Feature Selection

Mohammad Hosseini

Outline

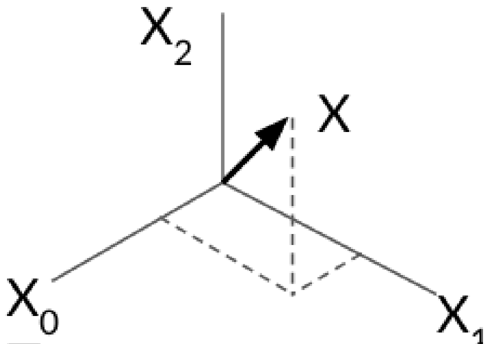
- Introduction to Feature Spaces
- Introduction to Feature Selection
- Filter Methods
- Wrapper Methods
- Embedded Methods

Feature space

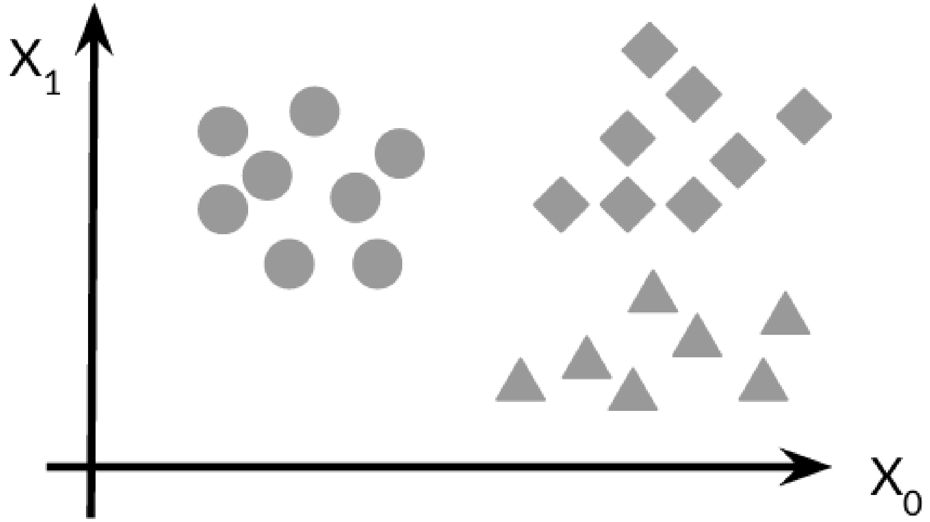
- N dimensional space defined by your N features
- Not including the target label

$$X = \begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_d \end{bmatrix}$$

Feature vector



Feature space (3D)



Scatter plot (2D)

Feature space

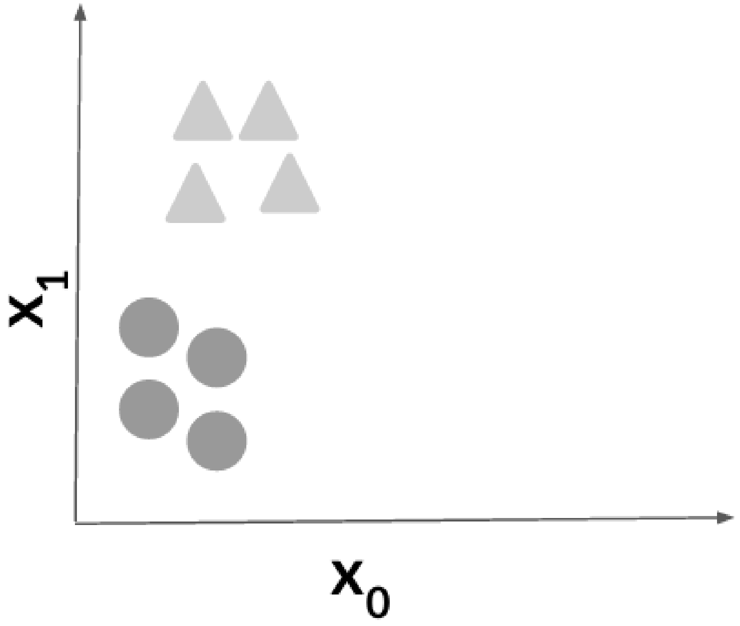
3D Feature Space

No. of Rooms X_0	Area X_1	Locality X_2	Price Y
5	1200 sq. ft	New York	$40,000
6	1800 sq. ft	Texas	$30,000

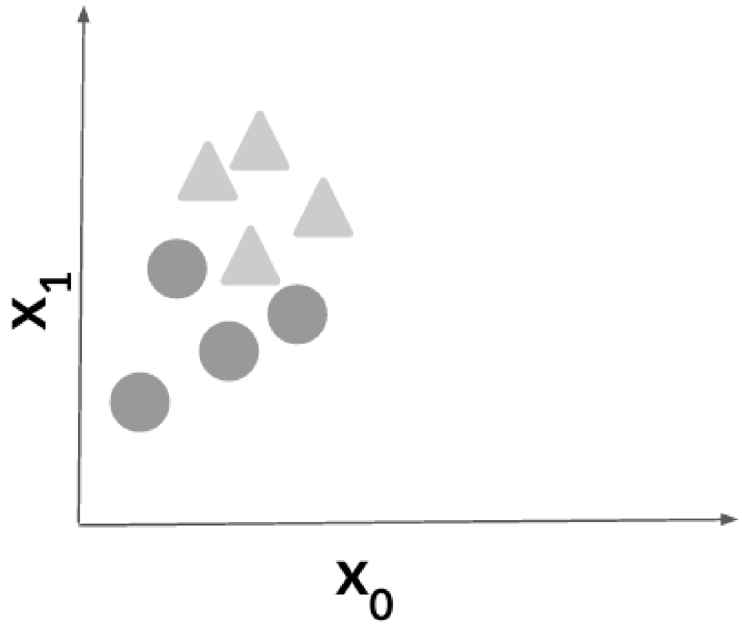
$Y = \mathbf{f}(X_0, X_1, X_2)$

f is your ML model acting on feature space X_0, X_1, X_2

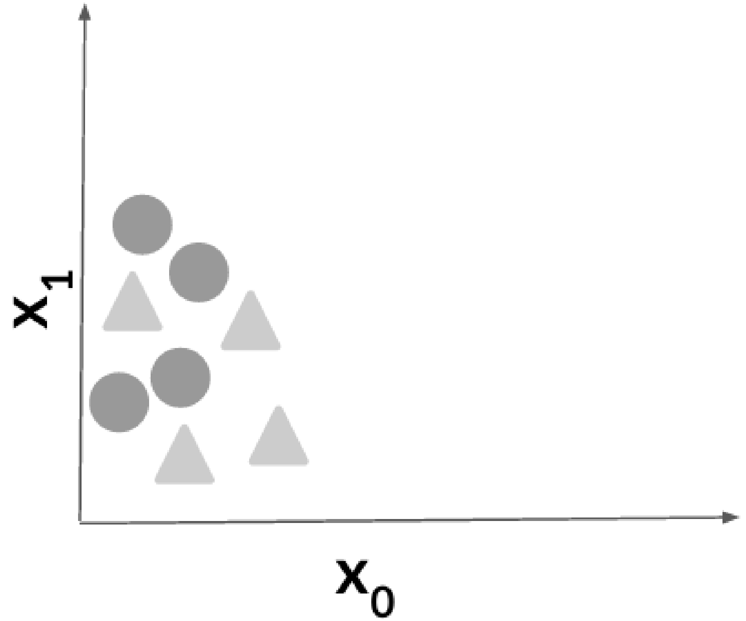
2D Feature space - Classification



Ideal

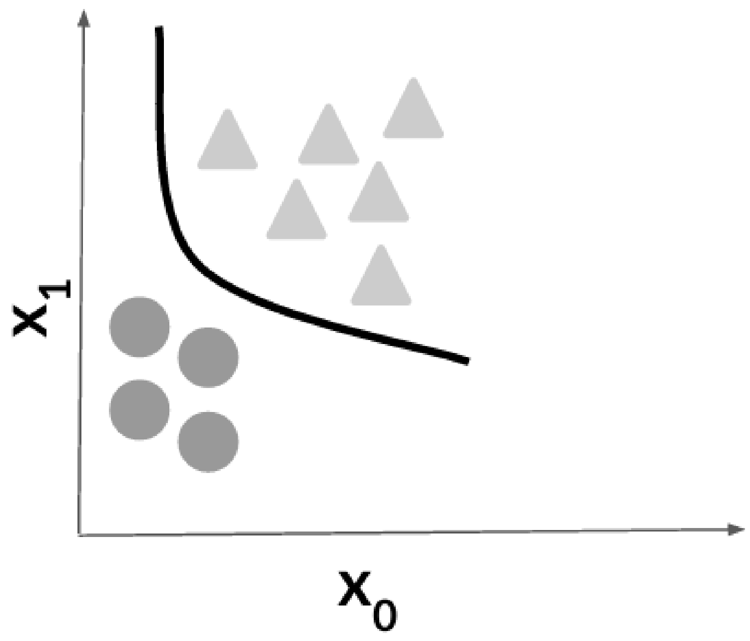


Realistic



Poor

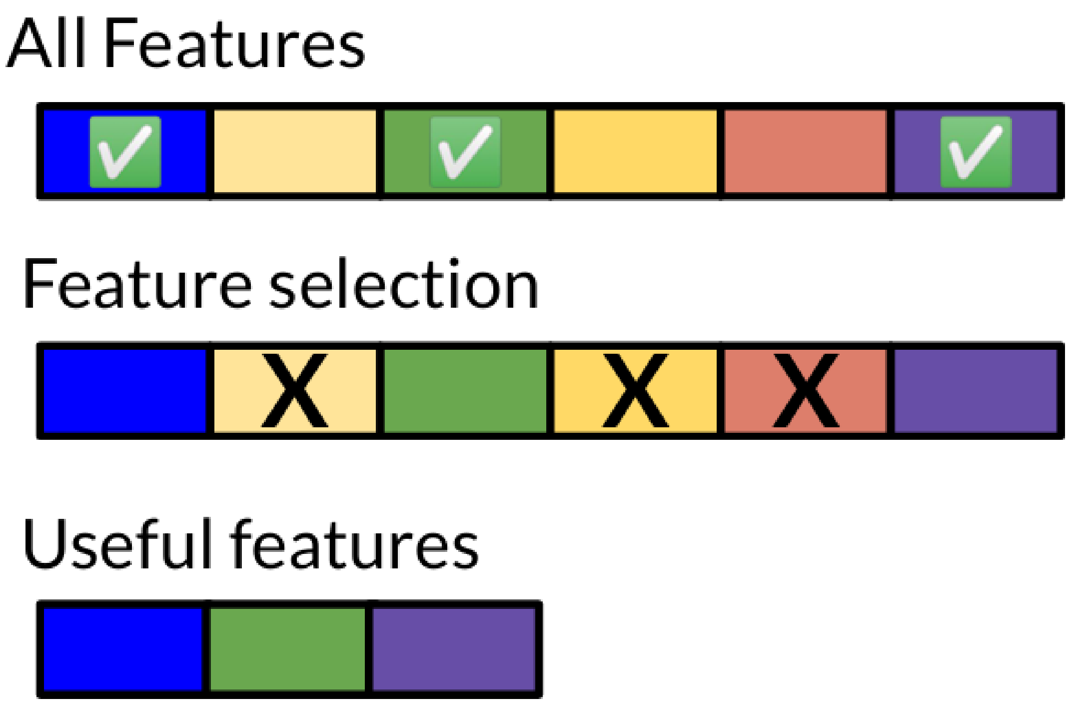
Drawing decision boundary



Model learns decision boundary

Boundary used to classify data points

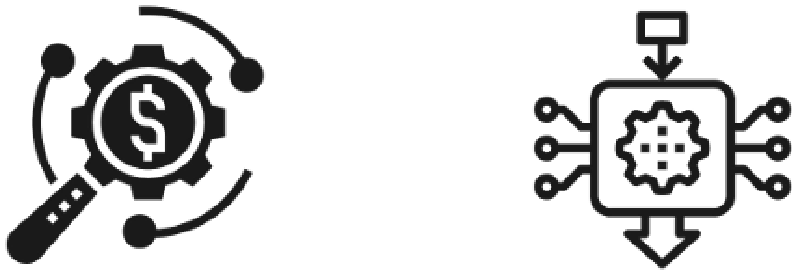
Feature selection



- Identify features that best represent the relationship
- Remove features that don't influence the outcome
- Reduce the size of the feature space
- Reduce the resource requirements and model complexity

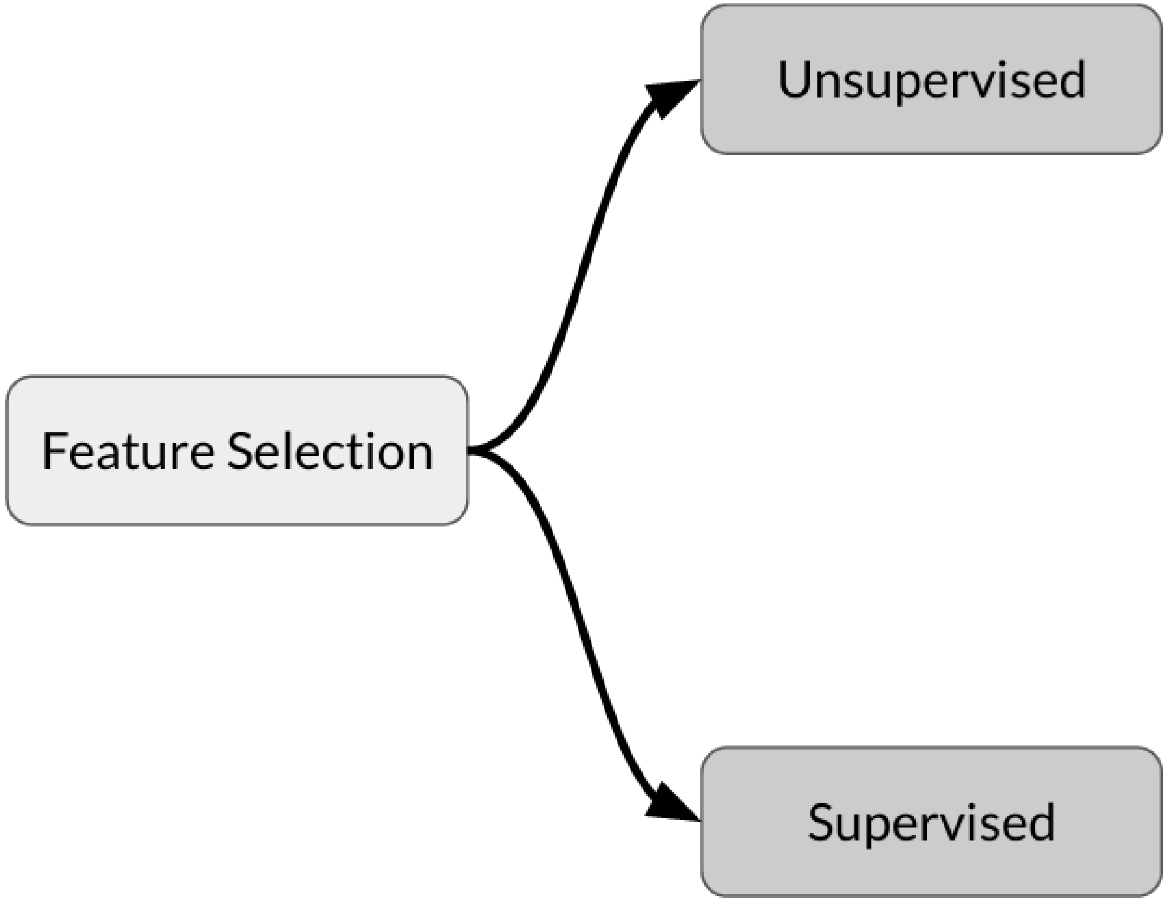
Why is feature selection needed?

Reduce storage and I/O requirements



Minimize training and inference costs

Feature selection methods



Unsupervised feature selection

1. **Unsupervised**

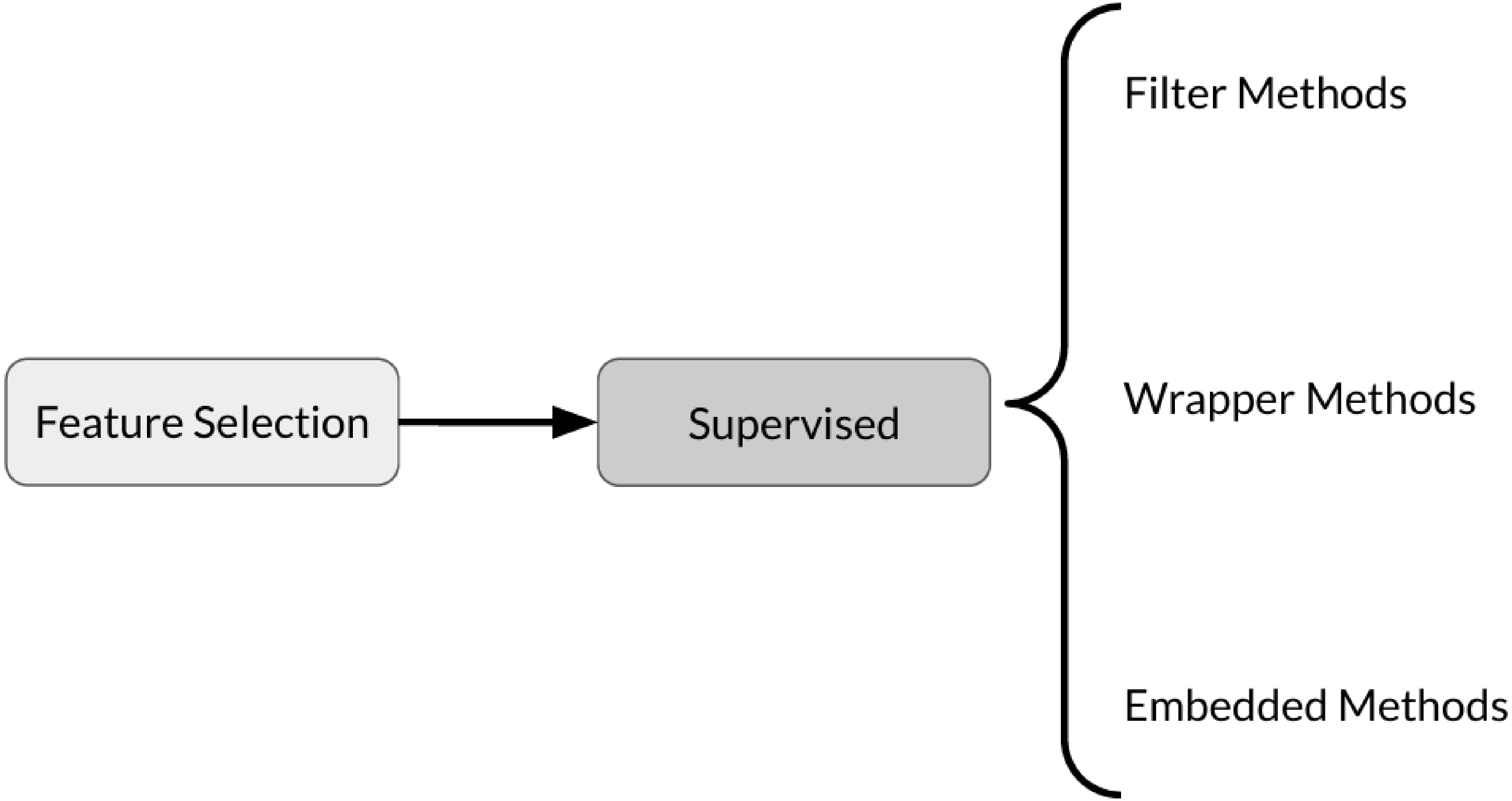
- Features-target variable relationship not considered
- Removes redundant features (correlation)

Supervised feature selection

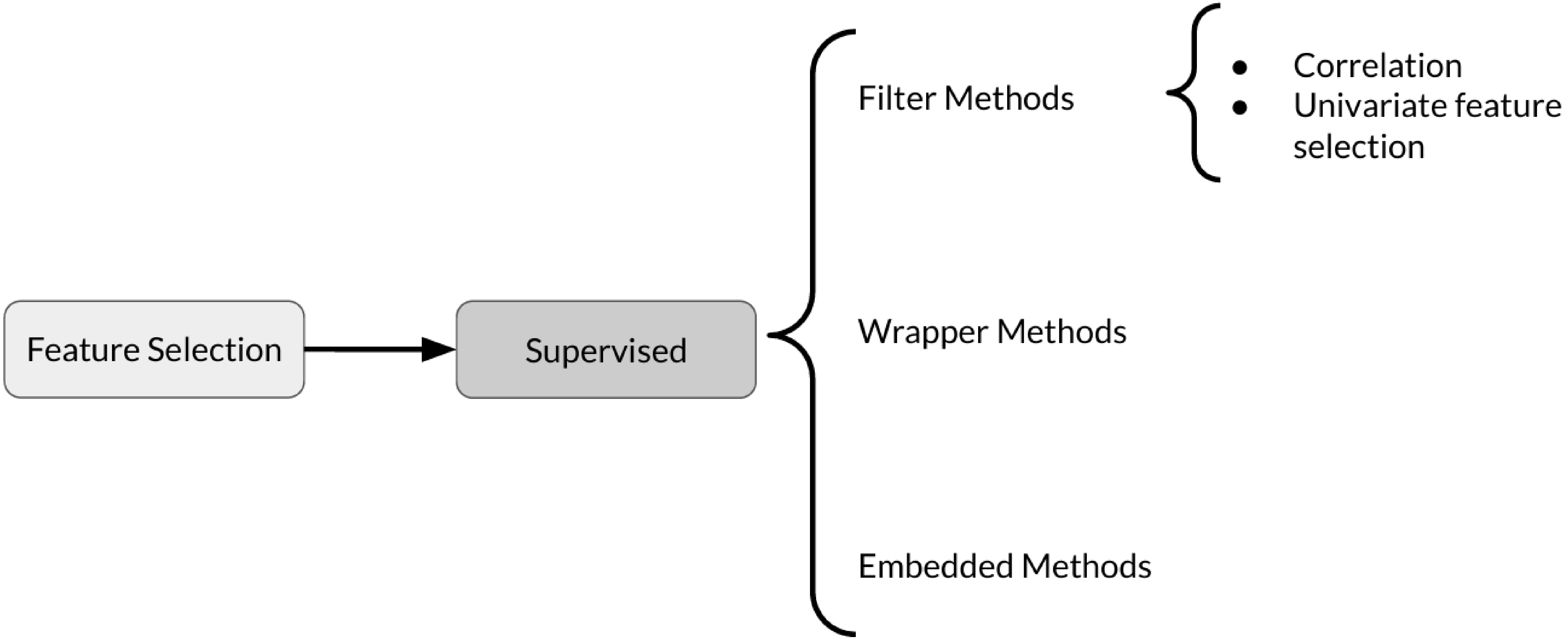
2. **Supervised**

- Uses features-target variable relationship
- Selects those contributing the most

Supervised methods



Filter methods



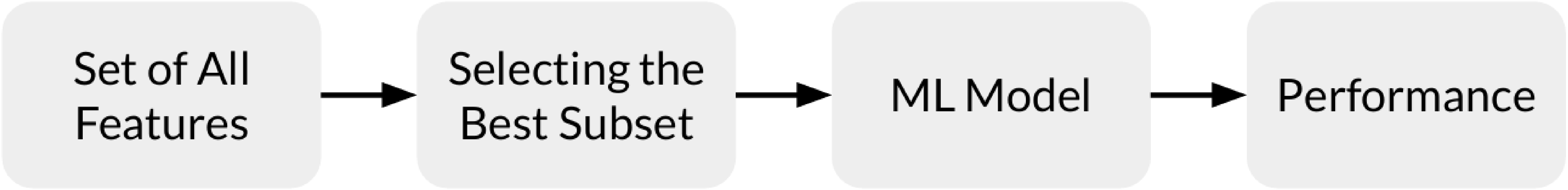
Filter methods

- Correlated features are usually redundant
    - Remove them!

Popular filter methods:

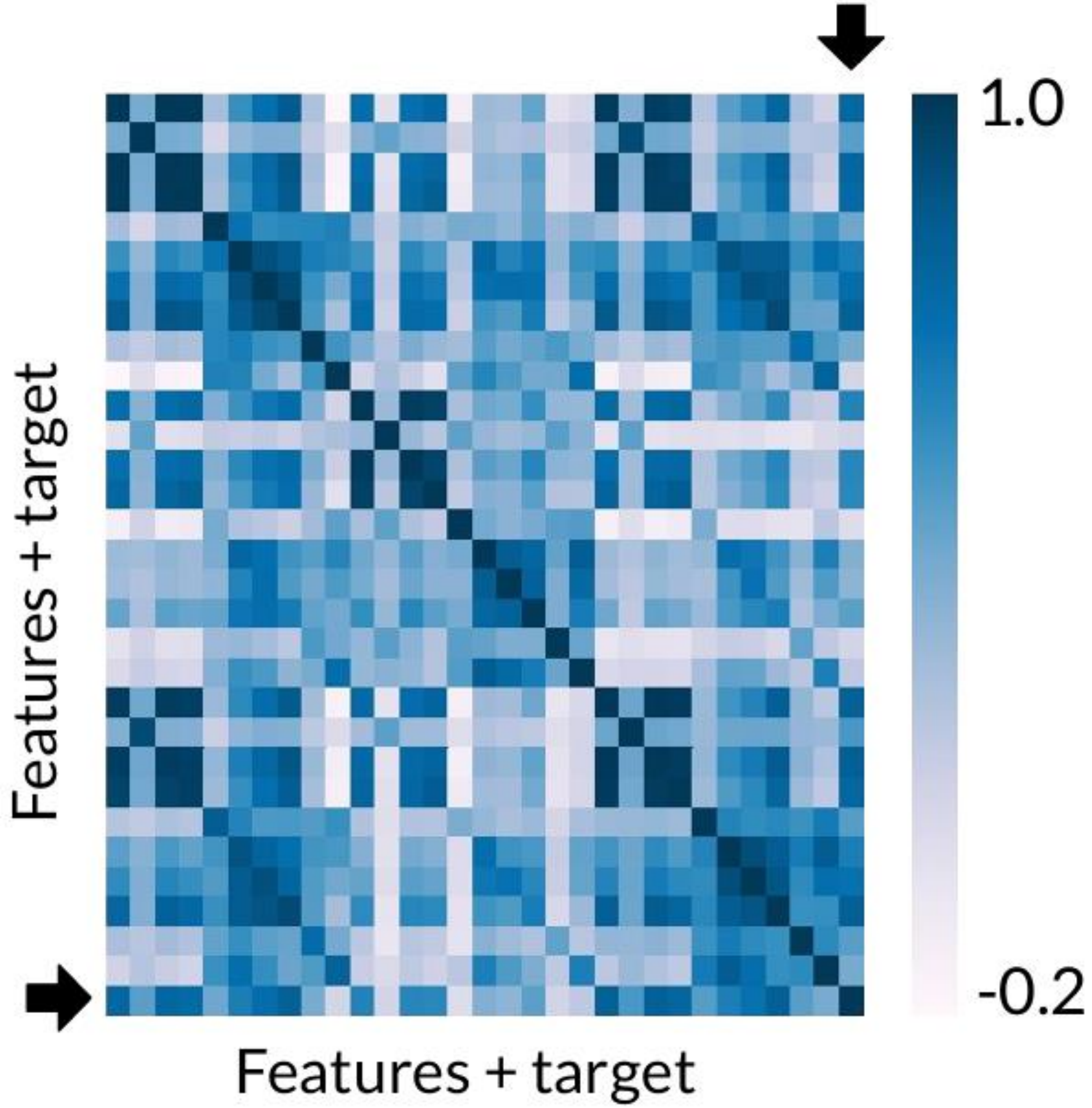
- Pearson Correlation
    - Between features, and between the features and the label
- Univariate Feature Selection

Filter methods



Correlation matrix

- Shows how features are related:
  - To each other (Bad)
  - And with target variable (Good)
- Falls in the range [-1, 1]
  - 1 High positive correlation
  - -1 High negative correlation



Feature comparison statistical tests

- Pearson’s correlation: Linear relationships
- Kendall Tau Rank Correlation Coefficient: Monotonic relationships & small sample size
- Spearman’s Rank Correlation Coefficient: Monotonic relationships

Other methods:

- Mutual information
- F-Test
- Chi-Squared test

Univariate feature selection in SKLearn

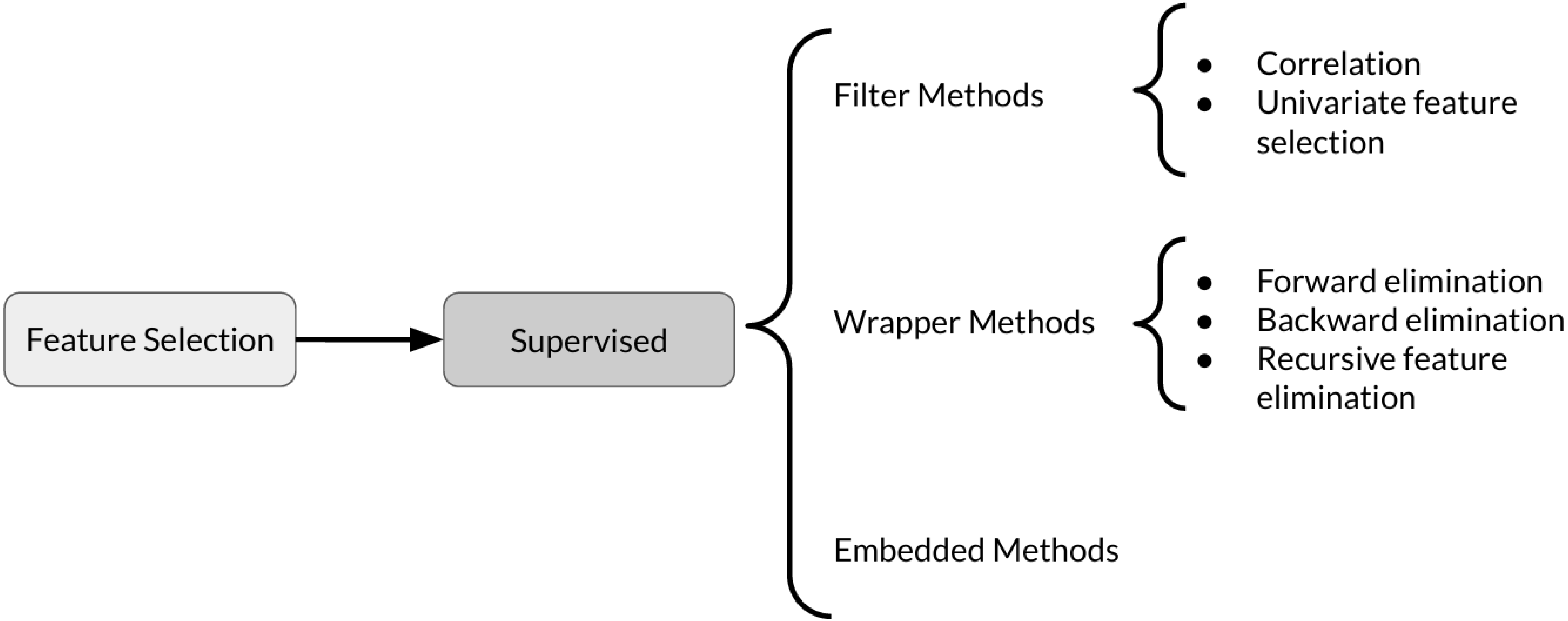
SKLearn Univariate feature selection routines:

1. **`SelectKBest`**
2. `SelectPercentile`
3. `GenericUnivariateSelect`

Statistical tests available:

- Regression: `f_regression`, `mutual_info_regression`
- Classification: `chi2`, `f_classif`, `mutual_info_classif`

Wrapper methods



Wrapper methods

Popular wrapper methods

1. Forward Selection
2. Backward Selection
3. Recursive Feature Elimination

Forward selection

1. Iterative, greedy method
2. Starts with 1 feature
3. Evaluate model performance when **adding** each of the additional features, one at a time
4. Add next feature that gives the best performance
5. Repeat until there is no improvement

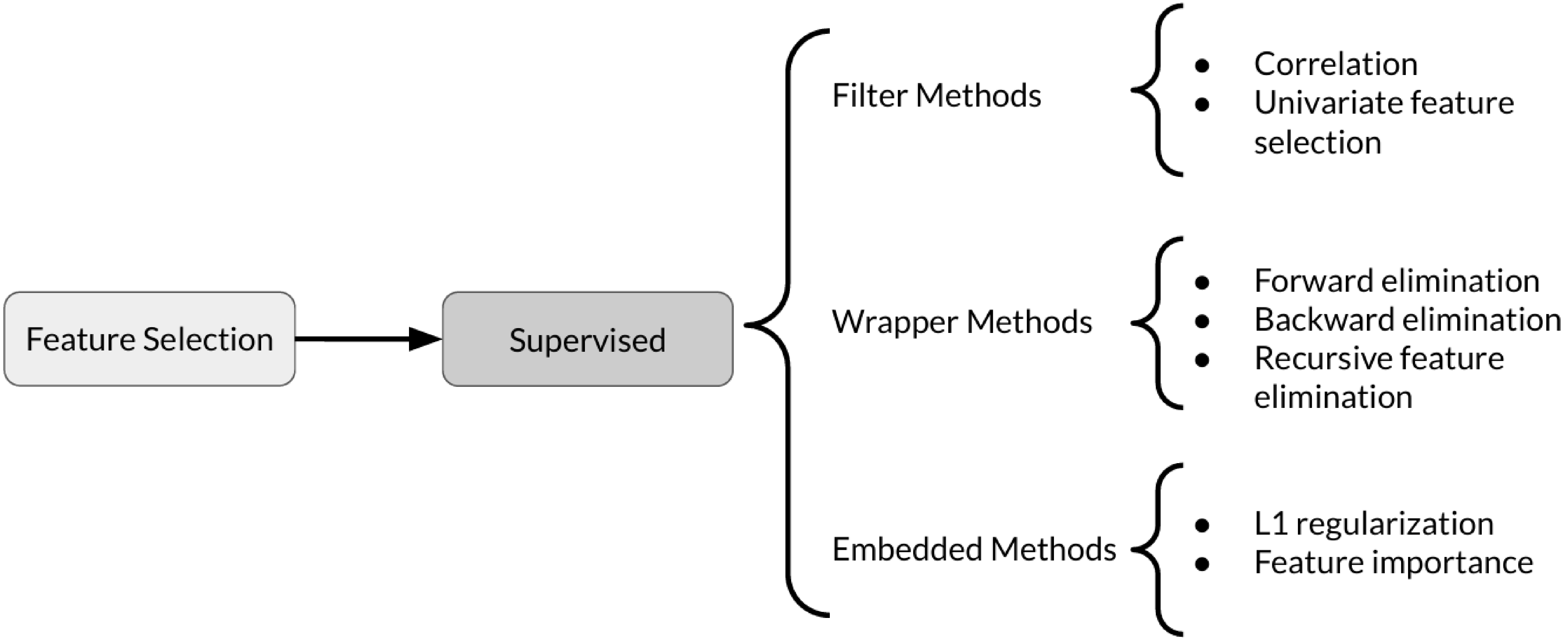
Backward elimination

1. Start with all features
2. Evaluate model performance when **removing** each of the included features, one at a time
3. Remove next feature that gives the best performance
4. Repeat until there is no improvement

Recursive feature elimination (RFE)

1. Select a model to use for evaluating feature importance
2. Select the desired number of features
3. Fit the model
4. Rank features by importance
5. Discard least important features
6. Repeat until the desired number of features remains

Embedded methods



Feature importance

- Assigns scores for each feature in data
- Discard features scored lower by feature importance

Feature importance with SKLearn

- Feature Importance class is in-built in Tree Based Models (eg., `RandomForestClassifier`)
- Feature importance is available as a property `feature_importances_`
- *We can then use* `SelectFromModel` *to select features from the trained model based on assigned feature importances.*

Extracting feature importance

```
def feature_importances_from_tree_based_model_():

  X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2,
                                                      stratify=Y, random_state = 123)
  model = RandomForestClassifier()
  model = model.fit(X_train,Y_train)

  feat_importances = pd.Series(model.feature_importances_, index=X.columns)
  feat_importances.nlargest(10).plot(kind='barh')
  plt.show()

  return model
```

Model Performance Evaluation

Actual Value

Predicated Value		Positive	Negative
	Positive	5600	600
	Negative	500	3300

Confusion Matrix

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$f1 - score = \frac{2.precision.recall}{precision + recall}$$

Model Performance Evaluation