1) What is the model learning rate? We should choose a high learning rate for getting quick results on training data. comment on the statement.

Ans:
-> learning rate is a hyperparameter that controls how much the model's weights are updated during training after each iteration.
-> The statement *"We should choose a high learning rate for getting quick results on training data"* is partially incorrect.
-> While a high learning rate can make training faster initially, it may cause the model to overshoot the optimal solution, leading to unstable training or divergence.
-> A very high learning rate can also prevent the model from converging to a minimum.
-> Therefore, an appropriate or moderate learning rate is preferred to ensure stable and accurate learning.

2) Explain one-hot encoding .Does the dimensionality if the dataset increase,decrese or remains constant after applying the technique

Ans:
-> One-Hot Encoding is a method used to represent categorical variables in a numerical format by creating separate columns for each category.
-> Each category is represented by a binary value, where 1 indicates the presence of that category and 0 indicates absence.
->After applying one-hot encoding, the dimensionality of the dataset increases, because a single categorical column is replaced by multiple binary columns equal to the number of unique categories.

3) A financial institution has just hire, you to build a system which will decide what car insurance package to offer to different clients. The information recorded about the clients are -gender, age group (under 20, 20-35, 35-55 and over 55), credit rating, occupation, number of accidents in the last year, car make, model, year and the city where the person lives.

The financial institution has a database of roughly 20000 current clients. Explain step by step how you would use a logistic regression approach to build the system.

Your answer should contain:

a. Type of problem you are dealing with

b. How would you cater the categorical data e g. occupation, car make, model, city etc using them as features in your logistic regression model

c. Other relevant details

Ans:

a. Type of problem

This is a **supervised learning classification problem**.
Since the goal is to decide **which car insurance package** to offer to a client based on past data, logistic regression can be used as:

- **Binary classification** (e.g., offer Package A vs Not A), or
- **Multiclass classification** (multiple insurance packages using multinomial logistic regression).

b. Handling categorical data as features

Logistic regression requires **numerical input**, so categorical variables must be encoded.

- **Gender, age group, credit rating**:
  These can be encoded using **one-hot encoding** (or ordinal encoding if order matters, such as age groups or credit rating).
- **Occupation, car make, car model, city**:
  These are nominal categorical variables, so **one-hot encoding** is applied to convert each category into binary features.
- **Number of accidents, car year**:
  These are already numerical and can be used directly (after scaling if needed).

After encoding, all features are converted into a numerical feature matrix suitable for logistic regression.

c. Other relevant details (steps to build the system)

1. **Data collection**
   Use the existing dataset of 20,000 clients with their features and the insurance package previously offered.

2. **Data preprocessing**
    - Handle missing values
    - Encode categorical variables
    - Normalize numerical features (age, car year, accidents)
3. **Train–test split**
   Split data into training and testing sets (e.g., 80% training, 20% testing).
4. **Model training**
   Train a logistic regression model using the processed features and known insurance packages.
5. **Model evaluation**
   Evaluate performance using accuracy, precision, recall, confusion matrix, and ROC-AUC.
6. **Prediction & deployment**
   For a new client, input their details into the model to predict the most suitable insurance package.

Conclusion:
Logistic regression is suitable because it is interpretable, efficient for large datasets, and works well for classification problems involving structured client data.

4 Write the steps in designing an Unsupervised Machine Learning model. How do we change the penalty parameter in case we have a condition of underfitting using regularization technique?

Ans:

Steps in designing an Unsupervised ML model

1. **Problem Definition**
   Identify the objective (e.g., clustering customers, dimensionality reduction, anomaly detection).

2. **Data Collection**
   Gather raw, unlabeled data from relevant sources.

3. **Data Preprocessing**

   - Handle missing values
   - Remove duplicates
   - Scale/normalize features (important for distance-based models)

4. **Feature Selection / Extraction**

   - Remove irrelevant features
   - Apply PCA or autoencoders if dimensionality is high

5. **Model Selection**
   Choose an algorithm based on the task:

   - Clustering: K-Means, DBSCAN, Hierarchical
   - Dimensionality reduction: PCA, t-SNE
   - Anomaly detection: Isolation Forest, LOF

6. **Hyperparameter Tuning**
   Examples:

   - Number of clusters (K)
   - Distance metric
   - Min samples (DBSCAN)

7. **Model Training**
   Fit the model on the unlabeled data.

8. **Evaluation & Validation**
   Use internal metrics:

   - Silhouette Score
   - Davies–Bouldin Index
   - Reconstruction error (for PCA/autoencoders)

9. **Interpretation & Deployment**
   Interpret clusters or reduced dimensions and deploy the model.

Handling underfitting using regularization

- **Underfitting** means the model is too simple.
- Regularization penalty **prevents overfitting**, but **too much regularization causes underfitting**.

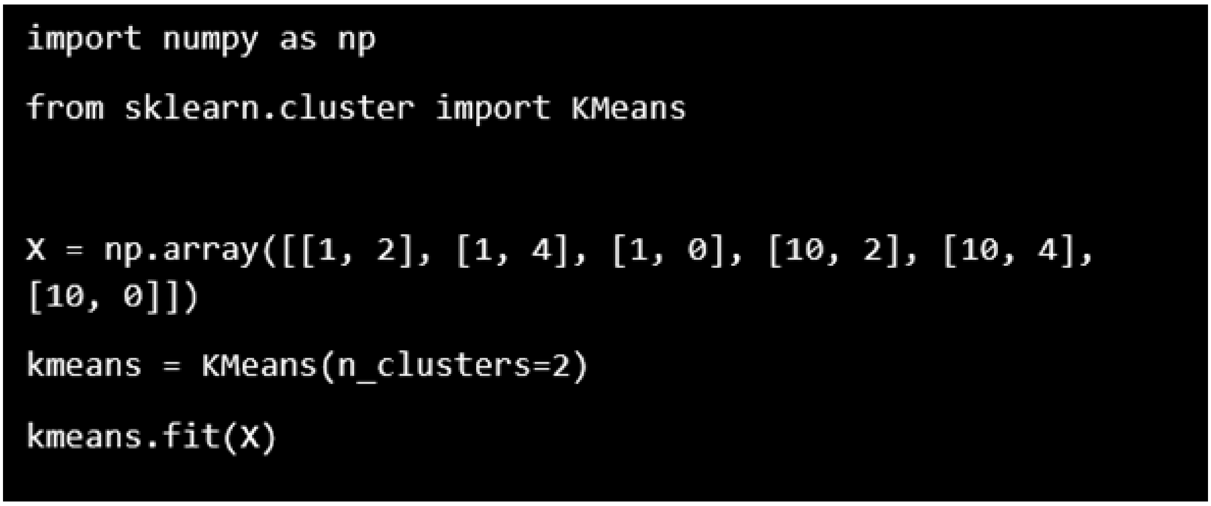
What to do if the model is underfitting?

Decrease the penalty parameter

Regularization	Penalty Parameter	Action for Underfitting
L1 / L2	λ (lambda)	Decrease λ
Ridge / Lasso	α (alpha)	Decrease α
SVM	C	Increase C (less regularization)

Conclusion:
👉 Reduce regularization strength so the model can learn more complex patterns.

5 What would be the expected output of the follow code snippet?



```
import numpy as np

from sklearn.cluster import KMeans


X = np.array([[1, 2], [1, 4], [1, 0], [10, 2], [10, 4],
[10, 0]])

kmeans = KMeans(n_clusters=2)

kmeans.fit(X)
```

Ans:

- No output is printed on the screen because there is no `print()` statement.

- The data is clustered into 2 clusters.

Cluster centers (expected):

```
[[ 1.  2.]

 [10.  2.]]
```

Cluster labels (one valid output):

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

(Label numbers may interchange, but grouping remains the same.)