**Interview Questions with Answers:**

**1) Recommendation System**

**Q.1)** **Can you explain the difference between user-based and item-based collaborative filtering?**

**Ans:**

1. **user-based** **collaborative filtering** :

Find users similar to the target user and recommend items those similar users liked.

**How it works:**

1. Compute similarity between users (e.g., cosine similarity, Pearson correlation).
2. Identify a neighbourhood of users most similar to the target user.
3. Aggregate the preferences of these similar users to recommend new items.

**b) Item-based collaborative filtering**

Find items similar to what the target user liked and recommend those items.

**How it works:**

1. Compute similarity between items (based on user ratings or interactions).
2. Look at items the user liked.
3. Recommend similar items based on item similarity scores.

**Q.2) What is collaborative filtering, and how does it work?**

**Ans:** Collaging the multiple users in a group according to their behaviour. If user 1 watch on horror movie and give good rating.

If user 2 also like to watch horror movie and give a good rating, so whenever user 1 watch any horror movie user 2 will get recommendation.

**2) RANDOM FOREST**

**Q.1) Explain Bagging and Boosting methods. How is it different from each other.**

**Ans :**

1. **Bagging: Bagging is a homogenous weak learner. It works parallel to improve the accuracy on every different random sample data and then combines the result. It helps to reduce variance and prevent overfitting**
2. **Boosting: Boosting is a homogenous weak learner it will sequentially build model one after another where each new model correct the mistakes of previous one so it help us to make better prediction and improve the accuracy.**

**Q.2) Explain how to handle imbalance in the data.**

**Ans: o handle imbalanced data, you can:**

1. **Resample the data:**
   * **Oversampling the minority class (e.g., SMOTE)**
   * **Undersampling the majority class**
2. **Use appropriate evaluation metrics:**
   * **Precision, Recall, F1-score, AUC-ROC instead of accuracy**
3. **Use class weights in models to penalize misclassifying the minority class more**
4. **Anomaly detection techniques if the imbalance is extreme**
5. **Ensemble methods like Balanced Random Forest or Boosting techniques (e.g., XGBoost with scale\_pos\_weight)**

**3)MLR**

**Q.1)** What is Normalization & Standardization and how is it helpful?

**Ans:** **Normalization** and **Standardization** are techniques used to *rescale* or *transform* data features so they can be better used in machine learning models

**Normalization :** Scaling data to a **fixed range**, typically **[0, 1]** or **[-1, 1]**.

**Standardization:** Scaling data so it has a **mean of 0** and a **standard deviation of 1.**

**Q.2)** What techniques can be used to address multicollinearity in multiple linear regression?

**Ans:** Remove highly correlated predictors – Drop one of the correlated variables.

Combine variables – Use techniques like Principal Component Analysis (PCA).

Regularization – Apply Ridge Regression (L2) or Lasso Regression (L1).

Increase sample size – Can reduce variance and impact of multicollinearity.

Centering variables – Subtract the mean to reduce correlation among interaction terms.

**4) Logistic Regression**

**Q.1)** What is the difference between precision and recall?

Ans: **Precision:** The proportion of **true positive** predictions out of all **positive predictions** made by the model.

**Recall:** The proportion of true positive predictions out of all actual positive instances.

**Q.2)** What is cross-validation, and why is it important in binary classification?

**Ans:** Cross-validation is a technique where the dataset is split into multiple subsets (called *folds*), and the model is trained and tested multiple times, each time using a different fold as the test set and the rest as the training set.

**Imp in binary classification:**

Reduces Overfitting Risk

More Reliable Model Selection

Helps Tune Thresholds & Metrics

**5) KNN**

**Q.1)** What are the key hyperparameters in KNN?

**Ans:** Number of neighbors.

* Small = overfitting, Large = underfitting.

 **distance metric**: How distance is measured (e.g., Euclidean, Manhattan).

* Affects how "closeness" is defined.

 **weights**:

* 'uniform': Equal weight to all neighbors.
* 'distance': Closer neighbors weigh more.

 **algorithm**: Method to compute neighbors (auto, kd\_tree, ball\_tree, brute).

* Affects speed.

 **leaf\_size**: Only for tree-based methods.

* Impacts speed/memory tradeoff.

**Q.2)** What distance metrics can be used in KNN

**Ans:** **Euclidean Distance (L2 norm)**

* Formula: ∑(xi−yi)2\sqrt{\sum (x\_i - y\_i)^2}∑(xi​−yi​)2

**6) Decission Tree**

**Q.1)** What are some common hyperparameters of decision tree models, and how do they affect the model's performance?

**Ans: max\_depth**

* **Description**: Maximum depth of the tree.
* **Effect**:
  + A smaller max\_depth limits the model's complexity, reducing overfitting but possibly underfitting the data.
  + A larger max\_depth increases model complexity and can lead to overfitting, especially on small or noisy datasets.

**2. min\_samples\_split**

* **Description**: Minimum number of samples required to split an internal node.
* **Effect**:
  + Higher values prevent the tree from growing deep, controlling overfitting.
  + Lower values allow more splits, which may capture more patterns but increase overfitting risk.

**3. min\_samples\_leaf**

* **Description**: Minimum number of samples required to be at a leaf node.
* **Effect**:
  + Helps smooth the model by requiring each leaf to have a minimum number of samples.
  + Prevents small, possibly noisy, splits at the leaves.

**4. max\_features**

* **Description**: Number of features to consider when looking for the best split.
* **Effect**:
  + Reduces model variance by limiting the number of features considered at each split.
  + Common in ensemble methods (like Random Forests) to add diversity among trees.

**5. max\_leaf\_nodes**

* **Description**: Maximum number of leaf nodes in the tree.
* **Effect**:
  + Like max\_depth, it limits model complexity and helps control overfitting.

**6. criterion**

* **Description**: Function used to measure the quality of a split.
  + Options: "gini" (default), "entropy" for classification.
* **Effect**:
  + Both usually yield similar results; "entropy" is more computationally expensive but sometimes leads to better splits.

**7. splitter**

* **Description**: Strategy used to choose the split at each node ("best" vs "random").
* **Effect**:
  + "best" tries all possible splits and chooses the best.
  + "random" selects a random subset, often used to increase randomness in ensemble methods.

**Q.2)** What is the difference between the Label encoding and One-hot encoding?

**Ans:** **Label Encoding :**Converts each unique category into a unique integer.

**One-hot encoding :** Creates a binary column for each category, and places a 1 in the column corresponding to the category and **0 elsewhere.**