/\* Remove library \*/

/\* Final project \*/

ODS HTML;

ODS LISTING CLOSE;

ODS GRAPHICS ON;

/\*Creating new dataset as "ad" and reading the data\*/

LIBNAME Project2 "E:\Users\axr180085\Desktop\Project";

**data** ad;

set Project2.DATA;

**run**;

/\* Create train and test datasets. 70% of sample in train \*/

/\*output 1 \*/

**proc** **surveyselect** data=ad out=ad\_smp outall samprate=**0.7** seed=**10**;

**run**;

**data** ad\_train ad\_test;

set ad\_smp;

if selected then output ad\_train;

else output ad\_test;

**run**;

/\* 1. Linear probability model\*/

/\* Generate indicator variables using glm\_mod to run proc reg for the categorical variables \*/

**proc** **glmmod** data=ad\_smp outdesign=LPM\_ind noprint;

class device\_platform\_class;

model install = device\_volume wifi resolution device\_height device\_width publisher\_id\_class device\_os\_class device\_make\_class device\_platform\_class/noint;

weight selected;

**run**;

/\* Generate indicator variables using glm\_mod for train dataset \*/

**proc** **glmmod** data=ad\_train outdesign=ad\_train\_ind noprint;

class device\_platform\_class;

model install = device\_volume wifi resolution device\_height device\_width publisher\_id\_class device\_os\_class device\_make\_class device\_platform\_class/noint;

**run**;

/\* Generate indicator variables using glm\_mod for test dataset \*/

**proc** **glmmod** data=ad\_test outdesign=ad\_test\_ind noprint;

class device\_platform\_class;

model install = device\_volume wifi resolution device\_height device\_width publisher\_id\_class device\_os\_class device\_make\_class device\_platform\_class/noint;

**run**;

/\*output 2,3 \*/

**proc** **contents** data= ad\_train\_ind;

**run**;

/\* Initial Model: Linear probability model using PROC reg \*/

/\*output 4,5,6 \*/

**proc** **reg** data=ad\_train\_ind PLOTS(MAXPOINTS=**100000**);

model install = col1 - col10 ;

/\* OUTPUT OUT=LPM pred=p ;\*/

**quit**;

/\* Trial 1\*/

/\* Note: A new model was tried by taking log for all predictor variables, however the result was not better than the initial model and hence not used\*/

/\* Log model - not required \*/

**data** ad\_logged;

set Project2.DATA;

log\_dw = log(device\_width);

log\_res = log(resolution);

log\_dv = log(device\_volume);

log\_dh = log(device\_height);

**run**;

/\*output 7 \*/

**proc** **surveyselect** data=ad\_logged out=ad\_smp\_logged outall samprate=**0.6** seed=**76**;

**run**;

**data** ad\_train\_logged ad\_test\_logged;

set ad\_smp\_logged;

if selected then output ad\_train\_logged;

else output ad\_test\_logged;

**run**;

**proc** **glmmod** data=ad\_smp\_logged outdesign=ad\_outputlog noprint;

class device\_platform\_class;

model install = publisher\_id\_class device\_make\_class device\_platform\_class device\_os\_class device\_height device\_width resolution device\_volume wifi

log\_dw log\_dv log\_res log\_dh;

weight selected;

**run**;

/\*output 8,9 \*/

**proc** **contents** data=ad\_outputlog;

**run**;

/\*output 10,11 \*/

**proc** **reg** data=ad\_outputlog;

Withoutlog : model install = col2 - col11;

Logmodel : model install = col2- col3 col6-col8;

weight selected;

output out=ad\_problog pred=p;

**quit**;

/\* Log model - not required \*/

/\*Though we have decided to use the non-log model, to choose the best predictors we use the ITERATIVE APPROACH\*/

/\* ITERATIVE APPROACHES \*/

/\* Trial- Stepwise Selection\*/

/\*output 12 \*/

**proc** **glmselect** data=ad\_train\_ind plots=all;

model install = col1- col10/selection=stepwise(select=sl) stats=all showpvalues;

**run**;

/\*output 13,14 \*/

**proc** **contents** data=ad\_train\_ind ;

**run**;

/\* After all these trials looking at ASE value we still decide to use the 8 predictors from trial 3 in our final model \*/

/\* Final Model: Linear probability model using PROC reg \*/

/\*output 19,20 \*/

**proc** **reg** data=ad\_train\_ind plots(maxpoints=**100000**);

model install = col2- col3 col6-col8 ;

**quit**;

/\* EVEN THOUGH THE PREDICTED PROBABILITIES ARE BETWEEN 0 AND 1, RESIDUALS MAY NOT BE NORMAL(NEED PLOTS TO PROVE THIS) \*/

/\* Since the assumption is violated, the std error estimates will be wrong and hence cannot decide on the signifcance of a predictor \*/

/\* based on p values. Also a unit change in X does not have the same impact on probability \*/

/\* Logistic Regression \*/

/\* Initial model - Logistic Regression \*/

/\*output 21,22,23 \*/

**proc** **logistic** data=ad\_train\_ind;

logit: model install (event='1') = col1 - col10;

/\* OUTPUT OUT=LPM1 pred=p ;\*/

**run**;

/\* Trial 2 - Stepwise Selection\*/

/\*output 21,22,23 \*/

**proc** **logistic** data=ad\_train\_ind outest=ans covout;

logit: model install (event='1') = col1 - col10

/ selection=stepwise

slentry=**0.25**

slstay=**0.35**

details

lackfit;

**run**;

/\* Based on these 4 trials we see that for trials 2,3,4 forward, backward and stepwise selection method the -2 log L values is 7854.786 for all 3 models.\*/

/\* However, this is higher than the intial model where the value is 7854.596. Hence we choose the final model with predictors from trial 2 \*/

/\* Thus the final predictors are col1 - col 8 \*/

/\* Final model - Logistic Regression \*/

/\* (i) Estimation of the model without considering rare events\*/

/\*output 24,25 \*/

**proc** **logistic** data=ad\_smp;

class device\_platform\_class;

logit: model install (event='1') = device\_volume wifi resolution device\_height device\_width publisher\_id\_class device\_os\_class device\_make\_class device\_platform\_class;

/\* OUTPUT OUT=LPM1 pred=p ;\*/

**run**;

/\* We actually do not need to consider modeling of rare events in this case as the No. of rare event(event =1) = 680 in absolute number \*/

/\*which is considerably high. Also as per rule of thumb 20 events per independent variable is required and for this model there are 10 independent \*/

/\* variables. Hence 10\*20 =200< 680, thus modeling of rare events not reqd \*/

/\*(ii)b. Estimate the model using oversampling approach for handling rare events and then applying the correction to obtain the corrected intercept \*/

/\*output 26 \*/

**proc** **freq** data=ad\_smp;

table install / out=fullpct(where=(install=**1**) rename=(percent=fullpct));

title "response counts in full data set";

**run**;

**data** sub;

set ad\_smp;

if install =**1** or (install =**0** and ranuni(**75302**)<**1**/**119**) then output;

**run**;

/\*output 27 \*/

**proc** **freq** data=sub;

table install / out=subpct(where=(install =**1**) rename=(percent=subpct));

title "Response counts in oversmp, subset data set";

**run**;

**data** sub;

set sub;

if \_n\_=**1** then set fullpct(keep=fullpct);

if \_n\_=**1** then set subpct(keep=subpct);

p1=fullpct/**100**; r1=subpct/**100**;

w=p1/r1; if install =**0** then w=(**1**-p1)/(**1**-r1);

off=log( (r1\*(**1**-p1)) / ((**1**-r1)\*p1) );

**run**;

/\*output 28,29 \*/

**proc** **logistic** data=sub;

class device\_platform\_class;

model install (event="1")=device\_volume wifi resolution device\_height device\_width publisher\_id\_class device\_os\_class device\_make\_class device\_platform\_class;

output out=out p=pnowt;

title "True Parameters: -8.1248 (intercept)";

title2 "Unadjusted Model";

**run**;

/\*output 30 \*/

**proc** **logistic** data=out;

class device\_platform\_class;

model install (event="1")=device\_volume wifi resolution device\_height device\_width publisher\_id\_class device\_os\_class device\_make\_class device\_platform\_class;

weight w;

output out=out p=pwt;

title2 "Weight-adjusted Model";

**run**;

/\*output 31 \*/

**proc** **logistic** data=out;

class device\_platform\_class;

model install (event="1")=device\_volume wifi resolution device\_height device\_width publisher\_id\_class device\_os\_class device\_make\_class device\_platform\_class / offset=off;

output out=out xbeta=xboff;

title2 "Offset-adjusted Model";

**run**;

/\* Classification and ROC \*/

/\* Initial Model - Linear probability model \*/

/\* Making predictions for test observations \*/

/\*output 32 \*/

**proc** **reg** data=LPM\_ind PLOTS(MAXPOINTS=**100000**);

linear: model install = col1 - col10 ;

output out=ad\_linear\_initial p=linear\_predictions;

weight selected;

**quit**;

/\* To plot ROC curve based on predictions from linear model \*/

/\*output 33 \*/

**proc** **logistic** data=ad\_linear\_initial plots=roc(id=prob);

model install (event='1') = col1 - col10/ nofit;

roc pred=linear\_predictions;

where selected=**0**;

**run**;

/\* Final Model: Linear probability model using PROC reg \*/

/\* Making predictions for test observations \*/

**proc** **reg** data=LPM\_ind PLOTS(MAXPOINTS=**100000**);

linear: model install = col1 - col8 ;

output out=ad\_linear\_final p=linear\_predictions;

weight selected;

**quit**;

/\* To plot ROC curve based on predictions from linear model \*/

/\*output 34 \*/

**proc** **logistic** data=ad\_linear\_final plots=roc(id=prob);

model install (event='1') = col1 - col8/ nofit;

roc pred=linear\_predictions;

where selected=**0**;

**run**;

/\*Initial Model- Logistic regression \*/

/\* Make predictions on test data \*/

**proc** **logistic** data=ad\_train\_ind;

logit: model install (event='1') = col1 - col10;

score data=ad\_test\_ind out=ad\_logit\_predict\_initial;

**run**;

/\*ROC curve on test data \*/

/\*output 35 \*/

**proc** **logistic** data=ad\_logit\_predict\_initial plots=roc(id=prob);

model install (event='1') = col1 - col10/ nofit;

roc pred=p\_1;

**run**;

/\* Final model - Logistic Regression \*/

/\* Estimation of the model without considering rare events\*/

**proc** **logistic** data=ad\_train\_ind;

logit: model install (event='1') = col1 - col8;

score data=ad\_test\_ind out=ad\_logit\_predfin;

**run**;

/\*ROC curve on test data \*/

/\*output 36 \*/

**proc** **logistic** data=ad\_logit\_predfin plots=roc(id=prob);

model install (event='1') = col1 - col8/ nofit;

roc pred=p\_1;

**run**;

/\* PART 2 \*/

/\* (i) ROC Table for Initial Logistic Regression Model \*/

/\*output 37 \*/

**proc** **logistic** data=ad\_train\_ind outmodel=train\_roc\_1;

logit: model install (event='1') = col1 - col10;

**run**;

**proc** **logistic** inmodel=train\_roc\_1;

score data=ad\_test\_ind outroc=logged\_roc\_2;

**run**;

/\* Finding total cost for each probability threshold based on misclassification \*/

**data** ad\_threshold1;

set logged\_roc\_2;

total\_cost1 = \_FALPOS\_\***0.01** + \_FALNEG\_\***1**;

**run**;

/\*output 36 \*/

**proc** **sql**;

create table total\_cost1 as(select\*,min(total\_cost1) as min\_cost from ad\_threshold1);

**run**;

/\* (ii) ROC Table for Final Logistic Regression Model \*/

/\*output 38 \*/

**proc** **logistic** data=ad\_train\_ind outmodel=train\_roc\_2;

logit: model install (event='1') = col1 - col8;

**run**;

**proc** **logistic** inmodel=train\_roc\_2;

score data=ad\_test\_ind outroc=logged\_roc\_2;

**run**;

/\* Finding total cost for each probability threshold based on misclassification \*/

**data** ad\_threshold2;

set logged\_roc\_2;

total\_cost2 = \_FALPOS\_\***0.01** + \_FALNEG\_\***1**;

**run**;

**proc** **sql**;

create table total\_cost2 as(select\*,min(total\_cost2) as min\_cost from ad\_threshold2);

**run**;

/\* (ii) ROC Table for Initial Linear Probability Model \*/

**data** tab\_a (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.001** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_a as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from tab\_a;

**quit**;

**data** table\_b (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.005** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_b as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_b;

**quit**;

**data** table\_c (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.010** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_c as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_c;

**quit**;

**data** table\_d (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.015** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_d as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_d;

**quit**;

**data** table\_e (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.020** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_e as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_e;

**quit**;

**data** table\_f (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.025** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_f as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_f;

**quit**;

**data** table\_g (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.030** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_g as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_g;

**quit**;

**data** table\_h (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.035** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_h as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_h;

**quit**;

**data** table\_i (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.040** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_i as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_i;

**quit**;

**data** table\_j (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.045** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_j as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_j;

**quit**;

**data** table\_k (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.050** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_k as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_k;

**quit**;

**data** initial\_linear\_roc\_temp;

input probability false\_positive false\_negative;

datalines;

0.001 36061 0

0.005 32438 0

0.010 7611 0

0.015 125 0

0.020 0 0

0.025 0 0

0.030 0 0

0.035 0 0

0.040 0 0

0.045 0 0

0.050 0 0

;

**run**;

**data** initial\_linear\_roc;

set initial\_linear\_roc\_temp;

total\_cost = false\_positive\***0.01** + false\_negative\***1**;

**run**;

/\* (ii) ROC Table for Final Linear Probability Model \*/

**data** tab\_a1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_final;

where selected=**0**;

if linear\_predictions > **0.001** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_a1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from tab\_a1;

**quit**;

**data** table\_b1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.005** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_b1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_b1;

**quit**;

**data** table\_c1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.010** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_c1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_c1;

**quit**;

**data** table\_d1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.015** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_d1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_d1;

**quit**;

**data** table\_e1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.020** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_e1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_e1;

**quit**;

**data** table\_f1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.025** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table cnt\_f1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_f1;

**quit**;

**data** table\_g1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.030** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_g1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_g1;

**quit**;

**data** table\_h1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.035** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_h1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_h1;

**quit**;

**data** table\_i1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.040** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_i1 as select\*,count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_i1;

**quit**;

**data** table\_j1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.045** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_j1 as select count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_j1;

**quit**;

**data** table\_k1 (keep=install selected linear\_predictions predicted false\_pos false\_neg);

set ad\_linear\_initial;

where selected=**0**;

if linear\_predictions > **0.050** then predicted=**1**;

if install=**0** and predicted=**1** then false\_pos=**1**;

if install=**1** and predicted=**0** then false\_neg=**1**;

**run**;

**proc** **sql**;

create table count\_k1 as select count(false\_pos) as cnt\_fp, count(false\_neg) as cnt\_fn from table\_k1;

**quit**;

**data** final\_temp;

input probability false\_positive false\_negative;

datalines;

0.001 36067 0

0.005 32438 0

0.010 7611 0

0.015 125 0

0.020 0 0

0.025 0 0

0.030 0 0

0.035 0 0

0.040 0 0

0.045 0 0

0.050 0 0

;

**run**;

**data** final\_linear\_roc;

set final\_temp;

total\_cost = false\_positive\***0.01** + false\_negative\***1**;

**run**;