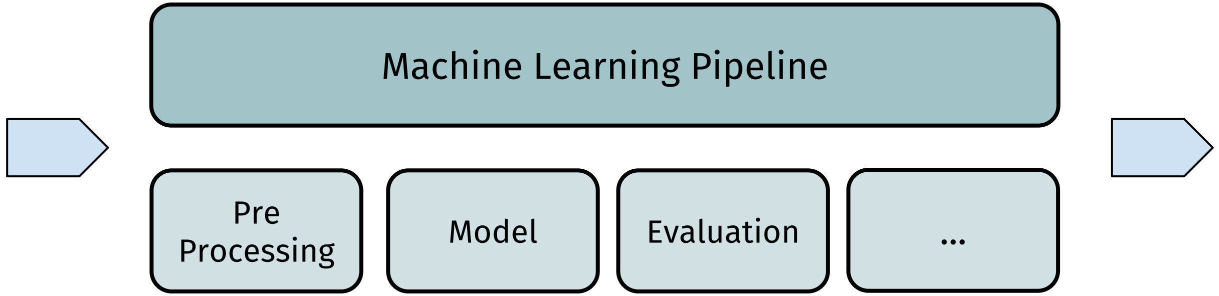
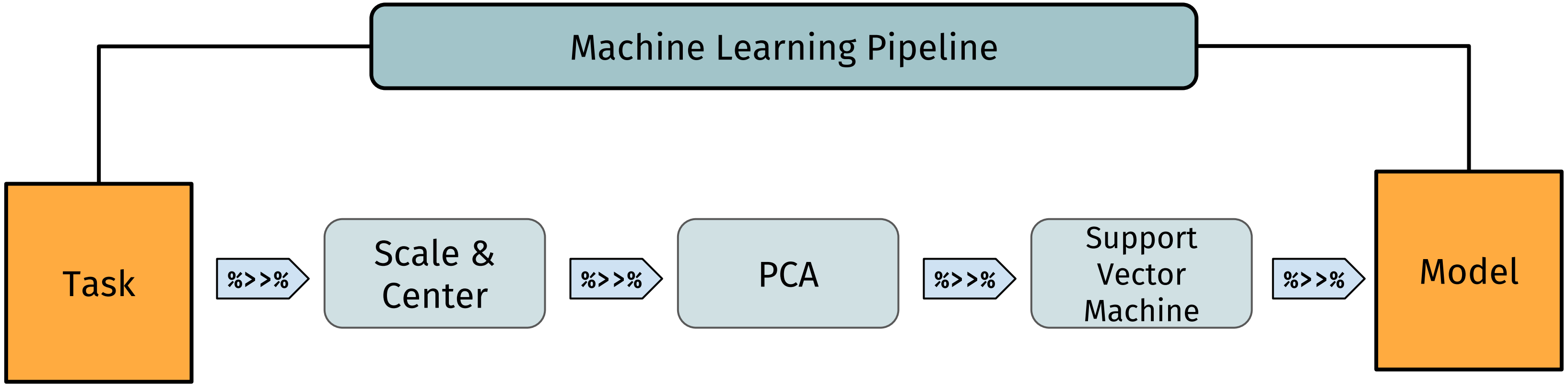
- Many Machine Learning Workflows consist of multiple steps, such as preprocessing, computing features, or imputing missing data
- This is often a long winded and complicated process, and properly separating train and test data is very difficult



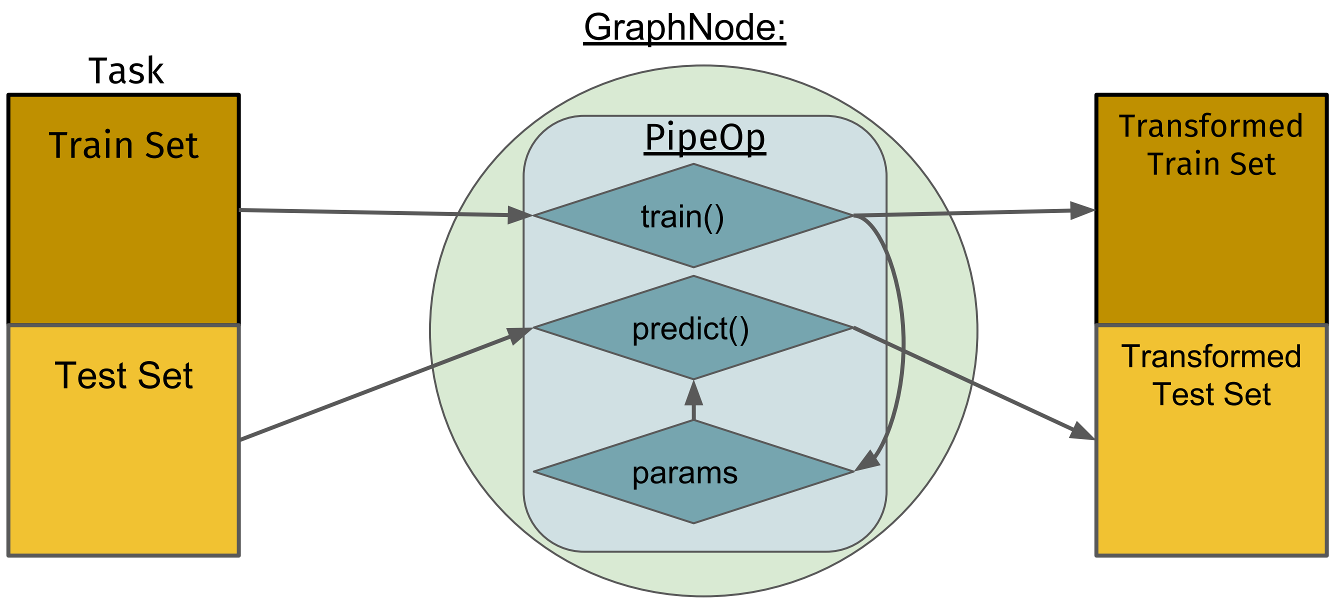
- Pipelines allow us to specify many difficult steps that are often undertaken in a few, concise lines
- By integrating pipelines with mlr3 and mlr3 tuning we can jointly tune over all hyperparameters the pipeline exposes.

- <u>Pipelines provide:</u>

    - Multiple widely used operations
      (Scaling, PCA, Variable Selection, Imputation, Stacking and many others)
    - A clean, extendable interface for custom pipeline operators
    - A simple operator connection operator: **%>>%**
    - An abstraction for parallelization



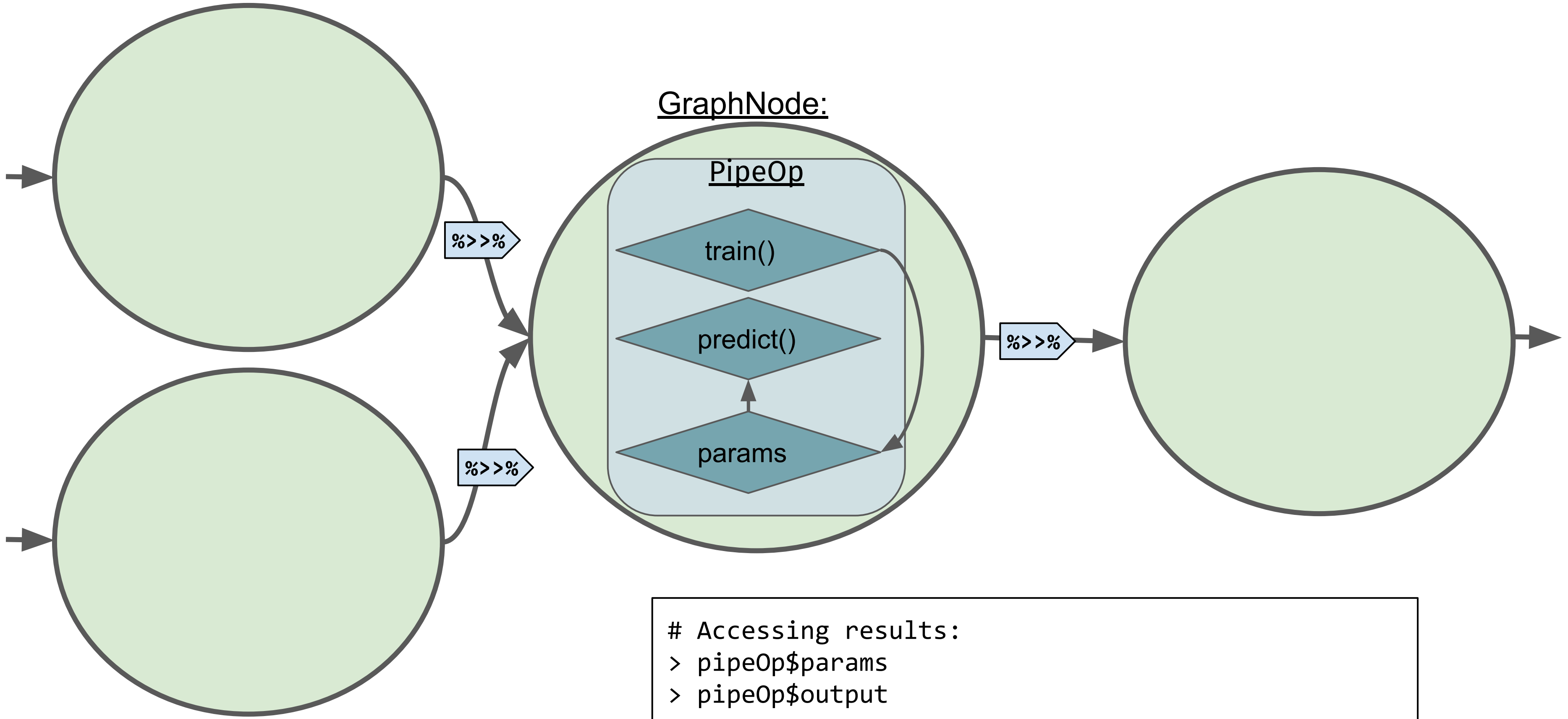
```
# Pseudo Code:
> pipeOpScale() %>>% pipeOpPCA() %>>% pipeOpLearner("svm")
```



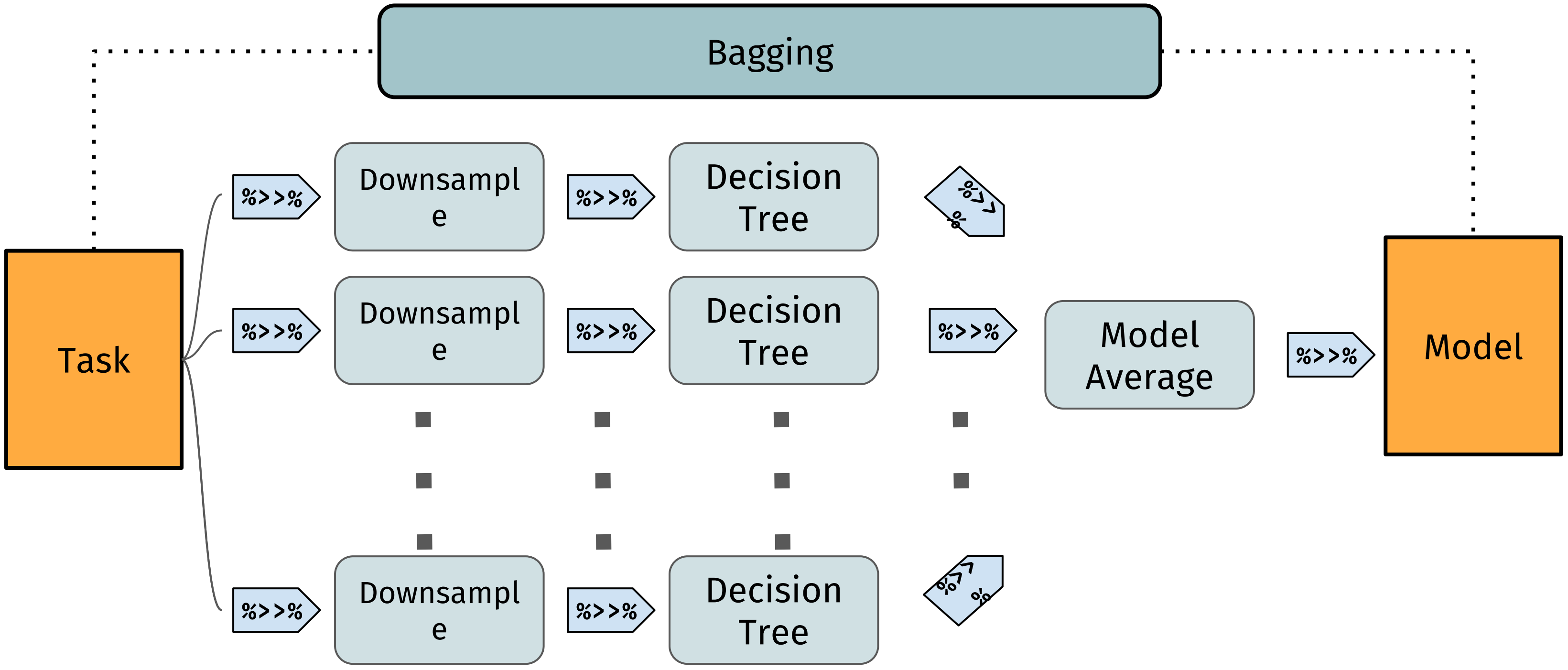
- `train()` saves transformation params and outputs transformed training data.

- `predict()` uses params and outputs transformed test data

Multiple GraphNode's can be connected with " %>>% "



```
# Accessing results:
> pipeOp$params
> pipeOp$output
```



```
# Pseudo Code:
> rep(100, pipeOpDownsample() %>>% pipeOpLearner(“rpart”)) %>>% pipeOpModelAverage()
```