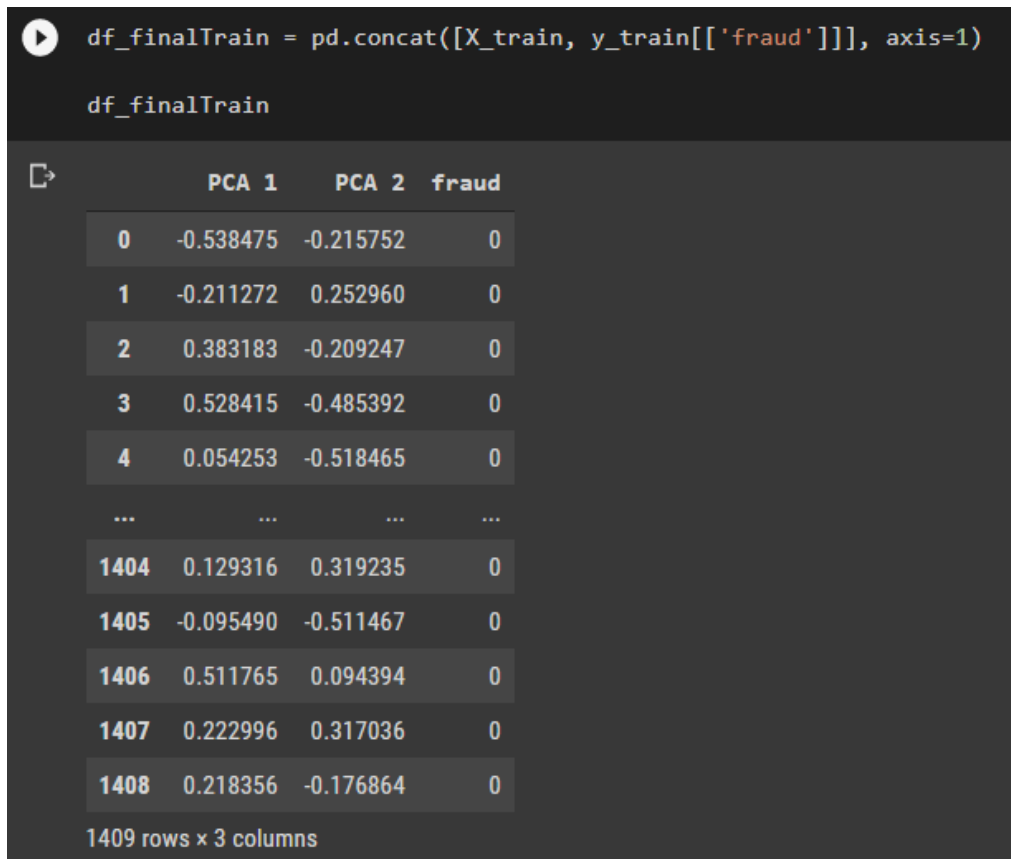
from sklearn.decomposition import PCA

model_pca = PCA(n_components=2)
model_pca.fit(X_train)

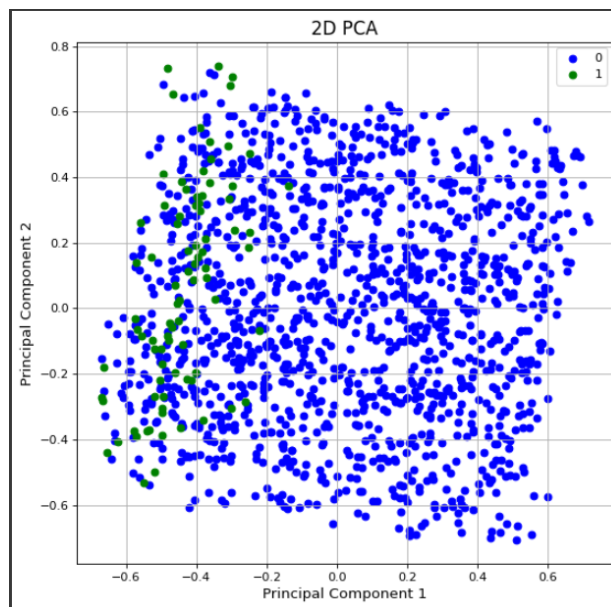
X_train = model_pca.transform(X_train)
X_val = model_pca.transform(X_val)

# 2-D of X_train and 2-D of X_val
print(X_train[:5])
# principalDataframe = pd.DataFrame(data = princiComponents, columns= ['PCA 1', 'PCA 2'])

```



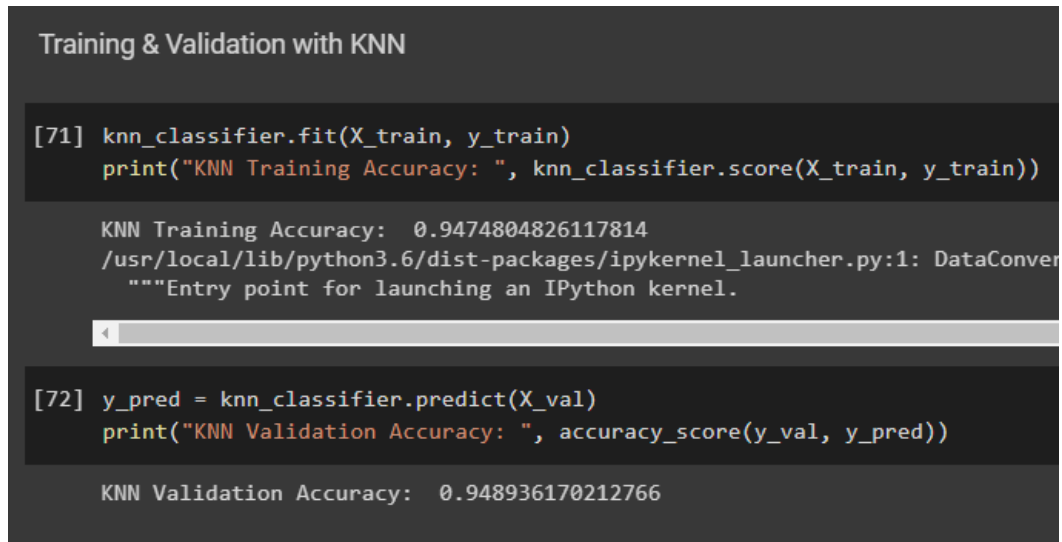
Gambar 3 (a): Hasil ekstraksi fitur dengan cara mereduksi dimensi menggunakan PCA



Gambar 3 (b): Persebaran data fraud dari ekstraksi fitur

Dapat terlihat pada gambar 3 (b) persebaran nilai fraud yang bernilai 0 atau negatif dari kecurangan lebih banyak tersebar dibandingkan dengan fraud yang bernilai 1 atau positif kecurangan. Selanjutnya adalah membuat model training dan validasi dengan menggunakan teknik klasifikasi. Klasifikasi yang digunakan adalah K Nearest Neighbor (KNN) dan Random Forest. Tujuan dari menggunakan 2 teknik yaitu untuk membandingkan performansi model manakah yang terbaik untuk digunakan sebelum melakukan prediksi pada data testing. Akurasi training dan validasi dari KNN masing masing

mendapatkan 94.74% dan 94.89% dengan 445 dari 470 data validasi terprediksi benar. Sedangkan akurasi training dan validasi dari Random Forest masing masing mendapatkan 94.03% dan 94.46% dengan 444 dari 470 data validasi terprediksi benar. Gambar 4 (a) adalah hasil akurasi training dan validasi dari KNN sedangkan gambar 4 (b) adalah hasil akurasi training dan validasi dari Random Forest.



The screenshot shows a Jupyter Notebook interface with a dark theme. The title of the notebook is "Training & Validation with KNN". There are two code cells. The first cell, labeled [71], contains code to fit a KNN classifier and print the training accuracy. The output shows a training accuracy of 0.9474804826117814. The second cell, labeled [72], contains code to predict on validation data and print the validation accuracy. The output shows a validation accuracy of 0.948936170212766.

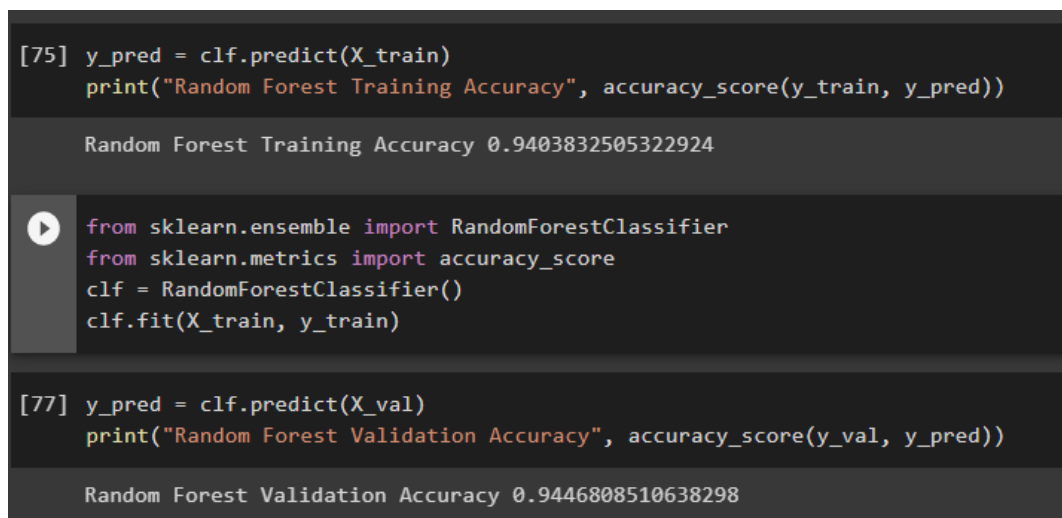
```
[71] knn_classifier.fit(X_train, y_train)
      print("KNN Training Accuracy: ", knn_classifier.score(X_train, y_train))

      KNN Training Accuracy:  0.9474804826117814
      /usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:1: DataConver
      """Entry point for launching an IPython kernel.

[72] y_pred = knn_classifier.predict(X_val)
      print("KNN Validation Accuracy: ", accuracy_score(y_val, y_pred))

      KNN Validation Accuracy:  0.948936170212766
```

Gambar 4 (a): Akurasi training dan validasi dari KNN



The screenshot shows a Jupyter Notebook interface with a dark theme. There are two code cells. The first cell, labeled [75], contains code to fit a Random Forest classifier and print the training accuracy. The output shows a training accuracy of 0.9403832505322924. The second cell, labeled [77], contains code to predict on validation data and print the validation accuracy. The output shows a validation accuracy of 0.9446808510638298.

```
[75] y_pred = clf.predict(X_train)
      print("Random Forest Training Accuracy", accuracy_score(y_train, y_pred))

      Random Forest Training Accuracy 0.9403832505322924

[77] y_pred = clf.predict(X_val)
      print("Random Forest Validation Accuracy", accuracy_score(y_val, y_pred))

      Random Forest Validation Accuracy 0.9446808510638298
```

Gambar 4 (b): Akurasi training dan validasi dari Random Forest

G. Analisis Hasil Eksperimen

Penyelesaian masalah yang diberikan oleh DMC 2019 diharuskan dapat membuat prediksi pada data testing, hasil model prediksi nantinya disimpan pada data test diberikan nama kolom "fraud" atau label "0" dan "1". Dari kedua model klasifikasi yang digunakan dapat dilihat akurasi yang terbaik didapatkan oleh hasil running KNN. Oleh karena itu dalam memprediksi model ketika menggunakan data test akan menggunakan klasifikasi KNN. Fitur dalam data test akan di ekstrak seperti yang dilakukan kepada data train menggunakan PCA. Ketika selesai mengekstraksi fitur, KNN akan memprediksi nilai fraud dengan label "0" atau "1" seperti pada model training yang telah dilakukan, pada gambar 5 (a) akan diperlihatkan tabel hasil dari prediksi model testing KNN. Karena pada gambar 5 (a) tidak terlalu

terlihat nilai label yang mengalami kecurangan atau label “1”, maka gambar 5 (b) akan memperlihatkan nilai index mana saja yang mengandung label fraud = 1, dan pada gambar 5 (c) akan memperlihatkan prediksi tebakan pada file yang berformat CSV.

grandTotal	lineItemVoids	scansWithoutRegistration	quantityModifications	scannedLineItemsPerSecond	valuePerSecond	lineItemVoidsPerPosition	fraud
88.48	4	8	4	0.014989	0.189465	0.571429	0
58.99	7	6	1	0.026892	0.058755	0.259259	0
14.00	4	5	4	0.006173	0.086420	4.000000	0
84.79	9	3	4	0.026316	0.159380	0.642857	0
42.16	4	0	0	0.021348	0.047371	0.210526	0
...
59.10	2	2	0	0.012771	0.075479	0.200000	0
98.90	9	5	4	0.050360	0.355755	0.642857	0
5.41	6	6	4	0.030000	0.018033	0.666667	0
33.97	2	5	3	0.005906	0.022290	0.222222	0
56.97	11	7	2	0.019231	0.039128	0.392857	0

Gambar 5 (a): hasil prediksi yang telah ditambahkan kedalam kolom tabel testing

df_test[df_test["fraud"]==1]

	trustLevel	totalScanTimeInSeconds	grandTotal	lineItemVoids	scansWithoutRegistration	quantityModifications	scannedLineItemsPerSecond	valuePerSecond	lineItemVoidsPerPosition	fraud
10	3	714	94.29	8	7	0	0.016807	0.132059	0.666667	1
14	3	1196	83.77	11	10	0	0.004181	0.070042	2.200000	1
50	3	1087	40.46	1	10	0	0.017479	0.037222	0.052632	1
84	3	1277	26.15	0	8	0	0.008614	0.020478	0.000000	1
102	3	1077	86.33	6	10	0	0.002786	0.080158	2.000000	1
...
497950	3	1709	64.66	7	8	0	0.009362	0.037835	0.437500	1
497968	3	1191	69.48	4	6	0	0.001679	0.058338	2.000000	1
498036	3	1115	12.44	2	7	0	0.008072	0.011157	0.222222	1
498039	3	1377	87.61	6	2	0	0.007988	0.063624	0.545455	1
498100	3	1151	11.37	3	1	0	0.021720	0.009878	0.120000	1

10262 rows x 10 columns

Gambar 5 (b): lokasi indeks mana saja dengan kondisi fraud = 1.

	A	B	C	D	E	F	G	H	I	J	K
1		trustLevel	totalScanTimeInSeconds	grandTotal	lineItemVoids	scansWithoutRegistration	quantityModifications	scannedLineItemsPerSecond	valuePerSecond	lineItemVoidsPerPosition	fraud
2	0	4	467	88.48	4	8	4	0.014989	0.189465	0.571429	0
3	1	3	1004	58.99	7	6	1	0.026892	0.058755	0.259259	0
4	2	1	162	14	4	5	4	0.006173	0.08642	4	0
5	3	5	532	84.79	9	3	4	0.026316	0.15938	0.642857	0
6	4	5	890	42.16	4	0	0	0.021348	0.047371	0.210526	0
7	5	5	1072	12.67	3	4	1	0.01959	0.011819	0.142857	0
8	6	3	259	93.75	0	7	0	0.100386	0.361969	0	0
9	7	2	1528	47.35	2	9	5	0.009817	0.030988	0.133333	0
10	8	6	816	80.89	9	4	0	0.017157	0.09913	0.642857	0
11	9	4	16	31.91	7	7	4	1.3125	1.994375	0.333333	0
12	10	3	714	94.29	8	7	0	0.016807	0.132059	0.666667	1
13	11	5	1077	66.16	5	8	3	0.015785	0.06143	0.294118	0
14	12	4	1301	84.35	3	10	5	0.021522	0.064835	0.107143	0
15	13	3	1429	47.95	8	1	3	0.003499	0.033555	1.6	0
16	14	3	1196	83.77	11	10	0	0.004181	0.070042	2.2	1

Gambar 5 (c): Tabel prediksi data testing yang telah ditambahkan dengan kolom fraud berformatkan CSV.

H. Ringkasan Model

Jadi dalam penyelesaian masalah deteksi fraud untuk mendapatkan hasil prediksi mengenai apakah terjadi fraud atau tidak dapat menggunakan model prediksi KNN dan hasil ekstraksi fitur dari PCA.

Dimana fitur PCA mengandung hasil variansi dari fitur – fitur dataset utama yaitu dari kolkom “trust level” hingga “line item voids” atau 9 kolom.

I. Interpretasi Model

Model yang diperoleh telah memenuhi tujuan yang telah disebutkan pada bagian B yaitu mendeteksi apakah terjadi fraud atau penipuan terhadap daftar transaksi penjualan alat tulis menggunakan scanner otomatis yang terdapat pada data test. Hal ini dikarenakan hasil model KNN dapat memprediksi apakah terjadi fraud atau tidak pada setiap baris dataset nya dan mampu untuk ditambahkan kedalam data test.