

## **Regression Flow-A**

Shaillay Kumar Dogra  
Scientific Editor – QSAR World  
[editor@qsarworld.com](mailto:editor@qsarworld.com)

### **Notes:**

1. This Jython script works in Sarchitect Designer version 2.3
2. Learn about Sarchitect Designer – <http://www.strandls.com/sarchitect/index.html>
3. Get Sarchitect – <http://www.strandls.com/sarchitect/freetrial.php>

***The actual script follows this discussion. It is also accessible directly from the webpage in .py format.***

### **Discussion:**

The script provided here automates regression modeling. It is an implementation of the modeling workflow depicted in Figure 1. The following steps get executed:

- 1) Top 400 descriptors are selected based on correlation with endpoint – a “Feature Ranking” report of all the descriptors is displayed as well as a “Ranked Subset” with top 400 descriptors gets created. (User can change the cutoff values in line #67 of the script.)
- 2) An auto-correlation filter is applied with a cutoff of 0.9 on this “Ranked Subset” – a report and a scatter-plot is shown on the “Non Redundant Features” as well as a “Non Redundant Descriptors” subset containing non-correlated descriptors gets created. (User can change the cutoff values in line #102 of the script.)
- 3) “Regression Forest” modeling algorithm is called on the “Non Redundant Descriptors” – a “Regression Forest” report is displayed in a table format. User can now select any of the given models and choose to “Train Model” on that. (User can change various regression forest related parameters in line #116 of the script.)

Results for selected model(s) that were run are displayed. User can now save the model(s).

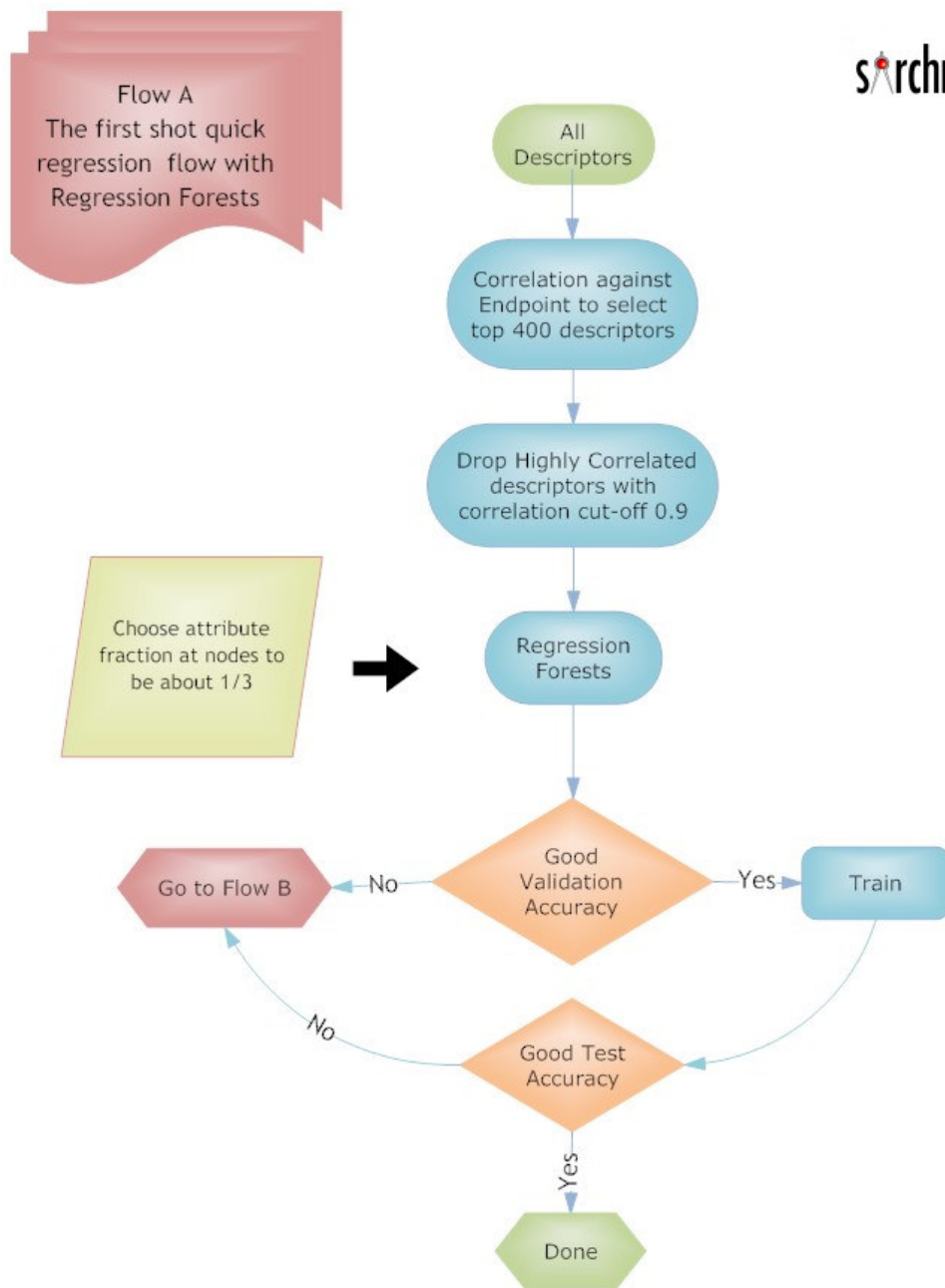


Figure 1

**Cite this as:**

Dogra, Shaillay K., "Script for automated Regression Flow-A" from QSARWorld – free online resource for QSAR modeling. <http://www.qsarworld.com/virtual-workshop.php>

```
##-----  
##  
##  
## Sarchitect Designer 2.3 script  
## implementing "Regression Flow-A"  
##  
##  
##  
## Shaillay Kumar Dogra  
## editor@qsarworld.com  
## August 05, 2007  
##  
##-----  
  
import script  
from script.dataset import *  
from script.algorithm import *  
from script.project import *  
from script.view import *  
from script.omega import createComponent, showDialog  
from javax.swing import *  
from com.strandgenomics.cube.dataset import *  
import jarray  
  
##-----  
## GET LIST OF CONTINUOUS, UNMARKED COLUMNS  
def getColumnlist(dataset):  
    ## Get columns, assumption: continuous and unmarked columns  
    indices_continuous =  
DatasetUtil.getContinuousColumnIndices(dataset)  
    indices_nm_continuous =  
script.project.removeMarkedColumns(dataset,indices_continuous)  
    columnList = indices_nm_continuous  
    #print columnList  
    return columnList  
  
##-----  
##  
def getIndexedIntArray(rowHeaderLabels,dataset):  
    from com.strandgenomics.cube.framework.data import ArrayUtil,  
DefaultIntArray  
    size = rowHeaderLabels.getSize()  
    array = DefaultIntArray(size)  
    for i in range(size):  
        colName = rowHeaderLabels.get(i)  
        c = dataset.getColumn(colName)  
        array.add(dataset.indexOf(c))  
    return ArrayUtil.createIndexedIntArray(array)  
  
##-----  
##  
def getStringArray(rowHeaderLabels):  
    array = []  
    from com.strandgenomics.cube.framework.data import DefaultIntArray
```

```

    size = rowHeaderLabels.getSize()
    for i in range(size):
        array.append(rowHeaderLabels.get(i))
    return array

##-----

dataset = script.project.getActiveDataset()

## Correlation against endpoint to select top N descriptors
result =
script.algorithm.FeatureSelection(test="correlation",select="Based on
rank", rank=400).execute()

inputs = result.getInputs()
dataset = inputs["dataset"]
rankDataset = result["results"]
rowHeaderLabels = result["rowHeaderLabels"]

nameColumn = ColumnFactory.createStringColumn("Descriptor",
getStringArray(rowHeaderLabels))
columnList = []
columnList.append(nameColumn)
for i in range(rankDataset.getColumnCount()):
    columnList.append(rankDataset.getColumn(i))

columns = jarray.array(columnList, IColumn)
newDataset =
DatasetFactory.createDataset(rankDataset.getName(), columns)

node = script.project.getActiveDatasetNode()

from com.strandgenomics.cube.framework.selection import
MappedSelectionModel, DummySelectionModel
from com.strandgenomics.cube.framework.filter import DummyFilterModel
selModel =
MappedSelectionModel(node.getContext().getColumnSelectionModel(),
getIndexedIntArray(rowHeaderLabels, dataset))

newnode = script.project.addFolderNode("Feature Ranking", node)
script.view.RankFeaturesView(node=newnode, dataset=newDataset,
title=newDataset.getName(), rowSelectionModel=selModel,
columnSelectionModel = DummySelectionModel.INSTANCE, filterModel =
DummyFilterModel(newDataset.getRowCount())).show()

pvalue = inputs['pvalue']
rank = inputs['rank']
select = inputs['select']
script.algorithm.SelectFeatures(node = node, featureselection =
newDataset, dataset = dataset, pvalue = pvalue, rank = rank, select =
select).execute(displayResult=1)

```

```
## Drop auto-correlated descriptors with correlation more than cutoff
result =
script.algorithm.RemoveRedundant(options="Correlation").execute(interac
tive=0, displayResult=1, newThread=0)
non_corr = result['SelectedIndicesList'][89] ##count starts from zero,
89 is 90, => auto-correlation cut off of 0.90

colIndices = ArrayUtil.createIndexedIntArray(to_java(non_corr))
script.project.getActiveDatasetNode().addChildDatasetNode("Non
Redundant Descriptors", columnIndices=colIndices, setActive=1,
addMarkedColumns=1)
script.view.Table(rowHeight=80).show()

node = script.project.getActiveDatasetNode()
dataset=script.project.getActiveDataset()

collist = getColumnlist(dataset)
endpoint = DatasetUtil.getMarkedColumnIndices(dataset, "classlabel")
endpoint = ArrayUtil.createIndexedIntArray(endpoint)
endpoint = endpoint.get(0)

algo =
script.algorithm.RegressionTreeBaggingX(classLabelColumn=endpoint, colum
nIndices=collist, splitCriterion="StandardDeviation", leafImpurity="0.01"
, fraction="0.3,0.6,0.9", numModels="10,50,100", fractionAtEachNode="0.3,0
.6,0.9")
algo.execute(interactive=0, displayResult=1, newThread=0, lockProject=0)

##
## END
##
```

---

End of Document