Feature Engineering – Assignment

Name: Palak Khandelia
Date: 26 August 2025

Q1. What is a parameter?

Answer:
A parameter is a value inside a model that is learned from the data.
They define how input features are mapped to output predictions.
Example: In linear regression (y = mx + c), slope (m) and intercept (c) are parameters.

Q2. What is correlation? What does negative correlation mean?

Answer:

- Correlation measures the strength and direction of the relationship between two variables.
- Range: -1 to +1
    - **+1:** Perfect positive correlation (both increase together).
    - **-1:** Perfect negative correlation (one increases, other decreases).
      Example:
- Study hours ↑ → Marks ↑ (positive)
- TV hours ↑ → Marks ↓ (negative)

Q3. Define Machine Learning. What are the main components in Machine Learning?

Answer:
Machine Learning is a branch of AI where systems learn from data and make predictions or decisions without being explicitly programmed.

Main Components:

1. Data
2. Features
3. Model
4. Training
5. Evaluation

Q4. How does loss value help in determining whether the model is good or not?

Answer:

- Loss function measures how far predictions are from actual values.
- Lower loss → Better model.
  Example: Mean Squared Error (MSE) in regression.

Q5. What are continuous and categorical variables?

Answer:

- Continuous Variable: Numeric values (e.g., height, salary).
- Categorical Variable: Categories/labels (e.g., Gender = Male/Female).

Q6. How do we handle categorical variables in Machine Learning?

Answer:
Techniques:

1. Label Encoding
2. One-Hot Encoding
3. Target Encoding

Q7. What do you mean by training and testing a dataset?

Answer:

- Training dataset: Used to train model.
- Testing dataset: Used to evaluate performance.

Q8. What is sklearn.preprocessing?

Answer:
A module in Scikit-learn used for preprocessing.
Includes:

- Scaling (StandardScaler, MinMaxScaler)
- Encoding (LabelEncoder, OneHotEncoder)
- Normalization

Q9. What is a Test set?

Answer:

A subset of dataset used to test model's performance on unseen data.

Q10. How do we split data for model fitting (training and testing) in Python?

```
from sklearn.model_selection import train_test_split
import pandas as pd

data = pd.DataFrame({"X":[1,2,3,4,5], "y":[2,4,6,8,10]})
X = data[["X"]]
y = data["y"]

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,random_state=42)
print("X_train:", X_train)
print("X_test:", X_test)
```

```
X_train:    X
2  3
0  1
3  4
X_test:    X
1  2
4  5
```

Q11. Why do we have to perform EDA before fitting a model to the data?

Answer:

EDA (Exploratory Data Analysis) helps to:

- Understand data distribution
- Detect missing values/outliers
- Identify correlations
- Decide preprocessing methods

Q12. What is correlation?

Answer:

Correlation measures the strength and direction of the relationship between two variables.

Q13. What does negative correlation mean?

Answer:

- Correlation measures the strength and direction of the relationship between two variables.

- Range: -1 to +1
    - **+1:** Perfect positive correlation (both increase together).
    - **-1:** Perfect negative correlation (one increases, other decreases).
      Example:
- Study hours ↑ → Marks ↑ (positive)
- TV hours ↑ → Marks ↓ (negative)

```
### Q14. How can you find correlation between variables in Python?
import pandas as pd

data = pd.DataFrame({
    "StudyHours":[2,4,6,8,10],
    "Marks":[20,40,60,80,100]
})
print(data.corr())
```

```
             StudyHours  Marks
StudyHours         1.0    1.0
Marks              1.0    1.0
```

Q15. What is causation? Explain difference between correlation and causation.

Answer:

- Correlation: Two variables move together but not necessarily cause-effect.
- Causation: One variable directly affects another.

Example:

- Ice cream sales ↑ and drowning cases ↑ (correlation, not causation).
- More study hours → Higher marks (causation).

Q16. What is an Optimizer? Types of optimizers?

Answer:
Optimizer: Algorithm to update model parameters to minimize loss.

Types:

1. Gradient Descent
2. SGD
3. Adam
4. RMSProp

Q17. What is sklearn.linear_model?

Answer:

A module containing linear models:

- Linear Regression
- Logistic Regression
- Ridge, Lasso

Q18. What does model.fit() do? What arguments must be given?

Answer:

- `model.fit()` is used to **train the machine learning model** on the training data.
- It adjusts the model's **parameters** based on the input features (X) and the target/output (y).
- It is called during the **training phase**.

Required Arguments:

1. `X_train` → Input features (independent variables).
2. `y_train` → Target values (dependent variable).

After calling `.fit()`, the model "learns" from the data and stores parameters (like slope and intercept in linear regression).

```
# Example of model.fit()
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
import pandas as pd

# Sample dataset
data = pd.DataFrame({
    "X":[1,2,3,4,5],
    "y":[2,4,6,8,10]
})

X = data[["X"]]   # Features
y = data["y"]     # Target

# Splitting data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Creating and training model
model = LinearRegression()
model.fit(X_train, y_train)

print("Model trained successfully!")
print("Coefficient:", model.coef_)
```

```
print("Intercept:", model.intercept_)
```

```
Model trained successfully!
Coefficient: [2.]
Intercept: 8.881784197001252e-16
```

Q19. What does model.predict() do? What arguments must be given?

Answer:

- `model.predict()` predicts output for given input features.
- Argument: `X_test`.
- It returns the predicted `y` values for unseen data.

```
# Example of model.predict()

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
import pandas as pd

# Sample dataset
data = pd.DataFrame({
    "X": [1, 2, 3, 4, 5],
    "y": [2, 4, 6, 8, 10]
})

X = data[["X"]]   # Features
y = data["y"]     # Target

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Train model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict on test set
y_pred = model.predict(X_test)

print("X_test values:\n", X_test)
print("Predicted y values:\n", y_pred)
print("Actual y values:\n", list(y_test))   # safe way
```

```
X_test values:
    X
1  2
4  5
Predicted y values:
 [ 4. 10.]
Actual y values:
 [4, 10]
```

Q20. What are continuous and categorical variables?

Answer:

- **Continuous Variables:**
  - Numeric variables that can take infinite values within a range.
  - They are measurable quantities.
  - Examples: Height (170.5 cm), Weight (60.2 kg), Salary (₹55,000), Temperature (36.6°C).
- **Categorical Variables:**
  - Variables that represent categories, labels, or groups.
  - They are qualitative, not numerical.
  - Examples: Gender (Male/Female), City (Delhi, Jaipur, Mumbai), Colors (Red, Blue, Green).

Continuous → Measurement (numbers)
Categorical → Labels (groups)

Q21. What is feature scaling? How does it help in Machine Learning?

Answer:

- **Feature Scaling** is the process of transforming all features into the same scale or range (like 0–1 or -1 to +1).
- Many ML models are sensitive to the scale of data (for example: KNN, SVM, Logistic Regression, Neural Networks).

Helps in:

1. Prevents features with larger values from dominating smaller features.
2. Improves speed and convergence of Gradient Descent.
3. Makes distance-based algorithms (KNN, Clustering) work correctly.

Example:

- Dataset: Age = [18, 22, 35], Salary = [20,000, 60,000, 1,00,000].
- Without scaling → Salary dominates Age because of large values.
- After scaling → Both Age and Salary contribute equally.

Q22. How do we perform scaling in Python?

```
from sklearn.preprocessing import StandardScaler, MinMaxScaler
import pandas as pd
```

```
# Sample data
data = pd.DataFrame({
    'Age': [25, 30, 45, 35, 50],
    'Salary': [50000, 60000, 80000, 75000, 90000]
})

# Standardization
scaler = StandardScaler()
standardized = scaler.fit_transform(data)

# Normalization
minmax = MinMaxScaler()
normalized = minmax.fit_transform(data)

print("Original Data:\n", data)
print("\nStandardized Data:\n", standardized)
print("\nNormalized Data:\n", normalized)
```

```
Original Data:
    Age  Salary
0   25   50000
1   30   60000
2   45   80000
3   35   75000
4   50   90000

Standardized Data:
 [[-1.29399328 -1.47029409]
 [-0.75482941 -0.77015405]
 [ 0.86266219  0.63012604]
 [-0.21566555  0.28005602]
 [ 1.40182605  1.33026608]]

Normalized Data:
 [[0.    0.   ]
 [0.2   0.25 ]
 [0.8   0.75 ]
 [0.4   0.625]
 [1.    1.   ]]
```

Q23. What is sklearn.preprocessing?

Answer:

- sklearn.preprocessing is a module in Scikit-learn for preprocessing data.
- It includes tools for:
    - Scaling (StandardScaler, MinMaxScaler)
    - Encoding (OneHotEncoder, LabelEncoder)
    - Normalization
    - Binarization

- It prepares raw data for training models.

Q24. How do we split data for model fitting (training and testing) in Python?

```
from sklearn.model_selection import train_test_split
import pandas as pd

# Sample dataset
data = pd.DataFrame({
    'Feature1': [10, 20, 30, 40, 50],
    'Feature2': [1, 2, 3, 4, 5],
    'Target':   [100, 200, 300, 400, 500]
})

X = data[['Feature1', 'Feature2']]  # Features
y = data['Target']                  # Target

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

print("X_train:\n", X_train)
print("\nX_test:\n", X_test)
print("\ny_train:\n", y_train)
print("\ny_test:\n", y_test)
```

```
X_train:
    Feature1  Feature2
4        50         5
2        30         3
0        10         1
3        40         4

X_test:
    Feature1  Feature2
1        20         2

y_train:
 4    500
2    300
0    100
3    400
Name: Target, dtype: int64

y_test:
 1    200
Name: Target, dtype: int64
```

Q25. Explain data encoding.

Answer:

- Data Encoding means converting categorical values into numbers.
- Types:
    1. Label Encoding → Converts categories to numbers (Male=0, Female=1).
    2. One-Hot Encoding → Creates dummy columns for each category.
    3. Ordinal Encoding → Assigns numbers based on order (Low=1, Medium=2, High=3).
- Encoding is needed because ML algorithms work with numerical data.

```
from sklearn.preprocessing import OneHotEncoder
import pandas as pd
data = pd.DataFrame({"Color":["Red","Blue","Green"]})
enc = OneHotEncoder(sparse_output=False)
print(enc.fit_transform(data))
```

```
[[0. 0. 1.]
 [1. 0. 0.]
 [0. 1. 0.]]
```