Algorithm questions

1. **How does regularization (L1 and L2) help in preventing over-fitting?**

   Regularization helps in preventing over-fitting by adding a penalizing term to the cost function. It adds bias to the algorithm which results in low variance of the model.

   L1 (Lasso regression) = summation of absolute values of coefficients.

   L2 (Ridge regression) = summation of squares of coefficients.

2. **Why is feature scaling important in gradient descent?**

   Feature scaling is important in gradient descent because the cost function involves finding the difference between predicted and actual values, and in order to have consistent values, all variables need to be in comparable range. This ensures similar step size for all the features which ensures that algorithm reaches local minima faster.

Problem Solving

1. **Given a dataset with missing values, how would you handle them before training an ML model?**

   There are two ways of handling missing data:

   a) By removing the columns or rows with missing data, if the data available is large.
   b) By replacing the values with mean (if data follows normal distribution), median (if values are infrequent) or mode (if values are discrete).

2. **Design a pipeline for building a classification model. Include steps for data pre-processing.**

   Steps for classification model:

   1. Data collection.
   2. Data cleaning/pre-processing:
      a) Removing/Replacing missing data.
      b) Removing outliers.
      c) Data Scaling/Normalization.
      d) Label Encoding of categorical variables (depending on the model).
      e) Feature Engineering. Removing one or more correlated variables.
      f) Dividing data into training and testing set.
   3. Model building.
   4. Model evaluation.
   5. Model deployment.

Coding

1. **Write a Python script to implement a decision tree classifier using Scikit-learn.**

   Assuming we already have training and testing data as x_train,x_test,y_train,y_test

   From sklearn.tree import DecisionTreeClassifier

   decisiontreemodel = DecisionTreeClassifier()

   decisiontreemodel.fit(x_train,y_train)

   decisiontreemodel.predict(x_test)

2. **Given a dataset, write code to split the data into training and testing sets using an 80-20 split.**

   from sklearn.model_selection import train_test_split

   X = dataset[Independent variables]

   y = dataset[dependent variable]

   x_train,x_test,y_train,y_test = train_test_split(X,y, test_size=0.2, random_state=42)

Case Study

A company wants to predict employee attrition. What kind of ML problem is this? Which algorithms would you choose and why?

Employee attrition is a classification problem because employee would either leave or not, so we have two target categories: yes and no. The problem being a binary classification problem, following algorithms can be applied:

a) Logistic Regression: Uses logistic function to answer 0, if the value is below the threshold value of 0.5, or 1, if the value is above the threshold value of 0.5

b) Decision Tree Classifier: calculates information gain of variables at each node, results in root node classifying incoming data as yes or no.

c) Support Vector Machine: uses hyper-plane to divide two target categories.

d) Neural Networks, if dataset is large: finds differentiating features in independent variables that result in corresponding target variable.