Julius Kühn-Institut
Bundesforschungsinstitut für Kulturpflanzen
Federal Research Centre for Cultivated Plants

Kann SAS® Diskriminanzfunktion?

Jörg Sellmann

joerg.sellmann@julius-kuehn.de

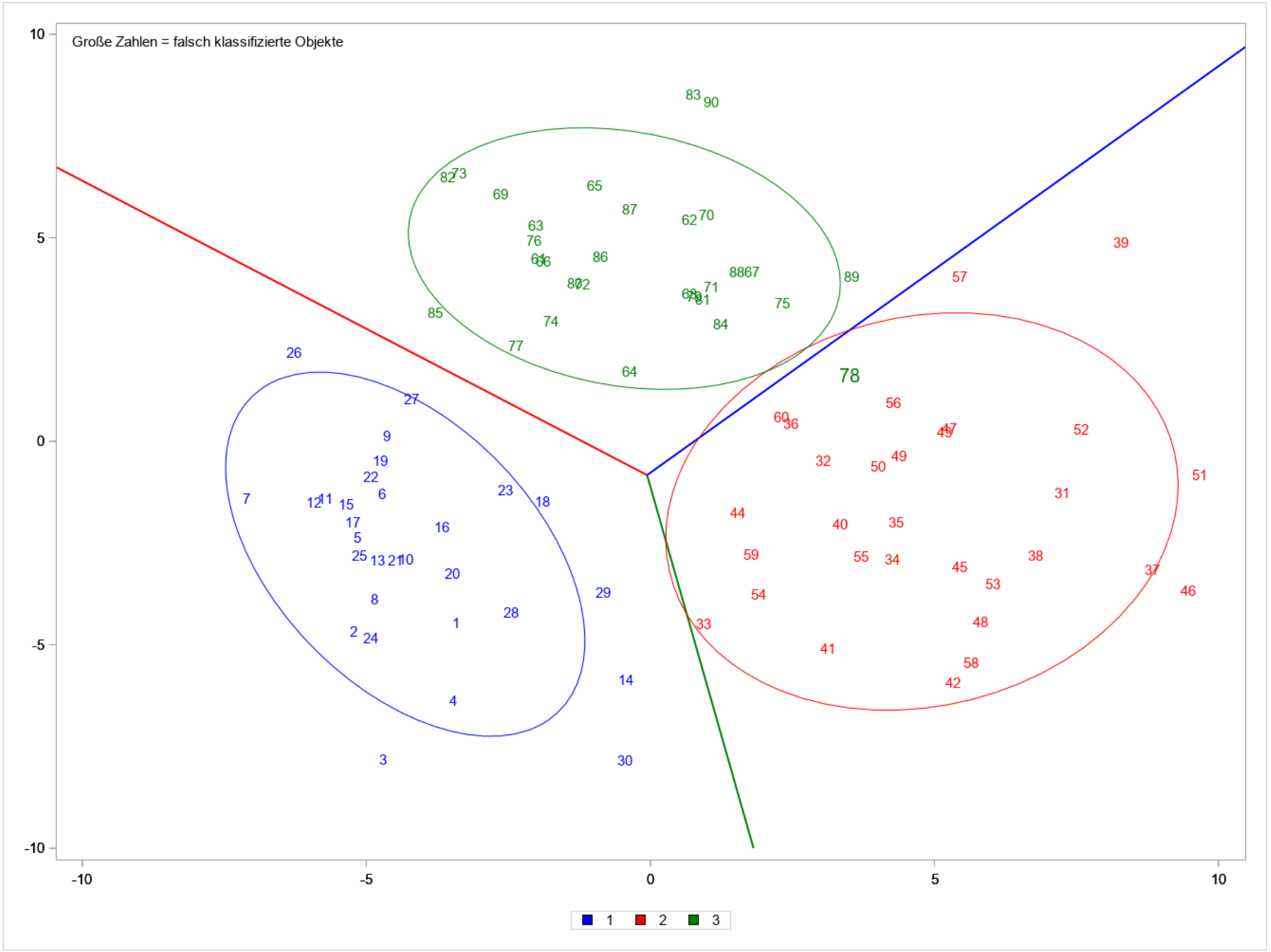
- **Zusammenfassung**

Die (lineare) Diskriminanzanalyse hat zum Ziel, anhand von Objekten, an denen mehrere Merkmale gemessen wurden, die aber a-priori einer bestimmten Klasse angehören, lineare Grenzen zwischen diesen zwei oder mehr Klassen zu ziehen, um neue Objekte einer dieser Klassen zuzuordnen.

SAS bietet mit der Prozedur `discrim` ein Werkzeug, diese Trennung(n) zu berechnen. Doch leider gibt es keine Grafik, die die lineare(n) Trennfunktion(en) zeichnet. Oder aber, ich kenne sie nicht.

Anhand der im Output zu findenden Daten ist es jedoch möglich, die Parameter der Funktion(en) zu errechnen und dann mittels `sgplot` auch grafisch in der Ebene darzustellen. Darüber hinaus kann anhand der ODS-Ergebnistabellen ermittelt werden, welche a-priori Objekte falsch klassifiziert würden. Dieses kann mittels Kodierung ebenso in der `sgplot`–Grafik verdeutlicht werden.

Als Nebeneffekt ist es dann ebenso möglich, die Wirkung der verwendeten a-priori Wahrscheinlichkeiten auf die Lage der Trennfunktionen anschaulich darzustellen.



SAS® kann,

aber wie?

- **Meine Zwischenschritte mit Stift und Papier → Mathematik Klasse 8?**

(a1+b1*y)/c1 = (a2+*b2*y)/c2

a1=2*&c12.
b1=(&p12.-&p22.)
c1=(&p11.-&p21.)

a2=2*&c23.
b2=(&p22.-&p32.)
c2=(&p21.-&p31.)

c2*(a1+b1*y) = c1*(a2+*b2*y)

c2*a1 - c1*a2 = c1*b2*y -c2*b1*y

c2*a1 - c1*a2 = y * (c1*b2 -c2*b1)

y = (c2*a1 - c1*a2)/(c1*b2 -c2*b1)

zero = ((&p21.-&p31.)*2*&c12. - (&p11.-&p21.)*2*&c23.) / ((&p11.-&p21.)*(&p22.-&p32.) - (&p21.-&p31.)*(&p12.-&p22.))

zero = 2 * ((&p21.-&p31.)*&c12. - (&p11.-&p21.)*&c23.) / ((&p11.-&p21.)*(&p22.-&p32.) - (&p21.-&p31.)*(&p12.-&p22.))

- **Was braucht ein SAS-Anwender?**

1. Den wichtigsten Tipp, den Ralf Minkenberg uns/mir 2014 mit auf den Weg gegeben hat.

2. Die Fähigkeit des verstehenden Lesens.

- **Der wichtigste Tipp**

```
ods trace on;

proc discrim
…
run;

ods trace off;
```

```
Output Added:
-------------
Name:         LinearDiscFunc
Label:        Linear Discriminant Function
Template:     Stat.Discrim.LinearDiscFunc
Path:         Discrim.LinearDiscFunc
-------------
```

- **Verstehendes Lesen**

- Den SAS-Output lesen und verstehen.
- Das Log lesen und verstehen.
- <F1> vernünftig nutzen.
- Aus allen Drei das heraus ziehen können, um ohne manuelles Copy/Paste einen vernünftigen Output zu generieren.

- **Das Programm - Konstanten**

```
ODS GRAPHICS / height=1000;

%LET sizemin=12px;                  /* Minimale Textgröße */

%LET ymin=-10;                      /* Y-Achse von */
%LET ymax=10;                       /* Y-Achse bis */
%LET xmin=-10;                      /* X-Achse von */
%LET xmax=10;                       /* X-Achse bis */
%LET step=5;                        /* Schrittweite */

%LET alpha=0.2;                     /* Größe der Streuungs-Ellipse */

%LET HK1=Prin1;                     /* Wahl der 1. Hauptachse */
%LET HK2=Prin2;                     /* Wahl der 2. Hauptachse */

%LET priors='1'=0.333 '2'=0.333 '3'=0.333;
                                    /* A-priori-Wahrscheinlichkeiten der 3 Gruppen */

%LET wdh=30;                        /* Stichprobenumfang pro Test-Gruppe */
%LET diff=4;                        /* Auseinanderschieben der 3 Test-Gruppen */
```

- **Das Programm - Gibt es Fehlklassifikationen?**

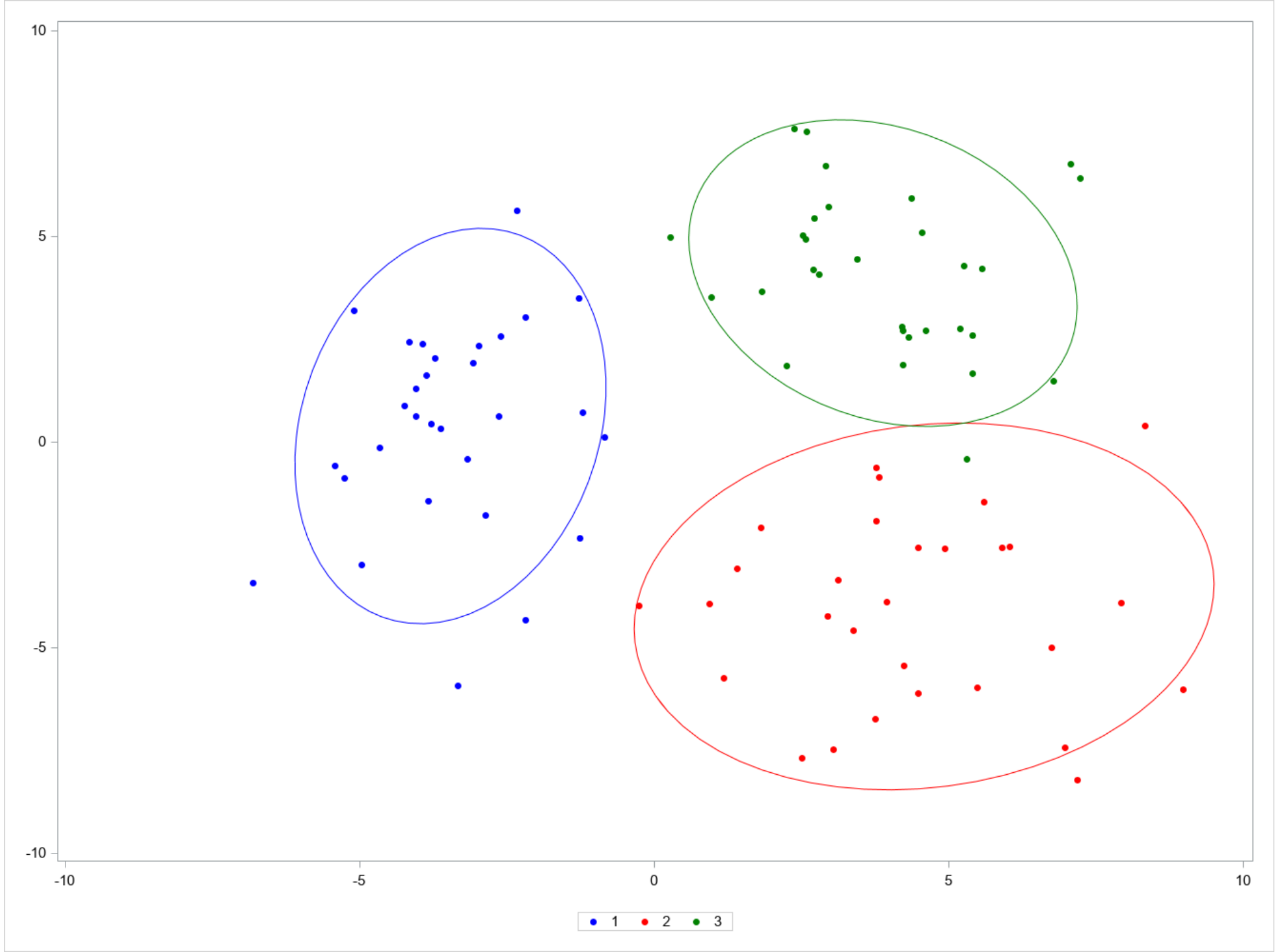
```
/* Sind im Datensatz Fehlklassifikationen enthalten? */
%macro sizemax(data);

        %Global sizemax;

        %LET sizemax=16px;

        proc sql noprint;
        select count(*) into :anz from &data. where size=2;
        quit;

        /* Wenn keine Fehlklassifizierung (size=2) --> sizemax=sizemin */
        %if &anz.=0 %then %LET sizemax=&sizemin.;

%mend;
```

- **Das Programm - Testdaten erstellen**

```
/* Ellipsoide 2-dimensionale Normalverteilung */
data tmp;
call streaminit(123);
do nr=1 to 3*&wdh.;
x=rand("Normal",0,2);
y=rand("Normal",0,3);
output;
end;
run;

/* Verschieben an 3 Punkte mit Abstand 2*Diff */
data three;
set tmp;
if nr<=&wdh. then do; grp=1; x=x-&diff.; end;
if (&wdh.+1)<=nr<=(2*&wdh.) then do; grp=2; t=x; x=y+&diff.; y=t-&diff.; end;
if (2*&wdh.+1)<=nr then do; grp=3; t=x; x=y+&diff.; y=t+&diff.; end;
drop t;
run;
```

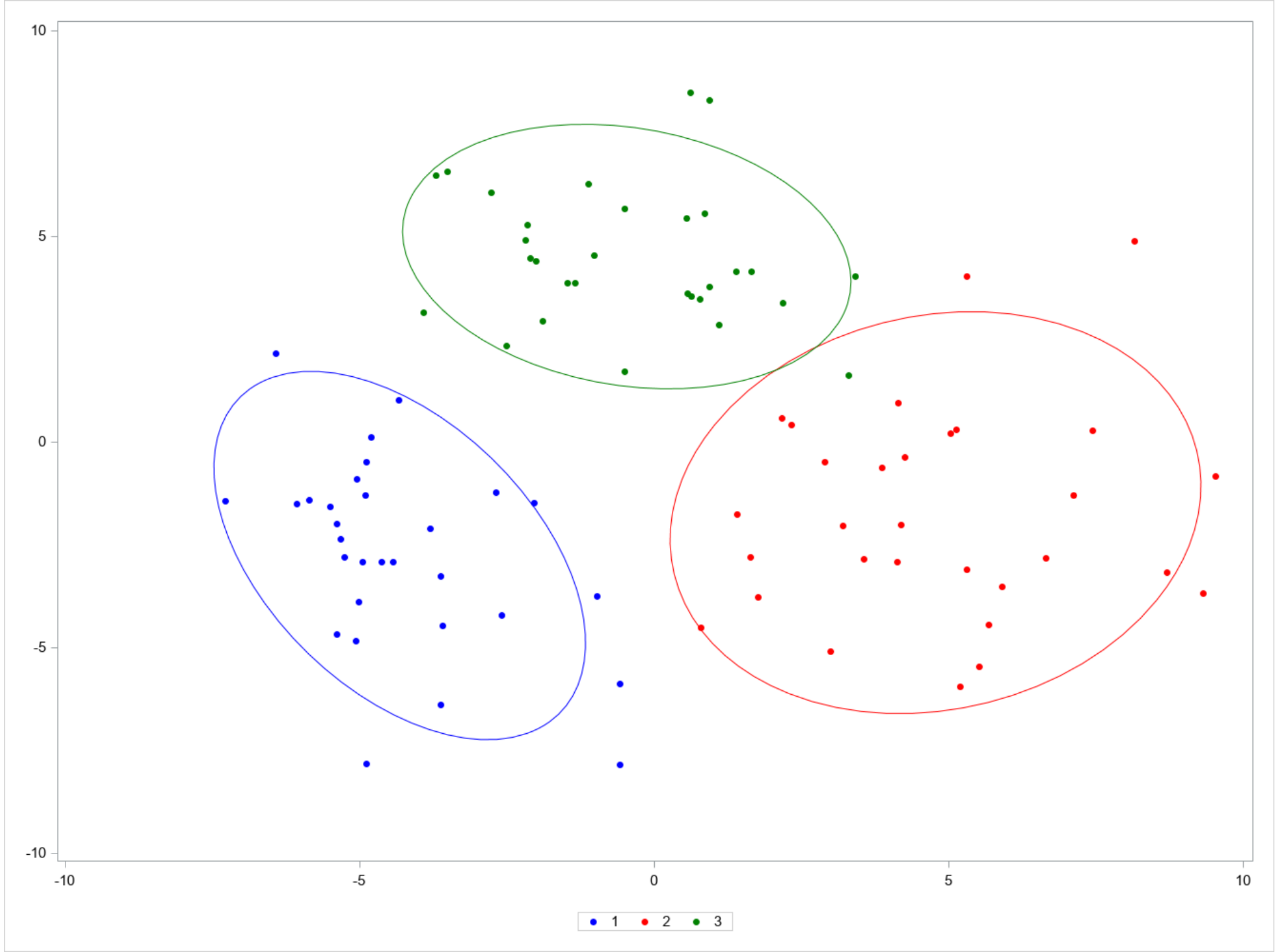
- **Das Programm - Testdaten ansehen**

```
/* Anschauen */
proc sgplot data=three;

styleattrs datacolors=(blue red green) datacontrastcolors=(blue red green);

scatter x=x y=y / group=grp markerattrs=(symbol=circlefilled);

ellipse x=x y=y / group=grp alpha=&alpha.;

xaxis display=(nolabel) values=(&xmin. to &xmax. by &step.);
yaxis display=(nolabel) values=(&ymin. to &ymax. by &step.);

label grp='00'x;

run;
```


10
5
0
-5
-10
-10
-5
0
5
10
1
2
3

- **Das Programm - Hauptkomponenten-Transformation**

```
/* Hauptachsentransformation */
proc princomp data=three out=pcathree cov;
id nr grp;
var x y;
run;

/* Anschauen */
proc sgplot data=pcathree;
styleattrs datacolors=(blue red green) datacontrastcolors=(blue red green);

scatter x=&HK1. y=&HK2. / group=grp markerattrs=(symbol=circlefilled);

ellipse x=&HK1. y=&HK2. / group=grp alpha=&alpha.;

xaxis display=(nolabel) values=(&xmin. to &xmax. by &step.);
yaxis display=(nolabel) values=(&ymin. to &ymax. by &step.);
label grp='00'x;

run;
```


10
5
0
-5
-10
-10
-5
0
5
10
1
2
3

- **Das Programm - Diskriminanzanalyse**

```
/* Diskriminanzanalyse mit den Originaldaten */
proc discrim data=three pool=yes method=normal listerr;
class grp;
priors &priors.;
var x y;
run;

/* Diskriminanzanalyse mit den transformierten Daten */
proc discrim data=pcathree method=normal pool=yes listerr outstat=stats out=outerror;
class grp;
priors &priors.;
var &HK1. &HK2.;
run;
```

Oder:

```
ods output LinearDiscFunc=stats;
proc discrim data=irispca method=normal pool=yes listerr out=outerror;
class grp;
priors &priors.;
var prin:;
run;
```

- **Das Programm - Diskriminanzanalyse**

The SAS System

The DISCRIM Procedure

Generalized Squared Distance to grp			
From grp	1	2	3
1	2.19722	23.51056	18.28464
2	23.51056	2.19722	15.13821
3	18.28464	15.13821	2.19722

Linear Discriminant Function for grp			
Variable	1	2	3
Constant	-4.53531	-4.01090	-3.13992
Prin1	-1.15061	1.13927	0.01134
Prin2	-0.68844	-0.22042	0.90886

- **Das Programm - Ergebnistabellen**

VIEWTABLE: Work.Stats (DISCRIMINANT CALIBRATION INFORMATION)

	grp	_TYPE_	_NAME_	Prin1	Prin2
61	.	LNDETER		2.9925854255	2.9925854255
62	1	LINEAR	_LINEAR_	-1.150609251	-0.688440094
63	1	LINEAR	_CONST_	-4.535307275	-4.535307275
64	2	LINEAR	_LINEAR_	1.1392685015	-0.22041992
65	2	LINEAR	_CONST_	-4.010902753	-4.010902753
66	3	LINEAR	_LINEAR_	0.0113407497	0.9088600138
67	3	LINEAR	_CONST_	-3.139916473	-3.139916473

VIEWTABLE: Work.Stats (Linear Discriminant Function)

	Variable	_1	_2	_3
1	Constant	-30.05954	-3.02893	-17.94902
2	Prin1	-25.00647	5.98287	19.02360
3	Prin2	5.06631	-1.88032	-3.18599
4	Prin3	-16.76299	3.10501	13.65798
5	Prin4	-16.39908	5.51683	10.88225

- **Das Programm - Diskriminanzanalyse**

The SAS System

The DISCRIM Procedure
Classification Results for Calibration Data: WORK.PCATHREE
Resubstitution Results using Linear Discriminant Function

Posterior Probability of Membership in grp						
Obs	From grp	Classified into grp		1	2	3
78	3	2	*	0.0001	0.7352	0.2647

* Misclassified observation

- **Das Programm - Ergebnistabellen**

VIEWTABLE: Work.Outerror

	nr	x	y	grp	Prin1	Prin2	_1	_2	_3	_INTO_
74	74	1.8280700708	3.6571925258	3	-1.888606125	2.9466003643	0.0196600124	0.0017460962	0.9785938915	3
75	75	5.4054326758	1.6680419353	3	2.1816125741	3.3796858322	0.0000801828	0.0973532521	0.9025665651	3
76	76	2.7184311114	5.4380859317	3	-2.185876505	4.9153447425	0.0012209182	0.0001379448	0.998641137	3
77	77	0.9663896456	3.5095420427	3	-2.508158843	2.3297954926	0.0993742392	0.0016004928	0.899025268	3
78	78	5.3035824056	-0.415026687	3	3.2975772568	1.6178200055	0.0001072589	0.7351801612	0.26471258	2
79	79	4.229751448	2.6986270316	3	0.6269940633	3.5454967974	0.0004085372	0.0152462864	0.9843451764	3
80	80	2.7007646088	4.1799735097	3	-1.476013969	3.8764731659	0.0028058997	0.0009905498	0.9962035505	3
81	81	4.313663702	2.5564314025	3	0.7774686772	3.4775390614	0.000380719	0.0194253245	0.9801939564	3
82	82	2.3776356388	7.6048910814	3	-3.711980238	6.4908426026	0.0005810555	4.1666613E-6	0.9994147779	3
83	83	7.0668345735	6.756858336	3	0.6103833092	8.4970628187	1.5543443E-7	0.0000566773	0.9999431673	3
84	84	4.2164830466	1.8651449549	3	1.095988882	2.8563579995	0.0006839085	0.0541082702	0.9452078213	3
85	85	0.276374	4.9866583254	3	-3.922743262	3.1403169444	0.1354586857	0.0001249552	0.8644163591	3
86	86	3.4469458606	4.4520128537	3	-1.022511509	4.5284910979	0.0005860983	0.0007930537	0.998620848	3
87	87	4.541424186	5.0006769827	3	0.495282811	5.6809050821	0.0000504621	0.0002915912	0.9995579467	3

- **Das Programm - Parameter ermitteln**

```
/* Variable mit den falsch klassifizierten Objekten erstellen */
proc sql noprint;
select nr into :wrong separated by ' ' from
(select 0 as nr from outerror union select nr from outerror where grp <>_INTO_);
quit;
```

%put &wrong;
0 78

- **Das Programm - Parameter ermitteln**

```
 /* Diskriminanzfuntionen laden */
proc sql noprint;
select &HK1.,&HK2. into :p11,:p12 from stats where grp=1 and _name_='_LINEAR_';
select &HK1.,&HK2. into :p21,:p22 from stats where grp=2 and _name_='_LINEAR_';
select &HK1.,&HK2. into :p31,:p32 from stats where grp=3 and _name_='_LINEAR_';
select &HK1. into :p01 from stats where grp=1 and _name_='_CONST_';
select &HK1. into :p02 from stats where grp=2 and _name_='_CONST_';
select &HK1. into :p03 from stats where grp=3 and _name_='_CONST_';
quit;
```

Oder:

```
proc sql noprint;
select _1,_2,_3 into :p11,:p21,:p31 from stats where Variable="&HK1.";
select _1,_2,_3 into :p12,:p22,:p32 from stats where Variable="&HK2.";
select _1,_2,_3 into :p01,:p02,:p03 from stats where Variable='Constant';
quit;
```

```
%put &p01. &p02.  &p03.    &p11.    &p12.    &p21.    &p22.    &p31.     &p32.;
-4.53531  -4.0109 -3.13992 -1.15061 -0.68844 1.139269 -0.22042 0.011341  0.90886
```

- **Das Programm - Trennlinien errechnen ← Versuch und Irrtum ☺**

```
/* Berechnung der Konstanten aus den Diskriminanzfuntionen */
%LET c12 = %sysevalf((&p01. - &p02.)/2);
%LET c13 = %sysevalf((&p01. - &p03.)/2);
%LET c23 = %sysevalf((&p02. - &p03.)/2);

/* Berechnung der Trennlinien von jeweils 2 Gruppen */
data trennlinie12;
y12=&ymin.; x12=-(2*&c12.+(&p12.-&p22.)*y12)/(&p11.-&p21.); output;
y12=&ymax.;  x12=-(2*&c12.+(&p12.-&p22.)*y12)/(&p11.-&p21.); output;
run;

data trennlinie23;
y23=&ymin.; x23=-(2*&c23.+(&p22.-&p32.)*y23)/(&p21.-&p31.); output;
y23=&ymax.;  x23=-(2*&c23.+(&p22.-&p32.)*y23)/(&p21.-&p31.); output;
run;

data trennlinie13;
y13=&ymin.; x13=-(2*&c13.+(&p12.-&p32.)*y13)/(&p11.-&p31.); output;
y13=&ymax.;  x13=-(2*&c13.+(&p12.-&p32.)*y13)/(&p11.-&p31.); output;
run;
```

- **Das Programm - Trennlinien errechnen**

VIEWTABLE: Work.Trennlinie12

	y12	x12
1	-10	1.8148513524
2	10	-2.272875554

VIEWTABLE: Work.Trennlinie13

	y13	x13
1	-10	12.545804427
2	10	-14.94760967

VIEWTABLE: Work.Trennlinie23

	y23	x23
1	-10	-9.239791902
2	10	10.78418126

- **Das Programm - Trennlinien errechnen**

```
/* Merken der Fehlklassifikationen */
proc sql noprint;

select nr into :wrong1 separated by ' ' from
(select 0 as nr from outerror
 union select nr from outerror where grp <>_INTO_ and _INTO_=1);

select nr into :wrong2 separated by ' ' from
(select 0 as nr from outerror
 union select nr from outerror where grp <>_INTO_ and _INTO_=2);

select nr into :wrong3 separated by ' ' from
(select 0 as nr from outerror
 union select nr from outerror where grp <>_INTO_ and _INTO_=3);

quit;
```

```
%put &wrong1. &wrong2. &wrong3. ;
     0          0 78       0
```

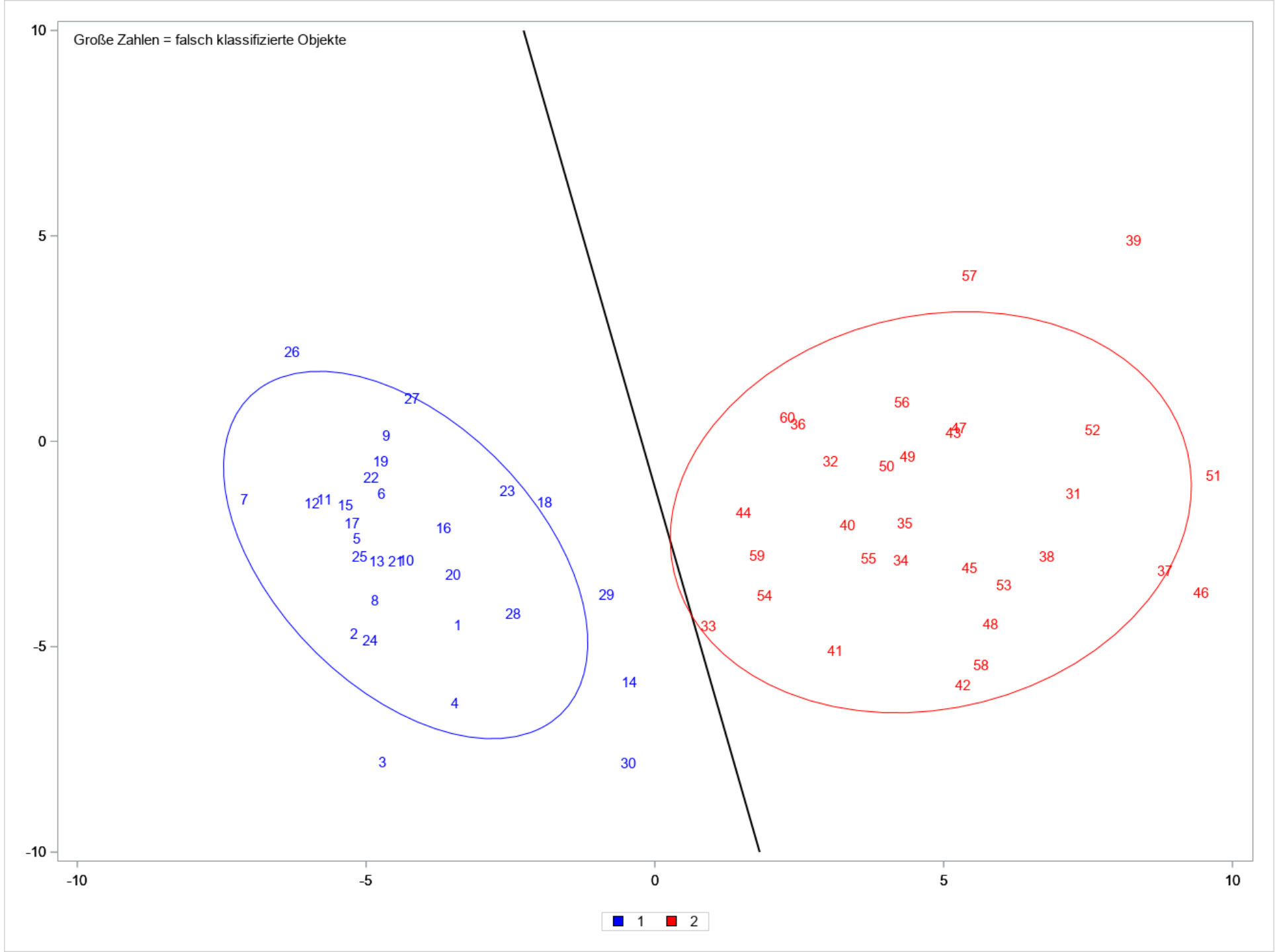
- **Das Programm - Trennlinien von 2 Gruppen errechnen**

```
/* Gruppen 2 und 3 */
data threelinie23;
merge pcathree (where=(grp<>1)) trennlinie23;
size=1;
if NR in (&wrong2. &wrong3.) then size=2;
run;

%sizemax(threelinie23);

proc sgplot data=threelinie23;
styleattrs datacolors=(red green) datacontrastcolors=(red green);
inset "Große Zahlen = falsch klassifizierte Objekte";
text x=&HK1. y=&HK2. text=nr /
     group=grp sizeresponse=size sizemin=&sizemin. sizemax=&sizemax.;
series x=x23 y=y23 / lineattrs=(color=black thickness=2);
ellipse x=&HK1. y=&HK2. / group=grp alpha=&alpha.;
xaxis display=(nolabel) values=(&xmin. to &xmax. by &step.);
yaxis display=(nolabel) values=(&ymin. to &ymax. by &step.);
label grp='00'x;
run;
```

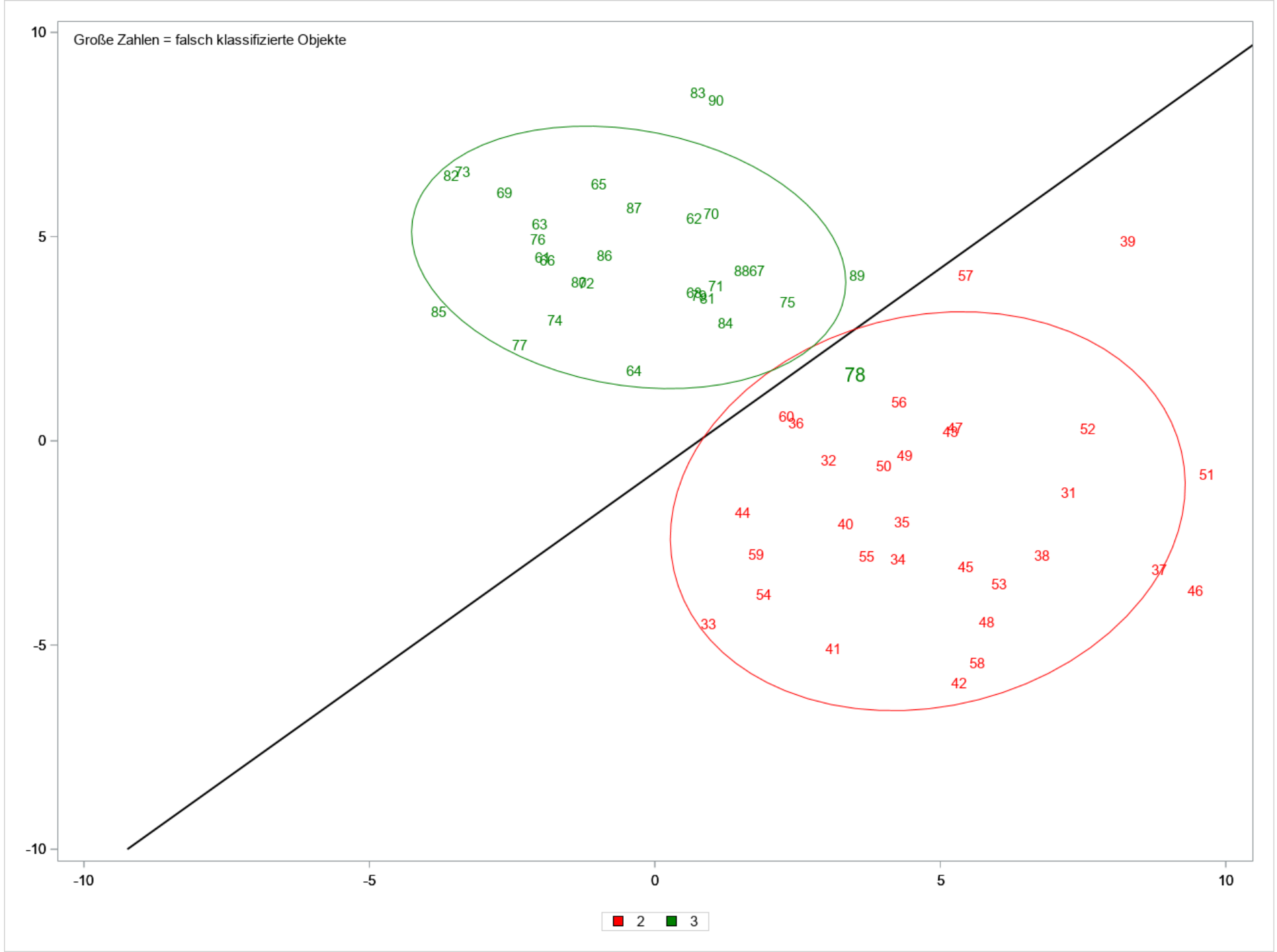
- **Das Programm - Trennlinien von 2 Gruppen errechnen**

VIEWTABLE: Work Threelinie23

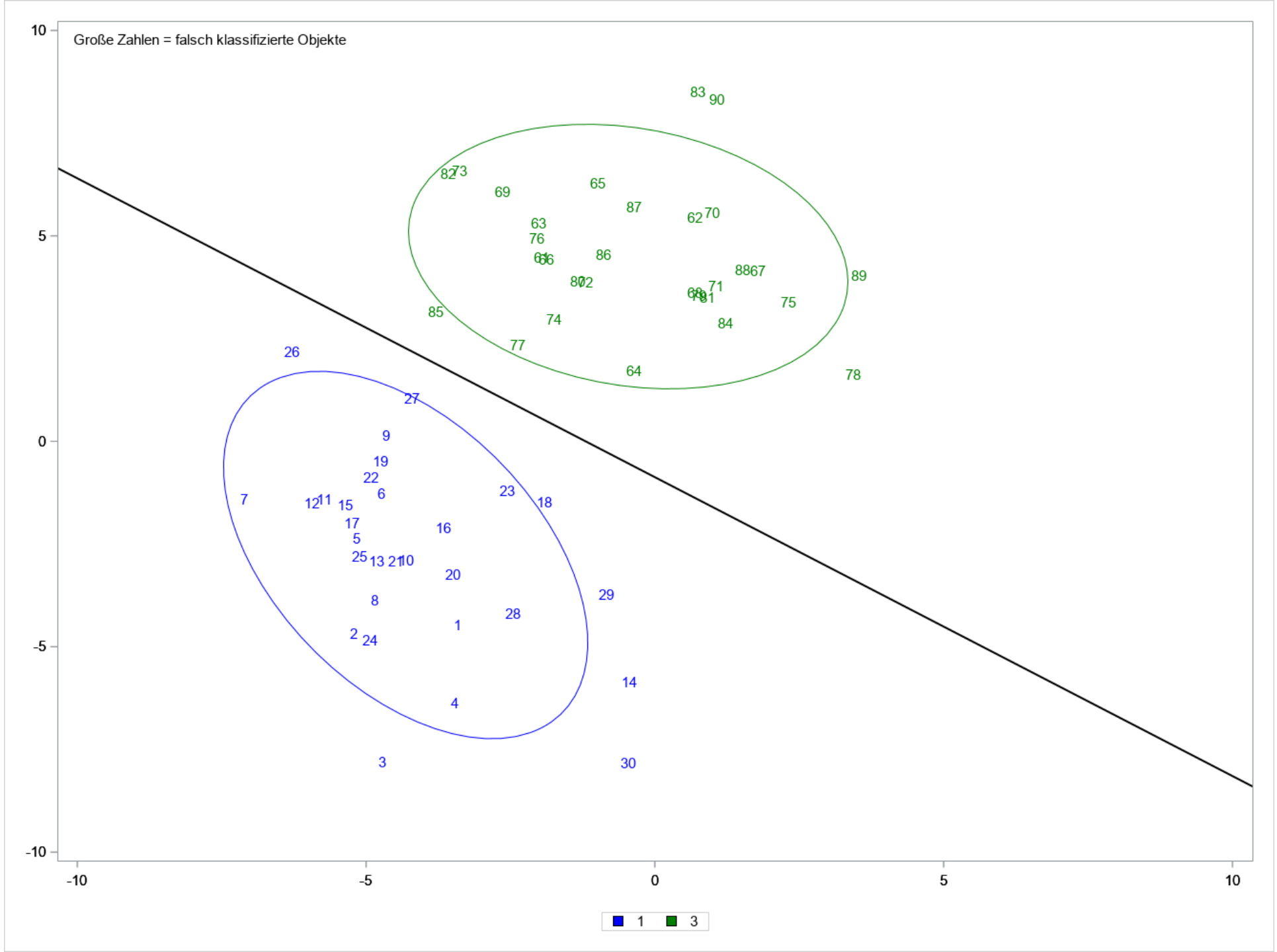
	nr	x	y	grp	Prin1	Prin2	y23	x23	size
1	31	6.7527496999	-4.987991659	2	7.1151955082	-1.286983837	-10	-9.239791902	1
2	32	3.7681999786	-1.912845859	2	2.9044746413	-0.49081268	10	10.78418126	1
3	33	-0.265173321	-3.978748157	2	0.7959305216	-4.502058643	.	.	1
4	34	3.3807556348	-4.59237034	2	4.1303076248	-2.904792394	.	.	1
5	35	3.9510457116	-3.894176196	2	4.1946504569	-2.005589506	.	.	1
6	36	3.8249845219	-0.846811716	2	2.3371784767	0.4135259399	.	.	1
47	77	0.9663896456	3.5095420427	3	-2.508158843	2.3297954926	.	.	1
48	78	5.3035824056	-0.415026687	3	3.2975772568	1.6178200055	.	.	2
49	79	4.229751448	2.6986270316	3	0.6269940633	3.5454967974	.	.	1
50	80	2.7007646088	4.1799735097	3	-1.476013969	3.8764731659	.	.	1
51	81	4.313663702	2.5564314025	3	0.7774686772	3.4775390614	.	.	1
52	82	2.3776356388	7.6048910814	3	-3.711980238	6.4908426026	.	.	1



www.julius-kuehn.de



www.julius-kuehn.de



www.julius-kuehn.de

- **Das Programm - die wichtigste Zeile → Schnittpunkt**

```
%LET zero = %sysevalf( 2 *
(   (&p21.-&p31.)*&c12. - (&p11.-&p21.)*&c23.   )   /
(   (&p11.-&p21.)*(&p22.-&p32.) - (&p21.-&p31.)*(&p12.-&p22.)   )
);
```

Erinnerung an Studium: alle Diskriminanzfunktionen schneiden sich im selben Punkt

- **Das Programm - verkürzte Trennlinien**

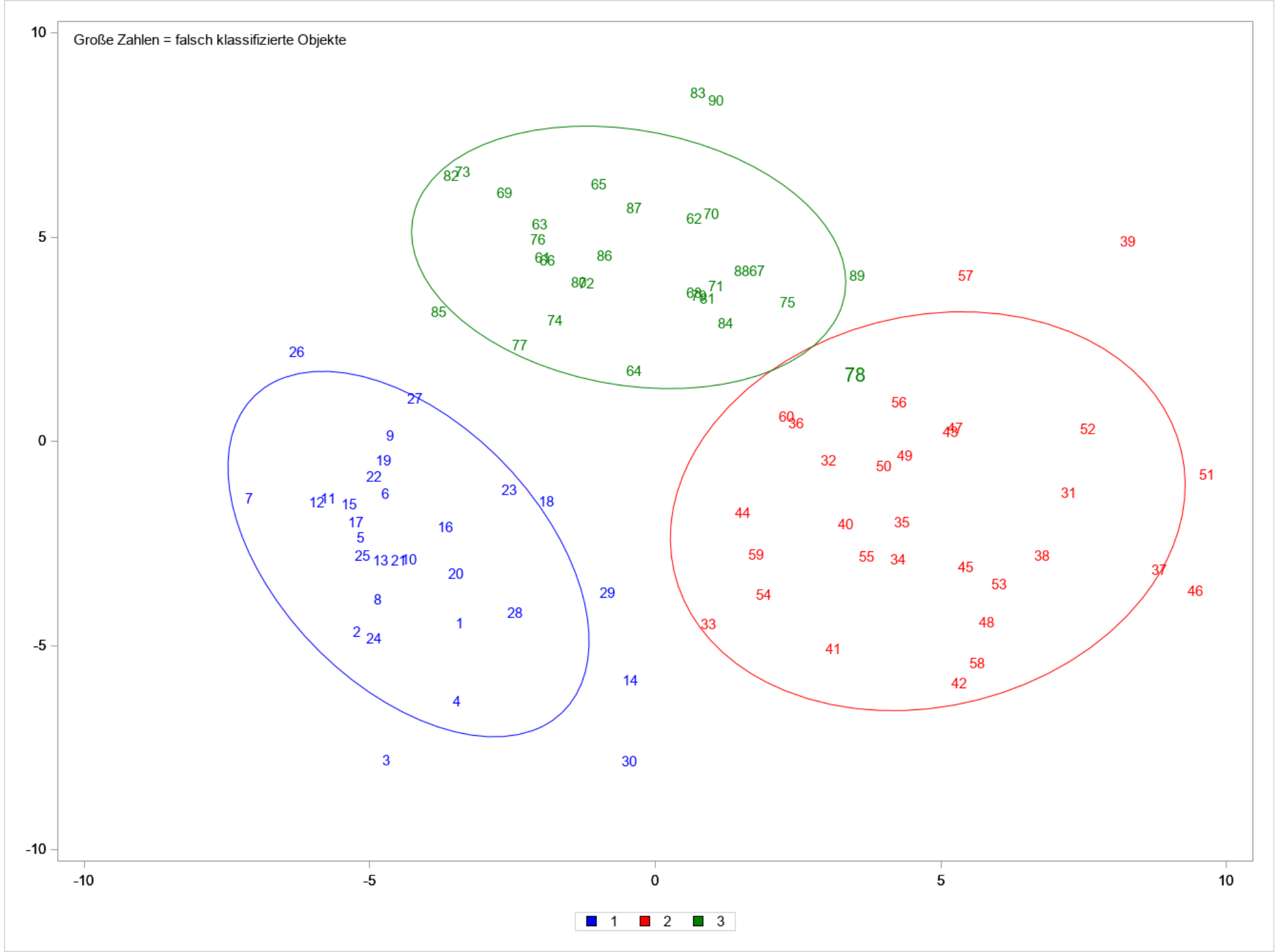
```
/* Erstellung der verkürzten Linien */

data trennlinie12;  /* green */
y12=&zero.; x12=-(2*&c12.+(&p12.-&p22.)*y12)/(&p11.-&p21.); output;
y12=&ymin.;  x12=-(2*&c12.+(&p12.-&p22.)*y12)/(&p11.-&p21.); output;
/* y12=&ymax.;  x12=-(2*&c12.+(&p12.-&p22.)*y12)/(&p11.-&p21.); output; */
run;

data trennlinie23;  /* blue */
y23=&zero.;  x23=-(2*&c23.+(&p22.-&p32.)*y23)/(&p21.-&p31.); output;
y23=&ymax.; x23=-(2*&c23.+(&p22.-&p32.)*y23)/(&p21.-&p31.); output;
/* y23=&ymin.; x23=-(2*&c23.+(&p22.-&p32.)*y23)/(&p21.-&p31.); output; */

run;

data trennlinie13;  /* red */
y13=&zero.; x13=-(2*&c13.+(&p12.-&p32.)*y13)/(&p11.-&p31.); output;
y13=&ymax.;  x13=-(2*&c13.+(&p12.-&p32.)*y13)/(&p11.-&p31.); output;
/* y13=&ymin.;  x13=-(2*&c13.+(&p12.-&p32.)*y13)/(&p11.-&p31.); output; */
run;
```

- **Das Programm - alles jetzt in einen Datensatz**

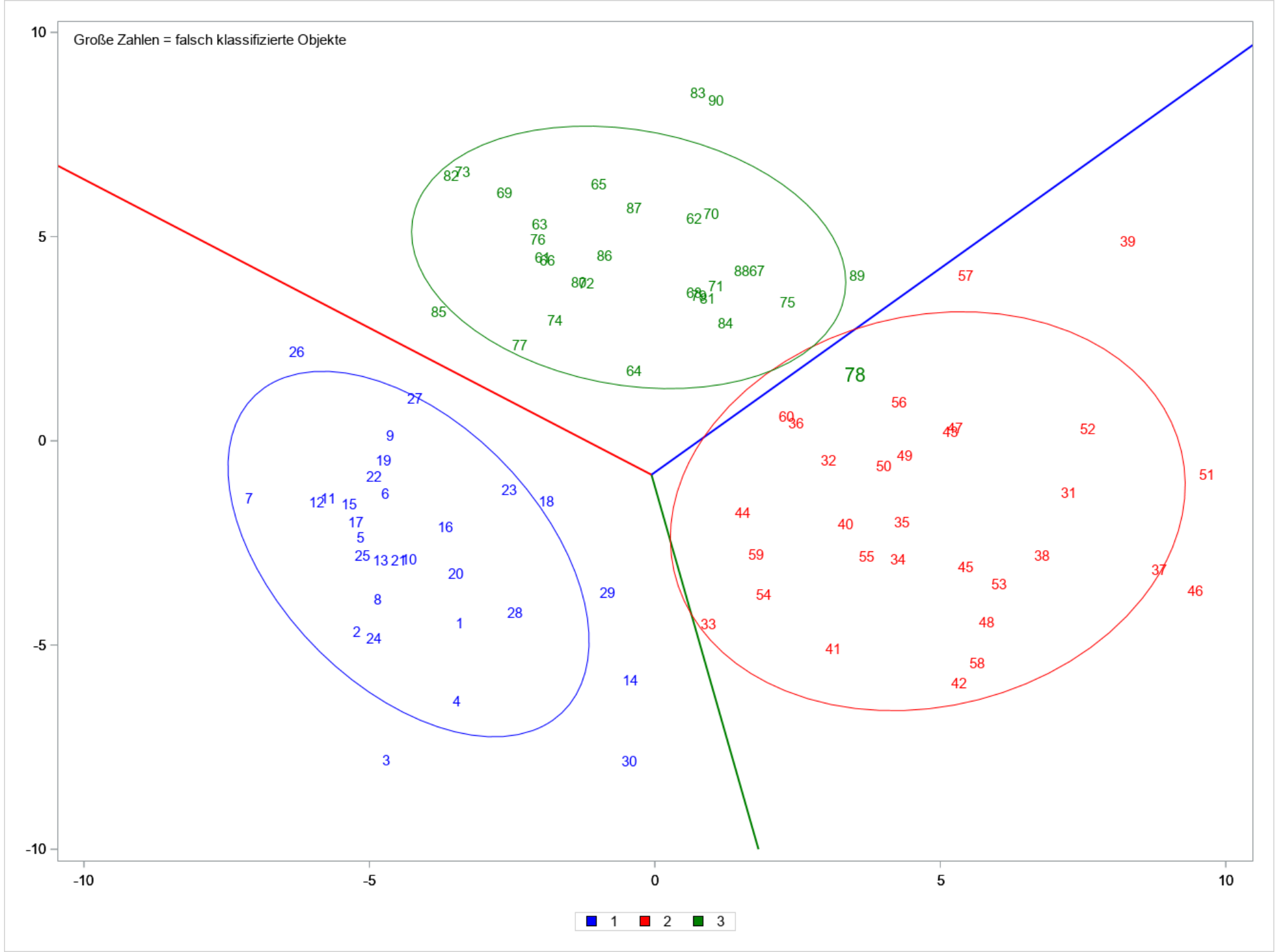
```
/* Gemeinsamer Datensatz */
data threelinie;
merge pcathree trennlinie12 trennlinie13 trennlinie23;
size=1;
if NR in (&wrong.) then size=2;
run;

/* Prüfen, ob Fehlklassifikationen */
%sizemax(threelinie);
```

- **Das Programm - alles jetzt in einen Datensatz**

```
/* Anschauen mit Trennlinien */
proc sgplot data=threelinie;

styleattrs datacolors=(blue red green) datacontrastcolors=(blue red green);

inset "Große Zahlen = falsch klassifizierte Objekte";

text x=&HK1. y=&HK2. text=nr /
             group=grp sizeresponse=size sizemin=&sizemin. sizemax=&sizemax.;

series x=x12 y=y12 / lineattrs=(color=green thickness=2);
series x=x13 y=y13 / lineattrs=(color=red thickness=2);
series x=x23 y=y23 / lineattrs=(color=blue thickness=2);

ellipse x=&HK1. y=&HK2. / group=grp alpha=&alpha.;

xaxis display=(nolabel) values=(&xmin. to &xmax. by &step.);
yaxis display=(nolabel) values=(&ymin. to &ymax. by &step.);
label grp='00'x;

run;
```

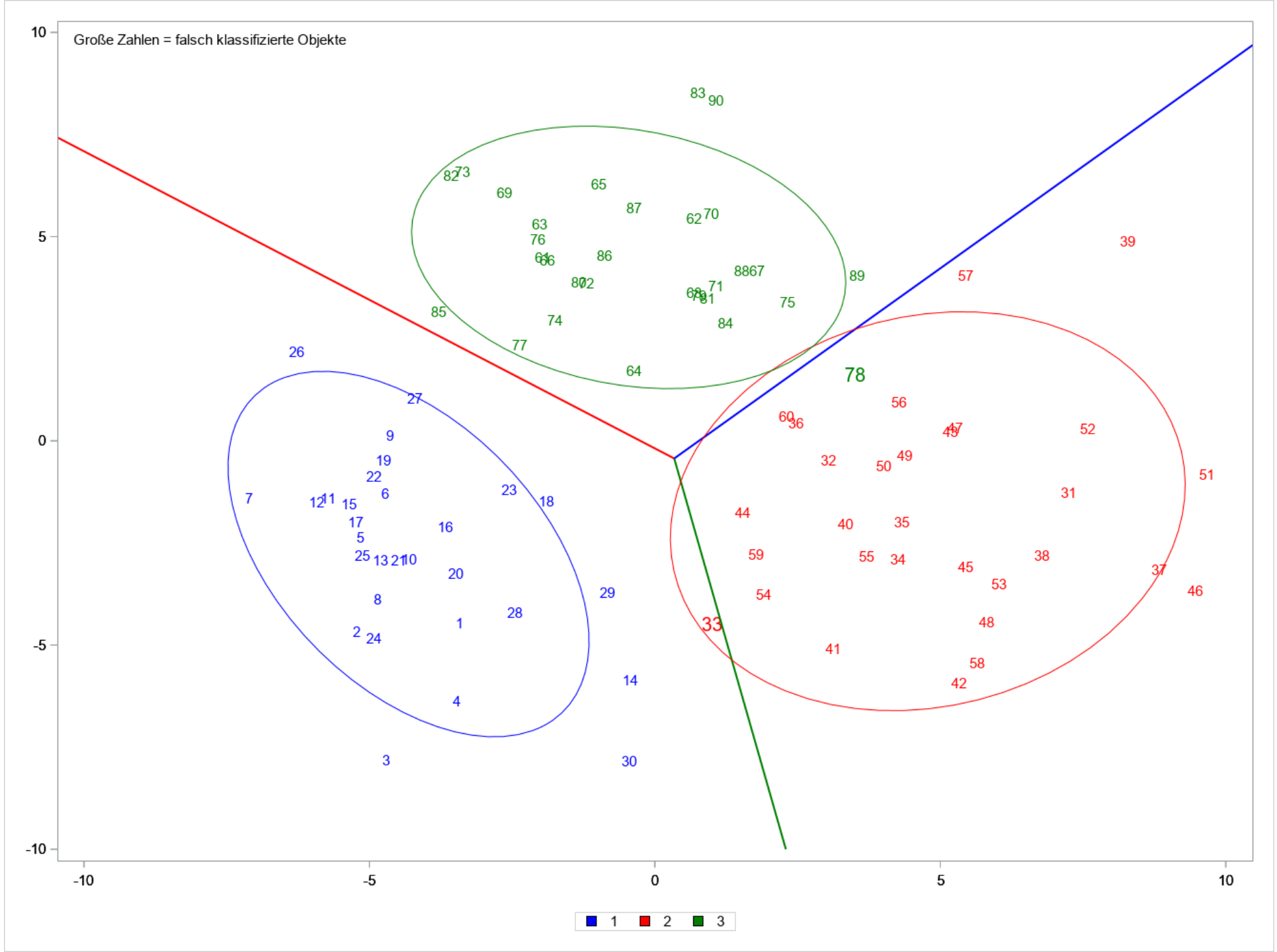


www.julius-kuehn.de



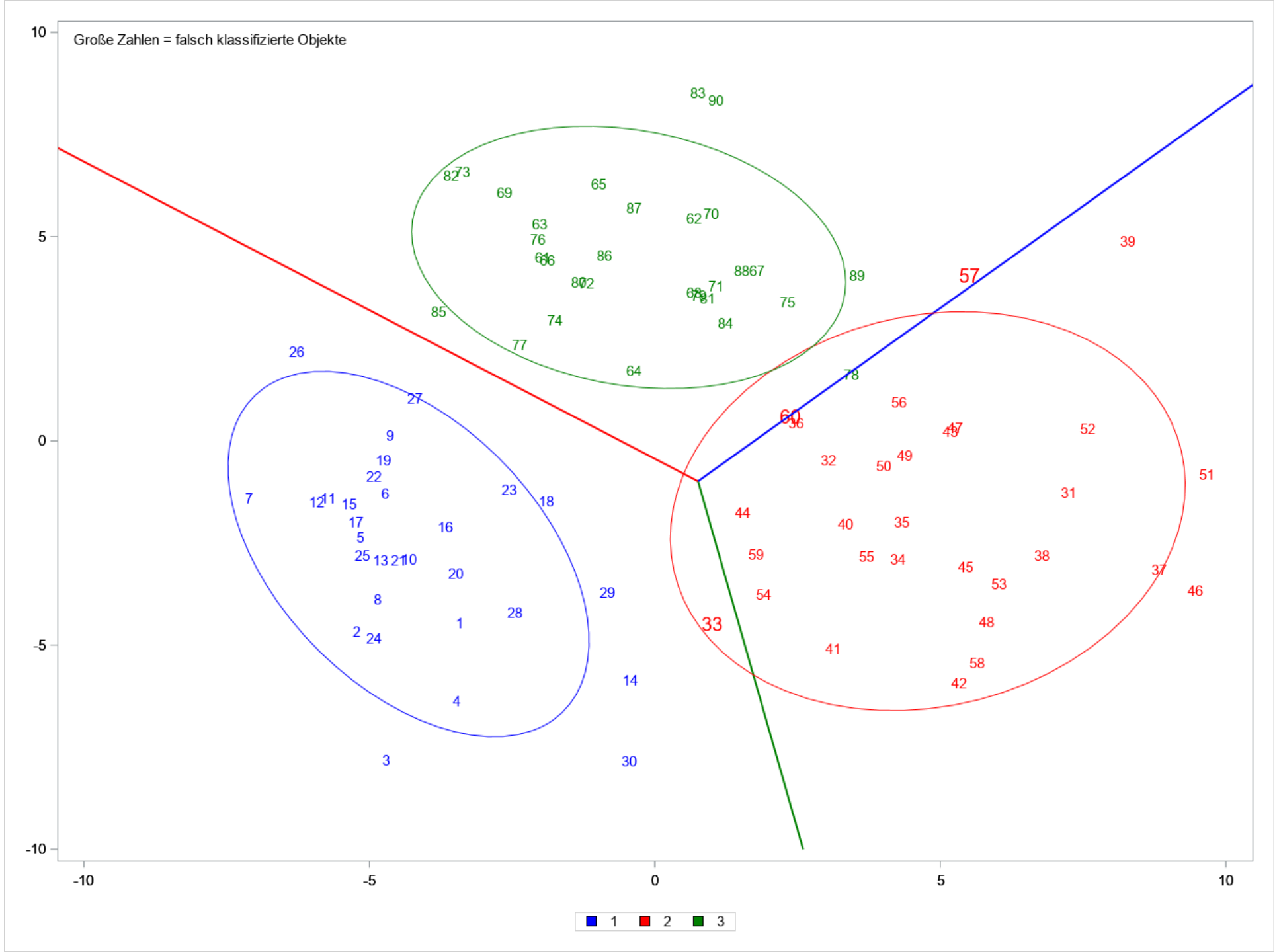
Blau = 0.33
Rot = 0.33
Grün = 0.33

www.julius-kuehn.de



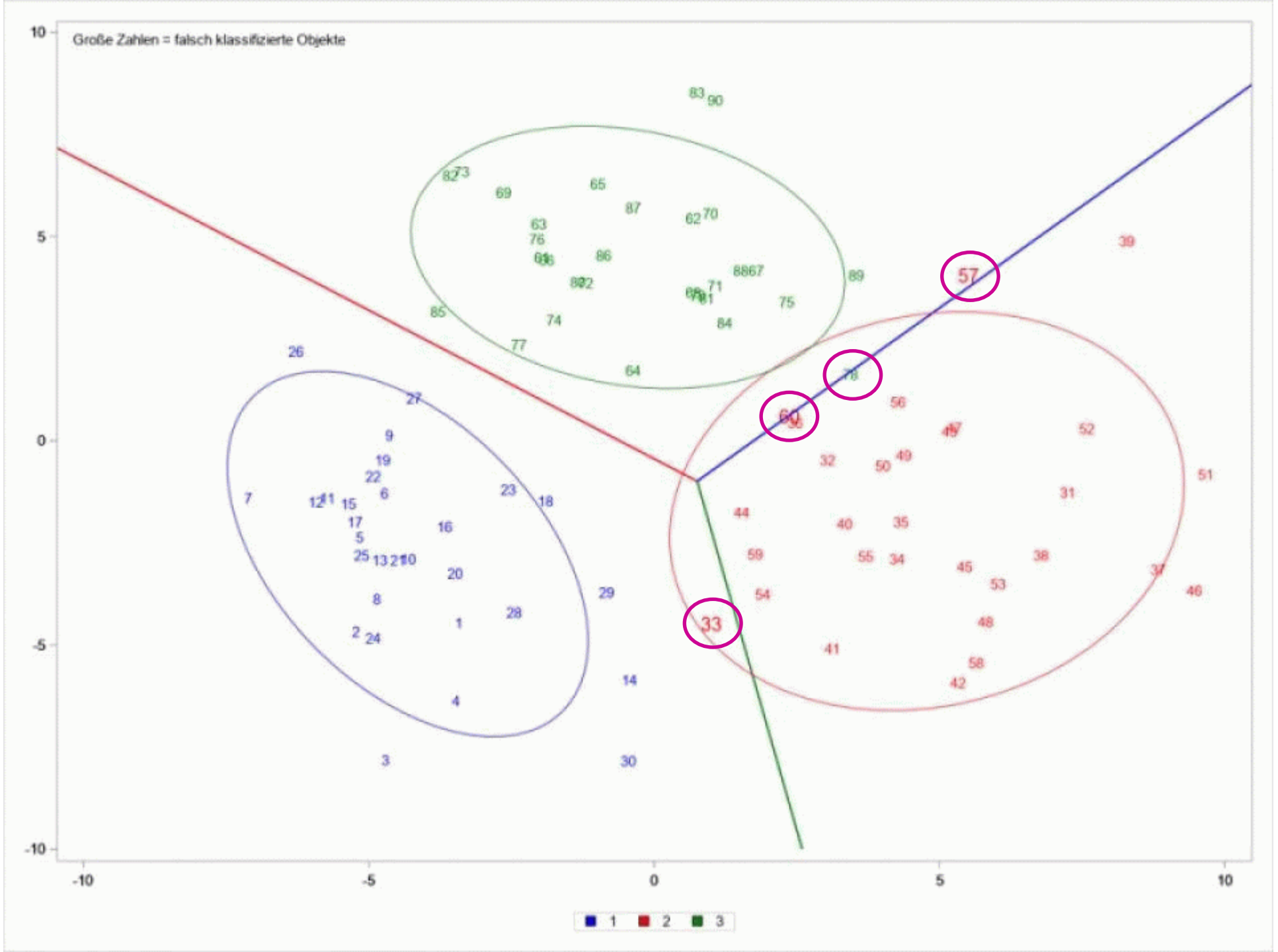
Blau = 0.60
Rot = 0.20
Grün = 0.20

www.julius-kuehn.de

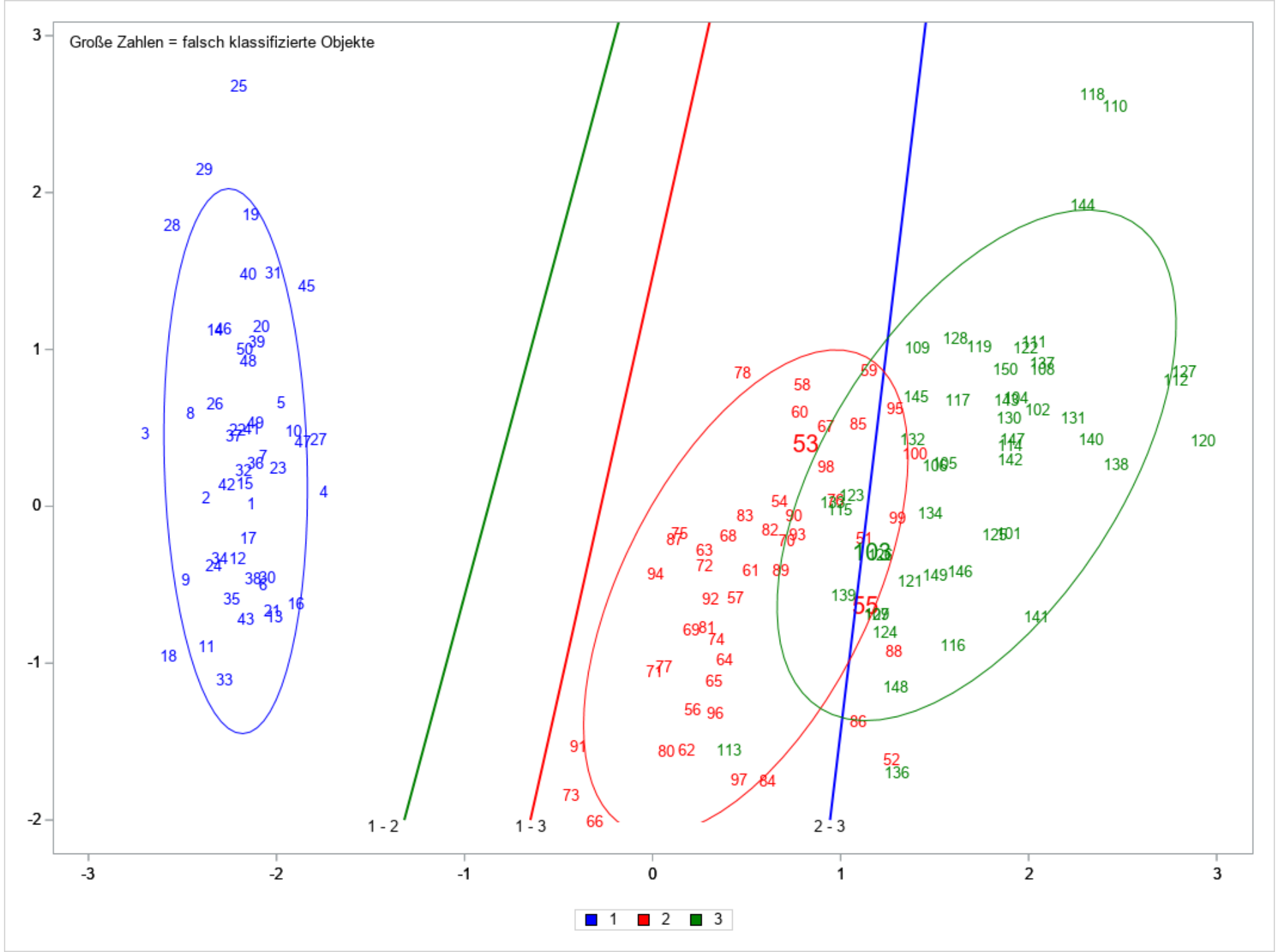


Blau = 0.60
Rot = 0.10
Grün = 0.30

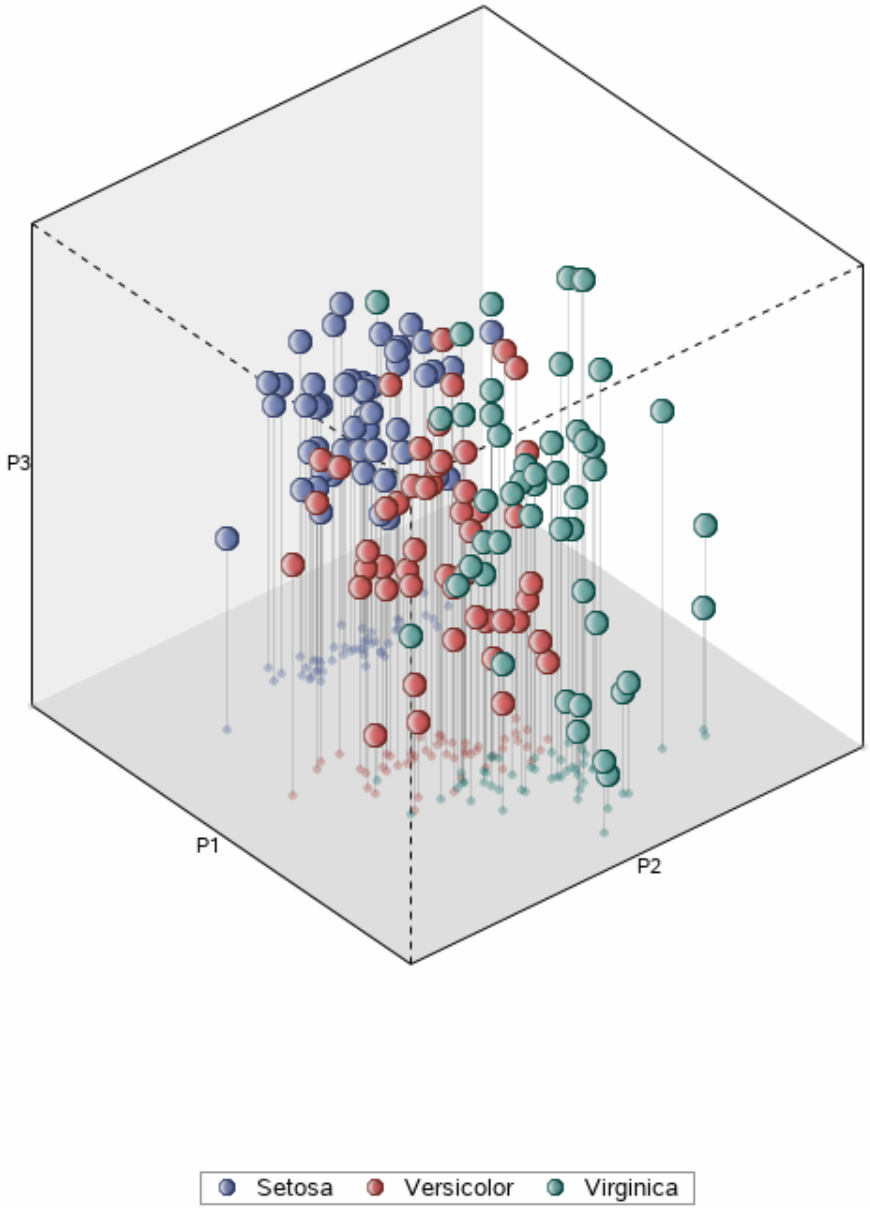
www.julius-kuehn.de



www.julius-kuehn.de



Es sieht nicht immer schön aus, hier am Beispiel der IRIS-Daten



Download Demo:

https://sf.julius-kuehn.de/sas