```
# separate features and target
X = data.data
y = data.target

# create the pipeline
estimators = [
    ('standardize' , StandardScaler()),
    ('svc', SVC())
]

# build the pipeline model
pipe = Pipeline(estimators)

# run the pipeline
kfold = KFold(n_splits=3, shuffle=True)
cv_result = cross_val_score(pipe, X, y, cv=kfold)

# evaluate the model performance
print("Accuracy: %.3f%% (%.3f%%)" % (cv_result.mean()*100.0, cv_result.
std()*100.0))
'Output':
Accuracy: 94.667% (0.943%)
```

Pipelines Using make_pipeline

Another method for building machine learning pipelines is by using the **make_pipeline** method. For the next example, we use PCA to select the best six features and reduce the dimensionality of the dataset, and then we'll fit the model using Random forests for regression.

```
from sklearn.pipeline import make_pipeline
from sklearn.svm import SVR
from sklearn import datasets
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.decomposition import PCA
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestRegressor
```

```
# load dataset
data = datasets.load_boston()

# separate features and target
X = data.data
y = data.target

# build the pipeline model
pipe = make_pipeline(
    PCA(n_components=9),
    RandomForestRegressor()
)

# run the pipeline
kfold = KFold(n_splits=4, shuffle=True)
cv_result = cross_val_score(pipe, X, y, cv=kfold)

# evaluate the model performance
print("Accuracy: %.3f%% (%.3f%%)" % (cv_result.mean()*100.0, cv_result.
std()*100.0))
'Output':
Accuracy: 73.750% (2.489%)
```

Pipelines Using FeatureUnion

Scikit-learn provides a module for merging the output of several transformers called **feature_union**. It does this by fitting each transformer independently to the dataset, and then their respective outputs are combined to form a transformed dataset for training the model.

FeatureUnion works in the same way as a Pipeline, and in many ways can be thought of as a means of building complex pipelines within a Pipeline.

Let’s see an example using FeatureUnion. Here, we will combine the output of recursive feature elimination (RFE) and PCA for feature engineering, and then we’ll apply the Stochastic Gradient Boosting (SGB) ensemble model for regression to train the model.

```
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn import datasets
```