Module 3 Activity

Assigned at the start of Module 3

Due at the end of Module 3

Weekly Discussion Forum Participation

Each week, you are required to participate in the module's discussion forum. The discussion forum consists of the week's Module Activity, which is released at the beginning of the module. You must complete/attempt the activity before you can post about the activity and anything that relates to the topic.

Grading of the Discussion

1. Initial Post:

Create your thread by **Day 5 (Saturday night at midnight, PST).**

2. Responses:

Respond to at least two other posts by **Day 7 (Monday night at midnight, PST).**

Grading Criteria:

Your participation will be graded as follows:

Full Credit (100 points):

- Submit your initial post by **Day 5.**
- Respond to at least two other posts by **Day 7.**

Half Credit (50 points):

- If your initial post is late but you respond to two other posts.
- If your initial post is on time but you fail to respond to at least two other posts.

No Credit (0 points):

- If both your initial post and responses are late.
- If you fail to submit an initial post and do not respond to any others.

Additional Notes:

- **Late Initial Posts:** Late posts will automatically receive half credit if two responses are completed on time.
- **Substance Matters:** Responses must be thoughtful and constructive. Comments like “Great post!” or “I agree!” without further explanation will not earn credit.
- **Balance Participation:** Aim to engage with threads that have fewer or no responses to ensure a balanced discussion.

Avoid:

- A number of posts within a very short time-frame, especially immediately prior to the posting deadline.
- Posts that complement another post, and then consist of a summary of that.

Module Activity: Building a Preprocessing Pipeline

Objective

Learn how to build a preprocessing pipeline in scikit-learn and apply it to the famous Iris dataset. Gain hands-on experience in handling missing values, scaling features, and understanding the importance of preprocessing pipelines.

Sample Code for Pipeline Syntax

Here’s an example to help you understand how to create a pipeline. This pipeline imputes missing values using the mean:

```python import pandas as pd import numpy as np from sklearn.pipeline import Pipeline from sklearn.impute import SimpleImputer

# Example dataset with missing values

data = pd.DataFrame({ 'Feature1': [1.0, np.nan, 3.0], 'Feature2': [np.nan, 2.0, 3.0] })
```

Define a pipeline with an imputer

pipeline = Pipeline([('imputer', SimpleImputer(strategy='mean'))])

Fit and transform the data

processed_data = pipeline.fit_transform(data)

print("Original Data:") print(data) print("\nProcessed Data:") print(processed_data)

Activity Instructions

Dataset Preparation

We will use the Iris dataset, randomly remove values to simulate missing data, and keep it in a Pandas DataFrame for you to preprocess.

Your Task

Build a preprocessing pipeline that:

- Imputes missing values using the median.
- Scales features to a `[0, 1]` range using `MinMaxScaler`.
- Add at least one more preprocessing step.

Reflection

At the end of the activity, answer the following questions:

1. What challenges did you face while handling missing data?
2. Why is it important to use a pipeline for preprocessing?

Dataset Setup

Run the following code to import the Iris dataset and simulate missing data. You will use this dataset for the activity.

```
import pandas as pd
import numpy as np
from sklearn.datasets import load_iris
```

```python
# Load the Iris dataset
iris = load_iris()
data = pd.DataFrame(iris.data, columns=iris.feature_names)

# Randomly introduce missing values in random cells
np.random.seed(42)
total_cells = data.size
num_missing = int(0.1 * total_cells)  # 10% of total cells
missing_indices = [(row, col) for row in range(data.shape[0]) for col
in range(data.shape[1])]
random_missing_indices = np.random.choice(len(missing_indices),
size=num_missing, replace=False)

for index in random_missing_indices:
    row, col = missing_indices[index]
    data.iat[row, col] = np.nan

print("Dataset with Missing Values:")
print(data.head(10))
```

```
Dataset with Missing Values:
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width
(cm)
0                5.1               3.5                NaN
0.2
1                4.9               3.0                1.4
0.2
2                4.7               3.2                NaN
0.2
3                4.6               3.1                1.5
0.2
4                5.0               3.6                1.4
0.2
5                5.4               3.9                1.7
0.4
6                NaN               3.4                1.4
0.3
7                5.0               NaN                NaN
0.2
8                4.4               2.9                1.4
0.2
9                4.9               3.1                1.5
0.1
```

Next Steps

1. **Build your pipeline** to preprocess the dataset.
2. **Test your pipeline** by fitting it to the Iris dataset and transforming it.
3. **Review the processed data** and reflect on how the pipeline simplifies your workflow.

```python
from sklearn.preprocessing import MinMaxScaler # To Scale features
from sklearn.preprocessing import QuantileTransformer # To capture non-linear relationships for models that are inherently linear
from sklearn.impute import SimpleImputer # To handle potential missing values
from sklearn.pipeline import Pipeline

# 1. Define the preprocess steps
pipeline_steps = [
    ('imputer', SimpleImputer(strategy='median')), # Step 1: Impute NaNs with median
    ('quantile_transform', QuantileTransformer(output_distribution='normal')), # Step 2: Transforms features using quantile information. It can spread out the most frequent values and reduce the impact of (marginal) outliers. Can map to a uniform or normal distribution.
    ('min_max_scaler', MinMaxScaler()), # Step 3: Scale features to [0, 1]
]

# 2. Create the pipeline
numerical_pipeline = Pipeline(steps=pipeline_steps)

# Fit and transform the data
data_processed_np = numerical_pipeline.fit_transform(data)

data_processed_df = pd.DataFrame(data_processed_np, columns=data.columns, index=data.index)
data_processed_df.head(10)
```

```
/lib/python3.12/site-packages/sklearn/preprocessing/_data.py:2785: UserWarning: n_quantiles (1000) is greater than the total number of samples (150). n_quantiles is set to n_samples.
  warnings.warn(
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	0.426220	0.612608	0.500000	0.384116
1	0.385771	0.494334	0.377125	0.384116
2	0.348052	0.552671	0.500000	0.384116
3	0.331942	0.534713	0.407396	0.384116
4	0.407396	0.626628	0.377125	0.384116
5	0.449199	0.680585	0.434628	0.439637
6	0.500809	0.590064	0.377125	

```
0.427301
7           0.407396          0.494334          0.500000
0.384116
8           0.287112          0.451968          0.377125
0.384116
9           0.385771          0.534713          0.407396
0.000000
```