Create a "train" and "test" data frame containing a single factor. The train set has three levels and the test set has two levels.

```
dfTrain <- data.frame(xf = c('a','b','c'))
dfTest <- data.frame(xf = c('a','b'))
```

We want to create a design matrix for dfTest that has the same encoding as dfTrain.

```
dummyObj <- dummyVars(~., dfTrain)
predict(dummyObj, newdata = dfTrain)
```

```
##   xf.a xf.b xf.c
## 1    1    0    0
## 2    0    1    0
## 3    0    0    1
```

```
predict(dummyObj, newdata = dfTest)
```

```
##   xf.a xf.b
## 1    1    0
## 2    0    1
```

Note that the design matrix for dfTest does not have the same columns as the design matrix for dfTrain. Looking at the code for predict.dummyVars I noticed that in line 123 of the dummyVars code the argument xlev = object$xlevels is referencing an element of the dummaryVars object that doesn't exist.
m <- model.frame(Terms, newdata, na.action = na.action, xlev = object$xlevels)

Although dummyVars object does not have the element 'xlevels', it instead has the element 'lvls'.

```
dummyObj$xlevels
```

```
## NULL
```

```
dummyObj$lvls
```

```
## $xf
## [1] "a" "b" "c"
```

Replacing object$xlevels with object$lvls and calling the new function predictDummy, we have the full list of variables as desired.

```
predictDummy(dummyObj, newdata = dfTest)
```

```
##   xf.a xf.b xf.c
## 1    1    0    0
## 2    0    1    0
```