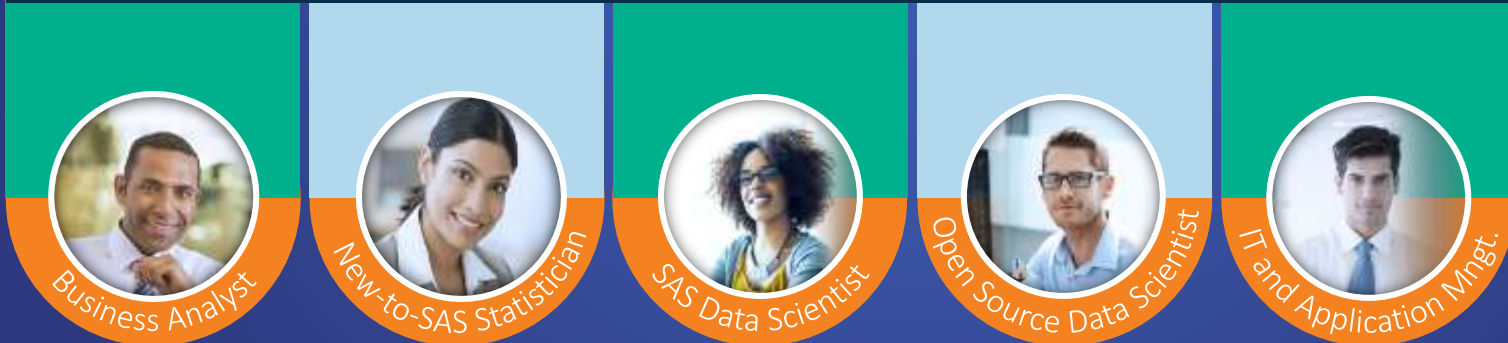


Analyse der NBA 1997 Daten durch 4 verschiedene Benutzer-Rollen

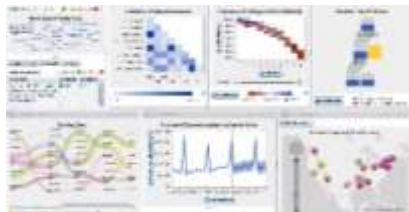
Offenheit der SAS Analytic Plattform für unterschiedliche Zugriffsarten

One Integrated Solution for Different User Types



Offenheit der SAS Analytic Plattform für unterschiedliche Zugriffsarten

Erfüllung der individuellen Anforderungen



Office
Integration



One Integrated Solution for Different User Types



#SASF17

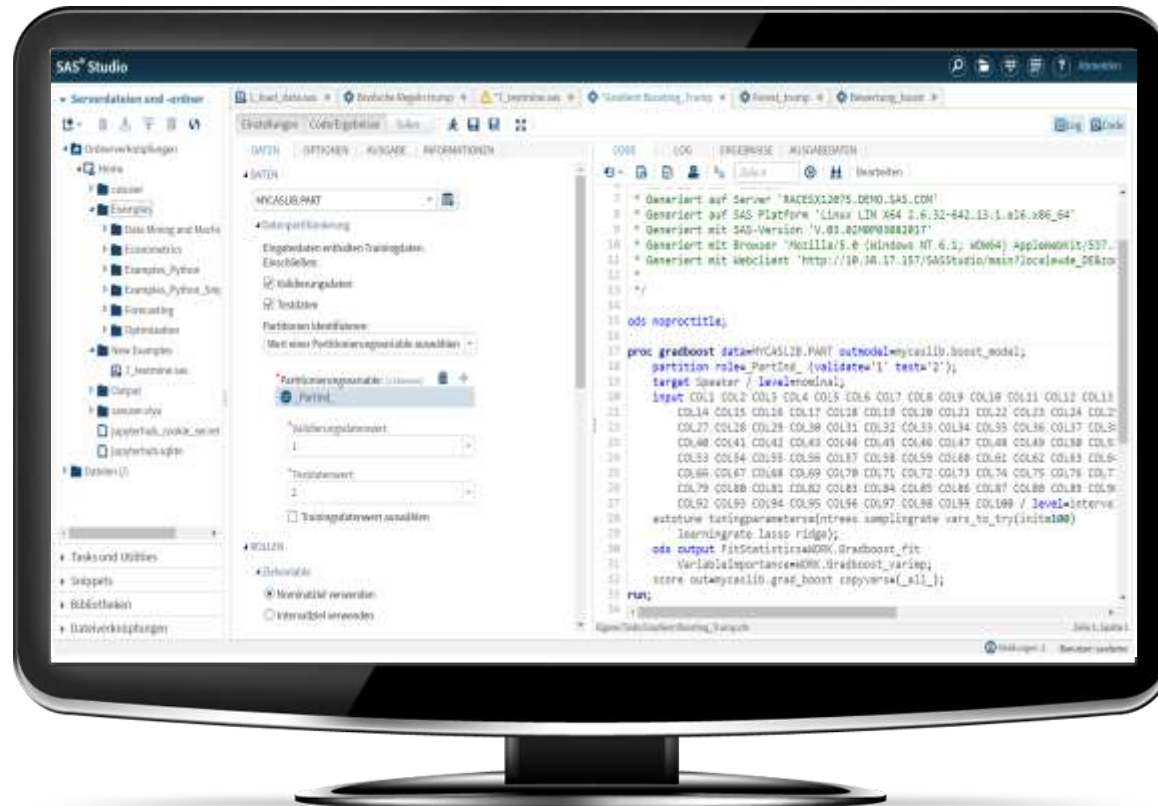


SASF17 / SASForum

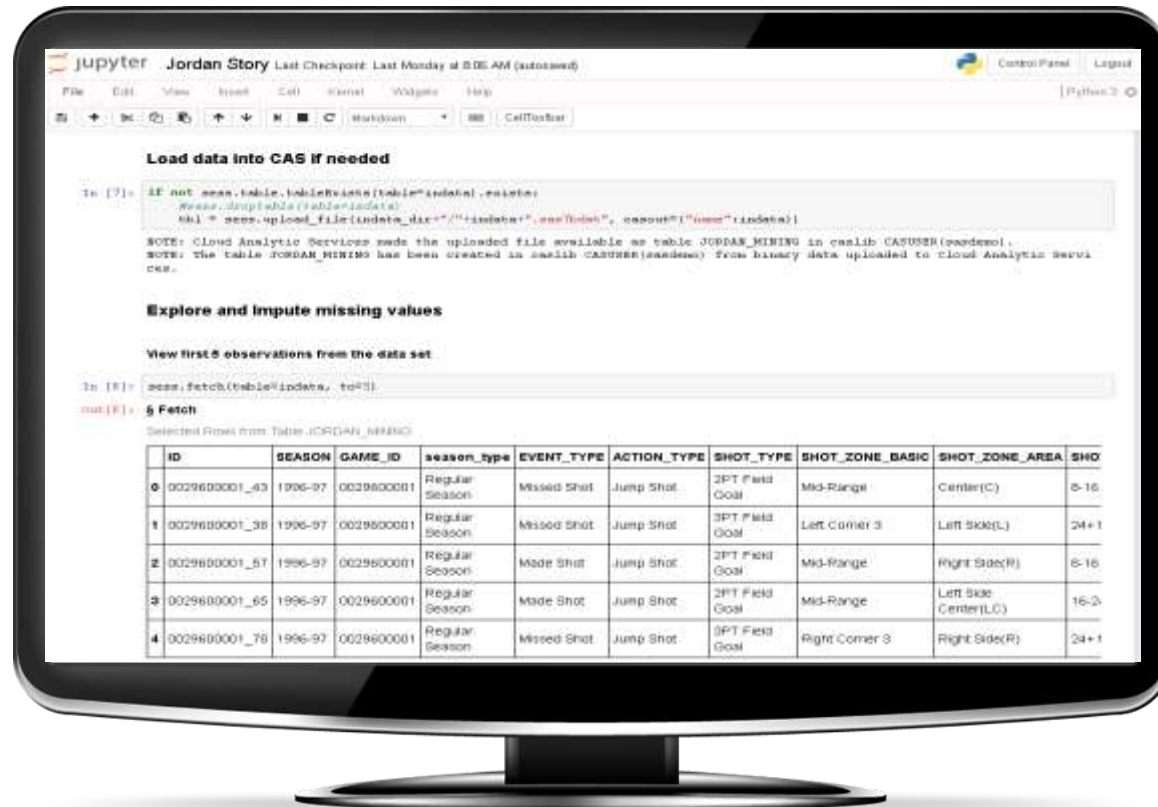
Copyright © SAS Institute Inc. All rights reserved.



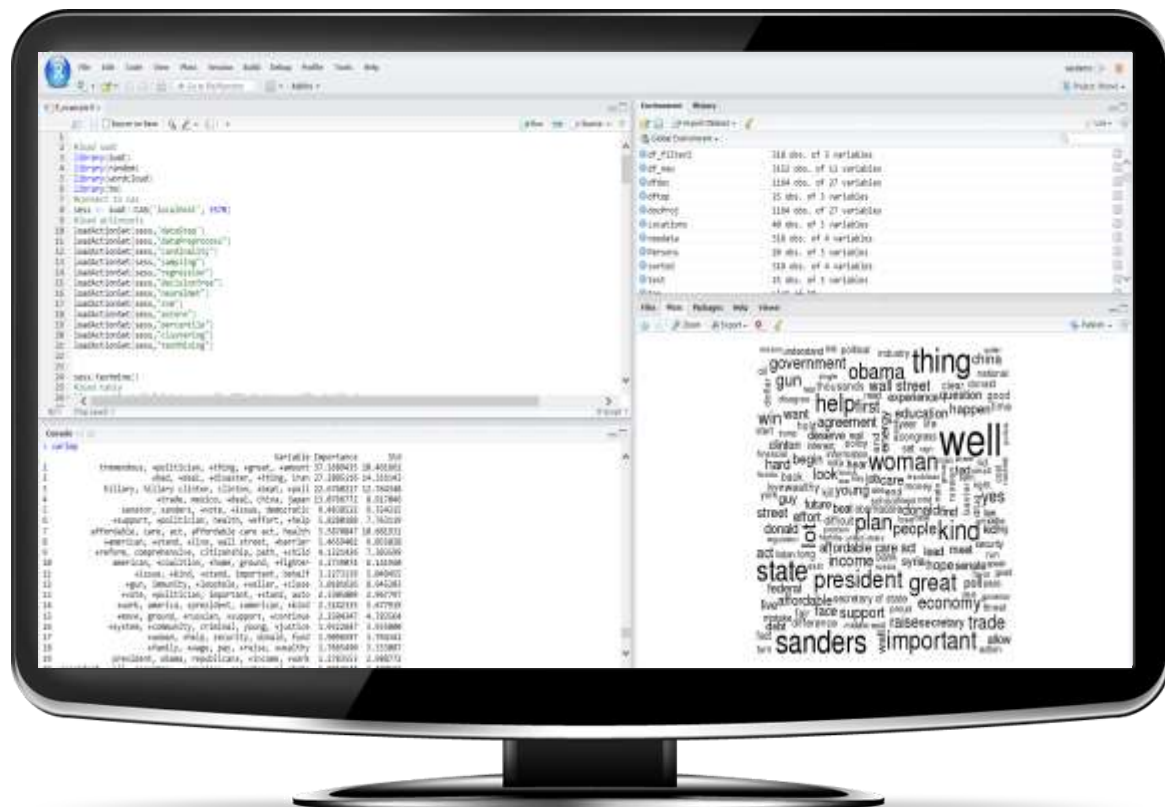
Code Generators



Python API



R



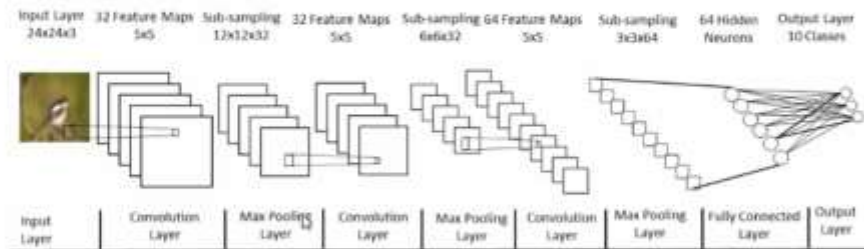
Opening the SAS Analytic Platform via Different Interfaces



Start the SAS Cloud Analytic Server
Load the DeepLearning Action Set

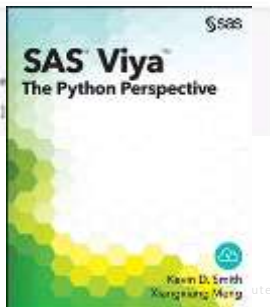
```
jupyter c1b10_tech_exchng_demo Last Checkpoint: 30 minutes ago (unsaved changes)  
File Edit View Insert Cell Help Python  
In [ ]: s = sas.CAS('cas01.sas.com', 8775)  
s.sessionprop.setsessionid('CASUSER', timeout=3.1536E7)  
s.loadactionset('deeplearn')
```

Define the Network Architecture



```
In [ ]: s.createModel(model=dict(name='convnet', replace=True), type='CNN')  
s.addLayer(model='convnet', name='data', type='input',  
            inputOpts=dict(channels=3, width=24, height=24, scale=1))  
s.addLayer(model='convnet', name='conv1', type='convolution',  
            convOpts=dict(nFilters=32, width=5, height=5, stride=1), srcLayers=['data'])  
s.addLayer(model='convnet', name='pool1', type='pooling',  
            poolingOpts=dict(width=2, height=2, pool='max'), srcLayers=['conv1'])  
s.addLayer(model='convnet', name='conv2', type='convolution',
```

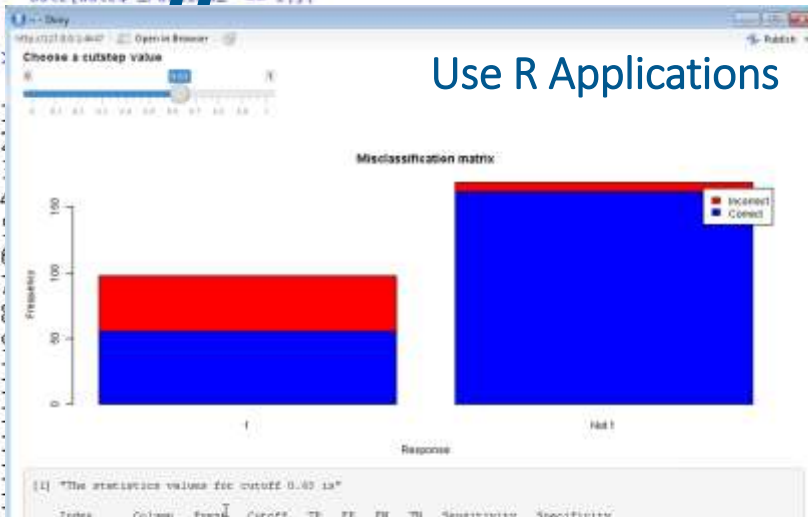
Define the Network Layers



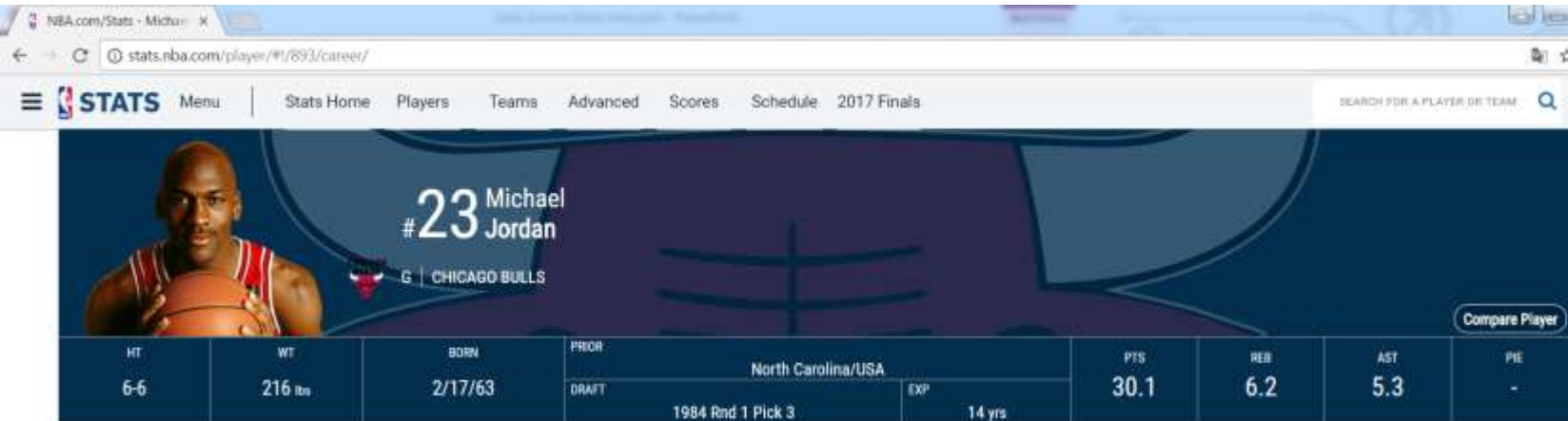
Use the CAS Random Forest
Display the Results in R-Studio

```
RStudio  
File Edit Code View Plots Session Build Debug Profile Tools Help  
Go to file/function Addins  
> dat1 = cas.read.csv(cas, file = '//sashq/root/u/sasyq/titanic_train.csv',  
+                     casOut = list(replace = TRUE))  
NOTE: Cloud Analytic Services made the uploaded file available as table TITANIC_TRAIN in ca  
slib CASUSER(sasyq)  
> rfout = cas.decisionTreeForestTrain(  
+   dat2[dat2$'_PassengerId' == 1,]
```

Use R Applications



Michael Jordan under pressure



The screenshot shows the NBA Stats website for Michael Jordan. The page features a header with the NBA logo and navigation links. Below the header is a large banner image of Michael Jordan in his Chicago Bulls uniform, holding a basketball. To the right of the image, his name and number are displayed. Below the banner is a table of his career statistics.

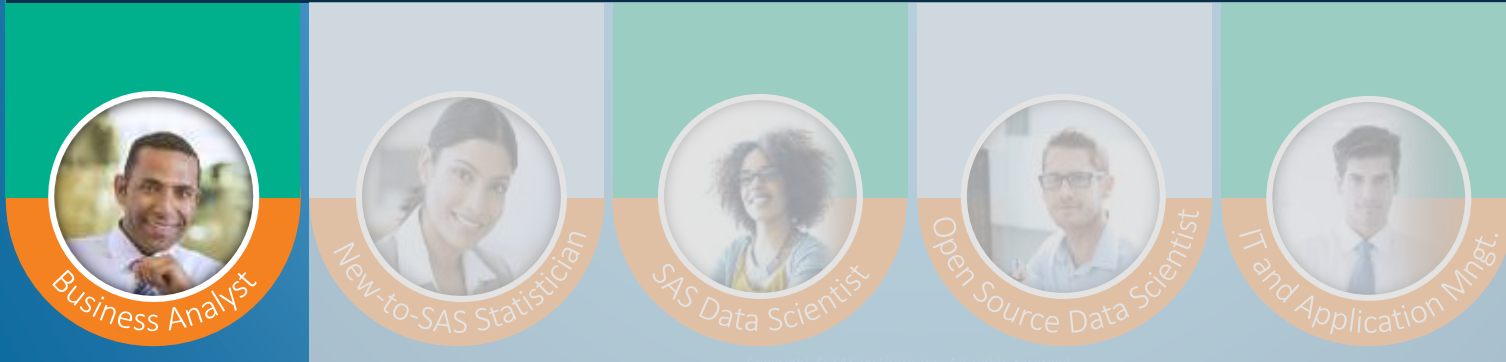
HT	WT	BORN	PRIOR	PTS	REB	AST	PIE
6-6	216 lbs	2/17/63	North Carolina/USA	30.1	6.2	5.3	-
			DRAFT				
			1984 Rnd 1 Pick 3	14 yrs			

<http://stats.nba.com>

Analyse der NBA 1997 Daten durch 4 verschiedene Benutzer-Rollen

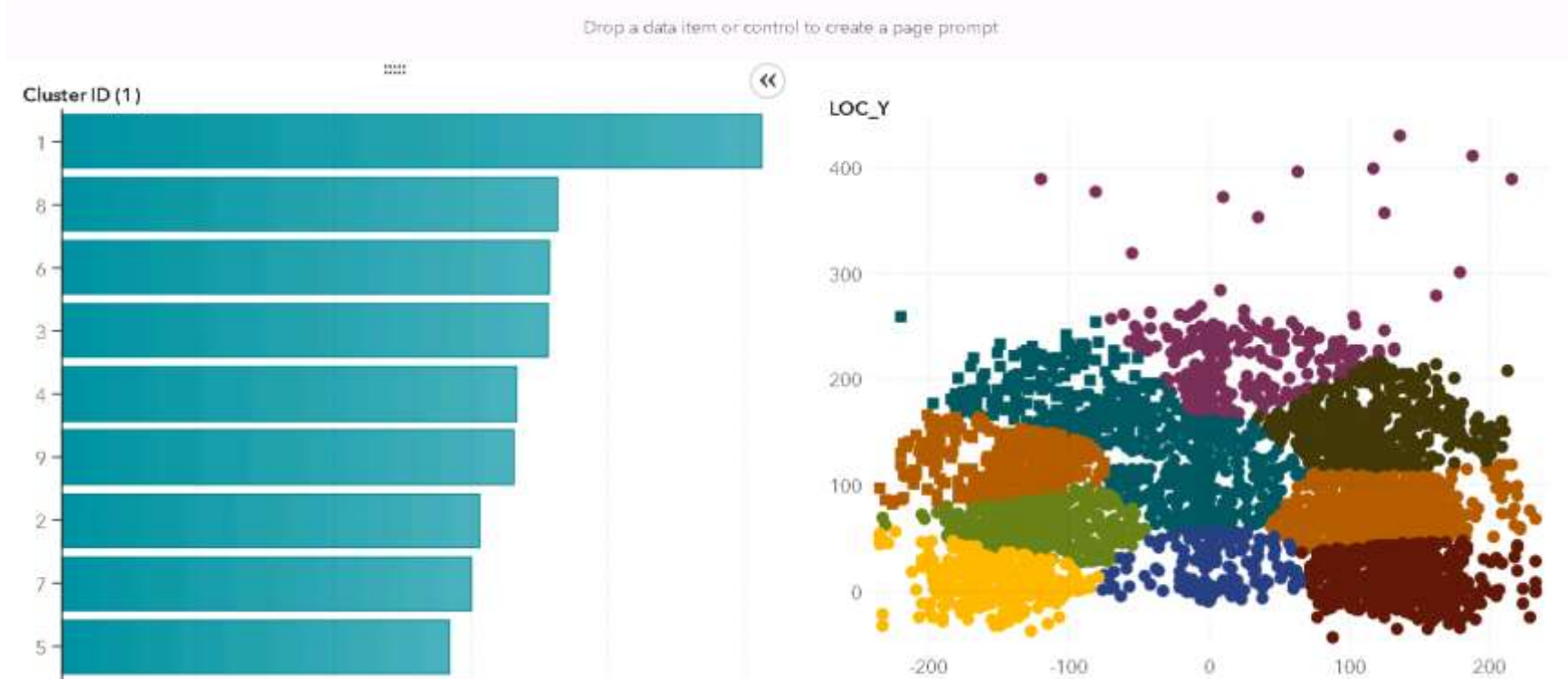
SAS Visual Analytics und SAS Visual Statistics für den Business Analyst

One Integrated Solution for Different User Types



SAS Visual Statistics für den Business Analyst

Point&Click Zugriff auf Machine Learning Methoden



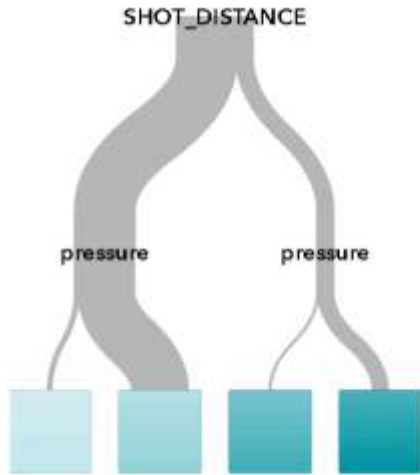
SAS Visual Statistics für den Business Analyst

Point&Click Zugriff auf Machine Learning Methoden

Drop a data item or control to create a page prompt

Decision Tree SHOT_MADE_FLAG ASE 0.245162 Observations Used 4,809

Tree



Variable Importance



Assessment



#SASF17

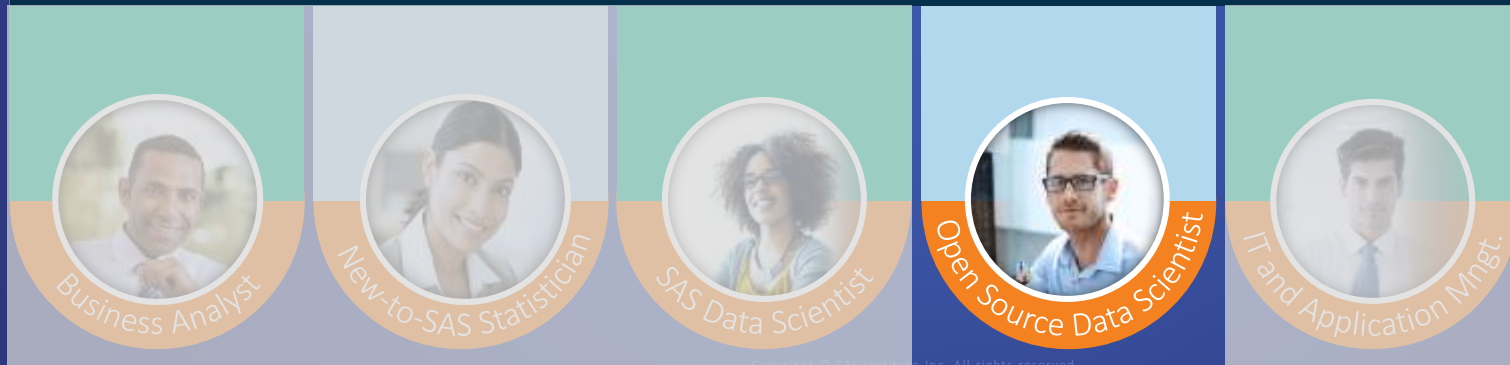


SASF17 / SASForum

Analyse der NBA 1997 Daten durch 4 verschiedene Benutzer-Rollen

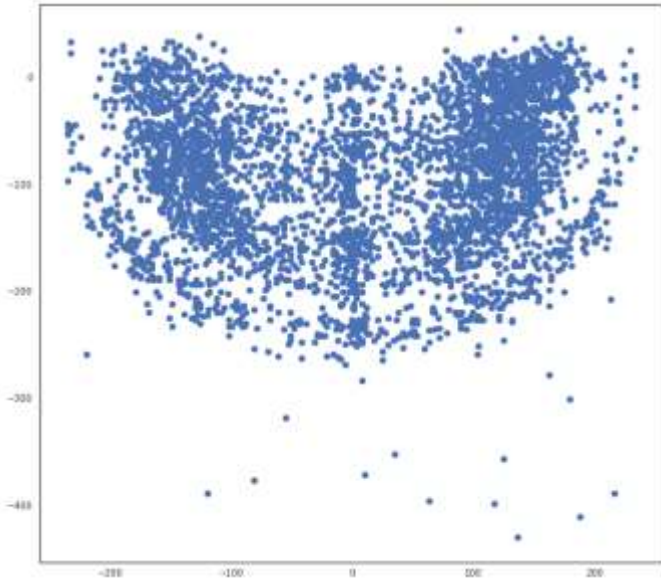
SAS-Python Integration für den Open Source Data Scientist

One Integrated Solution for Different User Types



How good was Jordan under pressure?

Some Graphs benefiting from the open source community



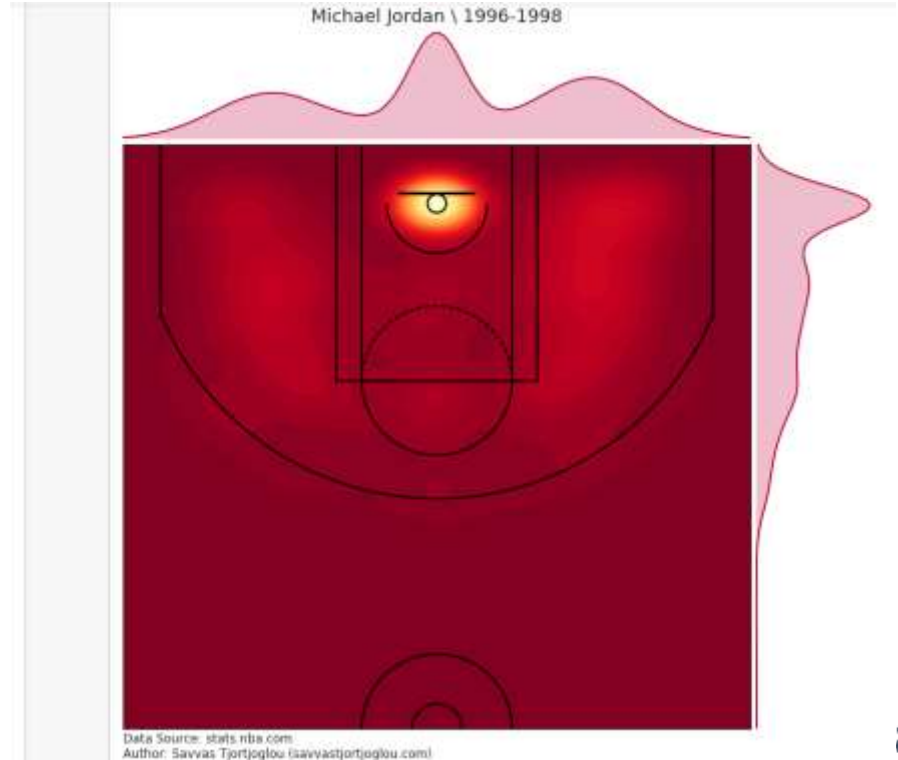
<http://savvastjortjoglou.com/nba-shot-sharts.html>



#SASF17



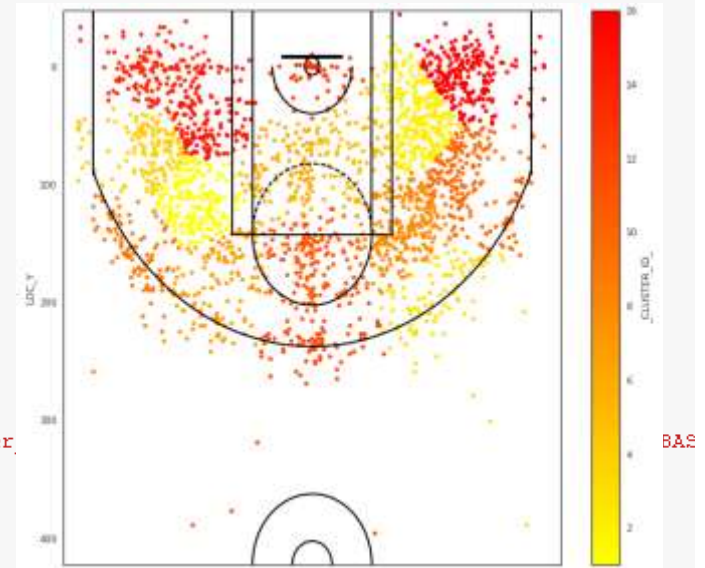
SASF17 / SASForum



How good was Jordan under pressure?

Clustering the Court into Shot Zones using CAS action set

```
clust=sess.clustering.kClus(  
  table={  
    "name":"jordan_mining"  
  },  
  inputs={"LOC_X","LOC_Y_MINUS"},  
  nClusters=30,  
  maxIters=10,  
  distanceNom="RELATIVEFREQ",  
  estimateNClusters={  
    "method":"ABC",  
    "B":10,  
    "minClusters":15,  
    "criterion":"ALL",  
    "align":"PCA"  
  },  
  kPrototypeParams={  
    "method":"USERGAMMA",  
    "value":10  
  },  
  output={"CasOut":{"name":"kClusOutputScore", "replace":True},  
    "copyVars":{"LOC_X","LOC_Y","LOC_Y_MINUS","Made","Lead_Player"},  
  },  
  display={"names":{"Modelinfo", "ClusterSumIntNom"}}  
)
```



How good was Jordan under pressure?

Using CAS Regression to investigate performance under pressure

Logistic

```
In [69]: lr = sess.regression.logistic(  
    table={"name": "kClusOutputScore"},  
    classVars=[{"vars": {"_CLUSTER_ID_", "Lead_Player_Before", "Homegame", "ACTION_TYPE", "end_of_game", "Overtime", "close_game",  
    model={  
        "depVars": [{"name": "Made", "options": {"event": "1"}}],  
        "effects": [{"vars": {"SHOT_DISTANCE", "_CLUSTER_ID_", "ACTION  
    },  
  
    outputTables={"names": "parameterestimates"}  
}  
sess.dataStep.runCode(  
    code="""data round; set parameterestimates(keep=Parameter DF  
do i = 1 to dim(_nums);  
    _nums{i} = round(_nums{i}, .001);  
end;  
drop i;  
run;""")  
sess.fetch(table="round")
```

Out [69]: § Fetch

Selected Rows from Table ROUND

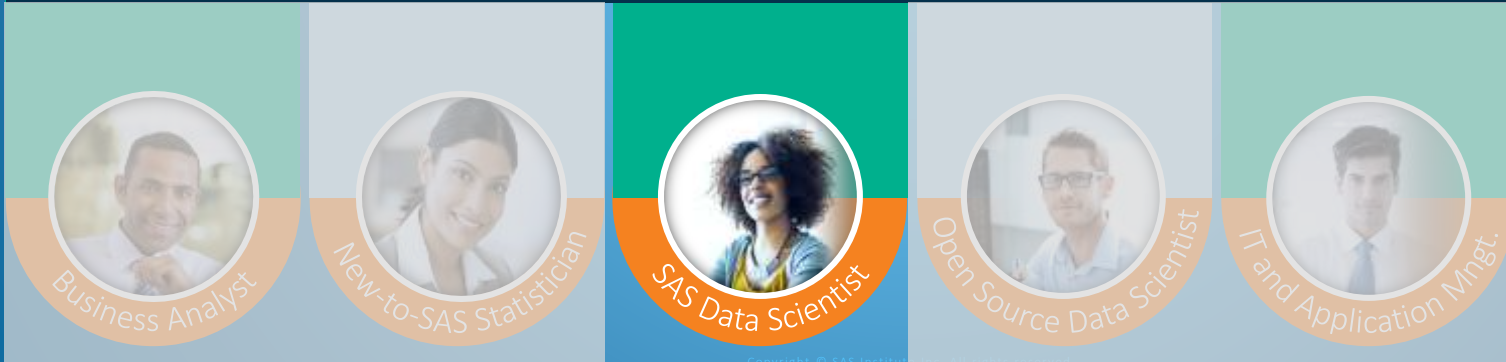
	Parameter	DF	Estimate	StdErr	ChiSq	ProbChiSq
0	Intercept	1.0	14.322	94.876	0.023	0.880
1	pressure_1	1.0	-0.385	0.188	4.193	0.041
2	pressure_0	0.0	0.000	NaN	NaN	NaN
3	Overtime_Regular	1.0	0.382	0.370	1.069	0.301
4	Overtime_Overtime	0.0	0.000	NaN	NaN	NaN
5	SHOT_DISTANCE	1.0	-0.035	0.014	6.500	0.011
6	ACTION_TYPE Tip Shot	1.0	-13.135	94.875	0.019	0.890
7	ACTION_TYPE Slam Dunk Shot	1.0	0.027	126.564	0.000	1.000
8	ACTION_TYPE Running Jump Shot	1.0	-12.634	94.876	0.018	0.894
9	ACTION_TYPE Layup Shot	1.0	-13.064	94.875	0.019	0.890
10	ACTION_TYPE Jump Shot	1.0	-14.106	94.875	0.022	0.882
11	ACTION_TYPE Hook Shot	1.0	-12.552	94.882	0.017	0.895
12	ACTION_TYPE Dunk Shot	1.0	-10.540	94.875	0.012	0.912
13	ACTION_TYPE Driving Layup Shot	1.0	-11.510	94.875	0.015	0.903
14	ACTION_TYPE Driving Dunk Shot	0.0	0.000	NaN	NaN	NaN



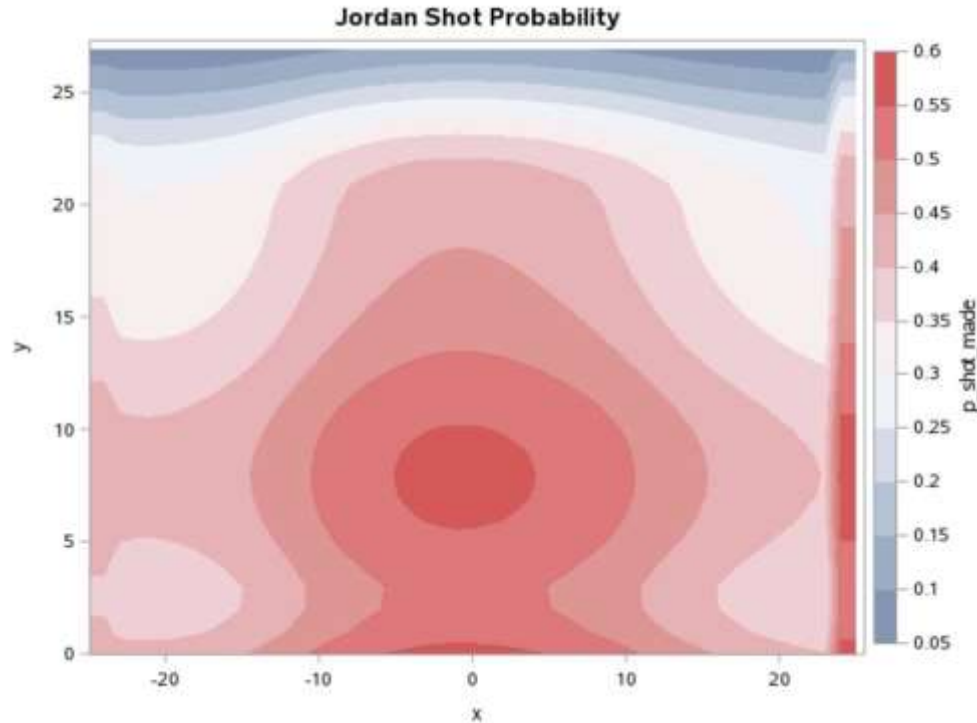
Analyse der NBA 1997 Daten durch 4 verschiedene Benutzer-Rollen

SAS Procedures für den SAS Data Scientist

One Integrated Solution for Different User Types



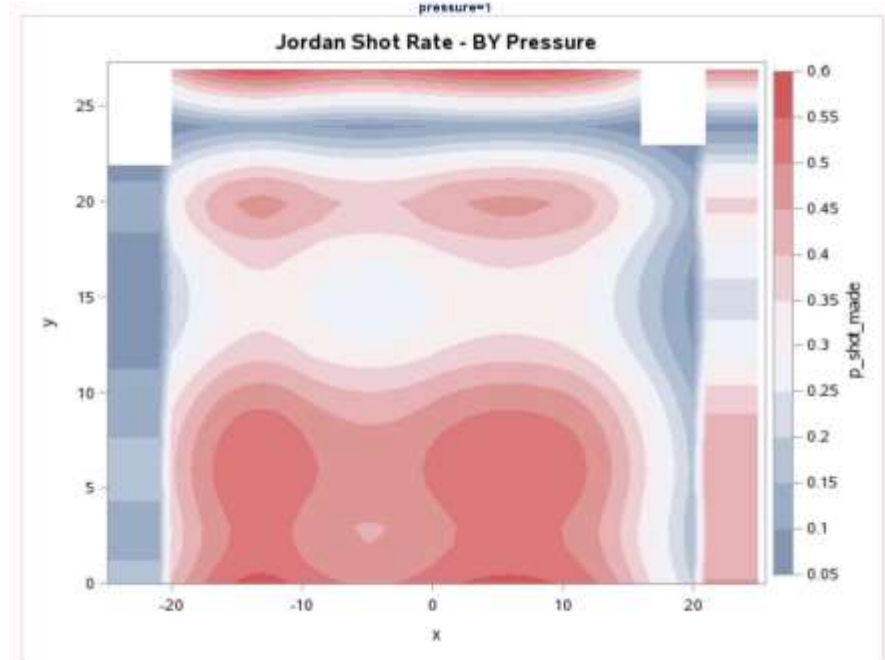
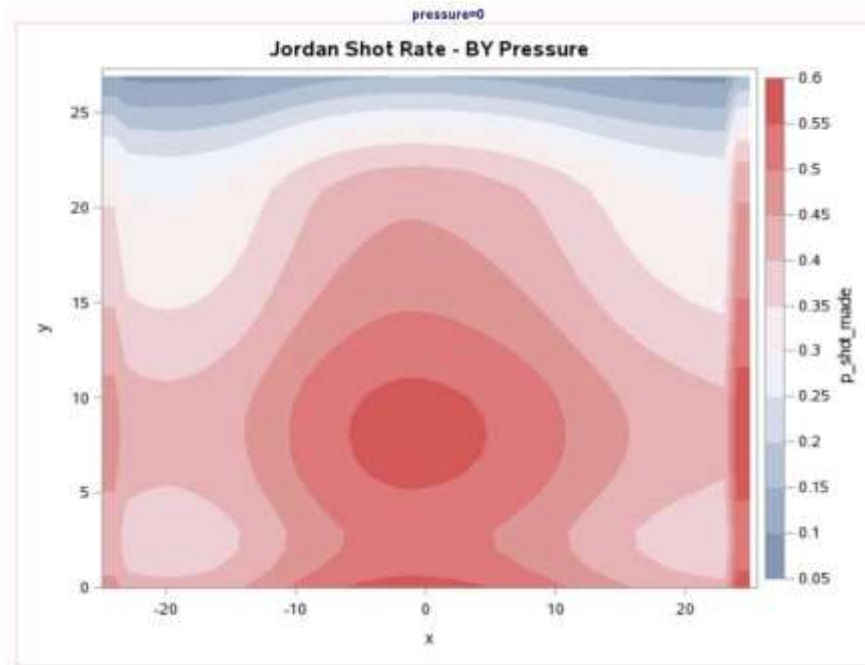
Vorhersage der Treffer-Wahrscheinlichkeit von Jordan im $[-25,25] \times [0,27]$ Grid



```
proc logselect data=sfdcas.Jordan;  
  where Shot_Distance <= 30;  
  effect spl = spline(X Y / degree=2);  
  model Shot_Made(event='1') = spl ;  
  output out=sfdcas.Jordan_pred  
  pred=p copyvars=(x y shot_distance  
  shot_made);  
  Code file='/opt/sasinside/  
  DemoData/SFD/JordanPred_0.sas';  
run;
```



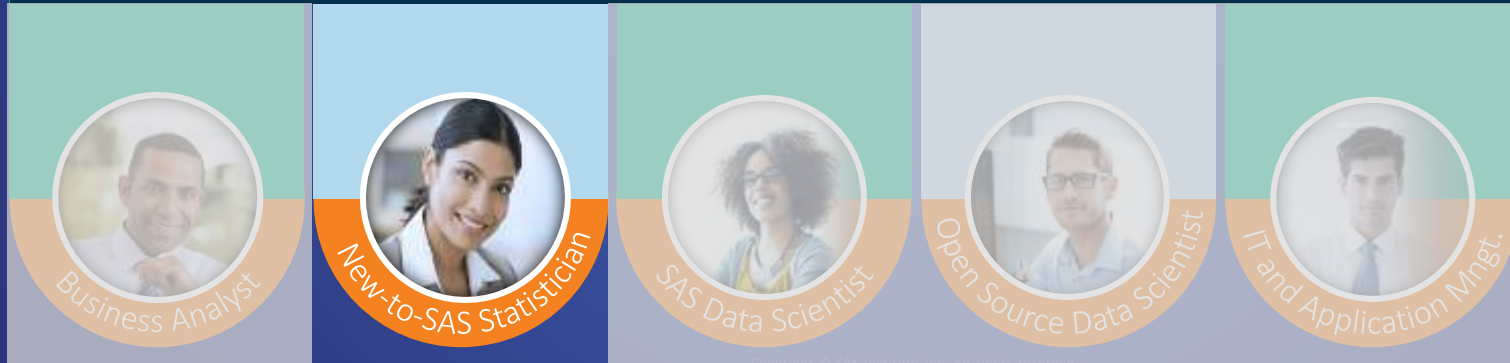
Vorhersage der Treffer-Wahrscheinlichkeit von Jordan getrennt nach Pressure ja/nein



Analyse der NBA 1997 Daten durch 4 verschiedene Benutzer-Rollen

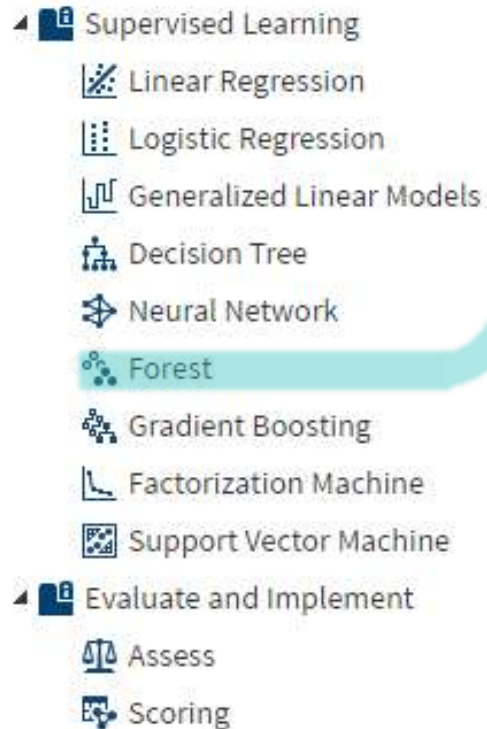
Data Mining/Machine Learnings Tasks im SAS Studio für den „New-to-SAS“ Statistician

One Integrated Solution for Different User Types



Data Mining und Machine Learnings Tasks im SAS Studio

Beispiel: Random Forests



FOREST

Forest or PROC FOREST used for classification models.

Generates many trees from different samples of training data.

Mode of all predictions is final prediction

Forest Competitive Differentiators

- Distributed and massively parallel
- Faster, more memory-efficient, and more scalable algorithm
- Deployable – Generated rules



Vordefinierte Tasks

SAS Studio

Code Generierung mit Tasks

Code Generierung

Optionen

Vordefinierte Tasks

SAS Studio

Code Generierung mit Tasks

The screenshot displays the SAS Studio interface with the 'Train and Validate Model' task selected. The left sidebar shows a tree of tasks, with 'Train and Validate Model' highlighted. The main window shows the configuration for this task, including options for the model type (Logistic Regression), the number of iterations (1000), and the number of bootstrap samples (100). The 'Options' section is expanded, showing various settings for the model, such as the number of iterations, the number of bootstrap samples, and the number of cross-validation folds. The 'Results' section shows a line graph titled 'Fehlklassifikationen nach Anzahl der Bäume' (Misclassification rates after number of trees) and a horizontal bar chart titled 'Variablenbedeutung' (Variable importance).

Options

Ergebnisse



#SASF17



SASF17 / SASForum

Vordefinierte Tasks

SAS Studio

Code Generierung mit Tasks

Aktivierung der
Autotuning
Funktionalität

Ergebnisse

SAS Studio

Code Generierung mit Tasks

Tuner Results Default and Best Configurations					
Evaluation	Maximum Tree Levels	Number of Trees	Number of Variables to Try	Bootstrap	Misclassification Error Percentage
0	21	100	14	0.600000	41.30
12	14	80	5	0.795267	40.06
45	14	80	5	0.791425	40.26
39	12	72	7	0.725718	40.54
30	12	70	7	0.705657	40.61
3	2	20	14	0.277778	40.68
17	5	34	12	0.403667	40.68
22	4	22	13	0.335616	40.68
24	5	34	12	0.394741	40.68
32	2	20	14	0.218658	40.68
33	6	33	12	0.415225	40.68



SAS Studio

Code Generierung mit Tasks

