Spring 2018

Mini Project 2

Group 2

Hanson Wu

Medha Tiwary

Animesh Kansal

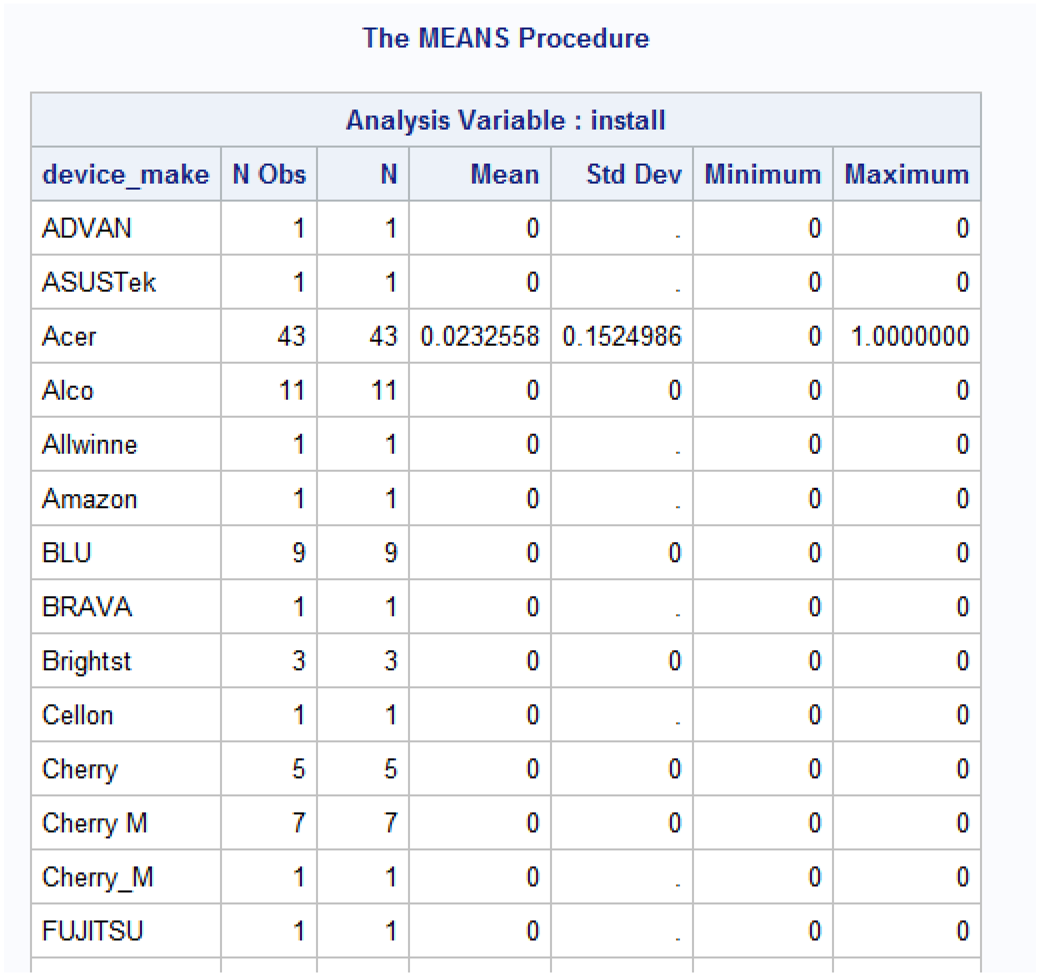
Jaahnavi Tiruthani

Windows User

[COMPANY NAME]

Question 1

First, we noticed that there was an extremely small amount of installations compared to total installations (less than 1%). In order to help alleviate this, we needed to clean the data. We used a PROC MEANS function on the variables publisher_id, device_make, and device_os to determine the average install rate for these variables. We noticed many variable categories had a value of 0 for the mean install and that there were not many observations for most of these categories (less than 20). An example for the variable device_make is shown below. The mean is 0 for most of the different device_make categories and often with fewer than 20 observations.



The MEANS Procedure

Analysis Variable : install						
device_make	N Obs	N	Mean	Std Dev	Minimum	Maximum
ADVAN	1	1	0	.	0	0
ASUSTek	1	1	0	.	0	0
Acer	43	43	0.0232558	0.1524986	0	1.0000000
Alco	11	11	0	0	0	0
Allwinne	1	1	0	.	0	0
Amazon	1	1	0	.	0	0
BLU	9	9	0	0	0	0
BRAVA	1	1	0	.	0	0
Brightst	3	3	0	0	0	0
Cellon	1	1	0	.	0	0
Cherry	5	5	0	0	0	0
Cherry M	7	7	0	0	0	0
Cherry_M	1	1	0	.	0	0
FUJITSU	1	1	0	.	0	0

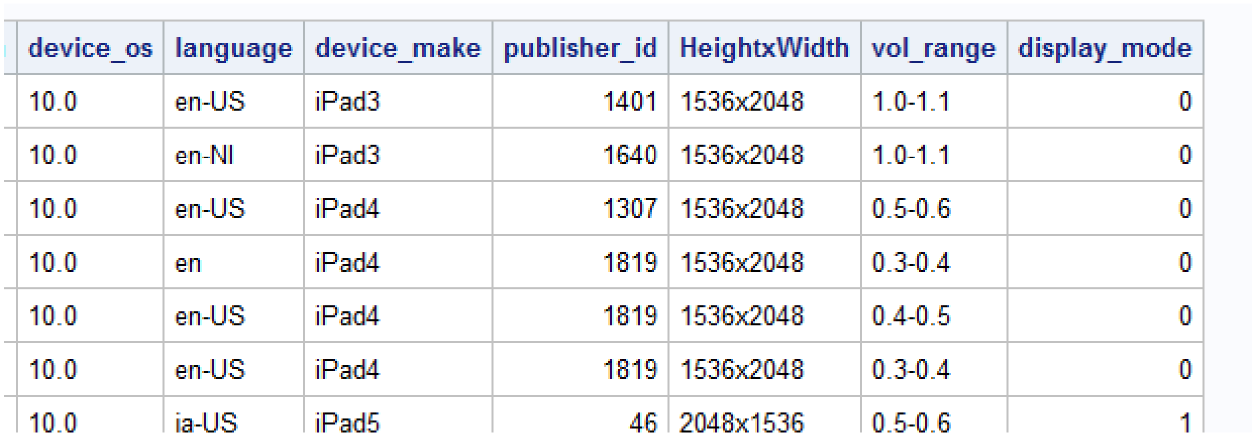
We wanted to make a model where the non-important categories in publisher_id, device_make, and device_os don't interfere so we had to drop rows with publishers, makes, and operating systems that weren't relevant. We did this by first merging the tables created from the PROC MEANS procedure with our training dataset. Then we created a new dataset using the WHERE clause to drop the row if the average install of the publisher_id, device_make, or device_os was 0. An example is shown below:

```
DATA new1;
SET new (WHERE=(pubinst>0));
RUN;

/*72426 rows left after initial data cleansing*/

/*12512 rows removed*/
```

We added two new variables to our resulting dataset, vol_range (volume range) and display_mode (portrait or landscape). We categorized the device_volume into 10 evenly-spaced bins to make our visualizations easier but device_volume was not used in the model. We created display_mode to represent a combination of the device_height and device_width variables. If device_height > device_width than a 1 was assigned (portrait mode) and if device_height<device_width than a 0 was assigned (landscape mode). We also added these variables to the testing dataset.



device_os	language	device_make	publisher_id	HeightxWidth	vol_range	display_mode
10.0	en-US	iPad3	1401	1536x2048	1.0-1.1	0
10.0	en-NI	iPad3	1640	1536x2048	1.0-1.1	0
10.0	en-US	iPad4	1307	1536x2048	0.5-0.6	0
10.0	en	iPad4	1819	1536x2048	0.3-0.4	0
10.0	en-US	iPad4	1819	1536x2048	0.4-0.5	0
10.0	en-US	iPad4	1819	1536x2048	0.3-0.4	0
10.0	ia-US	iPad5	46	2048x1536	0.5-0.6	1

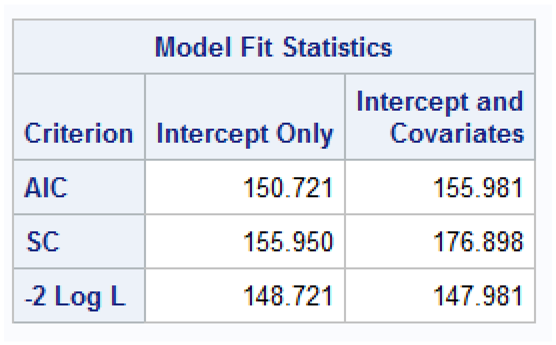
Our data is still extremely biased which would cause our model to categorize everything into the 'no install' so we needed to under-sample the data. We wanted a roughly 50/50 split of installs to non-installs so we implemented the code below:

```
data usd; /*usd = undersampled data*/
set newtrain; /*dataset created from previous steps*/
if install=1 or (install=0 and ranuni(72426)<1/105) then output;
run; /*1379 rows .50181 rate of install*/
/*making weighted undersampled data*/
data wusd;
set usd;
 w=0.00955/0.50181; if install=0 then w=(1-0.00955)/(1-0.50181);
run;
```

We kept all of the rows with installs (692 rows) and took a roughly equal amount of random non-installs. 672 is roughly 1/105th of 72426 which is the number of rows in our updated training dataset. We then added a weight variable w to our data. 0.955% of our updated training dataset was installs and we divided that by the ratio of installs:non-installs in our under-sampled dataset which was 0.50181. Using our weighted under-sampled data, we ran the following model:

```
proc logistic data=wusd;
CLASS publisher_id device_make device_platform device_os  display_mode
publisher_id:device_make;
 logit: MODEL install (EVENT='1') =  resolution wifi device_volume;
 weight w;
 score data=newtest out=test_logit_predict;
run;
```

We left out device_height and device_width as we felt those variables were well-represented in the resolution and the new display_mode variables. We included an interaction variable publisher_id:device_make because we decided that the combination of what type of phone a person is using as well as what type of app they are on depend on each other. This interaction variable also improved our ROC curve. We tested many other interaction variables but they did not improve our model or did not make sense as interaction variables. We obtained the following AIC and SC values:

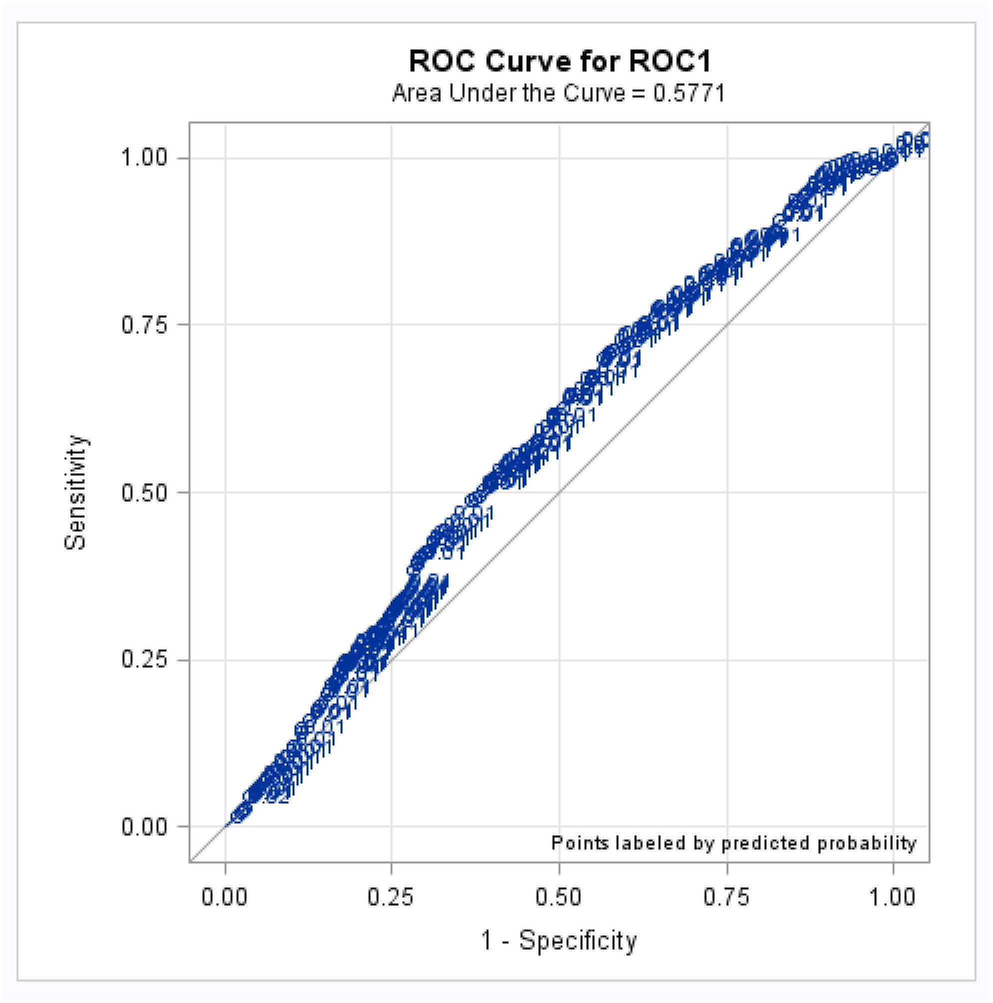


Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	150.721	155.981
SC	155.950	176.898
-2 Log L	148.721	147.981

We then used the following code to produce our ROC curve

```
proc logistic data=test_logit_predict plots=roc(id=prob);
CLASS publisher_id device_make device_platform device_os display_mode
publisher_id:device_make;
 MODEL install (EVENT='1') =  resolution wifi device_volume/ nofit;
 roc pred=p_1;
run;
```

The area under the ROC curve (shown on next page) was 0.5771.



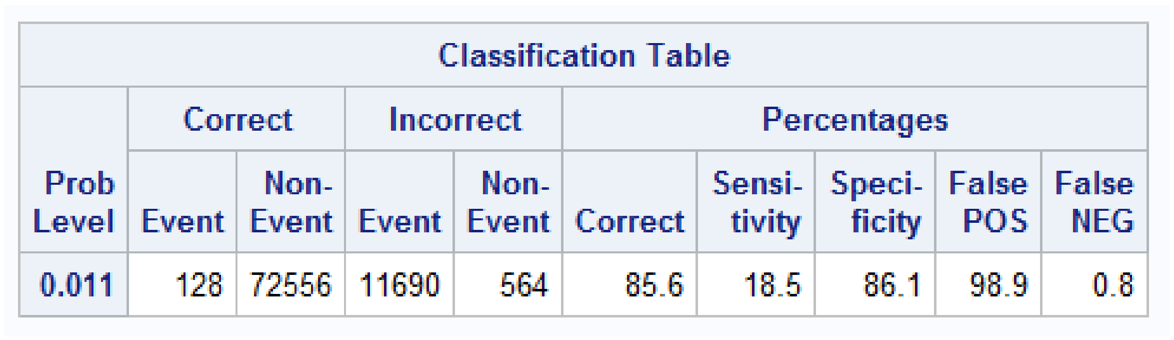
This was determined to be our best model. We also did PROC LOGISTIC models with the original training dataset, the updated training dataset, oversampled data from the original dataset, under sampled data from the original dataset, and oversampled data from the new training dataset (all included in the code) but the AIC and SC values as well as the ROC were not satisfactory.

Question 2

Our idea behind the classification schemes was to use our model and then adjust the pprob value so that SAS would categorize the values so that the ratio of errors would be as close as possible to the desired ratios. For the ratio of 25:1, we used the pprob value of 0.011

```
proc logistic data=newtrain1;
CLASS publisher_id device_make device_platform device_os  display_mode
publisher_id:device_make;
```

```
 logit: MODEL install (EVENT='1') =  resolution wifi device_volume / ctable
pprob=(0.011);
run;
```

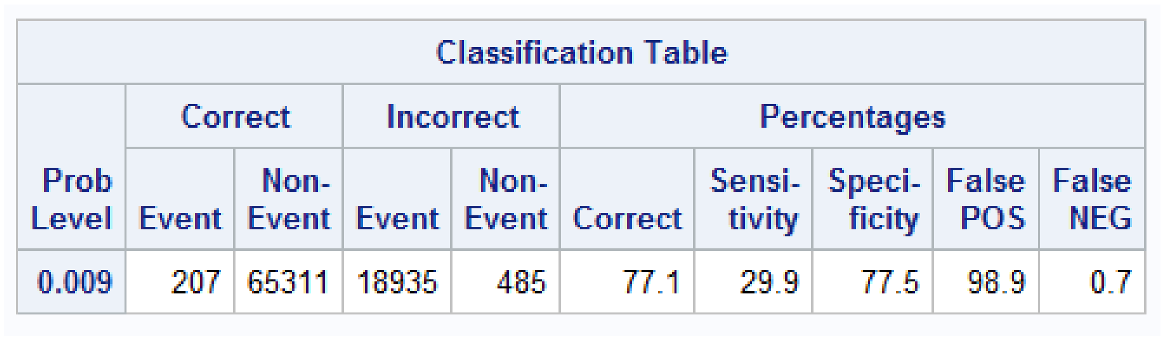


Classification Table									
	Correct		Incorrect		Percentages				
Prob Level	Event	Non-Event	Event	Non-Event	Correct	Sensi-tivity	Speci-ficity	False POS	False NEG
0.011	128	72556	11690	564	85.6	18.5	86.1	98.9	0.8

This created 11690 C1 errors and 564 C2 errors which is a ratio of about 20.7:1. This was as close as we could get because the true value of pprob is between 0.011 and 0.010.

For the ratio of 50:1, we used the following code and a pprob value of 0.0085

```
proc logistic data=newtrain1;
CLASS publisher_id device_make device_platform device_os  display_mode
publisher_id:device_make;
 logit: MODEL install (EVENT='1') =  resolution wifi device_volume / ctable
pprob=(0.0085);
run;
```

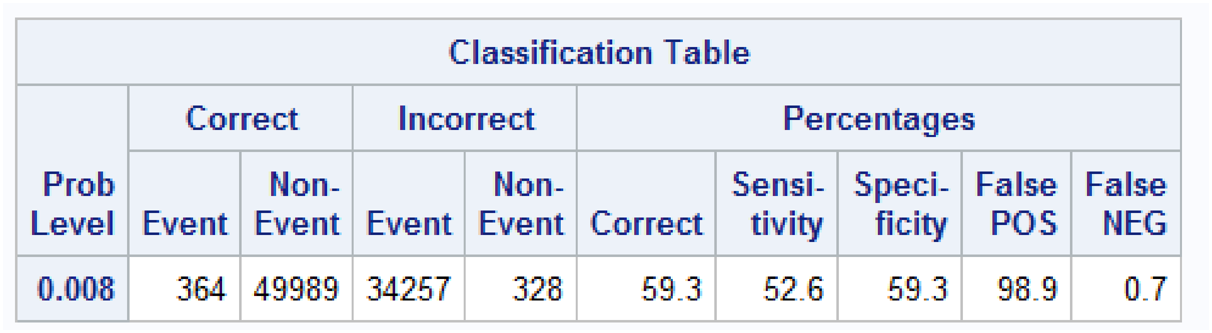


Classification Table									
	Correct		Incorrect		Percentages				
Prob Level	Event	Non-Event	Event	Non-Event	Correct	Sensi-tivity	Speci-ficity	False POS	False NEG
0.009	207	65311	18935	485	77.1	29.9	77.5	98.9	0.7

This created 18935 C1 errors and 485 C2 errors which is a ratio of about 39:1.

For the ratio of 100:1, we used the following code with a pprob value of 0.008:

```
proc logistic data=newtrain1;
CLASS publisher_id device_make device_platform device_os  display_mode
publisher_id:device_make;
 logit: MODEL install (EVENT='1') =  resolution wifi device_volume / ctable
pprob=(0.008);
run;
```

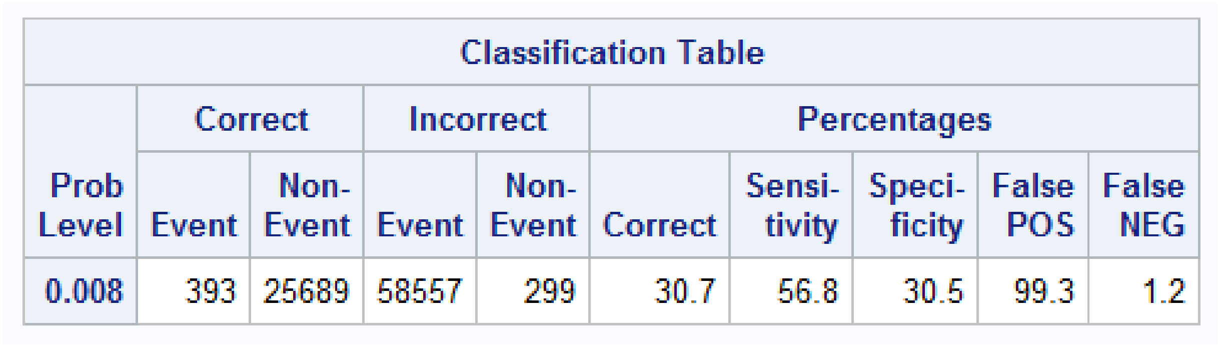


Classification Table									
	Correct		Incorrect		Percentages				
Prob Level	Event	Non-Event	Event	Non-Event	Correct	Sensi-tivity	Speci-ficity	False POS	False NEG
0.008	364	49989	34257	328	59.3	52.6	59.3	98.9	0.7

There were 34257 C1 errors and 328 C2 errors which is a ratio of about 104:1.

For the ratio 200, we used the following code with a pprob value of

```
proc logistic data=newtrain1;
CLASS publisher_id device_make device_platform device_os  display_mode
publisher_id:device_make;
 logit: MODEL install (EVENT='1') =  resolution wifi device_volume / ctable
pprob=(0.00788);
run;
```



Classification Table									
	Correct		Incorrect		Percentages				
Prob Level	Event	Non-Event	Event	Non-Event	Correct	Sensi-tivity	Speci-ficity	False POS	False NEG
0.008	393	25689	58557	299	30.7	56.8	30.5	99.3	1.2

There were 58557 C1 errors and 299 C2 errors which is a ratio of about 195.8:1. However, the following code changes the ratios a bit but keeps the C2 errors the same while decreasing the C1 errors which reduces overall cost.

```
proc logistic data=newtrain1;
CLASS publisher_id device_make device_platform device_os  display_mode
publisher_id:device_make;
 logit: MODEL install (EVENT='1') =  resolution wifi device_volume / ctable
pprob=(0.0079);
run;
```

Classification Table

	Correct		Incorrect		Percentages				
Prob Level	Event	Non-Event	Event	Non-Event	Correct	Sensi-tivity	Speci-ficity	False POS	False NEG
0.008	393	44999	39247	299	53.4	56.8	53.4	99.0	0.7