Feature Engineering Assignment Solution

1. What is a parameter?

A **parameter** is a configuration variable that is internal to the model and whose value can be estimated from data (e.g., weights in linear regression).

---

2. What is correlation?

Correlation measures the relationship between two variables — how one changes when the other changes.

What does negative correlation mean?

A negative correlation means that as one variable increases, the other tends to decrease (and vice versa). Example: More exercise, lower weight.

---

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

Machine Learning is a method where systems learn patterns from data to make decisions/predictions without being explicitly programmed.

Main components:

- **Data**
- **Model**
- **Loss function**
- **Optimizer**
- **Evaluation metric**

---

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

The **loss value** measures how well the model's predictions match the actual values. Lower loss = better model performance.

---

5. What are continuous and categorical variables?

- **Continuous**: Numeric values with infinite possible values (e.g., height, weight).
- **Categorical**: Values that represent groups or categories (e.g., gender, color).

---

6. How do we handle categorical variables in Machine Learning? Common techniques?

- **Label Encoding**: Converts categories into numbers.
- **One-Hot Encoding**: Creates binary columns for each category.
- **Ordinal Encoding**: For categories with order (e.g., low < medium < high).

---

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

- **Training set**: Used to train the model.
- **Testing set**: Used to evaluate how well the model generalizes to unseen data.

---

8 What is `sklearn.preprocessing`?

A module in Scikit-learn for data preprocessing tasks like scaling, encoding, normalization, etc.

---

9. What is a Test set?

A subset of data not used during training. It evaluates the model's performance on unseen data.

---

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

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2)
```

How do you approach a Machine Learning problem?

1. Understand the problem
2. Gather and clean data
3. Explore the data (EDA)
4. Preprocess the data
5. Choose a model
6. Train the model
7. Evaluate the model
8. Tune and optimize
9. Deploy and monitor

---

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

EDA (Exploratory Data Analysis) helps understand data patterns, distributions, outliers, and relationships. It guides cleaning and modeling.

---

12. What is correlation?

Correlation measures the relationship between two variables — how one changes when the other changes.

---

13. What does negative correlation mean?

A negative correlation means that as one variable increases, the other tends to decrease (and vice versa). Example: More exercise, lower weight.

---

14. How can you find a correlation between variables in Python?

```
import pandas as pd
df.corr()
```

You can also use **seaborn** for a heatmap:

```
import seaborn as sns
sns.heatmap(df.corr(), annot=True)
```

---

15. What is causation? Difference between correlation and causation?

- **Correlation**: A statistical relationship.
- **Causation**: One variable *causes* the other to change. Example: Ice cream sales and drowning may be correlated (summer), but one doesn't cause the other.

---

16. What is an Optimizer? What are different types of optimizers?

An **optimizer** adjusts model parameters to reduce loss.

Types:

- **SGD (Stochastic Gradient Descent)**: Basic optimizer.
- **Adam**: Combines momentum + adaptive learning rate.
- **RMSProp**: Scales learning rates using moving average.

Example with Adam:

```
optimizer = tf.keras.optimizers.Adam()
```

---

17. What is `sklearn.linear_model`?

A module in Scikit-learn that includes linear models like:

- Linear Regression
- Logistic Regression
- Ridge, Lasso, etc.

---

18. What does `model.fit()` do? Arguments?

It **trains** the model on data.

Arguments:

```
model.fit(X_train, y_train)
```

---

19. What does `model.predict()` do? Arguments?

It **predicts** outcomes for new/unseen data.

```
model.predict(X_test)
```

Continuous Variables

- These are **numerical values** that can take **any value within a range**.
- They are **measurable** and **can be divided infinitely** (theoretically).
- Examples:
    - Height (e.g., 172.3 cm)
    - Temperature (e.g., 36.5°C)
    - Salary (e.g., ₹45,000.75)
    - Age (e.g., 25.7 years)

📌 In Python/pandas: Usually stored as `float` or `int` types.

🔹 Categorical Variables

- These are **non-numeric (or nominal) variables** that represent **categories or groups**.
- They are **not measurable**, just labeled or classified.
- Two types:
    - **Nominal**: No natural order (e.g., Gender, Color, City)
    - **Ordinal**: Has a meaningful order (e.g., Low < Medium < High)

📌 In Python/pandas: Often stored as `object` or `category` types.

🧠 **Example:**

Age	Gender	Income	Education Level

25	Male	₹50,000	Graduate
30	Female	₹70,000	Postgraduate

- Age, Income → **Continuous**
- Gender, Education Level → **Categorical**

---

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

Feature scaling standardizes or normalizes data so that no feature dominates due to its scale.

Helps in:

- Improving convergence speed in gradient descent
- Enhancing performance of distance-based algorithms (e.g., KNN, SVM)

Techniques:

- Standardization (Z-score)
- Min-Max Scaling

Example:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

21 . How do we perform scaling in Python?

Scaling means transforming features to be on a similar scale so that models (especially ones like KNN, SVM, or Logistic Regression) perform better.

✅ **Common Scaling Techniques:**

- **StandardScaler** – Transforms data to have mean = 0 and standard deviation = 1.
- **MinMaxScaler** – Scales data to a fixed range, typically [0, 1].
- **RobustScaler** – Uses median and IQR; good for handling outliers.

📌 **Example using StandardScaler**:

```
from sklearn.preprocessing import StandardScaler
import numpy as np

# Sample data
data = np.array([[10, 20], [15, 30], [20, 40]])

# Apply scaling
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)

print(scaled_data)
```

🔹 22. What is `sklearn.preprocessing`?

`sklearn.preprocessing` is a **module in Scikit-learn** that provides tools for:

- Scaling (`StandardScaler`, `MinMaxScaler`)
- Encoding categorical features (`OneHotEncoder`, `LabelEncoder`)
- Generating polynomial features
- Normalizing data
- Handling missing values (with pipelines)

Think of it as your toolbox for cleaning and preparing raw data.

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

Use `train_test_split` from `sklearn.model_selection`.

Example:

```
from sklearn.model_selection import train_test_split

X = [[1], [2], [3], [4], [5], [6]]
y = [0, 0, 1, 1, 0, 1]

# 80% train, 20% test
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

print("Training data:", X_train)
print("Testing data:", X_test)
```

You always split before fitting the model to avoid data leakage.

🔹 24. Explain data encoding?

Data encoding is converting categorical data (like strings) into numerical form so models can understand them.

✅ **Types of Encoding:**

- **Label Encoding**: Assigns each category a number (good for ordinal data).

python
CopyEdit

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
data = ['Low', 'Medium', 'High', 'Medium']
encoded = le.fit_transform(data)
print(encoded)  # e.g., [1, 2, 0, 2]
```

- **One-Hot Encoding**: Creates binary columns for each category (good for nominal data).

python
CopyEdit

```
from sklearn.preprocessing import OneHotEncoder
import numpy as np

data = np.array([['Red'], ['Blue'], ['Green']])
encoder = OneHotEncoder(sparse=False)
encoded = encoder.fit_transform(data)
print(encoded)
```
