

NOTEBOOK UTS

Nama: Dilara Kynta Putri Raflita

NIM:1103204059

Kelas: TK44G4

Model data: Classification XGBoost

Dataset: [Sky server dataset](#)

1. Mengimport pustaka XGBoost untuk pemrosesan data dan evaluasi model
 - Library XGBoost, numpy, pandas, dan model yang terkait diimport

Input:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import xgboost as xgb
from sklearn.preprocessing import LabelEncoder
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix
```

2. Dataset “sky server.csv” diimport dari google drive menggunakan pandas

Input:

```
#mengimport file dataset dari google drive
from google.colab import drive
drive.mount('/content/drive')
```

output:

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

Drive sudah terhubung

3. Membaca dataset

- Dataset diambil dari lokasi yang ada di dalam kurung petik

Input:

```
data = pd.read_csv('/content/drive/MyDrive/sky_server.csv')
```

4. Menampilkan deskripsi data

Dataset ditampilkan dengan memberikan informasi kolom yang terdiri dari 10000 baris dan 18 kolom.

Input:

Data

output:

- Dataset memiliki 10000 baris dan 18 kolom
- Terdapat 18 atribut
- Terdapat fitur numerik dan kategorikal
- Output belum memberikan informasi tentang rentang dan sebaran nilai pada masing-masing kolom numerik
- Korelasi antar variabel numerik dapat dilihat melalui heatmap

	objid	ra	dec	u	g	r	i	z	run	rerun	camcol	field	specobjid	class	redshift	plate	mjd	fiberid
0	1.237650e+18	183.531326	0.089693	19.47406	17.04240	15.94699	15.50342	15.22531	752	301	4	267	3.722360e+18	STAR	-0.000009	3306	54922	491
1	1.237650e+18	183.598370	0.135285	18.66280	17.21449	16.67637	16.48922	16.39150	752	301	4	267	3.638140e+17	STAR	-0.000055	323	51615	541
2	1.237650e+18	183.680207	0.126185	19.38298	18.19169	17.47428	17.08732	16.80125	752	301	4	268	3.232740e+17	GALAXY	0.123111	287	52023	513
3	1.237650e+18	183.870529	0.049911	17.76536	16.60272	16.16116	15.98233	15.90438	752	301	4	269	3.722370e+18	STAR	-0.000111	3306	54922	510
4	1.237650e+18	183.883288	0.102557	17.55025	16.26342	16.43869	16.55492	16.61326	752	301	4	269	3.722370e+18	STAR	0.000590	3306	54922	512
...
9995	1.237650e+18	131.316413	51.539547	18.81777	17.47053	16.91508	16.68305	16.50570	1345	301	3	161	5.033450e+17	GALAXY	0.027583	447	51877	246
9996	1.237650e+18	131.306083	51.671341	18.27255	17.43849	17.07692	16.71661	16.69897	1345	301	3	162	5.033400e+17	GALAXY	0.117772	447	51877	228
9997	1.237650e+18	131.552562	51.666986	18.75818	17.77784	17.51872	17.43302	17.42048	1345	301	3	162	8.222620e+18	STAR	-0.000402	7303	57013	622
9998	1.237650e+18	131.477151	51.753068	18.88287	17.91068	17.53152	17.36284	17.13988	1345	301	3	163	5.033400e+17	GALAXY	0.014019	447	51877	229
9999	1.237650e+18	131.665012	51.805307	19.27586	17.37829	16.30542	15.83548	15.50588	1345	301	3	163	5.033410e+17	GALAXY	0.118417	447	51877	233

10000 rows x 18 columns

5. Membaca info data

Informasi dasar tentang dataset diperoleh dengan menggunakan `data.info()`

Input:

```
data.info()
```

output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 18 columns):
#   Column      Non-Null Count  Dtype
---  -
0   objid       10000 non-null  float64
1   ra          10000 non-null  float64
2   dec         10000 non-null  float64
3   u           10000 non-null  float64
4   g           10000 non-null  float64
5   r           10000 non-null  float64
6   i           10000 non-null  float64
7   z           10000 non-null  float64
8   run         10000 non-null  int64
9   rerun       10000 non-null  int64
10  camcol      10000 non-null  int64
11  field       10000 non-null  int64
12  specobjid   10000 non-null  float64
13  class       10000 non-null  object
14  redshift    10000 non-null  float64
15  plate       10000 non-null  int64
16  mjd         10000 non-null  int64
17  fiberid     10000 non-null  int64
dtypes: float64(10), int64(7), object(1)
memory usage: 1.4+ MB
```

- Sebagian besar kolom memiliki tipe data numerik (float64 dan int64), sedangkan satu kolom ('class') memiliki tipe data object.
- Tidak ada nilai null (NaN) dalam dataset. Semua kolom memiliki 10000 entri non-null.
- Data frame menggunakan sekitar 1.4 mb dari memori komputer

6. Visualisasi data

Input:

- Membuat pair plot menggunakan pustaka seaborn untuk fitur-fitur numerik tertentu dalam dataset.
- Dua library yang diimport adalah seaborn untuk visualisasi statistik data dan juga matplotlib.pyplot untuk visualisasi umum.
- Fitur-fitur numerik yang akan digunakan untuk membuat pair plot telah ditentukan
- Dataset diubah menjadi subset yang hanya berisi fitur numerik yang telah dipilih dan kolom target ('class').
- Fungsi dari pairplot digunakan untuk membuat matriks scatter plot dari fitur-fitur numerik yg dipilih.

```
#PAIR PLOT UNTUK FITUR NUMERIK
import seaborn as sns
import matplotlib.pyplot as plt

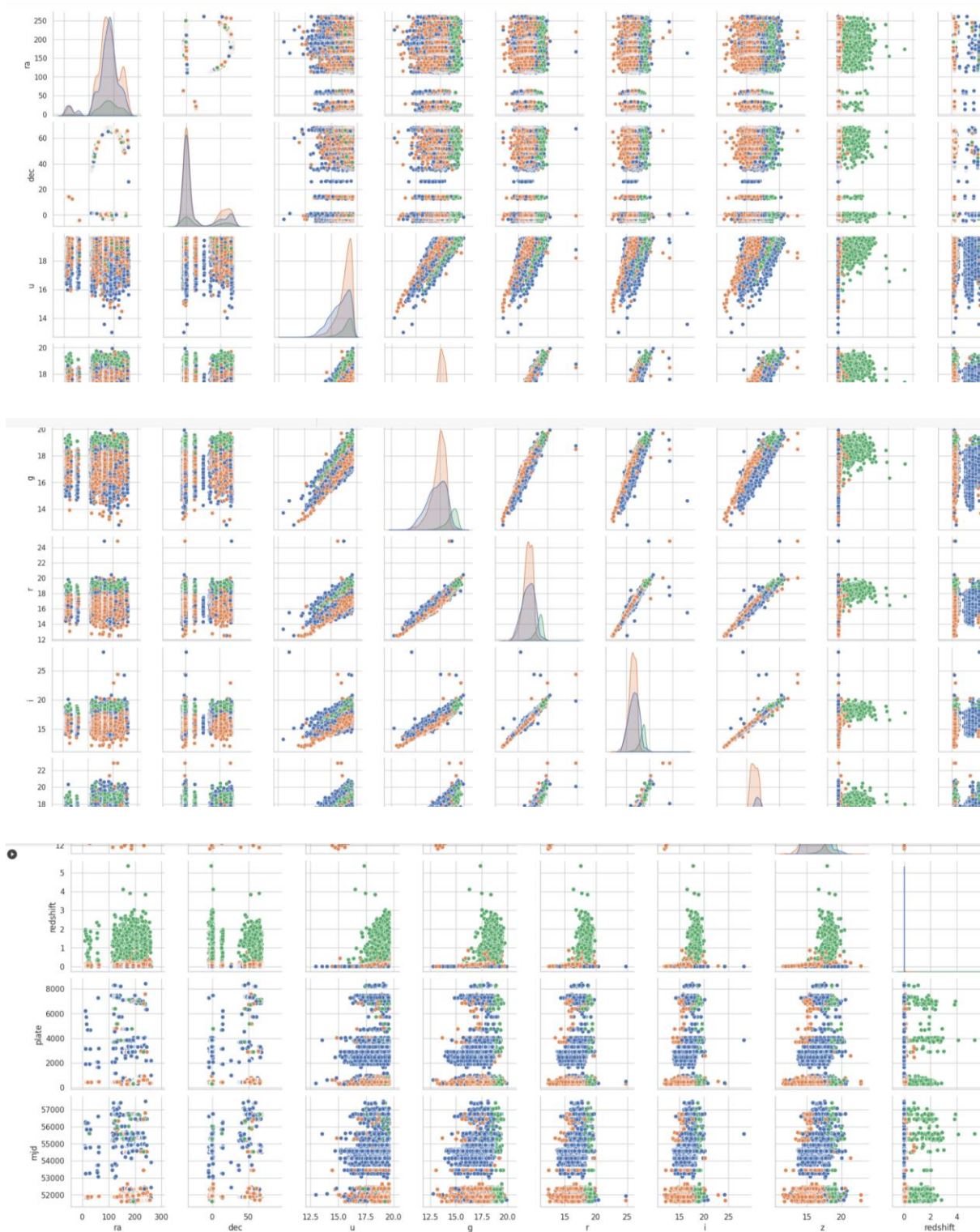
# Memilih fitur-fitur numerik
numeric_features = ['ra', 'dec', 'u', 'g', 'r', 'i', 'z', 'redshift',
                    'plate', 'mjd']

# Menggabungkan fitur numerik dengan target
data_numeric = data[numeric_features + ['class']]

# Membuat pair plot
sns.pairplot(data_numeric, hue='class')
plt.show()
```

output:

- Hasil visualisasi memberikan informasi tentang distribusi data, korelasi antar fitur, dan memisahkan data berdasarkan kelas.



7. Visualisasi korelasi antar variabel numerik

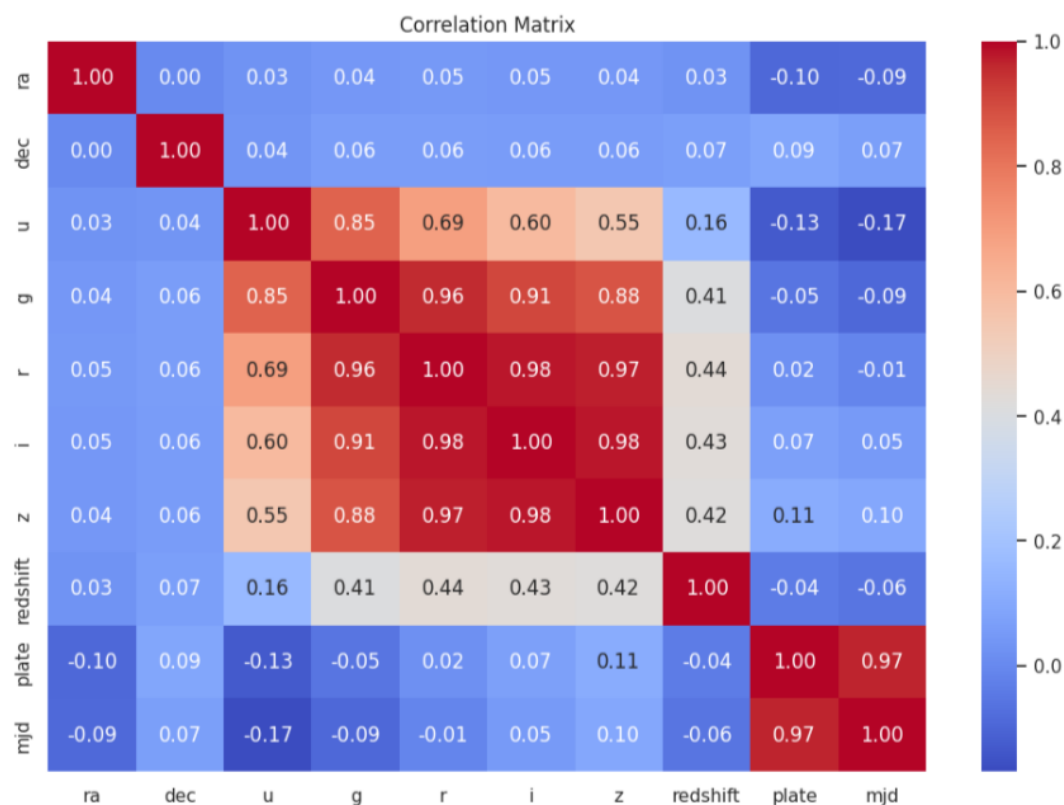
- Untuk membuat visualisasi matriks korelasi antar variabel numerik menggunakan *heatmap*.

Input:

```
# Visualisasi korelasi antar variabel numerik
plt.figure(figsize=(12, 8))
correlation_matrix = data[numeric_features].corr()
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
```

output:

- Visualisasi matriks korelasi ini memberikan gambaran tentang seberapa kuat hubungan antar variabel numerik
- Warna pada *hetmap* memberikan informasi tentang arah korelasi (positif atau negatif). Merah menunjukkan korelasi positif, sementara biru menunjukkan korelasi negatif.



8. Count plot untuk distribusi kelas

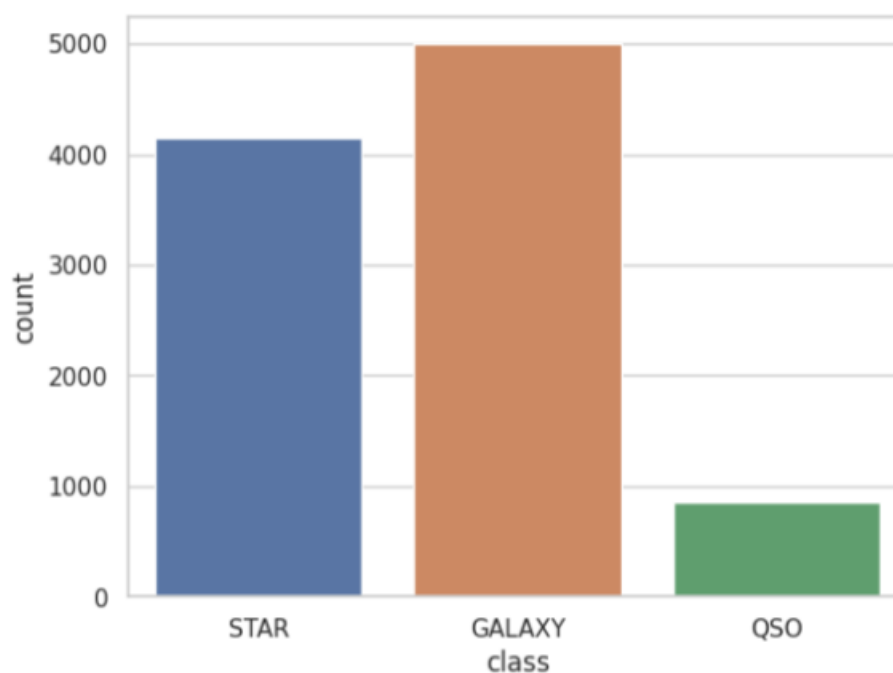
Input:

- Membuat count plot yang menampilkan distribusi kelas pada kolom 'class' dalam dataset. Dibuat dengan menggunakan seaborn 'countplot' dengan sumbu x ('class') dan menggunakan data dari dataset 'data'. Kemudian gambar count plot akan ditampilkan.

```
#count plot untuk distribusi kelas  
sns.countplot(x='class', data=data)  
plt.show()
```

output:

- Disumbu x terdapat kategori kelas yaitu 'star', 'galaxy', 'QSO'.
- Tinggi batang histogram pada sumbu y menunjukkan adanya jumlah observasi atau entri dalam dataset yang termasuk dalam setiap kategori kelas.
- Hasil 'QSO' memiliki jumlah data entri yang paling sedikit dibanding kelas lainnya. Hal ini menunjukkan adanya ketidakseimbangan dalam distribusi kelas.



9. Melakukan undersampling untuk ketidakseimbangan distribusi kelas

Input:

- Pada tahap ini dilakukan undersampling agar semua kelas terdistribusi dengan seimbang

```
from imblearn.under_sampling import RandomUnderSampler

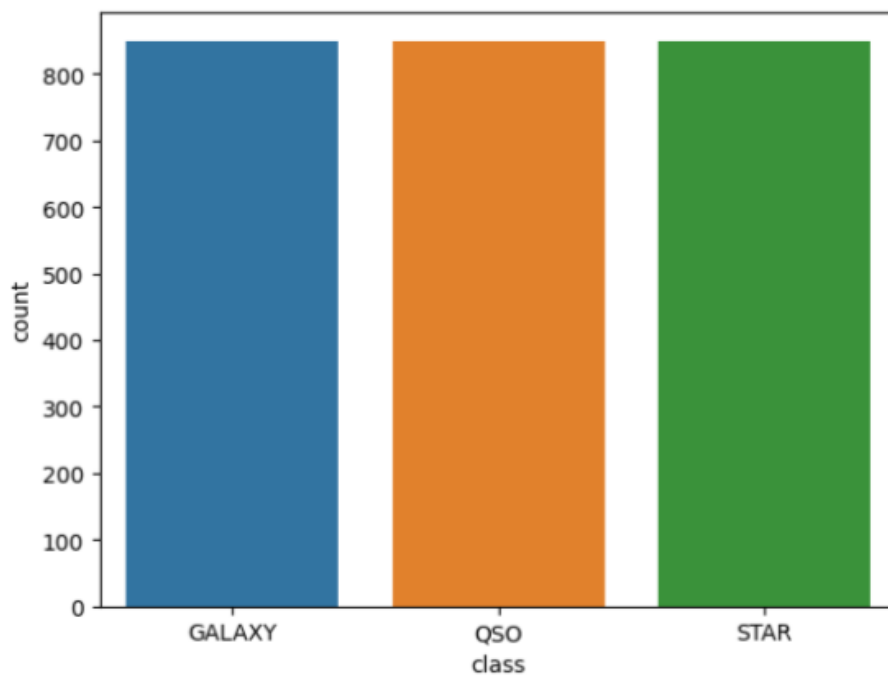
# Memisahkan fitur dan target
X = data.drop(['class'], axis=1)
y = data['class']

# Melakukan undersampling agar kelas 'STAR' tidak mendominasi
undersampler = RandomUnderSampler(sampling_strategy='not minority',
                                   random_state=42)
X_resampled, y_resampled = undersampler.fit_resample(X, y)

# Count plot setelah undersampling
sns.countplot(x=y_resampled)
plt.show()
```

output:

- Kelas 'galaxy', 'qso', dan 'star' sudah terdistribusi dengan baik



10. boxplot untuk memahami sebaran nilai

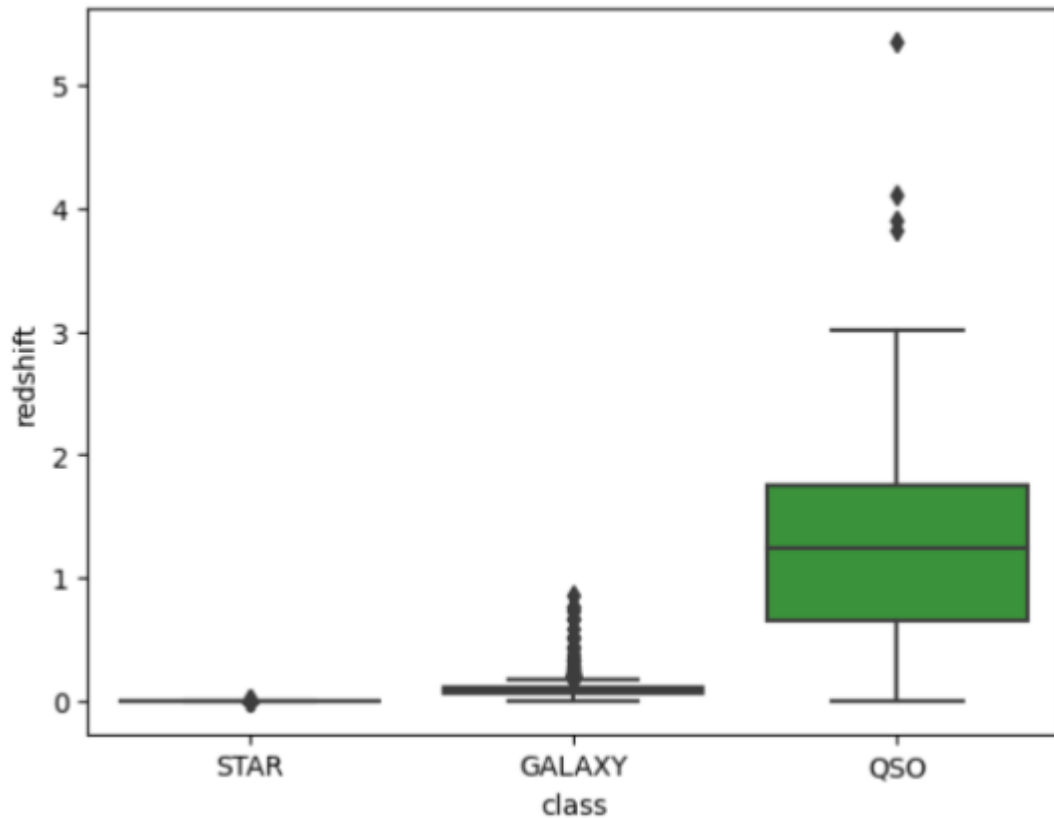
input:

- Boxplot memperlihatkan distribusi dan sebaran nilai 'redshift' untuk setiap kelas.
- `plt.show ()` untuk menunjukkan boxplot

```
#boxplot untuk memahami sebaran nilai
sns.boxplot(x='class', y='redshift', data=data)
plt.show()
```

output:

- Boxplot memperlihatkan distribusi dan sebaran nilai 'redshift' untuk setiap kelas
- Garis tengah di dalam kotak menunjukkan sebaran data di luar kuartil atas dan bawah
- Titik titik di luar menunjukkan potensial outlier
- Kelas 'STAR' memiliki nilai 'redshift' yang cenderung lebih rendah dan sebaran yang lebih padat
- Kelas 'GALAXY' memiliki sebaran nilai 'redshift' yang lebih besar dan cenderung lebih tinggi dibandingkan kelas 'STAR'
- Kelas 'QSO' memiliki sebaran nilai 'redshift' yang paling besar dengan beberapa nilai yang dianggap outlier



11. Inisialisasi dan melihat beberapa baris pertama data

Input:

```
print(data.head())
```

output:

	objid	ra	dec	u	g	r	i
0	1.237650e+18	183.531326	0.089693	19.47406	17.04240	15.94699	15.50342
1	1.237650e+18	183.598370	0.135285	18.66280	17.21449	16.67637	16.48922
2	1.237650e+18	183.680207	0.126185	19.38298	18.19169	17.47428	17.08732
3	1.237650e+18	183.870529	0.049911	17.76536	16.60272	16.16116	15.98233
4	1.237650e+18	183.883288	0.102557	17.55025	16.26342	16.43869	16.55492

	z	run	rerun	camcol	field	specobjid	class	redshift	plate
0	15.22531	752	301	4	267	3.722360e+18	STAR	-0.000009	3306
1	16.39150	752	301	4	267	3.638140e+17	STAR	-0.000055	323
2	16.80125	752	301	4	268	3.232740e+17	GALAXY	0.123111	287
3	15.90438	752	301	4	269	3.722370e+18	STAR	-0.000111	3306
4	16.61326	752	301	4	269	3.722370e+18	STAR	0.000590	3306

	mjd	fiberid
0	54922	491
1	51615	541
2	52023	513
3	54922	510
4	54922	512

12. Melakukan label encoding pada kolom 'class'

Input:

- Label encoding digunakan untuk mengubah nilai kategori menjadi bilangan bulat.
- Kolom 'class' yang awalnya berisi kategori 'STAR', 'GALAXY', dan 'QSO' diubah menjadi bilangan bulat
- Fungsi fit_transform digunakan untuk melakukan mapping antara nilai kategori dan bilangan bulat (fit), dan kedua yaitu mengubah kolom 'class' dengan bilangan bulat menggunakan (transform)

```
le = LabelEncoder()  
data['class'] = le.fit_transform(data['class'])
```

13. Memisahkan fitur dan target

Input:

- X merupakan variabel yang berisi fitur atau atribut dari dataset
- y merupakan variabel berisi target atau label dari dataset
- proses pemisahan dilakukan oleh data.drop dengan menghasilkan data frame baru yang tidak memiliki kolom 'class'

```
X = data.drop(['class'], axis=1)  
y = data['class']
```

14. Memisahkan data menjadi set pelatihan dan pengujian

Input:

- Proses pemisahan dataset menjadi dua set
- X_train sebagai variabel yg berisi fitur dari set pelatihan
- X_test sebagai variabel yg berisi fitur dari set pengujian
- y_train sebagai variabel yg berisi target dari set pelatihan
- y_test sebagai variabel yg berisi target dari set pengujian

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

15. Inisialisasi dan pelatihan model XGBoost

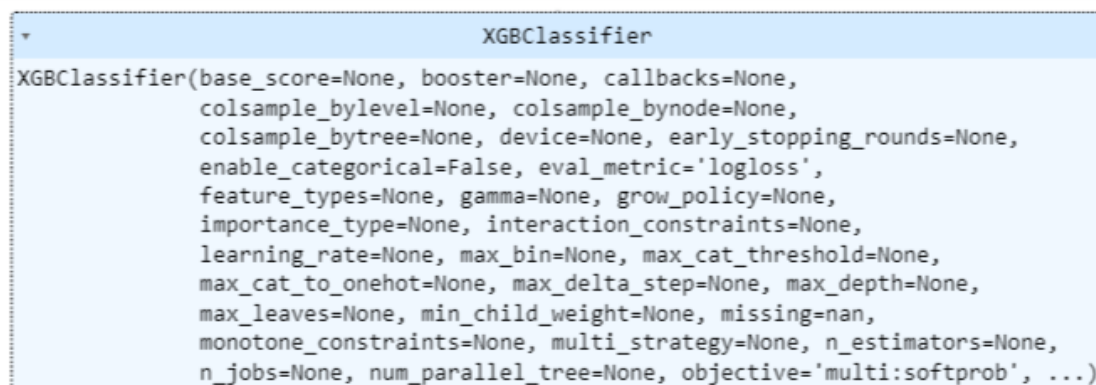
Input:

- Setelah pelatihan, model dapat digunakan untuk membuat prediksi pada data yang belum pernah dilihat

```
model = XGBClassifier(eval_metric='logloss')
model.fit(X_train, y_train)
```

output:

output ini menunjukkan konfigurasi dari model XGBoost yang telah diinisialisasi dan beberapa parameter yang diatur

A screenshot of a Jupyter Notebook cell showing the configuration of an XGBClassifier. The cell title is 'XGBClassifier'. The code displays the full parameter list for the classifier, with most parameters set to 'None' and 'eval_metric' set to 'logloss'.

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
               colsample_bylevel=None, colsample_bynode=None,
               colsample_bytree=None, device=None, early_stopping_rounds=None,
               enable_categorical=False, eval_metric='logloss',
               feature_types=None, gamma=None, grow_policy=None,
               importance_type=None, interaction_constraints=None,
               learning_rate=None, max_bin=None, max_cat_threshold=None,
               max_cat_to_onehot=None, max_delta_step=None, max_depth=None,
               max_leaves=None, min_child_weight=None, missing=nan,
               monotone_constraints=None, multi_strategy=None, n_estimators=None,
               n_jobs=None, num_parallel_tree=None, objective='multi:softprob', ...)
```

16. Melakukan prediksi pada set pengujian

Input:

- y_pred menyimpan hasil prediksi yang dihasilkan oleh model untuk set data uji. Setiap elemen dalam y_pred merupakan prediksi kelas untuk elemen yg sesuai dalam X_test. Hasil prediksi y_pred ini nanti akan digunakan untuk mengevaluasi kinerja model.

```
y_pred = model.predict(X_test)
```

17. Menampilkan hasil prediksi

Input:

```
print("Prediksi:", y_pred)
```

output:

output ini menunjukkan hasil prediksi dari model pada set data uji. Dalam kasus ini, hasil prediksi berupa array atau daftar nilai yang menunjukkan kelas yg diprediksi oleh model untuk setiap sampel dalam data uji.

```
Prediksi: [0 1 0 ... 2 2 0]
```

18. Menampilkan metrik evaluasi

Input:

- Akurasi dihitung menggunakan `accuracy_score(y_test, y_pred)`
- Hasilnya adalah proporsi sampel yg diprediksi dengan benar dari total sampel
- Laporan klasifikasi memberikan informasi lebih rinci tentang kinerja model pada setiap kelas, mencakup presisi, recall, dan f1-score untuk setiap kelas, serta nilai rata rata yg dihitung berdasarkan seluruh kelas

```
accuracy = accuracy_score(y_test, y_pred)
print(f'Akurasi: {accuracy}')
```



```
classification_rep = classification_report(y_test, y_pred)
print(f'\nLaporan Klasifikasi:\n{classification_rep}')
```

output:

Akurasi: 0.99

Laporan Klasifikasi:

	precision	recall	f1-score	support
0	0.99	0.99	0.99	996
1	0.98	0.94	0.96	190
2	1.00	1.00	1.00	814
accuracy			0.99	2000
macro avg	0.99	0.98	0.98	2000
weighted avg	0.99	0.99	0.99	2000

19. Menampilkan matriks konfusi

Input:

```
conf_matrix = confusion_matrix(y_test, y_pred)
print(f'\nMatriks Konfusi:\n{conf_matrix}')
```

output:

matriks ini memiliki tiga kolom dan tiga baris

- Kelas 0:

model memprediksi 989 sampel dengan benar sebagai kelas 0

Ada 7 kesalahan: 2 sampel dari kelas 0 diprediksi sebagai kelas 1, dan

4 sampel dari kelas 0 diprediksi sebagai kelas 2

- Kelas 1:

Model memprediksi 178 sampel dengan benar sebagai kelas 1

Tidak ada kesalahan yg tercatat

- Kelas 2:

Model memprediksi 813 sampel dengan benar sebagai kelas 2

Tidak ada kesalahan yg tercatat

Matriks Konfusi:

```
[[989  3  4]
 [ 12 178  0]
 [  1  0 813]]
```

20. Melakukan input new data untuk prediksi

Input:

```
new_data = pd.DataFrame({
    'objid': [123456789],
    'ra': [12.345],
    'dec': [45.678],
    'u': [18.9],
    'g': [17.5],
    'r': [16.2],
    'i': [15.8],
    'z': [15.5],
    'run': [789],
    'rerun': [301],
    'camcol': [4],
    'field': [205],
    'specobjid': [987654321],
    'redshift': [0.123],
    'plate': [456],
    'mjd': [55000],
    'fiberid': [42]
})
```

21. Melakukan prediksi data baru

Input:

```
new_data_pred = model.predict(new_data)
```

22. Menggunakan inverse transform untuk mendapat kelas asli

Input:

```
predicted_class = le.inverse_transform(new_data_pred)
```

23. Menampilkan hasil prediksi

Input:

```
print("Prediksi untuk data baru:", predicted_class)
```

output:

```
Prediksi untuk data baru: ['GALAXY']
```

24. Menambahkan data baru ke dalam dataframe asli

Input:

```
data_new = data.append(new_data, ignore_index=True)
```

25. Menampilkan data baru yang sudah ditambahkan

Input:

```
print("Data Baru yang Ditambahkan:")  
print(data_new.tail())
```

output:

Data Baru yang Ditambahkan:

	objid	ra	dec	u	g	r \
9996	1.237650e+18	131.306083	51.671341	18.27255	17.43849	17.07692
9997	1.237650e+18	131.552562	51.666986	18.75818	17.77784	17.51872
9998	1.237650e+18	131.477151	51.753068	18.88287	17.91068	17.53152
9999	1.237650e+18	131.665012	51.805307	19.27586	17.37829	16.30542
10000	1.234568e+08	12.345000	45.678000	18.90000	17.50000	16.20000

	i	z	run	rerun	camcol	field	specobjid	class \
9996	16.71661	16.69897	1345	301	3	162	5.033400e+17	0.0
9997	17.43302	17.42048	1345	301	3	162	8.222620e+18	2.0
9998	17.36284	17.13988	1345	301	3	163	5.033400e+17	0.0
9999	15.83548	15.50588	1345	301	3	163	5.033410e+17	0.0
10000	15.80000	15.50000	789	301	4	205	9.876543e+08	NaN

	redshift	plate	mjd	fiberid
9996	0.117772	447	51877	228
9997	-0.000402	7303	57013	622
9998	0.014019	447	51877	229
9999	0.118417	447	51877	233
10000	0.123000	456	55000	42

```
<ipython-input-61-23dfaddf3523>:2: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.concat instead.  
data_new = data.append(new_data, ignore_index=True)
```