Laporan Case-Based 1
Mata Kuliah Pembelajaran Mesin
Artificial Neural Network (ANN)

Wandi Yusuf Kurniawan – 1301218601 – IFX-45-GAB – IKN

Program Studi Sarjana Informatika

Fakultas Informatika

Universitas Telkom

Bandung

2022

PERNYATAAN KODE ETIK AKADEMIK UNIVERSITAS TELKOM

Dengan ini saya bersaksi bahwa hasil kerja ini **saya kerjakan sendiri** dan tidak mencontek hasil kerja dari mahasiswa lain dan tidak melakukan kecurangan dalam pengerjaan tugas ini. Jika kesaksian saya ini tidak benar, maka **saya bersedia menerima sanksi diberi minimum nilai E untuk mata kuliah ini dan/atau maksimum untuk semua mata kuliah pada semester ini.**

Wandi Yusuf Kurniawan

Pertama-tama, penulis melakukan import library yang dibutuhkan untuk mengerjakan tugas ini pada Jupyter Notebook menggunakan bahasa Python, diantaranya adalah:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import metrics
import keras
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import GridSearchCV
# from ann_visualizer.visualize import ann_viz
```

IKHTISAR KUMPULAN DATA YANG DIPILIH

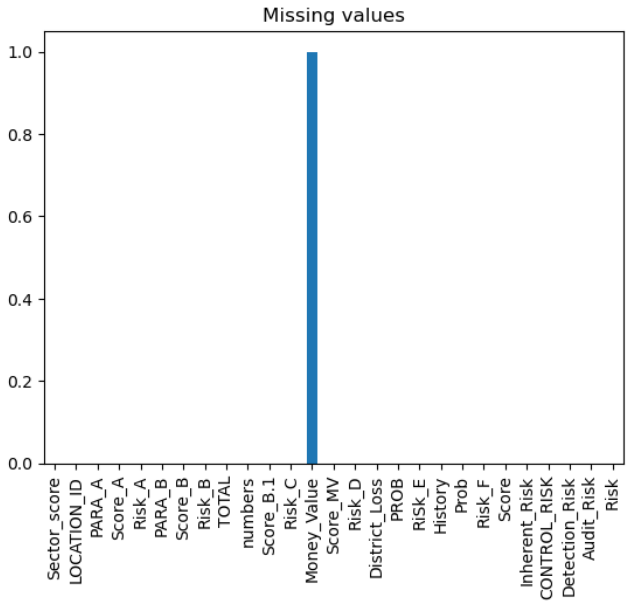
Kemudian inisialisasi variable `data` untuk membaca file .csv yang diambil dari https://archive.ics.uci.edu/ml/datasets/Audit+Data dan lakukan `.head()` untuk memastikan bahwa variable tersebut sudah berisi dataset yang nantinya akan dijadikan pembentukkan model neural network:

	Sector_score	LOCATION_ID	PARA_A	Score_A	Risk_A	PARA_B	Score_B	Risk_B	TOTAL	numbers	...	RiSk_E	History
0	3.89	23	4.18	0.6	2.508	2.50	0.2	0.500	6.68	5.0	...	0.4	0
1	3.89	6	0.00	0.2	0.000	4.83	0.2	0.966	4.83	5.0	...	0.4	0
2	3.89	6	0.51	0.2	0.102	0.23	0.2	0.046	0.74	5.0	...	0.4	0
3	3.89	6	0.00	0.2	0.000	10.80	0.6	6.480	10.80	6.0	...	0.4	0
4	3.89	6	0.00	0.2	0.000	0.08	0.2	0.016	0.08	5.0	...	0.4	0

5 rows × 27 columns

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 776 entries, 0 to 775
Data columns (total 27 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Sector_score    776 non-null    float64
 1   LOCATION_ID     776 non-null    object
 2   PARA_A          776 non-null    float64
 3   Score_A         776 non-null    float64
 4   Risk_A          776 non-null    float64
 5   PARA_B          776 non-null    float64
 6   Score_B         776 non-null    float64
 7   Risk_B          776 non-null    float64
 8   TOTAL           776 non-null    float64
 9   numbers         776 non-null    float64
 10  Score_B.1       776 non-null    float64
 11  Risk_C          776 non-null    float64
 12  Money_Value     775 non-null    float64
 13  Score_MV        776 non-null    float64
 14  Risk_D          776 non-null    float64
 15  District_Loss   776 non-null    int64
 16  PROB            776 non-null    float64
 17  RiSk_E          776 non-null    float64
 18  History         776 non-null    int64
 19  Prob            776 non-null    float64
 20  Risk_F          776 non-null    float64
 21  Score           776 non-null    float64
 22  Inherent_Risk   776 non-null    float64
 23  CONTROL_RISK    776 non-null    float64
 24  Detection_Risk  776 non-null    float64
 25  Audit_Risk      776 non-null    float64
 26  Risk            776 non-null    int64
dtypes: float64(23), int64(3), object(1)
memory usage: 163.8+ KB
```

Cek `.info()` pada data untuk mengetahui informasi kolom, berdasarkan table di bawah terdapat 776 baris dan 27 kolom, salah satu kolom, yaitu `Money_Value`, memiliki satu baris kosong atau null.



RINGKASAN PRA-PEMROSESAN DATA YANG DIUSULKAN

Penulis menghapus dua kolom yang tidak diperlukan, yaitu `'LOCATION_ID'` karena bukan bertipe int/float dan `'TOTAL'` karena penjumlahan dari kolom `'Risk_A'` dan `'Risk_B'`, serta satu baris pada kolom `Money_Value` yang harus diimputasi agar tidak ada nilai `N/A` sehingga jumlah kolomnya berkurang menjadi 25.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 776 entries, 0 to 775
Data columns (total 25 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Sector_score    776 non-null    float64
 1   PARA_A          776 non-null    float64
 2   Score_A         776 non-null    float64
 3   Risk_A          776 non-null    float64
 4   PARA_B          776 non-null    float64
 5   Score_B         776 non-null    float64
 6   Risk_B          776 non-null    float64
 7   numbers         776 non-null    float64
 8   Score_B.1       776 non-null    float64
 9   Risk_C          776 non-null    float64
 10  Money_Value     776 non-null    float64
 11  Score_MV        776 non-null    float64
 12  Risk_D          776 non-null    float64
 13  District_Loss   776 non-null    int64
 14  PROB            776 non-null    float64
 15  RiSk_E          776 non-null    float64
 16  History         776 non-null    int64
 17  Prob            776 non-null    float64
 18  Risk_F          776 non-null    float64
 19  Score           776 non-null    float64
 20  Inherent_Risk   776 non-null    float64
 21  CONTROL_RISK    776 non-null    float64
 22  Detection_Risk  776 non-null    float64
 23  Audit_Risk      776 non-null    float64
 24  Risk            776 non-null    int64
dtypes: float64(22), int64(3)
memory usage: 151.7 KB
```

Penulis melakukan `.describe()` untuk melihat statistic deskriptif apakah kolom tersebut sangat mempengaruhi label, yaitu pada kolom `'Risk'`, dengan mengecek simpangan baku atau std.

	Sector_score	PARA_A	Score_A	Risk_A	PARA_B	Score_B	Risk_B	numbers	Score_B.1	Risk_C	...
count	776.000000	776.000000	776.000000	776.000000	776.000000	776.000000	776.000000	776.000000	776.000000	776.000000	...
mean	20.184536	2.450194	0.351289	1.351029	10.799988	0.313144	6.334008	5.067655	0.223711	1.152964	...
std	24.319017	5.678870	0.174055	3.440447	50.083624	0.169804	30.072845	0.264449	0.080352	0.537417	...
min	1.850000	0.000000	0.200000	0.000000	0.000000	0.200000	0.000000	5.000000	0.200000	1.000000	...
25%	2.370000	0.210000	0.200000	0.042000	0.000000	0.200000	0.000000	5.000000	0.200000	1.000000	...
50%	3.890000	0.875000	0.200000	0.175000	0.405000	0.200000	0.081000	5.000000	0.200000	1.000000	...
75%	55.570000	2.480000	0.600000	1.488000	4.160000	0.400000	1.840500	5.000000	0.200000	1.000000	...
max	59.850000	85.000000	0.600000	51.000000	1264.630000	0.600000	758.778000	9.000000	0.600000	5.400000	...

8 rows × 25 columns

```
Risk_F            32.547120
Audit_Risk        29.098920
History           25.881699
Risk_D            23.385972
Risk_B            22.541967
Money_Value       22.167640
PARA_B            21.505270
Inherent_Risk      9.585590
Risk_A             6.484860
PARA_A             5.371838
Sector_score       1.451625
CONTROL_RISK       0.602668
Risk_E             0.312805
Score_MV           0.301392
Score_B            0.294041
Score_A            0.245496
District_Loss      0.240551
Risk_C             0.217266
Score_B.1          0.129007
Score              0.101007
Prob               0.098383
PROB               0.033093
numbers            0.002723
Detection_Risk     0.000000
dtype: float64
```

Karena jumlah kolomnya masih sangat banyak, maka dilakukanlah reduksi dimensi pada dataset. Pertama-tama, lakukan pemisahan dataset dengan variable `X` (untuk kolom fitur) dan `y` (untuk kolom target). Kemudian hitung nilai `normalized_X = X / X.mean()` dan ambil nilai `.var()` yang lebih dari satu.

Jika digambarkan dalam kolerasi antar kolom, maka hasilnya sebagai berikut:

	Sector_score	PARA_A	Risk_A	PARA_B	Risk_B	Money_Value	Risk_D	History	Risk_F	Inherent_Risk	Audit_Risk
Sector_score	1.0	-0.22	-0.22	-0.13	-0.13	-0.12	-0.12	-0.11	-0.1	-0.17	-0.092
PARA_A	-0.22	1.0	1.0	0.16	0.16	0.45	0.45	0.12	0.1	0.48	0.22
Risk_A	-0.22	1.0	1.0	0.17	0.17	0.45	0.45	0.12	0.11	0.48	0.22
PARA_B	-0.13	0.16	0.17	1.0	1.0	0.13	0.12	0.2	0.2	0.65	0.89
Risk_B	-0.13	0.16	0.17	1.0	1.0	0.13	0.12	0.2	0.2	0.65	0.89
Money_Value	-0.12	0.45	0.45	0.13	0.13	1.0	1.0	0.08	0.07	0.83	0.33
Risk_D	-0.12	0.45	0.45	0.12	0.12	1.0	1.0	0.08	0.07	0.83	0.33
History	-0.11	0.12	0.12	0.2	0.2	0.08	0.08	1.0	0.99	0.19	0.33
Risk_F	-0.1	0.1	0.11	0.2	0.2	0.07	0.07	0.99	1.0	0.17	0.33
Inherent_Risk	-0.17	0.48	0.48	0.65	0.65	0.83	0.83	0.19	0.17	1.0	0.75
Audit_Risk	-0.092	0.22	0.22	0.89	0.89	0.33	0.33	0.33	0.33	0.75	1.0

Terdapat empat pasang kolom yang korelasinya hampir satu (yang berwarna merah pekat), ini menandakan bahwa dataset tersebut masih memiliki redudansi fitur dan akan sangat berpengaruh pada performansi model yang akan dibuat. Penulis menghapus salah satu daripada keempat pasangan fitur, yaitu pada kolom `'Risk_A', 'Risk_B', 'Risk_D',` dan `'Risk_F'`.

	Sector_score	PARA_A	PARA_B	Money_Value	History	Inherent_Risk	Audit_Risk
Sector_score	1.0	-0.22	-0.13	-0.12	-0.11	-0.17	-0.092
PARA_A	-0.22	1.0	0.16	0.45	0.12	0.48	0.22
PARA_B	-0.13	0.16	1.0	0.13	0.2	0.65	0.89
Money_Value	-0.12	0.45	0.13	1.0	0.08	0.83	0.33
History	-0.11	0.12	0.2	0.08	1.0	0.19	0.33
Inherent_Risk	-0.17	0.48	0.65	0.83	0.19	1.0	0.75
Audit_Risk	-0.092	0.22	0.89	0.33	0.33	0.75	1.0

Setelah selesai reduksi fitur dengan menyisakan tujuh fitur yang siap dijadikan sebagai input model, penulis melakukan pemisahan dataset latih dan uji dengan menggunakan `train_test_split` dengan ukuran dataset uji sebesar 20%. Selanjutnya, variable `X_train` dan `X_test` dinormalisasi dengan menggunakan `StandardScaler()` agar memudahkan model dalam penyesuaian bobot antar-node dan antar-layer yang akan dibahas pada bagian selanjutnya.

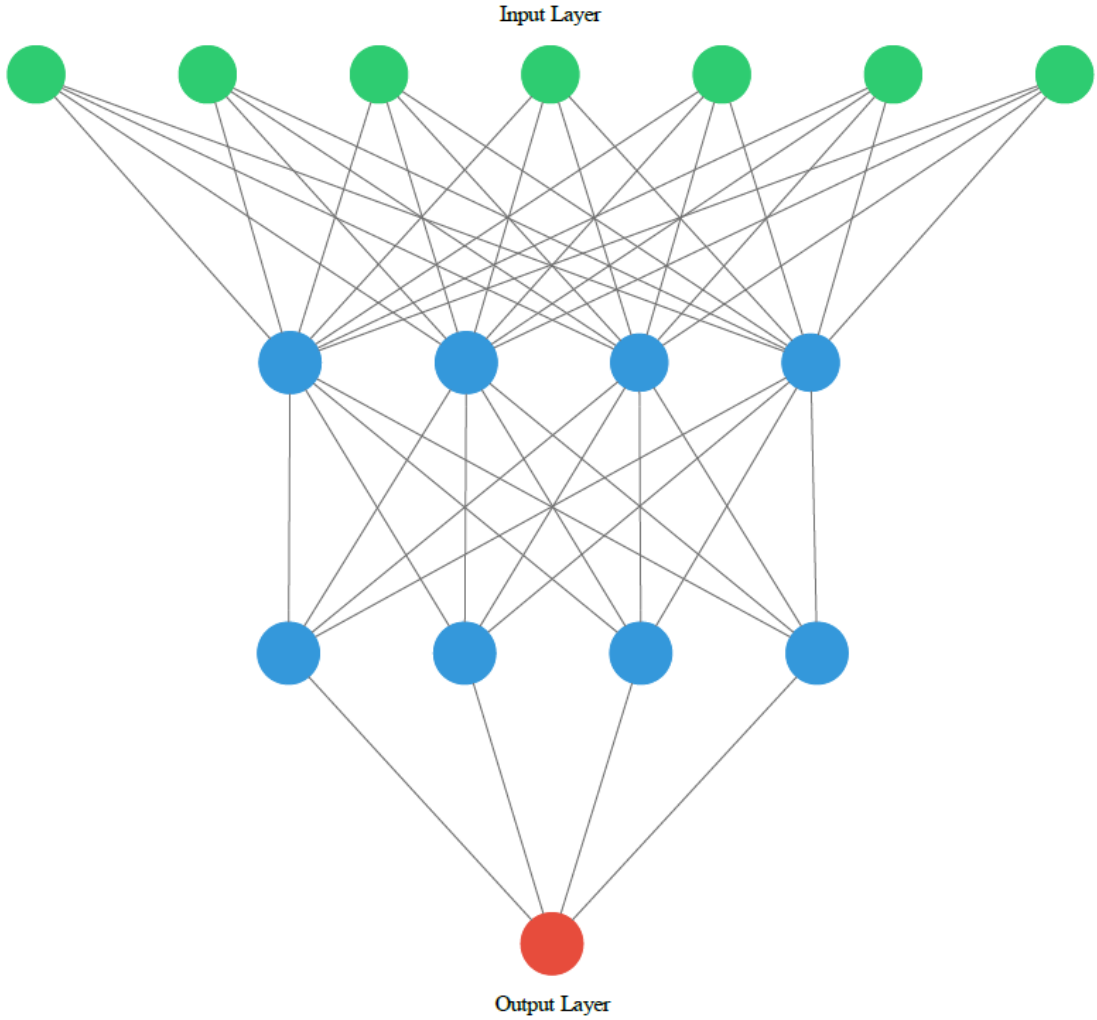
```
scaler = StandardScaler()
X_train_scaled = pd.DataFrame(scaler.fit_transform(X_train))
X_test_scaled = pd.DataFrame(scaler.transform(X_test))
```

```
X_train_scaled array:
            0         1         2         3         4         5         6
0    1.456590 -0.383186 -0.194351 -0.233513 -0.209291 -0.304822 -0.172317
1   -0.666856 -0.453727 -0.191808 -0.230453 -0.209291 -0.305178 -0.172402
2   -0.750676  0.081662 -0.208156 -0.233513 -0.209291 -0.274247 -0.165041
3    1.456590 -0.441066 -0.210881 -0.233513 -0.209291 -0.309688 -0.173475
4   -0.744924 -0.347011 -0.210881 -0.233513 -0.209291 -0.291809 -0.147404
..        ...       ...       ...       ...       ...       ...       ...
615 -0.666856  1.355020  1.138060 -0.020050  4.912830  0.881657  1.415583
616 -0.666856  0.738237  0.218906  0.432700  1.498083  0.470210  0.204326
617 -0.714929 -0.343394 -0.209610 -0.233513 -0.209291 -0.307275 -0.172900
618 -0.750676  0.054531 -0.203433 -0.233513 -0.209291 -0.274998 -0.165220
619 -0.666856  0.011121 -0.129320 -0.088909  1.498083 -0.193991 -0.111761

[620 rows x 7 columns]
X_test_scaled array:
            0         1         2         3         4         5         6
0   -0.750676 -0.345203 -0.190173 -0.225288 -0.209291 -0.296872 -0.170425
1    1.456590 -0.305410 -0.210881 -0.233513 -0.209291 -0.306721 -0.172769
2   -0.729310 -0.316263 -0.204705 -0.233513 -0.209291 -0.289792 -0.145964
3    1.456590 -0.309028 -0.200527 -0.233513 -0.209291 -0.304546 -0.172251
4   -0.744924 -0.160710 -0.210881 -0.233513 -0.209291 -0.281327 -0.139921
..        ...       ...       ...       ...       ...       ...       ...
151 -0.729310 -0.093787 -0.199437 -0.233513 -0.209291 -0.291730 -0.169202
152 -0.750676  0.964331 -0.159837 -0.233513 -0.209291 -0.175677 -0.103046
153 -0.729310 -0.280088 -0.208701 -0.229305 -0.209291 -0.304822 -0.172317
154  1.632448 -0.453727 -0.210881 -0.233513 -0.209291 -0.309965 -0.173540
155 -0.686578 -0.097404  0.338614 -0.220507 -0.209291  0.067268 -0.083780
```

MENERAPKAN ALGORITMA YANG DI PILIH

Algoritma yang saya rancang adalah artificial neural network (ANN) dengan dua hidden layer karena datasetnya hanya berupa fitur dan label saja, bukan merupakan data yang tidak terstruktur seperti gambar atau audio.



```
def keras_classifier(optimizer):
    cf = Sequential()
    cf.add(Dense(units = 4, kernel_initializer = 'uniform', activation = 'relu', input_dim = 7))
    cf.add(Dense(units = 4, kernel_initializer = 'uniform', activation = 'relu'))
    cf.add(Dense(units = 1, kernel_initializer = 'uniform', activation = 'sigmoid'))
    cf.compile(optimizer = optimizer, loss = 'binary_crossentropy', metrics = ['accuracy'])
    return cf
cf = KerasClassifier(build_fn = keras_classifier)
parameters = {'batch_size': [2, 4, 8, 16],
              'epochs': [32, 64],
              'optimizer': ['adam', 'rmsprop']}
gv_search = GridSearchCV(estimator = cf,
                         param_grid = parameters,
                         scoring = 'accuracy',
                         cv = 5)
gv_search = gv_search.fit(X_train_scaled, y_train)
best_param = gv_search.best_params_
best_acc = gv_search.best_score_

print(best_param, best_acc)
```

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 dense_1165 (Dense)           (None, 4)                 32

 dense_1166 (Dense)           (None, 4)                 20

 dense_1167 (Dense)           (None, 1)                 5

=================================================================
Total params: 57
Trainable params: 57
Non-trainable params: 0
```

```
{'batch_size': 2, 'epochs': 64, 'optimizer': 'adam'} 0.9290322580645161
```

ANN ini memiliki 57 parameter yang terdiri dari input sebanyak tujuh kolom fitur hasil reduksi, empat node di setiap hidden layer menggunakan fungsi aktivasi relu, dan satu output berupa kolom label menggunakan fungsi aktivasi sigmoid, kemudian di-compile untuk menghitung nilai loss menggunakan binary_crossentropy dan metrik akurasi yang biasanya akan meningkat pada setiap iterasi.

Hyperparameter yang digunakan ada tiga, yaitu batch_size (membagi dataset train menjadi beberapa bagian), epoch atau jumlah iterasi, dan optimizer untuk proses backpropagation atau penyesuaian bobot antar node atau antar layer.

GridSearchCV (cross-validation) digunakan untuk menentukan parameter terbaik dalam menghasilkan akurasi yang maksimal dengan membagi data latih menjadi 5 bagian, salah satunya digunakan untuk validasi.

Dari 16 kombinasi parameter, 4 dari parameter batch_size dan 2 dari masing-masing parameter epochs dan optimizer, dihasilkan parameter terbaik, yaitu batch_size = 2, epochs = 64, dan optimizer = ‘adam’, dengan akurasi data latih-validasi sebesar 92,9%.

EVALUASI HASIL

Pengukuran performansi model yang penulis lakukan ada empat, yaitu akurasi, presisi, recall, dan F1-score, dengan masing-masing rumus sebagai berikut:

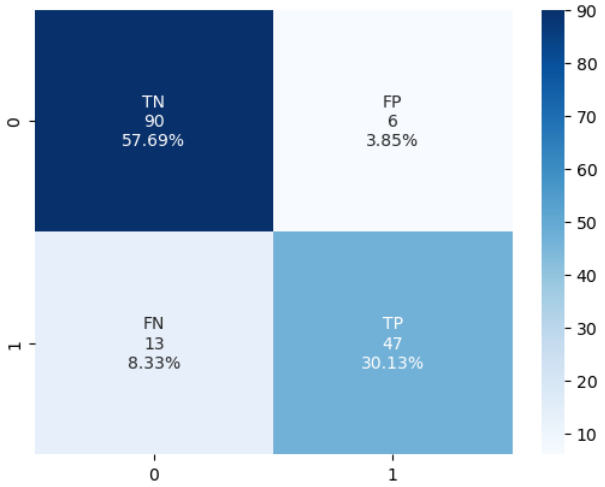
$$Akurasi = \frac{TP + TN}{TP + FN + FP + TN}$$

$$Presisi = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 - score = \frac{2 * Presisi * Recall}{Presisi + Recall}$$

Di mana TP adalah true positive (nilai prediksi dan actual sama-sama 1), FN adalah false negative (nilai prediksi 0, tetapi nilai actual 1), FP adalah false positive (nilai prediksi 1, tetapi nilai actual 0), dan mana TN adalah true negative (nilai prediksi dan actual sama-sama 0).



Setelah memasukkan data uji ke dalam model yang sudah dibuat dengan fungsi `.predict()`, hasil confusion matrix menunjukkan bahwa terdapat 90 data TN, 6 data FP, 13 data FN, dan 47 data TP.

	precision	recall	f1-score	support
0	0.8738	0.9375	0.9045	96
1	0.8868	0.7833	0.8319	60
accuracy			0.8782	156
macro avg	0.8803	0.8604	0.8682	156
weighted avg	0.8788	0.8782	0.8766	156

Dengan akurasi data uji sebesar 87,82% menunjukkan bahwa model tersebut sudah cukup baik dalam memprediksi nilai `'Risk'` dari 7 fitur, walaupun ada sedikit overfitting karena akurasi data latih-validasi sekitar 5,08% lebih baik daripada data uji.

Link Source Code, Laporan, dan Slide Presentasi : https://github.com/onedeetelyu/tugas_machine_learning
Link Video Presentasi : https://youtu.be/MuWaO5dUc6o

LAMPIRAN

Fungsi nilai z untuk StandardScaler()

$$z(x) = \frac{x - \mu}{\sigma}$$

μ rata-rata dari semua nilai x, σ standar deviasi dari semua nilai x

Fungsi RelU

$$f(x) = \max(0, x)$$

Fungsi Sigmoid

$$f(x) = \frac{1}{1 + e^{-x}}$$

Binary Crossentropy Loss

$$Loss = -\frac{1}{n}\sum_{i=1}^{n} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

n jumlah output, y_i nilai output aktual, $\hat{y}_i$ nilai output prediksi.

Optimizer untuk hyperparameter.

```
tf.keras.optimizers.Adam(
    learning_rate=0.001,
    beta_1=0.9,
    beta_2=0.999,
    epsilon=1e-07,
    amsgrad=False,
    name="Adam",
    **kwargs
)
```

```
tf.keras.optimizers.RMSprop(
    learning_rate=0.001,
    rho=0.9,
    momentum=0.0,
    epsilon=1e-07,
    centered=False,
    name="RMSprop",
    **kwargs
)
```

learning_rate: Tingkat pembelajaran untuk penyesuaian bobot antar-node dan antar-layer. Nilai defaultnya **0,001**.
beta_1: Laju peluruhan eksponensial untuk estimasi momen pertama. Nilai defaultnya **0.9**.
beta_2: Laju peluruhan eksponensial untuk estimasi momen kedua. Nilai defaultnya **0,999**.
epsilon: Sebuah konstanta kecil untuk stabilitas numerik. Epsilon ini adalah *"topi epsilon"* di paper *"Kingma dan Ba"* (di rumus sebelum bagian 2.1), bukan epsilon di algoritma 1. Nilai defaultnya **1e-7 (0,0000001)**.
amsgrad: Apakah akan menerapkan varian AMsGrad dari algoritma ini dari paper *"On the Convergence of Adam and Beyond"*. Nilai defaultnya **False**.
rho: Faktor pemotongan untuk gradien sebelumnya dan yang akan datang. Nilai defaultnya **0.9**.
momentum: Nilai skalar atau skalar Tensor. Nilai defaultnya **0,0**.
centered: Boolean. Jika bernilai True, gradien dinormalisasi dengan estimasi varians gradien; jika bernilai Salah, pada momen kedua yang tidak terpusat. Menyetel ini ke True dapat membantu pelatihan, tetapi sedikit memakan waktu komputasi dan memori. Nilai defaultnya **False**.

Sumber: https://keras.io/api/optimizers/adam/, https://keras.io/api/optimizers/rmsprop/.