Algorithm 1: Employee Turn out prediction using state of the art algorithms(dataset preperation AND Defined Functions)

```
dataset:=readCSV(fileName);
typeIndex:=dataset.column["type"].index;
satisfactionLevelIndex:=dataset.column["satisfactionLevel"].index;
lastEvaluationIndex:=dataset.column["lastEvaluation"].index;
salaryIndex:=dataset.column["salary"].index;
j:=0;
do
    dataset[j][typeIndex]:=dataset[j][typeIndex].factorize();
    dataset[j][satisfactionLevelIndex]:=dataset[j][satisfactionLevelIndex]*100;
    dataset[j][lastEvaluationIndex]:=dataset[j][lastEvaluationIndex]*100;
    dataset[j][salaryIndex]:=(dataset[j][salaryIndex].factorize())+1;
    j:=j+1;
while (j <= (dataset.length(row) − 1));
Def SBSFeatureSelector(Xtrain,Ytrain,Xtest,Ytest):
    classifier:=DecisionTreeClassifier(criterion="entropy",rstate=TRUE);
    sbs:=SBS(classifier,kFeatures:=1);
    sbs.fit(Xtrain,Ytrain);
    featureSet:=[length(k) for k in sbs.subsets];
    visualize(featureSet,sbs.scores,xlabel="accuracy",ylabel="number of features");
    display("indices: "+sbs.subsets+"AND"+"Scores"+sbs.scores);
    Select Only the best combinational features in Xtrain,Ytrain,XTest,Ytest;
    return Xtrain,Ytrain,Xtest,Ytest with best combinational features;
Def featureImportanceWRforest(Xtrain,Ytrain,Xtest,Ytest):
    featureLabels:=Dataset.Columns[which are selected by SBS];
    classifier:=RandomForestClassifier(criterio='entropy',estimators=1000,rstate=TRUE);
    classifier.fit(Xtrain,YTrain);
    importances:=classifier.featureImportances;
    indices:=sort(importances);
    f:=0;
    do
        display featureLabels[indices[f]] AND importances[indices[f]];
        f:=f+1;
    while (f <= selectedFeaturesFromSBS.length() − 1);
    return top three best features according to importance in Xtrain,Ytrain,Xtest,Ytest;
Def chiCalculation(Xtrain,Ytrain,Xtest,Ytest,kNumber):
    selector:=SelectorKBest(chi2,k=kNumber);
    Xtrain:=selector.transform(Xtrain);
    Xtest:=selector.transform(Xtest);
    scores=selector.scores;
    return Xtrain,Xtest,scores;
Def splitTrainTest(Xtrain,Ytrain,Xtest,Ytest,t):
    Xtrain,Xtest,Ytrain,Ytest:=TrainTestSplit(dataset.features,dataset.labels,tSize=t);
    return Xtrain,Xtest,Ytrain,Ytest;
```