

Binary Classification with Convolutional Neural Network Algorithm



Business Understanding

Proyek ini menggunakan data BPJS Hackathon (Fraud Detection) untuk melakukan prediksi dengan data yang telah kita miliki pendekatan yang digunakan yaitu metode Binary Classification with Convolutional Neural Network Algorithm.

A decorative graphic in the bottom-left corner consisting of several overlapping hexagons. The colors include dark teal, medium teal, and light green.

Data Understanding

Collecting data

Collecting data merupakan proses pengumpulan, pengukuran serta analisis data yang digunakan dalam penelitian.



Describe data

potensi terjadinya *fraud* pada klaim pelayanan Rumah Sakit maka set data yang digunakan dengan menggunakan algoritma *Convolutional Neural Network* (CNN) adalah *fraud_detection_train dataset*. *Dataset* ini terdiri dari 53 *variable* dengan total 200217 observasi.

No.	Nama atribut (variabel)	Tipe atribut	Deskripsi
1.	<i>visit_id</i>	Nominal	ID kunjungan
2.	<i>kdkc</i>	Nominal	Kode wilayah kantor cabang BPJS Kesehatan
3.	<i>dati2</i>	Nominal	Kode kabupaten/kota
4.	<i>typeppk</i>	Nominal	Kode tipe Rumah Sakit
5.	<i>jkpst</i>	Binary	Jenis kelamin peserta JKN-KIS
6.	<i>umur</i>	Nominal	Umur peserta saat mendapatkan pelayanan rumah sakit
7.	<i>jnspelsep</i>	Binary	Kode tingkat pelayanan 1 : rawat inap 2 : rawat jalan
8.	<i>los</i>	Numerik	Lama pasien dirawat di rumah sakit
9.	<i>cmg</i>	Numerik	Klasifikasi CMG (<i>Case Mix Group</i>)
10.	<i>severitylevel</i>	Numerik	Tingkat urgensi
11.	<i>diagprimer</i>	Nominal	Diagnosa primer
12.	<i>dx2_..._...</i>	Numerik	Diagnosa sekunder
13.	<i>proc..._...</i>	Nominal	Kode kelompok <i>procedure</i>
14.	<i>label</i>	Binary	<i>Flag Fraud</i> 1 : <i>fraud</i> 0 : tidak <i>fraud</i> .

Validation Data

Pada subbab ini berisi tahapan evaluasi, kelengkapan data dan kualitas data yang digunakan dalam mengerjakan proyek. Terjadinya missing value maupun noise pada data diakibatkan karena terjadinya kesalahan maupun error pada saat melakukan penginputan data.





Data Preparation

Pada tahapan data preparation berikut akan dijabarkan proses menyiapkan data, pemilahan variabel yang akan dianalisis, serta pembersihan data.

Data Selection

Data selection atau *feature* selection digunakan untuk memilih beberapa *feature* untuk membangun model klasifikasi. Proses seleksi dilakukan dengan melakukan penggabungan terhadap *feature* yang terkait menjadi satu selanjutnya memilih *feature* yang akan digunakan sebagai *input feature*

```
1 df_dropna['diagprimer'] = df_dropna['dx2_a00_b99'] + df_dropna['dx2_c00_d48'] + df_dropna['dx2_d50_d89'] + df_dropna['dx2_e00_e90']
```

Kode diatas digunakan untuk menyatukan value dari kolom diagprimer yang sumber dari masing masing tabel value tersebut

```
1 to_drop = ['visit_id', 'dx2_a00_b99', 'dx2_c00_d48', 'dx2_d50_d89', 'dx2_e00_e90', 'dx2_f00_f99', 'dx2_g00_g99', 'dx2_h00_h59']
2 df_dropna.drop(to_drop, inplace=True, axis = 1)
3 df_dropna
```

	kdkc	dati2	typeppk	jkpst	umur	jnspelsep	los	cmg	severitylevel	diagprimer	label
0	1107	150	SB	1	64	2	0	F	0	0	1
1	1303	200	C	0	45	1	0	E	3	5	1
2	1114	172	B	1	34	2	0	Q	0	0	1
3	601	90	SC	0	34	2	0	Q	0	0	1
4	1006	130	B	0	27	2	0	F	0	0	1
...
200212	2102	353	B	1	48	2	0	Z	0	0	0
200213	1308	212	SD	0	1	2	0	Q	0	0	0
200214	201	38	SB	1	3	2	0	Q	0	0	0
200215	1008	128	B	1	52	1	1	J	1	0	0
200216	1016	117	SC	0	4	1	1	K	1	0	0

200217 rows × 11 columns

Data Cleaning

Data Cleaning merupakan proses persiapan data dengan cara menghapus atau memodifikasi data yang salah, tidak akurat, tidak terformat maupun duplikat. Data yang rusak tentunya akan berpengaruh pada kinerja pada sistem.

```
1 df_dropna = df.dropna()  
2 print(df_dropna.shape)
```

```
(200217, 53)
```

```
df_dropna.duplicated().sum()
```

```
0
```

```
[19]: df_dropna.isna().sum()
```

```
[19]: visit_id      0  
      kdkc         0  
      dat12        0  
      typeppk      0  
      jkpst        0  
      umur         0  
      jnspelsep    0  
      los          0  
      cmg          0  
      severitylevel 0  
      diagprimer   0  
      dx2_a00_b99  0  
      dx2_c00_d48  0  
      dx2_d50_d89  0  
      dx2_e00_e90  0  
      dx2_f00_f99  0  
      dx2_g00_g99  0  
      dx2_h00_h59  0  
      dx2_h60_h95  0  
      dx2_i00_i99  0  
      dx2_j00_j99  0  
      dx2_k00_k93  0  
      dx2_l00_l99  0  
      dx2_m00_m99  0  
      dx2_n00_n99  0  
      dx2_o00_o99  0  
      dx2_p00_p96  0  
      dx2_q00_q99  0  
      dx2_r00_r99  0  
      dx2_s00_t98  0  
      dx2_u00_u99  0  
      dx2_v01_y98  0  
      dx2_z00_z99  0  
      proc00_13    0  
      proc14_23    0
```

Data Construct

Mengkonstruksi data merupakan bagian dari Data transformasi yang terdiri dari representasi fitur, menentukan korelasi dan mengintegrasikan data. representasi fitur digunakan untuk mengurangi kompleksitas, meningkatkan akurasi dan memilih fitur optimal.

```
1 df_dropna['jkpst'].replace(to_replace=['L','P'], value = [0,1], inplace = True)
```

Kode diatas digunakan untuk mengubah value kategorikal dari jenis kelamin berupa L dan P menjadi Binary

```
1 import numpy as np
2 # Numeric data type
3 data_num = df_dropna.select_dtypes(include=[np.number])
4
5 # Category data type
6 data_cat = df_dropna.select_dtypes(exclude=[np.number])
7
8 # Get dummies (data transformation)
9 transform_cat = pd.get_dummies(data_cat, prefix_sep='_', drop_first=True)
```

```
1 from numpy.core.defchararray import add
2 data_cat = transform_cat.assign(new=add('', np.arange(1, len(data_cat) + 1).astype(str)))
3 data_num = data_num.assign(new=add('', np.arange(1, len(data_num) + 1).astype(str)))
4 df_dropna = pd.concat([data_cat, data_num], axis=1)
5 df_dropna.drop(['new'], axis=1, inplace=True)
```

Kode diatas digunakan untuk transformasi data pada semua data kategorikal sehingga semua fitur memiliki nilai numerik.

```
1 df_dropna
```

	typeppk_B	typeppk_C	typeppk_D	typeppk_GD	typeppk_HD	typeppk_I1	typeppk_I2	typeppk_I3	typeppk_I4	typeppk_KB	...	cmg_Z	kdkc	dati2	jkp
0	0	0	0	0	0	0	0	0	0	0	...	0	1107	150	
1	0	1	0	0	0	0	0	0	0	0	...	0	1303	200	
2	1	0	0	0	0	0	0	0	0	0	...	0	1114	172	
3	0	0	0	0	0	0	0	0	0	0	...	0	801	90	
4	1	0	0	0	0	0	0	0	0	0	...	0	1008	130	
...
200212	1	0	0	0	0	0	0	0	0	0	...	1	2102	353	
200213	0	0	0	0	0	0	0	0	0	0	...	0	1308	212	
200214	0	0	0	0	0	0	0	0	0	0	...	0	201	38	
200215	1	0	0	0	0	0	0	0	0	0	...	0	1008	128	
200216	0	0	0	0	0	0	0	0	0	0	...	0	1016	117	

200217 rows × 55 columns

Labelling Data

Pada kasus Fraud Detection (Binary Classification) data dibagi menjadi data training dan data validation yang berbeda

```
1 X = df_dropna.drop('label', axis = 1)
2 y = df_dropna['label']
```

Kode diatas merupakan Feature Selection utk menentukan Input maupun Target Features

Data Integration

Pada tahap mengintegrasikan data dilakukan concatenation. Concatenation dapat dianggap sebagai sebuah pendekatan untuk menambahkan baris atau kolom ke data. Pendekatan ini dimungkinkan jika data terbagi menjadi beberapa bagian atau jika dilakukan perhitungan yang ingin ditambahkan ke set data yang sudah tersedia.

```
1 df_dropna = fraud.append(non_fraud, ignore_index = True).dropna()
2 df_dropna
```

C:\Users\ASUS\AppData\Local\Temp\ipykernel_11972\1985386300.py:1: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.concat instead.

```
df_dropna = fraud.append(non_fraud, ignore_index = True).dropna()
```

	typeppk_B	typeppk_C	typeppk_D	typeppk_GD	typeppk_HD	typeppk_I1	typeppk_I2	typeppk_I3	typeppk_I4	typeppk_KB	...	cmg_Z	kdkc	dati2	jk
0	0	0	0	0	0	0	0	0	0	0	...	0	1107	150	
1	0	1	0	0	0	0	0	0	0	0	...	0	1303	200	
2	1	0	0	0	0	0	0	0	0	0	...	0	1114	172	
3	0	0	0	0	0	0	0	0	0	0	...	0	601	90	
4	1	0	0	0	0	0	0	0	0	0	...	0	1006	130	
...
200505	0	0	0	0	0	0	0	0	0	0	...	0	1112	160	
200506	1	0	0	0	0	0	0	0	0	0	...	0	105	3	
200507	0	0	0	0	0	0	0	0	0	0	...	0	2102	354	
200508	0	0	0	0	0	0	0	0	0	0	...	0	1112	168	
200509	0	0	0	0	0	0	0	0	1	0	...	0	701	97	

200510 rows x 55 columns





Modeling

Pada bab ini dijelaskan mengenai pemilihan teknik modeling, menghasilkan test design, membangun model atau membuat pemodelan, dan menilai model yang telah dibangun. Model yang digunakan adalah Binary Classification with Convolutional Neural Network Algorithm

Modelling

Building Test Scenario

Teknik pemodelan yang dilakukan pada penelitian melibatkan penerapan cnn dalam melakukan prediksi jumlah kasus dan unit cost pada sebuah daerah akibat penambahan Rumah Sakit dari 200217 observasi dan 53 variable. Adapun feature yang digunakan pada dataframe terdiri atas kdkc, dati2, typeppk, jkpst, umur, jnspelsep, los, cmg, severitylevel diagprimer untuk input feature serta label yang menjadi target feature.

```
1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=0)
2 X_train.shape, X_test.shape
```

```
1 # Buat objek scaler
2 scaler = StandardScaler()
3 # Sesuaikan scaler dengan data
4 X_train = scaler.fit_transform(X_train)
5 # Mengubah data train dan test
6 X_test = scaler.transform(X_test)
7 y_train = y_train.to_numpy()
8 y_test = y_test.to_numpy()
```

Scaling standardisasi berfokus pada mengubah data mentah menjadi informasi yang dapat digunakan sebelum dianalisis.

```
1 X_train.shape
```

(180459, 54)

```
1 X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], 1)
2 X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)
```


Modelling

Build Model

Mendefenisikan model Convolutional Neural Network

```
1 model = Sequential()
2 model.add(Conv1D(32, 2, activation='relu', input_shape = (54, 1)))
3 model.add(BatchNormalization())
4 model.add(Dropout(0.1))
5
6 model.add(Conv1D(64, 2, activation='relu'))
7 model.add(BatchNormalization())
8 model.add(Dropout(0.2))
9
10 model.add(Flatten())
11 model.add(Dense(64, activation='relu'))
12 model.add(Dropout(0.4))
13
14 model.add(Dense(1, activation='sigmoid'))
15 model.summary()
```

Berikut kode untuk melakukan compile model dan fit model cnn

```
1 model.compile(optimizer='adam', loss = 'binary_crossentropy', metrics=['accuracy'])
```

```
1 history = model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test), verbose=1)
```

Hasil akhir compile model dan fit model

```
Epoch 9/10
5640/5640 [=====] - 36s 6ms/step - loss: 0.6616 - accuracy: 0.5936 - val_loss: 0.6482 - val_accuracy: 0.6238
Epoch 10/10
5640/5640 [=====] - 36s 6ms/step - loss: 0.6602 - accuracy: 0.5928 - val_loss: 0.6483 - val_accuracy: 0.6229
```



Evaluation

Pada bab ini dilakukan tahap Evaluation (Evaluasi) dengan tujuan untuk memprediksi seberapa baik model akhir akan bekerja nantinya sehingga diketahui apakah model tersebut layak digunakan atau tidak dan untuk membantu menemukan model yang paling mewakili pelatihan data

Evaluation

Berikut ditampilkan hasil evaluasi terhadap model yang dikembangkan

```
1 print(classification_report(y_test, y_pred_cnn))
2 print('precision_score:', precision_score(y_test, y_pred_cnn))
3 print('accuracy_score:', accuracy_score(y_test, y_pred_cnn))
4 print('recall_score:', recall_score(y_test, y_pred_cnn))
```

	precision	recall	f1-score	support
0	0.64	0.58	0.61	10059
1	0.61	0.67	0.64	9992
accuracy			0.62	20051
macro avg	0.62	0.62	0.62	20051
weighted avg	0.62	0.62	0.62	20051

```
precision_score: 0.6109589041095891
accuracy_score: 0.6228617026582215
recall_score: 0.6695356285028022
```



Deployment

Pada bab ini akan dijelaskan mengenai perencanaan fase penyebaran atau penggunaan model yang sudah dihasilkan, perencanaan pemantauan dan pemeliharaan

Deployment

- Membuat rencana deployment model
- Melakukan deployment model
- Melakukan rencana pemeliharaan
- Melakukan Pemeliharaan



Thank you

