

LAPORAN PRAKTIKUM MINGGU KE-11 PRAKTIKUM KECERDASAN BUATAN				Kelas : E NIM 2206159 Nama Lengkap : Adnan Fawwaz Maulana	
Intruksi:					
1. Unduh dataset berikut pada repository online (https://archive.ics.uci.edu/datasets), pilih sesuai dengan angka terakhir pada NIM masing-masing: 1) Car Evaluation; 2) Glass Identification; 3) Thyroid Disease; 4) Liver Disorders; 5) Real Estate Valuation; 6) Wholesale customers; 7) Balance Scale; 8) Website Phishing; 9) Auction Verification; 10) Parking Birmingham. (NIM = 0)					
2. Ubah file hasil download menjadi file csv/ excel yang dapat dibaca oleh tools yang akan digunakan!					
3. Lakukan Explanatory Data Analysis untuk mengetahui kondisi dataset yang anda miliki!					
4. Tentukan pekerjaan yang akan anda lakukan, pilih salah satu: a) clustering; b) classification; c) regression; atau d) forecasting.re					
5. Buatlah model learning dengan memilih satu algoritma yang dapat digunakan untuk jenis pekerjaan yang anda pilih.					
6. Lakukan evaluasi terhadap model yang anda buat!					
7. Sebagai panduan silakan lihat hasil diskusi kelompok pada Tugas L9 Kelas Teori minggu sebelumnya!					
Nama Dataset & link unduh		Auction Verification https://archive.ics.uci.edu/dataset/713/auction+verification			
Jumlah Baris Data		2043 rows x 9 columns			
Fitur	Nama Fitur			Type Data	
	process.b1.capacity process.b2.capacity process.b3.capacity process.b4.capacity property.price property.product property.winner verification.result verification.time			Integer integer integer integer integer integer integer True, False Date	
Class (label/ target) & typedata		Label/Integer			
Kondisi Dataset				Cara Mengatasi (Preprocessing Data)	
Duplikasi Baris Data (√)		Yes	√ No	Menghapus baris duplikat menggunakan drop_duplicates() dari Pandas.	
Missing Value (√)		Yes	√ No	Mengisi missing values atau menghapus baris/kolom yang mengandung missing values, tergantung pada jumlah dan pentingnya data yang hilang dnegan menggunakan # Mengisi missing values dengan nilai rata-rata kolom dataset.fillna(dataset.mean(), inplace=True) # Atau menghapus baris dengan missing values dataset.dropna(inplace=True)	
Outlier (√)		Yes	√ No	Menggunakan teknik seperti IQR untuk menghapus atau menangani outlier dengan menggunakan	
Normalisasi (√)	√	Yes	No	Menggunakan StandardScaler atau MinMaxScaler untuk menormalkan data.	
Perubahan Tipe Daya(√)		Yes	√ No	Mengonversi tipe data yang tidak sesuai menggunakan dataset['some_column'] = dataset['some_column'].astype(float)	
Imbalanced Dataset(√)		Yes	√ No	Menggunakan teknik oversampling atau undersampling, seperti SMOTE (Synthetic Minority Over-sampling Technique) Menggunakan from imblearn.over_sampling import SMOTE smote = SMOTE() x_res,y_res = smote.fit_resample(x,y)	

Reduksi Fitur (√)		Yes	√	No	Menggunakan PCA (Principal Component Analysis) atau metode seleksi fitur lainnya.				
Seleksi Fitur (√)		Yes	√	No	Menggunakan metode seperti SelectKBest atau RFE (Recursive Feature Elimination).				
Kondisi lainnya:									
Jenis Pekerjaan yang dipilih (√)		Clustering	√	Classification		Regresion		Forecasting	
Algoritma yang digunakan	Algoritma Classification								
Spliting Data (√)		60:40	√	70:30		80:20		90:20	
k-fold Cross Validation (√)		K=3		K=5		K=7		K=10	
Ukuran Evaluasi Model	rocess.b1.capacity		process.b2.capacity		process.b3.capacity		property.price		
Nilai Evaluasi	0.5776		0.5888		0.6883		0.6728		
Jumlah Ujicoba	3								

BUKTI EKSPERIMEN - masukan gambar hasil *screenshot* yang menampilkan bagian:

1. Tampilan Dataset

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	process.b	process.b	process.b	process.b	property.f	property.f	property.f	verification	verification	time									
2	0	0	2	1	59	1	0	FALSE	163.3167										
3	0	0	2	1	59	2	0	FALSE	200.86										
4	0	0	2	1	59	4	0	FALSE	154.8889										
5	0	0	2	1	59	6	0	FALSE	108.64										
6	0	0	2	1	60	1	0	TRUE	85.46667										
7	0	0	2	1	60	1	3	TRUE	82										
8	0	0	2	1	60	1	4	FALSE	154.25										
9	0	0	2	1	60	2	0	TRUE	80.95										
10	0	0	2	1	60	2	3	TRUE	78										
11	0	0	2	1	60	2	4	FALSE	197.01										
12	0	0	2	1	60	4	0	TRUE	80.25										
13	0	0	2	1	60	4	3	TRUE	77.91667										
14	0	0	2	1	60	4	4	FALSE	144.6944										
15	0	0	2	1	60	6	0	TRUE	88.12										
16	0	0	2	1	60	6	3	TRUE	78.84										
17	0	0	2	1	60	6	4	TRUE	78.15										
18	0	0	2	1	69	3	0	FALSE	112.05										
19	0	0	2	1	69	5	0	FALSE	322.0556										
20	0	0	2	1	70	3	0	TRUE	87.1										

2. Explanatory Data Analysis

```
np.set_printoptions(suppress=True)
```

```
dataset = pd.read_csv('data.csv')
```

```
print(dataset)
```

```
↵
   process.b1.capacity  process.b2.capacity  process.b3.capacity  \
0                    0                    0                    2
1                    0                    0                    2
2                    0                    0                    2
3                    0                    0                    2
4                    0                    0                    2
...                  ...                  ...                  ...
2038                  2                    3                    2
2039                  2                    3                    2
2040                  2                    3                    2
2041                  2                    3                    2
2042                  2                    3                    2

   process.b4.capacity  property.price  property.product  property.winner  \
0                    1                59                1                0
1                    1                59                2                0
2                    1                59                4                0
3                    1                59                6                0
4                    1                60                1                0
...                  ...              ...              ...              ...
2038                  1                90                5                0
2039                  1                90                5                1
```

```
✓ 0d ↵
   2039          1          90          5          1
   2040          1          90          5          2
   2041          1          90          5          3
   2042          1          90          5          4
```

```
   verification.result  verification.time
0                  False          163.316667
1                  False          200.860000
2                  False          154.888889
3                  False          108.640000
4                   True           85.466667
...                  ...              ...
2038                  True           82.425000
2039                  True          1316.983333
2040                  False          9365.450000
2041                  False          8474.025000
2042                  True           82.008333
```

```
[2043 rows x 9 columns]
```

3. Preprocessing Data

```
np.set_printoptions(suppress=True)
```

```
dataset = pd.read_csv('data.csv')
```

```
print(dataset)
```

```
↵
   process.b1.capacity  process.b2.capacity  process.b3.capacity  \
0                    0                    0                    2
1                    0                    0                    2
2                    0                    0                    2
3                    0                    0                    2
4                    0                    0                    2
...                  ...                  ...                  ...
2038                  2                    3                    2
2039                  2                    3                    2
2040                  2                    3                    2
2041                  2                    3                    2
2042                  2                    3                    2

   process.b4.capacity  property.price  property.product  property.winner  \
0                    1                59                1                0
1                    1                59                2                0
2                    1                59                4                0
3                    1                59                6                0
4                    1                60                1                0
...                  ...              ...              ...              ...
2038                  1                90                5                0
2039                  1                90                5                1
```

```
✓ 0d ↵
   2039          1          90          5          1
   2040          1          90          5          2
   2041          1          90          5          3
   2042          1          90          5          4
```

```
   verification.result  verification.time
0                   False          163.316667
1                   False          200.860000
2                   False          154.888889
3                   False          108.640000
4                    True           85.466667
...                  ...              ...
2038                  True           82.425000
2039                  True          1316.983333
2040                  False          9365.450000
2041                  False          8474.025000
2042                  True           82.008333
```

```
[2043 rows x 9 columns]
```

4. Pemodelan

5. Evaluasi Model

```
classifier = Sequential()

classifier.add(Dense(units = 10, input_dim = 4, kernel_initializer = 'uniform', activation = 'relu'))
classifier.add(Dense(units = 6, kernel_initializer = 'uniform', activation = 'relu'))
classifier.add(Dense(units = 1, kernel_initializer = 'uniform', activation = 'sigmoid'))

classifier.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])

modelANN = classifier.fit(xTrain, yTrain, batch_size = 10, epochs = 10)
```

```
Epoch 1/10
143/143 [=====] - 3s 4ms/step - loss: 0.6883 - accuracy: 0.5776
Epoch 2/10
143/143 [=====] - 1s 3ms/step - loss: 0.6728 - accuracy: 0.5888
Epoch 3/10
143/143 [=====] - 1s 4ms/step - loss: 0.6660 - accuracy: 0.5888
Epoch 4/10
143/143 [=====] - 1s 5ms/step - loss: 0.6635 - accuracy: 0.5888
Epoch 5/10
143/143 [=====] - 1s 6ms/step - loss: 0.6625 - accuracy: 0.5888
Epoch 6/10
143/143 [=====] - 1s 6ms/step - loss: 0.6622 - accuracy: 0.5888
Epoch 7/10
143/143 [=====] - 0s 2ms/step - loss: 0.6611 - accuracy: 0.5888
Epoch 8/10
143/143 [=====] - 0s 2ms/step - loss: 0.6606 - accuracy: 0.5888
Epoch 9/10
143/143 [=====] - 0s 2ms/step - loss: 0.6604 - accuracy: 0.5888
Epoch 10/10
143/143 [=====] - 0s 2ms/step - loss: 0.6602 - accuracy: 0.5888
```