

# From Code to No-Code

## Alternative Paths for Data Analysts

Lincoln H. Groves, PhD | SAS Institute, Inc.  
Nebraska SAS User Group Session | May 14, 2024

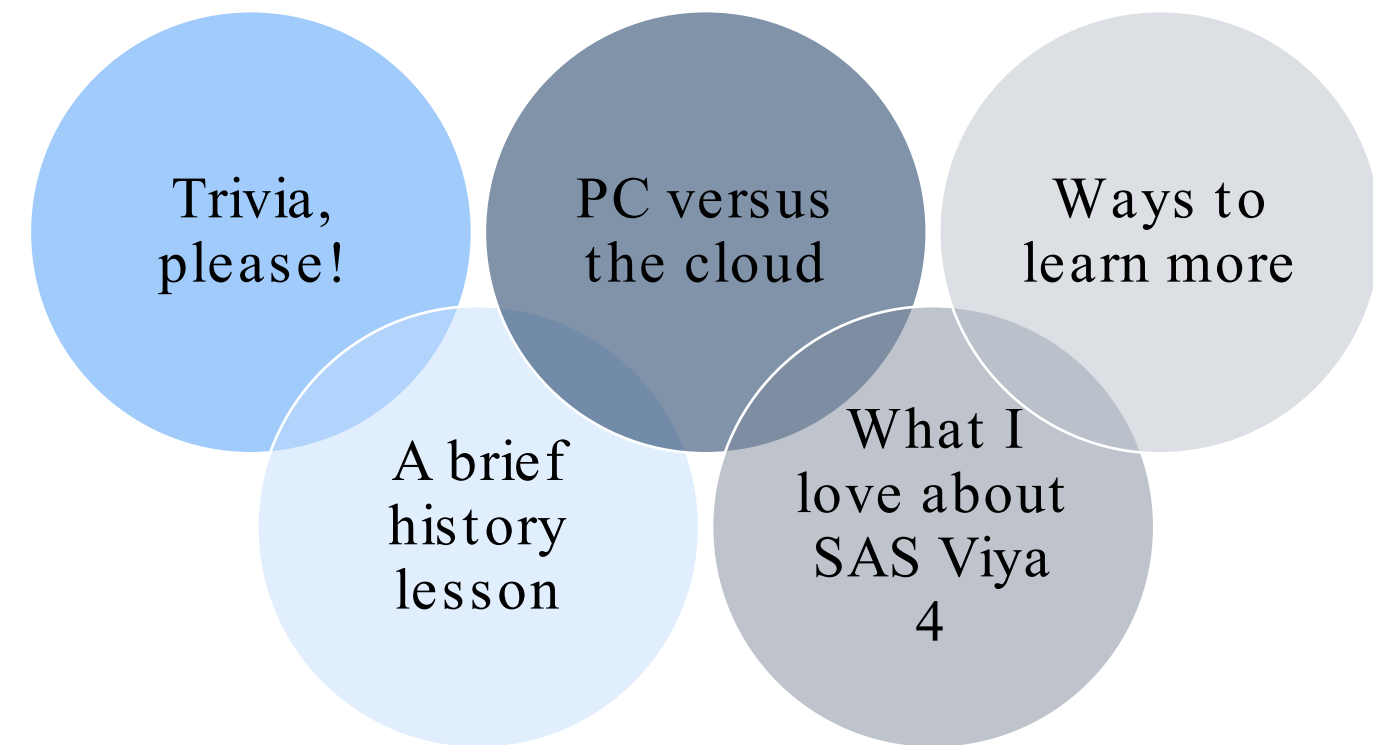


# Wait. Who is this guy?

And what is this session?



Lincoln H. Groves, PhD

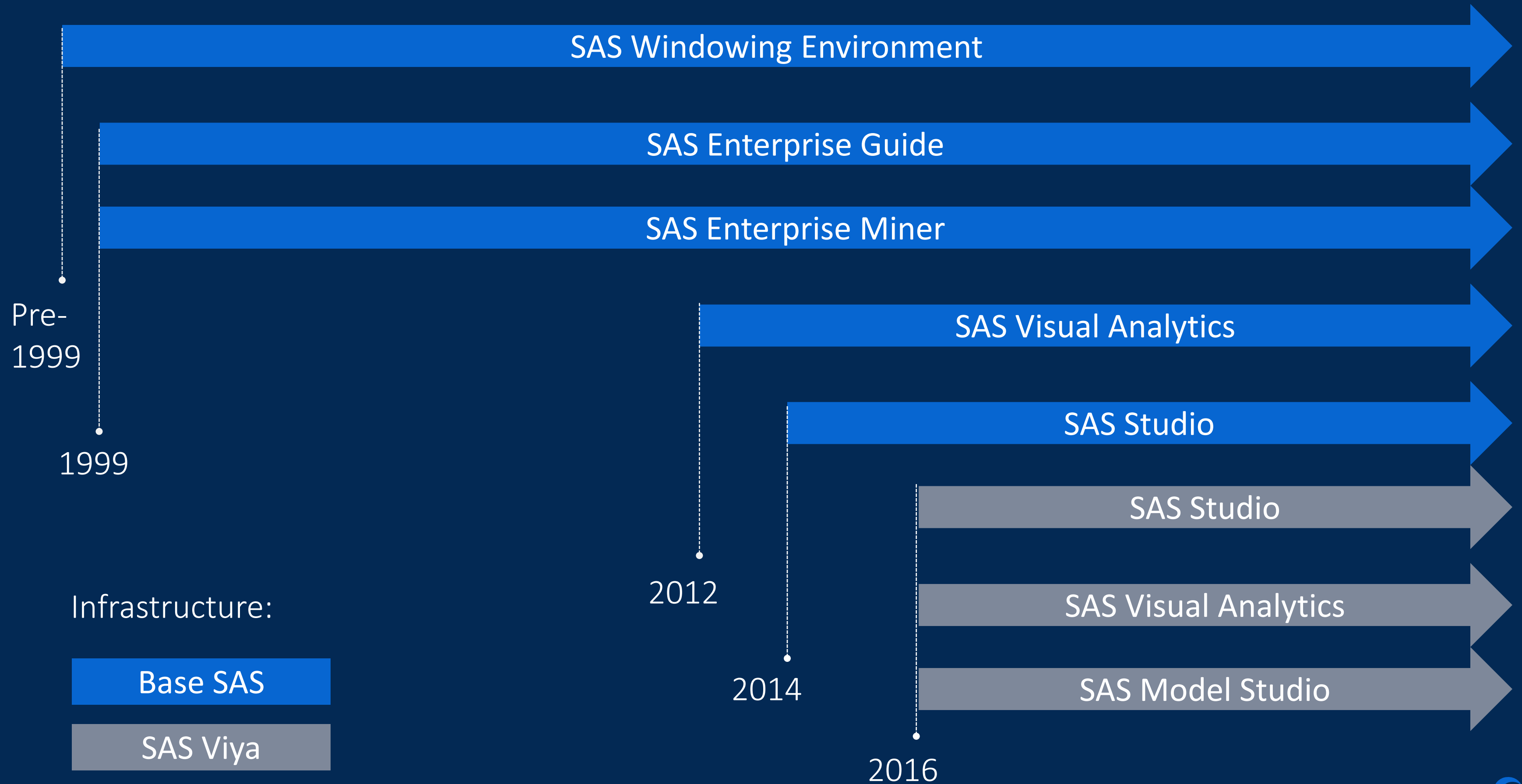


Initiator



SAS Product Timeline(s)

# Historical Timeline | Commercial Products





Why are there two platforms?

Key Development Details

# Why 2 Platforms?

What is old is new again...

Mainframe



PC



Cloud

[This Photo](#) by Unknown Author is licensed under [CC BY-NC](#)

# PC versus Cloud

What is the big deal?



Versus

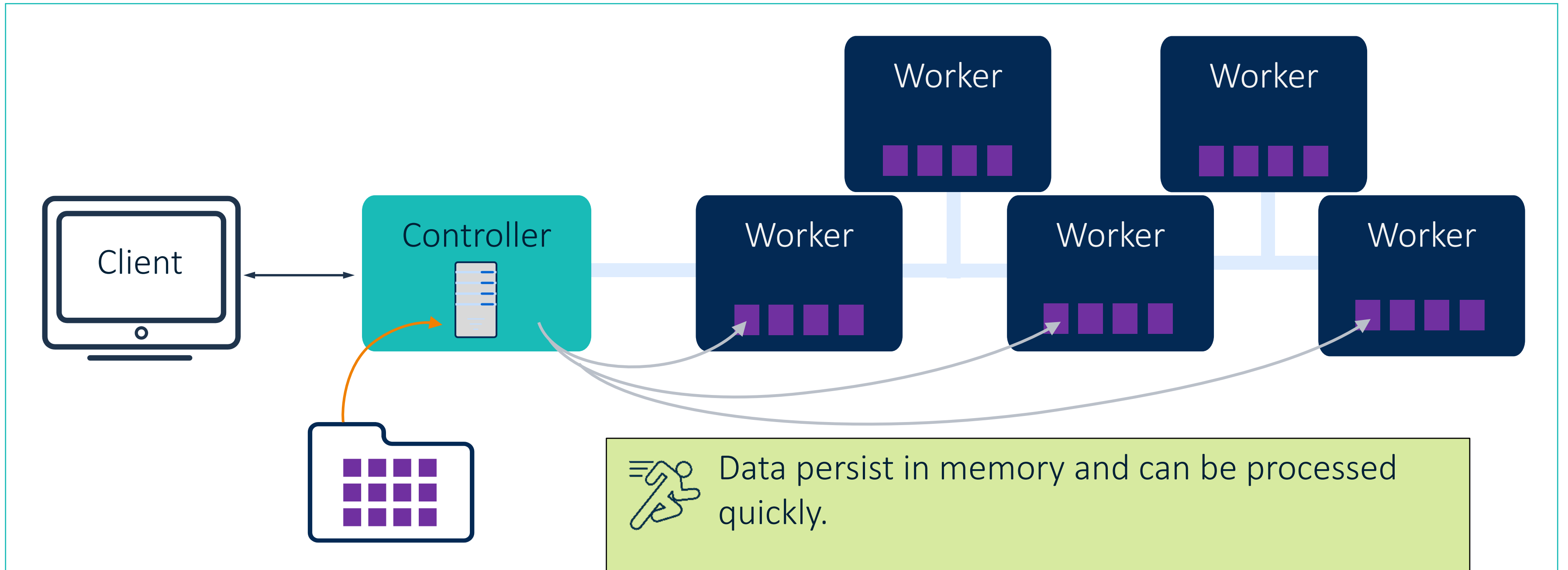




[This Photo](#) by Unknown Author is licensed under [CC BY-NC-ND](#)



# CAS Distributed Environment

## Big Deal, Part I

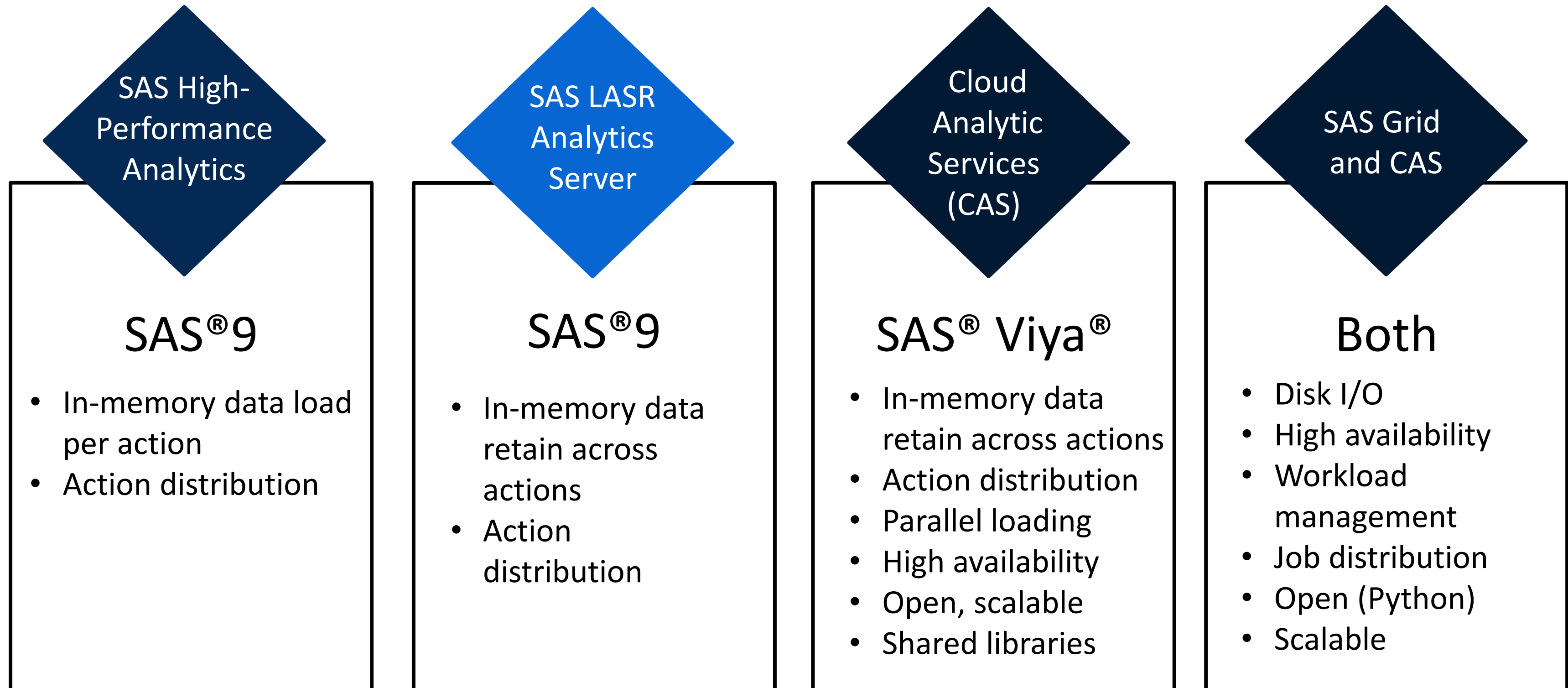


-  Data persist in memory and can be processed quickly.
-  SAS Viya algorithms are designed for distributed environments.



# Distributed Processing and SAS

A bit more history...



# Main Takeaways from SAS Cloud Analytics Services (CAS)

Cause that last slide was a lot...

## SAS Platform

### SAS Viya

- an open, cloud-enabled, analytic run-time environment

### SAS Cloud Analytic Services (CAS)



in-memory engine



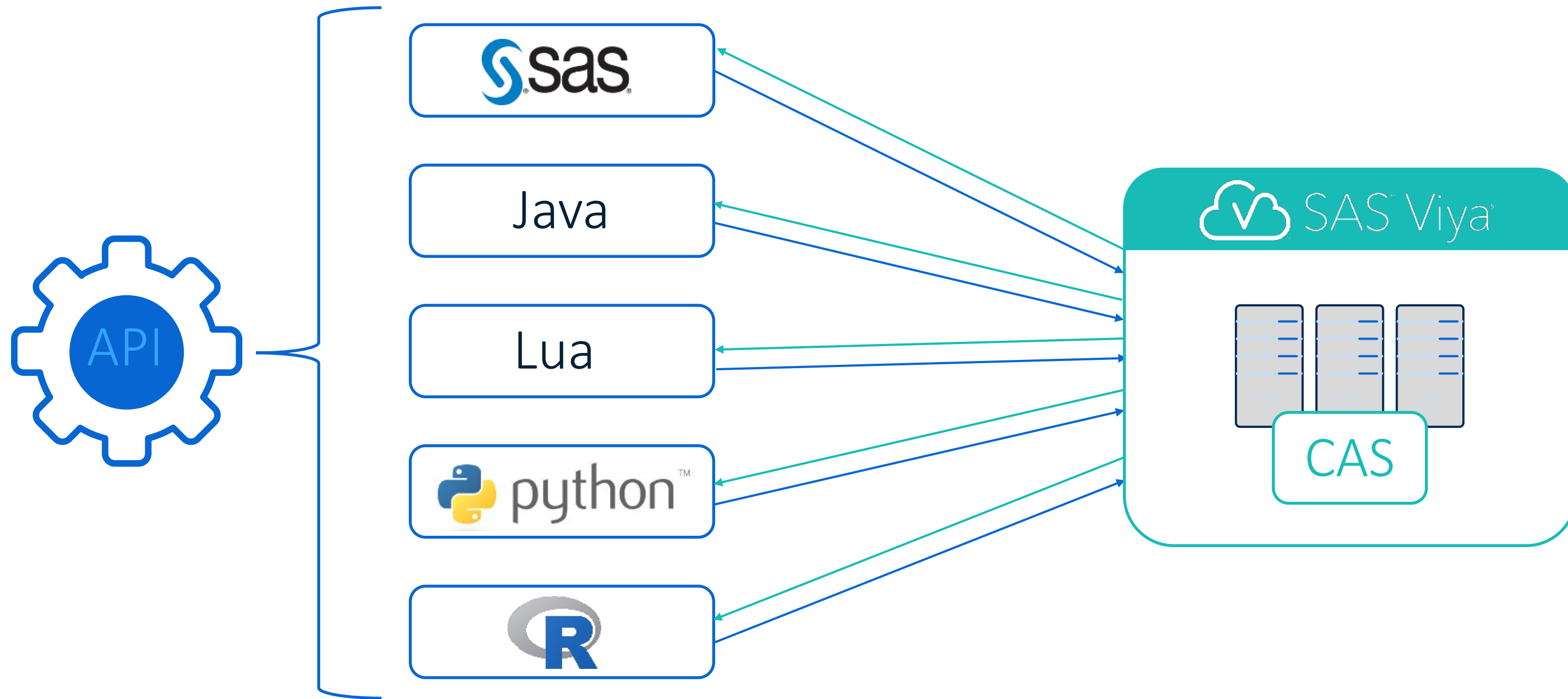
fast processing



data of  
any size

# SAS Viya is truly open

## Big Deal, Part 2



# Why SAS Viya?

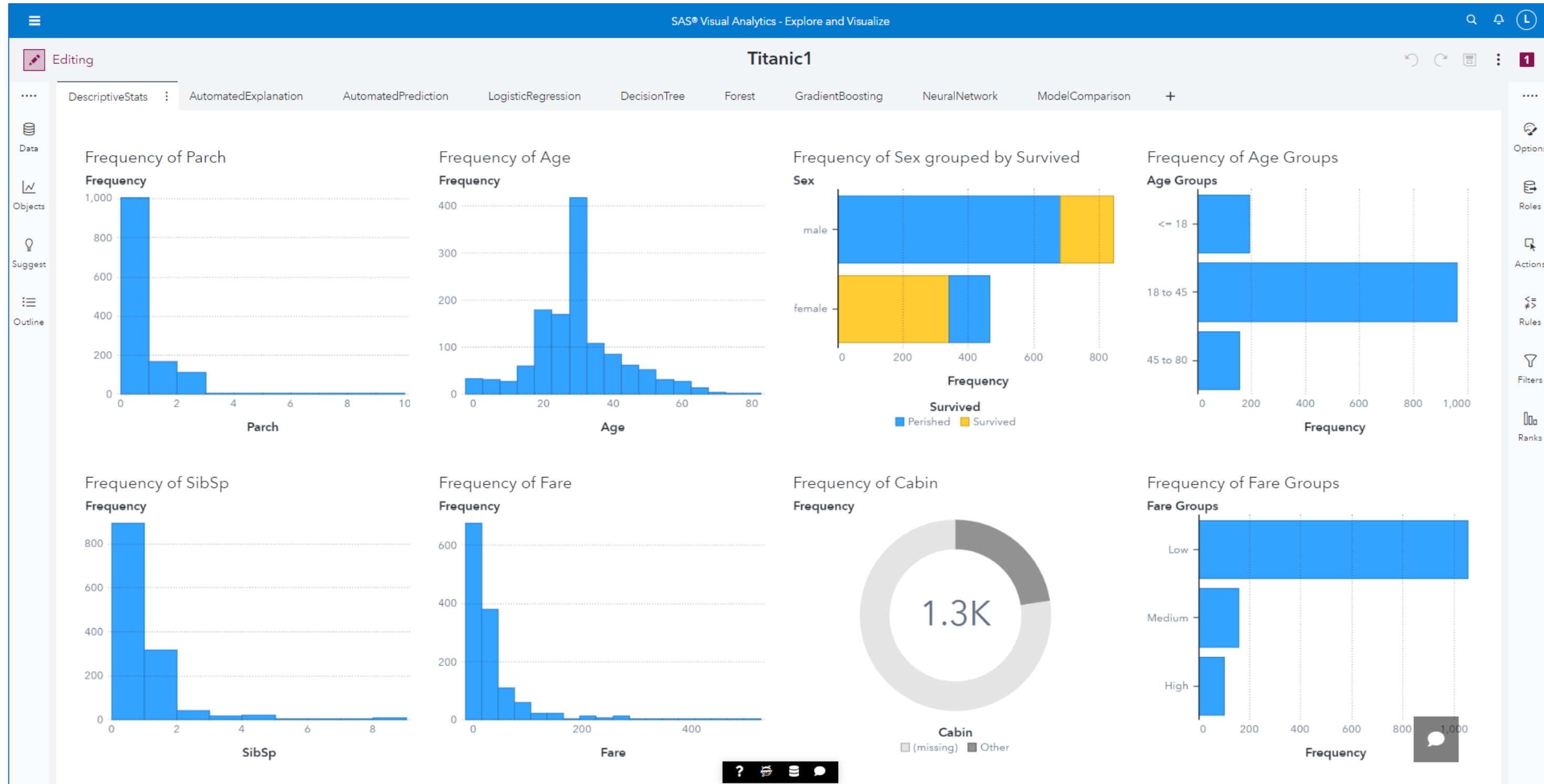
i.e., the top things I love about it...by product

# SAS Visual Analytics

Dashboarding approach to analytics = analytics for all

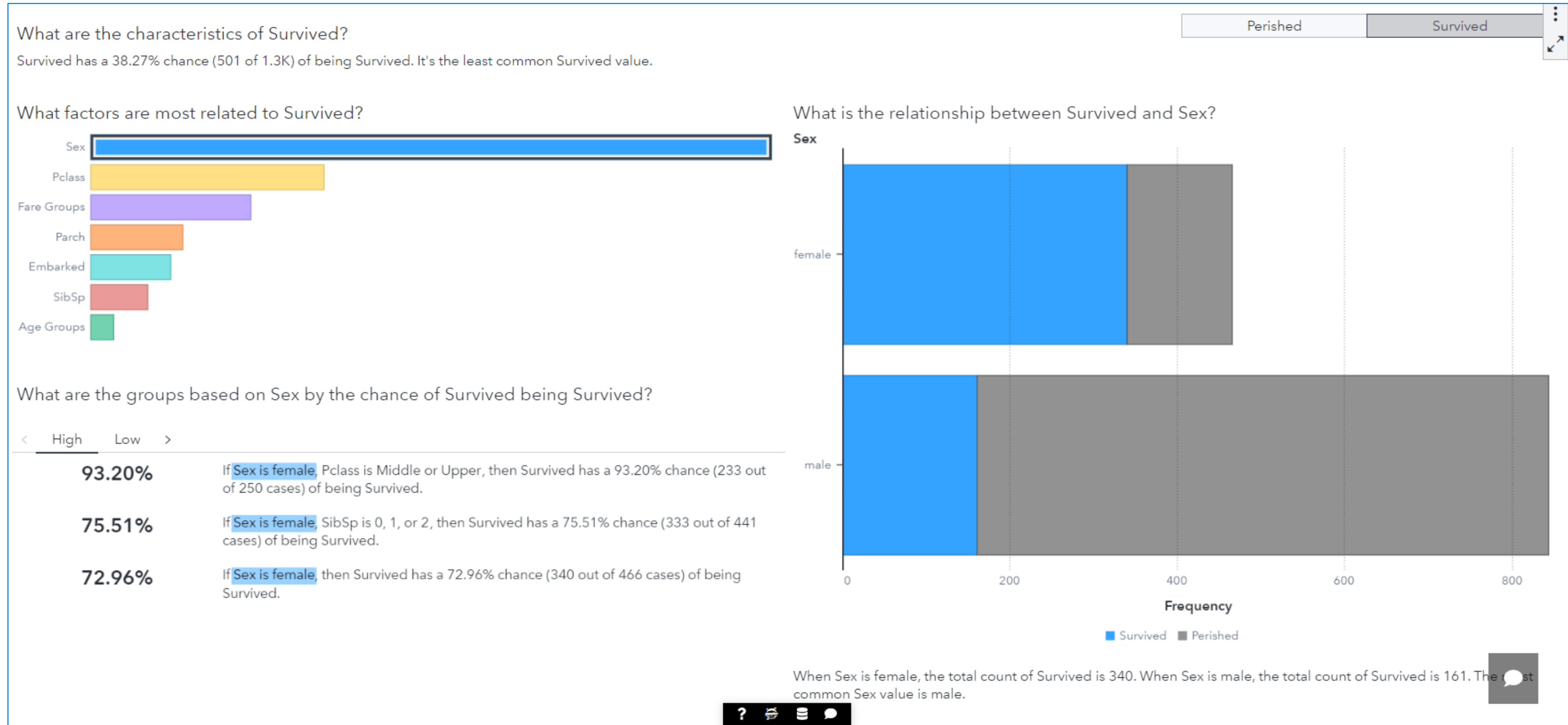
# SAS Visual Analytics

Utilize dynamic dashboards to help with storytelling



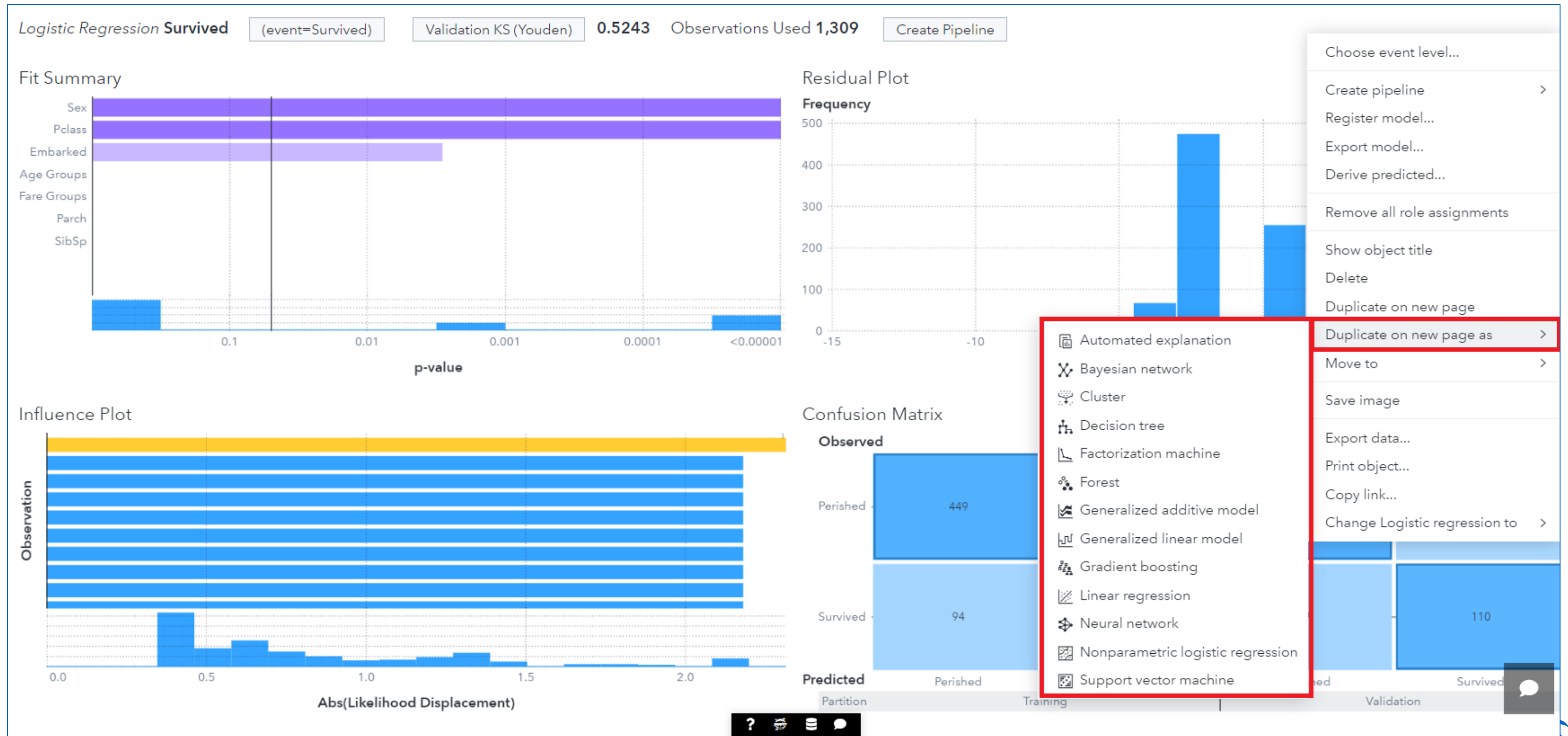
# SAS Visual Analytics

## Access AI Insights | Automated Explanation



# SAS Visual Analytics

Run new models with the click of a button | Duplicate on new page as...





# SAS Studio

Programmers guide to analytics

# SAS Studio

## Learn to code, Part 1: Tasks

The screenshot displays the SAS Studio interface for developing SAS code. The main window is titled "SAS® Studio - Develop SAS Code" and shows a task configuration for "Characterize Data" on the dataset "TITANIC.TITANIC".

**Task Configuration:**

- DATA:** TITANIC.TITANIC
- Filter:** (none)
- AUTOMATIC CHARACTERIZATION:**
  - Variables: PassengerId, Survived, Pclass, Name, Sex
- CUSTOM CHARACTERIZATION:** (expanded)
- ROLES:** (expanded)

**Code Editor:**

```
1 /*
2 *
3 * Task code generated by SAS® Studio 5.2
4 *
5 * Generated on '5/17/23, 12:41 PM'
6 * Generated by 'Lincoln.Groves@sas.com'
7 * Generated on server 'pdcesx23104'
8 * Generated on SAS platform 'Linux LIN X64 3.10.0-862.9.1.el7.x86_64'
9 * Generated on SAS version 'V.03.05M0P111119'
10 * Generated on browser 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (
11 * Generated on web client 'https://v4e055.vfe.sas.com/SASStudioV/main?locale=en_US&lau
12 */
13
14 ods noproctitle;
15
16 /*** Analyze categorical variables ***/
17 title "Frequencies for Categorical Variables";
18
19 proc freq data=TITANIC.TITANIC;
20     tables Survived Pclass Name Sex Ticket Cabin Embarked / plots=(freqplot);
21 run;
22
23 /*** Analyze numeric variables ***/
24 title "Descriptive Statistics for Numeric Variables";
25
26 proc means data=TITANIC.TITANIC n nmiss min mean median max std;
27     var PassengerId Age SibSp Parch Fare;
28 run;
29
30 title;
31
32 proc univariate data=TITANIC.TITANIC noprint;
33     histogram PassengerId Age SibSp Parch Fare;
34
```

# SAS Studio

## Learn to code, Part 2: Snippets

The screenshot displays the SAS Studio interface. On the left is a 'Snippets' pane with a search filter and a tree view of snippet categories. The main area is a code editor for a file named 'Compare Two ML Algorithms.sas'. The code is as follows:

```
1  /*****  
2  /* This example illustrates fitting and comparing two Machine  
3  /* Learning algorithms for predicting the binary target in the  
4  /* HMEQ data set. The steps include:  
5  /*  
6  /* (1) PREPARE AND EXPLORE  
7  /*   a) Check data is loaded into CAS  
8  /*  
9  /* (2) PERFORM SUPERVISED LEARNING  
10 /*   a) Fit model using Logistic Regression  
11 /*   b) Fit a model using a Decision Tree  
12 /*  
13 /* (3) EVALUATE AND IMPLEMENT  
14 /*   a) Score the data  
15 /*   b) Assess model performance  
16 /*   c) Generate ROC and Lift charts  
17  /***/  
18  
19  /***/  
20 /* Setup and initialize for later use in the program  
21  /***/  
22  
23 /* Define a CAS engine libref for CAS in-memory data tables */  
24 libname mycaslib cas caslib=casuser;  
25  
26 /* Specify the data set names */  
27 %let casdata      = mycaslib.hmeq_prepped;  
28 %let partitioned_data = mycaslib.hmeq_part;  
29  
30 /* Specify the data set inputs and target */  
31 %let class_inputs  = reason job;  
32 %let interval_inputs = im_clage clno im_debtinc loan mortdue value im_yoj im_ninq derog im_delinq;  
33 %let target        = bad;  
34
```

The status bar at the bottom shows 'Compare Two ML Algorithms.sas [temp] Line 1 Column 1 UTF-8' and system icons for help, refresh, and other functions.

# SAS Studio

## Learn to Code, Part 3 | GitHub Integration

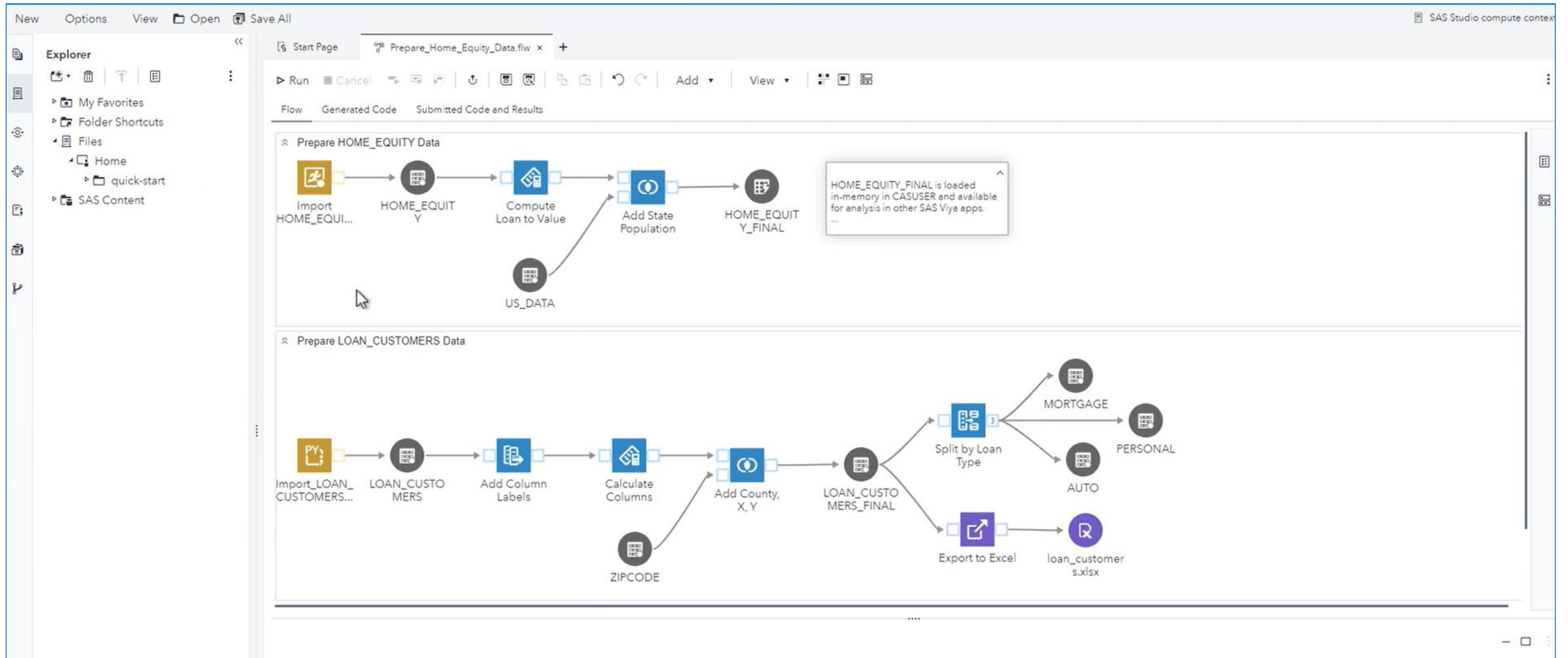
The screenshot displays the SAS Studio interface for developing SAS code. The top menu bar includes 'New', 'Options', 'View', 'Open', and 'Save All'. The main workspace is titled 'SAS Studio - Develop SAS Code' and shows the 'VulnerablePopulations' repository on the 'main' branch. The interface is divided into several sections:

- Git Repositories (Left Panel):** Shows the current repository 'VulnerablePopulations' with the 'main' branch selected. Other repositories listed include 'SAS\_Viya\_Programming', 'SAS\_Viya\_Machine\_Learning', 'casuser', and 'SASpy'.
- Commit and History (Top Center):** Displays the current repository 'VulnerablePopulations', branch 'main', and the last pull and push actions: 'Pull main' (Last pulled: Feb 27, 2023, 10:2...) and 'Push main' (Last pushed: Feb 27, 2023, 10:26:43...).
- Unstaged Changes (0) and Staged Changes (0) (Middle):** Both sections show 'No items', indicating no changes are currently staged or unstaged.
- Commit Comment (Bottom):** A text input field labeled 'Enter a commit comment' and a 'Commit' button.
- Working with Git (Right Panel):** A guide titled 'WORKING WITH GIT.' with the following steps:
  - Edit:** Edit files in your working directory. As you work, the files you change and save are added to the Unstaged Changes area of the Commit tab.
  - Stage:** Move the changed files that you want to keep together to the Staged Changes area. It is good practice to group related changes together in the same commit.
  - Commit:** Enter a short description of your changes and commit the files to your local repository. Each commit represents a snapshot of the changes since your last commit.
  - Push:** Share your committed changes with others by pushing them to the remote repository.

The bottom status bar shows 'VulnerablePopulations' with '(0)' changes and '(0)' submissions. The bottom right corner includes a 'Recover (1)' button and a 'Submission (0)' button.

# SAS Studio

## New frontiers: Custom Steps + SAS Studio Flows

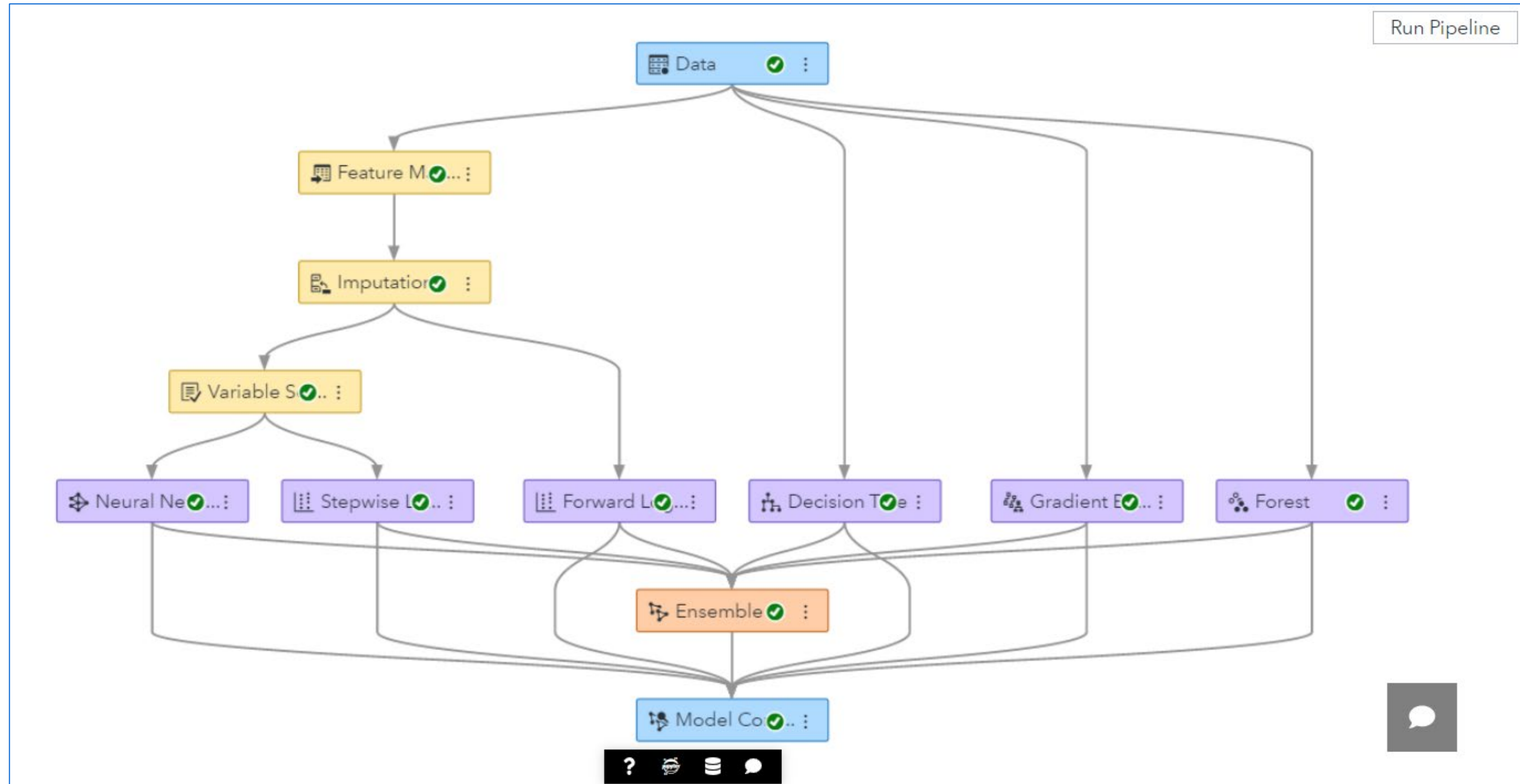


# SAS Model Studio

Where machine learning experts hang out

# SAS Model Studio

Get to the good stuff faster | Automated Pipelines



# SAS Model Studio

Use an AI friend for data prep| Feature Machine

The screenshot displays the SAS Model Studio interface. On the left, a pipeline diagram shows three nodes: 'Feature Machine', 'Imputation', and 'Variable Selection', connected in a sequence. A 'Run Pipeline' button is visible at the top right of the pipeline area. On the right, the configuration panel for the 'Feature Machine' node is open. It includes a description, a list of transformation policies, and input variable screening options.

**Feature Machine**

Description: Generates features that address one or more identified transformation policies.

Transformation Policy

- Cardinality
- Entropy
- Kurtosis
- Missingness
- Outliers
- Qualitative variation
- Skewness

Input Variable Screening

- Coefficient of variation
- Group rare levels

Leakage percent threshold: 90

Mutual information threshold: 0.05

Redundancy threshold:

The Feature Machine node generates new features by performing variable transformations to improve data quality and model accuracy. For more information, see [Overview of Feature Machine](#) in the Model Studio reference documentation.



# SAS Model Studio

Automatically tune those hyperparameters, Part 1 | single node

**Gradient Boosting** 🔍 🔒 ?

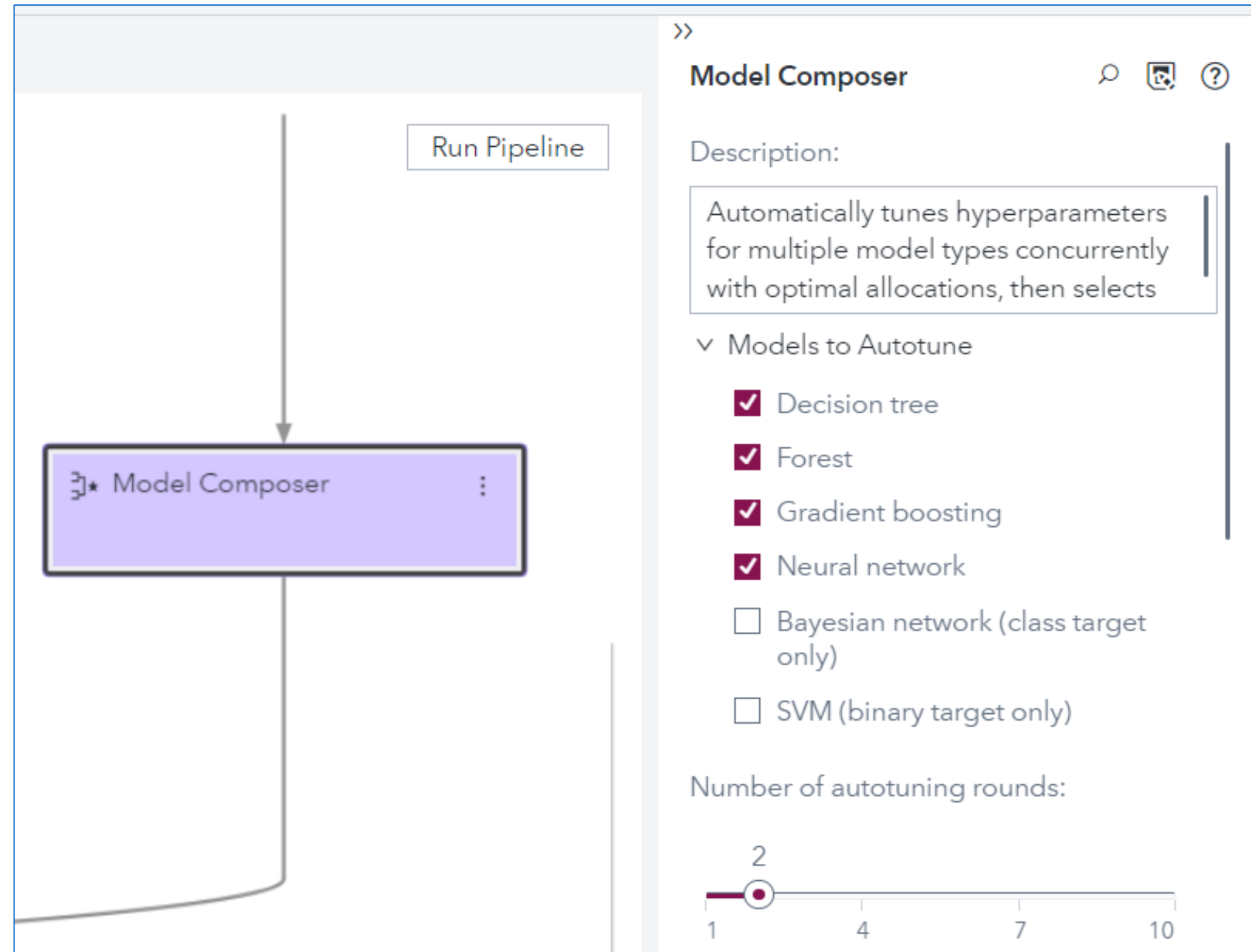
- ▾ Perform Autotuning
- > L1 Regularization
- > L2 Regularization
- > Learning Rate
- > Maximum Depth
- > Minimum Leaf Size
- > Number of Interval Bins
- > Number of Inputs per Split
- > Number of Trees
- > Subsample Rate
- > Search Options
- > General Options

Model Comparison

| Cham... ↓ | Name                                      | Algorithm Name      | KS (Youden) |
|-----------|---|---------------------|-------------|
| ★         | Gradient Boosting (Autotuning)            | Gradient Boosting   | 0.5866      |
|           | Decision Tree (Autotuning)                | Decision Tree       | 0.5385      |
|           | Ensemble                                  | Ensemble            | 0.5775      |
|           | Forest (Autotuning)                       | Forest              | 0.5665      |
|           | Forward Logistic Regression (Autotuning)  | Logistic Regression | 0.5450      |
|           | Stepwise Logistic Regression (Autotuning) | Logistic Regression | 0.5434      |

# SAS Model Studio

## Automatically tune those hyperparameters, Part 2 | Model Composer node



The screenshot displays the SAS Model Studio interface. On the left, a pipeline diagram shows a 'Model Composer' node highlighted in purple, with a 'Run Pipeline' button above it. An arrow points from the button to the node. On the right, the configuration panel for the 'Model Composer' node is open, showing the following settings:

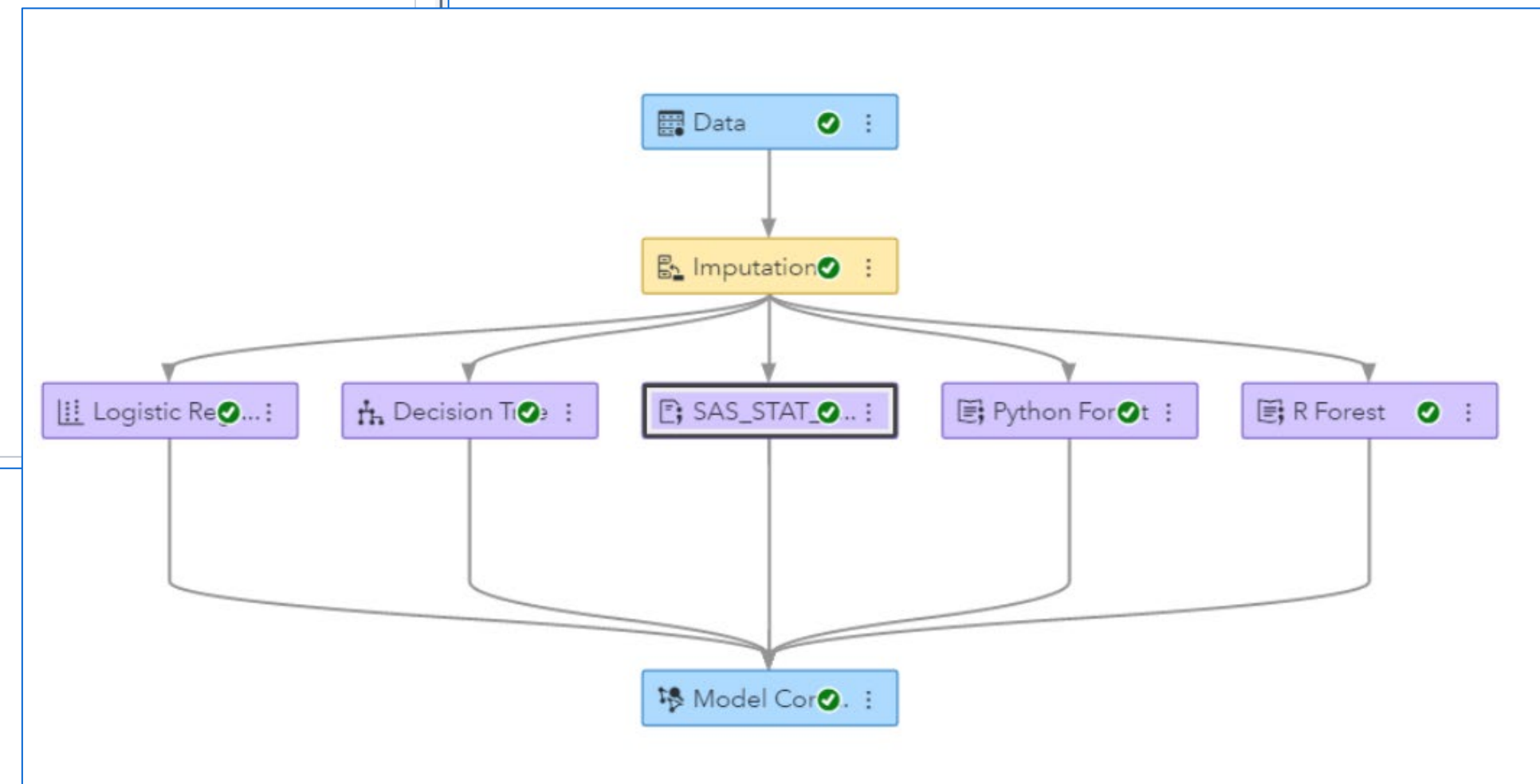
- Description:** Automatically tunes hyperparameters for multiple model types concurrently with optimal allocations, then selects
- Models to Autotune:**
  - Decision tree
  - Forest
  - Gradient boosting
  - Neural network
  - Bayesian network (class target only)
  - SVM (binary target only)
- Number of autotuning rounds:** 2 (slider range 1 to 10)

# SAS Model Studio

Incorporate SAS 9 Code | SAS Code node

The screenshot shows the SAS Model Studio interface. On the left, there is a 'Macros' panel with a search filter and a list of 'DATA: VARIABLES' including dm\_text, dm\_interval\_input, dm\_offset, dm\_into\_var, dm\_unary\_input, dm\_predicted\_var, dm\_id, dm\_binary\_input, dm\_key, and dm\_nominal\_input. The main area displays 'Training code' with the following SAS 9 code:

```
1 /* SAS code */
2 proc LOGISTIC data=&dm_data;
3 class %dm_nominal_input %dm_binary_input %dm_dec_target;
4 model %dm_dec_target(event="&dm_dec_event")= %dm_interval_input
5 %dm_binary_input
6 %dm_nominal_input /
7 selection=stepwise
8 slentry=0.3
9 slstay=0.35
10 details
11 lackfit ;
12 where &dm_partitionvar=&dm_partition_train_val;
13 ods output fitstatistics=&dm_data_outfit;
14 code file="&dm_file_scorecode";
15 run;
16
```



# SAS Model Studio

Incorporate R + Python code | Open-Source Code node

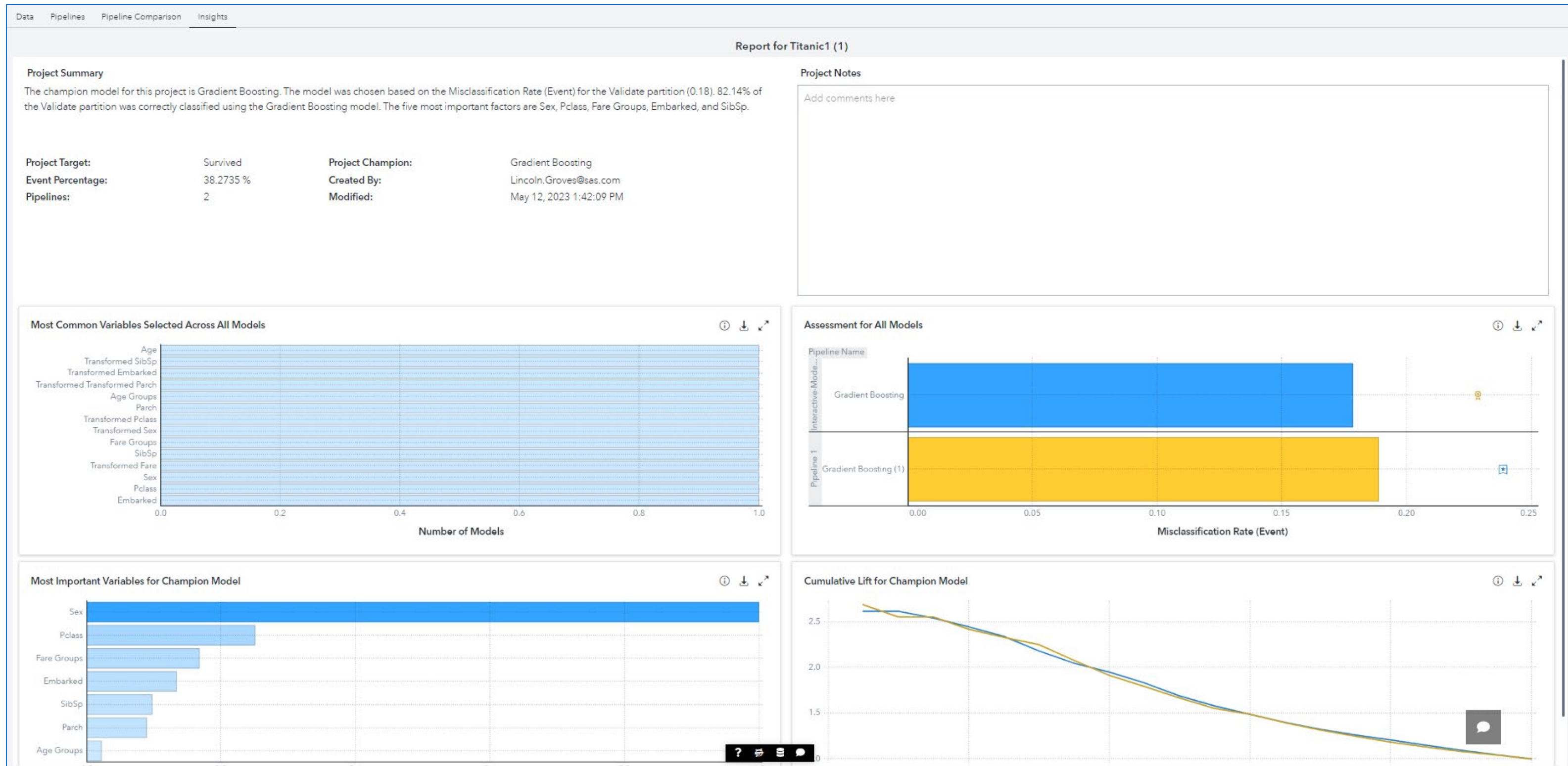
The image displays two overlapping windows from the SAS Model Studio interface. The top window, titled "\_Demo1 > Python Forest", shows a code editor with Python code for training a Random Forest model. The code includes imports for sklearn, data manipulation with pandas, and model fitting. A "Python Variables" panel on the left lists variables such as dm\_class\_input, dm\_classtarget\_intovar, dm\_classtarget\_level, dm\_dec\_target, dm\_input, dm\_inputdf, dm\_interval\_input, dm\_model, dm\_nodedir, dm\_partition\_train\_val, dm\_partitionvar, dm\_predictionvar, dm\_scoreddf, and dm\_traindf. The bottom window, titled "\_Demo1 > R Forest", shows R code for the same task. It includes comments about prepending Python or R code and uses the randomForest package. A "R Variables" panel on the left lists variables including dm\_class\_input, dm\_classtarget\_intovar, dm\_classtarget\_level, dm\_dec\_target, dm\_input, dm\_inputdf, dm\_interval\_input, dm\_model, dm\_model\_formula, dm\_nodedir, dm\_partition\_train\_val, dm\_partitionvar, dm\_predictionvar, dm\_scoreddf, dm\_traindf, node\_data.csv, and node\_scored.csv. Both windows feature a toolbar with standard editing icons and a search filter.

```
16 # the actual code that was executed. START ENTERING YOUR CODE ON THE NEXT LINE
17
18 from sklearn import ensemble
19
20 # Get full data with inputs + partition indicator
21 dm_input.insert(0, dm_partitionvar)
22 fullX = dm_inputdf.loc[:, dm_input]
23
24 # Dummy encode class variables
25 fullX_enc = pd.get_dummies(fullX, columns=
26
27 # Create X (features/inputs); drop partiti
28 X_enc = fullX_enc[fullX_enc[dm_partitionva
29 X_enc = X_enc.drop(dm_partitionvar, 1)
30
31 # Create y (labels)
32 y = dm_traindf[dm_dec_target]
33
34 # Fit RandomForest model w/ training data
35 params = {'n_estimators': 100, 'max_depth
36 dm_model = ensemble.RandomForestClassifie
37 dm_model.fit(X_enc, y)
38 print(dm_model)
39
```

```
3 # NOTE THAT A FEW LINES OF PYTHON OR R CODE ARE ADDED BEFORE YOUR CODE; FOR EXAMPLE:
4 # Python:
5 # dm_class_input = ["class_var_1", "class_var_2"]
6 # dm_interval_input = ["numeric_var_1", "numeric_var_2"]
7 # R:
8 # dm_class_input <- c("class_var_1", "class_var_2")
9 # dm_interval_input <- c("numeric_var_1", "numeric_var_2")
10 #
11 # For Python, use the Node Configuration section of the Project Settings to prepend
12 # any configuration code, which is executed before the above code. During execution,
13 # this code is automatically prepended to every node that runs Python code.
14 #
15 # After running the node, the Python or R code window in the node results displays
16 # the actual code that was executed. START ENTERING YOUR CODE ON THE NEXT LINE.
17
18 library(randomForest)
19
20 # RandomForest
21 dm_model <- randomForest(dm_model_formula, ntree=100, mtry=5, data=dm_traindf, importance=
22
23 # Score
24 pred <- predict(dm_model, dm_inputdf, type="prob")
25 dm_scoreddf <- data.frame(pred)
26 colnames(dm_scoreddf) <- c("P_CHURN0", "P_CHURN1")
27
28 # Print/plot model output
29 png("rpt_forestMsePlot.png")
30 plot(dm_model, main='randomForest MSE Plot')
31 dev.off()
32
33 write.csv(imodel, file="rpt_forestIMP.csv", row.names=TRUE)
```

# SAS Model Studio

Need help with storytelling? | Use AI generated Insights

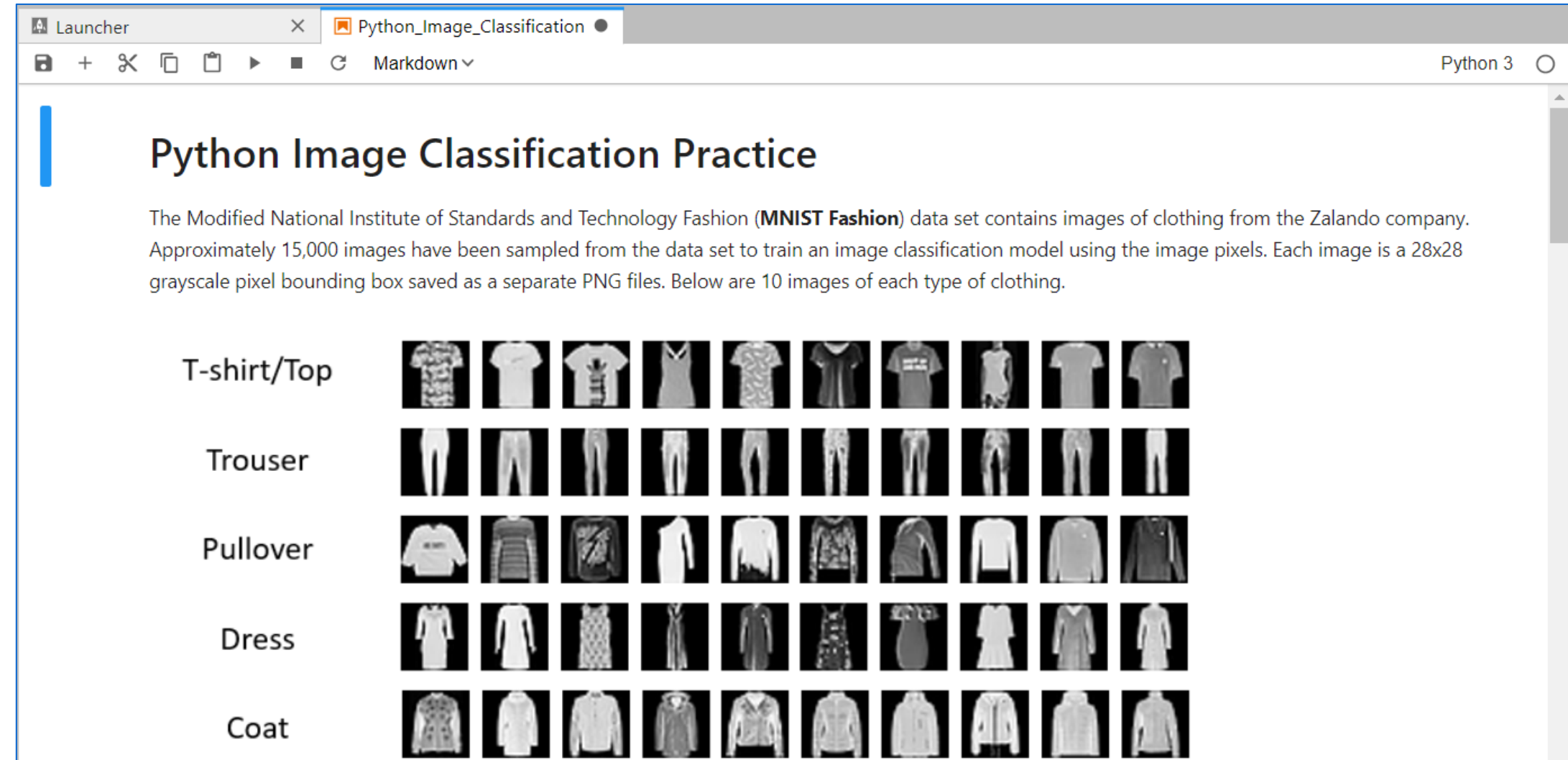
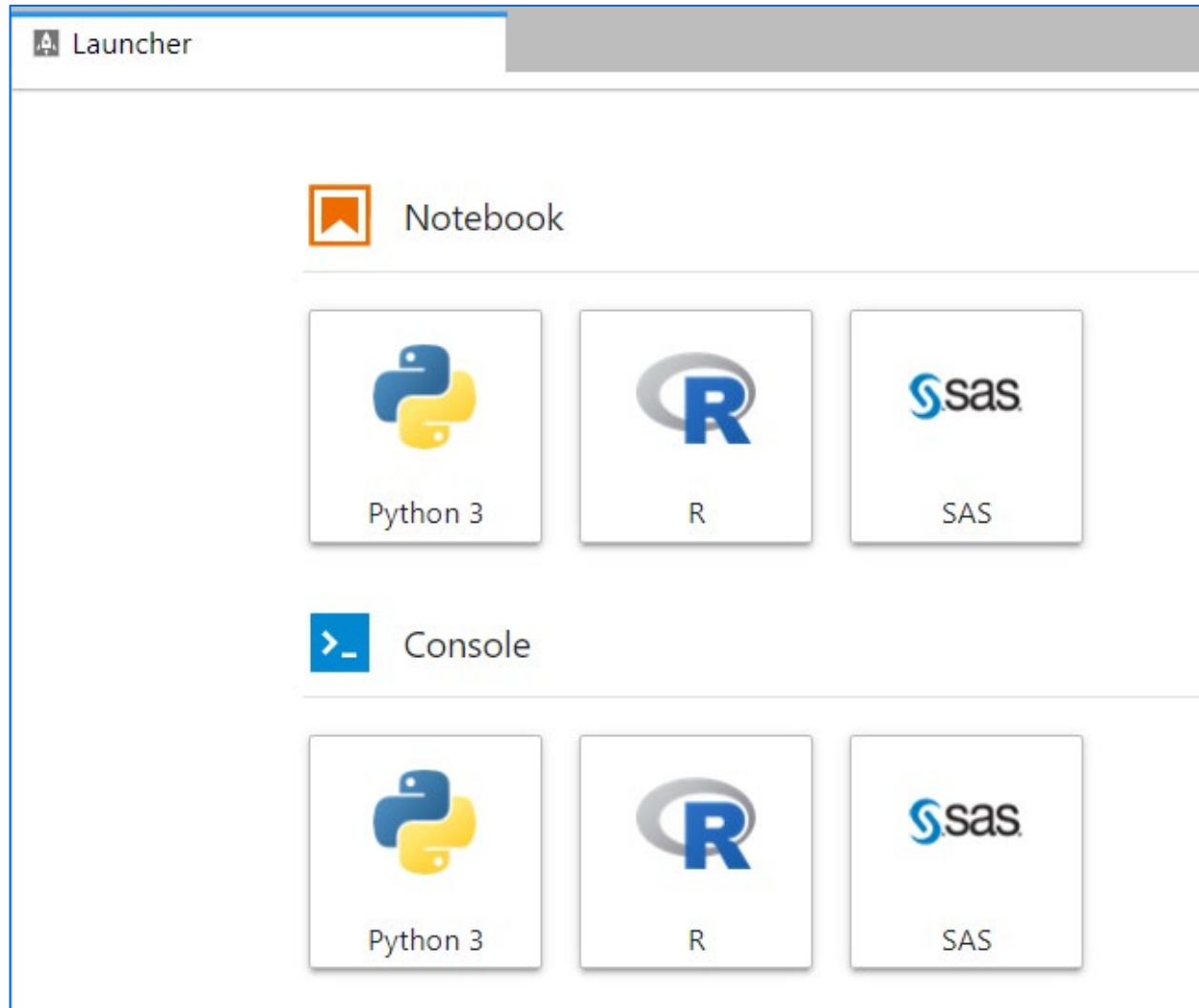


# Jupyter

Where to get your open-source fix

# Jupyter

Better together | Run R, Python, and SAS Notebooks



# Jupyter

Prefer to wrangle data in open source? | Convert open-source data to SAS

The screenshot shows a JupyterLab environment. On the left is a file browser with a search bar and a list of files and folders. The selected file is 'Python\_to\_SAS.ipynb'. The main area is a code editor for 'Python\_to\_SAS.ipynb' running on Python 3. The code includes:

```
display(df.head())
display(df.tail())

[6]: # from seaborn.datasets Load MPG
mpg_df = sns.load_dataset('mpg')

[7]: glimpse(mpg_df)

398 rows and 9 columns

  mpg  cylinders  displacement  horsepower  weight  acceleration  model_year  origin  name
0  18.0         8         307.0         130.0   3504         12.0         70    usa  chevrolet chevelle malibu
1  15.0         8         350.0         165.0   3693         11.5         70    usa  buick skylark 320
2  18.0         8         318.0         150.0   3436         11.0         70    usa  plymouth satellite
3  16.0         8         304.0         150.0   3433         12.0         70    usa  amc rebel sst
4  17.0         8         302.0         140.0   3449         10.5         70    usa  ford torino

  mpg  cylinders  displacement  horsepower  weight  acceleration  model_year  origin  name
393  27.0         4          140.0          86.0   2790         15.6         82    usa  ford mustang gl
394  44.0         4           97.0          52.0   2130         24.6         82  europe  vw pickup
395  32.0         4          135.0          84.0   2295         11.6         82    usa  dodge rampage
396  28.0         4          120.0          79.0   2625         18.6         82    usa  ford ranger
397  31.0         4          119.0          82.0   2720         19.4         82    usa  chevy s-10

[8]: import saspy

[ ]: sas = saspy.SASsession(cfgname='iomlinux')

[ ]: sas.dataframe2sasdata(df=mpg_df, table='mpg_python', libref='PY_CONV')
```

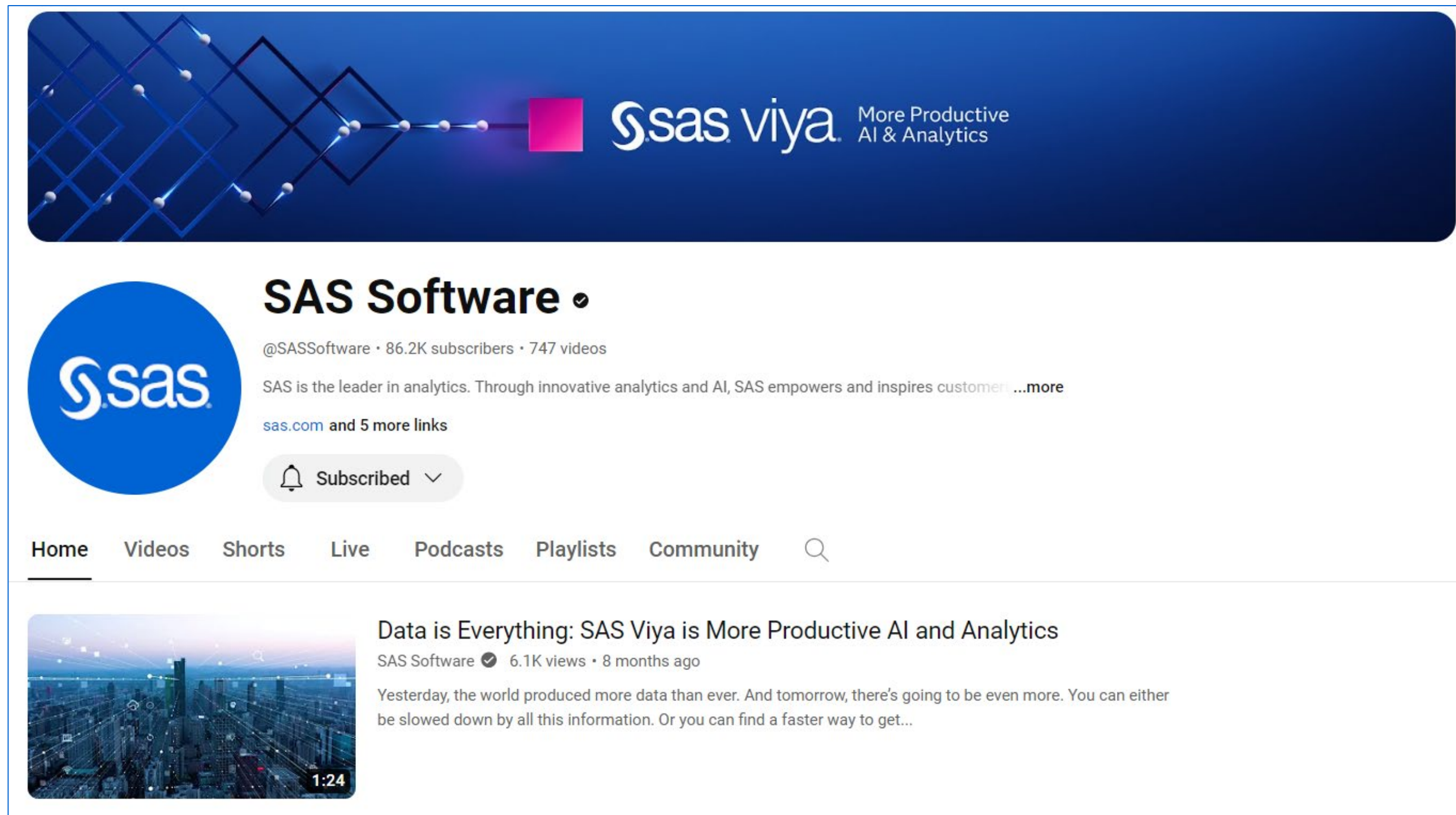


# Want to Learn more?

New year = new (analytics) you?

# Other options: Start with the free stuff!

## Part 1: SAS YouTube Channel



**SAS viya** More Productive AI & Analytics

**SAS Software** ✓

@SASSoftware · 86.2K subscribers · 747 videos

SAS is the leader in analytics. Through innovative analytics and AI, SAS empowers and inspires customer...more

[sas.com](https://sas.com) and 5 more links

Subscribed

Home Videos Shorts Live Podcasts Playlists Community

**Data is Everything: SAS Viya is More Productive AI and Analytics**

SAS Software ✓ 6.1K views · 8 months ago

Yesterday, the world produced more data than ever. And tomorrow, there's going to be even more. You can either be slowed down by all this information. Or you can find a faster way to get...

1:24

# Other options: Start with the free stuff!

## Part 2: SAS Communities

### SAS SUPPORT COMMUNITY

Get SAS tips, share your knowledge, and find out about upcoming SAS-related events.

  [Start a discussion](#)

56,450 Solutions 6,693 Library Articles 2,507 Users online 35 Groups


#### RECENT SOLUTIONS

- ✓ [SAS Studio how to go to a specific observation during review?](#)  
*Answered in SAS Studio*
- ✓ [Filter multiple columns with the same criteria](#)  
*Answered in SAS Programming*
- ✓ [proc transpose dataset](#)  
*Answered in SAS Procedures*
- ✓ [Synapse access using Azure Service Principal Authentication](#)  
*Answered in Administration and Deploy...*

[All recent solutions](#)

#### FEATURED ARTICLE

##### The Future of Viewing and Editing SAS® Datasets




SAS Explore 2023 Highlight: @ScottPLeslie of SAS guides you through what DataViewer can do as well as future capabilities.

[Check it out.](#)

#### FEATURED WEBINAR

##### Ask the Expert Webinar: Modernize Your Graphs Using ODS Graphics

[Watch on demand.](#)

# Other options: Start with the free stuff!

## Part 3: SAS Learning Subscription Trial

### SAS Learning Subscription

Get a glimpse of unlimited learning and training options.

Free 7-day trial. Start now. >

Starting your free SAS Learning Subscription trial is as easy as 1, 2, 3!

STEP 1

**1**

Set up a SAS profile. If you already have one, sign in.

STEP 2

**2**

Accept the license agreement and select SAS Learning Subscription (Free Trial) on the My Training page.

STEP 3

**3**

Select the courses you'd like to sample.

Don't wait – start your 7-day free trial today!

Free 7-day trial. Start now. >

# Level up to a SAS Learning Subscription

...when it's time

|   | Learning Subscription | Premium Subscription |
|---|-----------------------|----------------------|
| 200+ On-Demand Courses*                         | ✓                     | ✓                    |
| Certification Practice Exams and Prep Guides*   | ✓                     | ✓                    |
| Shareable Digital Badges                        | ✓                     | ✓                    |
| Hands-on Lab Hours                              | ✓                     | ✓                    |
| Usage Tracking Reports                          | ✓                     | ✓                    |
| Unlimited Public Live Web Courses               |                       | ✓                    |
| Academy for Data Science                        |                       | ✓                    |
| Custom Dashboard                                |                       | ✓                    |
| Training Points                                 |                       | ✓                    |
| Learning Needs Assessment and Adoption Services |                       | ✓                    |

*\*Included in free trial*  
*View discounts and guidelines*

Buy now >

START FREE TRIAL

Contact us >

START FREE TRIAL

# Last Plug for Learning SAS Viya

Join us in the 2024 SAS Hackathon!



Visit the website  
[sas.com/hackathon](https://sas.com/hackathon)



## Engage and Learn

- SAS Innovate in Las Vegas: April 16-19
- SAS Innovate On Tour Boot Camps in 16 global locations

April – June



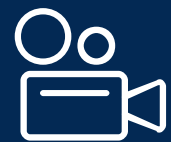
## Kickoff 30 May Registration Learning (VLE)

May – August



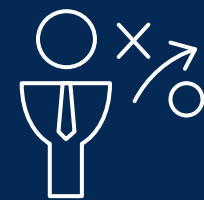
## ONE Month Hack

September 16– October 11



## Record and Upload

October 1-17



## Jury Voting

October 18-28



## Awards

November



## Story Promotion

December



The Results

# Thank you!

Connect with me on LinkedIn:

<https://www.linkedin.com/in/lincoln-h-groves/>

