1. A healthcare startup wants to build a model to predict the onset of diabetes. The dataset has 50 features, but the team wants to use only the most relevant ones. How can they apply feature selection effectively

To select features effectively, the team can start with **filter methods** like correlation, chi-square, or ANOVA to remove irrelevant features.
Next, they can use **embedded methods** such as Lasso or tree-based models to rank important features.
Wrapper methods like Recursive Feature Elimination (RFE) can further refine the selection.
They should validate the chosen subset with **cross-validation** to ensure accuracy.
This reduces overfitting, speeds up training, and improves interpretability.

2.You're creating a new project for managing student data. The project name is student_portal. What is the step-by-step command to start it?write logic for this django project

Run `django-admin startproject student_portal` to create the project and `cd student_portal`.
Create an app with `python manage.py startapp students`.
In **models.py**, define a `Student` model with fields like name, roll_number, email, and course.
In **views.py**, fetch all students and display them in a template.
Finally, add the app in **settings.py**, include URLs, migrate, and run the server.

3. A company is analyzing customer purchase patterns with 200+ behavioral features. How can they reduce dimensionality without losing predictive power?

Use **dimensionality reduction techniques** to handle 200+ features.

Apply **PCA (Principal Component Analysis)** to combine correlated features into fewer components.

Try **t-SNE or UMAP** for visualization, but mainly for clustering and pattern discovery.

Use **feature selection methods** (like mutual information, recursive feature elimination, or regularization).

Balance between reducing features and keeping high **predictive power** by testing with model performance.

4.A digital library wants to recommend books to readers based on what similar readers liked. How should they design this system?

Use a **collaborative filtering** approach for recommendations.

Collect data on what books each reader has read or rated.

Find **similar readers** using methods like cosine similarity or matrix factorization.

Recommend books that those similar readers liked but the current reader hasn't read yet.

Continuously update the system as new reader interactions are added.

5. A bank wants to assess the risk level of credit applicants using only the most important financial indicators. How can they reduce the number of features?

Apply **feature selection techniques** to pick the most important indicators.

Use **filter methods** like correlation or mutual information to remove irrelevant features.

Apply **wrapper methods** such as Recursive Feature Elimination (RFE).

Use **embedded methods** like Lasso (L1 regularization) or tree-based feature importance.

Keep only the key features that improve **risk prediction accuracy**.

6. A news app wants to recommend articles based on both article similarity and user reading history. How can they implement a hybrid system?

Use a **content-based filtering** system to recommend articles similar to ones the user already read.

Use **collaborative filtering** to suggest articles popular among similar readers.

Combine both methods into a **hybrid recommendation system**.

Weight or blend the scores from content-based and collaborative filtering.

Continuously update recommendations as user reading history grows.

7.**Question:**
You're building a spam detection model and have thousands of text features from emails. How do you identify the most useful ones?

Convert emails into features using **TF-IDF or word embeddings**.

Apply **feature selection** methods like chi-square test or mutual information to find strong predictors.

Use **embedded methods** such as L1 (Lasso) regularization or tree-based importance.

Remove low-variance or redundant features to reduce noise.

Keep only the most useful text features that improve spam classification accuracy.

8. An ed-tech platform wants to recommend courses based on what similar learners have enrolled in. What steps would you take?

Collect learner data on **course enrollments and ratings**.

Use **collaborative filtering** to find learners with similar enrollment patterns.

Recommend courses that similar learners have taken but the target learner hasn't.

Optionally, add **content-based filtering** using course topics or skills.

Continuously update recommendations as new learners join and enroll.

9.You're developing a car price prediction tool. With 100+ features (e.g., brand, mileage, engine type), how do you reduce complexity?

Use **feature selection** to keep only the most relevant predictors (e.g., mileage, engine size).

Apply **correlation analysis** to remove redundant features.

Use **embedded methods** like Lasso regression or tree-based feature importance.

Apply **dimensionality reduction** (e.g., PCA) if features are highly correlated.

Keep a smaller, high-impact feature set that improves model performance and reduces complexity.

10.
How do you recommend products to new users who haven't interacted with anything yet?

This is called the **cold start problem**.

Show **popular or trending products** as a starting point.

Use **demographic information** (age, location, etc.) if available.

Ask users for **initial preferences** through a short survey or onboarding.

Gradually switch to **personalized recommendations** as they interact.