header%20ipynb.png

Hands-On

Hands-On ini digunakan pada kegiatan Microcredential Associate Data Scientist 2021

Tugas Mandiri Pertemuan 16

Pertemuan 16 (enambelas) pada Microcredential Associate Data Scientist 2021 menyampaikan materi mengenai Membangun model: Evaluasi. silakan Anda kerjakan Latihan 1 s/d 5. Output yang anda lihat merupakan panduan yang dapat Anda ikuti dalam penulisan code :)

Jawab pertanyaan di bawah ini dengan bahasa masing-masing?

1. Apa perbedaan antara data latih, data validasi, dan data test?
2. Bagaimana cara kita menilai performa suatu model?
3. Apa itu Confusion Matrix? Jelaskan secara lengkap!
4. Apa itu Classification Report dari sklearn?

Jawab:

1. Perbedaan data latih, data validasi dan data test
- Data latih : Bagian dari kumpulan data tempat model dilatih
- Data validasi : Bagian dari dataset untuk mengevaluasi model selama tahap membangun model
- Data test : Bagian dari kumpulan data untuk mengevaluasi kinerja model keseluruhan akhir

1. Menilai performa suatu model salah satu teknik yang dapat digunakan untuk mengukur kinerja suatu model adalah confusion matrix dengan menghitung nilai akurasinya.

2. Confusion matrix adalah sebuah tabel yang sering digunakan untuk mengukur kinerja dari model klasifikasi di machine learning. Confusion matrix menampilkan dan membandingkan nilai aktual atau nilai sebenarnya dengan nilai hasil prediksi model yang dapat digunakan yakni seperti Accuracy (akurasi), Precision, Recall, dan F1-Score atau F-Measure.

3. Classification Report digunakan untuk menampilkan hasil dari perhitungan confusion matrix (akurasi, Recall, dan F1-Score)

Kali ini kita akan menggunakan data untuk memprediksi kelangsungan hidup pasien yang telah mengalami operasi payudara. Dengan informasi yang dimiliki terkait pasien, kita akan membuat model untuk memprediksi apakah pasien akan bertahan hidup dalam waktu lebih dari 5 tahun atau tidak.

Lebih Lengkapnya kalian bisa membaca informasi tentang dataset di link berikut: https://raw.githubusercontent.com/jbrownlee/Datasets/master/haberman.names

Buat model Klasifikasi (Model/Algoritma Bebas) untuk memprediksi status pasien dengan ketentuan sebagai berikut:

1. Bagi kedua data ini menjadi data training dan data test dengan test_size=0.25.
2. Pelajar tentang metrics roc_auc_score kemudian buatlah model dan evaluasi dengan menggunakan teknik cross-validation dengan scoring 'roc_auc'. Baca https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html untuk menggunakan metric roc_auc saat cross-validation.
3. Berapa score rata2 dari model dengan teknik cross-validation tersebut?
4. Prediksi data test dengan model yang telah kalian buat!
5. Bagaimana hasil confusion matrix dari hasil prediksi tersebut?
6. Bagaimana classification report dari hasil prediksi tersebut?
7. Seberapa baik model anda dalam memprediksi seorang pasien mempunyai status positive?
8. Seberapa baik model anda dalam memprediksi seorang pasien mempunyai status negatif?

Load Dataset

```
# import library pandas
import pandas as pd

# Load Dataset
url =
'https://raw.githubusercontent.com/jbrownlee/Datasets/master/haberman.
csv'
list_cols = ['Age', "Patient's Years", "N_positive_ax",
"survival_status"]
df = pd.read_csv(url, names=list_cols)

# tampilkan 5 baris awal dataset dengan function head()
df.head()
```

```
   Age  Patient's Years  N_positive_ax  survival_status
0   30               64              1                1
1   30               62              3                1
2   30               65              0                1
3   31               59              2                1
4   31               65              4                1
```

```
# hitung jumlah masing" data pada kolom survival_status
df['survival_status'].value_counts()
```

```
1    225
2     81
Name: survival_status, dtype: int64
```

Build Model

```
#import library train test split dan cross val
from sklearn.model_selection import train_test_split, cross_val_score

#import library Logistic regression
from sklearn.linear_model import LogisticRegression

#import library roc auc score
from sklearn.metrics import roc_auc_score

#import library scale
from sklearn.preprocessing import scale

#import library numpy
import numpy as np

## pemisahan feature dan target (data target : 'survival_status')
X = df.drop('survival_status', axis = 1)
Xs = scale(X)
y = df['survival_status']

NO 1
## pemisahan variabel test dan train dari data Xs dan y
# test size= 25%, random state = 42, dan stratify = y
X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.25,random_state=42)

## pembuatan objek model
model_logReg = LogisticRegression(random_state = 42)

## latih model
model_logReg.fit(X_train, y_train)

## prediksi.
y_predict = model_logReg.predict(X_test)

NO 2
## menghitung cross_val_score dengan scoring = 'roc_auc'
## parameter cv = 10
score = cross_val_score(model_logReg, X, y, scoring = 'roc_auc', cv =
10)
print(score)

[0.44021739 0.80978261 0.67391304 0.69021739 0.70380435 0.79292929
 0.875      0.62784091 0.67613636 0.61363636]

NO 3
# cetak rata-rata nilai rata-rata auc score
score.mean()
```

```
0.6903477711901624
```

NO 4

```
# Prediksi data test dengan model yang telah kalian buat
auc_score = roc_auc_score(y_test, y_predict)
auc_score
```

```
0.5681818181818182
```

NO 5

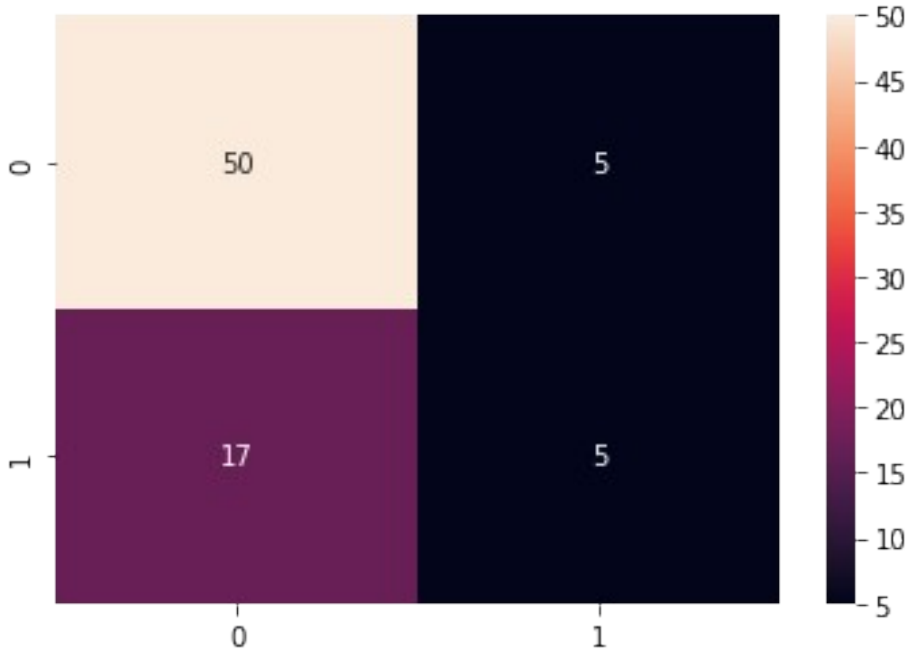
```
# import library confusion matrix dan classification report
from sklearn.metrics import classification_report, confusion_matrix

# apply confusion matrix dan cetak nilai confusion matrix
cm = confusion_matrix(y_test, y_predict, labels = (1,2))
cm
```

```
array([[50,  5],
       [17,  5]])
```

```
# visualisasikan nilai confusion matrix ke dalam diagram heatmap
import seaborn as sns
import matplotlib.pyplot as plt

ax= plt.subplot()
sns.heatmap(cm, annot=True, fmt='g', ax=ax);
```



NO 6

```
# cetak nilai classification_report
print(classification_report(y_test, y_predict))
```

	precision	recall	f1-score	support
1	0.75	0.91	0.82	55
2	0.50	0.23	0.31	22
accuracy			0.71	77
macro avg	0.62	0.57	0.57	77
weighted avg	0.68	0.71	0.67	77

NO 7

- Bagaimana hasil confusion matrix dari hasil prediksi tersebut? *jawab disini*
- Bagaimana classification report dari hasil prediksi tersebut? *jawab disini*
- Seberapa baik model anda dalam memprediksi seorang pasien mempunyai status positive? dari hasil classification_report diatas *jawab disini*
- Seberapa baik model anda dalam memprediksi seorang pasien mempunyai status negatif? dari hasil classification_report diatas *jawab disini*
- Hasil Confusion Matrix : dari confusion matrix diatas dari 10 positif, model memprediksi ada 50 diprediksi negatif, dan dari 67 negatif, model memprediksi ada 17 diprediksi positif.
- Hasil classification report : dari classification report menunjukkan pada hasil tinggi pada baris 1 dengan presisi 0.75, recall 0.91, f1-score 0.82, dan support 55.
- prediksi positif 17
- prediksi negatif 50

Jelaskan dengan bahasa sendiri!

1. Apa itu Bias dan Variance?
2. Apa itu Overfitting dan Underfitting?
3. Apa yang bisa kita lakukan untuk mengatur kompleksitas dari model?
4. Bagaimana model yang baik?
5. Kapan kita menggunakan GridSearchcv dan kapan menggunakan RandomizedSearchCV?

Jawab

1. Bias : teknik algoritma machine learning untuk melakukan penyederhanaan dalam mempelajari kumpulan data dengan tidak mempertimbangkan semua informasi. Variance : variabilitas dalam prediksi model

2. Overfitting : penyimpangan data dari kenyataan disebabkan karena kesalahan pengukuran, faktor acak yang berpengaruh, variabel yang tidak teramati, dan

korelasi sampah Underfitting : masalah yang berlawanan, di mana model gagal mengenali kompleksitas nyata dalam data

3. Untuk mengatur kompleksitas dari model denan meningkatkan fitur masukan yang akan menyebabkan overfitting
4. Model yang baik adalah model yang bisa menjelaskan data tanpa terpengaruh oleh data noise
5. Grid Search dapat digunakan jika data bekerja dengan jumlah kecil pada hyperparameter. Namun, jika jumlah parameter sangat tinggi dan besarnya pengaruh tidak seimbang, pilihan yang lebih baik adalah menggunakan Pencarian Acak.

---

1. Bagi kedua data berikut ini menjadi data training dan data test dengan test_size=0.25.
2. Import library KNN dan GridSearchCV.
3. Gunakan algoritma KNN dan fungsi GridSearchCV untuk hyperparameter tuning dan model selection.
4. jumlah fold bebas!, gunakan scoring 'roc_auc'
5. Definisikan kombinasi hyperparameter untuk model selection dengan GridSearchCV. kombinasi Hyperparameter bebas, baca lagi dokumentasi KNN di link berikut https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html untuk memahami lagi jenis2 hyperparameter di algorithma KNN.
6. Latih model terhadap data training.
7. Apa hyperparameter terbaik untuk kombinasi hyperparameter kalian?
8. Berapa score validasi terbaik dari model tersebut?
9. Prediksi probabilitasi output dari model yang telah di buat terhadap data test. note : gunakan method .predict_proba() untuk menghasilkan output probabilitas
10. Berapa nilai score roc_auc untuk data test? (y_predict)
11. Apakah model anda termasuk baik, overtting, atau underfitting?

Load Dataset

```
# import library pandas
import pandas as pd

# Load Dataset
url =
```

```
'https://raw.githubusercontent.com/jbrownlee/Datasets/master/haberman.
csv'
list_cols = ['Age', "Patient's Years", "N_positive_ax",
"survival_status"]
df2 = pd.read_csv(url, names=list_cols)

# tampilkan 5 baris awal dataset dengan function head()
df2.head()
```

	Age	Patient's Years	N_positive_ax	survival_status
0	30	64	1	1
1	30	62	3	1
2	30	65	0	1
3	31	59	2	1
4	31	65	4	1

```
# hitung jumlah masing" data pada kolom survival_status
df2['survival_status'].value_counts()
```

```
1    225
2     81
Name: survival_status, dtype: int64
```

```
NO 1
# 1. pembagian variabel train dan test
# test size= 25%, random state = 42, dan stratify = y
X = df2.drop('survival_status', axis = 1)
y = df2['survival_status']

X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.25,random_state=42)

NO 2
# 2. import library KNN dan GridSearchCv
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GridSearchCV

NO 3-6
# 3. tuning hyperparameter dengan GridSearchCV (parameter cv=10)
## build model KNN
model_knn = KNeighborsClassifier()
param_grid = {'n_neighbors' : np.arange(3,51), 'weights' :
['uniform','distance']}
gscv = GridSearchCV(model_knn, param_grid, scoring='roc_auc', cv = 10)
gscv.fit(X_train, y_train)

GridSearchCV(cv=10, estimator=KNeighborsClassifier(),
             param_grid={'n_neighbors': array([ 3,  4,  5,  6,  7,  8,
9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
       20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
36,
```

```
       37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50]),
                         'weights': ['uniform', 'distance']},
             scoring='roc_auc')
```

NO 7

```
# 7. parameter terbaik
gscv.best_params_
```

```
{'n_neighbors': 20, 'weights': 'uniform'}
```

NO 8

```
# 8. score validasi terbaik
gscv.best_score_
```

```
0.7268627450980393
```

NO 9

```
# 9. prediksikan probabilitas masing-masing data test
y_predict = gscv.predict_proba(X_test)
y_predict
```

```
array([[0.9 , 0.1 ],
       [0.95, 0.05],
       [0.8 , 0.2 ],
       [0.95, 0.05],
       [0.7 , 0.3 ],
       [0.45, 0.55],
       [0.85, 0.15],
       [0.75, 0.25],
       [0.4 , 0.6 ],
       [0.8 , 0.2 ],
       [0.9 , 0.1 ],
       [0.75, 0.25],
       [0.7 , 0.3 ],
       [0.35, 0.65],
       [0.9 , 0.1 ],
       [0.9 , 0.1 ],
       [0.9 , 0.1 ],
       [0.9 , 0.1 ],
       [0.5 , 0.5 ],
       [0.8 , 0.2 ],
       [0.8 , 0.2 ],
       [0.9 , 0.1 ],
       [0.85, 0.15],
       [0.95, 0.05],
       [0.75, 0.25],
       [0.9 , 0.1 ],
       [1.  , 0.  ],
       [0.8 , 0.2 ],
       [0.65, 0.35],
       [0.65, 0.35],
```

```
       [0.95, 0.05],
       [0.45, 0.55],
       [0.85, 0.15],
       [0.75, 0.25],
       [0.85, 0.15],
       [0.7 , 0.3 ],
       [0.7 , 0.3 ],
       [0.75, 0.25],
       [0.55, 0.45],
       [0.9 , 0.1 ],
       [0.4 , 0.6 ],
       [0.7 , 0.3 ],
       [0.4 , 0.6 ],
       [0.85, 0.15],
       [1.  , 0.  ],
       [0.8 , 0.2 ],
       [0.9 , 0.1 ],
       [0.7 , 0.3 ],
       [0.85, 0.15],
       [0.8 , 0.2 ],
       [0.4 , 0.6 ],
       [0.85, 0.15],
       [0.85, 0.15],
       [0.75, 0.25],
       [1.  , 0.  ],
       [0.45, 0.55],
       [0.65, 0.35],
       [0.8 , 0.2 ],
       [0.65, 0.35],
       [1.  , 0.  ],
       [0.9 , 0.1 ],
       [0.65, 0.35],
       [0.75, 0.25],
       [0.75, 0.25],
       [0.45, 0.55],
       [0.9 , 0.1 ],
       [0.5 , 0.5 ],
       [0.75, 0.25],
       [1.  , 0.  ],
       [0.55, 0.45],
       [0.75, 0.25],
       [0.7 , 0.3 ],
       [0.4 , 0.6 ],
       [0.8 , 0.2 ],
       [0.85, 0.15],
       [0.9 , 0.1 ],
       [0.95, 0.05]])

# nilai rata-rata probabilitas data test
y_predict.mean()
```

```
0.5
```

NO 10

```
# 10. nilai score roc_auc
kurang_5th = gscv.predict_proba(X_test)[:,1]
print(kurang_5th)
```

```
[0.1  0.05 0.2  0.05 0.3  0.55 0.15 0.25 0.6  0.2  0.1  0.25 0.3  0.65
 0.1  0.1  0.1  0.1  0.5  0.2  0.2  0.1  0.15 0.05 0.25 0.1  0.   0.2
 0.35 0.35 0.05 0.55 0.15 0.25 0.15 0.3  0.3  0.25 0.45 0.1  0.6  0.3
 0.6  0.15 0.   0.2  0.1  0.3  0.15 0.2  0.6  0.15 0.15 0.25 0.   0.55
 0.35 0.2  0.35 0.   0.1  0.35 0.25 0.25 0.55 0.1  0.5  0.25 0.   0.45
 0.25 0.3  0.6  0.2  0.15 0.1  0.05]
```

NO 11

Jawab

Baik

1. Ulangi tahap di atas (soal 4, no 1 - 8) namun kali ini menggunakan algoritma DecisionTreeClassifier dan kalian bisa menggunakan RandomizedSearchCV apabila process training lama. pelajari algoritma DecisionTreeClassifier di linkberikut: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html?highlight=decisiontreeclassifier#sklearn.tree.DecisionTreeClassifier
2. Bandingkan scorenya dengan Algoritma KNN, mana yang lebih baik?

Note : Data Science adalah experiment, sangat di dimungkinkan memerlukan beberapa kali percobaan untuk mendapatkan hasil yang terbaik! Happy Coding :)

NO 1

```
# 1. import algoritma DecisionTreeClassifier
from sklearn.tree import DecisionTreeClassifier

# Build model decision tree classifier
model_tree = DecisionTreeClassifier()
params = {'criterion' : ['entropy','gini'], 'splitter' : ['best',
'random'],
         'min_samples_split' : np.arange(2,50)}
gscv = GridSearchCV(model_tree, param_grid = params, cv = 10, scoring
= 'roc_auc')
gscv.fit(X_train, y_train)
```

```
GridSearchCV(cv=10, estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['entropy', 'gini'],
                         'min_samples_split': array([ 2,  3,  4,  5,
6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
35,
       36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]),
```

```
                        'splitter': ['best', 'random']},
             scoring='roc_auc')
```

```
# parameter terbaik
gscv.best_params_
```

```
{'criterion': 'gini', 'min_samples_split': 28, 'splitter': 'best'}
```

```
# score validasi terbaik
gscv.best_score_
```

```
0.7355882352941177
```

NO 2

Jawab

Score dari algoritma Decission Tree lebih baik dibandingkan dengan KNN