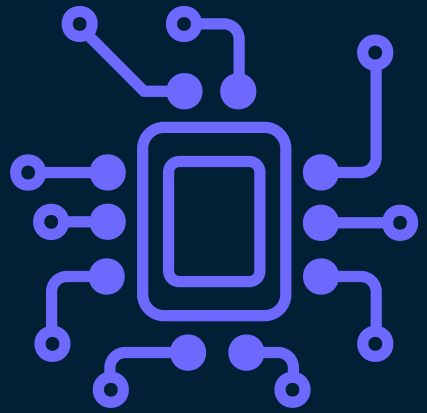
SAS in Action – SAS Viya CAS Action Sets

Tamara Fischer

Principal Data Scientist

Global Technology Practice

SAS® Viya …



ACTION…

…IN-MEMORY CAS

Procedures and Action Sets

PROCEDUREs

```
proc forest...;
...;
run:
```

```
proc gradboost...;
...;
run:
```

```
proc CAS;
   loadactionset 'explainModel';
   explainModel.shapleyExplainer /...;
   ...;
   run;
quit;
```

ACTION SETs

decisionTree	autotune	Explain Model

ACTIONs

forestTrain	tuneForest	linearExplainer
gbtreeTrain	tuneGradientBoostTree	partialDependence
dtreeTrain	tuneDecisionTree	shapleyExplainer
...	...	...

Procedures and Action Sets

PROCEDUREs

```
proc forest...;
...;
run:
```

```
proc gradboost...;
...;
run:
```

```
proc CAS;
   loadactionset 'explainModel';
   explainModel.shapleyExplainer /...;
   ...;
   run;
quit;
```

ACTION SETs

decisionTree	autotune	Explain Model

ACTIONs

forestTrain	tuneForest	linearExplainer
gbtreeTrain	tuneGradientBoostTree	partialDependence
dtreeTrain	tuneDecisionTree	shapleyExplainer
...	...	...

Getting Started?

- Welcome to SAS Programming Documentation
- What's New
- Learning SAS Viya
- **Syntax Quick Links**
  - Quick Links for SAS Procedures
  - SAS Language Elements by Name, Product, and Category
  - **Actions and Action Sets by Name and Product**
    - Actions by Name
    - **Action Sets by Name**
    - Action Sets by Product
- Advanced Analytics
- Data Access
- Cloud Analytic Services
- SAS Language Reference
- Migrating to UTF-8

Example Data Sets

- SAS Code Debugging
- Output and Graphics
- In-Database Technology
- Security and Administration
- SAS Servers
- Using the batch Plug-In for the SAS Viya CLI
- SAS Data Quality

Action Sets by Name

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Action Set	Syntax Name	Description
Access Control	accessControl	Provides actions for managing user access to resources
Active Machine Learning	activeLearn	Provides an action set for performing active learning which interactively query the user in order to minimize the labeling effort
Aggregate Loss Modeling	cdm	Provides an action for modeling aggregate losses by using the compound distribution models
Aggregation	aggregation	Provides actions for aggregating the values of one or more variables
Analytic Store Scoring	aStore	Provides actions for scoring using an analytic store
Association Rule Mining	ruleMining	Provides actions for association rule mining
Audio	audio	Provides actions for processing audio data
Autotune	autotune	Provides actions to tune machine learning algorithm hyperparameters for individual or multiple model types

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

- Active Learning Action Set
- Analytic Store Scoring Action Set
- Association Rule Mining Action Set
- Audio Action Set
- Autotune Action Set
- Bayesian Net Classifier Action Set
- BioMedImage Action Set
- Boolean Rule Action Set
- Data Science Pilot Action Set
- **Explain Model Action Set**
  - **Syntax**
  - Details
  - Examples
  - References
- Factorization Machine Action Set
- Fair AI Tools Action Set
- Fast k-Nearest Neighbor Action Set
- Generative Adversarial Network Action Set
- Generalized Linear Multitask Learning Action Set
- Graph-Based

Explain Model Action Set: Syntax

Provides actions for explaining already trained models.

Table of Actions

Action Name	Description
linearExplainer	Uses linear models to explain already trained models. Supports global linear surrogates as well as the local methods: LIME and KERNEL SHAP.
partialDependence	Computes the partial dependence of an already trained model.
shapleyExplainer	Computes Shapley value estimates for a query given a reference table

Last updated: November 12, 2021

Documentation Examples

CASL

```
proc cas;
   loadactionset "explainModel";
   explainModel.shapleyExplainer / table           = "DMAGECR"
                                   query           = "QUERY"
                                   modelTable      = "FOREST_ASTORE"
                                   modelTableType  = "ASTORE"
                                   predictedTarget = "P_good_badbad"
                                   inputs          = {{name = "age"},
                                                      {name = "amount"},
                                                      {name = "coapp"},
                                                      {name = "duration"},
                                                      {name = "foreign"},
                                                      {name = "job"}
                                                     }
                                   nominals        = {{name = "coapp"},
                                                      {name = "foreign"},
                                                      {name = "job"}
                                                     }
                                   depth           = 1
                                   ;
   run;
quit;
```

PYTHON

```
s.loadactionset(actionset="explainModel")
s.shapleyExplainer(
  table           = {"name" : "DMAGECR"},
  query           = {"name" : "QUERY"},
  modelTable      = {"name" : "FOREST_ASTORE"},
  modelTableType  = "ASTORE",
  inputs          = ["age", "amount", "coapp", "duration",
                     "foreign", "job"],
  nominals        = ["coapp", "foreign", "job"],
  depth           = 1
 )
```

Tips

Start Page | * SAS Program.sas | * Neural Network.ctk

Run | Cancel | Copy to My Tasks | Code to Flow | (0)

SAS Studio

Data | Options | Output | Information

Code | Log | Edit Code

- Methods
  - Automatically tune selected options
  - Data Preparation
    - Include missing values
    - Standardization method: Midrange (default)
      - Scale so that the midrange is 0, the miniumum is -1, and the maximum is 1
  - Hidden Layers
    - Include hidden layers
      - Each row represents a different hidden layer:

Number of neurons	Activation function
50	Hyperbolic tangent (default

      - Note: The neural network can have at most 10 hidden layers.
    - Allow direct connections between input and target neurons
  - Training Details
  - Optimization
- Code Generation
  - Use NNET procedure
  - Use CAS procedure

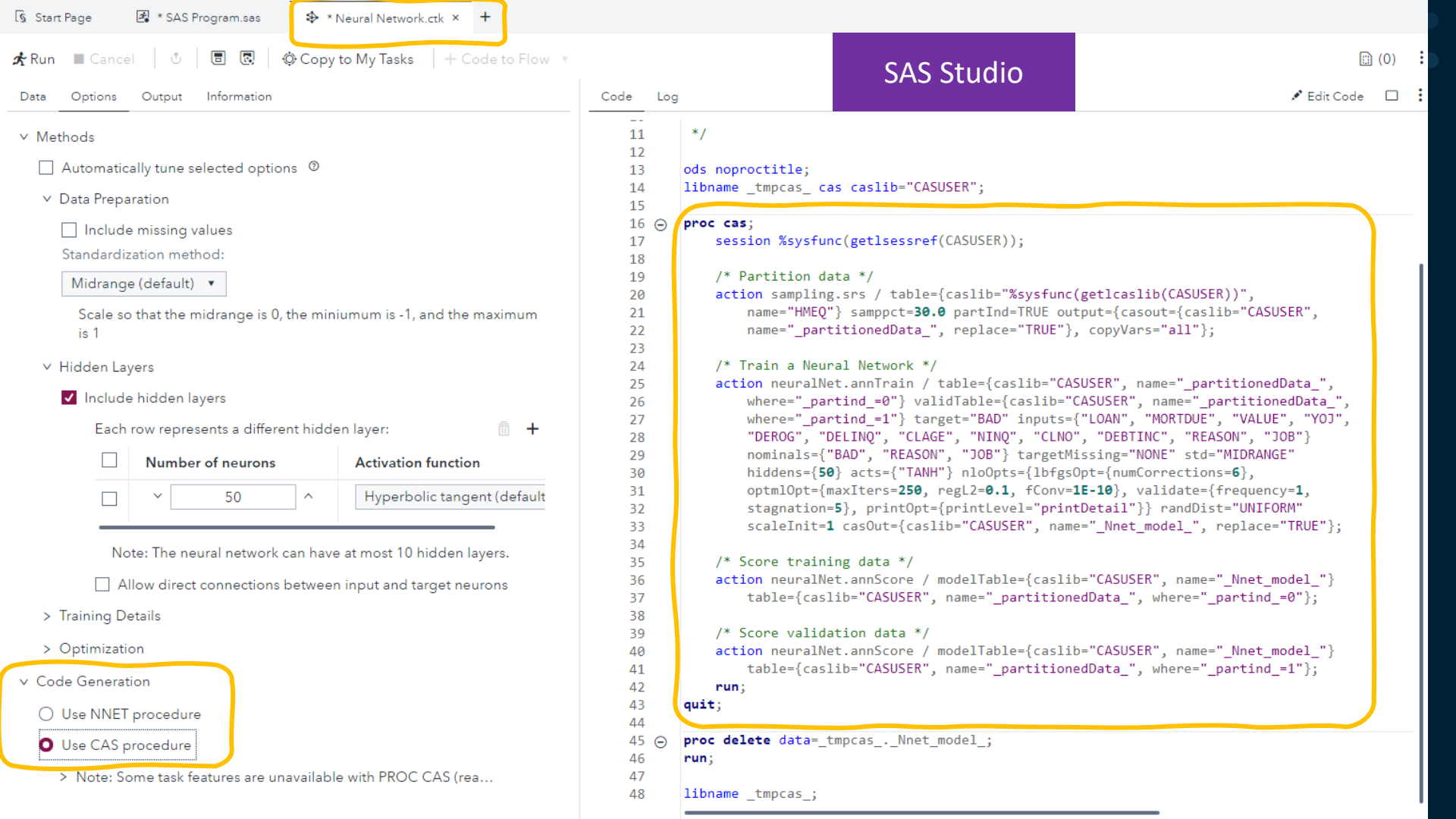
```
/*
 *
 * Task code generated by SAS® Studio 6.0
 *
 * Generated on '11/17/21, 10:33 AM'
 * Generated on server 'sas-launcher-734ae793-22a1-4655-827f-0d307f988468-k2z91'
 * Generated on SAS platform 'Linux LIN X64 3.10.0-1160.31.1.el7.x86_64'
 * Generated on SAS version 'V.04.00M0P091221'
 * Generated on browser 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 * Generated on web client 'https://dachsasviya41.ger.sas.com/SASStudio/main?locale=en_US&lau
 */

ods noproctitle;
libname _tmpcas_ cas caslib="CASUSER";

proc nnet data=CASUSER.HMEQ;
    partition fraction(validate=0.3);
    target BAD / level=nominal;
    input LOAN MORTDUE VALUE YOJ DEROG DELINQ CLAGE NINQ CLNO DEBTINC /
        level=interval;
    input REASON JOB / level=nominal;
    hidden 50;
    train stagnation=5 outmodel=_tmpcas_._Nnet_model_;
    optimization regL2=0.1;
run;

proc delete data=_tmpcas_._Nnet_model_;
run;

libname _tmpcas_;
```


Start Page
* SAS Program.sas
* Neural Network.ctk
Run
Cancel
Copy to My Tasks
Code to Flow
SAS Studio
Data
Options
Output
Information
Code
Log
Edit Code
Methods
Automatically tune selected options
Data Preparation
Include missing values
Standardization method:
Midrange (default)
Scale so that the midrange is 0, the miniumum is -1, and the maximum is 1
Hidden Layers
Include hidden layers
Each row represents a different hidden layer:
Number of neurons
Activation function
50
Hyperbolic tangent (default
Note: The neural network can have at most 10 hidden layers.
Allow direct connections between input and target neurons
Training Details
Optimization
Code Generation
Use NNET procedure
Use CAS procedure
Note: Some task features are unavailable with PROC CAS (rea...
*/
ods noproctitle;
libname _tmpcas_ cas caslib="CASUSER";
proc cas;
session %sysfunc(getlsessref(CASUSER));
/* Partition data */
action sampling.srs / table={caslib="%sysfunc(getlcaslib(CASUSER))",
name="HMEQ"} samppct=30.0 partInd=TRUE output={casout={caslib="CASUSER",
name="_partitionedData_", replace="TRUE"}, copyVars="all"};
/* Train a Neural Network */
action neuralNet.annTrain / table={caslib="CASUSER", name="_partitionedData_",
where="_partind_=0"} validTable={caslib="CASUSER", name="_partitionedData_",
where="_partind_=1"} target="BAD" inputs={"LOAN", "MORTDUE", "VALUE", "YOJ",
"DEROG", "DELINQ", "CLAGE", "NINQ", "CLNO", "DEBTINC", "REASON", "JOB"}
nominals={"BAD", "REASON", "JOB"} targetMissing="NONE" std="MIDRANGE"
hiddens={50} acts={"TANH"} nloOpts={lbfgsOpt={numCorrections=6},
optmlOpt={maxIters=250, regL2=0.1, fConv=1E-10}, validate={frequency=1,
stagnation=5}, printOpt={printLevel="printDetail"}} randDist="UNIFORM"
scaleInit=1 casOut={caslib="CASUSER", name="_Nnet_model_", replace="TRUE"};
/* Score training data */
action neuralNet.annScore / modelTable={caslib="CASUSER", name="_Nnet_model_"}
table={caslib="CASUSER", name="_partitionedData_", where="_partind_=0"};
/* Score validation data */
action neuralNet.annScore / modelTable={caslib="CASUSER", name="_Nnet_model_"}
table={caslib="CASUSER", name="_partitionedData_", where="_partind_=1"};
run;
quit;
proc delete data=_tmpcas_._Nnet_model_;
run;
libname _tmpcas_;

Start Page | VA_explain_gen_code_v2.sas | * SAS Program 1.sas | * SAS Program.sas × | * Neural Network.ctk +

Run | Cancel | Copy to My Snippets | + Code to Flow | Debug | Clear Log

Code | Log

```
1   cas;
2   caslib _all_ assign;
3
4   libname _tmpcas_ cas caslib="CASUSER";
5   proc nnet data=CASUSER.HMEQ;
6       partition fraction(validate=0.3);
7       target BAD / level=nominal;
8       input LOAN MORTDUE VALUE YOJ DEROG DELINQ CLAGE NINQ CLNO DEBTINC /
9           level=interval;
10      input REASON JOB / level=nominal;
11      hidden 50;
12      train stagnation=5 outmodel=_tmpcas_._Nnet_model_;
13      optimization regL2=0.1;
14  run;
15
16  proc cas;
17     builtins.history result=r/ casout={caslib='CASUSER' name='history' };
18  run;
19
20
21  proc print data=casuser.history(keep=command ordinal where=(ordinal ge 159));
22  run;
23
```

Start Page | * SAS Program.sas | * Neural Network.ctk

Run | Cancel | Copy to My Snippets | Code to Flow | Debug | Clear Log | Nov 17, 2021, 10:41:18 AM

Code | Log | Results

Obs	ordinal	command
1	159	action sampling.srs / table={name='HMEQ', caslib='CASUSER(gertsc)'}, sampPct=30, partInd=true, output={casOut={name='_partitionedData_', caslib='CASUSER(gertsc)', replace=true}, copyVars='ALL'};
2	160	action neuralNet.annTrain / table={name='_partitionedData_', caslib='CASUSER(gertsc)', where='_partind_=0'}, inputs={{name='LOAN'}, {name='MORTDUE'}, {name='VALUE'}, {name='YOJ'}, {name='DEROG'}, {name='DELINQ'}, {name='CLAGE'}, {name='NINQ'}, {name='CLNO'}, {name='DEBTINC'}, {name='REASON'}, {name='JOB'}}, nominals={{name='BAD'}, {name='REASON'}, {name='JOB'}}, validTable={name='_partitionedData_', caslib='CASUSER(gertsc)', where='_partind_=1'}, target='BAD', casOut={name='_Nnet_model_', caslib='CASUSER(gertsc)', replace=true}, hiddens={50}, acts={'TANH'}, std='MIDRANGE', randDist='UNIFORM', scaleInit=1, nloOpts={optmlOpt={regL2=0.1, maxIters=250, fConv=1E-10}, lbfgsOpt={numCorrections=6}, printOpt={printLevel='printDetail'}, validate={frequency=1, stagnation=5}}, targetMissing='NONE';
3	161	action neuralNet.annScore / table={name='_partitionedData_', caslib='CASUSER(gertsc)', where='_partind_=0'}, modelTable={name='_Nnet_model_', caslib='CASUSER(gertsc)'};
4	162	action neuralNet.annScore / table={name='_partitionedData_', caslib='CASUSER(gertsc)', where='_partind_=1'}, modelTable={name='_Nnet_model_', caslib='CASUSER(gertsc)'};
5	163	action table.tableInfo / name='_NNET_MODEL_', caslib='CASUSER(gertsc)', quiet=true;
6	164	action table.dropTable / name='_NNET_MODEL_', caslib='CASUSER(gertsc)';

Additional Information

- Blog Series from Peter Styliadis: https://blogs.sas.com/content/sgf/2021/08/06/cas-action-a-series-on-fundamentals/
- Action Sets by Name: https://go.documentation.sas.com/doc/en/pgmsascdc/v_020/allprodsactions/actionSetsByName.htm
- Developer‘s Guide to Writing Custom Tasks: https://go.documentation.sas.com/doc/en/webeditorcdc/v_013/webeditordg/titlepage.htm?homeOnFail