Statistik Software – Projektbericht (Gruppe C)

Autoren:
Bieber, Okan 874666
Wahid Far, Okhtay 870485
Muhammet Can, Öz 876287

Gruppenparameter für Gruppe C:
N_A=25
Verteilung A: Normalverteilung mit mu=27 und sigma=5

N_B=25
Verteilung B: Normalverteilung mit mu=30 und sigma=5Seed: 5557

Index:

Aufgabenstellungen:

Teilaufgabe a)

In dieser Teilaufgabe musste man zwei Datensätze erzeugen welche Altersangaben simulieren. Diese hingen von den jeweiligen Werten ab die vorgegeben waren. Wir waren Gruppe C somit war für uns gegeben dass sowohl Gruppe A als auch Gruppe B 25 Individuen beinhaltete.

Die Altersangaben sollten jedoch als Zufallszahlen generiert werden hierfür war als Hinweis gegeben dass wir diese über die in SAS definierte RAND() Funktion machen sollen. Hierfür waren für unsere Gruppe weitere Informationen gegeben und zwar dass es in beiden Fällen eine Normalverteilung seien musste, ein μ von 27 für Gruppe A und ein μ von 30 für Gruppe B. In beiden Gruppen betrug das $\sigma = 5$.

Die letztendliche Funktion für z.B. Gruppe A sieht in der SAS-Schreibweise folgendermaßen aus:
ROUND(RAND('Normal',27 ,5)) .

Sobald der Schritt vollendet ist fügen wir die Tabellen vertikal aneinander über den PROC SQL Befehl OUTER UNION CORR. Das OUTER UNION haben wir verwendet damit nicht schon beim Zusammenlegen der Tabellen die Gruppen vermischt werden, somit erzeugen wir eine Tabelle mit 50 Observationen. Die ersten 25 sind die Gruppe A und die darauf folgenden 25 sind die Gruppe B. Durch das zusätzliche Statement CORR garantieren wir, dass alle Variablen durch ihre Namen zugeordnet werden.

Teilaufgabe b)

In dieser Teilaufgabe müssen die noch ungemischten Informationen durch die TTEST Prozedur analysiert werden. Als Eingabe der Prozedur benötigen wir die Tabelle aus der Aufgabe a).
Durch den Zusatz „sides“ können wir spezifizieren ob es ein Einseitiger oder ein Zweiseitiger t-Test werden soll. Da wir ja zwei Gruppen haben und das über die Variable „Gruppe“ festlegen, ist es relativ selbsterklärend, dass wir einen Zweiseitgen t-Test benötigen. Innerhalb der Prozedur müssen wir die zu betrachtende Variable angeben, welche wir als „alter“ bezeichnet haben. Das Statement „class“ trennt über die Variable „Gruppe“ die Altersangaben in die jeweiligen Ursprungsgruppen.

```
/* t test und bestimmung mittelwert usw. */
PROC TTEST DATA=WORK.BEIDEGRUPPEN SIDES=2 PLOTS=none  ;
    CLASS Gruppe;
    VAR alter;
RUN;
```

Die Prozedur TTEST

Variable: alter

Gruppe	Methode	N	Mittelwert	Std.abw.	Std.fehler	Minimum	Maximum
A		25	27.4000	4.2622	0.8524	20.0000	36.0000
B		25	29.9600	4.3730	0.8746	23.0000	40.0000
Diff (1-2)	Gepoolt		-2.5600	4.3180	1.2213		
Diff (1-2)	Satterthwaite		-2.5600		1.2213		

Gruppe	Methode	Mittelwert	95% CL Mittelwert		Std.abw.	95% CL Std.abw	
A		27.4000	25.6406	29.1594	4.2622	3.3281	5.9294
B		29.9600	28.1549	31.7651	4.3730	3.4146	6.0835
Diff (1-2)	Gepoolt	-2.5600	-5.0156	-0.1044	4.3180	3.6009	5.3945
Diff (1-2)	Satterthwaite	-2.5600	-5.0157	-0.1043			

Methode	Varianzen	DF	t-Wert	Pr > |t|
Gepoolt	Gleich	48	-2.10	0.0414
Satterthwaite	Ungleich	47.968	-2.10	0.0414

Gleichheit der Varianzen				
Methode	Num DF	Den DF	F-Wert	Pr > F
Folded F	24	24	1.05	0.9010

Teilaufgabe c)

Diese Teilaufgabe war die womöglich aufwendigste Teilaufgabe. Letztendlich musste man immer wieder einzelne Altersangaben auf „Missing“ setzen, also „ alter = . ; “.
Man sollte das nun mit einer gewissen prozentualen Anzahl der Altersangaben machen, das heißt in 2% Schritte von 0% bis zu 50%. Daraufhin folgt soll ein t-Test, der jedes Mal die Tabelle neu betrachtet, wenn die angegebene Anzahl an Werten auf „Missing“ gesetzt wurde. Schließlich mussten wir spezifische Angaben aus dem t-Test extrahieren und fassten alle Daten in einer Endtabelle zusammen. Die benötigten Werte sind einmal die Mittelwerte mit dem zugehörigem

Konfidenzintervall, die Differenz der Mittelwerte auch hier mit zugehörigem Konfidenzintervall und der p-Wert des t-Testes.
Wir entnehmen nur jenen Mittelwert der Differenz aus dem t-Test, wo die Spalte Methode „Gepooled“ gilt. Wir gehen nämlich davon aus, dass die Varianz zwischen beiden Gruppen identisch ist.

Eine weitere Angabe die uns übergeben wurde war der Seed. Diesen mussten wir hier verwenden, um die Tabelle durchzumischen. Wir sind letztendlich so herangegangen, dass wir eine neue Tabelle erzeugt haben, welche 50 Spalten lang ist und mit über „RAND()“ generierten Werten gefüllt ist.
Um eine willkürliche Zahl mit „RAND()“ ausgeben zu lassen, benötigen wir eine Zahl zum Initiieren. Dafür ist der Seed zuständig, welcher in unserer Gruppe den Wert 5557 hat und mit „ Call streaminit(seed)“ eingeleitet wird .Die nun entstandene Tabelle fügen wir dann horizontal an die ursprüngliche Tabelle hinzu und sortieren noch alles in derselben SQL Prozedur mit „ORDER BY x“. Hierbei ist x die Spalte mit den zufällig generierten Werten. Somit werden die Altersangaben durchmischt.

Ab hier beginnt nun die Makrofunktion. In dieser Makrofunktion umklammert eine Schleife alles Notwendige. Zu allererst wird über ein Datastep die Werte auf „Missing“ gesetzt. Die jeweilige Prozentzahl hängt von der Laufvariablen in der Makroschleife ab. Darauf folgt auch schon der nächste t-Test, der ebenfalls zweiseitig ist. Um später die Werte weiterverwenden zu können, welche vom t-Test ausgegeben werden, müssen wir den Tabellen über den Befehl „ods output“ Namen zuordnen.
Der darauf folgende „PROC SQL“ Block ist dafür zuständig nun eben diese relevanten Daten in eine Zeile zu bringen damit wir wie angekündigt am Ende der Makroschleife für jede Prozentzahl die Informationen in eine Zeile bekommen. Dazu haben wir mehrere Tabellen im „PROC SQL“ Block erstellt und diese dann zusammengefügt.
Wenn wir dann die Makroschleife durch den Code 25-mal durchlaufen hat können wir zu guter Letzt anhand „SET end_daten:“ jegliche Zeilen in eine große Tabelle zusammenfügen welche aus 25 Observationen besteht. Hier ist nochmal ein Ausschnitt dieser Endtabelle zu sehen:

Beob.	MeanA	LCLA	UCLA	MeanB	LCLB	UCLB	MeanDiff	p
1	29.1600	27.1815	31.1385	29.6800	27.3367	32.0233	-0.5200	0.7279
2	29.2222	26.6948	31.7496	29.3478	26.8378	31.8579	-0.1256	0.9425
3	29.2222	26.6948	31.7496	29.0455	26.4949	31.5960	0.1768	0.9194
4	29.2222	26.6948	31.7496	28.6190	26.1032	31.1349	0.6032	0.7265
5	29.2222	26.6948	31.7496	28.5000	25.8591	31.1409	0.7222	0.6822
6	29.2222	26.6948	31.7496	28.8947	26.2407	31.5488	0.3275	0.8522
7	29.2222	26.6948	31.7496	29.0000	26.1921	31.8079	0.2222	0.9020

Die Befehle für PROC SQL sind folgendermaßen:

```
*t-Test;
        ODS OUTPUT ConfLimits=KIntervall;
        ODS OUTPUT TTests=p_wert;
        PROC TTEST DATA=plot_sortiertTTest&i SIDES=2 PLOTS=none ;
            CLASS Gruppe;
            VAR alter;
        RUN;

        *In diesem SQL Block werden dem t-Test die informationen entnommen;
        PROC SQL;
                CREATE TABLE plot_011 AS
                SELECT Mean AS MeanA, LowerCLMean AS LCLA,  UpperCLMean AS UCLA ,variable
                FROM KIntervall As a
                WHERE Class='A';

                CREATE TABLE plot_022 AS
                SELECT Mean AS MeanB, LowerCLMean AS LCLB,  UpperCLMean AS UCLB ,variable
                FROM KIntervall AS b
                WHERE Class='B';

                CREATE TABLE plot_033 AS
                SELECT Mean AS MeanDiff, variable
                FROM KIntervall
                WHERE Variances='Gleich';

                CREATE TABLE plot_011_022 AS
                SELECT *
                FROM plot_011 AS a
                LEFT JOIN plot_022 AS b ON a.variable=b.variable;

                CREATE TABLE plot_all AS
                SELECT *
                FROM plot_011_022
                AS c
                LEFT JOIN plot_033 AS d ON c.variable=d.variable;

                CREATE TABLE p_wert_plot AS
                SELECT Probt AS p, variable FROM p_wert;

                CREATE TABLE komplett_plot AS
                SELECT * FROM plot_all AS e
                LEFT JOIN p_wert_plot AS f ON e.variable=f.variable;
        QUIT;
```

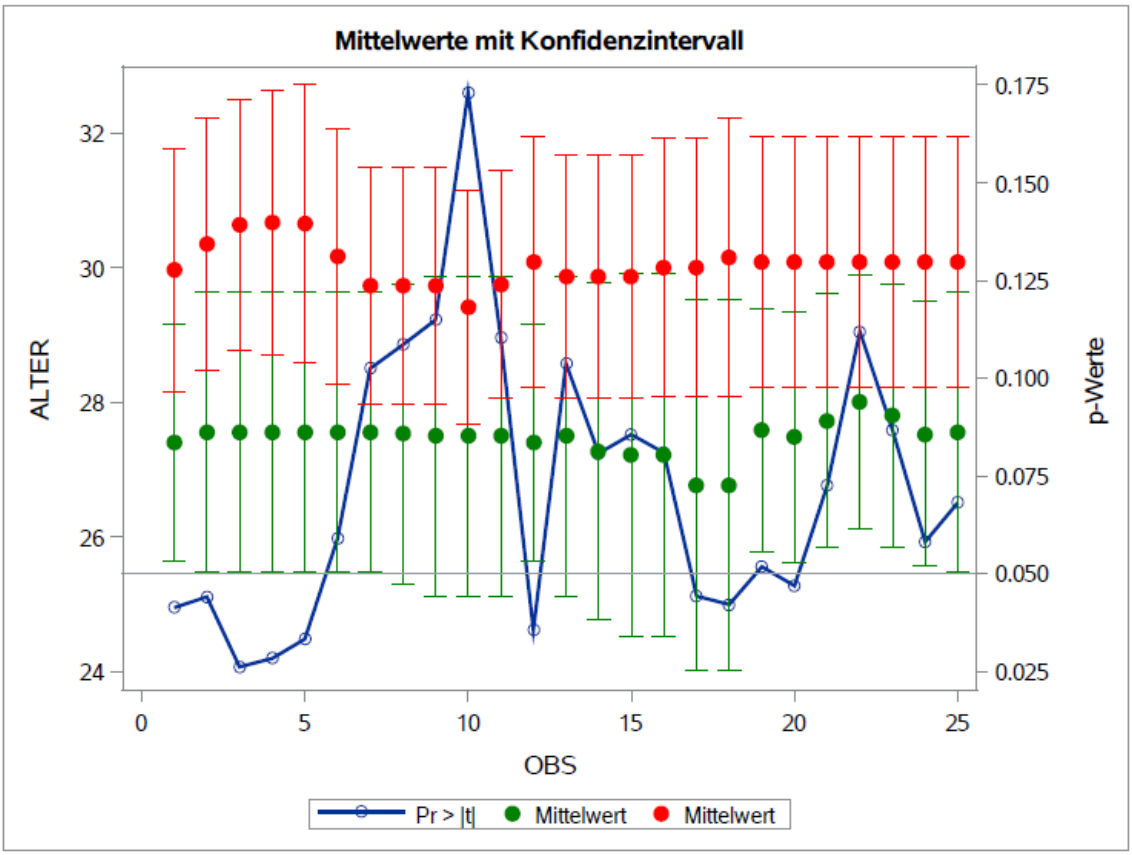
Teilaufgabe d)

Wir erstellen hier eine Grafik, wo folgende Werte und Achsen existieren:

1) Mittelwerte mit den dazugehörigen Konfidenzintervallen beider Gruppen für alle Prozentangaben
2) Ein Alphalevel als Linie mit alpha=0.05
3) Linke y-Achse wo die Werte aus der Spalte „Alter“ angezeigt werden
4) Rechte y-Achse, wo die p-Werte angezeigt werden
5) X-Achse, wo die Anzahl der Beobachtungen angezeigt werden

Zu Beginn haben wir eine id-Spalte in den Datensatz eingebettet, um die Werte für die x-Achse zu definieren. Wir hatten nämlich keine Variable für die Anzahl der Beobachtungen.
Wir nutzen dazu „PROC SGPLOT“ mit den SCATTER Befehlen. Denn durch die SCATTER Befehle wird die Modellierung der

Konfidenzintervalle erleichtert. Es kann nämlich passieren, dass sich die Konfidenzintervalle überlappen.



Teilaufgabe e)

Wir erstellen hier eine Grafik, wo folgende Werte und Achsen existieren:

1) Differenz der Mittelwerte mit den dazugehörigen Konfidenzintervallen beider Gruppen für alle Prozentangaben
2) Referenzlinie, wenn die Differenz der Mittelwerte 0 beträgt
3) X-Achse mit den Werten der Beobachtungen
4) Linke y-Achse gibt die Differenz der Mittelwerte wieder
5) Rechte y-Achse gibt die p-Werte wieder

Zu Beginn haben wir zwei weitere Spalten zu dem Hauptdatensatz hinzugefügt: UCLDiff und LCLDiff.
Das Ziel war es ein Konfidenzintervall zu erzeugen, da wir das für die grafische Darstellung benötigen.

Anhand der Grafik kann man erkennen, dass es einen Zusammenhang zwischen den Werte der Differenz der Mittelwerte und den p-Werten zu jeder Prozentangabe existiert.