20250530_01

May 30, 2025

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

[24]:
```
# We use Titanic dataset to exercise missing values
data = sns.load_dataset("titanic")
data.head()
```

[24]:
```
   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  22.0      1      0   7.2500        S  Third
1         1       1  female  38.0      1      0  71.2833        C  First
2         1       3  female  26.0      0      0   7.9250        S  Third
3         1       1  female  35.0      1      0  53.1000        S  First
4         0       3    male  35.0      0      0   8.0500        S  Third

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
2  woman       False  NaN  Southampton   yes   True
3  woman       False    C  Southampton   yes  False
4    man        True  NaN  Southampton    no   True
```

[4]:
```
data.isnull().sum()
```

[4]:
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
```

```
alone                0
dtype: int64
```

[8]:
```
# First strategy : Drop
data_dropped = data.dropna()
print("Original:", data.shape)
print("After drop:", data_dropped.shape)
```

```
Original: (891, 15)
After drop: (182, 15)
```

[22]:
```
# Second strategy : Forward/Backward fill
data_forward = data.ffill()
print("Forward fill preview:")
data_forward.isnull().sum()
```

```
Forward fill preview:
```

[22]:
```
survived       0
pclass         0
sex            0
age            0
sibsp          0
parch          0
fare           0
embarked       0
class          0
who            0
adult_male     0
deck           1
embark_town    0
alive          0
alone          0
dtype: int64
```

[19]:
```
# Third strategy : Mean fill (numeric only)
data_mean = data.fillna(data.mean(numeric_only = True))
print("Mean fill preview:")
data_mean.isnull().sum()
```

```
Mean fill preview:
```

[19]:
```
survived          0
pclass            0
sex               0
age               0
sibsp             0
parch             0
fare              0
```

```
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

[49]:
```python
from sklearn.preprocessing import MinMaxScaler, StandardScaler
import numpy as np
```

[29]:
```python
numeric_col = data_mean[['age']]
```

[30]:
```python
# Min-Max Scaling
minmax = MinMaxScaler()
scaled_minmax = minmax.fit_transform(numeric_col)
```

[50]:
```python
# Standardization (Z-score)
standard = StandardScaler()
scaled_zscore = standard.fit_transform(numeric_col)
```

[51]:
```python
# Add back to DataFrame for comparison
scaled_data = pd.DataFrame({"original": numeric_col.values.ravel(),
                          "minmax_scaled": scaled_minmax.ravel(),
                          "zscore_scaled": scaled_zscore.ravel()})
scaled_data.head()
```

[51]:
```
   original  minmax_scaled  zscore_scaled
0      22.0       0.271174      -0.592481
1      38.0       0.472229       0.638789
2      26.0       0.321438      -0.284663
3      35.0       0.434531       0.407926
4      35.0       0.434531       0.407926
```

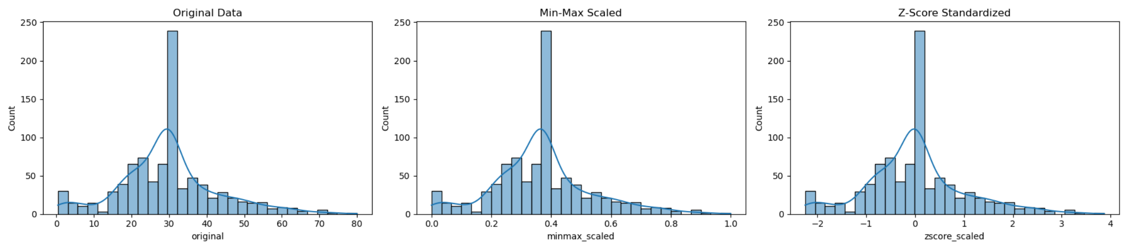
[52]:
```python
fig, ax = plt.subplots(1, 3, figsize = (18, 4))

sns.histplot(scaled_data["original"], ax = ax[0], kde = True)
ax[0].set_title("Original Data")

sns.histplot(scaled_data["minmax_scaled"], ax = ax[1], kde = True)
ax[1].set_title("Min-Max Scaled")

sns.histplot(scaled_data["zscore_scaled"], ax = ax[2], kde = True)
ax[2].set_title("Z-Score Standardized")
```

```
plt.tight_layout()
plt.show()
```


Original Data
Count
original
Min-Max Scaled
Count
minmax_scaled
Z-Score Standardized
Count
zscore_scaled