CATALOG ORDERS

CONSUMER BEHAVIOR

Badhrinathan, Saundarya

MARKETING PREDICTIVE ANALYTICS WITH SAS

Data Extraction and Transformation

Twelve years of catalog company order data consisting of 14,448 orders across 10,000 households. Distribution methods include current customers, affiliate's customers, and web origination customers.

Data was imported utilizing SAS PROC IMPORT with Dates and Numerical variables stored in their respective formats.

PROC MEANS and PROC UNIVARIATE is used to verify the distribution of data and any missing data.

- Zip code is missing in 5 households:
  761420276, 768112578, 768112860, 1108288688, 1108831396

GROSS_PRODUCT_REVENUE_AMOUNT

Description :
The Gross Product Revenue (GPR) is the total amount purchased on this order.

As of 1/25/2007:
This field will be changed to include all shipping, handling and tax amounts. Nothing will be subtracted from the GPR. Any returns, cancels, coupons, etc. will have to be subtracted manually.

Pre-2007:
The Gross Product Revenue (GPR) is the total amount purchased on this order. This total does not include shipping/handling and taxes. Coupon amounts and additional charges (to the company) are NOT removed from GPR - these charges must be removed manually if needed. The return and cancel amounts are subtracted from the GPR.

GROSS_PRODUCT REVENUE_AMOUNT is stored consistently across the time frame of the data. Before doing further analysis we should make sure that it is consistent with the latest method in which data is stored.

```
DATA catalog_date_consistent;
SET catalog_consistent;
IF input_date < '25JAN2007'd THEN
gross_product_revenue_amount = gross_product_revenue_amount +
shipping_and_handing_amount + sales_tax_amount +
cancel_amount + returned_amount ;
ELSE
gross_product_revenue_amount = gross_product_revenue_amount;
RUN;
```

OFFER_CODE

Description :
A 3-digit code associated with an offer. There was logic to the coding prior to Fall 2005. Currently, there is no specific logic to the code.

GIFT_CERTIFICATE_REDEEMED_IND

Description :
Indicates whether or not a gift certificate was used to pay for this purchase.
Y - Gift certificate used to pay for purchase
<blank> - Gift certificate not used

Need to handle blank value as 'N'

```
DATA catalog_consistent;
SET catalog;
if GIFT_CERTIFICATE_REDEEMED_IND = ' ' then
GIFT_CERTIFICATE_REDEEMED_IND = 'N';
RUN;
```

COUPON_REDEEMED_INDICATOR

Description :
Denotes whether or not a coupon was redeemed on this order. Coupons are usually issued for irate customers.
Y - Coupon redeemed
<blank> - No coupon used

Need to handle blank value as 'N'

```
DATA catalog_consistent;
SET catalog_consistent;
if COUPON_REDEEMED_INDICATOR = ' ' then
COUPON_REDEEMED_INDICATOR = 'N';
RUN;
```

PAYMENT_CATEGORY_CODE

Description :
The payment type.
1 - Cash/Check
2 - Credit Card
3 - A/R
4 – (not used)
5 - Coupon/Credit
<blank> - suspended

Need to handle blank value.

```
DATA catalog_consistent;
SET catalog_consistent;
if PAYMENT_CATEGORY_CODE = ' ' then
PAYMENT_CATEGORY_CODE = 'S';
            RUN;
```

Marketing Opportunity Exploration

1. Customer segmentation Analysis:

   Perform K-means to get customer segments:
   a) Get the data down to a customer or in this case household level
   b) Variables to be used in K-means
      i. No.of orders
      ii. Total Cost of orders
      iii. No. of print orders
      iv. No. of web orders
      v. No. of orders with offer/promotions/coupons/gift certificate

   Segments are categorized like:
   a) Big spenders
   b) Small/single/one time customers
   c) Discount /promotion driven customers
   d) Repeat customers
   e) Web customers
   f) Print customers

The segments formed can be used to do targeted marketing.

2. Geo Analysis:
   a) Get list of zip-codes by state
   b) Add a new column to the orders called state
   c) Calculate the ratios for web/catalogue orders
   d) Across time see how the ratio of overall national demand has changed across states
   e) Calculate top 5 and bottom 5 states – can be used for overall company strategies

Analysis:

The top 5 states are California, New York, Pennsylvania, Florida and Texas.
We should concentrate on increasing marketing efforts in these states.

The bottom 5 states are North Dakota, South Dakota, District of C, Wyoming, Montana.
We should think about minimising efforts here or look into why orders are low in these states.

We also see there is a growing trend for orders from web catalogs over the years.
Therefore, we should concentrate on increasing web traffic to maximise orders.

3. Payment analysis:

a) Over time show how the ratio of credit, cash, cheques have changed
b) Can show which states prefer what kind of payment
c) Can show which type of customers prefer what kind of payment

4. Miscellaneous analysis:
    a) How have returns changed over time?
       Used normalized returns value and get no. of returns/total orders that year.
       Used returns as a % of total orders to compare across years else there will be steady growth in no. of returns, since growth in orders is expected
    b) Top 5 zips when it comes to returns in each state
    c) Top 10 customers per state with max returns

    We find that returns from Hawaii are more compared to returns from other states. We should find the reason behind these returns to avoid future losses. We find that a few zipcodes from states like Texas, New Jersey and Illinois have high returns. These might belong to the poorer neighborhoods and we should try to cut losses by taking more stringent measures.

Conclusion

The analysis was conducted to find segments of customers in order to understand their underlying needs and find the right product positioning strategy for them. We should try to understand why there aren't a lot of repeat customers.

The total orders by state tells us which states to concentrate on and to maximize our marketing efforts on and the platform of concentration seems to be headed towards web traffic in the recent years.

We should also try to find out reasons for returns and minimize losses.

Appendix

```
DATA catalog;
INFILE "Data Set 8\Catalog-Data.txt";
INPUT id 1-10 order_number 11-24 input_date MMDDYY10. subscription_indicator $ 35-35
subscription_quantity 36-42 catalog_item_indicator $ 43-43 catalog_item_quantity 44-50
number_of_recepients_qty 51-64 gross_product_revenue_amount 8.2
shipping_and_handling_amount 8.2 sales_tax_amount 81-88 cancel_amount 89-96
returned_amount 97-104 accounts_receivable_amount 105-112 refund_date MMDDYY10.
refund_amount 123-130 refund_status_code $ 131-131 refund_type_code $ 132-132
refund_reason_code $ 133-133 offer_type_code $ 134-134 offer_drop_code $ 135-135
offer_code $ 136-138 gift_certificate_redeemed_ind $ 139-139 gift_certificate_amount
140-147
coupon_redeemed_indicator $ 148-148 coupon_amount 149-156 gift_indicator $ 157-157
payment_type_code 158-161 payment_category_code $ 162-162 payment_status_code $
163-163
order_type_code $ 164-164 ancillary_item_indicator $ 165-165 ancillary_item_quantity
166-179
additional_charges_amount 180-187 additional_charge_code $ 188-189 web_item_indicator $
190-190
web_item_quantity 191-197 write_off_amount 198-205 division_code $ 206-207
individual_id 208-234 rank 235-241 zip 242-246;
FORMAT input_date MMDDYY10. refund_date MMDDYY10.;
RUN;
```

```
PROC IMPORT DATAFILE="US 2000 Census Data\Household Income Distribution.csv"
OUT=HH_Income
DBMS=csv
replace;
GETNAMES=yes;
RUN;
```

```
PROC IMPORT DATAFILE="US 2000 Census Data\Household Type by Householder Age and
Family Type.csv"
OUT=HH_Type
DBMS=csv
replace;
GETNAMES=yes;
RUN;
```

```
PROC IMPORT DATAFILE="US 2000 Census Data\Population by Age and Gender.csv"
OUT=Population
DBMS=csv
replace;
GETNAMES=yes;
RUN;


PROC IMPORT DATAFILE="US 2000 Census Data\Urban Rural Housing Units.csv"
OUT=Urban_Rural
DBMS=csv
replace;
GETNAMES=yes;
RUN;


PROC IMPORT DATAFILE="us_postal_codes.csv"
OUT=Postal_Code
DBMS=csv
replace;
GETNAMES=yes;
RUN;


DATA Postal_Code;
SET Postal_Code(rename=(postal_code=zip));
RUN;


DATA catalog_consistent;
SET catalog;
if GIFT_CERTIFICATE_REDEEMED_IND = ' ' then
GIFT_CERTIFICATE_REDEEMED_IND = 'N';
else GIFT_CERTIFICATE_REDEEMED_IND = GIFT_CERTIFICATE_REDEEMED_IND;
RUN;


DATA catalog_consistent;
SET catalog_consistent;
if COUPON_REDEEMED_INDICATOR = ' ' then
COUPON_REDEEMED_INDICATOR = 'N';
else COUPON_REDEEMED_INDICATOR = COUPON_REDEEMED_INDICATOR;
```

```
RUN;


DATA catalog_consistent;
SET catalog_consistent;
if PAYMENT_CATEGORY_CODE = ' ' then
PAYMENT_CATEGORY_CODE = '6';
else PAYMENT_CATEGORY_CODE = PAYMENT_CATEGORY_CODE;
RUN;


DATA catalog_date_consistent;
SET catalog_consistent;
IF input_date < '25JAN2007'd THEN
gross_product_revenue_amount = gross_product_revenue_amount +
shipping_and_handling_amount + sales_tax_amount +
cancel_amount + returned_amount ;
ELSE
gross_product_revenue_amount = gross_product_revenue_amount;
RUN;


DATA catalog_clean(DROP=write_off_amount INDIVIDUAL_ID
ACCOUNTS_RECEIVABLE_AMOUNT);
SET catalog_date_consistent;
RUN;


PROC GLM DATA=catalog_clean;
CLASS subscription_indicator catalog_item_indicator gift_certificate_redeemed_ind
additional_charge_code web_item_indicator;
MODEL gross_product_revenue_amount = subscription_indicator catalog_item_indicator
gift_certificate_redeemed_ind
additional_charge_code web_item_indicator
subscription_quantity catalog_item_quantity
cancel_amount returned_amount;
RUN;


PROC SORT DATA=catalog_clean;
BY zip;
RUN;
```

```
PROC SORT DATA=Postal_Code;
BY zip;
RUN;


DATA catalog_state;
MERGE catalog_clean Postal_Code;
BY zip;
RUN;


PROC SQL;
CREATE TABLE catalog_state AS
SELECT a.*, b.state
FROM catalog_clean a LEFT JOIN Postal_Code b
ON a.zip=b.zip;
QUIT;


DATA refund_order;
SET catalog_clean;
IF refund_date ne . then output;
RUN;


* average refund amount for each reason code;

PROC PRINT DATA=refund_order;
RUN;


PROC SQL;
CREATE TABLE refund_reason AS
SELECT refund_reason_code, avg(refund_amount) AS avg_refund_amount,
count(refund_reason_code) AS No_orders_refunded
FROM refund_order GROUP BY refund_reason_code;
QUIT;

PROC PRINT DATA=refund_reason;
RUN;
```

```
/*
B - balance due - 205
O - overpayment - 246
As these reasons can not be controlled by company, we analysis only R- return - 639
AVG refund amount for 'R - Return' is more
we will try to see how this refund is taken place*/

PROC SQL;
CREATE TABLE refund_type AS
SELECT refund_type_code, avg(refund_amount) AS avg_refund_amount,
count(refund_type_code) AS No_orders_refunded
FROM refund_order
WHERE refund_reason_code = 'R'
GROUP BY refund_type_code;
QUIT;

PROC PRINT DATA=refund_type;
RUN;

/* most of the refund amount is in terms of credit card credit c - 606 orders*/



/* Insight 1*/

PROC SQL;
CREATE TABLE per_household AS
SELECT count(order_number) AS number_of_orders,
sum(gross_product_revenue_amount) AS Total_Cost_of_orders, SUM(catalog_item_quantity)
AS total_catalog_quantity, sum(web_item_quantity) AS total_web_quantity,
sum(gift_certificate_amount) AS total_giftcertificate_redeemed
FROM catalog_clean
GROUP BY id;
QUIT;
PROC PRINT DATA=per_household;
RUN;



/* hpclus is ran to determine number of clusters in the data */

proc hpclus
data= per_household
outstat=k_means_cluster
maxclusters= 10
maxiter= 100
seed= 54321                    /* set seed for pseudo-random number generator */
```

```
NOC= ABC(B= 1 minclusters= 3 align= PCA);  /* select best k between 3 and 8 using ABC */
score out= OutScore;
input number_of_orders Total_Cost_of_orders total_catalog_quantity total_web_quantity
total_giftcertificate_redeemed;                                    /* input variables */
ods output ABCStats= ABC;                          /* save ABC criterion values for plotting */
run;
                        /* clustering output indicated that 7 clusters is most appropriate */

proc print data=k_means_cluster;
run;



proc sgplot
data= ABC;
scatter x= K y= Gap / markerattrs= (color= 'STPK' symbol= 'circleFilled');
xaxis grid integer values= (3 to 10 by 1);
yaxis label= 'ABC Value';
run;



/* fastclus is used with 5 clusters as preiously identified*/
proc fastclus
data= per_household
out= cluster_final
maxiter= 100
converge= 0 /* run to complete convergence */
radius= 100 /* look for initial centroids that are far apart */
maxclusters= 7
summary;
run;

PROC PRINT DATA=cluster_final;
RUN;



/* Scatter plot observe various clusters in the data
proc sgplot data=cluster_final;
scatter y= Total_Cost_of_orders x=number_of_orders / group=CLUSTER ;
run;
*/
PROC SORT DATA=cluster_final;
BY CLUSTER;
RUN;
```

```
PROC PRINT DATA=cluster_final;
RUN;


/* Insight 2 */


DATA catalog_state_clean;
SET catalog_state;
IF zip=. THEN DELETE;
RUN;



PROC SQL;
CREATE TABLE catalog_orders_state AS
SELECT state,count(catalog_item_indicator) AS total_catalog_orders,
SUM(catalog_item_quantity) AS total_catalog_quantity
FROM catalog_state_clean
WHERE catalog_item_indicator='Y'
GROUP BY state;
QUIT;


PROC SQL;
CREATE TABLE web_orders_state AS
SELECT state,count(web_item_indicator) AS total_web_orders,
sum(web_item_quantity) AS total_web_quantity
FROM catalog_state_clean
WHERE web_item_indicator='Y'
GROUP BY state;
QUIT;



PROC SQL;
CREATE TABLE other_orders_state AS
SELECT state,count(order_number) AS total_other_orders
FROM catalog_state_clean
WHERE web_item_indicator='N' AND catalog_item_indicator='N'
GROUP BY state;
QUIT;


PROC SQL;
CREATE TABLE total_orders_state AS
```

```
SELECT state,count(order_number) AS total_orders
FROM catalog_state_clean
Group BY state;
QUIT;


DATA orders_by_state;
MERGE catalog_orders_state web_orders_state other_orders_state total_orders_state;
BY state;
RUN;



DATA orders_by_state;
SET orders_by_state;
if state=' ' then delete;
RUN;

PROC SORT DATA=orders_by_state;
BY total_orders;
RUN;

PROC PRINT DATA=orders_by_state;
RUN;


/* Bottom 5 States */


PROC SQL OUTOBS=5;
CREATE TABLE bottom_5_states AS
SELECT * FROM orders_by_state
ORDER BY total_orders;
QUIT;

PROC PRINT DATA=bottom_5_states;
RUN;


/* Top 5 States */


PROC SQL OUTOBS=5;
CREATE TABLE top_5_states AS
SELECT * FROM orders_by_state
```

```
ORDER BY total_orders DESC;
QUIT;

PROC PRINT DATA=top_5_states;
RUN;
```

/*Insight 3 */

```
PROC SQL;
CREATE TABLE catalog_order_year AS
SELECT year(input_date) AS year,count(catalog_item_indicator) AS total_catalog_orders,
SUM(catalog_item_quantity) AS total_catalog_quantity
FROM catalog_state_clean
WHERE catalog_item_indicator='Y'
GROUP BY year;
QUIT;


PROC SQL;
CREATE TABLE web_order_year AS
SELECT year(input_date) AS year,count(web_item_indicator) AS total_web_orders,
sum(web_item_quantity) AS total_web_quantity
FROM catalog_state_clean
WHERE web_item_indicator='Y'
GROUP BY year;
QUIT;


PROC SQL;
CREATE TABLE other_order_year AS
SELECT year(input_date) AS year,count(order_number) AS total_other_orders
FROM catalog_state_clean
WHERE web_item_indicator='N' AND catalog_item_indicator='N'
GROUP BY year;
QUIT;


DATA catalog_web_ratio_year;
MERGE catalog_order_year web_order_year other_order_year;
BY year;
RUN;
```

```
DATA catalog_web_ratio_year;
SET catalog_web_ratio_year;
ratio= total_web_orders/total_catalog_orders;
RUN;

PROC PRINT DATA=catalog_web_ratio_year;
RUN;

/* orders from web catalog have increased in the recent years */


/*Insight 4*/


PROC SQL;
CREATE TABLE returned_per_state AS
SELECT state,count(order_number) AS total_orders,
avg(returned_amount) AS returned_amount
FROM catalog_state_clean
GROUP BY state
ORDER BY returned_amount DESC;
QUIT;


PROC PRINT DATA=returned_per_state;
RUN;

* Returned amount in hawaii state is high compared to others, Should avoid future losses;


PROC SQL;
CREATE TABLE returned_per_zip AS
SELECT zip, state, count(order_number) AS total_orders,
avg(returned_amount) AS returned_amount
FROM catalog_state_clean
WHERE returned_amount != 0
GROUP BY zip
ORDER BY returned_amount DESC;
QUIT;

PROC PRINT DATA=returned_per_zip;
RUN;
```