



PURCHASE PREDICTION FOR ALLSTATE'S CAR INSURANCE ONLINE CUSTOMERS

Amit Kukreja

28-June-2022

APPROACH: THE SIX STEP DATA SCIENCE METHOD



Problem Statement: Develop a **predictive model** to predict the coverage plan that Allstate's online customer would eventually buy.

Context : AllState's online customer's make changes to product features (vectors) to generate multiple quotes. Each revision increases the probability of losing the customer. If the eventual purchase can predict sooner, the quoting process is shortened and Allstate is more likely to get the customer's business.

Criteria for Success: A model that can accurately predict customer's final vector choices early in the shopping window.

Scope : The model shall be developed based on data of approx. 97,000 customers provided by Allstate.

Constraints: The predictive power of the model would be limited by:

- Non-availability of data on competitive products
- Lack of important customer parameters like household income, car model and driving history.

Stakeholders to provide Key Insight:

1) Ben Bell – Springboard Mentor

Data Sources: Dataset of 97,000 customers provided by Allstate on Kaggle

<https://www.kaggle.com/c/allstate-purchase-prediction-challenge/data>



DATA WRANGLING

Key Outcomes

1. CONVERTED THE TRAINING DATASET FROM LONG FORM TO A WIDE FORM

- Training data had **665,249** observations for **97,009** unique customers.
- Each row had quote details of a customer
- Converted this to wide form, each row containing entire purchase history of the customer

	customer_ID	shopping_pt	record_type	day	time	state	location	group_size	homeowner	car_age	
0	10000000	1	0	0	08:35	IN	10001	2	0	2	.
1	10000000	2	0	0	08:38	IN	10001	2	0	2	.
2	10000000	3	0	0	08:38	IN	10001	2	0	2	.
3	10000000	4	0	0	08:39	IN	10001	2	0	2	.
4	10000000	5	0	0	11:55	IN	10001	2	0	2	.
5	10000000	6	0	0	11:57	IN	10001	2	0	2	.
6	10000000	7	0	0	11:58	IN	10001	2	0	2	.
7	10000000	8	0	0	12:03	IN	10001	2	0	2	.
8	10000000	9	1	0	12:07	IN	10001	2	0	2	.
9	10000005	1	0	3	08:56	NY	10006	1	0	10	.
10	10000005	2	0	3	08:56	NY	10006	1	0	10	.

A customer across multiple rows (each observation is a **QUOTE**)

	customer_ID	shopping_pt	record_type	day	time	state	location	group_size	homeowner	car_age	...	C_p
0	10000000	9	1	0	12:07	IN	10001	2	0	2	...	
1	10000005	6	1	3	09:09	NY	10006	1	0	10	...	
2	10000007	8	1	4	14:26	PA	10008	1	0	11	...	
3	10000013	4	1	4	09:31	WV	10014	2	1	3	...	
4	10000014	6	1	1	17:50	MO	10015	1	0	5	...	

5 rows × 259 columns

One customer per row (each observation is a **CUSTOMER** now)

2. UNDERSTOOD EACH FEATURE AND CATEGORIZED IT

- **25** features categorized as follows:
 - **6 numerical variables:** shopping_pt, car_age, age_oldest and age_youngest, duration_previous & cost.
 - **10 categorical variables:** day, state, location, group_size, car_value, risk_factor, C_previous, record_type, homeowner, married_couple
 - **1 datetime variable:** time
 - **7 Response Variables or Target :** Product features (aka vectors) (A - G). They take **discrete** values.



EXPLORATORY DATA ANALYSIS

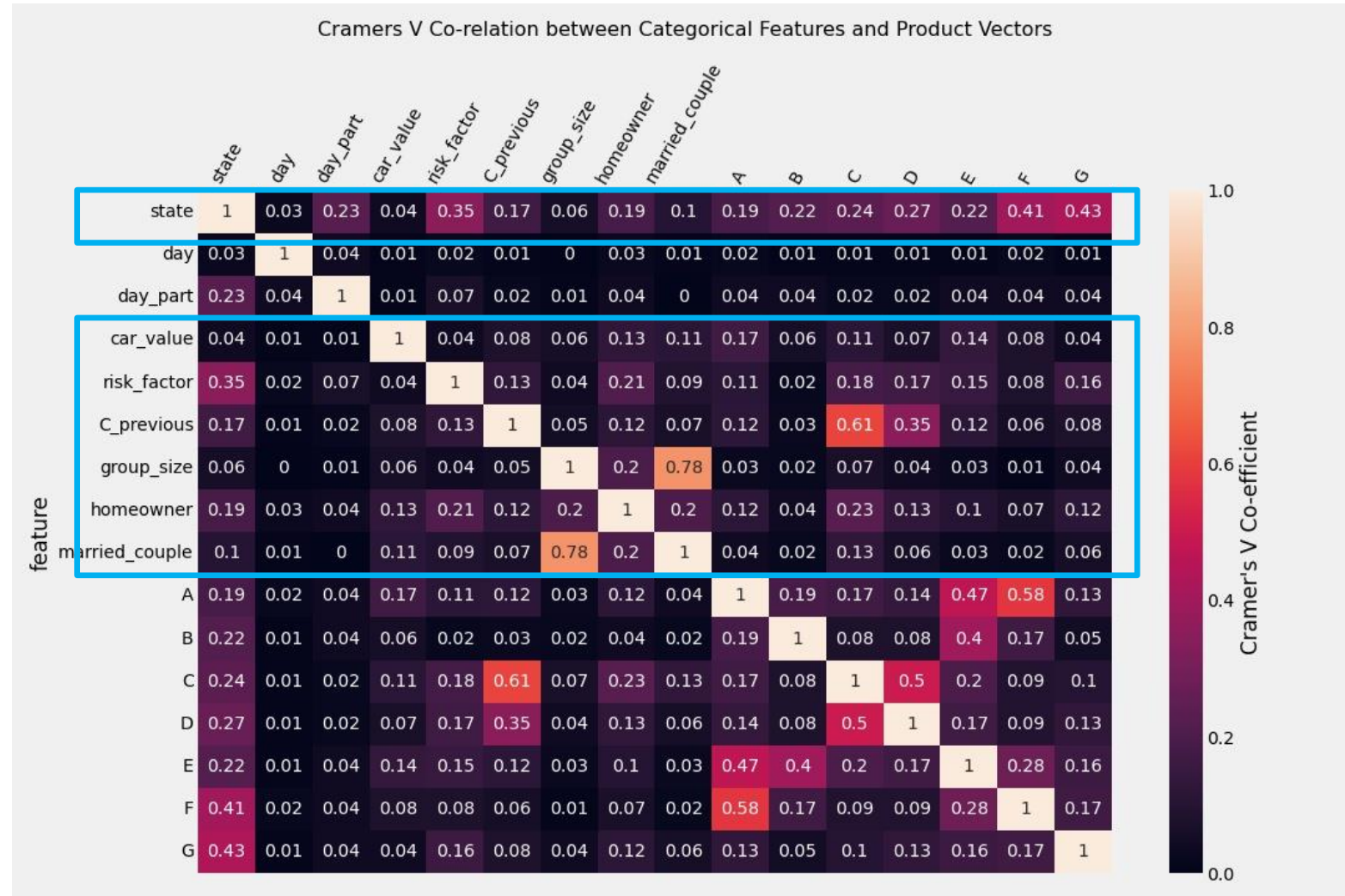
Key Learnings

IMPORTANT NOMENCLATURE

- **baseline_model** : This refers to predicting the final vector choice based on the customer's selection of the vector at the most recent shopping pt, say shopping pt 2.
- **'A'**: Customer's **final choice** for vector **A** i.e. the target we are predicting for vector A.
- **'A_2'**: Customer's choice for vector **A** at shopping pt **2**.
- **'A_2_2'**: **One-hot encoded** feature for vector **A**, at shopping pt **2**, for class **2**.

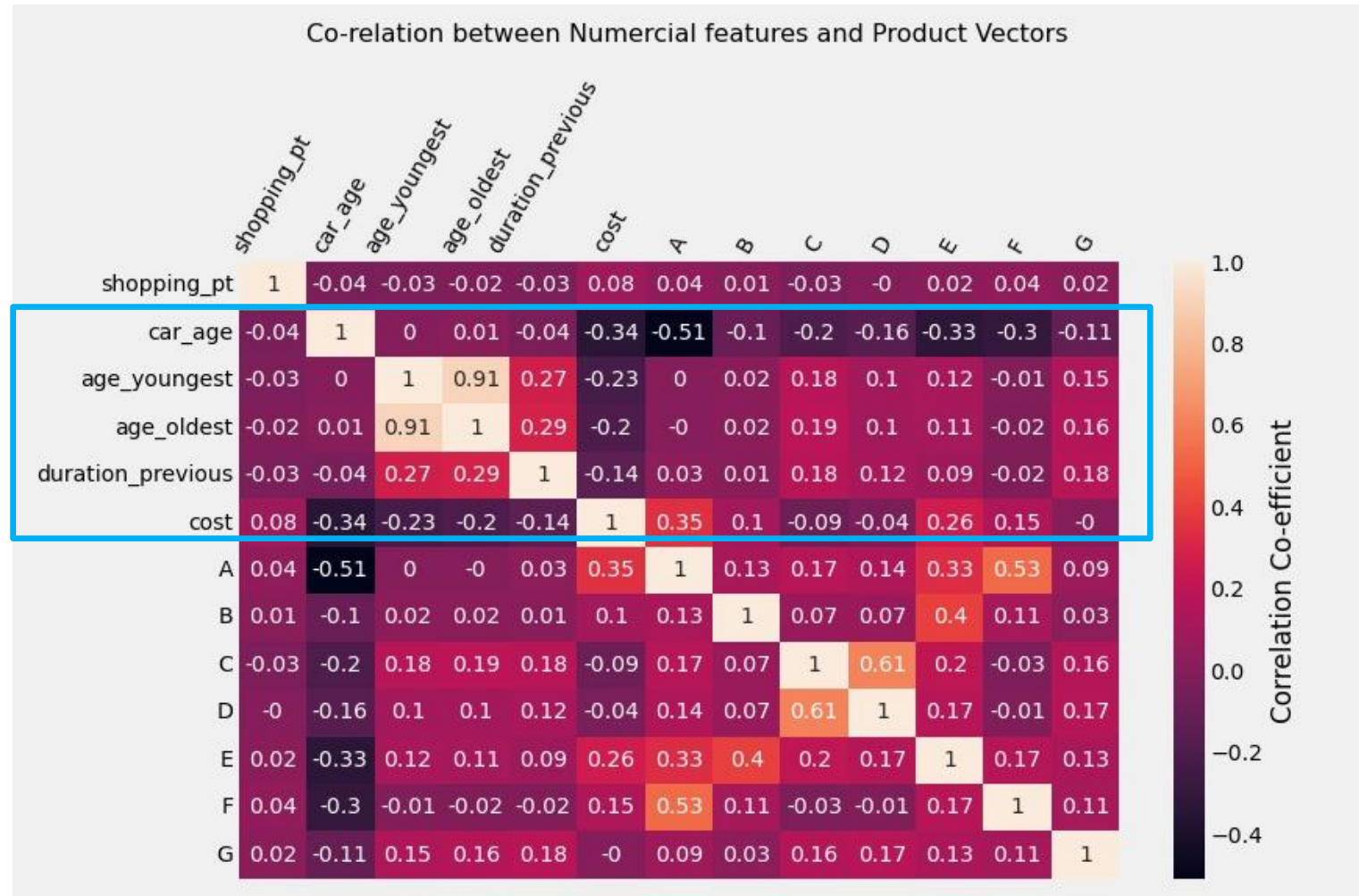
1. CORRELATIONS OF CATEGORICAL FEATURES WITH TARGET VECTORS

- Used **Cramer's V Correlation** to determine which of the categorical features had influence over the target vectors
- Basis this analysis, prioritized **7 out of 9** categorical variables for model development
 - state, car_value, risk_factor, C_previous, group_size, homeowner, married_couple



2. CORRELATIONS OF NUMERICAL FEATURES WITH TARGET VECTORS

- Analysed correlation of the numerical features with the target
- Basis this analysis, prioritized **5 out of 6** numerical variables for model development
 - car_age, age_youngest, age_oldest, duration_previous and cost



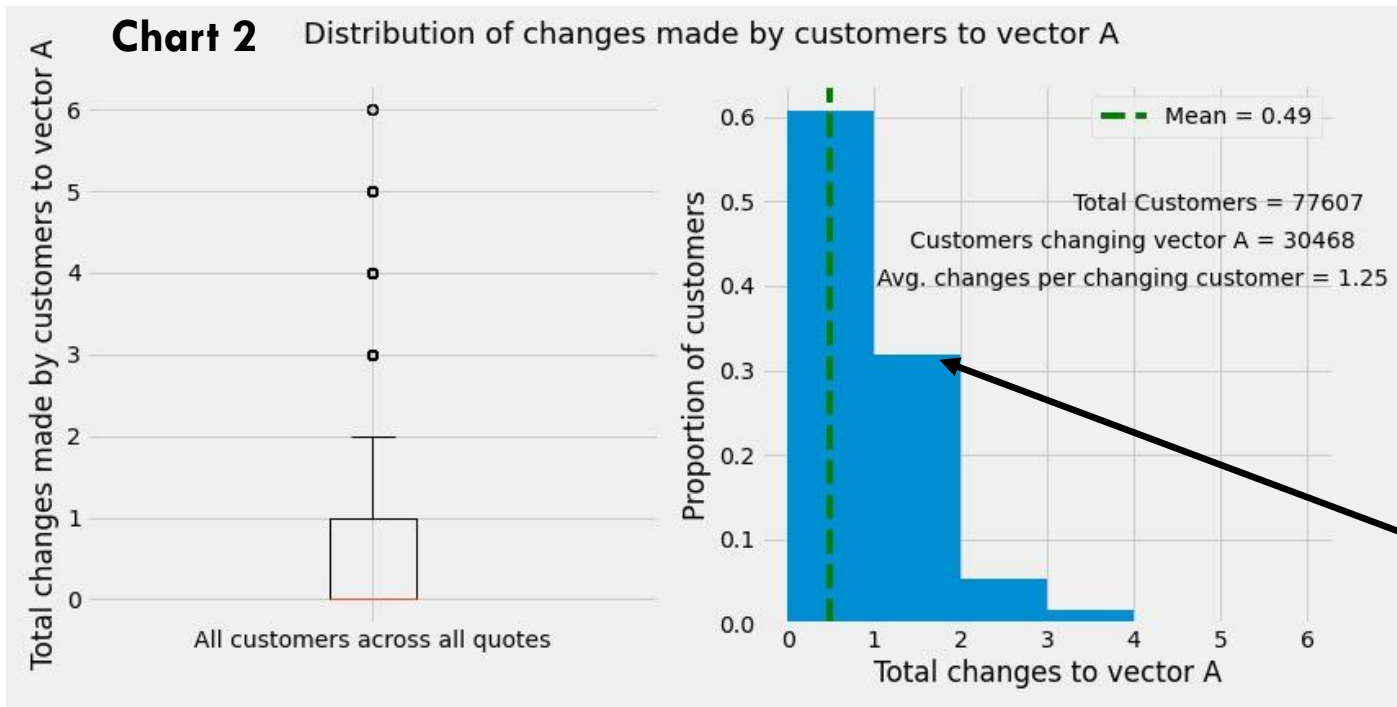
3A. CUSTOMERS CHANGE VECTORS EARLY ON

- The pattern of changes across every vector is very similar to that shown in Chart 1 & 2 for vector A
 - Most customers change vectors early on, at shopping points 2 & 3 (Chart 1)
 - Out of those who do change the vector, approx. 80% change it only once (chart 2)



After shopping point 4, **less than 5%** of active customers change vector A at any given shopping pt.

80% of customers who change vector A, change it only **once**.



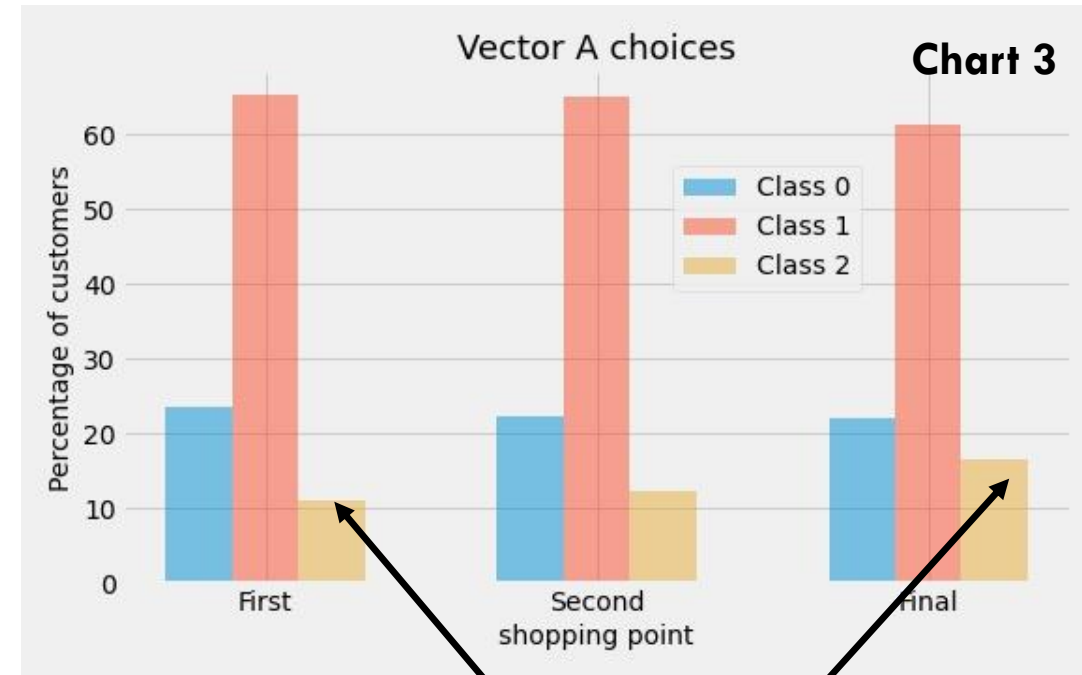
3B. VECTOR CLASSES ARE DYNAMIC THROUGHOUT QUOTING

- The footprint of different vector A classes shifts as a results of these changes
 - Classes 0 and 1 have declines from shopping point 1 through to the final shopping point (chart 3)
 - Class 2 has gains, even through it is the smallest class (chart 3)
 - But **no class is static**, there are customers entering and exiting every class (Table 1)

e.g. **856** customers moved from class 1 for **A_2**(A at shopping pt 2) to class 0 for **A** (their final A choice)

Table 1: How A_2 and A compare

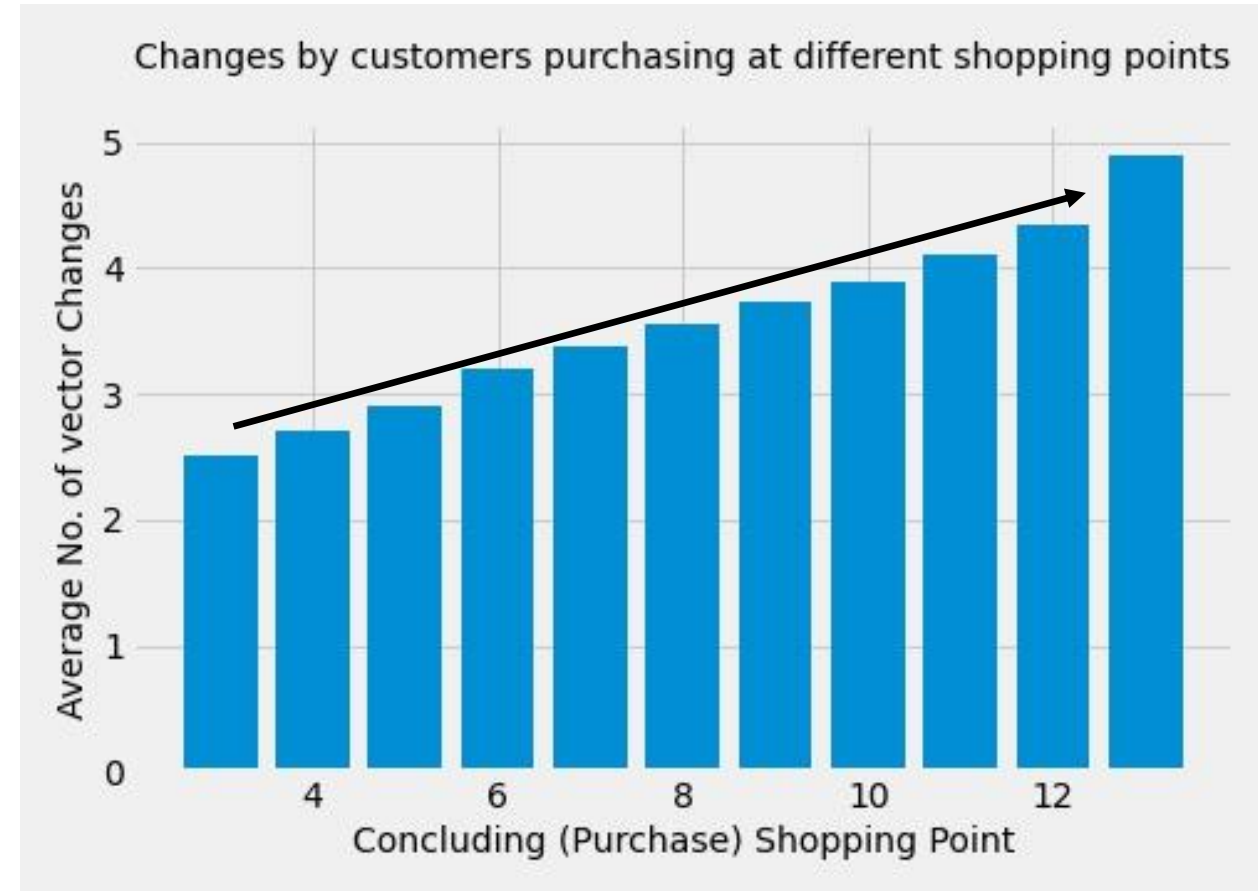
	A_2 →	Class 0	Class 1	Class 2	total
	A ↓				
Class 0		3264	856	148	4268
Class 1		769	10785	369	11923
Class 2		258	1110	1843	3211



Customers choosing class 2 increase from **10% to 18%** of total customers during the shopping cycle

4. CUSTOMERS WHO TAKE LONGER TO DECIDE MAKE MORE CHANGES

- This chart shows that no. of changes made by customers increase if they stay longer in the shopping window





PRE-PROCESSING

Key
Transformations

FEATURE ENGINEERING AND TRANSFORMATIONS

- **Extracted** quote 1, 2 and final target vectors:
 - Customers make the most changes at Quote 2. So models can learn the most with quote 1 & 2 and understand how customers make further changes
- **Engineered** 2 features:
 - **changes_step_2**: No. of vectors changed by a customer at quote 2. This might have some bearing on further changes a customer makes.
 - **cost_diff_step_2**: Difference in policy cost between quote 1 and 2. This could influence if vector changes would be retained or not
- Categorical features such as state, risk_factor etc. transformed with **dummy encoding**
- Numerical features scaled using **StandardScaler**
- Post transformation, the final features dataset has **94 columns**

```
X.head()
```

	shopping_pt	car_age_2	age_oldest_2	age_youngest_2	duration_previous_2	cost_1	cost_2	changes_step_2	cost_diff_step_2	state_AR	...	F_1_
0	-0.929538	0.306686	1.718107	1.331208	-1.064370	-0.519793	-0.583795	-0.871664	-0.043014	0	...	
1	0.071276	-0.901461	-1.151705	-1.015758	-1.064370	-0.716734	0.867553	2.086860	2.452875	0	...	
2	0.571683	-0.211091	0.742371	0.587048	-1.064370	-0.362240	-0.352421	-0.279959	0.051766	0	...	
3	-0.929538	-1.246646	-0.061176	0.071860	1.933252	-0.657652	-0.920340	-0.871664	-0.327356	0	...	
4	0.571683	-1.246646	-1.094309	-0.958515	-0.422023	1.804113	1.750982	-0.279959	-0.264169	0	...	

5 rows × 94 columns

Engineered
Features



MODELING | Making
Predictions

MODELING METRICS

The key objective is to predict **each customer's correct class** for each of the 7 vectors.

As such, the key metrics for model evaluation would be '**micro accuracy**' i.e. the overall correct class predictions across all customers.

MODELING APPROACH FOR EACH OF THE 7 VECTORS

- **Step 1:** Trained a basic model based on 3 different classifiers - Random Forest, Gradient Boost and XG Boost.
 - For binary vectors (B and E), a Logistic Regression classifier was also evaluated.
- **Step 2:** For each classifier, applied 60 iterations of RandomizedSearchCV & measured performance of different hyperparameter combinations on the **training** dataset.
 - This gives a **95% probability** of finding the best hyperparameters
- **Step 3:** Applied each of these 60 hyperparameters sets on the **validation** set and measured micro accuracy.
- **Step 4:** Chose the best set of hyperparameters as the one that performs **most consistently** across the training and validation set.
- **Step 5:** Based on the best hyperparameters, compared the performance of different classifiers on the test set to then choose the best classifier for each vector.

VECTOR A: MODEL & PERFORMANCE

- 'baseline_model' as benchmark:
 - Table 1 shows that recall based on simply predicting vector A = A_2 is **81.91%**

Table 1: How A_2 and A compare

	A_2 →	Class 0	Class 1	Class 2	total
A ↓					
Class 0		3264	856	148	4268
Class 1		769	10785	369	11923
Class 2		258	1110	1843	3211

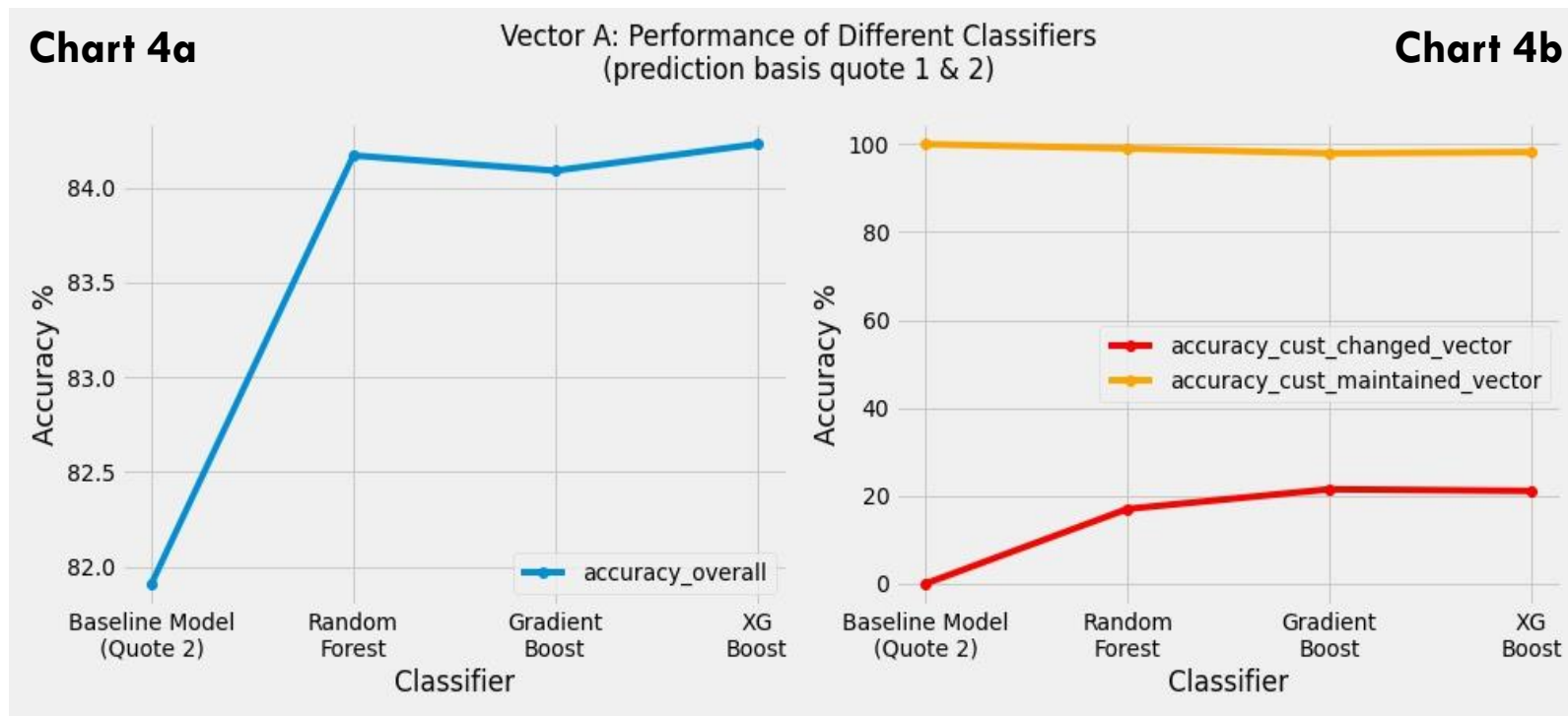
Total customers =
sum of total column

A same as A_2 i.e. these customers didn't change their vector A choice from quote 2 till purchase

$$\begin{aligned}\text{Micro Accuracy Score} &= \frac{\text{sum of diagonal}}{\text{sum of total column}} \\ &= \frac{(3264 + 10785 + 1843)}{(4268 + 11923 + 3211)} \\ &= \mathbf{81.91\%}\end{aligned}$$

VECTOR A: MODELS PERFORMED MUCH BETTER THAN BASELINE IN PREDICTING CUSTOMERS WHO CHANGED THEIR CLASS

- **Chart 4a** shows that the best classifier (XGBoost) had an accuracy score of **84.23%**
- **Chart 4b** shows the score breakup between customers:
 - **98 – 99%** accuracy of classifiers for customers who maintained their vector A class (baseline: **100%**)
 - **18 - 22%** accuracy of classifiers for customers who changed their vector A class (baseline: **0%**)



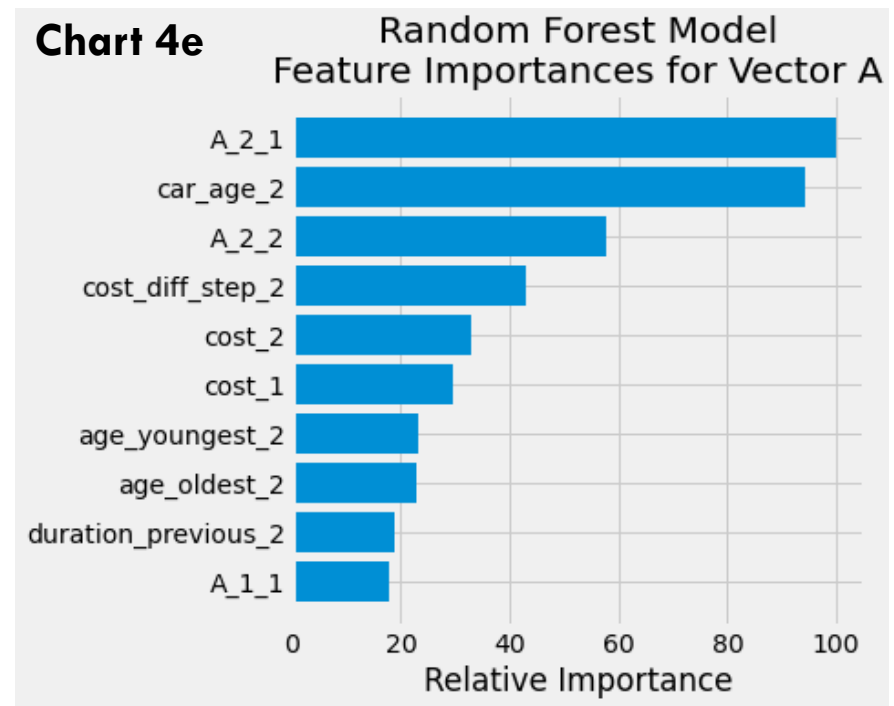
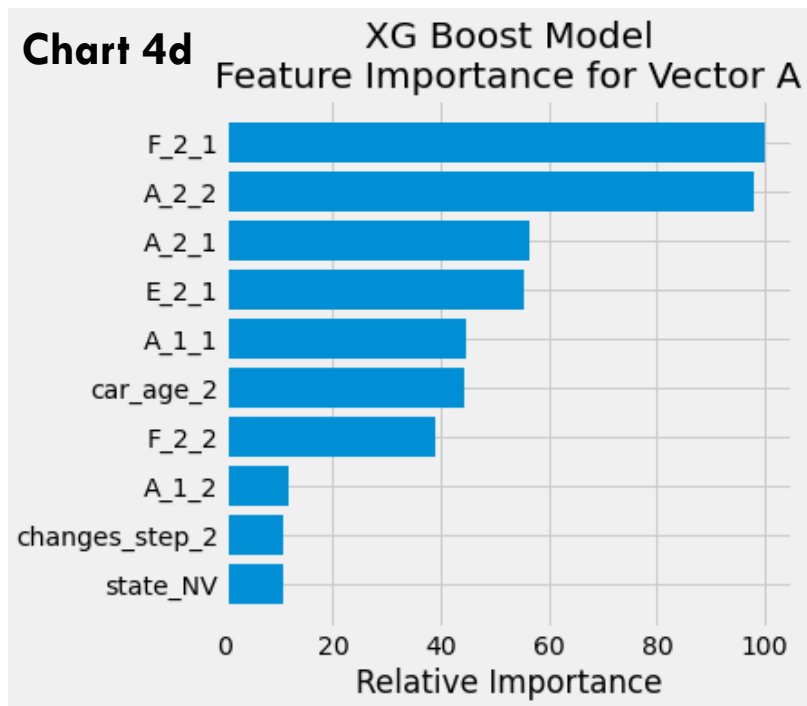
VECTOR A: MODELS PERFORMED BETTER THAN BASELINE ACROSS ALL SHOPPING POINTS

- **Chart 4c** shows that the models had improved predictions for customers purchasing at all shopping points
- **Especially** noteworthy is the superior performance between shopping points 7 to 11
 - Based on knowledge of only shopping point 1 and 2, the classifiers could predict customers much further out in the shopping window



VECTOR A: FEATURE IMPORTANCES OF TOP 2 MODELS

- **Only 4** amongst top 10 features were common between XGBoost (score 84.23%) and Random Forest(score: 84.17%)
 - These were: A_2_1, A_2_2, car_age_2, A_1_1
- **Engineered features** (changes_step_2) and (cost_diff_step_2) are present in top 10
- Could the models be **combined** to improve performance ?

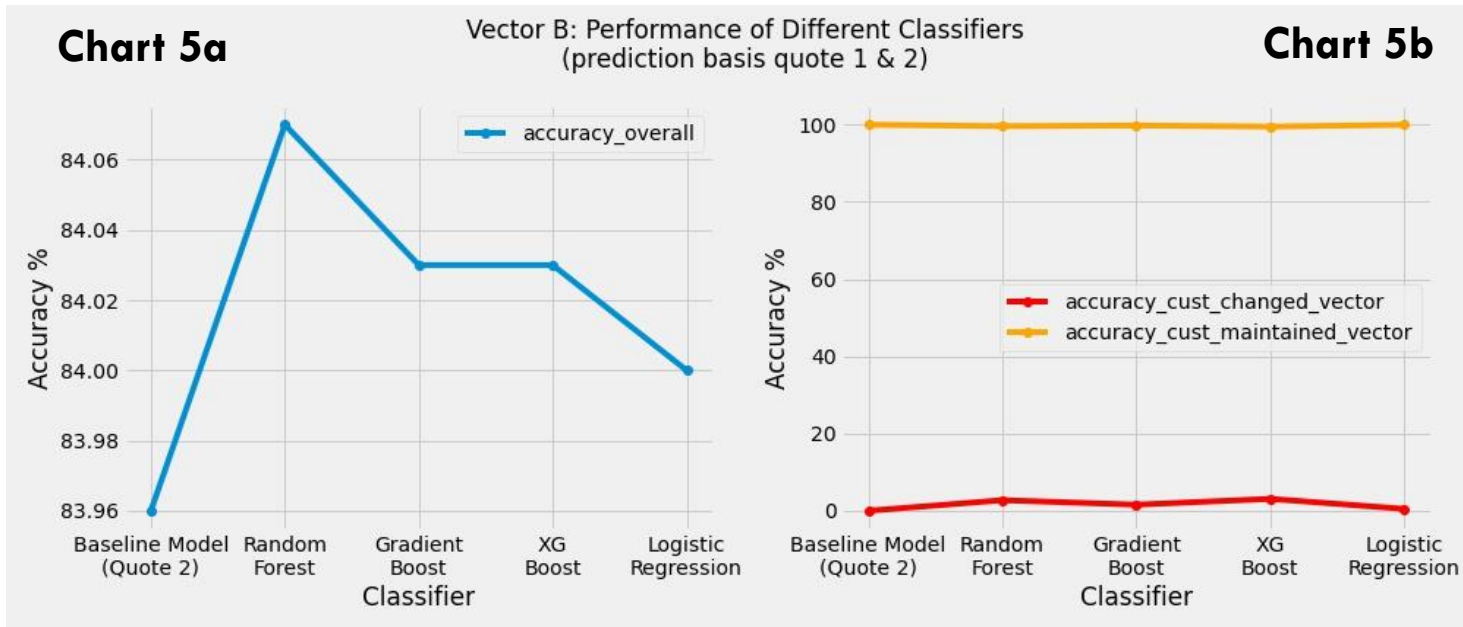


VECTOR B: MODEL & PERFORMANCE

- 'baseline_model' score: Table 2 shows that predicting $B = B_2$ gives a **83.96%** accuracy
- **Chart 5a** shows that the best classifier (Random Forest) had a score of **84.07%**, a **very small improvement**

Table 2: B_2 and B

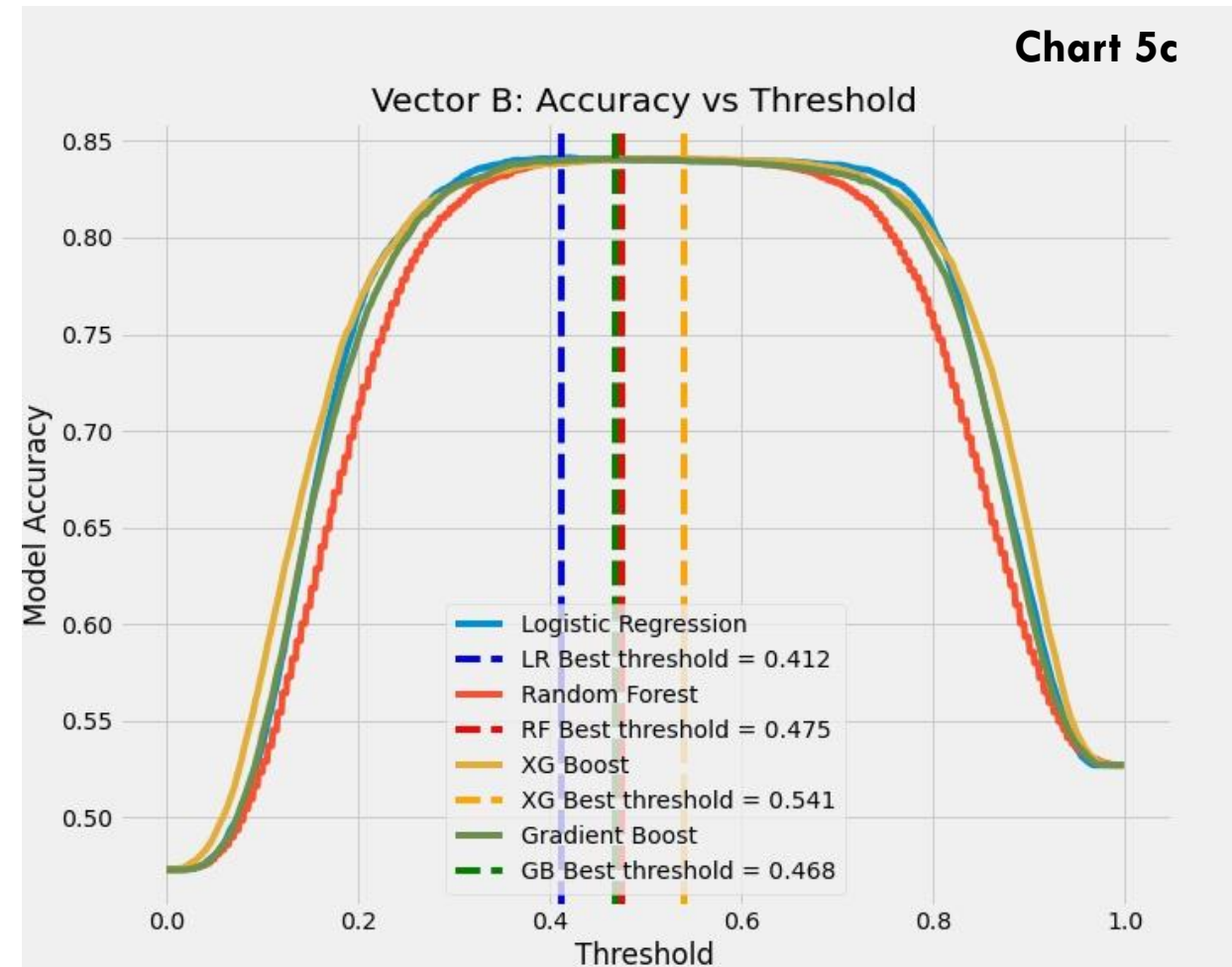
	Class 0	Class 1	total
B_2 →	0	1	
B ↓			
Class 0	9018	1208	10226
Class 1	1904	7272	9176



$$\begin{aligned}\text{Micro Accuracy Score} &= \frac{(9018 + 7272)}{(10226 + 9176)} \\ &= \mathbf{83.96\%}\end{aligned}$$

VECTOR B: THRESHOLDING

- **Thresholding made improvements to all classifiers** with Logistic regression returning a score of **84.14%**, an improvement of **0.18%** over baseline.
- Vector B turned out to be the **hardest** to make improvements





FINDINGS

KEY FINDINGS

- The models gave a performance improvement ranging from **0.18% to 2.8%** over the baseline
- Random Forest & XGBoost were the best classifiers, each being the best for 3 of the 7 vectors
- The average improvement was **1.59%**, the best performance was for **vector C at 2.8%**
- For customers that changed vector choice, the performance improvement ranged from 0.48% to 27.02% for vector C. On average, the models predicted **14.72%** of such customers correctly.

Table 3	recall_baseline_model	best_model	recall_best_model	improvement_over_baseline	recall_cust_changed_vector	recall_cust_maintained_vector
vector						
A	81.91	xg	84.23	2.32	21.11	98.17
B	83.96	lr	84.14	0.18	0.48	99.96
C	80.18	xg	82.98	2.80	27.02	96.81
D	85.09	rf	86.71	1.62	16.29	99.04
E	83.80	xg	84.72	0.92	13.07	98.58
F	81.04	rf	82.63	1.59	11.36	99.30
G	74.31	rf	75.98	1.67	11.46	98.28



IDEAS FOR FURTHER RESEARCH |

IDEAS FOR FURTHER DEVELOPMENT

- Creating an ensemble of different classifiers based on certain voting rules or by combining the probabilities of different models
- More engineered features: e.g. time difference between quotes
- More data to get the models to learn better about features like location