\*\*\*\* BODNAR WEIGHTED PROBABILISTIC BIAS ANALYSIS

WITH 5 BMI CATEGORIES AND 12 SETS OF VALIDATION DATA\*\*\*\*

\* Options;

options obs = max spool ps=**40** ls=**124**;

\* dummy file to turn the SAS log on and off (remove comments once the program runs well

and you increase reptot to 100,000 to avoid the program pausing once the log is full);

/\*

filename dumfl dummy;

proc printto log=dumfl; run;

\*/

\*filename myfile 'J:\Sensitivity analysis\Tim weighted results\log\_sens\_bmi\_032613.log';

\*proc printto log=myfile;

**run**;

\*/

\*set library names;

libname sensdir 'F:\Sensitivity analysis';

libname writeout 'F:\Sensitivity analysis\Tim weighted results';

**run**;

\*create a reduced data set to help the program run quickly;

**data** anal\_1;

set sensdir.bodnarweightsv3;

if bmicat=**.** or married=**.** or agecat=**.** or edcat=**.** or parity=**.** or smoker=**.** or race=**.** or ptb32=**.** or hospital=**.** or metro=**.** or private=**.** then delete;

keep agecat gwgcat bmicat ptb32 edcat married race hospital metro parity smoker private frequency;

**run**;

**data** reduce (compress=yes);

set anal\_1;

\*create validation data groups;

valgrp = gwgcat\***100**+race\***10**+ptb32;

**run**;

\*bias analysis;

**data** writeout.combine1 (keep = agecat gwgcat bmicat ptb32 edcat married race hospital metro parity smoker private frequency newfreqt newbmigrp repcnt compress=yes);

\*validation groups;

\*read validation data into array;

array valchck{**12**} valchck1-valchck12 (**110** **111** **210** **211** **310** **311** **120** **121** **220** **221** **320** **321**);

\*validation data;

array valdata {**12**,**5**,**5**}

(

/\*<20th % GWGzscore, white race, term birth\*/

**28** **5** **0** **0** **0**

**0** **20** **4** **0** **0**

**0** **3** **4** **0** **0**

**0** **2** **7** **22** **0**

**0** **0** **2** **4** **25**

/\*<20th % GWGzscore, white race, preterm birth\*/

**10** **4** **0** **0** **0**

**0** **22** **1** **0** **0**

**0** **1** **5** **0** **0**

**0** **0** **3** **10** **1**

**0** **0** **0** **2** **8**

/\*20-80th % GWGzscore, white race, term birth\*/

**22** **12** **0** **0** **0**

**0** **24** **0** **0** **0**

**0** **1** **5** **0** **0**

**0** **1** **1** **27** **2**

**0** **0** **1** **2** **29**

/\*20-80th % GWGzscore, white race, preterm birth\*/

**19** **7** **0** **0** **0**

**0** **21** **2** **0** **0**

**0** **0** **7** **0** **0**

**0** **0** **1** **23** **5**

**0** **0** **0** **2** **26**

/\*>80th % GWGzscore, white race, term birth\*/

**24** **6** **1** **0** **0**

**0** **18** **5** **0** **0**

**0** **0** **7** **1** **0**

**0** **0** **2** **20** **5**

**0** **0** **0** **1** **30**

/\*>80th % GWGzscore, white race, preterm birth\*/

**10** **7** **0** **0** **0**

**0** **20** **4** **1** **0**

**0** **0** **4** **0** **0**

**0** **0** **1** **14** **6**

**0** **0** **0** **1** **24**

/\*<20th % GWGzscore, black race, term birth\*/

**20** **5** **0** **0** **0**

**1** **23** **0** **0** **0**

**0** **2** **6** **0** **0**

**0** **2** **13** **13** **2**

**0** **0** **4** **2** **24**

/\*<20th % GWGzscore, black race, preterm birth\*/

**5** **1** **0** **0** **0**

**0** **22** **1** **0** **0**

**0** **1** **4** **0** **0**

**0** **1** **2** **9** **0**

**0** **0** **0** **0** **2**

/\*20-80th % GWGzscore, black race, term birth\*/

**21** **8** **0** **0** **0**

**1** **16** **0** **0** **0**

**0** **0** **12** **2** **1**

**0** **0** **4** **15** **10**

**0** **1** **0** **2** **30**

/\*20-80th % GWGzscore, black race, preterm birth\*/

**5** **4** **0** **0** **0**

**0** **19** **0** **0** **0**

**0** **0** **8** **1** **0**

**0** **0** **4** **15** **3**

**0** **0** **0** **0** **27**

/\*>80th % GWGzscore, black race, term birth\*/

**17** **12** **1** **0** **0**

**0** **15** **5** **0** **0**

**0** **0** **10** **2** **0**

**0** **1** **5** **18** **6**

**0** **0** **0** **1** **29**

/\*>80th % GWGzscore, black race, preterm birth\*/

**1** **3** **0** **0** **0**

**0** **16** **1** **0** **0**

**0** **1** **7** **3** **0**

**0** **0** **0** **10** **5**

**0** **0** **0** **0** **16**

)

; array ppv{**12**,**5**} ppv1-ppv60;

array limits{**12**,**5**,**6**} limits1-limits360;

do cnta=**1** to **12**;

do cntb=**1** to **5**;

ppv{cnta,cntb}= valdata{cnta,cntb,cntb}/(valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**});

if cntb eq **1** then do;

limits{cnta,cntb,**1**} = **0**;

limits{cnta,cntb,**2**} = **0**;

limits{cnta,cntb,**3**} = valdata{cnta,cntb,**2**}/max((valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**4**} = limits{cnta,cntb,**3**}+valdata{cnta,cntb,**3**}/max((valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**5**} = limits{cnta,cntb,**4**}+valdata{cnta,cntb,**4**}/max((valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**6**} = **1**; end;

else if cntb eq **2** then do;

limits{cnta,cntb,**1**} = **0**;

limits{cnta,cntb,**2**} = valdata{cnta,cntb,**1**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**3**} = limits{cnta,cntb,**2**};

limits{cnta,cntb,**4**} = limits{cnta,cntb,**3**}+valdata{cnta,cntb,**3**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**5**} = limits{cnta,cntb,**4**}+valdata{cnta,cntb,**4**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**6**} = **1**; end;

else if cntb eq **3** then do;

limits{cnta,cntb,**1**} = **0**;

limits{cnta,cntb,**2**} = valdata{cnta,cntb,**1**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**3**} = limits{cnta,cntb,**2**}+valdata{cnta,cntb,**2**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**4**} = limits{cnta,cntb,**3**};

limits{cnta,cntb,**5**} = limits{cnta,cntb,**4**}+valdata{cnta,cntb,**4**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**4**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**6**} = **1**; end;

else if cntb eq **4** then do;

limits{cnta,cntb,**1**} = **0**;

limits{cnta,cntb,**2**} = valdata{cnta,cntb,**1**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**3**} = limits{cnta,cntb,**2**}+valdata{cnta,cntb,**2**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**4**} = limits{cnta,cntb,**3**}+valdata{cnta,cntb,**3**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**5**}),**0.1**);

limits{cnta,cntb,**5**} = limits{cnta,cntb,**4**};

limits{cnta,cntb,**6**} = **1**; end;

else if cntb eq **5** then do;

limits{cnta,cntb,**1**} = **0**;

limits{cnta,cntb,**2**} = valdata{cnta,cntb,**1**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}),**0.1**);

limits{cnta,cntb,**3**} = limits{cnta,cntb,**2**}+valdata{cnta,cntb,**2**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}),**0.1**);

limits{cnta,cntb,**4**} = limits{cnta,cntb,**3**}+valdata{cnta,cntb,**3**}/max((valdata{cnta,cntb,**1**}+valdata{cnta,cntb,**2**}+valdata{cnta,cntb,**3**}+valdata{cnta,cntb,**4**}),**0.1**);

limits{cnta,cntb,**5**} = **1**;

limits{cnta,cntb,**6**} = **1**; end;

end;

end;

array newfreq{**5**} newfreq1-newfreq5;

if seed eq **.** then seed=-**1**;

\*use 100 iterations to start, ultimately increase reptot to 20k to 100k;

repcnt = **1**; reptot = **100**;

do while (repcnt le reptot);

\*iterations;

do reccnt = **1** to last;

set reduce point=reccnt nobs=last;

\*clear the previous counts;

do clrcnt= **1** to **5**; newfreq{clrcnt}=**0**; end;

\*find the right validation row;

do i=**1** to **12**;

if valchck{i} eq valgrp then do;

if ppv{i,bmicat} eq **1** then newfreq{bmicat}= frequency;

else newfreq{bmicat}=ranbin(seed,frequency,ppv{i,bmicat});

deltafreq=frequency-newfreq{bmicat};

if deltafreq gt **0** then do;

do slidecnt = **1** to deltafreq;

slide = ranuni(seed);

do colcnt=**1** to **5**;

if limits{i,bmicat,colcnt} le slide le limits{i,bmicat,colcnt+**1**} then newfreq{colcnt} = newfreq{colcnt}+**1**;

end;

end;

end;

end;

end;

do newbmigrp = **1** to **5**;

if newfreq{newbmigrp} gt **0** then do;

newfreqt = newfreq{newbmigrp};

output;

end;

end;

end;

repcnt = repcnt+**1**;

end;

stop;

**run**;

\*check the original against one iteration;

**proc** **means** data=work.reduce fw=**6** n sum;

class ptb32 agecat;

var frequency;

**run**;

**proc** **means** data=writeout.combine1 (where=(repcnt=**1**)) fw=**6** n sum;

class ptb32 agecat;

var newfreqt;

**run**;

**proc** **means** data=work.reduce fw=**6** n sum;

class ptb32 bmicat;

var frequency;

**run**;

**proc** **means** data=writeout.combine1 (where=(repcnt=**1**)) fw=**6** n sum;

class ptb32 newbmigrp;

var newfreqt;

**run**;

\* first the conventional model without bias analysis;

**proc** **logistic** descending data=anal\_1;

class agecat (ref='0') bmicat (ref='2') edcat (ref='0') race (ref='1') hospital (ref='0') / param=ref;

model ptb32 = agecat bmicat edcat race hospital married metro parity smoker private / cl ;

weight frequency;

**run**;

\* second models with sensitivity analysis results;

**proc** **logistic** descending data=writeout.combine1 outest=sensdir.allsens covout noprint;

class agecat (ref='0') newbmigrp (ref='2') edcat (ref='0') race (ref='1') hospital (ref='0') / param=ref;

model ptb32 = agecat newbmigrp edcat race hospital married metro parity smoker private / cl ;

weight newfreqt;

by repcnt;

**run**;

**data** anal\_3 (where=(\_type\_='PARMS')); set writeout.allsens;

keep agecat1 agecat2 edcat1 edcat2 edcat3 hospital1 hospital2

married metro parity private race2 smoker

newbmigrp1 newbmigrp3 newbmigrp4 newbmigrp5

\_type\_;

**run**;

\*invoke SAS interactive matrix language to get the parameter estimates;

\*NOTE: you must change icount to the number of iterations (reptot);

**proc** **iml**;

use writeout.allsens;

read all into alldat;

c=**2**:**19**; d=**2**:**19**; /\*1st col is intercept. Col 2 starts parameters. end at # of vars in "keep" above + 1. e.g. 25 vars + \_type\_ + 1 =27 after :\*/

do icount = **1** to **100**;

pf=(icount-**1**)\***19**+**1**; /\* (icount-1)\*x+1 change x to the same number as in c=2:xx above \*/

parama = alldat[pf,c];

cova = alldat[d,c];

parama = t(parama);

stda = j(**18**,**1**,**0**); /\*(x,1,0) - x should be number of vars in "keep" statement above\*/

covroot = root(cova);

covrtt = t(covroot);

stda = normal(stda);

pouta = parama+covrtt\*stda;

poutb = t(pouta);

if icount = **1** then create boota from poutb;

append from poutb;

d=d+**19**; /\* d+x where x=number of vars in keep above + 1 \*/

end;

**quit**;

**run**;

\*This statement creates the bootstrapped dataset;

**data** writeout.bootb (rename=(col1=intercept col2=agecat1b col3=agecat2b col4=newbmigrp1b col5=newbmigrp3b

col6=newbmigrp4b col7=newbmigrp5b col8=edcat1b col9=edcat2b col10=edcat3b col11=race2b

col12=hospital1b col13=hospital2b col14=marriedb col15=metrob col16=parityb

col17=smokerb col18=privateb)); set boota;

**run**;

\*Combine the parameter estimates taking account of only with misclassification with the

bootstrapped parameter estimates (adding in sampling error);

**data** writeout.combine; merge anal\_3 (drop=\_type\_) writeout.bootb;

**run**;

\* use proc univariate to generate tabular results, written to data set pctls;

**proc** **univariate** data=writeout.combine cipctldf (type=asymmetric) noprint;

var agecat1 agecat2 newbmigrp1 newbmigrp3 newbmigrp4 newbmigrp5

edcat1 edcat2 edcat3 race2 hospital1 hospital2 married metro parity smoker private

agecat1b agecat2b newbmigrp1b newbmigrp3b newbmigrp4b newbmigrp5b

edcat1b edcat2b edcat3b race2b hospital1b hospital2b marriedb metrob parityb smokerb privateb;

\* create output dataset with min, max, mean, median mode of paramter estimates with

and without bootstrapping;

output out=writeout.results

median=

mdagecat1 mdagecat2 mdnewbmigrp1 mdnewbmigrp3 mdnewbmigrp4 mdnewbmigrp5

mdedcat1 mdedcat2 mdedcat3 mdrace2 mdhospital1 mdhospital2 mdmarried mdmetro mdparity mdsmoker mdprivate

mdagecat1b mdagecat2b mdnewbmigrp1b mdnewbmigrp3b mdnewbmigrp4b mdnewbmigrp5b

mdedcat1b mdedcat2b mdedcat3b mdrace2b mdhospital1b mdhospital2b mdmarriedb mdmetrob mdparityb mdsmokerb mdprivateb

var=

agecat1 agecat2 newbmigrp1 newbmigrp3 newbmigrp4 newbmigrp5

edcat1 edcat2 edcat3 race2 hospital1 hospital2 married metro parity smoker private

agecat1b agecat2b newbmigrp1b newbmigrp3b newbmigrp4b newbmigrp5b newbmigrp6b

edcat1b edcat2b edcat3b race2b hospital1b hospital2b marriedb metrob parityb smokerb privateb

pctlpts = **2.5** **97.5**

pctlpre =

agecat1 agecat2 newbmigrp1 newbmigrp3 newbmigrp4 newbmigrp5

edcat1 edcat2 edcat3 race2 hospital1 hospital2 married metro parity smoker private

agecat1b agecat2b newbmigrp1b newbmigrp3b newbmigrp4b newbmigrp5b

edcat1b edcat2b edcat3b race2b hospital1b hospital2b marriedb metrob parityb smokerb privateb

pctlname = p025 p975;

**run**;

**data** sensdir.results\_exp;

set writeout.results;

agecat1e=exp(mdagecat1);

agecat2e=exp(mdagecat2);

newbmigrp1e=exp(mdnewbmigrp1);

newbmigrp3e=exp(mdnewbmigrp3);

newbmigrp4e=exp(mdnewbmigrp4);

newbmigrp5e=exp(mdnewbmigrp5);

edcat1e=exp(mdedcat1);

edcat2e=exp(mdedcat2);

edcat3e=exp(mdedcat3);

race2e=exp(mdrace2);

hospital1e=exp(mdhospital1);

hospital2e=exp(mdhospital2);

marriede=exp(mdmarried);

metroe=exp(mdmetro);

paritye=exp(mdparity);

smokere=exp(mdsmoker);

privatee=exp(mdprivate);

agecat1be=exp(mdagecat1b);

agecat2be=exp(mdagecat2b);

newbmigrp1be=exp(mdnewbmigrp1b);

newbmigrp3be=exp(mdnewbmigrp3b);

newbmigrp4be=exp(mdnewbmigrp4b);

newbmigrp5be=exp(mdnewbmigrp5b);

edcat1be=exp(mdedcat1b);

edcat2be=exp(mdedcat2b);

edcat3be=exp(mdedcat3b);

race2be=exp(mdrace2b);

hospital1be=exp(mdhospital1b);

hospital2be=exp(mdhospital2b);

marriedbe=exp(mdmarriedb);

metrobe=exp(mdmetrob);

paritybe=exp(mdparityb);

smokerbe=exp(mdsmokerb);

privatebe=exp(mdprivateb);

agecat1ep025=exp(agecat1p025);

agecat2ep025=exp(agecat2p025);

newbmigrp1ep025=exp(newbmigrp1p025);

newbmigrp3ep025=exp(newbmigrp3p025);

newbmigrp4ep025=exp(newbmigrp4p025);

newbmigrp5ep025=exp(newbmigrp5p025);

edcat1ep025=exp(edcat1p025);

edcat2ep025=exp(edcat2p025);

edcat3ep025=exp(edcat3p025);

race2ep025=exp(race2p025);

hospital1ep025=exp(hospital1p025);

hospital2ep025=exp(hospital2p025);

marriedep025=exp(marriedp025);

metroep025=exp(metrop025);

parityep025=exp(parityp025);

smokerep025=exp(smokerp025);

privateep025=exp(privatep025);

agecat1bep025=exp(agecat1bp025);

agecat2bep025=exp(agecat2bp025);

newbmigrp1bep025=exp(newbmigrp1bp025);

newbmigrp3bep025=exp(newbmigrp3bp025);

newbmigrp4bep025=exp(newbmigrp4bp025);

newbmigrp5bep025=exp(newbmigrp5bp025);

edcat1bep025=exp(edcat1bp025);

edcat2bep025=exp(edcat2bp025);

edcat3bep025=exp(edcat3bp025);

race2bep025=exp(race2bp025);

hospital1bep025=exp(hospital1bp025);

hospital2bep025=exp(hospital2bp025);

marriedbep025=exp(marriedbp025);

metrobep025=exp(metrobp025);

paritybep025=exp(paritybp025);

smokerbep025=exp(smokerbp025);

privatebep025=exp(privatebp025);

agecat1ep975=exp(agecat1p975);

agecat2ep975=exp(agecat2p975);

newbmigrp1ep975=exp(newbmigrp1p975);

newbmigrp3ep975=exp(newbmigrp3p975);

newbmigrp4ep975=exp(newbmigrp4p975);

newbmigrp5ep975=exp(newbmigrp5p975);

edcat1ep975=exp(edcat1p975);

edcat2ep975=exp(edcat2p975);

edcat3ep975=exp(edcat3p975);

race2ep975=exp(race2p975);

hospital1ep975=exp(hospital1p975);

hospital2ep975=exp(hospital2p975);

marriedep975=exp(marriedp975);

metroep975=exp(metrop975);

parityep975=exp(parityp975);

smokerep975=exp(smokerp975);

privateep975=exp(privatep975);

agecat1bep975=exp(agecat1bp975);

agecat2bep975=exp(agecat2bp975);

newbmigrp1bep975=exp(newbmigrp1bp975);

newbmigrp3bep975=exp(newbmigrp3bp975);

newbmigrp4bep975=exp(newbmigrp4bp975);

newbmigrp5bep975=exp(newbmigrp5bp975);

edcat1bep975=exp(edcat1bp975);

edcat2bep975=exp(edcat2bp975);

edcat3bep975=exp(edcat3bp975);

race2bep975=exp(race2bp975);

hospital1bep975=exp(hospital1bp975);

hospital2bep975=exp(hospital2bp975);

marriedbep975=exp(marriedbp975);

metrobep975=exp(metrobp975);

paritybep975=exp(paritybp975);

smokerbep975=exp(smokerbp975);

privatebep975=exp(privatebp025);

**run**;

**proc** **print** data=sensdir.results\_exp;

var

agecat1e agecat1ep025 agecat1ep975

agecat2e agecat2ep025 agecat2ep975

newbmigrp1e newbmigrp1ep025 newbmigrp1ep975

newbmigrp3e newbmigrp3ep025 newbmigrp3ep975

newbmigrp4e newbmigrp4ep025 newbmigrp4ep975

newbmigrp5e newbmigrp5ep025 newbmigrp5ep975

edcat1e edcat1ep025 edcat1ep975

edcat2e edcat2ep025 edcat2ep975

edcat3e edcat3ep025 edcat3ep975

race2e race2ep025 race2ep975

hospital1e hospital1ep025 hospital1ep975

hospital2e hospital2ep025 hospital2ep975

marriede marriedep025 marriedep975

metroe metroep025 metroep975

paritye parityep025 parityep975

smokere smokerep025 smokerep975

privatee privateep025 privateep975

agecat1be agecat1bep025 agecat1bep975

agecat2be agecat2bep025 agecat2bep975

newbmigrp1be newbmigrp1bep025 newbmigrp1bep975

newbmigrp3be newbmigrp3bep025 newbmigrp3bep975

newbmigrp4be newbmigrp4bep025 newbmigrp4bep975

newbmigrp5be newbmigrp5bep025 newbmigrp5bep975

edcat1be edcat1bep025 edcat1bep975

edcat2be edcat2bep025 edcat2bep975

edcat3be edcat3bep025 edcat3bep975

race2be race2bep025 race2bep975

hospital1be hospital1bep025 hospital1bep975

hospital2be hospital2bep025 hospital2bep975

marriedbe marriedbep025 marriedbep975

metrobe metrobep025 metrobep975

paritybe paritybep025 paritybep975

smokerbe smokerbep025 smokerbep975

privatebe privatebep025 privatebep975;

**run**;

**quit**;