Lesson 8: Feature Selection

Lecturer: **Dr. Nguyen Tuan Long**
Email: ntlong@neu.edu.vn
Mobile: 0982 746 235

What is Feature Selection?

- Feature selection is the process of **reducing the number of input variables** when developing a predictive model.

Why Select Features? The Core Motivations

- **Reduce Overfitting & Improve Generalization:** Simpler models are less prone to memorizing noise and generalize better to new data.
- **Decrease Computational Time & Cost:** Fewer features mean significantly faster model training and prediction.
- **Enhance Model Interpretability & Explainability:** A model with 10 well-understood features is far more transparent than one with 500, which is crucial for building trust and explaining decisions.

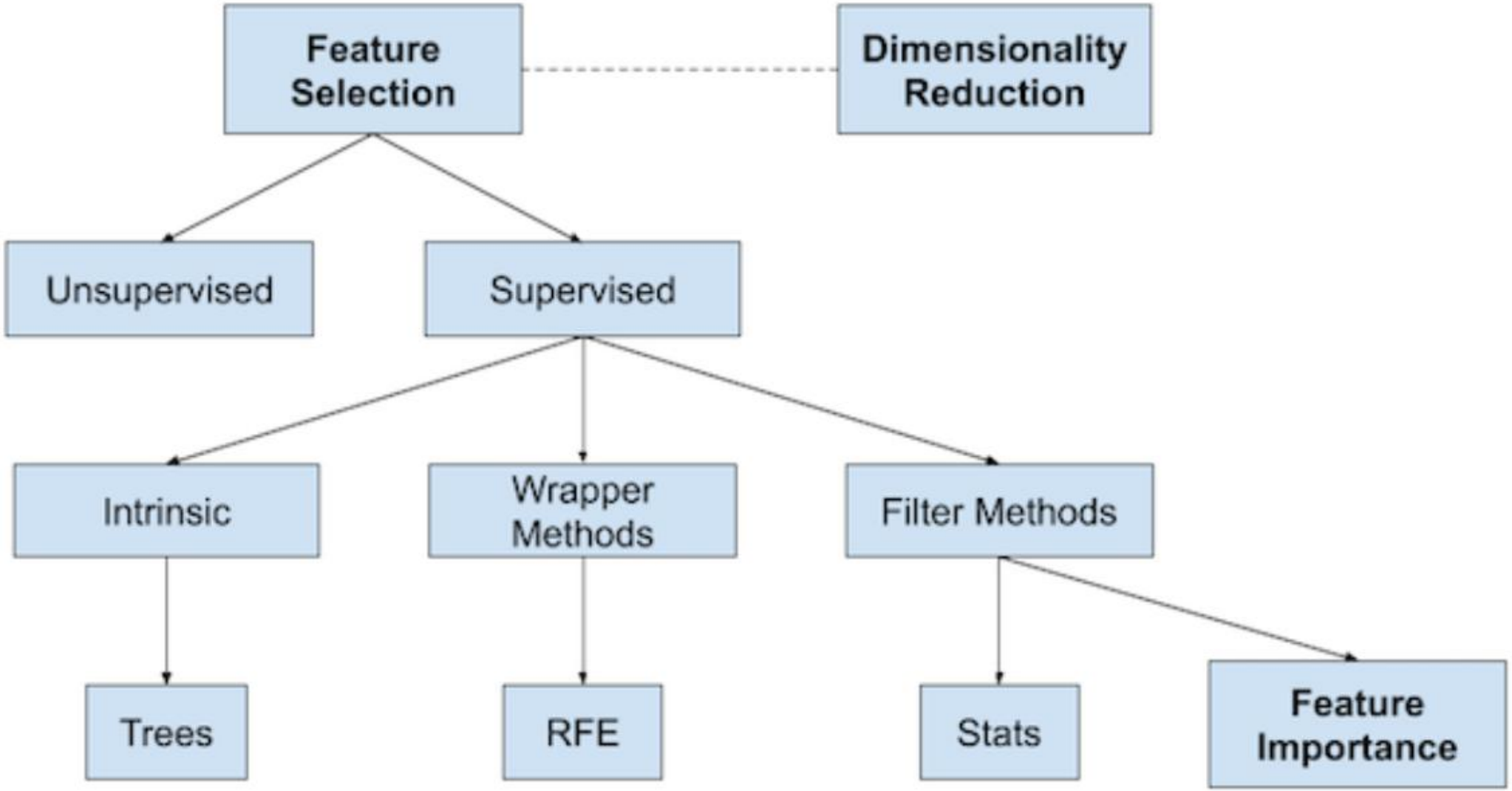
What method are available?

Unsupervised vs. Supervised Methods

The primary distinction lies in whether the target variable is used in the selection process.

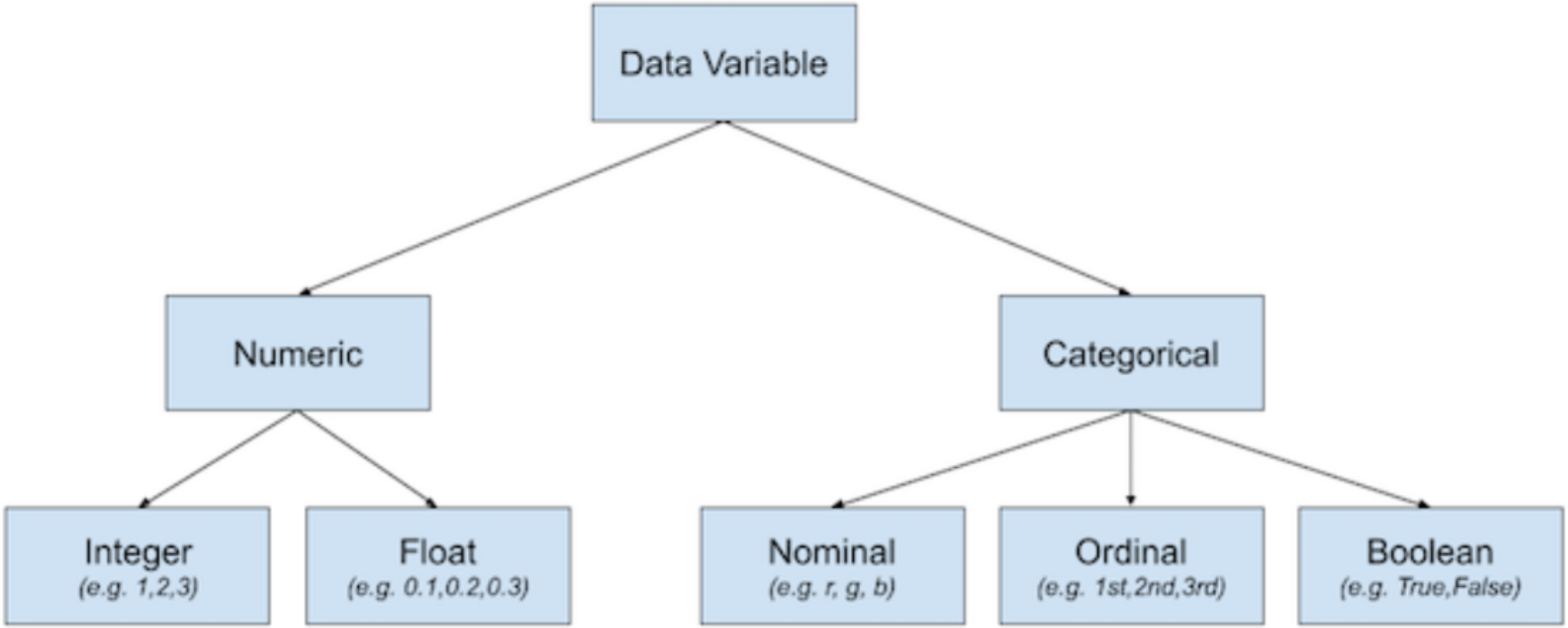
- **Unsupervised Methods:** Ignore the target variable. These are general data cleaning steps to remove redundant or uninformative features (e.g., removing features with low variance or high correlation).
- **Supervised Methods:** Use the target variable to eliminate irrelevant variables. These methods are classified into three main groups:
  1. **Filter Methods:** Use statistical tests to score and rank features based on their relationship with the target, *before* modeling. They are fast but blind to feature interactions.
  2. **Wrapper Methods:** Use a specific model's performance to search for the best subset of features. They are powerful but can be computationally very expensive.
  3. **Embedded (Intrinsic) Methods:** Feature selection is an integrated part of the model's training process (e.g., LASSO regression, Random Forest).

Overview of Feature Selection Techniques

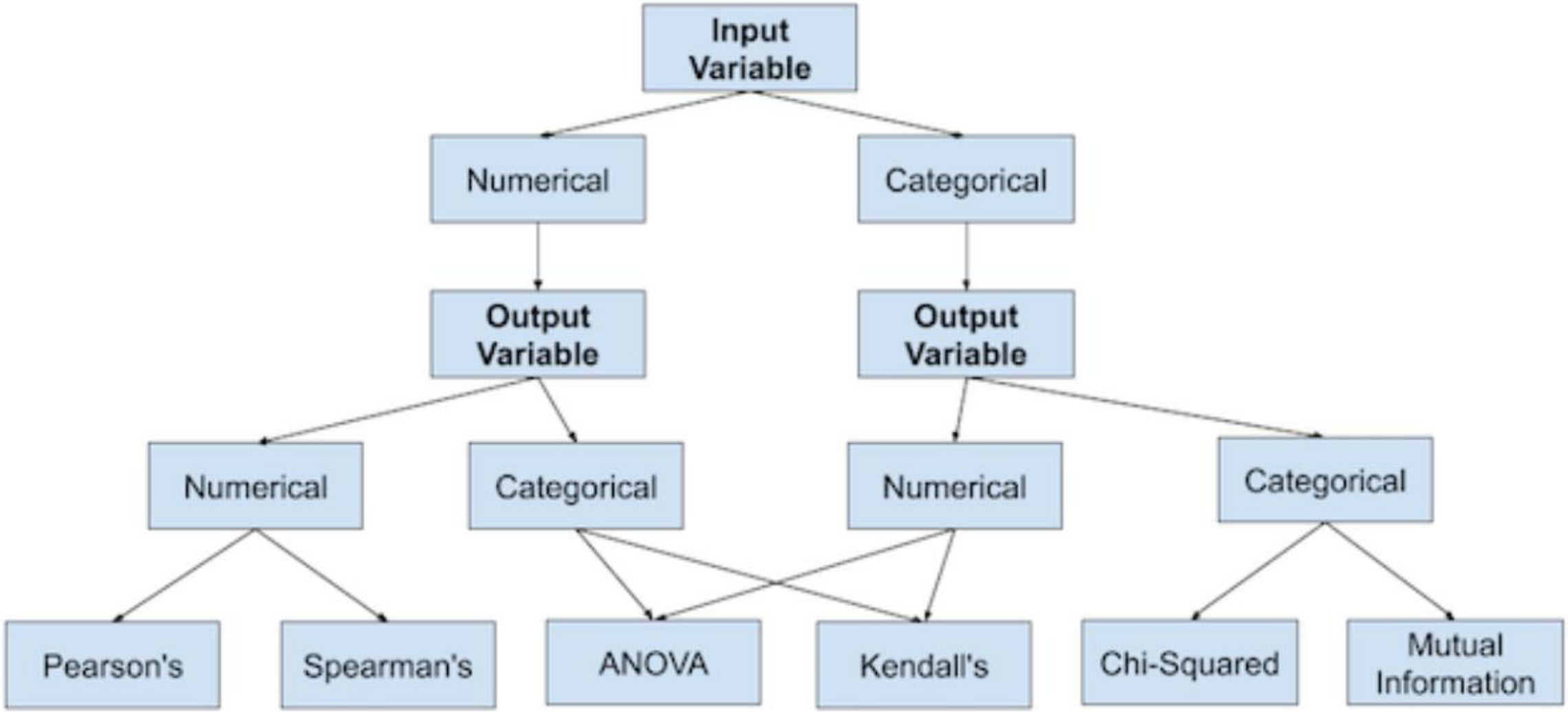


Part I: Filter Methods

- Evaluate the relationship between each input feature and the target variable using statistical tests.
- Fast and effective for getting an initial overview.



Diagrams Filter methods



1. Categorical Input & Categorical Output

1.1 Chi-Squared (χ^2)

Definition: A statistical hypothesis test to determine if there is a significant association between two categorical variables.

Hypotheses:

- **H0 (Null Hypothesis)**: The two variables are independent (the feature is not related to the target).
- **H1 (Alternative Hypothesis)**: The two variables are dependent (the feature is related to the target).

How it works: Calculates the χ^2 statistic for each feature. A higher χ^2 value provides more evidence to reject H0, meaning the feature is useful for the model.

Formula: $\chi^2 = \sum \frac{(O-E)^2}{E}$

(Where O is the observed frequency and E is the expected frequency).

Advantages: Simple, fast, easy to interpret.

Disadvantage: Requires expected frequencies not to be too small.

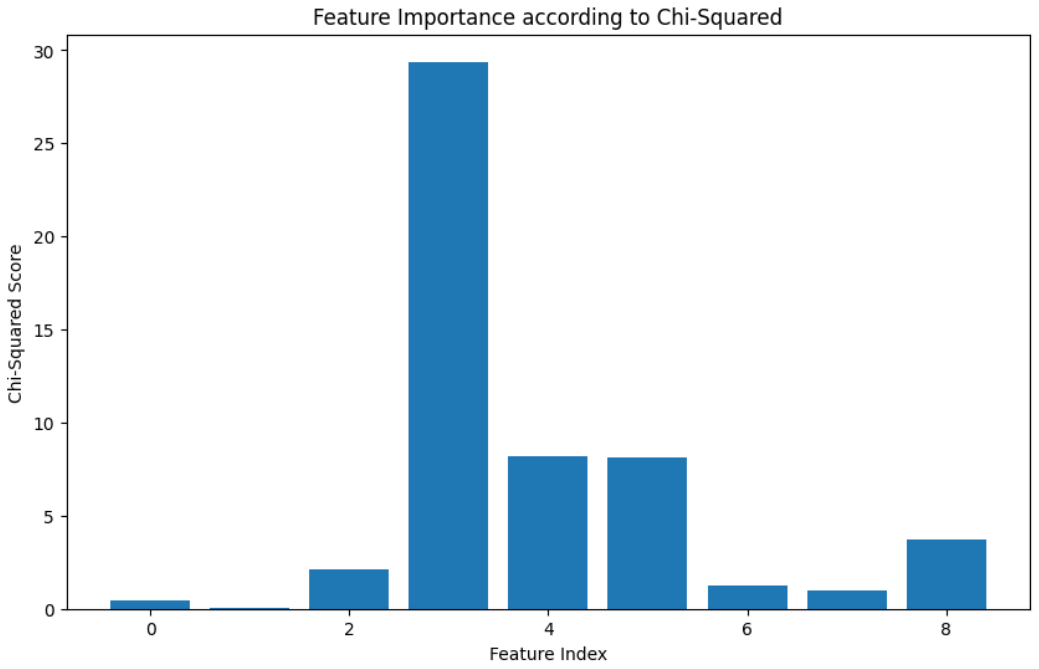
1.3. Tools with Scikit-learn

```
from sklearn.feature_selection import SelectKBest
```

K: ***int or "all", default=10***
Number of top features to select. The "all" option bypasses selection, for use in a parameter search.

```
SelectKBest(score_func=chi2, k='all')
```

f_classif
ANOVA F-value between label/feature for classification tasks.
mutual_info_classif
Mutual information for a discrete target.
chi2
Chi-squared stats of non-negative features for classification tasks.
f_regression
F-value between label/feature for regression tasks.
mutual_info_regression
Mutual information for a continuous target….



Lab 1 Applying the Chi-Squared Filter Method

Breast Cancer Dataset

```
Indices of 4 selected features: [np.int64(3), np.int64(4), np.int64(5),
np.int64(8)] Accuracy with SelectKBest(chi2, k=4): 0.747
```

2. Numerical Input & Categorical Output

2.1. ANOVA F-Statistic

Definition: The ANOVA F-test is a statistical test used to determine if there is a significant difference in the mean of a numerical variable across two or more groups. If a feature's mean value varies significantly across classes, it is likely a good predictor.

Formula: $F = \frac{\text{variance between groups}}{\text{variance within groups}}$

The larger the F-value, the more distinct the groups are, and the more important the feature.

When to Use?

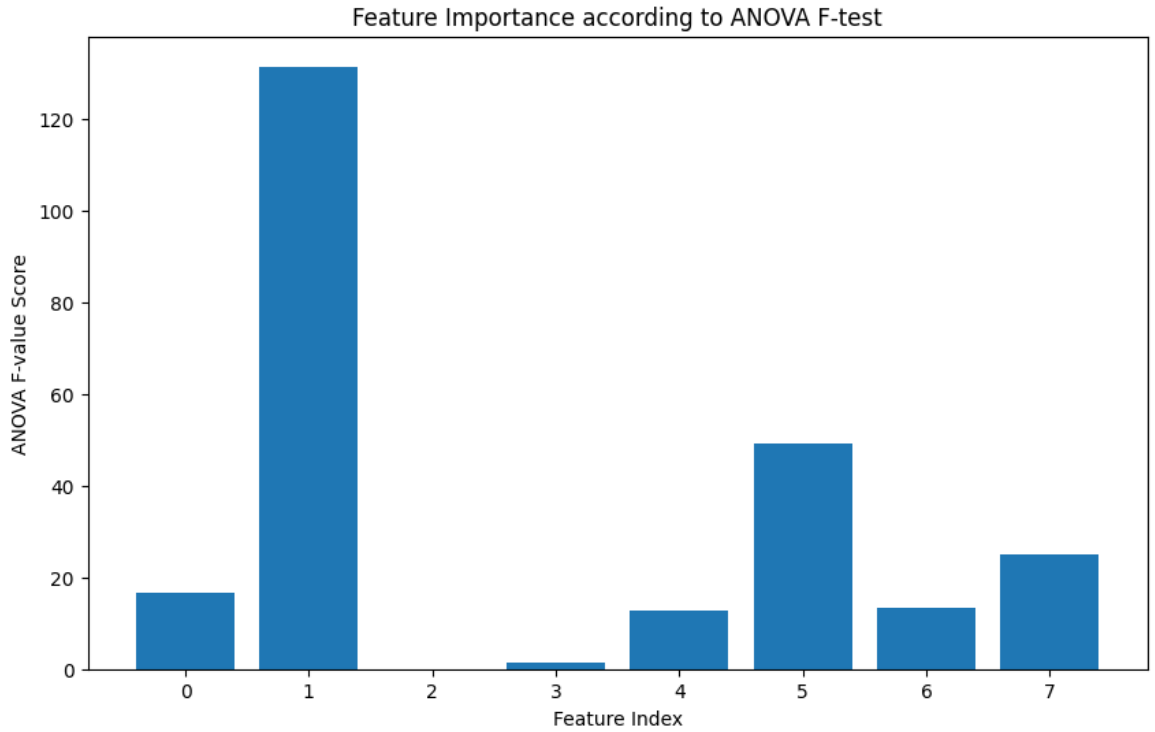
Use when you have a classification problem and numerical input features (e.g., 'age', 'income', 'temperature').

- **Pros:** Very effective at finding linear relationships (differences in means).

- **Cons:** Assumes that the data is normally distributed and has equal variance among groups (though it's fairly robust in practice).

Lab 2: Applying ANOVA F-test

Pima Indians Diabetes Dataset Introduction: This dataset records medical metrics for women of the Pima Indian tribe and aims to predict whether a person has diabetes. The input features are numerical, and the target is categorical.




```
SelectKBest(score_func=f_classif, k=4)
```

3. Numerical Input & Numerical Output

Common methods include:

- **Pearson's Correlation Coefficient:** Measures linear relationships.
- **Spearman's Rank Coefficient:** Measures monotonic relationships (including non-linear).
- **Mutual Information:** Measures general dependency.

3. Numerical Input & Numerical Output

3.1. Pearson's Correlation Coefficient

Definition: Pearson's correlation coefficient measures the **linear** relationship between two numerical variables. Its value ranges from -1 to +1.

Formula: $r_{xy} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{i=1}^{n}(y_i-\bar{y})^2}}$

When to Use?

Use for regression problems with numerical inputs, assuming a linear relationship.

- **Pros:** Fast, easy to understand, provides both magnitude and direction of the relationship.

- **Cons**: Only captures linear relationships.

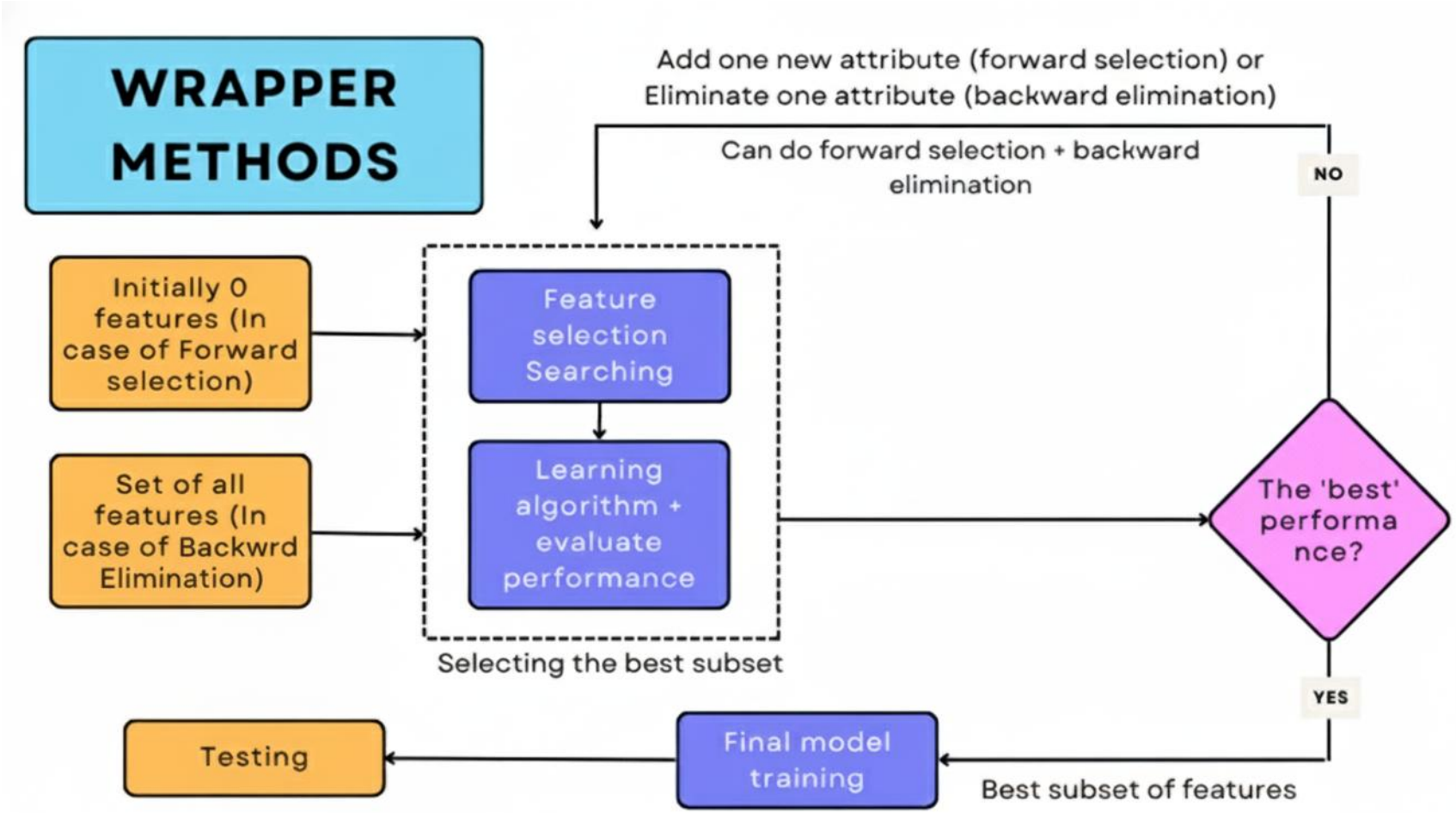
The Filter Method Decision Matrix

Input Variable Type	Output Variable Type	Recommended Test	Scikit-learn Function
Categorical	**Categorical**	Chi-Squared (χ^2), Mutual Information	chi2, mutual_info_classif
Numerical	**Categorical**	ANOVA F-test, Mutual Information	f_classif, mutual_info_classif
Numerical	**Numerical**	Pearson Correlation	f_regression

Lab 3: Applying Filter Methods for Regression Problems

Part II: Wrapper Methods

- Search for feature subsets based on the performance of a specific predictive model.
- "Wraps" around a machine learning algorithm, using it to evaluate each subset.



Search Strategies

- Since testing all possible feature subsets is computationally infeasible (for N features, there are $2^N - 1$ subsets), wrapper methods use heuristic search strategies to efficiently find a good subset:

- **Stepwise Methods:** Simpler methods that involve systematically adding or removing features from the model until no further improvement is observed. Examples include:
  - **Forward Selection**: Starts with an empty set and iteratively adds the feature that improves model performance the most.
  - **Backward Elimination:** Starts with all features and iteratively removes the least important feature. RFE is a prime example of this strategy.

Recursive Feature Elimination (RFE)

Principle: A wrapper-style feature selection method that works by recursively building a model and removing the weakest feature(s) at each step until the desired number of features is reached.

RFE Workflow:

1. Train a model on the entire set of features (e.g., LogisticRegression, DecisionTreeClassifier). The model must provide a measure of feature importance.
2. Find and eliminate the feature with the lowest importance.
3. Repeat the process with the remaining features until the number of features is reduced to k (the desired number).

Pros: Often results in better model performance than filter methods because it considers feature interactions.

Cons: High computational cost, risk of overfitting if not used carefully with cross-validation.

Tools with sklearn


```
from sklearn.feature_selection import RFE
```

```
RFE(estimator=DecisionTreeClassifier(), n_features_to_select=5, step=1)
```

Estimator: ***Estimator instance***

A supervised learning estimator with a fit method that provides information about feature importance (e.g. coef_, feature_importances_).

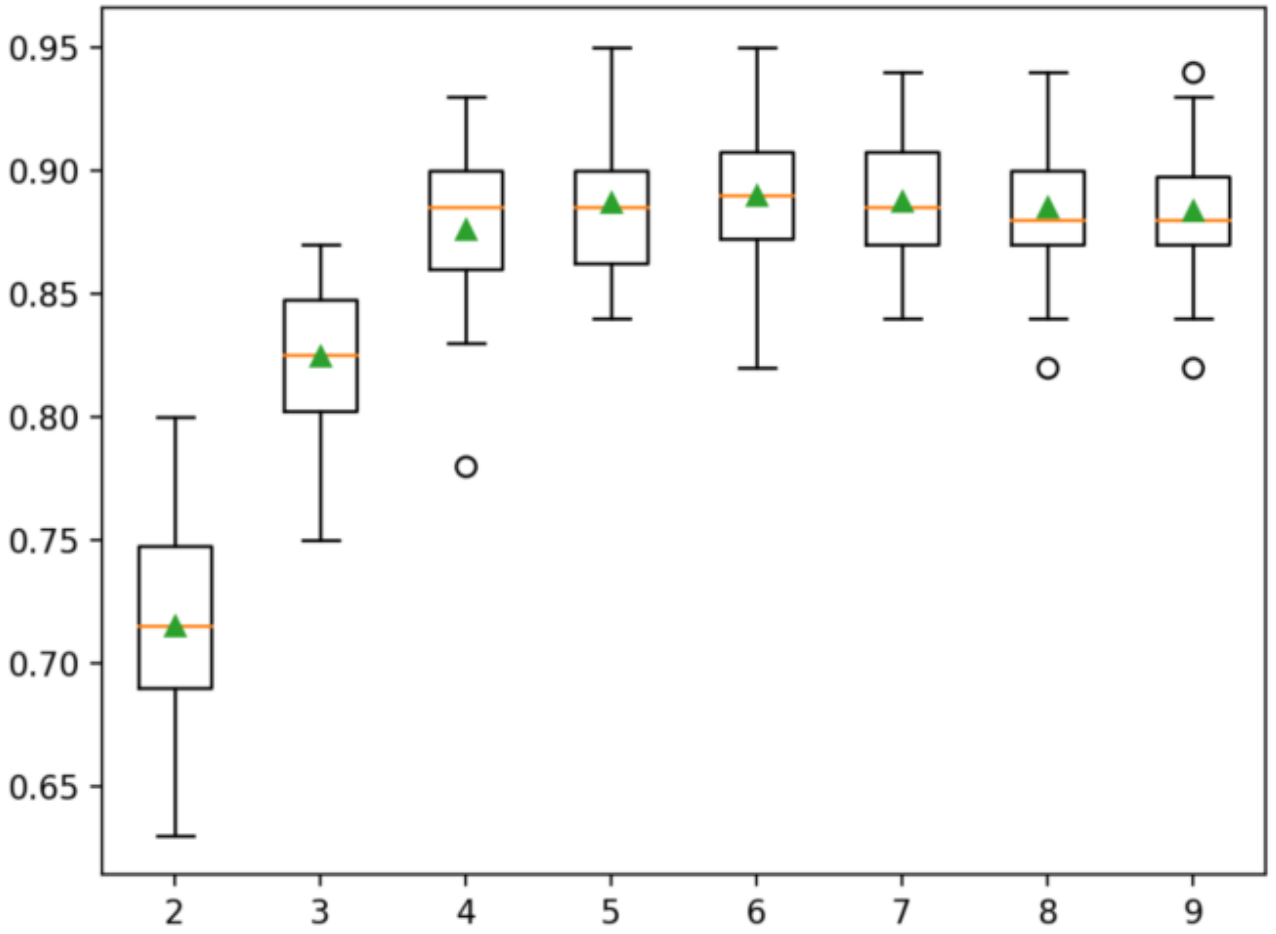
n_features_to_select: ***int or float, default=None***

The number of features to select. If None, half of the features are selected. If integer, the parameter is the absolute number of features to select. If float between 0 and 1, it is the fraction of features to select.

Step: ***int or float, default=1***

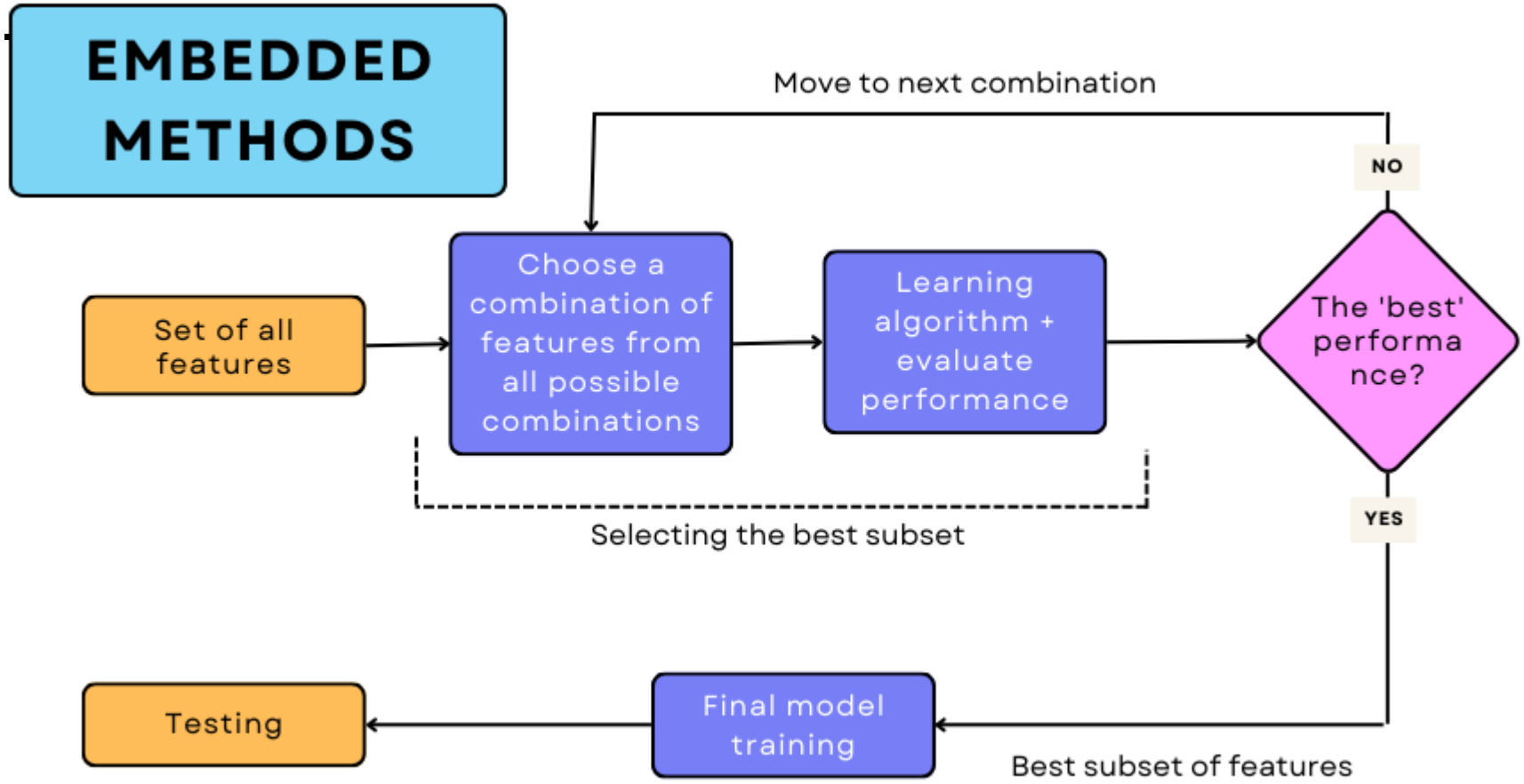
If greater than or equal to 1, then step corresponds to the (integer) number of features to remove at each iteration. If within (0.0, 1.0), then step corresponds to the percentage (rounded down) of features to remove at each iteration.

Lab 4: Feature Selection with RFE



Part III: Embedded Methods

- Feature selection is performed internally as part of the model's training process.
- Efficient and powerful.



Embedded Methods - Examples

1.Coefficients as Importance

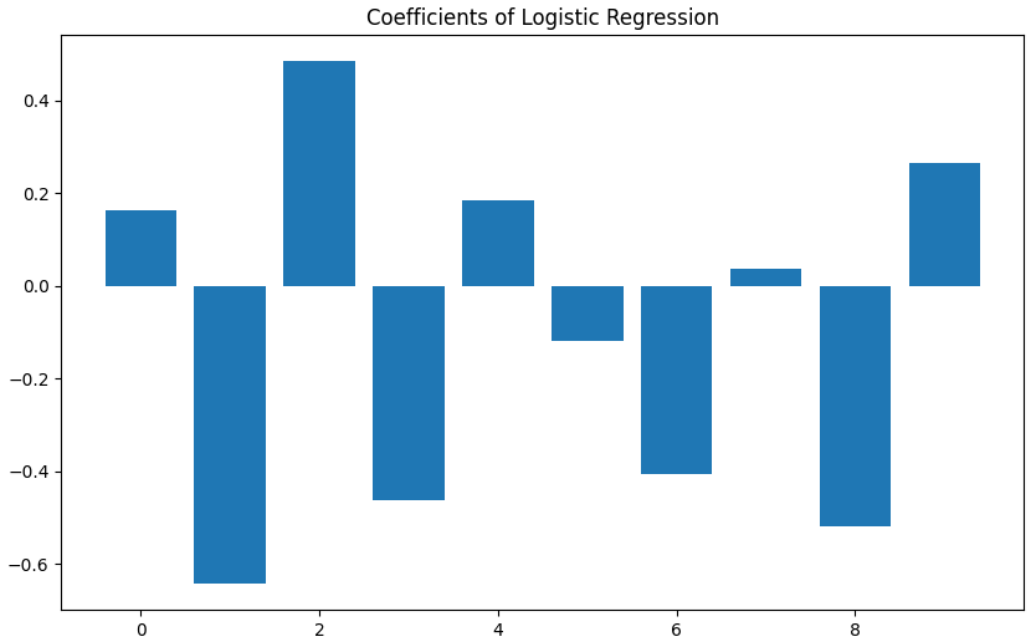
Definition: Linear models (e.g., **LogisticRegression**, **LinearRegression**) assign a coefficient to each feature. The absolute value of this coefficient can be used as an importance score.

Note: Data must be scaled for the coefficients to be fairly comparable.

```
X_clf, y_clf =
make_classification(n_samples=1000,
n_features=10, n_informative=5,
n_redundant=5, random_state=1)

model_lr = LogisticRegression(solver='lbfgs')
model_lr.fit(X_clf, y_clf)

importance_lr = model_lr.coef_[0]
```



Embedded Methods - Examples

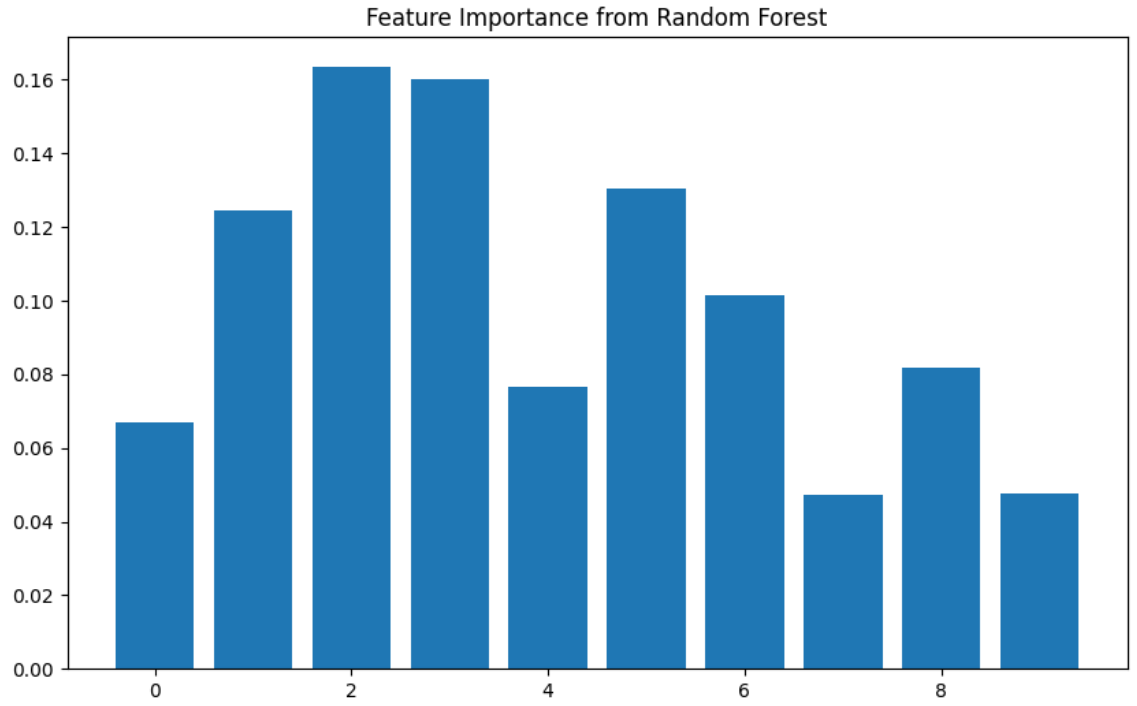
2. Decision Tree Importance

Definition: Models like **DecisionTree** and **RandomForest** calculate importance based on how much a feature improves the "purity" of nodes in the tree (e.g., by reducing Gini impurity) each time it is selected for a split.

Note: When using tree-based models. It's very powerful as it can capture non-linear interactions. However, it can be biased towards high cardinality categorical features and numerical features.

```
model_rf =
RandomForestClassifier(random_state=1)
model_rf.fit(X_clf, y_clf)

importance_rf =
model_rf.feature_importances_
```



Embedded Methods - Examples

3. Permutation Importance

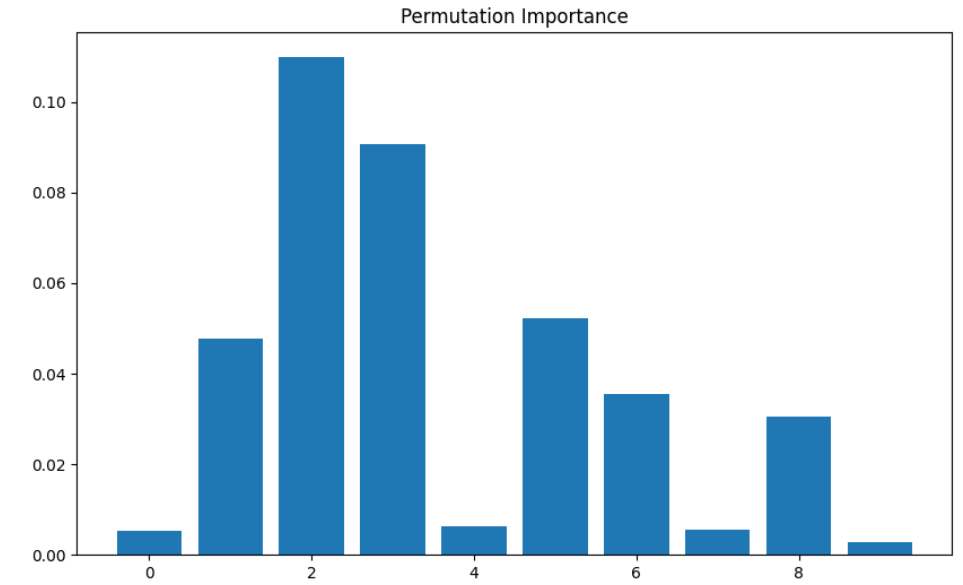
Definition: A model-agnostic method. It works by randomly shuffling the values of a single feature and measuring the resulting decrease in model performance.

How it works: The feature that causes the largest performance drop when shuffled is considered the most important. This is one of the most reliable methods.

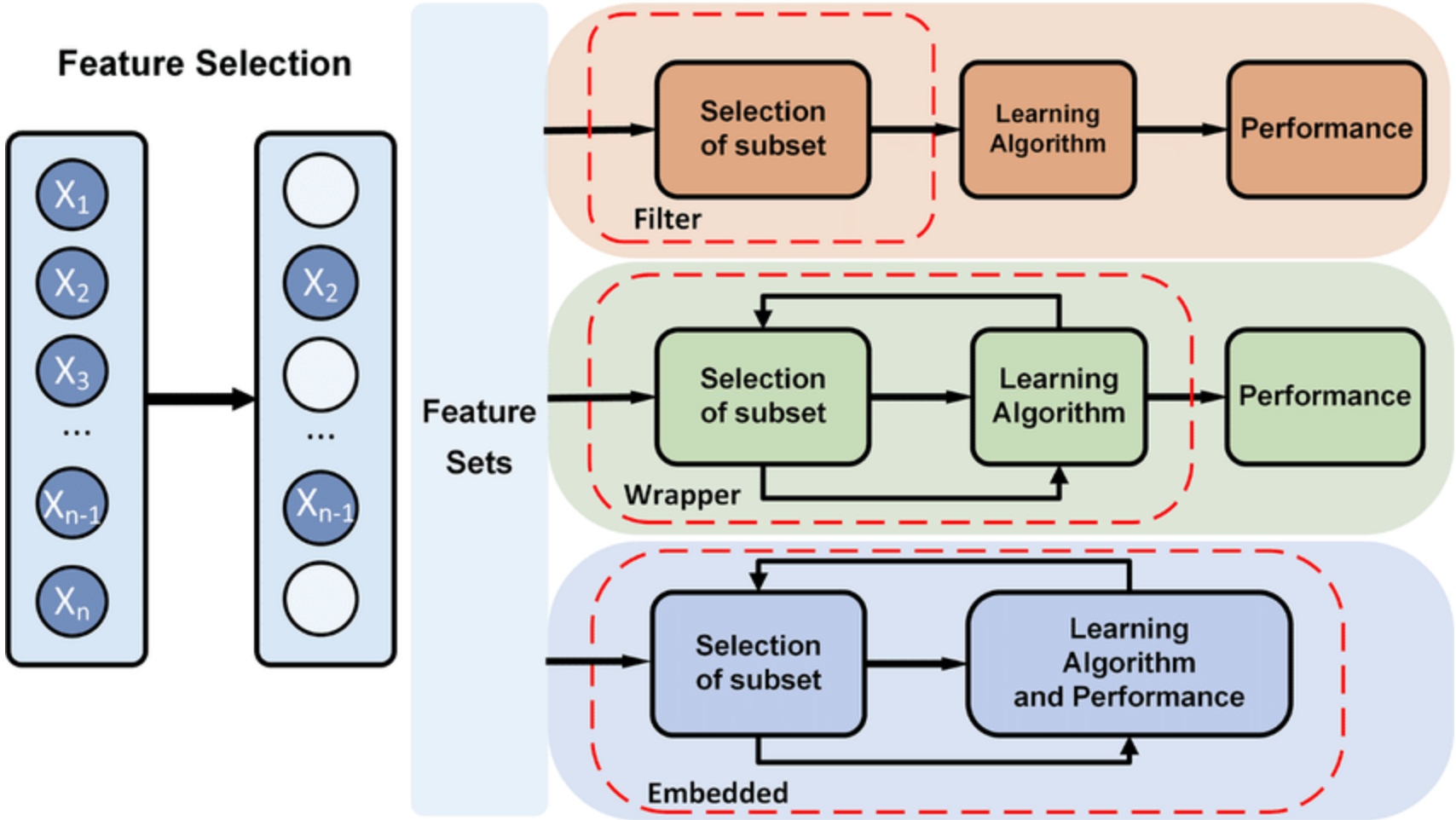
```
from sklearn.inspection import permutation_importance

model_rf = RandomForestClassifier(random_state=1)
model_rf.fit(X_clf, y_clf)

results_perm = permutation_importance(model_rf,
                                      X_clf, y_clf,
                                      scoring='accuracy',
                                      n_repeats=10,
                                      random_state=1,
                                      n_jobs=-1)
importance_perm = results_perm.importances_mean
```



Feature Selection



Summary

Method Type	Method Name	Input Type	Output Type	Pros	Cons
Filter	**Chi-Squared**	Categorical	Categorical	Fast, simple	Only indicates independence, not strength
	ANOVA F-test	Numerical	Categorical	Fast, effective for linear relationships	Relies on data distribution assumptions
	Pearson's Corr.	Numerical	Numerical	Fast, gives direction and strength	Only captures linear relationships
	Mutual Info.	Any	Any	Powerful, captures non-linear relationships	Needs more data, computationally slower
Wrapper	**RFE**	Any	Any	High performance, considers feature interactions	Very computationally expensive, risk of overfitting
Embedded	**Feature Importance**	Any	Any	Integrated into the model, efficient	Depends on the chosen model

General Advice

1. **Start with Filter methods** to get a quick overview and eliminate obviously irrelevant features.
2. **Use Embedded methods** (like Feature Importance from RandomForest) to get a more reliable feature ranking.
3. **If performance is the top priority** and you have sufficient computational resources, **use Wrapper methods** like RFECV to fine-tune the final feature set.