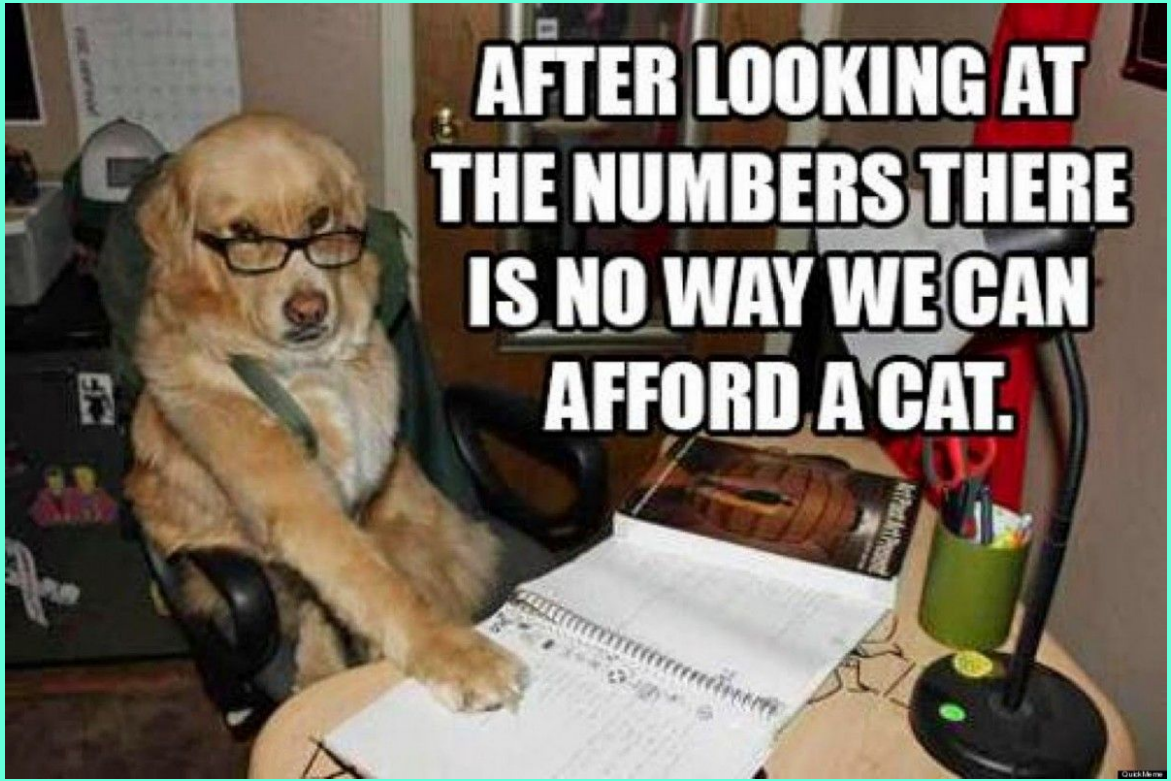
Nomis Solutions

Pricing Optimization for Consumer Credit Providers

Team 8

Joseph Miguel, Jeremy Grace, Jonathan Hilgart, & Victor Vulovic


AFTER LOOKING AT
THE NUMBERS THERE
IS NO WAY WE CAN
AFFORD A CAT.

Agenda

- Nomis Business Case & Opportunity Analysis
- Analytical Procedure
- Results
- Call to Action

Nomis Business Case & Opportunity Analysis

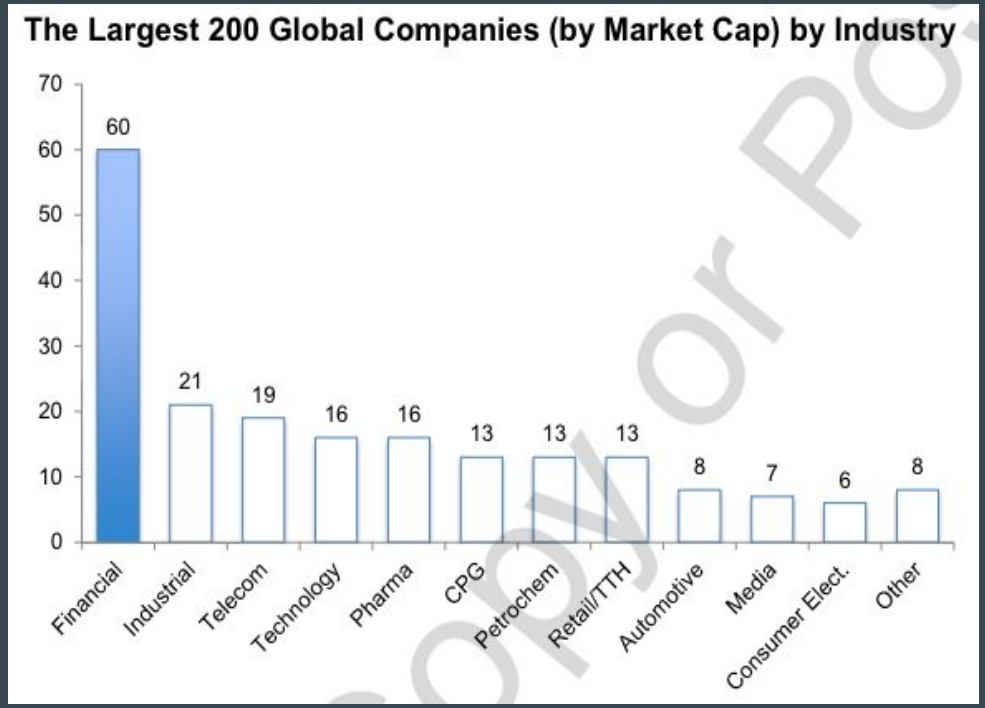
$9,000,000,000

US Financial Services Firms

~~~

The premier untapped vertical for Analytics
~~~

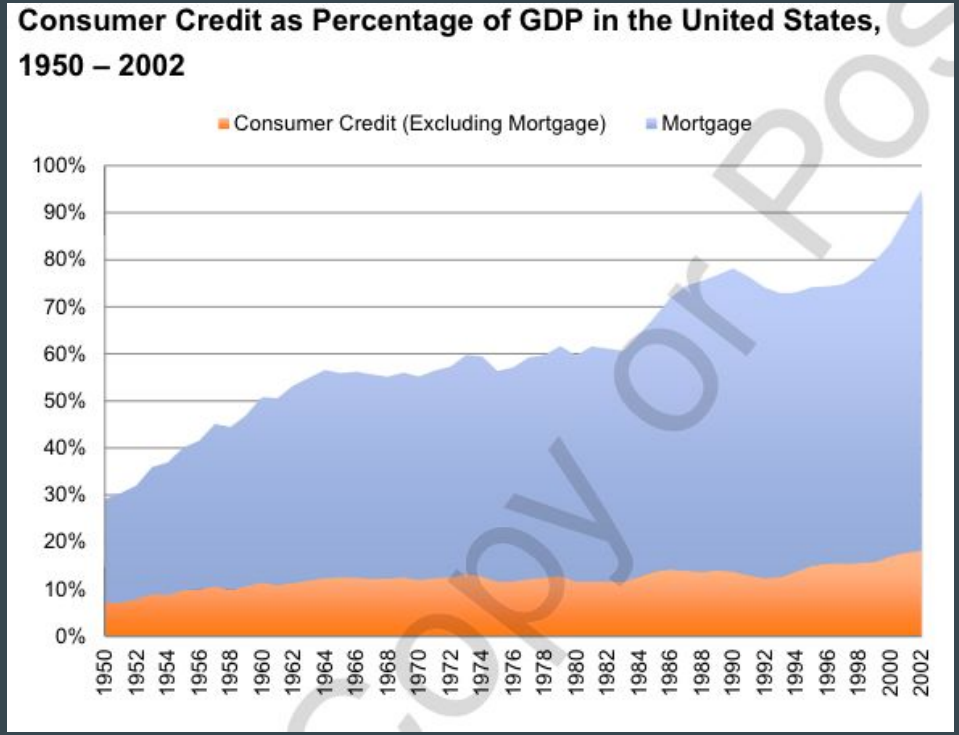
Market Size - Financial Services Industry



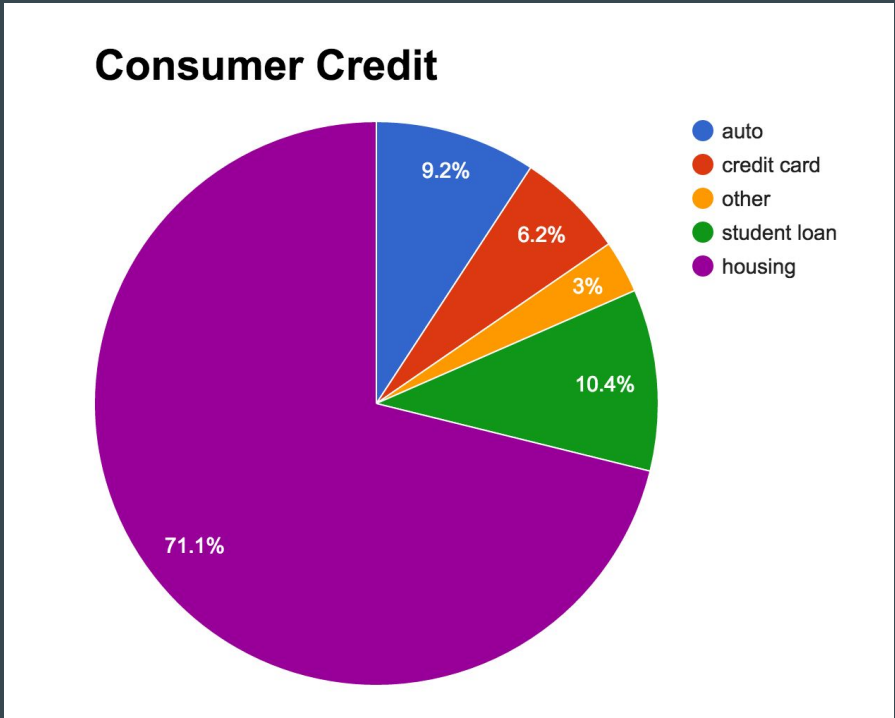
- 30% of Fortune 200 companies in Financial Services industry
- 9441 banks in the US alone
- Total estimated assets under management: $9 Trillion

Market Size

- Total Consumer Credit:
  - $10.33 Trillion
  - 95% of GDP
  - $90k/household
- Auto-loans:
  - 9.22% of total consumer credit
  - $952.5 Million



Market Opportunity

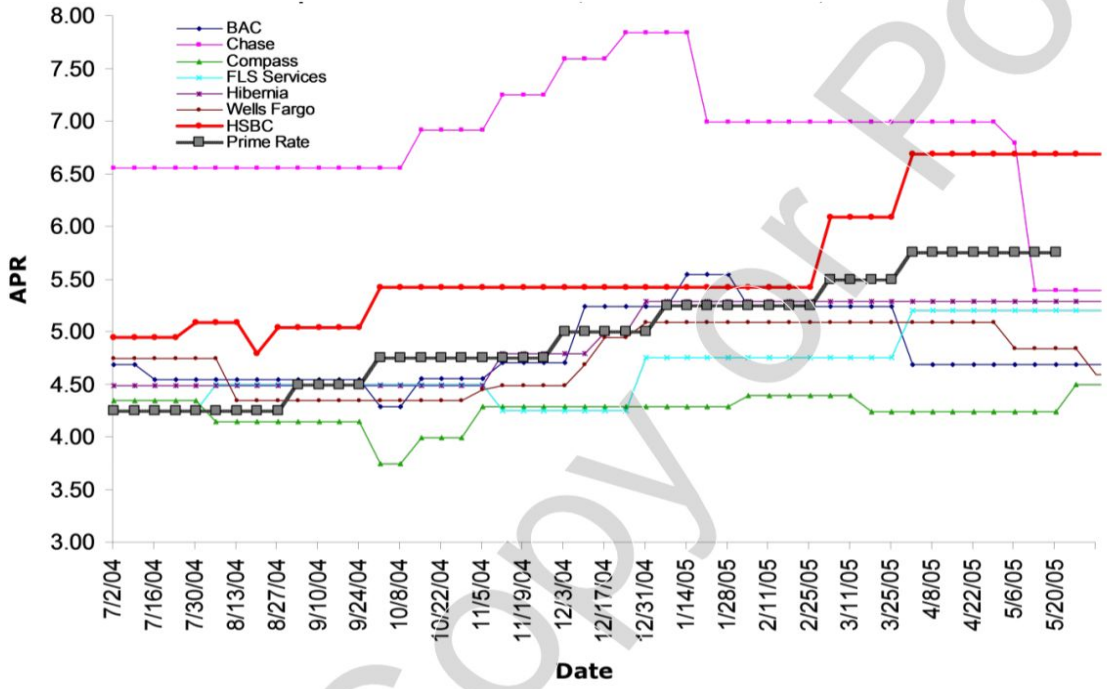


- Housing:
  - 71.1% of total consumer credit
  - $7.35 Trillion
- Auto-loans:
  - 9.22% of total consumer credit
  - $952.5 Million
- Growth:
  - Estimated 3.33% YOY

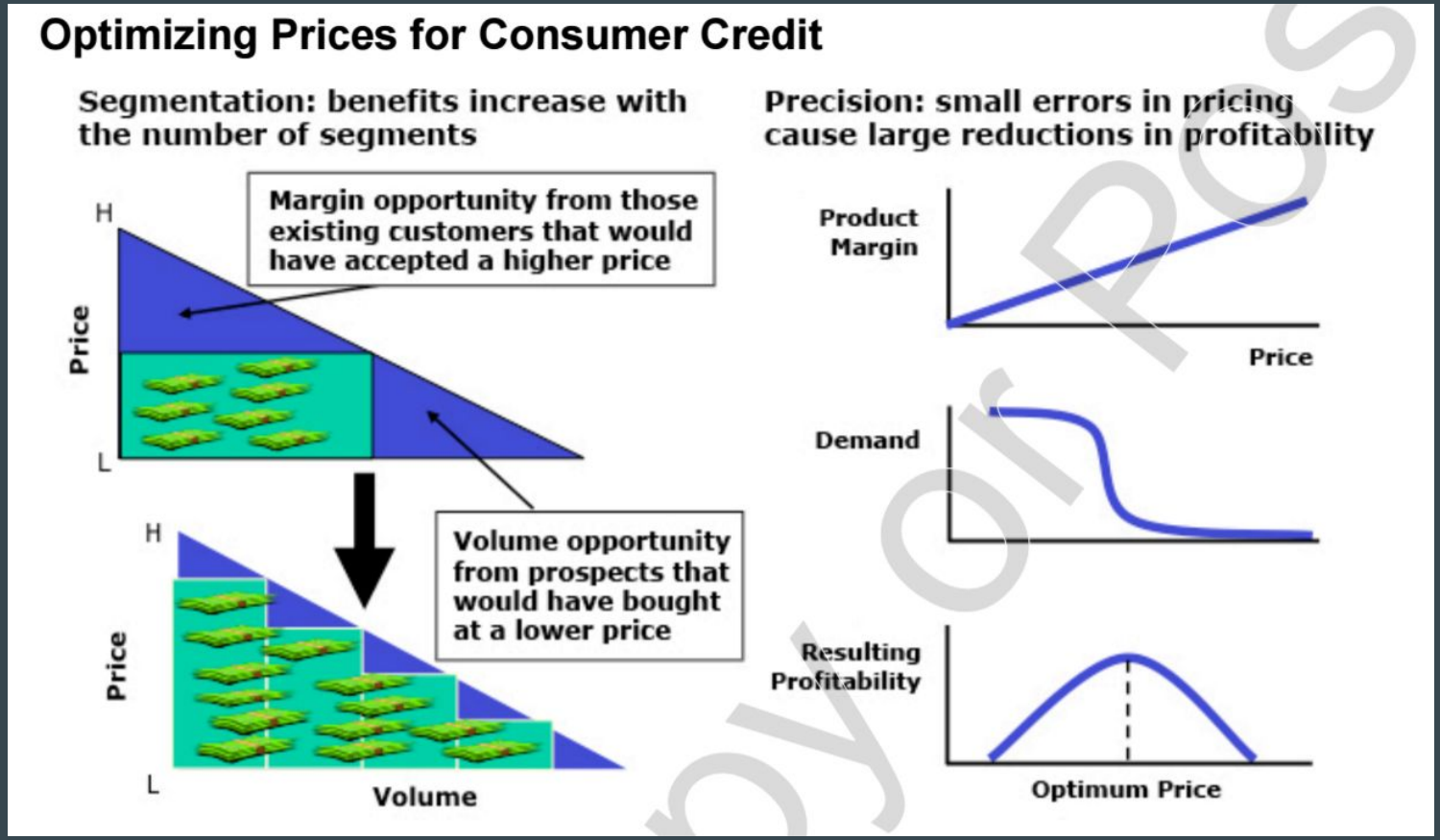
Market Opportunity

Exhibit 6

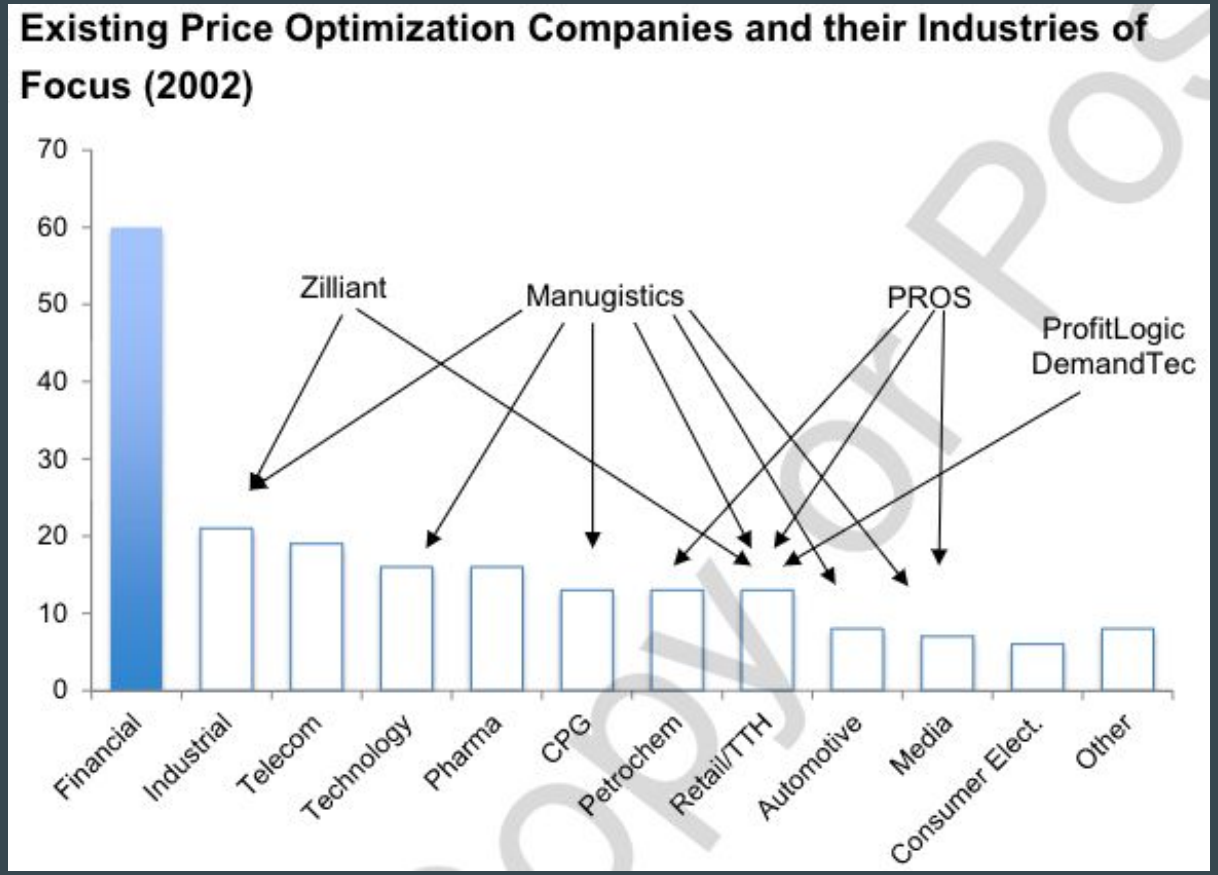
Auto Loan Pricing in Houston Market, mid-2004 to mid-2005*



Market Opportunity



Competitive Landscape



Analytical Procedure

Outcome Map


Perform EDA and calculate aggregate NPV
Build optimal segmentation model
Customize Prices per Segment
Maximize Profitability
Secure Customer Contract

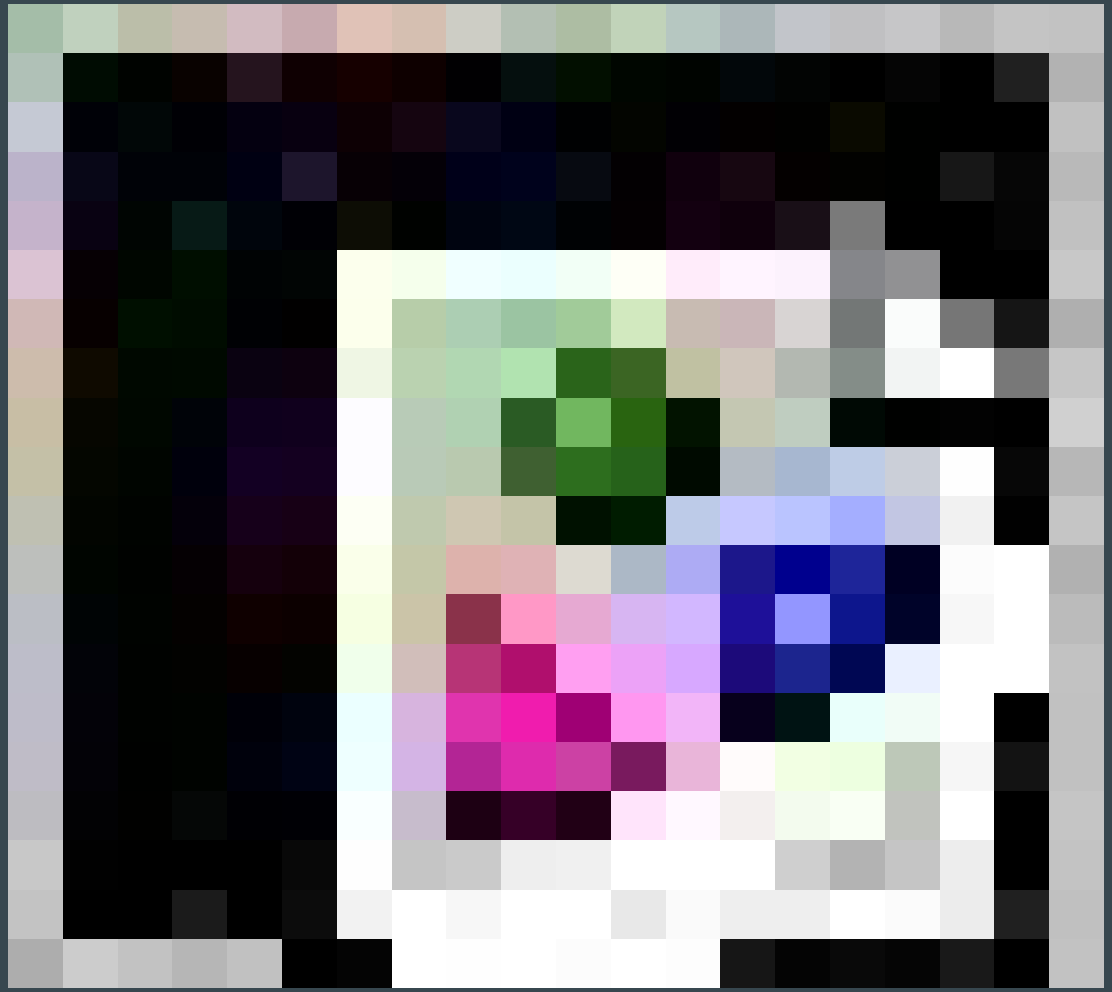
Overview

1. Year of reference: 2002
2. Zero default risk
3. Full Interest Collection
4. Goal: Move from tiered based pricing to pricing per statistical segment

Current Profitability

1. NPV in 2002 money: **$171.3 million**

2. Process: annual NPV discounted to 2002 (year 0)



Distribution of Originated ‘Spread of our rate vs competitors’ vs ‘FICO’.

Positive spread indicates our rate is higher than our competitors

Source Origination and their average Spread

N = New, U = Used, R = Refinance

	car__type	count(car__type)	avg(spread__rate_vs_competition)
0	U	41816	0.012476
1	N	119059	0.006549
2	R	47210	0.008398

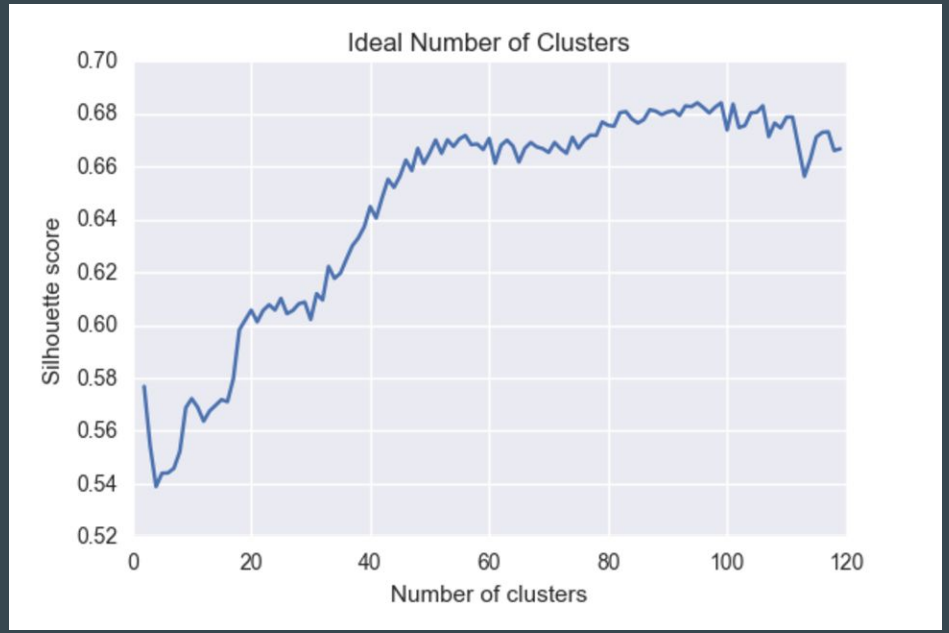
Statistical Segmentation

- Silhouette Score
  - 3 fold CV on 10k samples

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Which can be also written as:

$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i) \end{cases}$$



EDA & NPV → SEGMENT OPT. → PRICE OPT. → MAX PROFIT → SOLD

Predict Outcome Per Segment

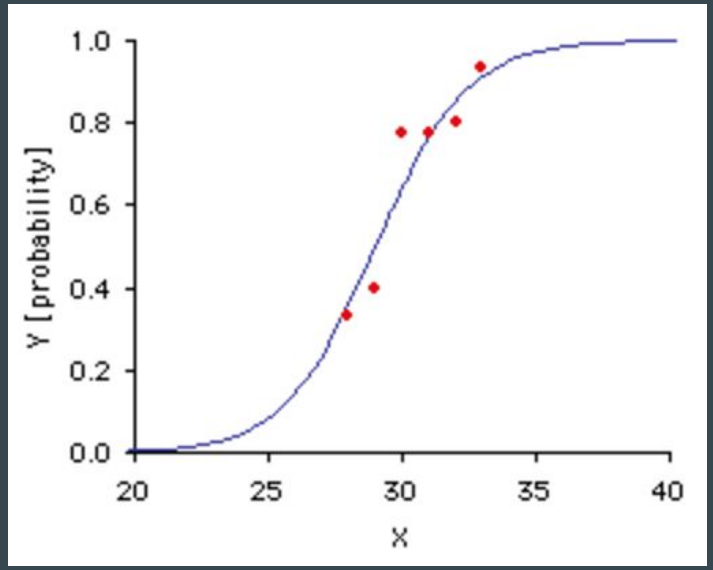
- Logistic Regression
  - One model trained per cluster
- Variables:
  - FICO, tier, term, amount, rate, parner bin, previous rate, competition rate, cost of funds

Predict Outcome Per Segment

```
def create_model_per_cluster(dataframe_in,list_of_clusters):
    """Create a logistic regression model for each cluster to predict outcome.
    Return a dictionary where the key
    is the cluster number and the value is the trained model.
    Also returns a dictionary of RMSE per cluster"""
    rmse_per_cluster = {}
    list_of_models_per_cluster = {}
    for cluster in list_of_clusters:
        cluster_df = dataframe_in[dataframe_in.cluster_number==cluster]
        df_X =cluster_df.loc[:,('tier', 'FICO', 'Term', 'Amount', 'Rate', 'Partner Bin', \
       'previous_rate', 'competition_rate', 'cost_of_funds',\
        'car_type_N', 'car_type_R', 'car_type_U')]
        df_y = cluster_df['Outcome']
        # train test split
        X_train, X_test, y_train, y_test = train_test_split(
        df_X, df_y, test_size=0.33)
        # create the classifier
        Log_classifier_model = LogisticRegression()
        Log_classifier_model .fit(X_train,y_train)
        classifier_predictions = Log_classifier_model .predict(X_test)
        rmse = np.linalg.norm(y_test - classifier_predictions)/sqrt(len(y_test))
        rmse_per_cluster[cluster]=rmse
        list_of_models_per_cluster[cluster]=Log_classifier_model
    return rmse_per_cluster,list_of_models_per_cluster
```

Price (APR) Sensitivity Analysis

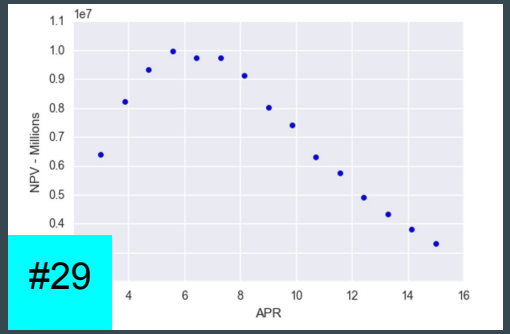
- Per Cluster Sensitivity
  - Start at APR of 1% and go up to 15%
  - Assume all loans start at the same time
  - Predict binary outcome

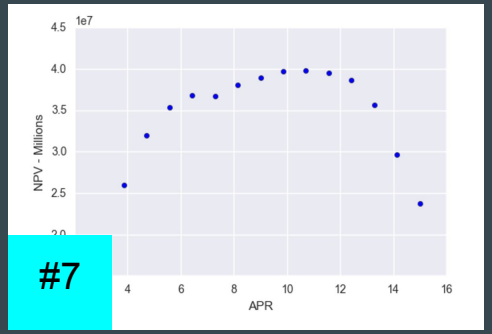


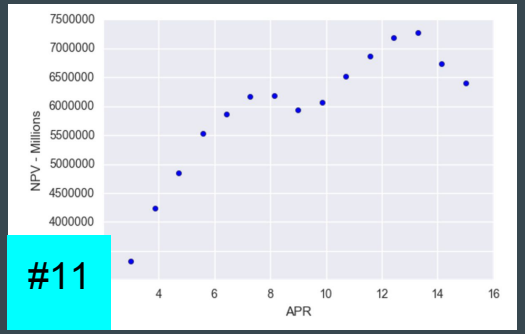
Price (APR) Sensitivity Analysis

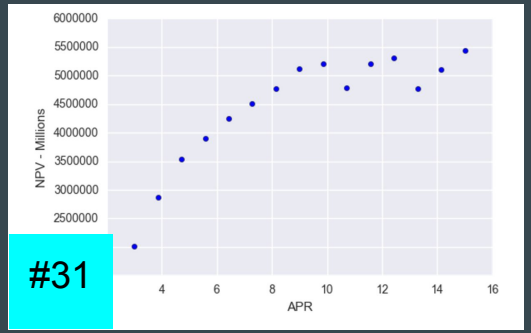
```
def outcome_per_apr_rate(dataframe_in,dict_of_models,list_of_clusters,list_of_apr):
    """Change the APR per cluster and predict outcome.
    Return a dictionary of a list of tuplues where the initial key = cluster number and value = a list
    The second (inside) list has the apr as the key and NPV and the value for each tuple.
    Also returns each clusters max APR rate as well as max NPV rate in the form of a dict."""
    cluster_apr_npv = defaultdict(list)
    max_apr_per_cluster = defaultdict(int)
    max_npv_per_cluster = defaultdict(int)
    for cluster in list_of_clusters: ## which cluster are we looking at
        current_model = dict_of_models[cluster]
        apr_npv_per_cluster = []
        cluster_df = dataframe_in[dataframe_in.cluster_number==cluster]
        for apr_rate in list_of_apr:
            # Change the APR rate for the entire cluster
            cluster_df.Rate = apr_rate
            df_X =cluster_df.loc[:,('tier', 'FICO', 'Term', 'Amount', 'Rate', 'Partner Bin', \
           'previous_rate', 'competition_rate', 'cost_of_funds',\
            'car_type_N', 'car_type_R', 'car_type_U')]
            df_y = cluster_df.loc[:,'Outcome']
            cluster_df['predictions']=current_model.predict(df_X)
            cluster_df['npv'] = \
            cluster_df.apply(lambda x: npv_loan_amount(x['predictions']*x['Amount'],
                                                       x['Term'],
                                                       x['Rate'],
                                                       x['cost_of_funds']), axis=1)
            apr_npv_per_cluster.append((apr_rate,sum(cluster_df['npv'])))
        cluster_apr_npv[cluster] = apr_npv_per_cluster
        # get max apr and max npv for each cluster
        max_apr_per_cluster[cluster] = max(cluster_apr_npv[cluster] ,key=itemgetter(1))[0]
        max_npv_per_cluster[cluster] = max(cluster_apr_npv[cluster] ,key=itemgetter(1))[1]
        print("Finished calculating NPV for cluster {}".format(cluster))
    return cluster_apr_npv, max_apr_per_cluster, max_npv_per_cluster
```

Optimal Prices per Segment





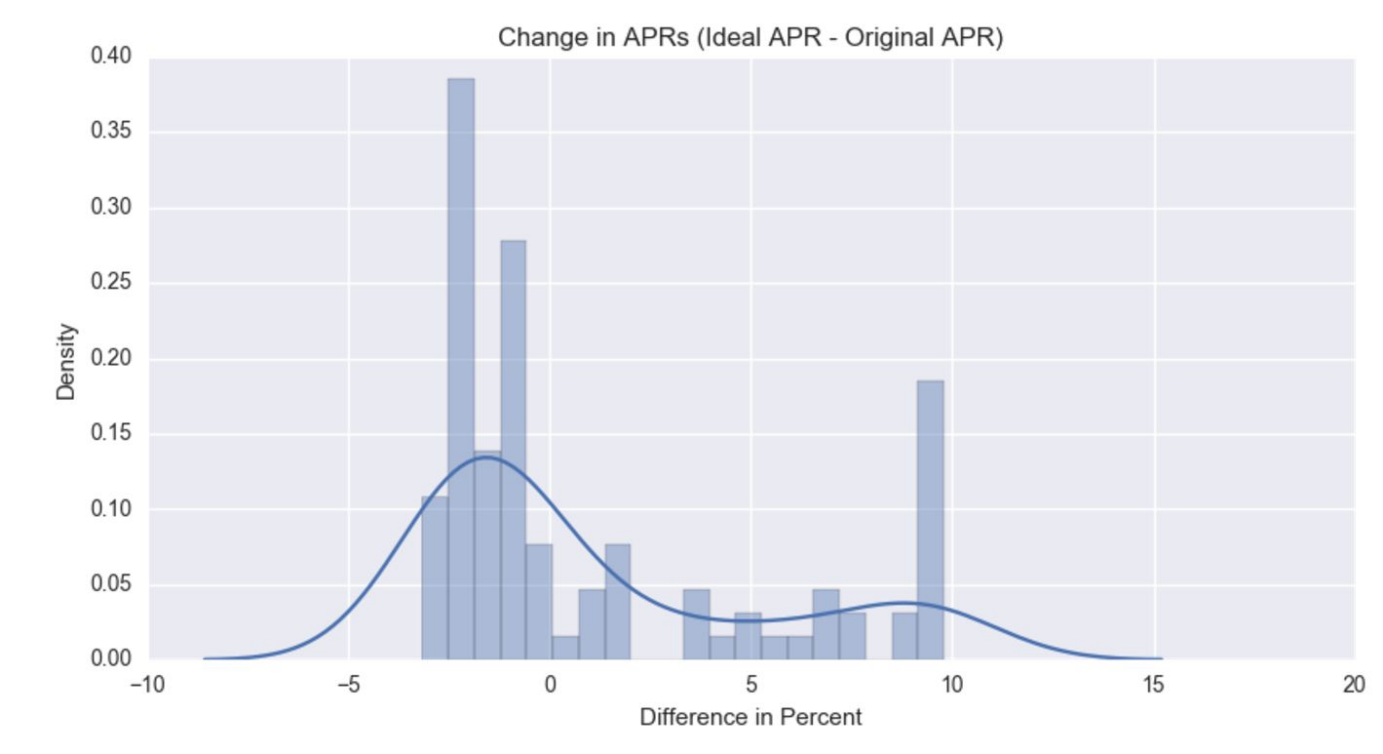




- *Assume max APR we can charge is 15%*

Results

How Do Optimal APRs Change?



EDA & NPV → SEGMENT OPT. → PRICE OPT. → MAX PROFIT → SOLD

NPV Comparison

NPV of Original
4-tier System

$171.3 million

NPV of Optimized
Customer Segments

~ $1.5 billion

Results

How does this translate into added value for E-car?

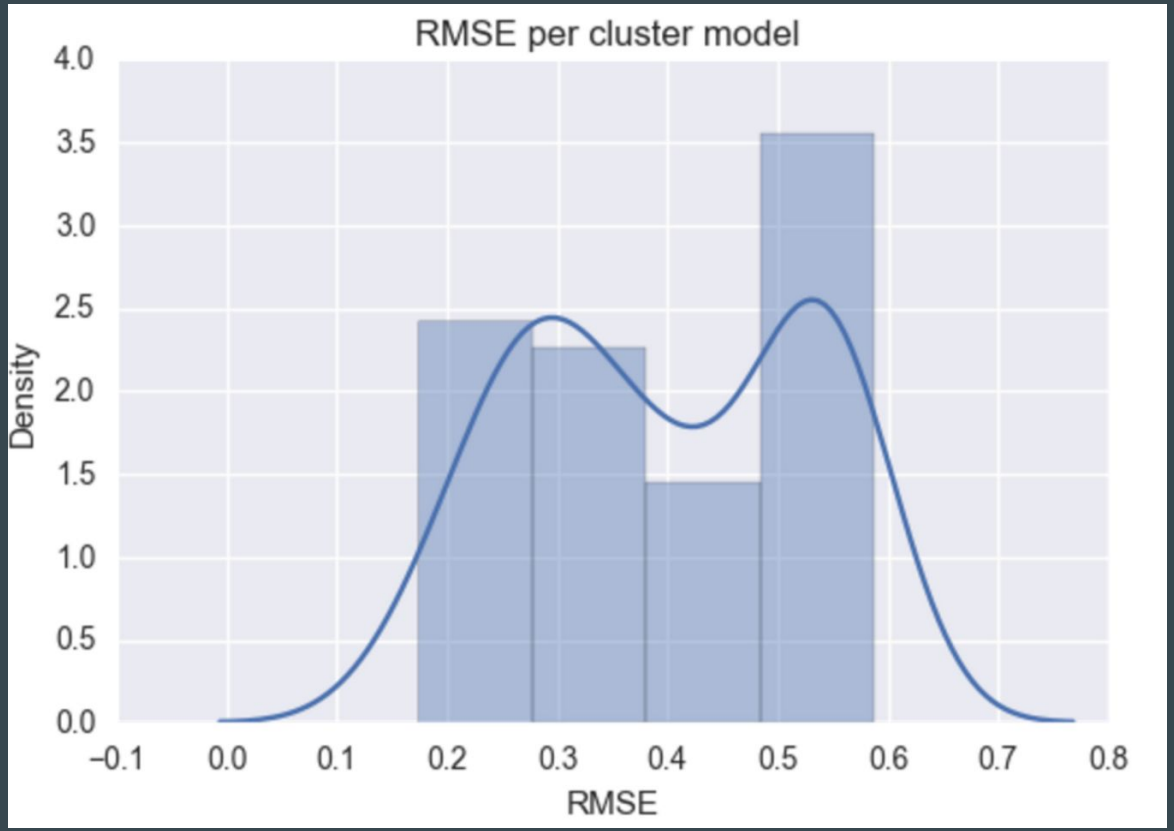
Over 8x increase in Profitability*

Model Error

- Even with positive results, there is a lot of variance in outcome

- Increased accuracy likely with more info
  - Ex: principal payoff records

- *Predicting outcome as 1 or 0*



Call to Action

Conclusion

- Goal to capture 10% of the services firm market by maximizing the margin and volume of loans
- Recommend increasing APR to 80% of ‘ideal’ rate for either rate increase or decrease and see results
- Next steps include choosing APR based upon PI and MIRR

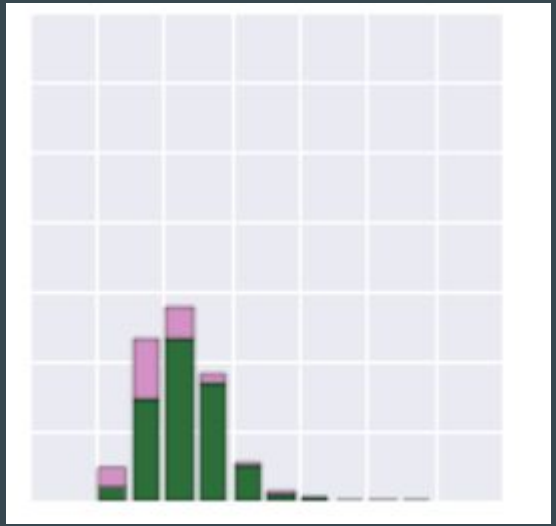
Thank You

Questions

Distributions

FCO

Amount



Spread

Means and Standard Deviations of us and them

summary	fico	spread__rate_vs_competition
count	208085	208085
mean	726.731407838143	0.00815971622173604 4
stddev	44.783564051538	0.014363978607644651
min	587	-0.0185999999999999 9
max	854	0.1127999999999998