Fan Season Ticket Holder and Rejecter Segmentation & Classification

Udacity Machine Learning Engineer Nanodegree Capstone

Tejas Bala

Table of Contents

Introduction

I currently work within the Business Intelligence Department for the New Orleans Pelicans basketball team. The department focuses on analytics that span ticket sales, digital media, marketing and corporate partnerships. We constantly strive to provide our sales team with new tools and analysis to make their efforts more efficient: one major project there is lead scoring. There are several approaches to a problem like this, one being a Recency-Frequency-Monetary Value model and another being a likeliness model that leans on machine learning.

We have a dataset that tracks Customer Journey across several main data entities: SeatGeek tracks primary and some secondary ticket sale data, CRM tracks sales representative interactions, Marketo tracks email and web interactions (when captured), Yinzcam tracks mobile app interactions and Fanatics tracks online merchandise purchase behavior.

I have access to a dataset and approval to use it for the purpose of a machine learning model for my capstone for this course. I have been requested to not use any Personally Identifiable Information and to only provide a sample of the rest of the data. I hope that this is not an issue if notebooks and documentation are available for evaluation.

Problem Statement

Predicting revenue and product demand is a core requirement for optimizing efforts and maximizing profits. In sports, ticket packages are the equivalent to a subscription whereas your single game is a one-time purchase and it is a constant process to try to assess and convert our single game buyers into plan holders.

How can we predict what a single game buyer's likelihood is to buy a ticket plan? Can we segment our Season Ticket Plan Holders and compare our Single Game Buyers to them to see who is the most probable to convert? Can we predict who will buy Season Ticket and reject Season Tickets?

Datasets and Inputs

Below are previews of the Customer Journey tables from each data entity before aggregation queries.

SeatGeek Customer Journey

SSB_CRMS	cjsgClientG	cjsgClientC	cjsgTeam	cjsgSource	cjsgPlatfor	cjsgActivity	cjsgSecond	cjsgSeason	cjsgSeriesSl	cjsgEventN	cjsgEventD	cjsgStandN	cjsgPriceTy	cjsgOrigina	cjsgTransac	cjsgSiteNar	cjsgTicketV	cjsgScanne	cjsgDollarV	cjsgCustJo	cjsgLifeTim
9D7C26CA	0005331F-	51608315	Pelicans	SG	SEATGEEK	Purchase	Transfer	2019-2020	2019-20 Re	19-11/14 -	00:00.0	Balcony En	Single	Season Tick	50:20.0	Internet	2	2	0	1	0
79CEC8C0-	00089D12	51557891	Pelicans	SG	STUBHUB	Purchase	Resale	2018-2019	2018-19 Re	181110 - N	00:00.0	Lower Corr	Single	Season Tick	57:05.0	Internet	2	0	168	1	168
6107AA96-	001AAA2C-	51632505	Pelicans	SG	SEATGEEK	Purchase	Resale	2019-2020	2019-20 Re	20-3/01 - P	00:00.0	Balcony En	Single	Season Tick	45:37.0	Internet	2	2	190	1	190
E0A5290F-	002CA557-	8214542	Pelicans	SG	STUBHUB	Purchase	Resale	2018-2019	2018-19 Re	181116 - N	00:00.0	Balcony En	Single	Season Tick	32:25.0	Internet	11	10	154	1	822
E0A5290F-	002CA557-	8214542	Pelicans	SG	STUBHUB	Purchase	Resale	2018-2019	2018-19 Re	181116 - N	00:00.0	Balcony En	Single	Season Tick	36:03.0	Internet	9	8	126	2	822

Figure 1 - SeatGeek Customer Journey Table

cjsgLifeTim	cjsgLifeTim	cjsgLifeTim	cjsgLifeTim	cjsgLifeTim	cjsgLifeTim	cjsgLifeTim	cjsgLifeTim	cjsgLifeTim	cjsgLifeTim	cjsgLifeTimeNetInvestmentValue
1	2	2	0	1	2	2	0	0	0	0
1	2	0	168	1	2	0	0	0	0	168
1	2	2	190	1	2	2	0	0	0	190
7	48	45	822	7	48	45	0	0	0	822
7	48	45	822	7	48	45	0	0	0	822

Figure 2 - SeatGeek Customer Journey Table cont.

Marketed Customer Journey

SSB_CRMSV	cjmktLeadI	cjmktEmail	cjmktTeam	cjmktSourc	cjmktPlatfc	cjmktActiv	cjmktSecor	cjmktSeaso	cjmktSerie	cjmktActiv	cjmktActiv	cjmktStanc	cjmktPrice	cjmktOrigi	cjmktActiv	cjmktSiteN	cjmktActiv	cjmktScanr	cjmktDolla	cjmktCustJ	cjmktLifeTimeTotalActivityVolu
207FE5E3-(	1734981	REDACTED	Other	Marketo	MARKETO	CRM Sync	NULL	NULL	NULL	Contact	58:47.0	NULL	NULL	NULL	9/14/15	NULL	1	NULL	NULL	1	2
207FE5E3-(	1734981	REDACTED	Other	Marketo	MARKETO	Email Boun	NULL	NULL	NULL	Insider 9/1	46:07.0	NULL	NULL	NULL	9/18/15	NULL	1	NULL	NULL	2	2
6B48F545-	1734982	REDACTED	Saints	Marketo	MARKETO	Open Email	NULL	NULL	NULL	1/12/16 Sa	40:59.0	NULL	NULL	NULL	1/12/16	NULL	1	NULL	NULL	38	73
6B48F545-	1734982	REDACTED	Saints	Marketo	MARKETO	Open Email	NULL	NULL	NULL	1/21/16 Sa	22:29.0	NULL	NULL	NULL	1/21/16	NULL	1	NULL	NULL	39	73
6B48F545-	1734982	REDACTED	Saints	Marketo	MARKETO	Open Email	NULL	NULL	NULL	1/28/16 Sa	32:32.0	NULL	NULL	NULL	1/28/16	NULL	1	NULL	NULL	40	73

Figure 3 - Marketo Customer Journey Table

Yinzcam Customer Journey

SSB_CRMSV	cjyzYinzId	cjyzEmail	cjyzTeamOr	cjyzSource	cjyzPlatfor	cjyzActivity	cjyzSecond	cjyzSeason	cjyzSeriesSI	cjyzActivity	cjyzActivity	cjyzStandN	cjyzPriceTy	cjyzOrigina	cjyzActivity	cjyzSiteNar	cjyzActivity	cjyzScanne	cjyzDollarV	cjyzCustJou	cjyzLifeTimeTotalActivityVolum
7E1695DE-	50059FB5-	REDACTED	Pelicans	YinzCam	YINZCAM	PAGE_VIEW	NULL	NULL	NULL	Home Scre	25:50.0	NULL	NULL	NULL	2/13/20	NULL	2	NULL	NULL	73	272
7E1695DE-	50059FB5-	REDACTED	Pelicans	YinzCam	YINZCAM	PAGE_VIEW	NULL	NULL	NULL	Home Scre	06:03.0	NULL	NULL	NULL	2/12/20	NULL	8	NULL	NULL	72	272
7E1695DE-	50059FB5-	REDACTED	Pelicans	YinzCam	YINZCAM	PAGE_VIEW	NULL	NULL	NULL	Schedule:S	51:30.0	NULL	NULL	NULL	2/11/20	NULL	4	NULL	NULL	71	272
7E1695DE-	50059FB5-	REDACTED	Pelicans	YinzCam	YINZCAM	PAGE_VIEW	NULL	NULL	NULL	Schedule:S	23:30.0	NULL	NULL	NULL	2/10/20	NULL	2	NULL	NULL	70	272
7E1695DE-	50059FB5-	REDACTED	Pelicans	YinzCam	YINZCAM	PAGE_VIEW	NULL	NULL	NULL	Home Scre	23:23.0	NULL	NULL	NULL	2/10/20	NULL	2	NULL	NULL	69	272

Figure 4 - Yinzcam Customer Journey Table

Fanatics Customer Journey

SSB_CRMSV	cjftsClientI	cjftsClientE	cjftsTeam	cjftsSource	cjftsPlatfor	cjftsActivit	cjftsSecond	cjftsSeason	cjftsProduc	cjftsProduc	cjftsOrderD	cjftsProduc	cjftsProduc	cjftsSize	cjftsOrderD	cjftsPlayerI	cjftsProduc	cjftsScanne	cjftsDollarV	cjftsCustJo	cjftsLifeTim	cjftsLifeTim
8A16432E-	1.64E+08	REDACTED	Pelicans	Fanatics	FANATICS	Purchase	NULL	NULL	TEE	Men's Nike	00:17.8	PFM	MEN	M	11/27/17	NULL	1	NULL	34.99	1	34.99	1
8C5F2211-	1.64E+08	REDACTED	Pelicans	Fanatics	FANATICS	Purchase	NULL	NULL	TEE	Men's Fana	54:17.3	SCR	MEN	XXL	11/27/17	DeMarcus (	1	NULL	24.99	1	24.99	1
B5A70B8A-	1.64E+08	REDACTED	Pelicans	Fanatics	FANATICS	Purchase	NULL	NULL	TEE	Men's Fana	22:30.2	L-S	MEN	M	11/27/17	NULL	1	NULL	29.99	1	29.99	1
BB81F8B3-	1.64E+08	REDACTED	Pelicans	Fanatics	FANATICS	Purchase	NULL	NULL	SWT	Men's Red I	56:47.8	HOD	MEN	M	11/27/17	NULL	1	NULL	59.99	1	59.99	1
0E44CA8E-	1.64E+08	REDACTED	Pelicans	Fanatics	FANATICS	Purchase	NULL	NULL	TEE	Mens New (	10:57.5	NUM	MEN	S	11/27/17	Jrue Holida	1	NULL	27.99	1	27.99	1

Figure 5 - Fanatics Customer Journey Table

CRM Customer Journey

SSB_CRMSV	cjcrmConti	cjcrmConti	cjcrmTeam	cjcrmSourc	cjcrmPlatfc	cjcrmActiv	cjcrmSecor	cjcrmSeaso	cjcrmSerie	cjcrmActiv	cjcrmActiv	cjcrmStanc	cjcrmPrice	cjcrmOrigi	cjcrmActiv	cjcrmActiv	cjcrmActiv	cjcrmScanr	cjcrmDolla	cjcrmCustJ	cjcrmLifeTimeTotalActivityVol
F66C7C67-	6C229066-	REDACTED	NULL	CRM_Conta	CRM	CRM Create	NULL	NULL	NULL	CRM_Conta	54:53.0	Active	NULL	NULL	11/27/17	Pelicans Ad	1	NULL	NULL	1	1
86B89E57-	6E229066-	REDACTED	NULL	CRM_Conta	CRM	CRM Create	NULL	NULL	NULL	CRM_Conta	54:53.0	Active	NULL	NULL	11/27/17	Pelicans Ad	1	NULL	NULL	1	1
E844920F-	70229066-	REDACTED	NULL	CRM_Conta	CRM	CRM Create	NULL	NULL	NULL	CRM_Conta	54:54.0	Active	NULL	NULL	11/27/17	Pelicans Ad	1	NULL	NULL	1	1
97CECC26-	72229066-	REDACTED	NULL	CRM_Conta	CRM	CRM Create	NULL	NULL	NULL	CRM_Conta	54:54.0	Active	NULL	NULL	11/27/17	Pelicans Ad	1	NULL	NULL	1	1
8A2CD1C8-	74229066-	REDACTED	NULL	CRM_Conta	CRM	CRM Create	NULL	NULL	NULL	CRM_Conta	54:54.0	Active	NULL	NULL	11/27/17	Pelicans Ad	1	NULL	NULL	1	1

Figure 6 - CRM Customer Journey Table

Plan Rejecter List			
SSB_CRMSV	client_prospectdevelopmentname		
B7267635-	Lost		
D25310D8	Lost		
35E44FCE-(	Lost		
70E892A6-	Lost		
79788D7C-	Lost		

Figure 7 - Plan Rejecter List

Data Wrangling

To see the queries, reference localLibrary_dataQueries.py.

The desired output was 3 subsets of data: STM (Season Ticket Member), Lost (Plan Rejecters or Lost Accounts), nonSTM (Ticket Buyers that are neither STM or Lost).

The Customer Journey tables are grouped by major activity type across data entities and metrics aggregated.
STM data set are aggregated on all activity before the date of their first Season Ticket product. nonSTM data aggregates all activity since there is no date of purchase yet. Lost data aggregates all activity as well to capture those fans that continue to buy game tickets but have rejected season ticket plans.

To see the execution of queries reference data_acquisition.py.

The extracted CSV files were then uploaded to an S3 bucket.

Data Cleaning & Exploration

The first major step in cleaning and transformation is merging the datasets from different entities (SG, MKT, FTS, YZ, CRM) together per fan type (STM, nonSTM, Lost). From my prior knowledge of the data, I understand that the universe of fans is much smaller in SeatGeek than in Marketo or CRM, while I am unsure about Yinzcam and Fanatics. Either way we are specific to the universe of ticket buyers meaning across the data entities we only take the fans that are present in the SeatGeek dataset.

One important step with the SeatGeek data since it is in the format below where a fan can have multiple rows based on activity type and ticket type is to pivot so that each fan has one row with columns that represent all activities and ticket types. After the pivot is complete we take the fan's absolute minimum SeatGeek date of engagement and absolute maximum.

SSB_CRMSYSTEM_CONTACT_ID	Activity Type	Ticket Type	Metrics…
12345	Purchase	Primary	
12345	Sell	Primary	

Figure 8 - SeatGeek Before Pivot

Next step for the data is to clean up the date columns. These columns are in the format "YYYY-MM-DD HH:MM:SS" and we would like to transform them into a quantifiable value for model ingestion. A straightforward approach that dignifies the linear nature of the dates is to convert to in as is (YYYYMMDDHHMMSS). Once this is done we can create a feature for time difference that represents the length of engagement that a fan has within a data entity.

Once the data features are complete we can look at all of the features and determine if we need to drop any due to majority null. The figure below shows that we should drop all columns for Fanatics, Yinzcam and Secondary Market Ticket Transactions.

	STM_Pct_Null	nonSTM_Pct_Null	lost_Pct_Null
total_scanned	0.991327	0.988641	0.978643
secondary_sell_tickets	0.990306	0.977740	0.980927
secondary_sell_transactions	0.990306	0.977740	0.980927
secondary_sell_dollars	0.990306	0.977740	0.980927
secondary_purchase_transactions	0.985714	0.970766	0.946340
secondary_purchase_dollars	0.985714	0.970766	0.946340
secondary_purchase_tickets	0.985714	0.970766	0.946340
TotalYinzcamVolume	0.975510	0.975359	0.961316
LatestYinzcam_int	0.975510	0.975359	0.961316
EarliestYinzcam_int	0.975510	0.975359	0.961316
Yinzcam_diff	0.975510	0.975359	0.961316
TotalFanaticsProductQty	0.945918	0.963213	0.941840
TotalFanaticsDollarValue	0.945918	0.963213	0.941840
Fanatics_diff	0.945918	0.963213	0.941840
LatestFanatics_int	0.945918	0.963213	0.941840
EarliestFanatics_int	0.945918	0.963213	0.941840
TotalFanaticsTransactions	0.945918	0.963213	0.941840

Figure 9 - Columns Percent Null

Based on my understanding of the dataset and investigation and exploration, I understand the data contains a lot of incomplete records and thus I made the decision after dropping these columns to also drop all incomplete rows in the interest of a more robust model. This was feasible because of the size of the dataset and the amount of remaining rows.

```
23 #NUMBER OF ROWS WITHOUT MISSING VALUES
24 print(f"STM full rows count: {(STM.dropna().shape[0])}")
25 print(f"nonSTM full rows count: {(nonSTM.dropna().shape[0])}")
26 print(f"lost full rows count: {(lost.dropna().shape[0])}")
```

```
STM full rows count: 1499
nonSTM full rows count: 26497
lost full rows count: 5318
```

Figure 10 - Amount of full rows

Lastly, I took a look at the feature distributions across the fan types to see if I could pull out any high-level distinctions that would help with the clustering analysis. Nothing immediately jumped out but it was helpful to reference as I continued through the next portion.

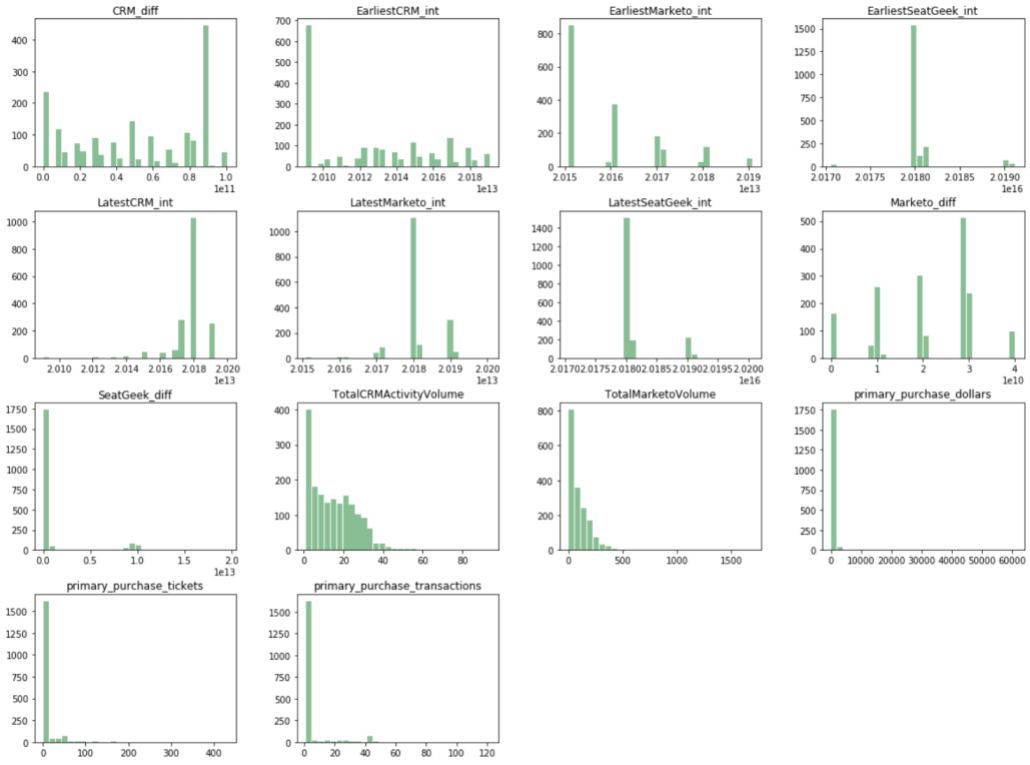


Figure 11 - STM Feature Distribution

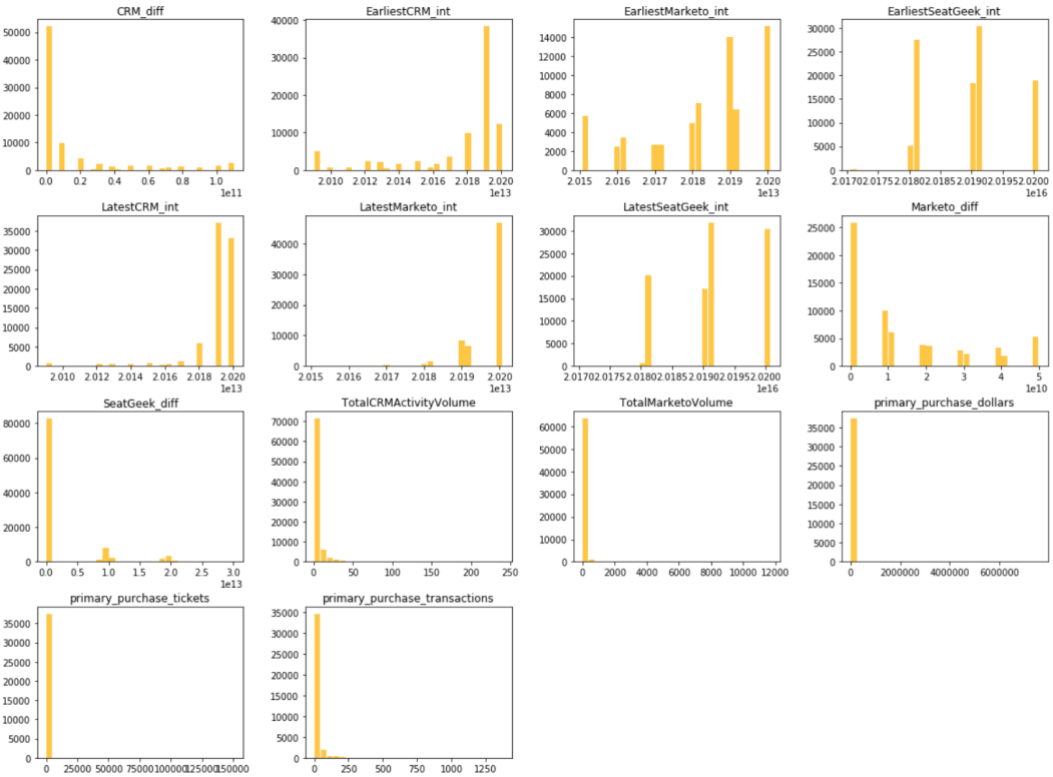


Figure 12 - nonSTM Feature Distribution

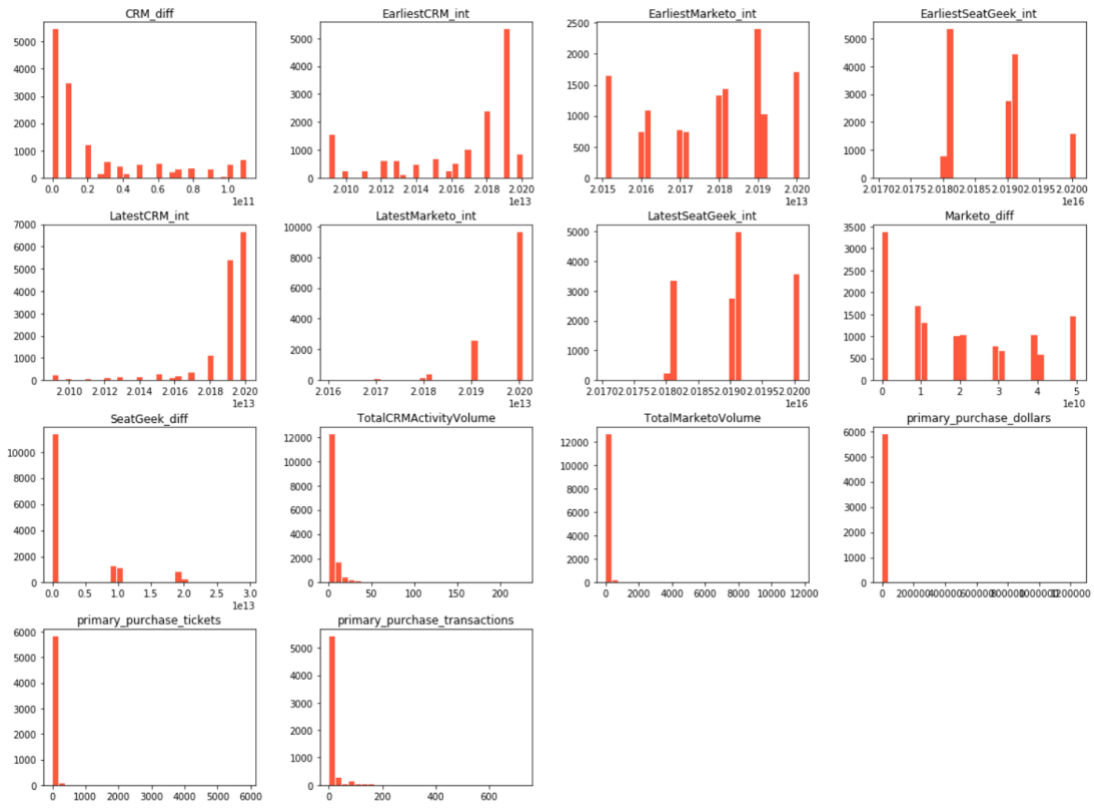


Figure 13 - Lost Feature Distribution

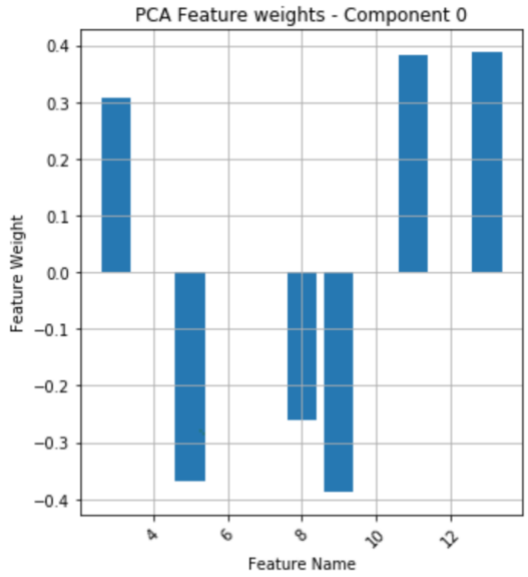
Fan Segmentation: Clustering Analysis

Before Cluster Analysis I standardized all of the features to give them equal weights as well as maximizing relative difference within a feature. After standardization I could perform PCA to reduce the dimension space a bit. One important note here, I chose not to remove correlated features for PCA as correlated features will group in components and strengthen the underlying data trend. See the figure below for the explained variance of the components. I determined 5 components is a good balance of reducing dimensions and explaining over 75% of the variance in the data.

```
array([0.31, 0.48, 0.61, 0.7 , 0.77, 0.84, 0.89, 0.93, 0.96, 0.99, 1.  ,
       1.  , 1.  , 1.  ])
```

Figure 14 - PCA Explained Variance

With PCA interpretability is very important since we are taking the raw features and turning them into components to represent the underlying data trend. See below for how this is displayed for a component.



	Feature	Description	FeatureWeight
2	13	Marketo_diff	0.388679
1	11	CRM_diff	0.382082
0	3	primary_purchase_dollars	0.308491
5	8	EarliestSeatGeek_int	-0.260681
4	5	EarliestCRM_int	-0.369438
3	9	EarliestMarketo_int	-0.387960

Figure 15 - PCA Explanation

For the Clustering Analysis since we have two "labelled" datasets (STM, Lost) I decided to create segmentations for both and apply our "population" (nonSTM) to those clusters.

In creating the STM Cluster, I looked at an Elbow graph to determine the proper k-value, no apparent "elbow" stuck out so I chose k=5 based on my knowledge of the domain (and also thinking about how to further explain to the sales staff 5 seemed like a good number of groups to determine defining characteristics). Before analyzing the clusters, I did the same for the Lost dataset. See below for an example of the silhouette measure and elbow graph used to determine an effect k-value.

```
For n_clusters = 2 The average silhouette_score is : 0.4164683719195629
For n_clusters = 3 The average silhouette_score is : 0.39526635920227876
For n_clusters = 4 The average silhouette_score is : 0.2921666505383061
For n_clusters = 5 The average silhouette_score is : 0.3088263590632571
For n_clusters = 6 The average silhouette_score is : 0.3197744058220088
For n_clusters = 7 The average silhouette_score is : 0.33864607318555245
For n_clusters = 8 The average silhouette_score is : 0.2987975503254404
For n_clusters = 9 The average silhouette_score is : 0.29091201166837116
```

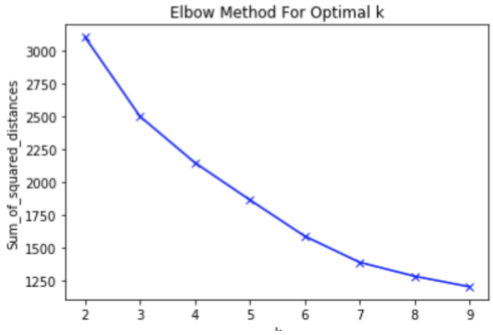


Figure 16 - STM Cluster Elbow Graph

After applying the nonSTM data to both the STM and Lost clusters we are able to pull out some high-level insights.

As expected, the vast majority of our general population will not fall in our most popular STM buckets. One insight that immediately jumps out is that the ~100 fans in Cluster 4 we should pursue heavily since a vast majority of our STM customers are similar to them, and same for the fans in Cluster 0.

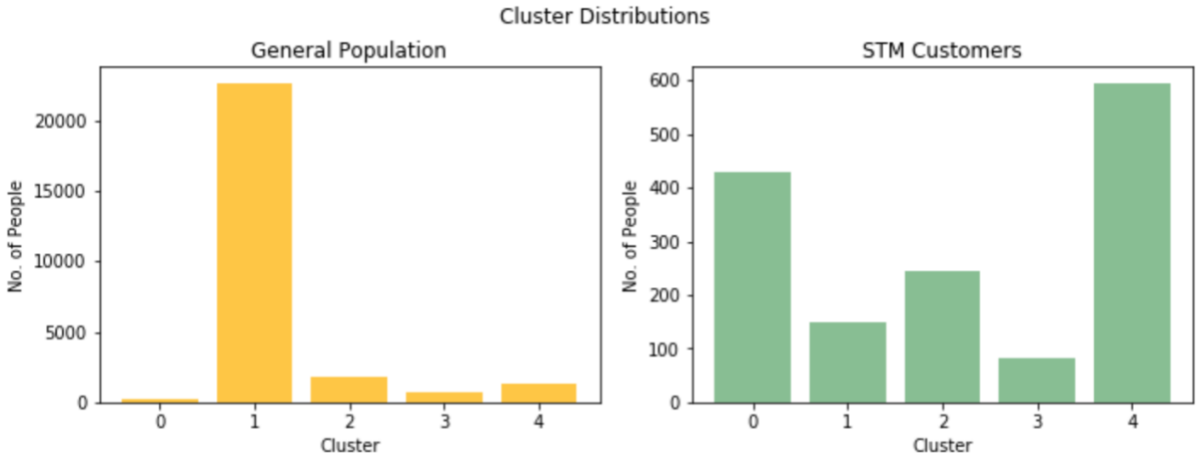


Figure 17 - STM Cluster vs Population

Below we do the same for our Lost Clusters. This distribution looks almost identical to our General Population which makes sense but brings about the question of 'who are our fringe cases and how can we convert them?'

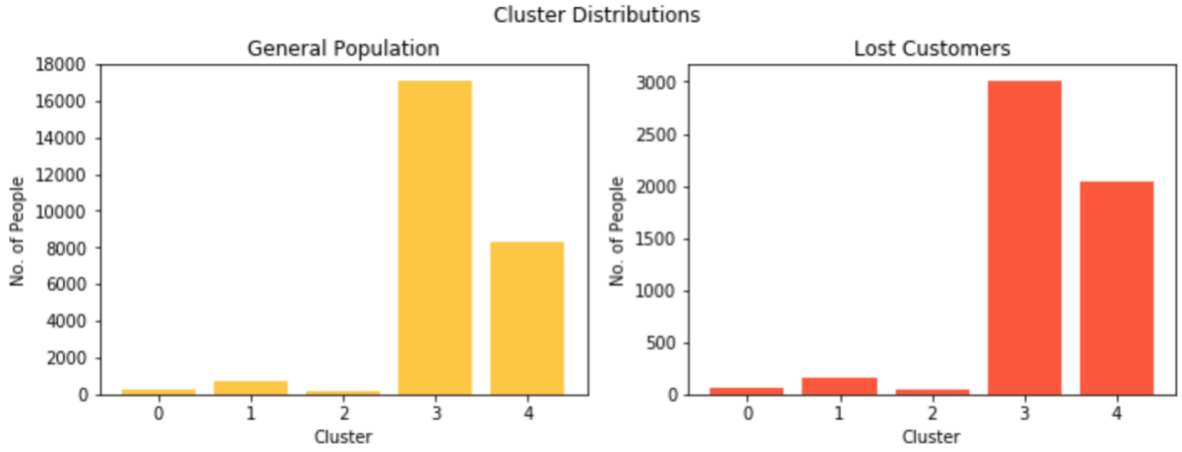


Figure 18 - Lost Cluster vs Population

For one more view at the data we take a look at the proportions rather than the raw values. In our STM graph the bars above the dashed line show the clusters that we have a better chance of converting fans into season ticket holders. Conversely, in the Lost graph the bars below the dash line show the clusters where we have a better chance of converting fans into season ticket holders.

Figure 19 - Cluster Proportion Analysis

Lastly, I wanted to be able to understand which components and which raw features when into each cluster and how it affected the breakouts. See below for an example of a table generated for this deeper understanding.

	Component	ComponentWeight	Feature	FeatureWeight
0	2	1.951816	primary_purchase_tickets	0.539049
1	2	1.951816	primary_purchase_dollars	0.537877
2	2	1.951816	LatestMarketo_int	-0.350761
3	2	1.951816	LatestSeatGeek_int	-0.362785
4	0	0.929874	Marketo_diff	0.388679
5	0	0.929874	CRM_diff	0.382082
6	0	0.929874	EarliestCRM_int	-0.369438
7	0	0.929874	EarliestMarketo_int	-0.387960
8	4	-0.248109	EarliestSeatGeek_int	0.531913
9	4	-0.248109	CRM_diff	0.406607
10	4	-0.248109	EarliestCRM_int	-0.268640
11	4	-0.248109	SeatGeek_diff	-0.461479

Figure 20 - Cluster Explanation Table

Season Ticket Buyer Classification: Supervised Learning

For model training we took only the data from STM and Lost as we want to classify any new data as one or the other. We remain with ~6.5K records to train on, of which ~20% are STM. I kept this class imbalance in mind as I proceeded.

```
Rejecter    5318
STM         1499
Name: target, dtype: int64

Rejecter    0.780109
STM         0.219891
Name: target, dtype: float64
```

Figure 21 - Data Size & Proportion

I held out 10% of the data as a holdout set to test my best model after a kfold evaluation of 6 models. I chose Logistic Regression, Decision Tree, Random Forrest, Gradient Boost, AdaBoost, XGBoost. This was based on my understanding of boosting algorithms and the benefits of them as well as interpretability of the model output. One concern was that my holdout, train and test sets would have different proportions of data.

```
Holdout Class Proportion:
Rejecter    0.778592
STM         0.221408
Name: target, dtype: float64

Train Class Proportion:
Rejecter    0.77934
STM         0.22066
Name: target, dtype: float64

Test Class Proportion:
Rejecter    0.784026
STM         0.215974
Name: target, dtype: float64
```

Figure 22 - Holdout, Train, Test Proportions

After running k-fold (with 5 splits) analysis, I found the average AUROC score for each model.

	Model	AUCROC_score
0	LogisticRegression	0.5
1	DecisionTreeClassifier	0.968825
2	RandomForestClassifier	0.997741
3	GradientBoostingClassifier	0.998576
4	AdaBoostClassifier	0.997389
5	XGBClassifier	0.998871

Figure 23 - Model Results

The results very much shocked me, I was not expecting a model to be as accurate these are showing to be. I chose AUROC as the ROC is a common evaluation method for classification problems. I tested the model on the holdout set that also returned an AUROC of **.99**. I then

tested the model on biased datasets of the holdout set filtered to just STM and just lost. Both had AUROC of ~.99.

For interpretability I also plotted the output of a trimmed Decision Tree. See below.

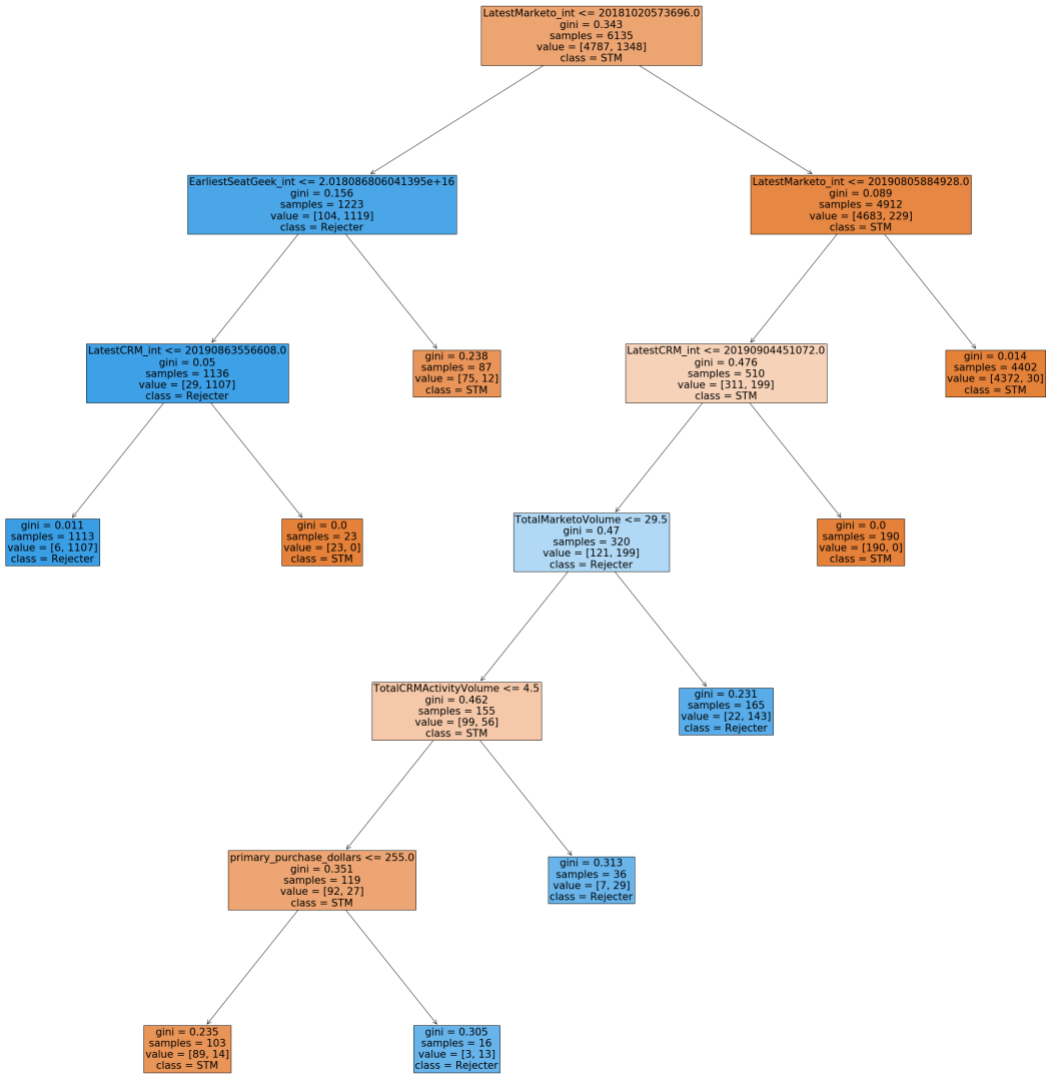


Figure 24 - Decision Tree Output

Improvements and Future Steps

The next step would be to predict on nonSTM and track accuracy as the sales team makes their efforts this upcoming season.

One major improvement would be to integrate some PII information: demographic and sale rep input. Additionally, having a date for when an account was "Lost" would be great and the model can turn to predicting *when* a fan will convert to a STM or Lost.