Poglavje 1

Statistični paket SAS

1.1 Podatki za obdelavo

Podatke za obdelavo podatkov vnesemo ali uvozimo iz prej pripravljene datoteke ali zbirke, ki smo jih shranili v izbrano mapo na disku v niz podatkov v SAS-u.

1.1.1 Poimenovanje map, datotek, tabel in stolpcev

Pri poimenovanju map, datotek in nizov podatkov (tabel) v SAS-u izbiramo nazive, ki se začenjajo s črko angleške abecede ali podčrtajem (_). Vsi nadaljnji znaki so lahko črke angleške abecede, številke, podčrtaj, kakor tudi nekatere druge znake. Izogibamo se šumnikom, večini posebnih znakov, ki imajo specifične pomene. Čeprav v nekaterih operacijskih sistemih lahko uporabljamo tudi presledke, jih raje izpuščamo. Datoteke lahko uporabljamo na računalnikih z različnim operacijskem sistemom, ki je manj toleranten na posebne znake.

Nazivi map, datotek ali tabele so lahko sestavljene le iz ene besede. Če bi radi sestavili naziv datoteke iz dveh ali več besed, lahko besede ločimo z velikimi začetnicami (npr. NizPodatkov) ali podčrtaji (niz_podatkov). Dobro nam služijo kratka imena, ki pa nas opomnijo na vsebino. Tabelo, kamor smo shranili preizkus jagnjet za leto 2019, lahko poimenujemo npr. jagnjeta2019.

Pri poimenovanju stolpcev v podatkovnih zbirkah, kamor sodijo tudi datoteke v excel-u in tabele v SAS-u, izbiramo nazive po istem principu kot pri imenih datotek. Morda bodo nazivi krajši, saj jih pri obdelavi podatkov pogosto uporabljamo - tipkamo.

Ob uvozu podatkov lahko izberemo knjižnico, v katero bomo shranili podatke (koda 1.1). Knjižnico navedemo pred nazivom tabele in ju ločimo s piko (.).

Shranjevanje v knjižnico “WORK” je samo začasno in ob izhodu iz programa SAS se delovne datoteke s podatki izbrišejo. Če jih shranimo pod “SASUSER”, bodo shranjene trajno in jih lahko ob naslednji obdelavi na istem računalniku spet uporabimo brez ponovnega uvoza. Podatke najdemo, če v oknu Explorer kliknemo dvakrat na ikono Libraries, pa naprej na knjižnico Work ali Sasuser in potem na želeni niz podatkov. Če knjižnice ne navedemo, bo nova tabela v knjižnici “WORK”

1.1.2 Vnos podatkov

Podatke za obdelavo lahko v SAS vnesemo (koda 1.1). Tak način običajno izberemo, kadar bomo delali z manjšim nizom podatkov. Pri večjih nizih podatkov vnesemo podatke predhodno v datoteko ali podatkovno zbirko.

Vnos podatkov začnemo s stavkom data, v katerem navedemo knjižnico in naziv niza podatkov (tabele), v katerega bomo začasno ali trajno shranili podatke.

DATA - stavek, s katerim bomo izbrali naziv tabele v SASovi zbirki podatkov, v katero bomo uvozili podatke.

INPUT - stavek, v katerem poimenujemo spremenljivke oz. stolpce v tabeli, kamor bomo shranili podatke. Istočasno določimo tudi format zapisa.

- Za numerične ali znakovne spremenljivke navedemo številko stolpca, kjer se vrednost spremenljivke začne, in številko stolpca, kjer se vrednost spremenljivke konča. Vrednosti ločimo s vezajem (-), presledki ne bodo motili.

	pasma	spol	rejec	MasaZac	StarOdst	MasaOdst	datum
1	T	1	KM	4	90	38.7	12.02.2019
2	T	2	KM	3.5	85	35.3	25.03.2019
3	T	1	ZA	3	95	37.9	23.02.2019
4	T	2	ZA	3.9	85	28.3	03.03.2019
5	JS	1	FT	3.4	88	26.2	15.02.2019
6	JS	2	FT	3.2	91	27.3	31.01.2019
7	JS	1	JU	3	97	33.4	17.02.2019
8	JS	2	JU	2.6	100	29.7	17.02.2019
9	B	1	KA	3.3	88	31.7	03.03.2019
10	B	2	KA	2.9	93	28.3	12.02.2019
11	B	1	RT	3.6	75	25	31.01.2019
12	B	2	RT	3.1	93	30.2	12.04.2019

Slika 1.1: Tabela jagnjitve2019 v knjižnici WORK

- Znakovne spremenljivke nakažemo z znakom "$".
- Če so med spremenljivkami presledki, lahko navajanje formata izpustimo. V tem primeru tudi ni nujno, da so vrednosti spremenljivk poravnane v stolpcih.
- Kadar navajamo formate, lahko spremenimo vrstni red spremenljivk (npr. input stavek v komentarju), posamezne stolpce preberemo tudi večkrat.
- Če so v spremenljivkah poleg številk tudi drugi znaki, bomo spremenljivko brali kot znakovno spremenljivko.
- Kot znakovno spremenljivko lahko preberemo tudi numerično spremenljivko, a potem se ne bo obnašala kot številka. Znakovne spremenljivke lahko uporabimo kot kvalitativne spremenljivke in ne kot kvalitativne spremenljivke, zato z njimi ne moremo računati.

CARDS - stavek, ki napove, da bodo sledili zapisi s podatki. Podatki naj bodo zapisani tako, kot smo določili s formatom.

Algoritem 1.1 Sintaksa pri uvozu podatkov s končnico .dat

```
data work.jagnjeta2019;
   input pasma $ 1-2 spol 4 rejec $ 6-7  MasaZac StarOdst MasaOdst datum $ 22-32;
   /* input  spol 4 rejec $ 6-7  MasaZac StarOdst MasaOdst pasma $ 1-2;  */
   cards;     /* sledijo zapisi */
T  1 KM 4.0  90 38.7 12.02.2019
T  2 KM 3.5  85 35.3 25.03.2019
T  1 ZA 3.0  95 37.9 23.02.2019
T  2 ZA 3.9  85 28.3 03.03.2019
JS 1 FT 3.4  88 26.2 15.02.2019
JS 2 FT 3.2  91 27.3 31.01.2019
JS 1 JU 3.0  97 33.4 17.02.2019
JS 2 JU 2.6 100 29.7 17.02.2019
B  1 KA 3.3  88 31.7 03.03.2019
B  2 KA 2.9  93 28.3 12.02.2019
B  1 RT 3.6  75 25.0 31.01.2019
B  2 RT 3.1  93 30.2 12.04.2019
;
run;
```

Po uspešnem uvozu bomo v knjižnici WORK dobili tabelo (slika 1.1) z vnesenimi podatki. Ker se vse informacije v knjižnici WORK po zaključku dela v programu SAS izbrišejo, je primerno, da kodo 1.1 shranimo.

1.1.3 Uvoz podatkov

Kodo za uvoz podatkov s končnico .dat, .lst ali .prn, v deklarativnem delu napišemo sami (koda 1.2).

Algoritem 1.2 Sintaksa pri uvozu podatkov s končnico .dat

```
data NizPodatkov;
  infile "N:\PEDAGOGI\MKovac\_podatki\podatki.dat" lrecl=n missover;
  input znakovna_sprem $ 1-9 numericna_sprem 12-15;
 /* Znakovne spremenljivke nakažemo z znakom $, ostale so numerične spremenljivke */
run;
```

INFILE - mesto (N:\PEDAGOGI\MKovac_podatki\) in ime datoteke (podatki.dat), kjer se nahajajo podatki, ki jih želimo uvoziti

LRECL = n - logična dolžina vrstice (število znakov s presledki v eni vrstici - zapisu)

MISSOVER - v primeru krajše vrstice preprečimo branje v novi vrsti

INPUT - poimenujemo spremenljivke (podatke v stolpcih) in definiramo tip spremenljivk (numerični, znakovni, datum) ter položaj spremenljivk. Imen spremenljivk v istem stavku INPUT ne ponavljamo. Privzeti tip spremenljivk je numeričen (cela ali realna števila). Spremenljivke znakovnega tipa moramo označiti z znakom "$". Pri spremenljivkah znakovnega tipa moramo biti previdni, saj bo SAS prebral samo prvih osem znakov. Če za imenom spremenljivke v stavku INPUT (koda 1.3) navedemo prvi in zadnji stolpec polja, s tem natančno določimo mesto podatka. Tak način deklariranja ima več prednosti:

- znakovne spremenljivke so lahko dolge do 200 znakov ter lahko vsebujejo tudi presledke,
- polja lahko beremo v kateremkoli vrstnem redu,
- manjkajočih podatkov nam ni potrebno posebej označiti,
- del podatkov lahko pri branju izpustimo (preskočimo) ali polje podatka ali dele polja lahko večkrat beremo.

Algoritem 1.3 Primer uvoza podatkov s končnico .dat

```
%let PATH = N:\PEDAGOGI\MKovac\_podatki\
/* tako navedemo mesto, kjer najdemo datoteke na disku */
data mladice;
   infile "&PATH.mladice.dat" lrecl=50 missover; /* kako uporabimo PATH */
   input id $ 1-9 genotip $ 12-15 spol $ 17 dm 21-28 mtp 35-39 s 49-50;
run;

data mladice1;
   infile "&PATH.mladice.dat" lrecl=50 missover;
   input id $ 1-9 genotip $ spol $ dm  mtp s;
      /* Če podatke beremo po vrsti, ni potrebno navesti mesta spremenljivke
        v podatkih, izjema so spremenljivke znakovnega tipa, daljše od 9 mest*/
run;
```

V SAS lahko uvozimo podatke v formatih MS Excel (*.xls), dBASE (*.dbf), Lotus (*.wk1, *.wk3, *.wk4) in

datoteke, kjer so podatki ločeni z tabulatorji (Tab Delimited File, *.txt), ali z vejico (Comma Separated Values, *.csv) v SAS lahko uvozimo. V menijski vrstici pod "File" izberemo opcijo "Import". Odpre se nam pogovorno okno, kjer najprej izberemo format datoteke. Z miško kliknemo gumb "Next>", nakar nas SAS vpraša po lokaciji naše datoteke s podatki. Ko to izberemo, se premaknemo naprej ("Next>") in pod "Member" vpišemo ime, ki smo ga izbrali za niz podatkov v programu SAS. Pri imenovanju niza podatkov se držimo pravila, da se ime niza podatkov (kasneje pa tudi stolpcev) lahko začne s _ (podčrtajem) ali črko angleške abecede. Vsi nadaljnji znaki so lahko črke angleške abecede, številke ali _. V imenih niza podatkov (ali stolpcev) ne smemo uporabljati presledkov. Hkrati lahko izberemo, v katero knjižnico naj shrani podatke. Shranjevanje v knjižnico "WORK" je samo začasno in ob izhodu iz programa SAS se delovne datoteke s podatki izbrišejo. Če jih shranimo pod "SASUSER", bodo shranjene trajno in jih lahko ob naslednji obdelavi na istem računalniku spet uporabimo brez ponovnega uvoza. V naslednjem koraku nas vpraša za mesto in ime datoteke, kamor želimo shraniti kodo za uvoz podatkov (to za uspešen uvoz podatkov ni obvezno). Uvoz končamo s klikom na gumb "Finish". V oknu Log nam program napiše, ali smo bili pri uvozu podatkov uspešni. SAS pri uvažanju sam sestavi in izvede proceduro IMPORT (koda 1.4).

Algoritem 1.4 Uvoz podatkov iz datoteke v MS Excel formatu

```
proc import out= work.NizPodatkov
     datafile = "&PATH.podatki.xls" /* datoteka, ki jo bomo prebrali */
     dbms     = xls replace;
     sheet    = "ListName";         /* naziv lista v excelovi datoteki */
     getnames = yes;    /* naj SAS privzame imena spremenljivk (stolpcev) iz excelove datoteke */
run;
```

Algoritem 1.5 Primer uvoza podatkov iz datoteke v MS Excel formatu

```
proc import out= work.mladice
    datafile = "&PATH.mladice.xls" /* uporabimo PATH, ki smo ga na začetku določili */
    dbms     = EXCEL2000 replace;
    sheet    = "List1";
    getnames = yes;
run;
```

1.2 Ustvarjanje podnizov

V deklarativnem delu lahko, poleg uvoza podatkov, posamezne nize tudi spreminjamo (koda 1.6). Lahko ustvarjamo nove podnize ali pa dodajamo posamezne stolpce ... Iz obstoječega niza podatkov (NizPodatkov), ki ga navedemo v stavku SET, bomo naredili novega (NovNizPodatkov), ki ga poimenujemo v stavku DATA.

DATA - nov niz podatkov (nov in obstoječi niz sta lahko tudi enaka)

SET - obstoječi niz podatkov

Pri kreiranju novega niza podatkov lahko uporabimo stavke naslednje pogojne stavke, s katerimi lahko izberemo ali izbrišemo zapise (vrstice), če le te izpolnjujejo navedeni pogoj.

WHERE ... - kjer je izpolnjen navedeni pogoj

IF ... - če je izpolnjen naveden pogoj

IF ... **THEN** .... - če je izpolnjen navedeni pogoj, potem bo izvedel ukaz v nadaljevanju

Iz že obstoječega niza podatkov smo izbrali samo del podatkov s pomočjo pogojev (koda 1.6). Ustvarili smo nov niz podatkov, kjer so le ženske živali (WHERE SPOL=1) genotipa 22 (IF GENOTIP=22). V novem nizu podatkov smo izbrisali podatke, kjer je bil rejec 3000 (IF REJEC=3000 THEN DELETE). Zaporedne prasitve, ki so višje od 7, smo združili v 7 (IF PARITY>7 THEN PARITY=7), ki pa jo bomo navajali praviloma kot razred "7 in več", "7+" ali podobno. Vseh stavkov v sintaksi procedure ne potrebujemo vedno in nekatere lahko izpustimo.

V WHERE in IF stavkih lahko sestavljamo pogoje z uporabo

- aritmetičnih operacij, kot so seštevanje (+), odštevanje (-), množenje(*), deljenje (/) in potenciranje (**)
- primerjav, kot so "je enako" (= ali EQ), "ni enako" (^=, ~=, ¬= NE), večji kot (> ali GT), manjši kot (< ali LT), večji kot (>= ali GE), manjši kot (<= ali LE) in vsebovano v (IN)
- logičnih operacij, kot so in (& ali AND), ali (| ali OR) in zanikanje (~, ^, ¬ ali NOT) in
- drugih operacij, kot so || za sestavljanje znakovnih spremenljivk, () za spremembo vrstnega reda operacij, + pred vrednostjo za pozitivno vrednosti in - za negativno vrednost.

Algoritem 1.6 Ustvarjanje podnizov

```
data NovNizPodatkov; set NizPodatkov;
   where spol=1;               /* vzeli bomo podatke, kjer je spol 1 */
   if genotip=22;              /* vzemimo podatke, če je genotip 22  */
   if rejec=3000 then delete;  /* rejca z oznako 3000 bomo izbrisali */
   if parity>7 then parity=7;  /* zadnja zaporedna prasitev bo 7+    */
run;
```

Algoritem 1.7 Primer dodajanja novih stolpcev

```
data mladice; set mladice;
   rojeni   = zivorojeni + mrtvorojeni;
   star_kor = star-330;
   leto     = year(dtRoj)
   mesec    = month(dtRoj);
   sezona   = 100*leto+mesec:
   cetrt    = qrt(dtRoj);
   danVTednu= weekday(dtRoj);
   zap      = put(ZapLakt,1.);
   gnezdo   = strip(Rmati)||strip(zap);
run;
```

Algoritem 1.8 Procedura MEANS

```
proc means data = NizPodatkov;
   var kvantitativna_sprem1 kvantitativna_sprem2 kvantitativna_sprem3;
run;

proc means data = NizPodatkov maxdec=2 N mean std min max var cv stderr mode median;
   var kvantitativna_sprem1 kvantitativna_sprem2 kvantitativna_sprem3;
run;
```

Uporablja se lahko tudi izraz BETWEEN-AND, IS NULL ali IS MISSING za manjkajoče vrednosti, LIKE za isto kot, =* za podobno kot. Obstajajo še nekatere druge opcije, ki jih poiščemo v priročniku, ko jih potrebujemo. Kadar je več možnosti, običajno si izberemo tisto, ki nam je bolj všečna, ostale pa morda potrebujemo pri branju kod drugih avtorjev. Če nekdo uporablja več različnih programskih jezikov, se lahko zapis teh operacij razlikuje, kar nam lahko povzroča težave.

Stavek WHERE lahko uporabljamo samo pri ustvarjanju novih nizov podatkov iz že obstoječih nizov podatkov v podatkovni zbirki SAS, medtem ko lahko IF stavke uporabljamo tudi pri uvozu podatkov. Stavek WHERE je praviloma bolj učinkoviti, saj izbira podatke, predno jih shrani v vektorje, IF stavek pa se izvede na že prebranih podatkih. WHERE stavek ne moremo vključiti v IF stavek.

1.3 Ustvarjanje novih spremenljivk

V obstoječi niz podatkov mladice (koda 1.7) smo dodali nove stolpce. Dodajamo pa lahko tudi nove spremenljivke, ki jih z matematičnimi funkcijami izpeljemo iz obstoječih. SAS prepozna vse običajne matematične operatorje: +, -. *, / ter ** za eksponent. Možnosti za določanje novih spremenljivk so številne, tako si z nekaj brskanja po priročniku ali spletu in iznajdljivosti lahko določimo številne spremenljivke.

- Rojene pujske smo izračunali iz živorojenih in mrtvorojenih pujskov.
- Starost pa smo korigirali na izbrano konstanto.
- Iz datuma rojstva smo izluščili leto (funkcija YEAR) in mesec (funkcija MONTH). Iz njiju smo sestavili sezono kot v formatu YYYYMM. Vse vrednosti so cela števila (INTEGER). Dobimo lahko tudi četrtletje (funkcija QRT), dan v tednu (funkcija WEEKDAY) itd.
- Zaporedno laktacijo smo uvozili kot celo število, številka matere pa je v tem primeru tekst, zato bomo tudi zaporedno laktacijo zapisali kot tekst (char). To naredimo s funkcijo PUT. Spodnji zapis pove, da bomo porabili samo eno mesto. Številčni zapisi so v datotekah poravnani desno, znakovne spremenljivke pa leto.
- V zadnji vrstici smo oznako matere in novo znakovno kodo sestavili z operatorjem || za znakovne operacije. Da bi ne imeli presledkov pri oznakah smo uporabili funkcijo STRIP, s katero odstranimo presledke spredaj in zadaj.

1.4 Procedura MEANS

S proceduro MEANS izračunamo opisnih statistike za kvantitativne spremenljivke, kot so povprečje, varianca, standardni odklon in pogledamo minimum ter maksimum (koda 1.8). Je procedura, s katero lahko pregledujemo podatke.

Algoritem 1.9 Stavka CLASS in WHERE v proceduri MEANS

```
proc means data = NizPodatkov;
   var kvantitativna_sprem1 kvantitativna_sprem2 kvantitativna_sprem3;
   class kvalitativna_sprem;
run;

proc means data = NizPodatkov;
   var kvantitativna_sprem1 kvantitativna_sprem2 kvantitativna_sprem3;
   where spol='2' and genotip='1255';
run;
```

Algoritem 1.10 Stavek OUTPUT

```
proc means data = mladice n mean std;
   var rojeni;
   output out=mladice_out n=no mean=avg std=std;
run;

proc means data = mladice n mean std;
   var rojeni;
   output out=mladice_out n= mean= std= / autoname autolabel;
run;
```

PROC MEANS - procedura za izračun opisnih statistik

DATA - navedemo ime niza podatkov

MAXDEC =n - maksimalno število decimalnih mest pri izpisu opisnih statistik

N MEAN STD MIN MAX VAR CV STDERR MODE MEDIAN - izpis posameznih opisnih statistik (število meritev, povprečje, standardni odklon, minimum, maksimum, varianca, koeficient variabilnosti, standardna napaka srednje vrednosti, modus, mediana)

VAR - naštejemo kvantitativne spremenljivke

CLASS - opisne statistike računamo po razredih oz. nivojih za posamezen vpliv, npr. ločeno za spol 1 in spol 2. Namesto CLASS lahko uporabimo tudi stavek BY.

WHERE - opisne statistike računamo samo za določen del podatkov, npr. le za spol 2 in genotip 1255 (koda 1.9)

S proceduro MEANS si lahko v stavku OUTPUT (koda 1.10) pripravimo osnovne statistične parametre v obliki niza podatkov, ki pa ga lahko kasneje izvozimo v obliki primerni npr. za MS Excel.

OUT – nov niz podatkov, kamor shranimo rezultate

n=no mean=avg std=std - opisne statistike, ki jih izračunamo (na desni strani enačaja je ime stolpca v novem nizu podatkov)

n = mean = std= / autoname autolabel - SAS naj stolpce poimenuje sam

Vseh stavkov v sintaksi procedure ne potrebujemo vedno in nekatere lahko izpustimo.

1.5 Procedura FREQ

Procedura FREQ je uporabna pri pregledu strukture podatkov, ko nas zanima klasifikacija vplivov, zastopanost podatkov v razredih (nivojih) vplivov oz. v katerem od razredov vpliva nimamo podatkov. Stavku PROC FREQ (koda 1.11) sledi stavek TABLE ali TABLES, kjer navedemo, za katero spremenljivko nas zanima zastopanost podatkov. Opcije NOROW, NOCOL in NOPERCENT povzročijo, da v izpisu ni relativnih frekvenc po vrsticah in stolpcih ter za celotno tabelo.

1.6 Procedura UNIVARIATE

S proceduro UNIVARIATE lahko:

- izračunamo opisne statistike

Algoritem 1.11 Sintaksa v proceduri FREQ

```
proc freq data = mladice;
   table sezona;
   table genotip;
   tables sezona*genotip;
run;

proc freq data = mladice;
   tables sezona genotip;
          sezona*genotip /norow nocol nopercent;
run;
```

Algoritem 1.12 Splošna sintaksa v proceduri UNIVARIATE

```
proc univariate data = NizPodatkov normal plot;
  var kvantitatitivna_sprem;
  histogram kvantitatitivna_sprem1/normal;
run;
```

- izračunamo mediano, modus, razmik, kvantile ...
- narišemo porazdelitve
- poiščemo ekstremna opazovanja in ekstremne vrednosti
- testiramo lokacijske parametre
- testiramo normalnost porazdelitve za zvezne lastnosti.

V proceduri UNIVARIATE (koda 1.12) za vzorce, ki obsegajo do 2000 opazovanj, upoštevamo Shapiro-Wilk statistiko W (Wn). Za večje vzorce pa Kolmogorov (D) test. Porazdelitev je normalna v primeru, če je p-vrednost večja od 0,05. Zadostna kriterija, da porazdelitev definiramo kot normalno, sta koeficienta asimetričnost (skewness) in sploščenosti (kurtosis). Če sta v mejah med -1 in 1, smatramo, da je porazdelitev normalna, v nasprotnem primeru porazdelitev ni normalna.

Na normalnost porazdelitve testiramo ostanke, s katerimi delamo pri statističnih analizah (procedura GLM ...). S tem komponente, ki jih predstavljajo sistematski vplivi, niso vključene v izračune in ne povzročajo odstopanj v porazdelitvi.

PROC UNIVARIATE - procedura za testiranje normalnosti porazdelitve

DATA - navedemo ime niza podatkov

NORMAL - povemo proceduri, da naj testira normalnost porazdelitve

PLOT - grafični prikaz porazdelitve

VAR - ime spremenljivke, ki jo testiramo

HISTOGRAM - program nariše grafikone za porazdelitev (četrti stavek v kodi)

NORMAL - na grafu nariše krivuljo za normalno porazdelitev

CLASS - porazdelitev testiramo ločeno po razredih oz. nivojih vpliva, npr. ločeno za spol 1 in spol 2. Namesto CLASS lahko uporabimo tudi stavek BY.

HISTOGRAM - program nariše grafikone za porazdelitev

Algoritem 1.13 Dodatne opcije v proceduri UNIVARIATE

```
proc univariate data = NizPodatkov normal plot;
   class kvalitativne_sprem;
   var kvantitatitivne_sprem;
   histogram kvantitativne_sprem/normal midpoints=-90 to 90 by 5 outhistogram=DistBLUP;
   probplot  kvantitativne_sprem/normal midpoints=-90 to 90 by 5;
run;
```

Algoritem 1.14 Sintaksa v proceduri GCHART za grafični prikaz porazdelitev

```
proc gchart data = NizPodatkov;
   vbar kvantitatitivne_sprem/ type=freq;
   vbar kvantitatitivne_sprem/ type=percent ;
run;
```

Algoritem 1.15 Sintaksa v proceduri GPLOT za risanje razsevnih grafikov

```
proc gplot data = NizPodatkov;
   plot y*x;
run; quit;
```

MIDPOINTS = minimum TO maksimum BY korak– definiramo merilo na x osi

OUTHISTOGRAM – podatke za risanje porazdelitve shranimo v nov niz podatkov

PROBPLOT - program nariše grafikone za porazdelitev in verjetnost v output

Vseh stavkov v sintaksi procedure ne potrebujemo vedno in nekatere lahko izpustimo. Prav tako ni pomemben vrstni red ključnih besed v posameznem stavku.

1.7 Grafični prikaz

Za risanje grafikonov v SAS-u uporabljamo proceduri GCHART (koda 1.14) in GPLOT (koda 1.15).

PROC GCHART - procedura za risanje grafov

DATA - navedemo ime našega niza podatkov

VBAR - stolpični grafikon

TYPE – način prikaza (možnosti: število opazovanj (freq) ali delež opazovanj (percent)

PROC GPLOT - procedura za risanje razsevnih grafov

DATA - navedemo ime našega niza podatkov

PLOT - razsevni grafikon

y*x - odvisna spremenljivka * neodvisna spremenljivka

SYMBOLi - definiramo oblikovanje i

VALUE - oblika točke (opcije: plus, dot ...)

CV - barva točke

I - tip interpolacije (opcije: rl - linearna regresija, rq - kvadratna regresija, rc - kubna regresija, needle ...)

CI - barva interpolacije

x*y=1 - upošteva prvo oblikovanje

x*y=2 - upošteva drugo oblikovanje

OVERLAY - drugi graf prilepi čez prvega

NOFRAME - grafikon bo brez okvirja

Algoritem 1.16 Sintaksa v proceduri GPLOT za risanje razsevnih grafikov s polinomi različnih stopenj

```
symbol1 v=pluc cv=black i=rl ci=red;
symbol2 v=pluc cv=black i=rq ci=blue;
proc gplot data = NizPodatkov;
   plot y*x=1 y*x=2 / overlay noframe;
run;
```

Algoritem 1.17 Sintaksa v proceduri GLM

```
proc glm data = NizPodatkov;
   class kvalitativen_sprem;
   model lastnost = kvalitativne_sprem kvantitativne_sprem /solution xpx inverse;
   lsmeans vplivi_z_nivoji / pdiff stderr adjust=scheffe cl ;
   contrast 'nivo1 vs. nivo2' kvalitativna_sprem 1  -1  0  0 /e;
   estimate 'nivo1 vs. nivo2' kvalitativna_sprem 1  -1  0  0 /e;
   ouptput out=NizPodatkov_out p=yhat r=ehat;
run;
```

Tabela 1.1: Zapis enačbe modela v skalarni obliki in v SASu

Primer	Model v skalarni obliki	Zapis v SASu
Linearna regresija	$y_i = \mu + bx_i + e_i$	`model y = x;`
Polinom tretje stopnje	$y_i = \mu + b_I x_i + b_{II} x_i^2 + b_{III} x_i^3 + e_i$	`model y = x x*x x*x*x;`
Križna klasifikacija	$y_i = \mu + A_i + B_j + AB_{ij} + e_{ijk}$	`class A B;` `model y =A B A*B ;`
Ugnezden vpliv	$y_i = \mu + A_i + B_{ij} + e_{ijk}$	`class A B;` `model y =A B(A):`
Ugnezdena regresija	$y_i = \mu + A_i + b_i x_{ij} + e_{ij}$	`class A;` `model y =A x(A);`

1.8 Procedura GLM

Pri proceduri GLM (General Linear Models) lahko v model vključimo sistematske vplive z nivoji oz. razredi (spol, genotip, krma) in neodvisne spremenljivke (starost, telesna masa).

PROC GLM - procedura za posplošenih najmanjših kvadratov za linearne modele

DATA - naš niz podatkov v SASovi podatkovni zbirki

CLASS – navedemo vplive z nivoji oz. razredi (kvalitativne spremenljivke)

MODEL - tu zapišemo model, za modelom pa za "/" lahko napišemo različne opcije (SOLUTION, XPX, INVERSE ...)

LSMEANS (least square means) so ocene nivojev (tehtane srednje vrednosti) za kvalitativne vplive po metodi najmanjših kvadratov

PDIFF - izpišejo se p-vrednosti, ki povedo, če obstajajo statistično značilne razlike med nivoji vpliva

STDERR - izpiše se standardne napake za ocene pri nivojih vplivov

ADJUST=test - test, s katerim preveri ocene razlik med nivojev posameznega vpliva (opcije: tukey, scheffe, dunnet)

CL - izpišemo intervale zaupanja za ocene nivojev pri posameznih vplivih

CONTRAST - preverimo, če so razlike med nivoji statistično značilne

ESTIMATE - ocenimo razlike med nivoji po metodi (posplošnih) najmanjših kvadratov

OUTPUT - namenjen je zapisu originalnih podatkov skupaj s pričakovanimi vrednostmi (p pomeni pričakovana vrednost za y) in ostanki (r pomeni ocenjeni ostanek) v nov podatkovni niz (out = NizPodatkov_out)

Primere zapisa modela v proceduri GLM smo povzeli v tabeli 1.1.

PAZITE! Pri vključevanju polinomov višje stopnje v modelu nikoli ne izpustimo člene nižjih stopenj.

Algoritem 1.18 Osnovni primer zapisa v proceduri GLM

```
proc glm data=kunci;
   class genotip spol;
   model masa = genotip spol star_kor /solution;
run;
```

Algoritem 1.19 Primer zapisa interakcije in ugnezdene regresije v proceduri GLM

```
proc glm data = kunci;
   class genotip spol ;
   model masa = genotip spol genotip*spol star_kor(genotip)/solution;
run;
```

1.9 Korelacije

Proceduro CORR (koda 1.24) lahko uporabimo za izračun korelacijskih koeficientov med pari naključnih spremenljivk.

PROC CORR - procedura za izračun korelacijskih koeficientov

DATA - naš niz podatkov v SAS podatkovni zbirki

VAR - navedemo kvantitativne spremenljivke med katerimi želimo izračunati korelacijski koeficient

WITH - navedemo kvantitativne spremenljivke, s katerimi naj računa korelacijski koeficient (uporaba te opcije ni obvezna)

BY - navedemo skupine (kvalitativni vpliv) po katerih naj računa korelacijske koeficiente – npr. po rejcih (uporaba ni obvezna)

PAZITE! Korelacijo med spremenljivkami (paroma) lahko računamo, če so naključno vzorčene tako, da lahko predpostavljamo dvorazsežno normalno porazdelitev spremenljivk (angl. bivariate normal distribution). Zelo dobro je, če si povezave med pari spremenljivk narišemo na razsevni grafikon (angl. scatter plot). Pri tem lahko uporabimo proceduro GPLOT iz modula SAS/GRAPH (koda 14). Risanja pa se lahko lotimo tudi v MS Excelu. Pozorni moramo biti, ali je povezava med spremenljivkama linearna. Korelacijski koeficient namreč meri samo linearno povezanost!

1.10 Procedura SQL

Algoritem 1.20 Uporaba stavka CONTRAST v proceduri GLM

```
proc glm data = disek;
   class genotip spol;
   model dm = genotip / solution ;
   contrast 'razlika med G1 in G2' genotip 1  -1  0  0 /e;
                  /* ali obstaja razlika med genotipom 1 in genotipom 2 */
run;
```

Algoritem 1.21 Uporaba stavka ESTIMATE v proceduri GLM

```
proc glm data = kunci;
   class genotip;
   model masa = genotip / solution ;
   estimate 'lsmean G1' intercept 1 genotip 1 0 0 0 /e; run;
  /* Stavke lahko v GLMu poženemo posamično, a jih moramo zaključiti z run;
     če poženemo stavke skupaj, lahko ukaz run izpustimo */
   estimate 'lsmean G2' intercept 1 genotip 0 1  0 0 /e; run;
   estimate 'razlika G1-G2' intercept 0 genotip 1 -1 0 0 /e;
       /* ocenimo razliko med genotipoma 1 in 2*/
run;
```

Algoritem 1.22 Testiranje vseh členov regresijske krivulje - primer s kvadratno regresijo

```
proc glm data = kunci;
   class genotip spol;
   model masa = genotip spol star_kor star_kor*star_kor /solution ;run;
   contrast 'kv. reg.' intercept 0 star_kor 1 star_kor*star_kor 0,
                       intercept 0 star_kor 0 star_kor*star_kor 1/e;
run;
```

Algoritem 1.23 Testiranje možnosti poenostavitve ugnezdene regresije

```
proc glm data = kunci;
   class genotip spol;
   model masa = genotip spol star_kor star_kor*star_kor
                star_kor(genotip) star_kor*star_kor(genotip) /solution ;
run;
```

Algoritem 1.24 Sintaksa v proceduri CORR

```
proc corr data = NizPodatkov;
   var kvantitativna_sprem1 kvantitativna_sprem2;
   with kvantitativna_sprem1;
   by kvalitativna_sprem;
run;
```