Q1. Filter Method in Feature Selection

Definition:
The Filter method selects features **based on statistical measures** of their relationship with the target variable, **independent of any machine learning algorithm**.

How it works:

1. Calculate a **relevance score** for each feature (e.g., correlation, chi-square, mutual information).
2. Rank features by score.
3. Select the top features above a threshold.

Example:

- Using **Pearson correlation** to select features highly correlated with house price.

```
import pandas as pd
import numpy as np

correlations = df.corr()['Price'].abs()
selected_features = correlations[correlations > 0.5].index
```

Q2. Wrapper Method vs Filter Method

Feature	Filter Method	Wrapper Method
Selection criteria	Statistical measures (independent of model)	Model performance (dependent on ML algorithm)
Computation	Fast, simple	Slower, computationally expensive
Example	Correlation, Chi-square	Recursive Feature Elimination (RFE)

Key difference: Filter is independent of the model; Wrapper uses model performance to select features.

Q3. Embedded Feature Selection Techniques

Embedded methods select features during the **training of the model**.

Common techniques:

- **Lasso Regression (L1 regularization)** → shrinks irrelevant feature weights to zero
- **Ridge Regression (L2 regularization)** → reduces magnitude of feature weights
- **Decision Trees / Random Forests** → feature importance scores
- **Gradient Boosting models** → built-in feature selection

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()
model.fit(X, y)
importances = model.feature_importances_
```

Q4. Drawbacks of the Filter Method

- Ignores **feature interaction**
- May select redundant features
- Threshold selection is sometimes arbitrary
- Not tailored to a specific machine learning model

Q5. When to prefer Filter Method over Wrapper Method

- Large datasets with **many features** (computationally efficient)

- When **speed is important**
- Initial **feature reduction** before using more complex methods
- When you want **model-agnostic feature selection**

Q6. Feature Selection for Telecom Churn using Filter Method

Steps:

1. Calculate correlation of each feature with churn (target variable)
2. Use **Chi-square test** for categorical features
3. Rank features by relevance scores
4. Select top features (e.g., high correlation or p-value < 0.05)
5. Use selected features for modeling

```
from sklearn.feature_selection import SelectKBest, chi2

X_new = SelectKBest(chi2, k=10).fit_transform(X, y)
```

Q7. Using Embedded Method for Soccer Match Prediction

- Train a **Random Forest or XGBoost** model on the dataset
- Extract **feature importance scores** from the model
- Select top-ranked features (e.g., top 20% contributing to prediction)
- This **automatically considers feature interactions** and model performance

```
import xgboost as xgb
```

```
model = xgb.XGBClassifier()
model.fit(X, y)
importances = model.feature_importances_
```

Q8. Using Wrapper Method for House Price Prediction

Steps (Recursive Feature Elimination example):

1. Choose a base model (e.g., Linear Regression)
2. Recursively train the model while **removing least important feature at each step**
3. Evaluate model performance using cross-validation
4. Select the subset of features that **maximizes accuracy**

```
from sklearn.feature_selection import RFE
from sklearn.linear_model import LinearRegression

model = LinearRegression()
rfe = RFE(model, n_features_to_select=5)
X_selected = rfe.fit_transform(X, y)
```

Key idea: Wrapper methods are **computationally heavier** but **consider feature interactions** and model performance.

✅ Summary Table

Method	How it works	Pros	Cons	Example
Filter	Uses statistical measures	Fast, model-agnostic	Ignores interactions	Correlation, Chi-square
Wrapper	Uses model performance	Considers interactions	Slow, expensive	RFE, Forward/Backward selection

Embedded	Selects features during training	Fast, considers interactions	Model-specific	Lasso, Decision Trees, XGBoost