In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
#read file
df = pd.read_csv('[Dataset]_Train_(Keryawan).csv')
df
```

Out[2]:

	Employee_ID	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decisi
0	EID_23371	F	42.0	4	Married	Franklin	IT	
1	EID_18000	M	24.0	3	Single	Springfield	Logistics	
2	EID_3891	F	58.0	3	Married	Clinton	Quality	
3	EID_17492	F	26.0	3	Single	Lebanon	Human Resource Management	
4	EID_22534	F	31.0	1	Married	Springfield	Logistics	
...	...	...	...	...	...	...	...	
6995	EID_16328	F	23.0	5	Married	Franklin	Operarions	
6996	EID_8387	F	44.0	1	Married	Lebanon	R&D	
6997	EID_8077	F	49.0	3	Single	Springfield	IT	
6998	EID_19597	F	47.0	3	Married	Washington	Sales	
6999	EID_1640	F	58.0	3	Married	Franklin	IT	

7000 rows × 24 columns

In [3]:
```
#mengecheck data null
null_data = df[df.isna().any(axis=1)]
null_data
```

Out[3]:

	Employee_ID	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decisi
3	EID_17492	F	26.0	3	Single	Lebanon	Human Resource Management	
7	EID_1235	F	NaN	3	Married	Springfield	Sales	
8	EID_10197	M	40.0	4	Single	Springfield	Production	
15	EID_20121	F	NaN	3	Married	Springfield	Logistics	
19	EID_12947	M	32.0	3	Single	Lebanon	IT	
...	...	...	...	...	...	...	...	
6969	EID_18566	F	NaN	1	Single	Washington	R&D	
6975	EID_2706	F	52.0	3	Married	Clinton	R&D	
6976	EID_15099	M	28.0	4	Married	Franklin	Operarions	

	Employee_ID	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decisi
6981	EID_25181	M	NaN	3	Married	Springfield	Logistics	
6986	EID_17099	M	NaN	4	Single	Franklin	Human Resource Management	

1647 rows × 24 columns

In [4]:
```
#delete null data
df = df.dropna()
df
```

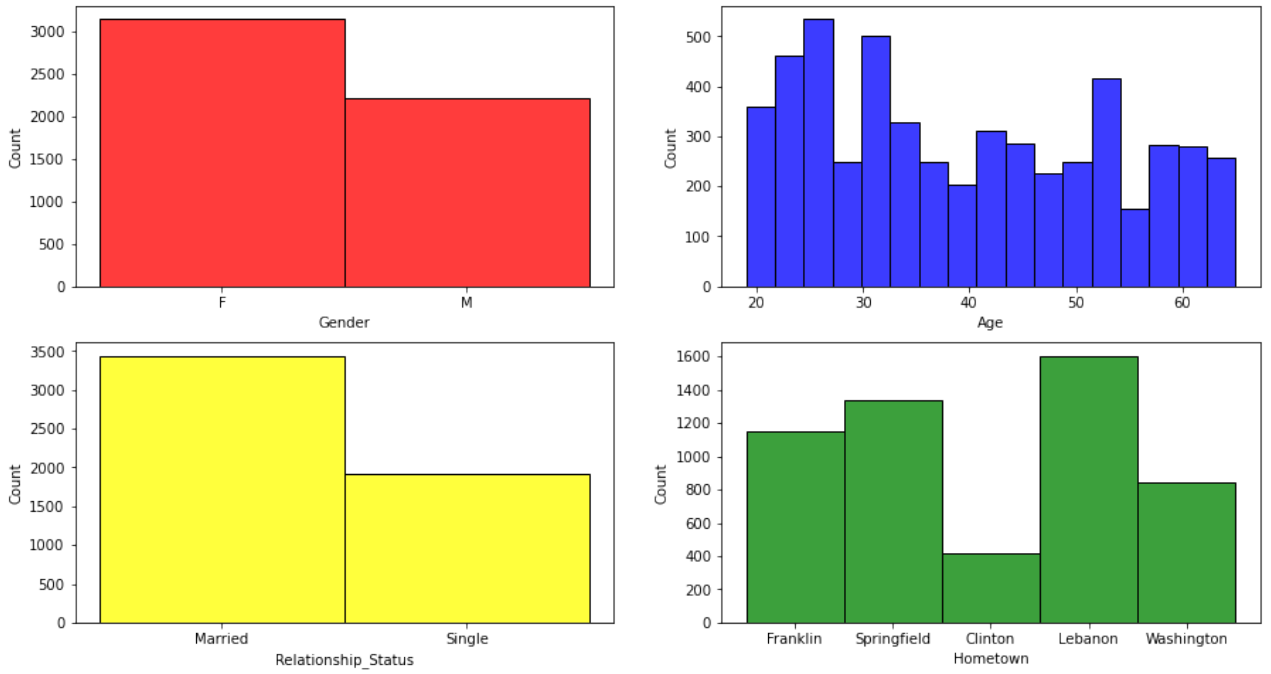
Out[4]:

	Employee_ID	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision
0	EID_23371	F	42.0	4	Married	Franklin	IT	
1	EID_18000	M	24.0	3	Single	Springfield	Logistics	
2	EID_3891	F	58.0	3	Married	Clinton	Quality	
4	EID_22534	F	31.0	1	Married	Springfield	Logistics	
5	EID_2278	M	54.0	3	Married	Lebanon	Purchasing	
...	...	...	...	...	...	...	...	
6995	EID_16328	F	23.0	5	Married	Franklin	Operarions	
6996	EID_8387	F	44.0	1	Married	Lebanon	R&D	
6997	EID_8077	F	49.0	3	Single	Springfield	IT	
6998	EID_19597	F	47.0	3	Married	Washington	Sales	
6999	EID_1640	F	58.0	3	Married	Franklin	IT	

5353 rows × 24 columns

In [5]:
```
#Visualisasi data
fig,axes = plt.subplots(2,2,figsize=(15,8))

sns.histplot(data=df,x='Gender', ax=axes [0,0],color='red')
sns.histplot(data=df,x='Age', ax=axes [0,1],color='blue')
sns.histplot(data=df,x='Relationship_Status', ax=axes [1,0],color='yellow')
sns.histplot(data=df,x='Hometown', ax=axes [1,1],color='green')
plt.show()
```



In [6]:
```
from sklearn.preprocessing import LabelEncoder
encode = LabelEncoder()
```

In [7]:
```
#Transformasi
df['Gender'] = encode.fit_transform(df['Gender'].values)
df['Relationship_Status'] = encode.fit_transform(df['Relationship_Status'].values)
df['Hometown'] = encode.fit_transform(df ['Hometown'].values)
df['Unit'] = encode.fit_transform(df['Unit'].values)
df['Decision_skill_possess'] = encode.fit_transform(df['Decision_skill_possess'].values
df['Compensation_and_Benefits'] = encode.fit_transform(df['Compensation_and_Benefits'].
df
```

```
<ipython-input-7-245873d11b90>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df['Gender'] = encode.fit_transform(df['Gender'].values)
<ipython-input-7-245873d11b90>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df['Relationship_Status'] = encode.fit_transform(df['Relationship_Status'].values)
<ipython-input-7-245873d11b90>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df['Hometown'] = encode.fit_transform(df ['Hometown'].values)
<ipython-input-7-245873d11b90>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

```
  df['Unit'] = encode.fit_transform(df['Unit'].values)
<ipython-input-7-245873d11b90>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df['Decision_skill_possess'] = encode.fit_transform(df['Decision_skill_possess'].value
s)
<ipython-input-7-245873d11b90>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df['Compensation_and_Benefits'] = encode.fit_transform(df['Compensation_and_Benefit
s'].values)
```

Out[7]:

	Employee_ID	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision_skill_
0	EID_23371	0	42.0	4	0	1	2	
1	EID_18000	1	24.0	3	1	3	3	
2	EID_3891	0	58.0	3	0	0	8	
4	EID_22534	0	31.0	1	0	3	3	
5	EID_2278	1	54.0	3	0	2	7	
...	...	...	...	...	...	...	...	
6995	EID_16328	0	23.0	5	0	1	5	
6996	EID_8387	0	44.0	1	0	2	9	
6997	EID_8077	0	49.0	3	1	3	2	
6998	EID_19597	0	47.0	3	0	4	10	
6999	EID_1640	0	58.0	3	0	1	2	

5353 rows × 24 columns

In [8]:
```
#delete kolom id
df_beda = df.drop(['Employee_ID'], axis=1)
df_beda = df_beda.drop(['Attrition_rate'], axis=1)
df_beda
```

Out[8]:

	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision_skill_possess	Time_
0	0	42.0	4	0	1	2	2	
1	1	24.0	3	1	3	3	0	
2	0	58.0	3	0	0	8	2	
4	0	31.0	1	0	3	3	2	
5	1	54.0	3	0	2	7	2	
...	...	...	...	...	...	...	...	
6995	0	23.0	5	0	1	5	1	

	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision_skill_possess	Time_
6996	0	44.0	1	0	2	9	0	
6997	0	49.0	3	1	3	2	3	
6998	0	47.0	3	0	4	10	1	
6999	0	58.0	3	0	1	2	3	

5353 rows × 22 columns

In [9]:
```
#Normalisasi data
from sklearn.preprocessing import MinMaxScaler
kolom = [col for col in df_beda.columns]
scaler = MinMaxScaler()
scaled = scaler.fit_transform(df_beda[kolom])
df_scaled = pd.DataFrame (scaled,columns=kolom)
```

In [10]:
```
df_scaled
```

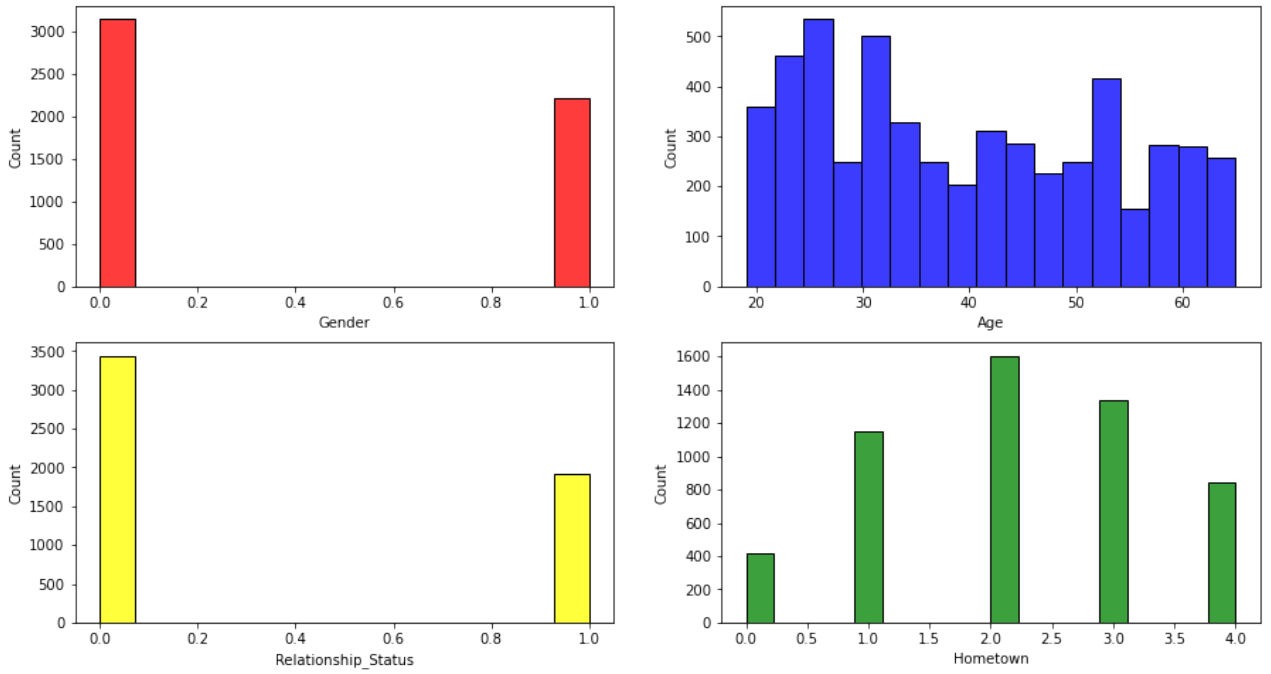
Out[10]:

	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision_skill_posses
0	0.0	0.500000	0.75	0.0	0.25	0.181818	0.66666
1	1.0	0.108696	0.50	1.0	0.75	0.272727	0.00000
2	0.0	0.847826	0.50	0.0	0.00	0.727273	0.66666
3	0.0	0.260870	0.00	0.0	0.75	0.272727	0.66666
4	1.0	0.760870	0.50	0.0	0.50	0.636364	0.66666
...	...	...	...	...	...	...	..
5348	0.0	0.086957	1.00	0.0	0.25	0.454545	0.33333
5349	0.0	0.543478	0.00	0.0	0.50	0.818182	0.00000
5350	0.0	0.652174	0.50	1.0	0.75	0.181818	1.00000
5351	0.0	0.608696	0.50	0.0	1.00	0.909091	0.33333
5352	0.0	0.847826	0.50	0.0	0.25	0.181818	1.00000

5353 rows × 22 columns

In [11]:
```
#Visualisasi data
fig,axes = plt.subplots(2,2,figsize=(15,8))

sns.histplot(data=df,x='Gender', ax=axes [0,0],color='red')
sns.histplot(data=df,x='Age', ax=axes [0,1],color='blue')
sns.histplot(data=df,x='Relationship_Status', ax=axes [1,0],color='yellow')
sns.histplot(data=df,x='Hometown', ax=axes [1,1],color='green')
plt.show()
```



In [12]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

In [13]:
```
#split data
X_train, X_test, y_train, y_test = train_test_split(df_scaled, df_scaled['Gender'], tes
```

In [14]:
```
#model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

Out[14]:
```
LogisticRegression(max_iter=1000)
```

In [15]:
```
#evaluasi model
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')
conf_matrix = confusion_matrix(y_test, y_pred)
print(f'Confusion Matrix:\n{conf_matrix}')
class_report = classification_report(y_test, y_pred)
print(f'Classification Report:\n{class_report}')
```

```
Accuracy: 1.0
Confusion Matrix:
[[598   0]
 [  0 473]]
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00       598
         1.0       1.00      1.00      1.00       473

    accuracy                           1.00      1071
   macro avg       1.00      1.00      1.00      1071
weighted avg       1.00      1.00      1.00      1071
```

In [16]:
```
df_scaled
```

Out[16]:

	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision_skill_posses
0	0.0	0.500000	0.75	0.0	0.25	0.181818	0.66666
1	1.0	0.108696	0.50	1.0	0.75	0.272727	0.00000
2	0.0	0.847826	0.50	0.0	0.00	0.727273	0.66666
3	0.0	0.260870	0.00	0.0	0.75	0.272727	0.66666
4	1.0	0.760870	0.50	0.0	0.50	0.636364	0.66666
...	...	...	...	...	...	...	..
5348	0.0	0.086957	1.00	0.0	0.25	0.454545	0.33333
5349	0.0	0.543478	0.00	0.0	0.50	0.818182	0.00000
5350	0.0	0.652174	0.50	1.0	0.75	0.181818	1.00000
5351	0.0	0.608696	0.50	0.0	1.00	0.909091	0.33333
5352	0.0	0.847826	0.50	0.0	0.25	0.181818	1.00000

5353 rows × 22 columns

In [17]:
```
#read data set test karyawan
new_data = pd.read_csv('[Dataset]_Test_(Karyawan).csv')
new_data = new_data.drop(['Employee_ID'], axis=1)
```

In [22]:
```
#pra prosesing
new_data = new_data.dropna()

#Transformasi
new_data['Gender'] = encode.fit_transform(new_data['Gender'].values)

kolom2 = [col for col in new_data.columns]
scaled2 = scaler.fit_transform(new_data[kolom2])
new_data_scaled = pd.DataFrame (scaled2,columns=kolom2)

new_data_scaled
```

```
<ipython-input-22-4a06046d79b4>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  new_data['Gender'] = encode.fit_transform(new_data['Gender'].values)
```

Out[22]:

	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision_skill_posses
0	0.0	0.282609	1.00	1.0	0.75	0.818182	0.66666
1	1.0	1.000000	0.25	1.0	0.50	0.181818	1.00000
2	1.0	0.717391	0.50	0.0	0.75	0.909091	1.00000
3	1.0	0.673913	1.00	1.0	1.00	0.363636	0.00000

	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision_skill_posses
4	0.0	0.543478	0.50	0.0	0.25	0.818182	0.66666
...	...	...	...	...	...	...	..
2316	1.0	0.608696	0.50	0.0	0.25	0.454545	0.33333
2317	0.0	0.347826	0.75	1.0	1.00	0.363636	0.66666
2318	0.0	0.282609	0.50	1.0	0.25	0.909091	1.00000
2319	0.0	0.695652	0.00	0.0	0.75	0.181818	0.33333
2320	0.0	0.565217	0.00	1.0	0.25	0.909091	1.00000

2321 rows × 22 columns

In [23]:
```
df_scaled
```

Out[23]:

	Gender	Age	Education_Level	Relationship_Status	Hometown	Unit	Decision_skill_posses
0	0.0	0.500000	0.75	0.0	0.25	0.181818	0.66666
1	1.0	0.108696	0.50	1.0	0.75	0.272727	0.00000
2	0.0	0.847826	0.50	0.0	0.00	0.727273	0.66666
3	0.0	0.260870	0.00	0.0	0.75	0.272727	0.66666
4	1.0	0.760870	0.50	0.0	0.50	0.636364	0.66666
...	...	...	...	...	...	...	..
5348	0.0	0.086957	1.00	0.0	0.25	0.454545	0.33333
5349	0.0	0.543478	0.00	0.0	0.50	0.818182	0.00000
5350	0.0	0.652174	0.50	1.0	0.75	0.181818	1.00000
5351	0.0	0.608696	0.50	0.0	1.00	0.909091	0.33333
5352	0.0	0.847826	0.50	0.0	0.25	0.181818	1.00000

5353 rows × 22 columns

In [24]:
```
#uji coba model
new_prediction = model.predict(new_data_scaled)
print(f'New Prediction: {new_prediction}')
```

```
New Prediction: [0. 1. 1. ... 0. 0. 0.]
```

In [ ]:
```

```