SAS数据清洗

由于数据收集、数据加载、数据处理等引发的数据质量问题，易形成脏数据。 经过数据探索分析，并结合数据场景，发现数据可能存在:

- 重复值;
- 缺失值;
- 异常值(离群点);
- 噪音数据;

(PS：业务上，账户信息表的CRED_LIMIT授信额度范围[0, 5,000,000])

(1)重复值

如账户信息表，XACCOUNT账户号、CRED_LIMIT授信额度；

XACCOUNT	CRED_LIMIT
0001487730	10,000
0001487731	50,000
0001487731	50,000
0001487732	30,000

(2)缺失值

如账户信息表，XACCOUNT账户号、CRED_LIMIT授信额度；

XACCOUNT	CRED_LIMIT
0001487730	10,000
0001487731	.
0001487732	50,000
0001487733	30,000

(3)异常值(离群点)

异常值指特殊的离群点，不一定错误。

如账户信息表，XACCOUNT账户号、CRED_LIMIT授信额度；

XACCOUNT	CRED_LIMIT
0001487730	10,000
0001487731	1,000,000
0001487732	50,000
0001487733	30,000

(4)噪音数据

噪音包括错误值或偏离期望的孤立点值。

如账户信息表，XACCOUNT账户号、CRED_LIMIT授信额度；

XACCOUNT	CRED_LIMIT
0001487730	10,000
0001487731	10,000,000
0001487732	50,000
0001487733	30,000

脏数据案例

```
OPTIONS COMPRESS = YES;

/* 数据案例 CAR */
DATA
    CARS1(KEEP=ID MAKE MODEL ORIGIN ETL_DT)
    CARS2(KEEP=ID MSRP1 MSRP2)
;
FORMAT ETL_DT DATE9. ID $8. MSRP1 MSRP2 DOLLAR10.;
SET SASHELP.CARS;
ID = COMPRESS("CAR9"||PUT(_N_,Z4.));
ETL_DT = "01APR2019"D;
MSRP1 = MSRP;
MSRP2 = INVOICE;
OUTPUT CARS1;
IF MSRP > 20000 THEN OUTPUT CARS2;
RUN;

PROC SORT DATA = CARS1;BY ID;RUN;
PROC SORT DATA = CARS2;BY ID;RUN;

DATA CARS3 CARS4;
MERGE CARS1(IN=A) CARS2(IN=B);
BY ID;
IF A;
IF _N_ IN (10,26,35,75,104,150) THEN ORIGIN = "";
IF _N_ IN (194) THEN DO;
MAKE = "#$Q@f#q^V";
MODEL = "$%g%^u@ed@#rf";
MSRP1 = .;
MSRP2 = .;
END;

IF _N_ IN (20, 21)
THEN DO;
```

```
    MSRP1 = -1;
END;

IF _N_ IN (40, 41, 42)
THEN DO;
    MSRP2 = -1;
END;

OUTPUT CARS3;
IF _N_ < 8 THEN OUTPUT CARS4;
RUN;


DATA CARS5;
SET CARS4;
IF _N_ > 4
THEN DO;
    ETL_DT = "19MAR2019"D;
    MSRP1 = MSRP1 - 432;
    MSRP2 = MSRP2 - 323;
END;
RUN;

DATA CAR;
SET CARS3 CARS5;
RUN;
/* 数据案例 CAR */

PROC DELETE DATA = CARS1 CARS2 CARS3 CARS4 CARS5;
RUN;

PROC SURVEYSELECT
    DATA = CAR METHOD = SRS N = 20
    OUT = CAR_DEMO;
RUN;

PROC PRINT DATA = CAR_DEMO;
RUN;
```

The SAS System

The SURVEYSELECT Procedure

Selection Method	Simple Random Sampling

Input Data Set	CAR
Random Number Seed	644640495
Sample Size	20
Selection Probability	0.045977
Sampling Weight	21.75
Output Data Set	CAR_DEMO

Obs	ETL_DT	ID	Make	Model	Origin	MSRP1	MSRP2
1	01APR2019	CAR90005	Acura	3.5 RL 4dr	Asia	$43,755	$39,014
2	01APR2019	CAR90026	Audi	S4 Avant Quattro		$49,090	$44,446
3	01APR2019	CAR90035	BMW	330xi 4dr		$37,245	$34,115
4	01APR2019	CAR90089	Chevrolet	SSR	USA	$41,995	$39,306
5	01APR2019	CAR90114	Dodge	Grand Caravan SXT	USA	$32,660	$29,812
6	01APR2019	CAR90233	Lincoln	LS V6 Premium 4dr	USA	$36,895	$33,929
7	01APR2019	CAR90245	Mazda	MPV ES	Asia	$28,750	$26,600
8	01APR2019	CAR90248	Mazda	RX-8 4dr automatic	Asia	$25,700	$23,794
9	01APR2019	CAR90253	Mercedes-Benz	ML500	Europe	$46,470	$43,268
10	01APR2019	CAR90285	Mercury	Monterey Luxury	USA	$33,995	$30,846
11	01APR2019	CAR90298	Mitsubishi	Lancer Evolution 4dr	Asia	$29,562	$27,466
12	01APR2019	CAR90299	Mitsubishi	Lancer Sportback LS	Asia	.	.
13	01APR2019	CAR90307	Nissan	Altima SE 4dr	Asia	$23,290	$21,580
14	01APR2019	CAR90333	Porsche	911 Carrera 4S coupe 2dr (convert)	Europe	$84,165	$72,206
15	01APR2019	CAR90368	Suzuki	Aeno S	Asia	.	.

				4dr			
16	01APR2019	CAR90370	Suzuki	Forenza S 4dr	Asia	.	.
17	01APR2019	CAR90375	Toyota	Sequoia SR5	Asia	$35,695	$31,827
18	01APR2019	CAR90379	Toyota	RAV4	Asia	$20,290	$18,553
19	01APR2019	CAR90409	Volkswagen	Passat GLS 4dr	Europe	$23,955	$21,898
20	01APR2019	CAR90422	Volvo	S80 2.9 4dr	Europe	$37,730	$35,542

```
PROC SORT DATA = CAR OUT = CAR_DUPK NODUPKEY;
BY ID;
RUN;

PROC SORT DATA = CAR OUT = CAR_UNIK NOUNIQUEKEY;
BY ID;
RUN;
```

```
423  ods listing close;ods html5 (id=saspy_internal) file=stdout
options(bitmap_mode='inline') device=svg style=HTMLBlue; ods
423! graphics on / outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: STDOUT
424
425  PROC SORT DATA = CAR OUT = CAR_DUPK NODUPKEY;
426  BY ID;
427  RUN;
NOTE: There were 435 observations read from the data set WORK.CAR.
NOTE: 7 observations with duplicate key values were deleted.
NOTE: The data set WORK.CAR_DUPK has 428 observations and 7 variables.
NOTE: Compressing data set WORK.CAR_DUPK increased size by 100.00 percent.
      Compressed is 2 pages; un-compressed would require 1 pages.
NOTE: PROCEDURE SORT used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

428
429  PROC SORT DATA = CAR OUT = CAR_UNIK NOUNIQUEKEY;
430  BY ID;
431  RUN;
NOTE: There were 435 observations read from the data set WORK.CAR.
NOTE: 421 observations with unique key values were deleted.
NOTE: The data set WORK.CAR_UNIK has 14 observations and 7 variables.
NOTE: Compressing data set WORK.CAR_UNIK increased size by 100.00 percent.
      Compressed is 2 pages; un-compressed would require 1 pages.
NOTE: PROCEDURE SORT used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

432
433  ods html5 (id=saspy_internal) close;ods listing;

434
```

```
/* DUPKEY */
PROC PRINT DATA = CAR_UNIK;
RUN;
```

The SAS System

Obs	ETL_DT	ID	Make	Model	Origin	MSRP1	MSRP2
1	01APR2019	CAR90001	Acura	MDX	Asia	$36,945	$33,337
2	01APR2019	CAR90001	Acura	MDX	Asia	$36,945	$33,337
3	01APR2019	CAR90002	Acura	RSX Type S 2dr	Asia	$23,820	$21,761
4	01APR2019	CAR90002	Acura	RSX Type S 2dr	Asia	$23,820	$21,761
5	01APR2019	CAR90003	Acura	TSX 4dr	Asia	$26,990	$24,647
6	01APR2019	CAR90003	Acura	TSX 4dr	Asia	$26,990	$24,647
7	01APR2019	CAR90004	Acura	TL 4dr	Asia	$33,195	$30,299
8	01APR2019	CAR90004	Acura	TL 4dr	Asia	$33,195	$30,299
9	01APR2019	CAR90005	Acura	3.5 RL 4dr	Asia	$43,755	$39,014
10	19MAR2019	CAR90005	Acura	3.5 RL 4dr	Asia	$43,323	$38,691
11	01APR2019	CAR90006	Acura	3.5 RL w/Navigation 4dr	Asia	$46,100	$41,100
12	19MAR2019	CAR90006	Acura	3.5 RL w/Navigation 4dr	Asia	$45,668	$40,777
13	01APR2019	CAR90007	Acura	NSX coupe 2dr manual S	Asia	$89,765	$79,978
14	19MAR2019	CAR90007	Acura	NSX coupe 2dr manual S	Asia	$89,333	$79,655

```
/* 处理重复值 */
PROC SORT DATA = CAR OUT = CAR_ETLDT;
BY ID DESCENDING ETL_DT;
RUN;

PROC SORT DATA = CAR_ETLDT OUT = CARD NODUPKEY;
BY ID;
RUN;
```

```
453  ods listing close;ods html5 (id=saspy_internal) file=stdout
options(bitmap_mode='inline') device=svg style=HTMLBlue; ods
453! graphics on / outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: STDOUT
454
455  /* 处理重复值 */
456  PROC SORT DATA = CAR OUT = CAR_ETLDT;
457  BY ID DESCENDING ETL_DT;
458  RUN;
NOTE: There were 435 observations read from the data set WORK.CAR.
NOTE: The data set WORK.CAR_ETLDT has 435 observations and 7 variables.
NOTE: Compressing data set WORK.CAR_ETLDT increased size by 100.00 percent.
      Compressed is 2 pages; un-compressed would require 1 pages.
NOTE: PROCEDURE SORT used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

459
460  PROC SORT DATA = CAR_ETLDT OUT = CARD NODUPKEY;
461  BY ID;
462  RUN;
NOTE: There were 435 observations read from the data set WORK.CAR_ETLDT.
NOTE: 7 observations with duplicate key values were deleted.
NOTE: The data set WORK.CARD has 428 observations and 7 variables.
NOTE: Compressing data set WORK.CARD increased size by 100.00 percent.
      Compressed is 2 pages; un-compressed would require 1 pages.
NOTE: PROCEDURE SORT used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

463
464  ods html5 (id=saspy_internal) close;ods listing;

465
```

```
/* BAD */
PROC FREQ DATA = CARD;
TABLES ETL_DT;
RUN;

PROC FREQ DATA = CARD(OBS=10);
TABLES MODEL;
RUN;
/* BAD */
```

The SAS System

The FREQ Procedure

ETL_DT	Frequency	Percent	Cumulative Frequency	Cumulative Percent
01APR2019	428	100.00	428	100.00

The SAS System

The FREQ Procedure

Model	Frequency	Percent	Cumulative Frequency	Cumulative Percent
3.5 RL 4dr	1	10.00	1	10.00
3.5 RL w/Navigation 4dr	1	10.00	2	20.00
A4 1.8T 4dr	1	10.00	3	30.00
A4 3.0 4dr	1	10.00	4	40.00
A41.8T convertible 2dr	1	10.00	5	50.00
MDX	1	10.00	6	60.00
NSX coupe 2dr manual S	1	10.00	7	70.00
RSX Type S 2dr	1	10.00	8	80.00
TL 4dr	1	10.00	9	90.00
TSX 4dr	1	10.00	10	100.00

```
/* MAKE */
PROC FREQ DATA = CARD NOPRINT;
TABLES MAKE/OUT = F_CAR_MAKE;
RUN;

PROC PRINT DATA = F_CAR_MAKE(OBS=10);
RUN;
```

The SAS System

Obs	Make	COUNT	PERCENT
1	#$Q@f#q^V	1	0.23364
2	Acura	7	1.63551
3	Audi	19	4.43925
4	BMW	20	4.67290
5	Buick	9	2.10280
6	Cadillac	8	1.86916
7	Chevrolet	27	6.30841
8	Chrysler	15	3.50467
9	Dodge	13	3.03738
10	Ford	23	5.37383

```
/* MODEL WHERE */
PROC FREQ DATA = CARD NOPRINT;
TABLES MODEL/OUT = F_CAR_MODEL_NOR;
RUN;

PROC FREQ DATA = CARD NOPRINT;
TABLES MODEL/OUT = F_CAR_MODEL_WHT;
WHERE SUBSTR(MODEL,1,1) ^= " ";
RUN;
```

```
545  ods listing close;ods html5 (id=saspy_internal) file=stdout
options(bitmap_mode='inline') device=svg style=HTMLBlue; ods
545! graphics on / outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: STDOUT
546
547  /* MODEL WHERE */
548  PROC FREQ DATA = CARD NOPRINT;
549  TABLES MODEL/OUT = F_CAR_MODEL_NOR;
550  RUN;
NOTE: There were 428 observations read from the data set WORK.CARD.
NOTE: The data set WORK.F_CAR_MODEL_NOR has 425 observations and 3
variables.
NOTE: Compressing data set WORK.F_CAR_MODEL_NOR increased size by 100.00
percent.
      Compressed is 2 pages; un-compressed would require 1 pages.
NOTE: PROCEDURE FREQ used (Total process time):
      real time           0.00 seconds
      cpu time            0.01 seconds

551
552  PROC FREQ DATA = CARD NOPRINT;
553  TABLES MODEL/OUT = F_CAR_MODEL_WHT;
554  WHERE SUBSTR(MODEL,1,1) ^= " ";
555  RUN;
NOTE: There were 1 observations read from the data set WORK.CARD.
      WHERE SUBSTR(MODEL, 1, 1) not = ' ';
NOTE: The data set WORK.F_CAR_MODEL_WHT has 1 observations and 3 variables.
NOTE: Compressing data set WORK.F_CAR_MODEL_WHT increased size by 100.00
percent.
      Compressed is 2 pages; un-compressed would require 1 pages.
NOTE: PROCEDURE FREQ used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

556
557  ods html5 (id=saspy_internal) close;ods listing;

558
```

```
/* ORIGIN MISSING */
PROC FREQ DATA = CARD NOPRINT;
TABLES ORIGIN/OUT=F_CAR_ORIGIN_M;
RUN;

PROC FREQ DATA = CARD NOPRINT;
TABLES ORIGIN/OUT=F_CAR_ORIGIN_NM MISSING;
RUN;

PROC UNIVARIATE DATA = CARD;
VAR MSRP1 MSRP2;
RUN;
```

The SAS System

The UNIVARIATE Procedure

Variable: MSRP1

Moments

N	329	**Sum Weights**	329
Mean	37287.2462	**Sum Observations**	12267504
Std Deviation	19476.3705	**Variance**	379329007
Skewness	3.01949083	**Kurtosis**	15.3801088
Uncorrected SS	5.81841E11	**Corrected SS**	1.2442E11
Coeff Variation	52.2333303	**Std Error Mean**	1073.76706

Basic Statistical Measures

Location		Variability	
Mean	37287.25	**Std Deviation**	19476
Median	32280.00	**Variance**	379329007
Mode	-1.00	**Range**	192466
		Interquartile Range	16475

Note: The mode displayed is the smallest of 15 modes with a count of 2.

Tests for Location: Mu0=0

Test	Statistic		p Value	
Student's t	**t**	34.72564	**Pr > |t|**	<.0001
Sign	**M**	162.5	**Pr >= |M|**	<.0001
Signed Rank	**S**	27139.5	**Pr >= |S|**	<.0001

Quantiles (Definition 5)	
Level	**Quantile**
100% Max	192465
99%	121770
95%	75000
90%	56665
75% Q3	41995
50% Median	32280
25% Q1	25520
10%	22000
5%	20585
1%	20140
0% Min	-1

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
-1	21	94820	262
-1	20	121770	271
20130	203	126670	272
20140	150	128420	263
20215	118	192465	335

Missing Values			
Missing Value	Count	Percent Of	
		All Obs	Missing Obs
.	99	23.13	100.00

The SAS System

The UNIVARIATE Procedure
Variable: MSRP2

Moments			
N	329	**Sum Weights**	329
Mean	33904.3951	**Sum Observations**	11154546
Std Deviation	17827.8412	**Variance**	317831921
Skewness	3.0062967	**Kurtosis**	15.0757898
Uncorrected SS	4.82437E11	**Corrected SS**	1.04249E11
Coeff Variation	52.5826846	**Std Error Mean**	982.8807

Basic Statistical Measures			
Location		Variability	
Mean	33904.40	**Std Deviation**	17828
Median	29405.00	**Variance**	317831921
Mode	-1.00	**Range**	173561
		Interquartile Range	15101

Tests for Location: Mu0=0				
Test	Statistic		p Value	
Student's t	**t**	34.49492	**Pr > |t|**	<.0001
Sign	**M**	161.5	**Pr >= |M|**	<.0001
Signed Rank	**S**	27136.5	**Pr >= |S|**	<.0001

Quantiles (Definition 5)

Level	Quantile
100% Max	173560
99%	113388
95%	69168
90%	51815
75% Q3	38376
50% Median	29405
25% Q1	23275
10%	20201
5%	19238
1%	18076
0% Min	-1

Extreme Observations

Lowest		Highest	
Value	**Obs**	**Value**	**Obs**
-1	42	88324	262
-1	41	113388	271
-1	40	117854	272
18076	118	119600	263
18380	176	173560	335

Missing Values

		Percent Of	
Missing Value	**Count**	**All Obs**	**Missing Obs**
.	99	23.13	100.00

```
/* 处理 */
DATA CAR_MSRP;
FORMAT IMSRP1 IMSRP2 $4.;
SET CARD;
IF MSRP1 = . THEN IMSRP1 = "MISS";
ELSE IF MSRP1 = -1 THEN IMSRP1 = "NEW";
ELSE IMSRP1 = "NORM";

IF MSRP2 = . THEN IMSRP2 = "MISS";
ELSE IF MSRP2 = -1 THEN IMSRP2= "NEW";
ELSE IMSRP2 = "NORM";
RUN;
/* 处理 */
```

```
603  ods listing close;ods html5 (id=saspy_internal) file=stdout
options(bitmap_mode='inline') device=svg style=HTMLBlue; ods
603! graphics on / outputfmt=png;
NOTE: Writing HTML5(SASPY_INTERNAL) Body file: STDOUT
604
605  /* 处理 */
606  DATA CAR_MSRP;
607  FORMAT IMSRP1 IMSRP2 $4.;
608  SET CARD;
609  IF MSRP1 = . THEN IMSRP1 = "MISS";
610  ELSE IF MSRP1 = -1 THEN IMSRP1 = "NEW";
611  ELSE IMSRP1 = "NORM";
612
613  IF MSRP2 = . THEN IMSRP2 = "MISS";
614  ELSE IF MSRP2 = -1 THEN IMSRP2= "NEW";
615  ELSE IMSRP2 = "NORM";
616  RUN;
NOTE: There were 428 observations read from the data set WORK.CARD.
NOTE: The data set WORK.CAR_MSRP has 428 observations and 9 variables.
NOTE: Compressing data set WORK.CAR_MSRP increased size by 100.00 percent.
      Compressed is 2 pages; un-compressed would require 1 pages.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

617  /* 处理 */
618
619  ods html5 (id=saspy_internal) close;ods listing;

620
```

```
/* 检查 */
PROC FREQ DATA = CAR_MSRP NOPRINT;
TABLES IMSRP1 * IMSRP2/MISSING OUT = CAR_IMSRP12;
RUN;
/* 检查 */

PROC PRINT DATA = CAR_IMSRP12(OBS=10);
RUN;
```

The SAS System

Obs	IMSRP1	IMSRP2	COUNT	PERCENT
1	MISS	MISS	99	23.1308
2	NEW	NORM	2	0.4673
3	NORM	NEW	3	0.7009
4	NORM	NORM	324	75.7009

补充：

SAS四则运算

(PS：此部分正是引起前绪介绍各种脏数据出现原因)

已知A=4，B=2，求

- A+B;
- A-B;
- A*B;
- A/B;

需注意：

- 缺失值
- 除数为0

```
DATA DEMO1;
A = 4;
B = 2;
/* 1 A + B*/
A1B = A + B;
A1B_SUM = SUM(A, B);

/* 2 A - B*/
A2B = A - B;
```

```
A2B_SUM = SUM(A, -B);

/* 3 A * B*/
A3B = A * B;

/* 4 A / B*/
A4B = A / B;
RUN;


DATA DEMO2;
A = 4;
B = .;
/* 1 A + B*/
A1B = A + B;
A1B_SUM = SUM(A, B);

/* 2 A - B*/
A2B = A - B;
A2B_SUM = SUM(A, -B);

/* 3 A * B*/
A3B = A * B;

/* 4 A / B*/
A4B = A / B;

RUN;



DATA DEMO3;
A = 4;
B = 0;

/* 1 A + B*/
A1B = A + B;
A1B_SUM = SUM(A, B);

/* 2 A - B*/
A2B = A - B;
A2B_SUM = SUM(A, -B);

/* 3 A * B*/
A3B = A * B;

/* 4 A / B*/
A4B = A / B;
RUN;


PROC PRINT DATA = DEMO1;
RUN;
PROC PRINT DATA = DEMO2;
RUN;
PROC PRINT DATA = DEMO3;
RUN;
```

The SAS System

Obs	A	B	A1B	A1B_SUM	A2B	A2B_SUM	A3B	A4B
1	4	2	6	6	2	2	8	2

The SAS System

Obs	A	B	A1B	A1B_SUM	A2B	A2B_SUM	A3B	A4B
1	4	.	.	4	.	4	.	.

The SAS System

Obs	A	B	A1B	A1B_SUM	A2B	A2B_SUM	A3B	A4B
1	4	0	4	4	4	4	0	.