Project 1–Predicting Catalog Demand

November 17, 2016

1 Business and Data Understanding

In this project we will explore whether or not to send a product catalog to a mailing list. Specificially, we need to decide if the likely profit resulting from sending this catalog (less the costs of sending the catalog) is greater than $\$10,000$, a cutoff price set by management.

To make this decision, we need data that will allow us to predict what customers on our mailing list will make purchases in response to the catalog, and how much they will spend. Let's investigate the data we have to get a better idea. We'll start by loading up our data and relevant libraries, and displaying the avalible data points about our customer base:

```
In [1]: %pylab inline

        import matplotlib.pyplot as plt
        import numpy as np
        import pandas as pd
        from sklearn.linear_model import LinearRegression
        from sklearn.feature_selection import f_regression

        import warnings
        warnings.filterwarnings(action="ignore",
                                module="scipy",
                                message="^internal gelsd")

        CUSTOMERS_FILE_NAME = "p1-customers.xlsx"
        MAILING_LIST_FILE_NAME = "p1-mailinglist.xlsx"

        customers = pd.read_excel(CUSTOMERS_FILE_NAME)
        mailing_list = pd.read_excel(MAILING_LIST_FILE_NAME)

        print "\nCustomer Columns\n"
        print "\n".join(customers.columns.values)
```

```
Populating the interactive namespace from numpy and matplotlib

Customer Columns

Name
```

```
Customer Segment
Customer ID
Address
City
State
ZIP
Avg Sale Amount
Store Number
Responded to Last Catalog
Avg Num Products Purchased
# Years as Customer
```

I'd be very curious to know exactly what is meant by `Avg Sale Amount` and `Avg Num Products Purchased`. I'm especially curious how we have the later data for the mailing list customers...

2 Analysis, Modeling, and Validation

Let's begin by noting that `Ave Sale Amount` is the variable we wish to predict. Also, we only want to consider those customers that `Responded to Last Catalog`

```
In [2]: customers.loc(customers["Responded to Last Catalog"] == "Yes")
        sales = customers["Avg Sale Amount"]
```

Let's consider which columns we want to use in our regression. `Name` is right out, as it should have no predictive power, and we would end up with a jillion dummy variables if we tried to use it. `Customer ID` and `Address` are out for the same reason. Now `City`, `State`, `ZIP`, and `Store Number` are interesting. We might indeed expect those to have predictive potentential. However, upon inspection, all the customers are from Colorado, and there are so many of the other catagories that our data starts looking very thin.

Finally, what about `# Years as Customer`? This field is problematic as the people on the mailing list have all been customers far shorter of a time than the population on the general customers list.

That leaves us with `Avg Num Products Purchased`, and `Customer Segment` as our predictor variables. The latter is a catagorical variable, so we'll convert it to dummy variables, with "Credit Card Only" as the base case.

One more thing: We need to save off the purchase probability of the mailing list, for use later. Predicting whether or not someone will buy from a catalog seems like far more of a diffiult task that what we're trying to do here, so I'm glad we've got that data to hand.

```
In [3]: predictor_vars = ["Avg Num Products Purchased", "Customer Segment"]

        customers = customers[predictor_vars]
        customers = pd.get_dummies(customers, drop_first=True)

        purchase_probability = mailing_list["Score_Yes"]
        mailing_list = mailing_list[predictor_vars]
        mailing_list = pd.get_dummies(mailing_list, drop_first=True)
```

Let's see if we've made a sensible decision in our predictor variables by running an

```
In [4]: print "\n".join(map(str,zip(
                    map(str,customers.columns.values),
                    f_regression(customers, sales)[1])
                          ))
```

```
('Avg Num Products Purchased', 0.0)
('Customer Segment_Loyalty Club Only', 0.77955187819458072)
('Customer Segment_Loyalty Club and Credit Card', 3.7698750322070151e-224)
('Customer Segment_Store Mailing List', 3.3369210186601561e-305)
```

I'm honestly not sure what it means if the p value of one dummy variable is large while the other two are small, but it certainly seems that we have selected our predictor variables properly (I also went through this whole process with lasso regression on all the avalible variables, and these are the only ones that got any weight). Recall that a p value of under about 0.05 suggests that the variable is indeed predictive.

Let's do some regression!

```
In [5]: X = customers
        y = sales

        lr = LinearRegression()

        lr.fit(X, y)
        print lr.coef_
        print lr.intercept_
```

```
[  66.97620492 -149.35572194  281.83876492 -245.4177445 ]
303.463471315
```

The predicted spending per customer who recieves the catalog (assuming they do make a purchase) is given by the following equation:

$$\begin{aligned} \mathit{spend} &= 303.46 + 66.98 \times \mathit{items} - 149.36 \times \mathit{loyaltyOnly} \\ &\quad + 281.84 \times \mathit{loyaltyAndCC} - 245.42 \times \mathit{mailingList}. \end{aligned}$$

Let's check out the R^2 value to see how good a model this is:

```
In [6]: lr.score(X, y)
```

```
Out[6]: 0.8368777093556734
```

Interpreting R^2 values is more of a shurg than an art, but this seems fine. I'm be super stoked with 0.9, and I'd be looking very ascance at 0.5, so 0.84 is pretty decent.

3 Making Business Decisions

Now let's predict how much each customer on the mailing list would spend, if they do indeed make a purchase. Then we'll multiply those values by the odds that that customer will make a purchase. We can sum those expected sales to get the gross take. We'll then multiply by our margin, subtract out the cost of sending the catalog, and we'll have the profit generated by sending the catalog!

```
In [7]: predicted_spending = lr.predict(mailing_list)
        weighted_spending = predicted_spending * purchase_probability
        gross = sum(weighted_spending)
        profit = .5 * gross
        cost = 6.5 * mailing_list.shape[0]
        cutoff = 10000
        net = profit - cost
        print net
```

```
21987.4356865
```

Looks like we can expect to make $\$21,987.44$ by sending this catalog, which is greater than the $\$10,000$ cutoff. Therefore, I reccomend we do indeed send the catalog!