Visit with us

Travel Package Purchase Prediction

PG-DSBA Project 5
Eric Green
April 2020

Background

In order to continue evolving the Visit with us business model and strengthen its market share, a new travel package is being launched which focuses on Wellness and Well-being. The strategic objective is to use existing customer data to model and predict the characteristics of customers who convert over in purchasing a travel product.

Through modeling we determine which of the customer characteristics are most influential to buying to subsequently use this information to adjust marketing and operational policies to increase revenue and market share through the successful launch of the new wellness travel package.

The findings and business recommendations in this presentation provide support for this.

Objectives

- ✓ Harness customer data to make marketing expenses more capitally efficient by targeting the best customers for promotion
- ✓ Develop a model to predict customers most likely to buy the new wellness travel package
- ✓ Identify policy changes and other business recommendations to implement an adaptive business model through successful deployment of new Wellness travel package
- ✓ Expand customer base through introduction of new travel package

Data Summary

Raw data shape: 4888 (rows) x 20 (columns)

1. **CustomerID**: Unique customer ID **(4888 non-null int64)**
2. **ProdTaken**: Product taken flag **(4888 non-null int64)**
3. **Age**: Age of customer **(4662 non-null float64**
4. **TypeofContact**: How customer was contacted - Company Invited or Self Inquiry **(4863 non-null object)**
5. **CityTier**: City tier **(4888 non-null int64)**
6. **Occupation**: Occupation of customer **(4888 non-null object)**
7. **Gender**: Gender of customer **(4888 non-null object)**
8. **NumberOfPersonVisited**: Total number of person came with customer **(4888 non-null int64)**
9. **PreferredPropertyStar**: Preferred hotel property rating by customer **(4862 non-null float64)**
10. **MaritalStatus**: Marital status of customer **(4888 non-null object)**
11. **NumberOfTrips**: Average number of the trip in a year by customer **(4748 non-null float64)**
12. **Passport**: The customer has passport or not **(4888 non-null int64)**
13. **OwnCar**: Customers owns a car flag **(4888 non-null int64)**
14. **NumberOfChildrenVisited**: Total number of children visit with customer **(4822 non-null float64)**
15. **Designation**: Designation of the customer in the current organization **(4888 non-null object)**
16. **MonthlyIncome**: Gross monthly income of the customer **(4655 non-null float64)**
17. **PitchSatisfactionScore**: Sales pitch satisfactory score **(4888 non-null int64)**
18. **ProductPitched**: Product pitched by a salesperson **(4888 non-null object)**
19. **NumberOfFollowups**: Total number of follow ups by sales person after sales pitch **(4843 non-null float64)**
20. **DurationOfPitch**: Duration of the pitch by a salesman to customer **(4637 non-null float64)**

Null Values	Unique Values
CustomerID 0 ProdTaken 0 Age 226 TypeofContact 25 CityTier 0 DurationOfPitch 251 Occupation 0 Gender 0 NumberOfPersonVisited 0 NumberOfFollowups 45 ProductPitched 0 PreferredPropertyStar 26 MaritalStatus 0 NumberOfTrips 140 Passport 0 PitchSatisfactionScore 0 OwnCar 0 NumberOfChildrenVisited 66 Designation 0 MonthlyIncome 233	CustomerID 4888 ProdTaken 2 Age 44 TypeofContact 2 CityTier 3 DurationOfPitch 34 Occupation 4 Gender 3 NumberOfPersonVisited 5 NumberOfFollowup 6 ProductPitched 5 PreferredPropertyStar 3 MaritalStatus 4 NumberOfTrips 12 Passport 2 PitchSatisfactionScore 5 OwnCar 2 NumberOfChildrenVisited 4 Designation 5 MonthlyIncome 2475

Observations

- Raw data has significant problems which require thorough preprocessing prior to EDA and modeling
- There are numerous nulls/na values observed across the data variables indicated above
- Variables not needed: CustomerID, NumberOfChildrenVisited
- Categorical variables are: TypeofContact, CityTier, Occupation, Gender, MaritalStatus, ProductPitched, PitchSatisfactionScore, Designation, Passport, OwnCar, Age, DurationOfPitch, NumberOfPersonVisited, NumberOfFollowups, PreferredPropertyStar, NumberOfTrips , **ProdTaken (target feature)**
- Continuous variables are: MonthlyIncome

Data Preprocessing

Stage 1 preprocessing

Data shape: 4188 (rows) x 18 (columns)

Columns names converted to be intuitive to work with

1. product_conversion
2. customer_age
3. contact_type
4. city_scale
5. pitch_duration
6. occupation
7. gender
8. visitors
9. followups
10. pitched_product
11. property_rating
12. marital_status
13. avg_annual_trips
14. passport
15. pitch_sat_score
16. car_owner
17. job_role
18. monthly_income

Stage 2 preprocessing

Data shape: 4188 (rows) x 39 (columns)

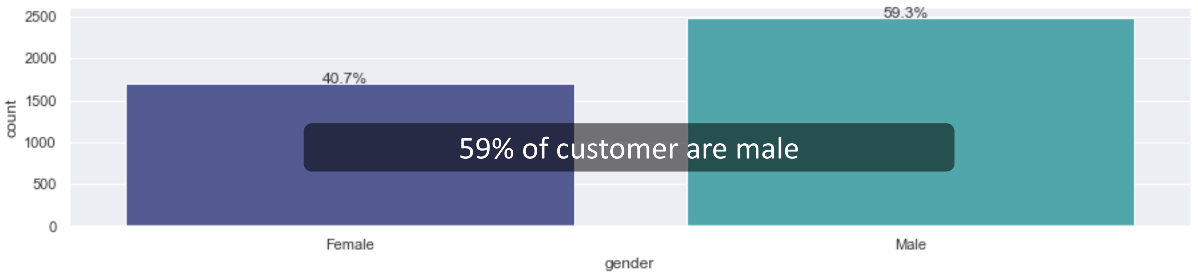
Columns type converted, ns/nulls dropped, dirty values replaced and categorical features one-hot encoded

0 product_conversion 4188 non-null int64
1 customer_age 4188 non-null int64
2 pitch_duration 4188 non-null int64
3 visitors 4188 non-null int64
4 followups 4188 non-null int64
5 property_rating 4188 non-null int64
6 avg_annual_trips 4188 non-null int64
7 passport 4188 non-null int64
8 car_owner 4188 non-null int64
9 monthly_income 4188 non-null float64
10 contact_type_Company_Invited 4188 non-null uint8
11 contact_type_Self_Enquiry 4188 non-null uint8
12 city_scale_1 4188 non-null uint8
13 city_scale_2 4188 non-null uint8
14 city_scale_3 4188 non-null uint8
15 occupation_Free_Lancer 4188 non-null uint8
16 occupation_Large_Business 4188 non-null uint8
17 occupation_Salaried 4188 non-null uint8
18 occupation_Small_Business 4188 non-null uint8
19 gender_Female 4188 non-null uint8
20 gender_Male 4188 non-null uint8
21 marital_status_Divorced 4188 non-null uint8
22 marital_status_Married 4188 non-null uint8
23 marital_status_Unmarried 4188 non-null uint8
24 pitched_product_Basic 4188 non-null uint8
25 pitched_product_Deluxe 4188 non-null uint8
26 pitched_product_King 4188 non-null uint8
27 pitched_product_Standard 4188 non-null uint8
28 pitched_product_Super_Deluxe 4188 non-null uint8
29 pitch_sat_score_1 4188 non-null uint8
30 pitch_sat_score_2 4188 non-null uint8
31 pitch_sat_score_3 4188 non-null uint8
32 pitch_sat_score_4 4188 non-null uint8
33 pitch_sat_score_5 4188 non-null uint8
34 job_role_AVP 4188 non-null uint8
35 job_role_Executive 4188 non-null uint8
36 job_role_Manager 4188 non-null uint8
37 job_role_Senior_Manager 4188 non-null uint8
38 job_role_VP 4188 non-null uint8

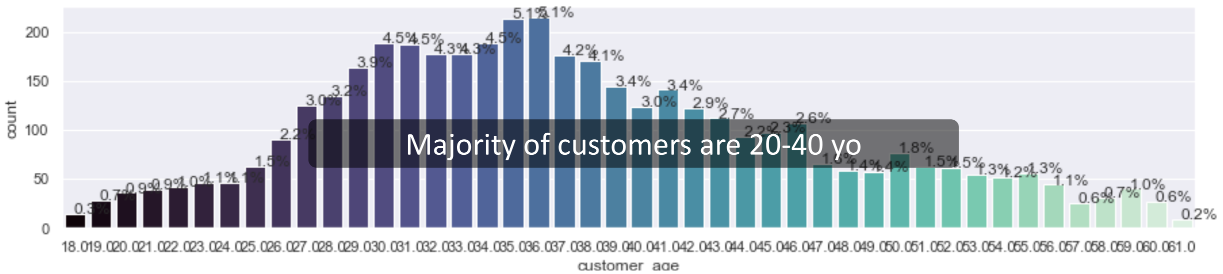
Observations

- Column names were converted to be intuitive to subsequent EDA and modeling work
- CustomerID and NumberOfChildrenVisitied were dropped as they should have little-to-no influence on target variable
- na/nans rows are dropped from the data set, given the sufficiency of the data size (post drop shape 4188 rows x 18 columns)
- 6 total extreme outliers were removed from pitch_duration, avg_annual_trips, monthly_income
- Values for gender, job_role, pitched_product, occupation, contact_type were cleansed to be more intuitive
- Gender value "Single" was converted to "Unmarried" to collapse in the 3 non-overlapping category values
- Overall, initial processed data shape reduced from 4888 rows x 20 columns to 4188 rows x 18 columns to utilize for prediction modeling (product_conversion)
    - This was a conscious decision to enable spending more time on prediction modeling

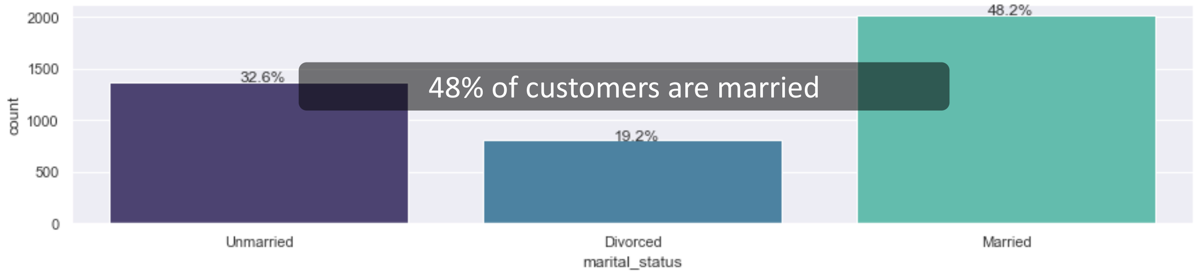
Univariate EDA – Customer Attributes



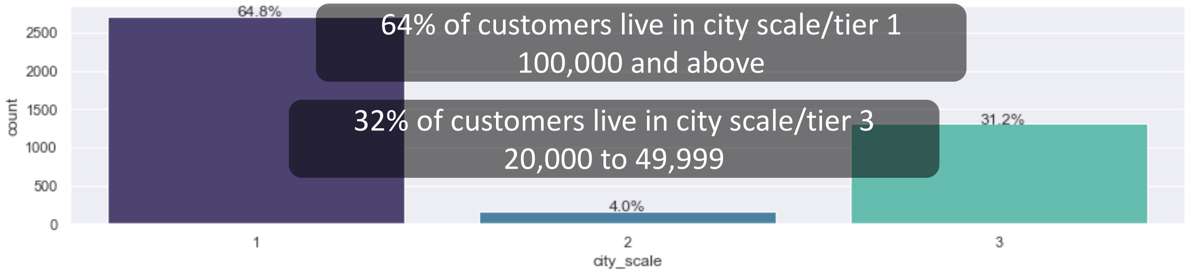
59% of customer are male



Majority of customers are 20-40 yo



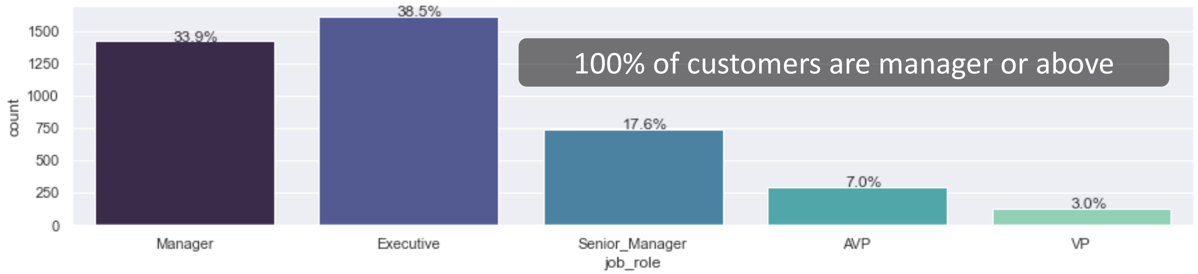
48% of customers are married



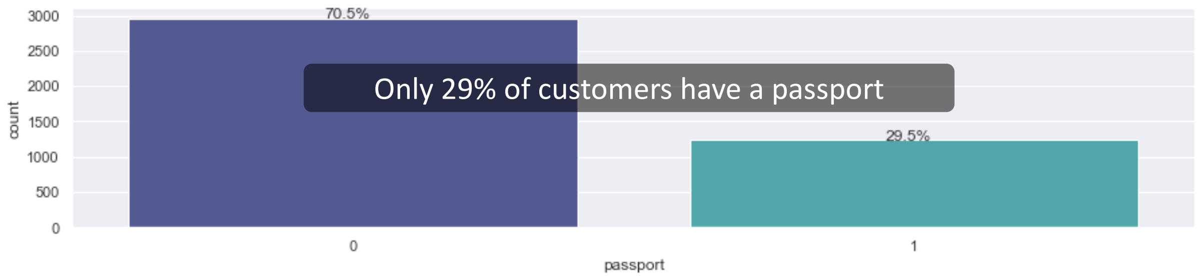
64% of customers live in city scale/tier 1 100,000 and above

32% of customers live in city scale/tier 3 20,000 to 49,999

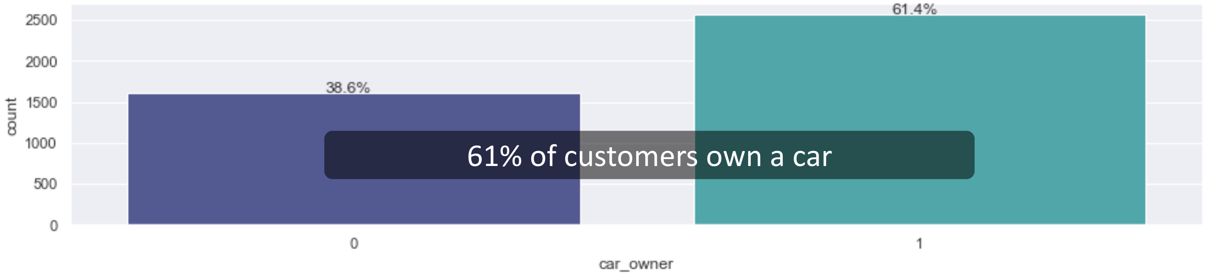
90% of customers identify as salaried/small business



100% of customers are manager or above

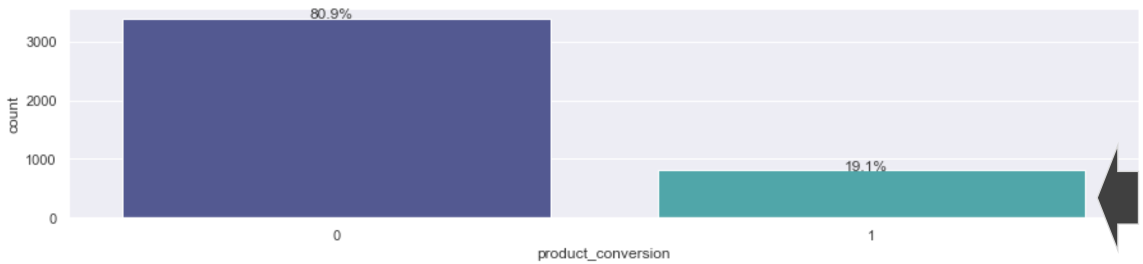


Only 29% of customers have a passport

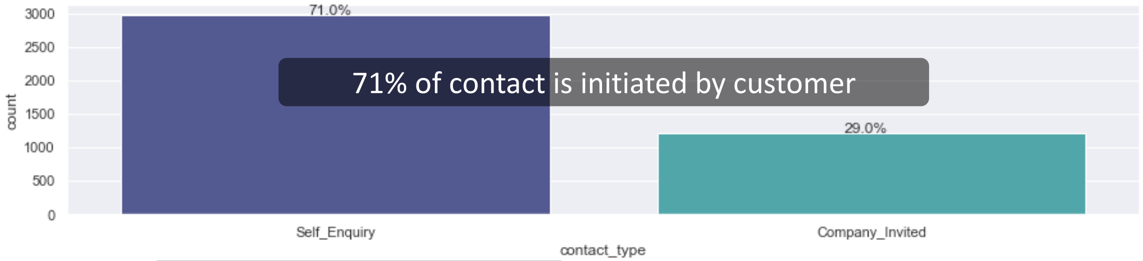


61% of customers own a car

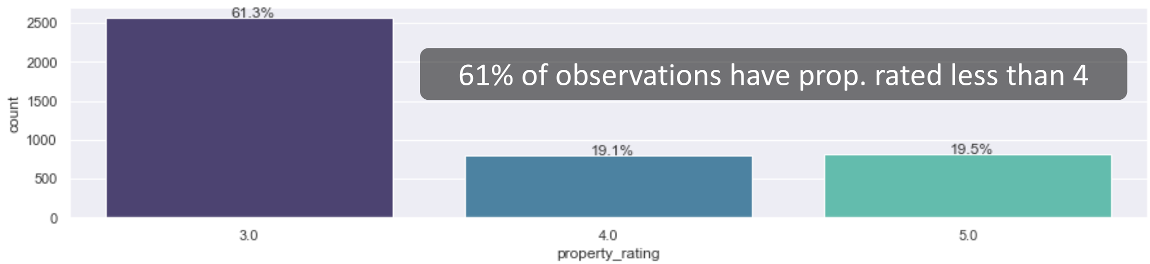
Univariate EDA – Customer Interaction Attributes



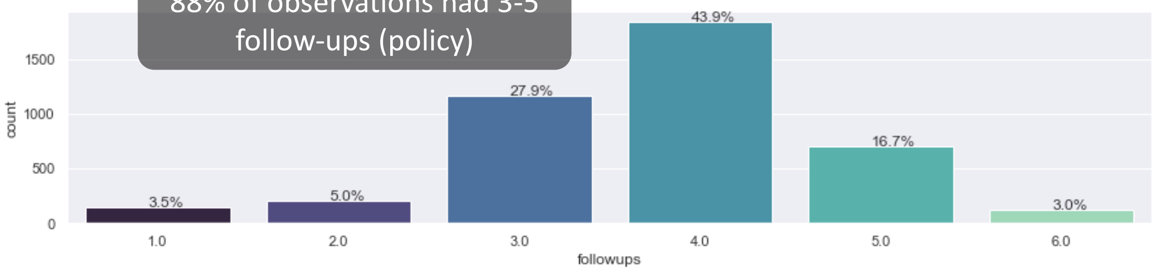
Target Variable - product conversion rate is 19% resulting from prior random marketing. This percentage needs to increase when tracked against launch of the new Wellness Travel Package



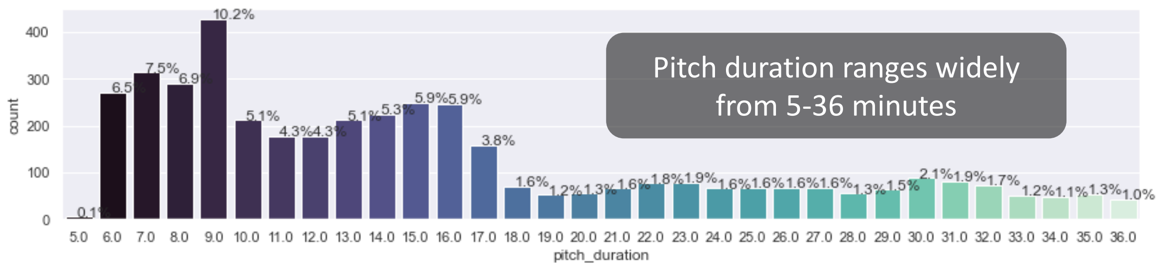
71% of contact is initiated by customer



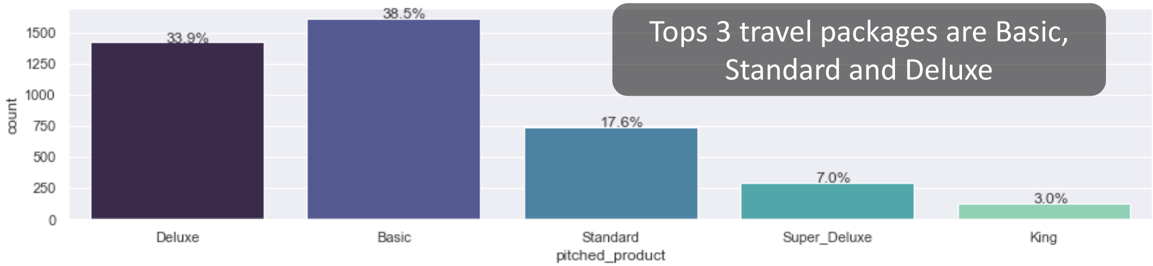
61% of observations have prop. rated less than 4



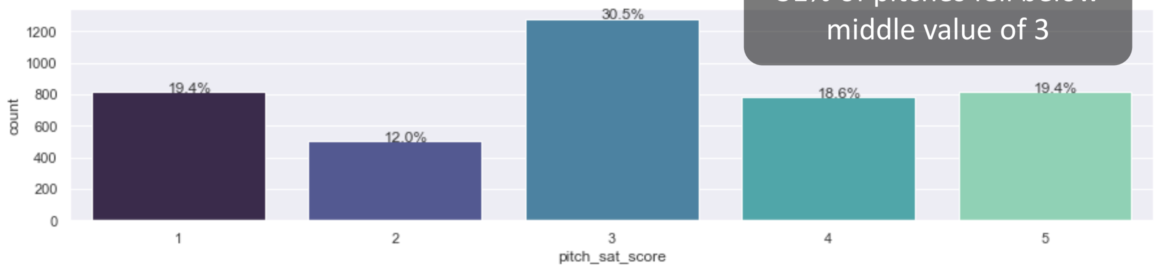
88% of observations had 3-5 follow-ups (policy)



Pitch duration ranges widely from 5-36 minutes



Tops 3 travel packages are Basic, Standard and Deluxe

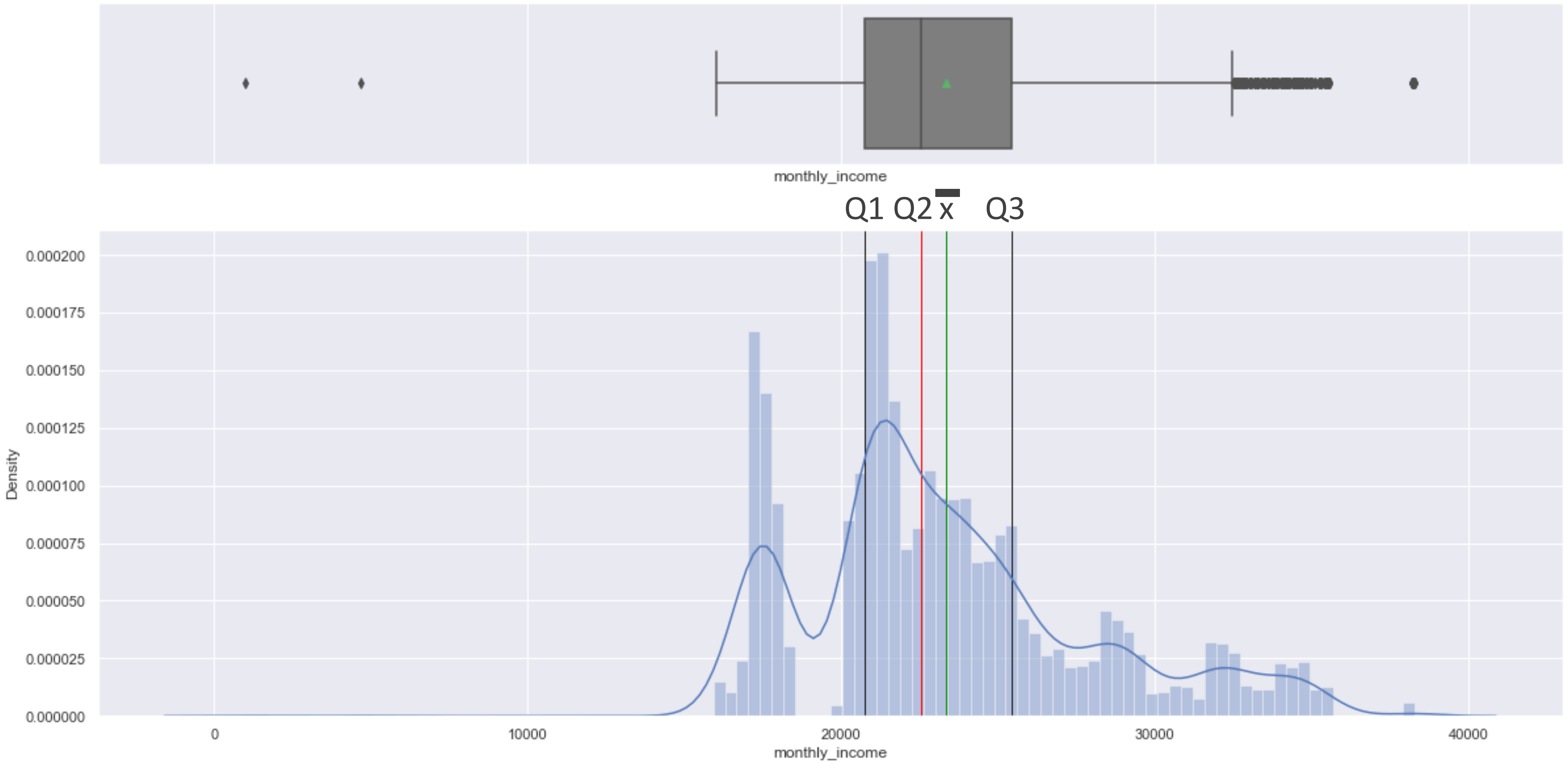


31% of pitches fell below middle value of 3

Univariate EDA – monthly_income

monthly_income is the sole continuous feature in the data set. Testing was performed by log scaling this feature. The log scaling regressed model performance and was thus reverted to thousands scale

The distribution is slightly skewed to the right and ranges from approximately from 16k to 36k

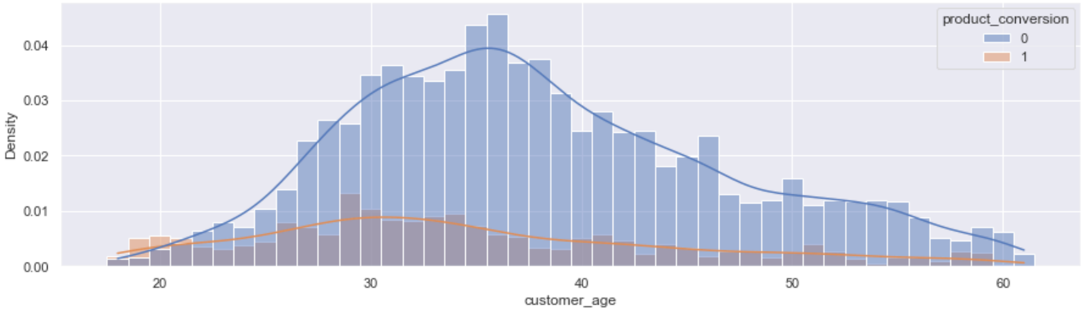


Bivariate EDA – Target Conversion Distributions

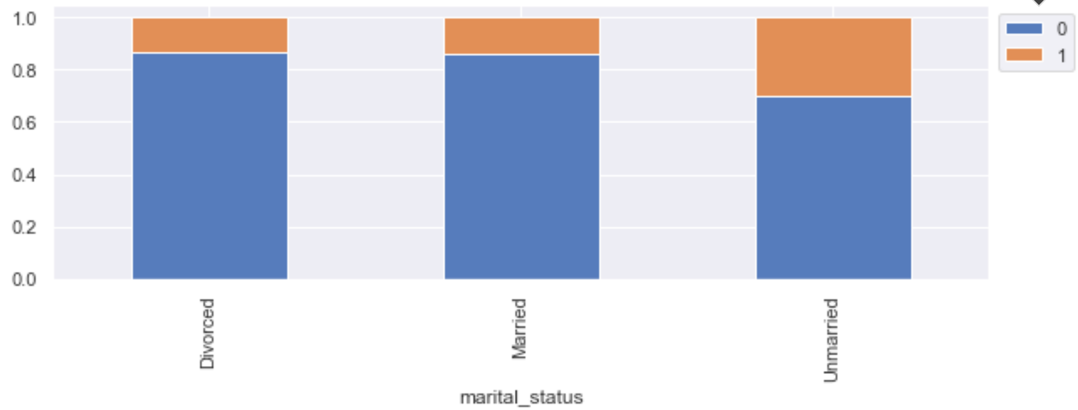
Customer Features

product_conversion
(target feature) 1=yes, 0=no

Product conversion is more common between ages 18-36

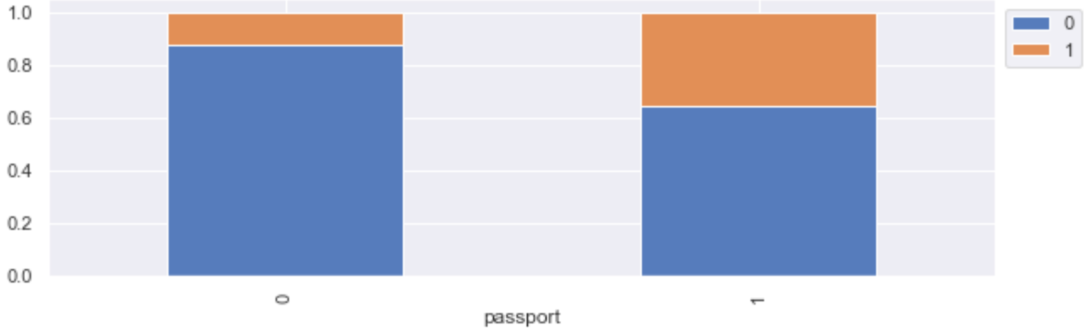


Unmarried customers are most likely to convert to buy a travel package



The general category of Executive is most likely to convert

It is more common for customers with passports to convert

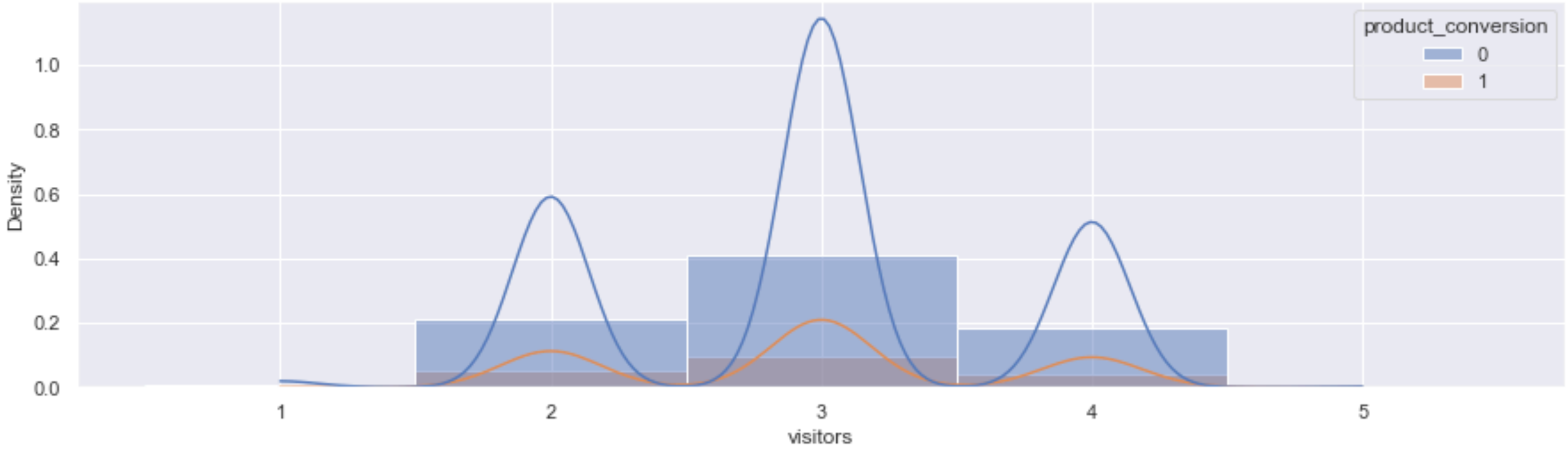


ubsequent model importance

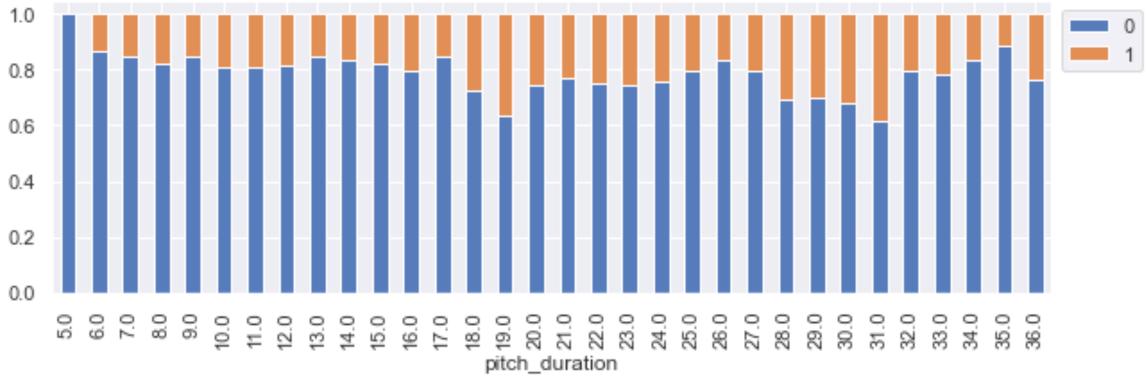
Bivariate EDA – Target Conversion Distributions

Customer Interaction Features

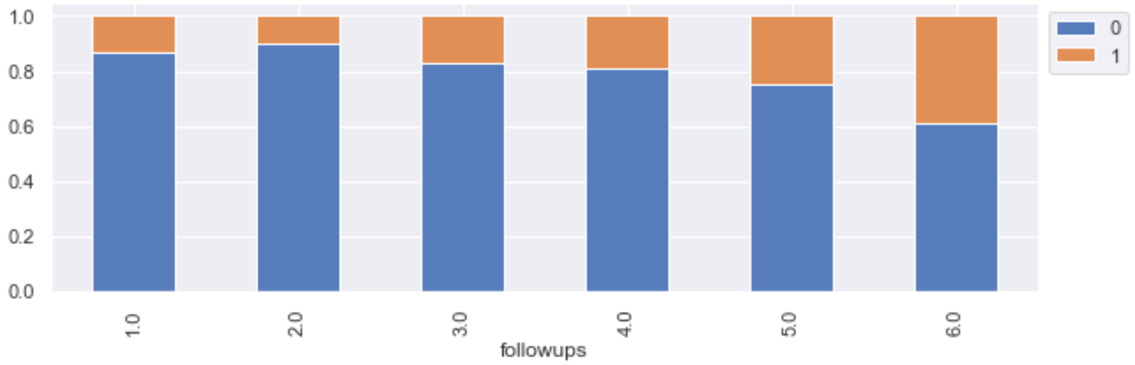
Between 2-4 visitors are most common among those that convert to buy a package



Pitch durations between 18-31 minutes observe higher frequencies of buy conversion



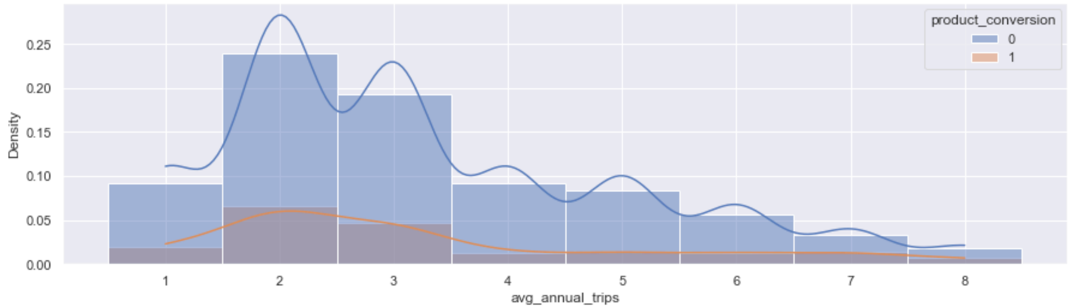
Buy conversions occur more frequently when there are 4-6 follow ups



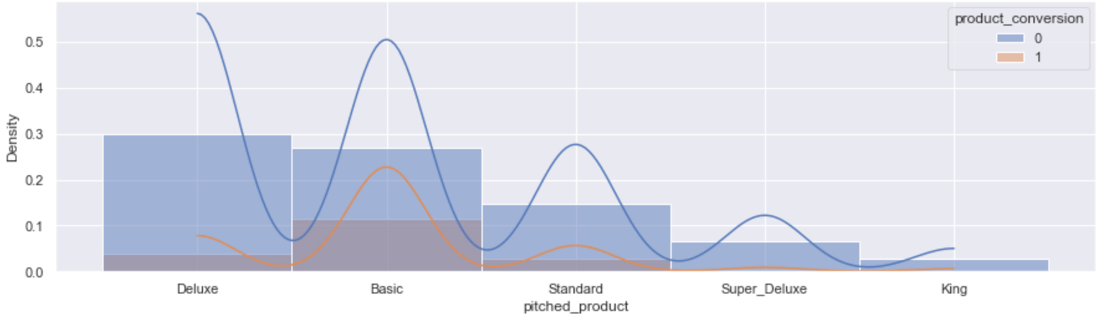
Bivariate EDA – Target Conversion Distributions

Additional Features

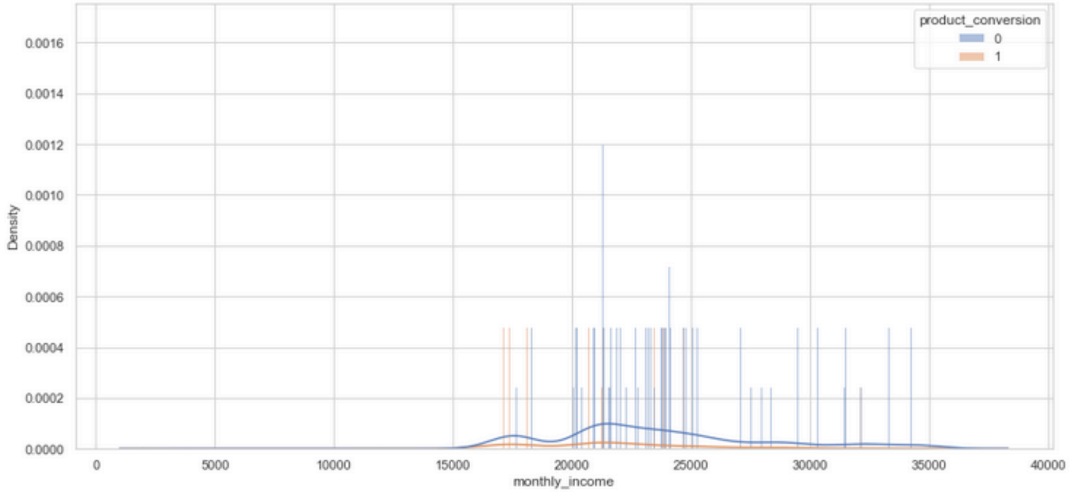
On average, between 2-3 annual trips are most common among those that convert to buy a package



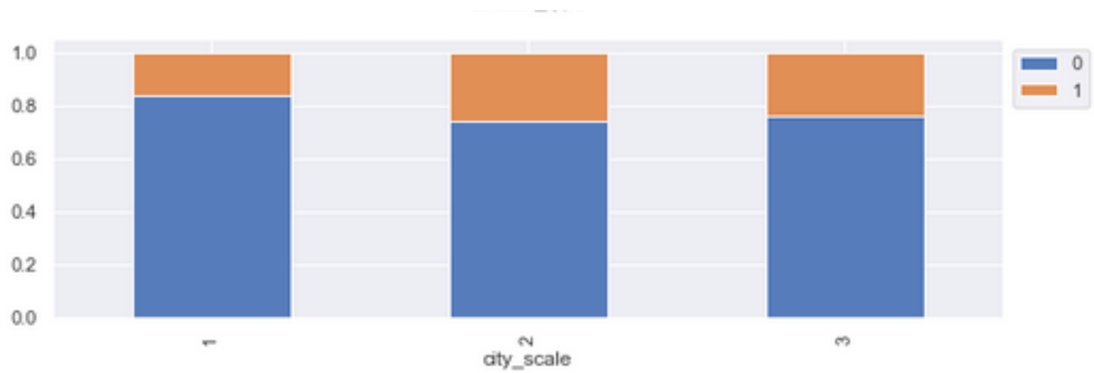
The most common travel package that converts is Basic follow by Standard and Deluxe



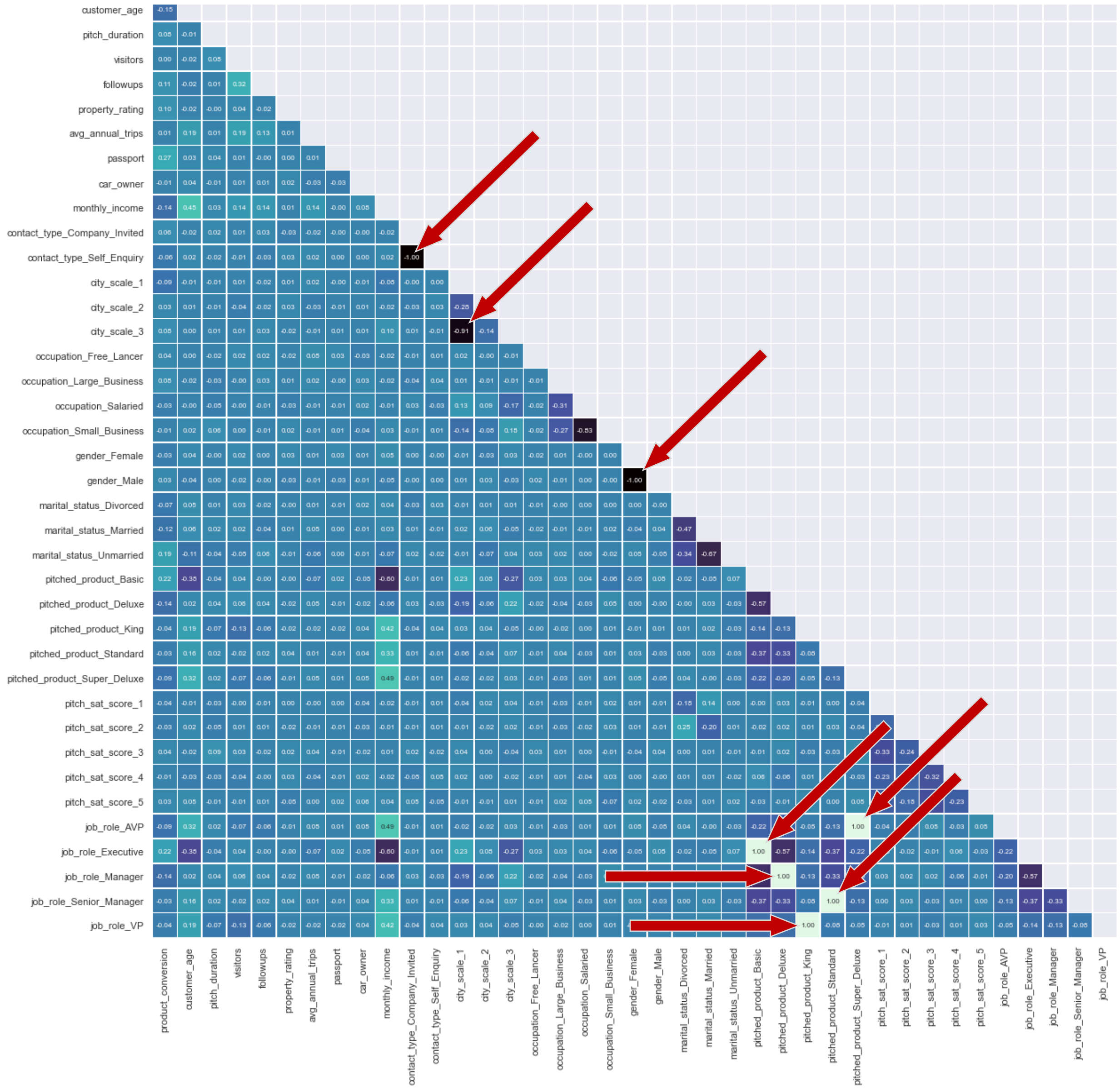
We can observe the monthly incomes 16k to 25k are more commonly associated with product_conversion.



City scale/tier 2 and 3 see more conversions



Multivariate EDA – Correlation Heatmap (w/collinearity)



Observations on

In the crosstab charts, we can see which independent feature values occur with high probably of target feature product_conversion = 1 (this gives us an intuition about which features are more important - confirm post-modeling)

In particular, the following can be observed in the data:

- city_scale 2 and 3 tend to have more product_conversions
- While Freelance occupation all have product_conversion = 1, there are only 2 observations in the data set for FreeLancer (thus this is not as impactful as it may appear visually in the crosstab chart)
- The majority of product_conversions occur with unmarried customers vs. married or divorced
- The largest proportion of product_conversion occurs with Basic package
- The largest proportion of product_conversion occurs with customer who have a passport
- The majority of product_conversion occur with customers under 40 years old
- No product_conversion can be observed with 1 or 5 visitors
- product_conversion increase from the number of follow-ups go from 3 to 4 to 5 to 6
- Most product_conversion occurs when property rating is highest (5)
- Number of average annual trips of 7 and 8 correspond the highest product_conversion

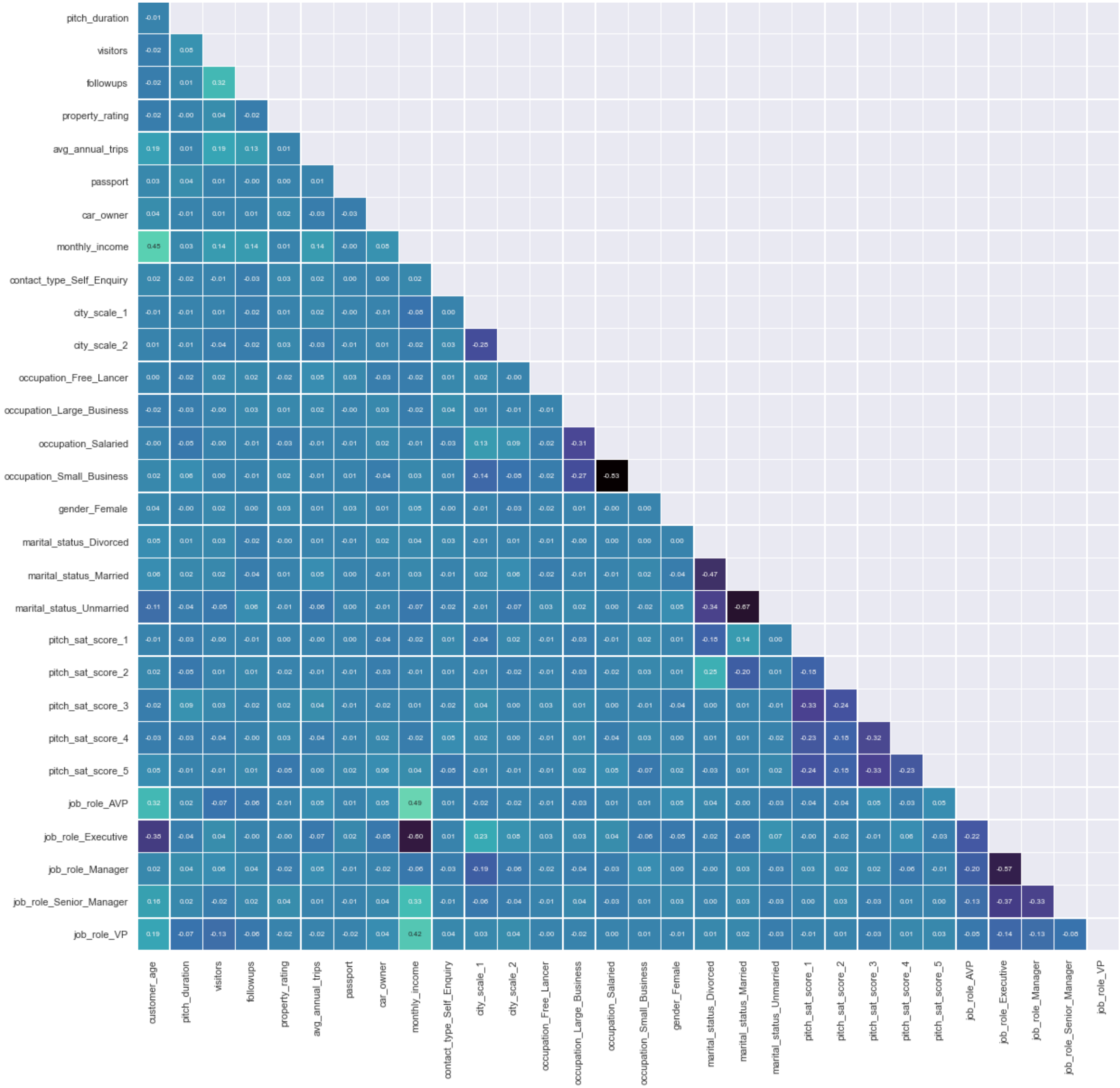
The distribution overlaps illustrate the density of product_conversion = 1 across the independent features (confirming the crosstab results)

The correlation heatmap of the intial preprocessed data show mutiple instances of collinearity:

- contact_type_Company_Invited and contact_type_Self_enquiry (-1)
- city_scale_3 and city_scale_1 (-.91)
- job_role_AVP and pitched_product_Super_Deluxe (1)
- job_role_Executive and pitched_product_PitchedProduct_Basic (1)
- job_role_Manager and pitched_product_Deluxe (1)
- job_role_Senior_Manager and pitched_product_Standard (1)
- job_role_VP and pitched_product_King (1)
- gender_Male and gender_Female (-1)

The collinear features determine each other and therefore will artificially amplify influence on the target feature in modeling (this will be removed)

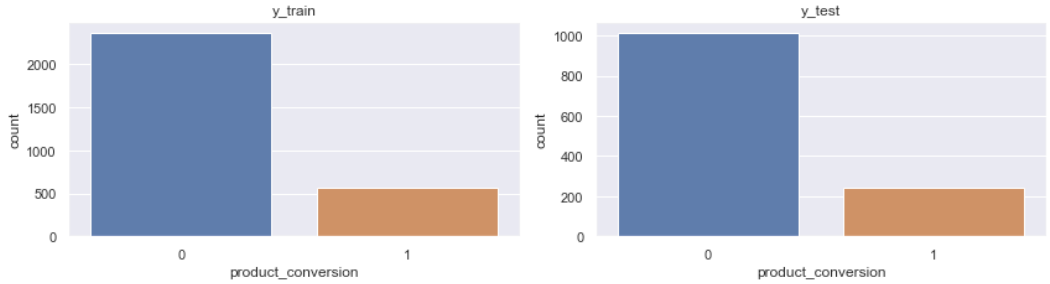
Train/Test Splitting – Correlation Heatmap (w/collinearity removed)



Observations

- Collinearity between features is removed by exclusion
- Resulting correlation heatmap confirms that collinearity has been removed from the data set
- Train/Test splitting is performed in preparation for modeling
- Use of stratify=y (target vector) ensures that ratio of product_conversion=1 remains the same between the train and test data sets
- Chart below confirms target stratification in both train and test data

Target variable count across train and test



Modeling Performance Objective

Performance Measures

- **Accuracy = TP + TN / TP + TN + FP + FN**

% of correct predictions overall

- **Recall = TP / TP + FN**

% of correct pos predictions of all predictions made (use when FN is very expensive e.g., loan default)

- **Precision = TP / TP + FP**

% of correct pos predictions of all pos data (use when FP is very expensive e.g., drone strike)

Given the business problem at hand (increasing conversion of travel packages sold), we will need to find a reasonable balance point between Precision and Recall.

If we incorrectly identify and market to customers that are not likely to buy (FP), the cost to the business is wastage of marketing budget/efforts, lost time and gradient opportunity cost over time vs. competitor strategies.

Conversely...

If we fail to identify and market to customers that are likely to buy (FN), the cost to the business is acute opportunity cost by missing out on sales which we could otherwise be getting today to strengthen business performance. This situation should be avoided as it has a greater immediate negative impact to the business performance.

Model Performance Objective - avoid FN's by maximizing Recall while trading off (limited) Precision and/or Accuracy (which we expect to decrease as a result)

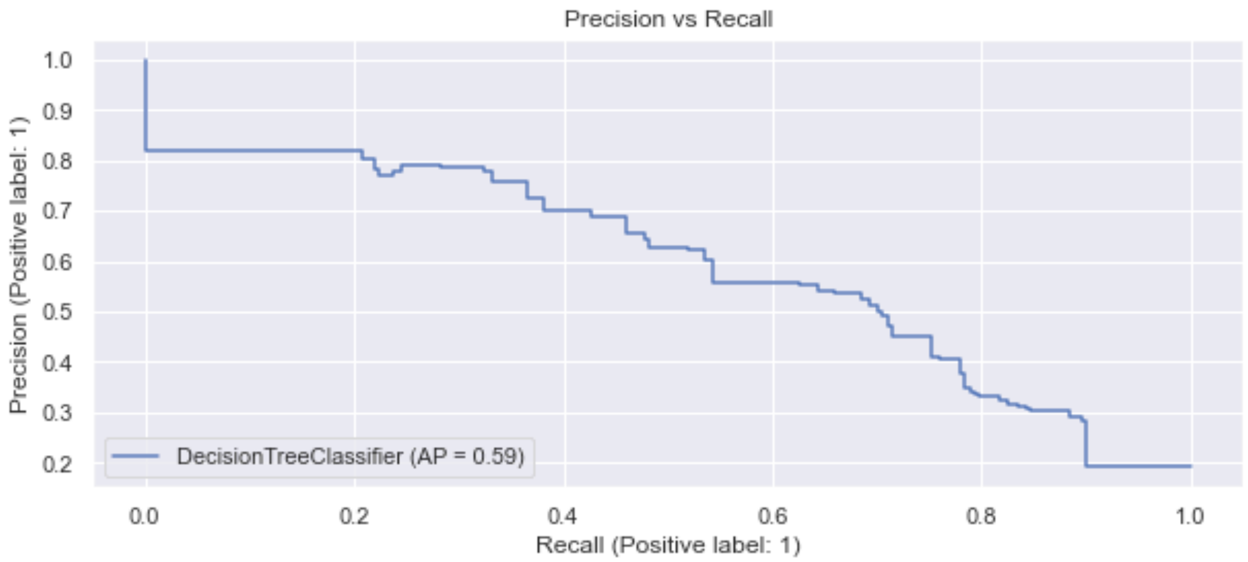
Predicted

Actual	Positive	Negative
Positive	TP	FN (type II)
Negative	FP (type I)	TN

Confusion Matrix

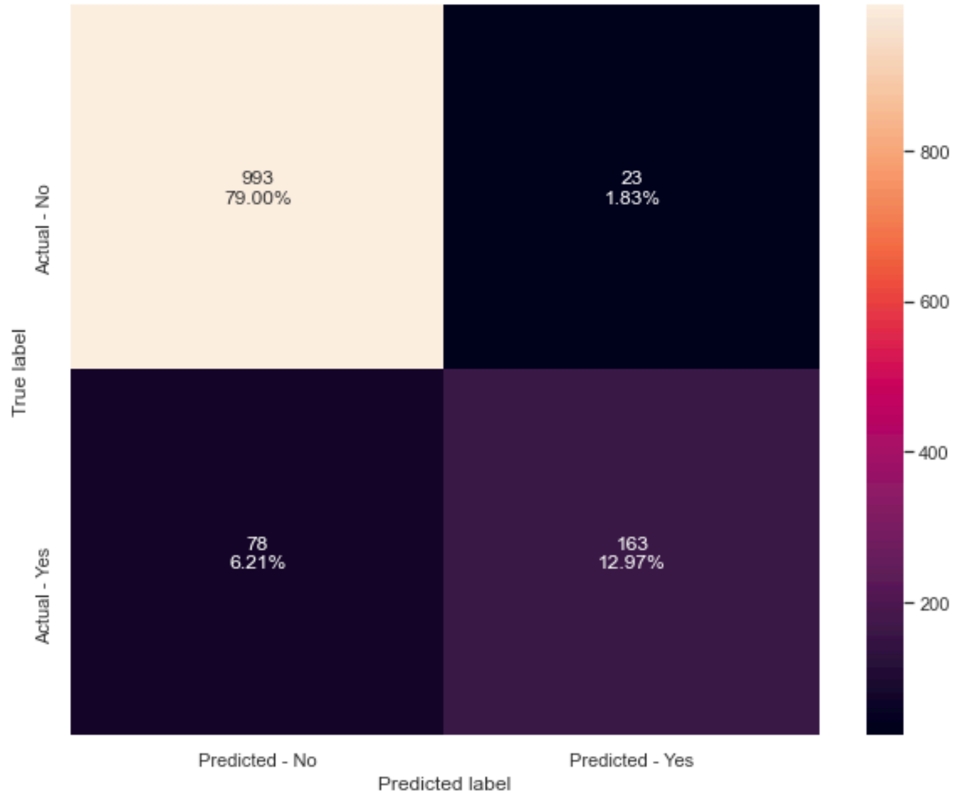
Decision Tree Classifier - Tuned

```
GridSearch/Fit time: 2.72 secs


Estimator:  DecisionTreeClassifier(class_weight={0: 0.2, 1: 0.8}, criterion='entropy',
                       max_depth=10, max_features='sqrt',
                       min_impurity_decrease=0.0001, random_state=1,
                       splitter='random')

Accuracy on training set:  0.8225861480723302
Accuracy on test set:  0.771678599840891
* Recall on training set:  0.8502673796791443
* Recall on test set:  0.7510373443983402
Precision on training set:  0.5224534501642936
Precision on test set:  0.44362745098039214
```



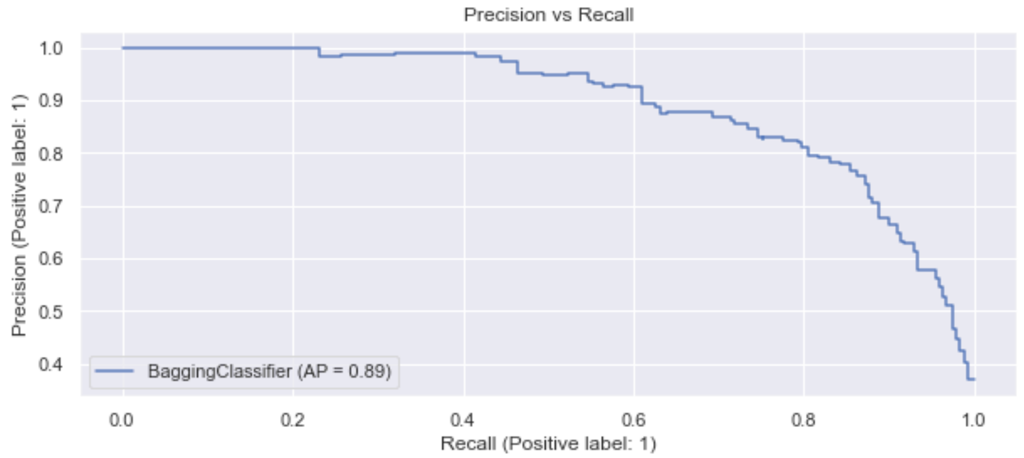
Bagging Classifier - Tuned

```
GridSearch/Fit time: 211.31 secs


Estimator:  BaggingClassifier(max_features=0.9, max_samples=0.9, n_estimators=100,
                  random_state=1)

Accuracy on training set:  1.0
Accuracy on test set:  0.9196499602227526
* Recall on training set:  1.0
* Recall on test set:  0.6763485477178424
Precision on training set:  1.0
Precision on test set:  0.8763440860215054
```



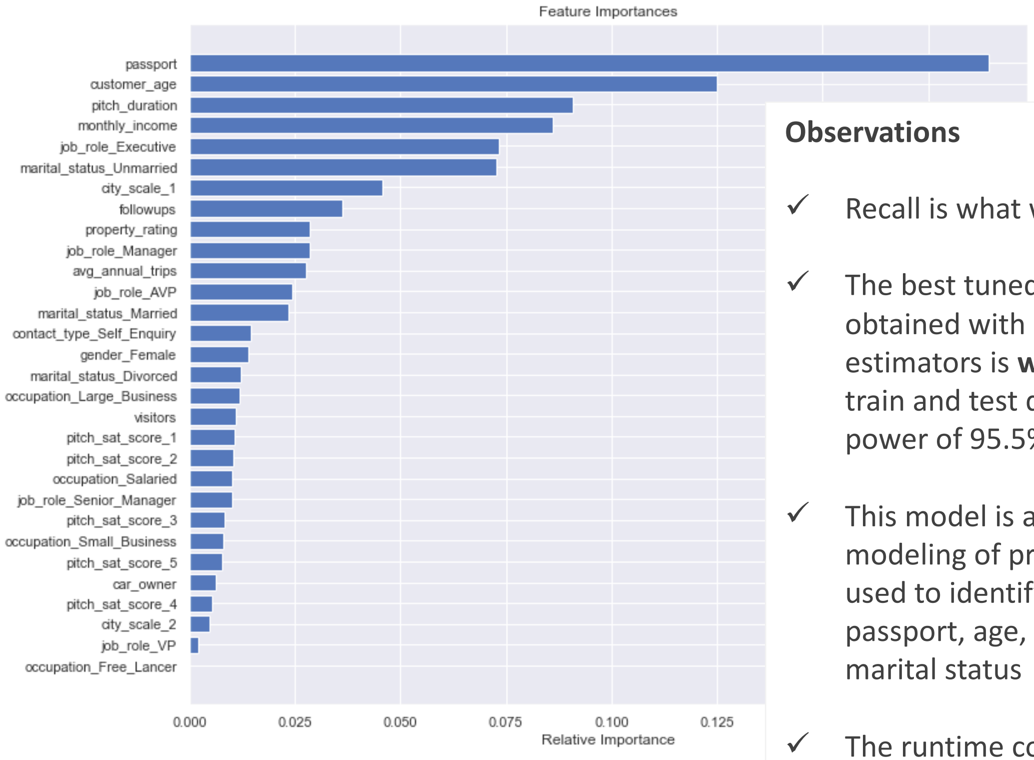
Observations

- ✓ Recall is what we want to optimize.
- ✓ The best tuned bagging classifier obtained with GridSearchCV and based on 100 estimators is **over fit** as the Recall gap between train and test data is wide with 100% prediction power within the training set.
- ✓ This model is not a good candidate for prediction modeling of product_conversion and is not likely to perform well on unseen observations.



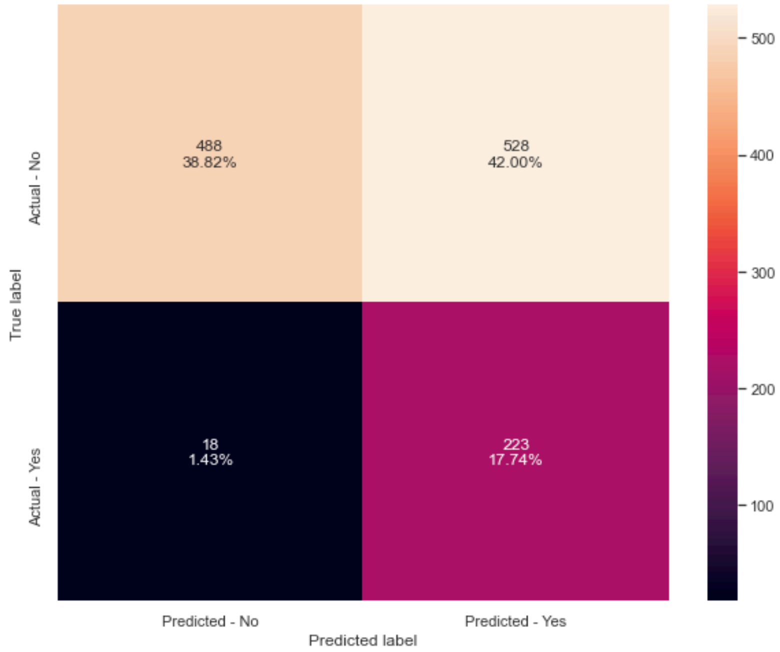
Random Forest Classifier - Tuned

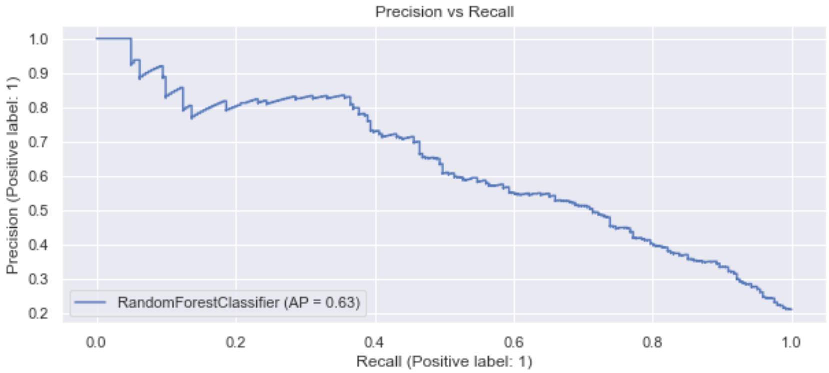
```
GridSearch/Fit time: 10253.9 secs


Estimator:  RandomForestClassifier(class_weight={0: 0.1, 1: 0.9}, max_depth=5,
                       max_features=0.2, max_samples=0.5, min_samples_leaf=6,
                       n_estimators=150, random_state=1)

Accuracy on training set:  0.601842374616172
Accuracy on test set:  0.5656324582338902
* Recall on training set:  0.9554367201426025
* Recall on test set:  0.9253112033195021
Precision on training set:  0.31942789034564956
Precision on test set:  0.2969374167776298
```



Observations

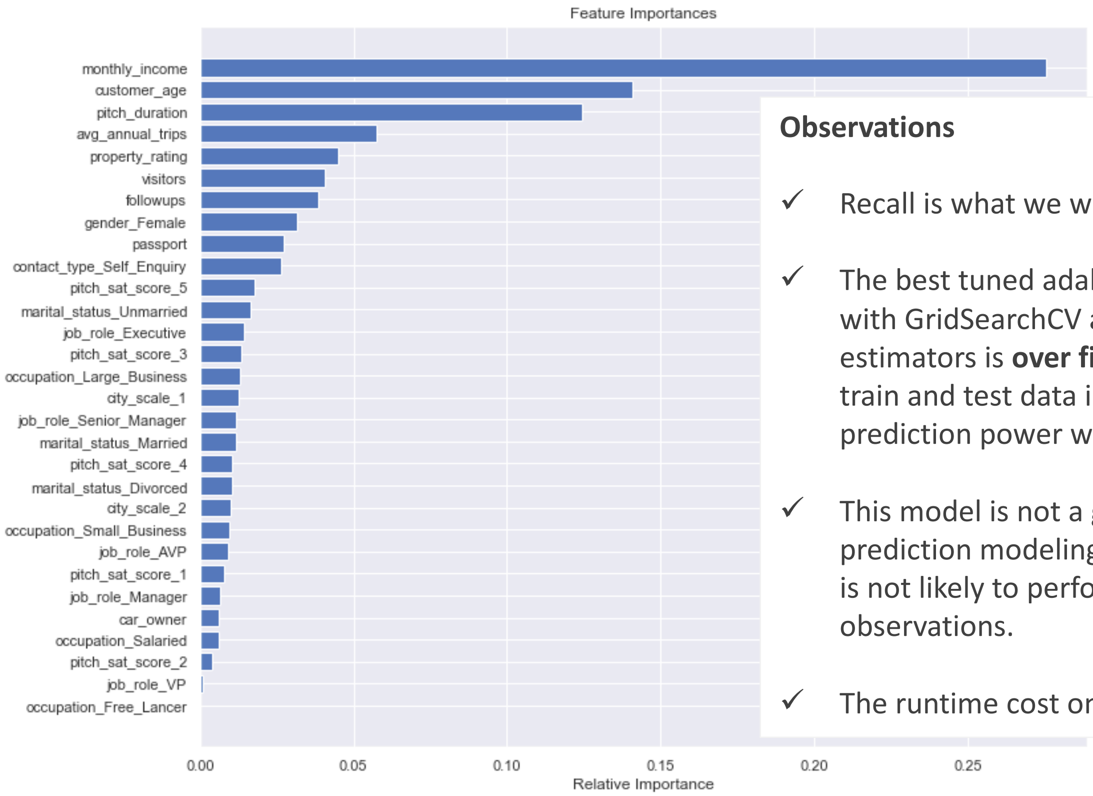
- ✓ Recall is what we want to optimize.
- ✓ The best tuned random forest classifier obtained with GridSearchCV and based on 150 estimators is **well fit** as the **Recall** gap between train and test data is small with predictive power of 95.5% on train and 92.5% on test data
- ✓ This model is a good candidate for prediction modeling of product_conversion and can be used to identify customers to market targeting passport, age, pitch_duration, income, job, marital status
- ✓ The runtime cost on a macbook pro 2.84 hours





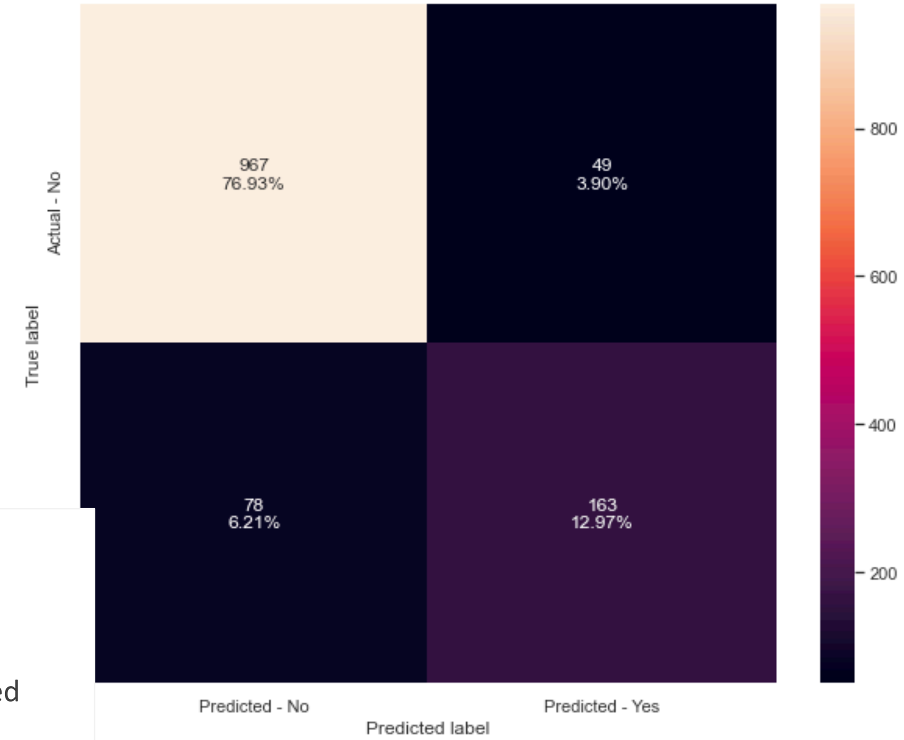
AdaBoost Classifier - Tuned

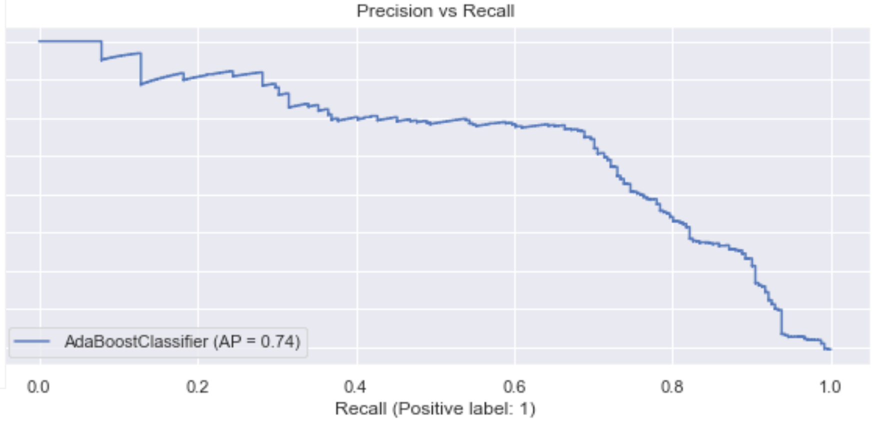
```
GridSearch/Fit time: 349.7 secs


Estimator:  AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=3),
                   learning_rate=1.2000000000000002, n_estimators=100,
                   random_state=1)

Accuracy on training set:  0.9993176390310474
Accuracy on test set:  0.8989657915672236
* Recall on training set:  0.9964349376114082
* Recall on test set:  0.6763485477178424
Precision on training set:  1.0
Precision on test set:  0.7688679245283019
```



Observations

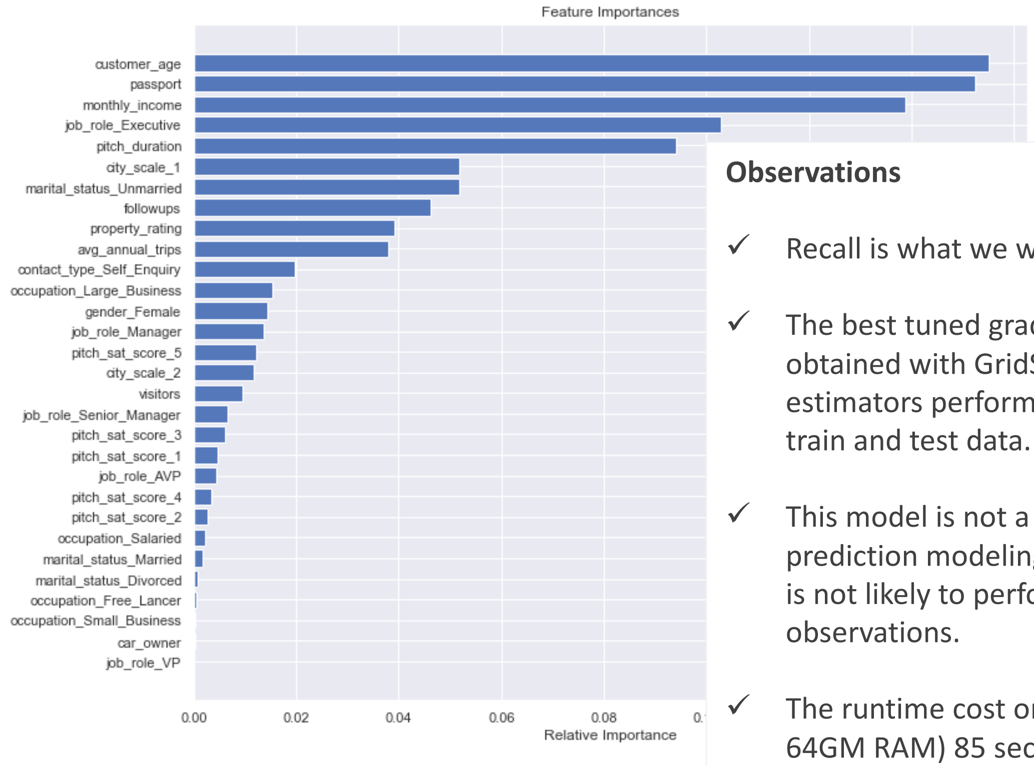
- ✓ Recall is what we want to optimize.
- ✓ The best tuned adaboost classifier obtained with GridSearchCV and based on 100 estimators is **over fit** as the Recall gap between train and test data is wide with 99.4% prediction power within the training set.
- ✓ This model is not a good candidate for prediction modeling of product_conversion and is not likely to perform well on unseen observations.
- ✓ The runtime cost on a macbook pro 5.8 mins





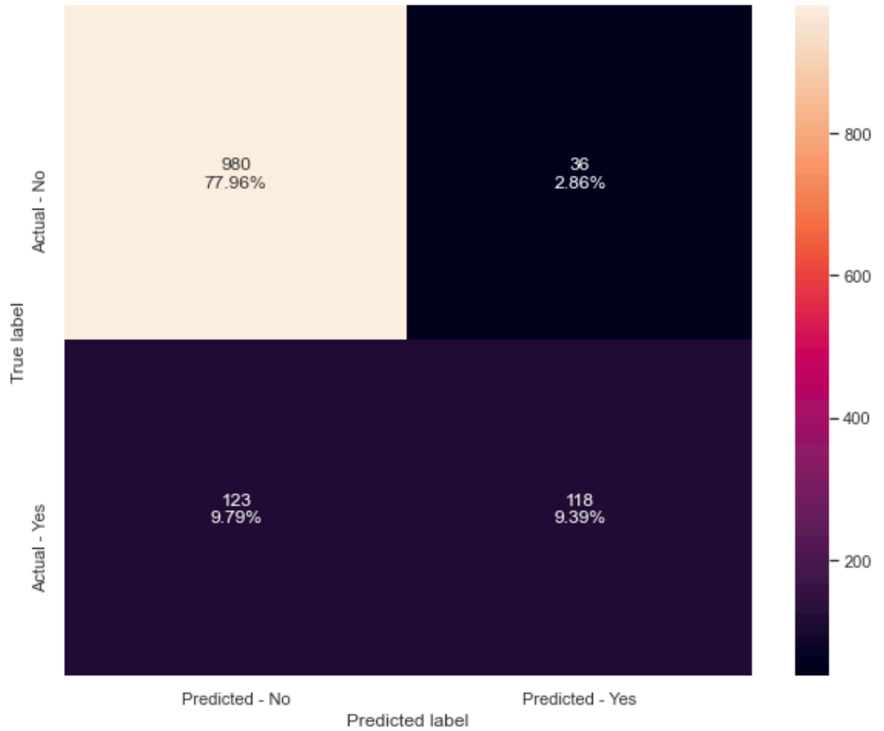
Gradient Boost Classifier - Tuned

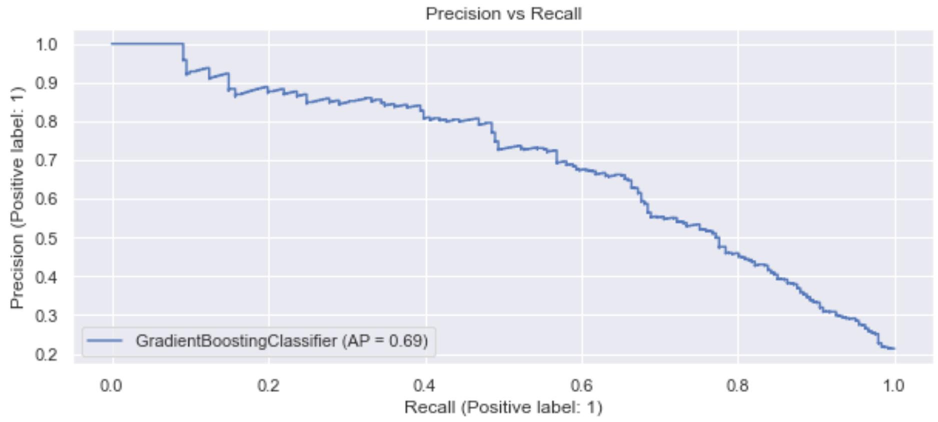
```
GridSearch/Fit time: 85.3 secs


Estimator:  GradientBoostingClassifier(init=AdaBoostClassifier(random_state=1),
                           max_features=0.8, n_estimators=150, random_state=1,
                           subsample=1)

Accuracy on training set:  0.9089048106448311
Accuracy on test set:  0.8735083532219571
* Recall on training set:  0.5793226381461676
* Recall on test set:  0.4896265560165975
Precision on training set:  0.9129213483146067
Precision on test set:  0.7662337662337663
```



Observations

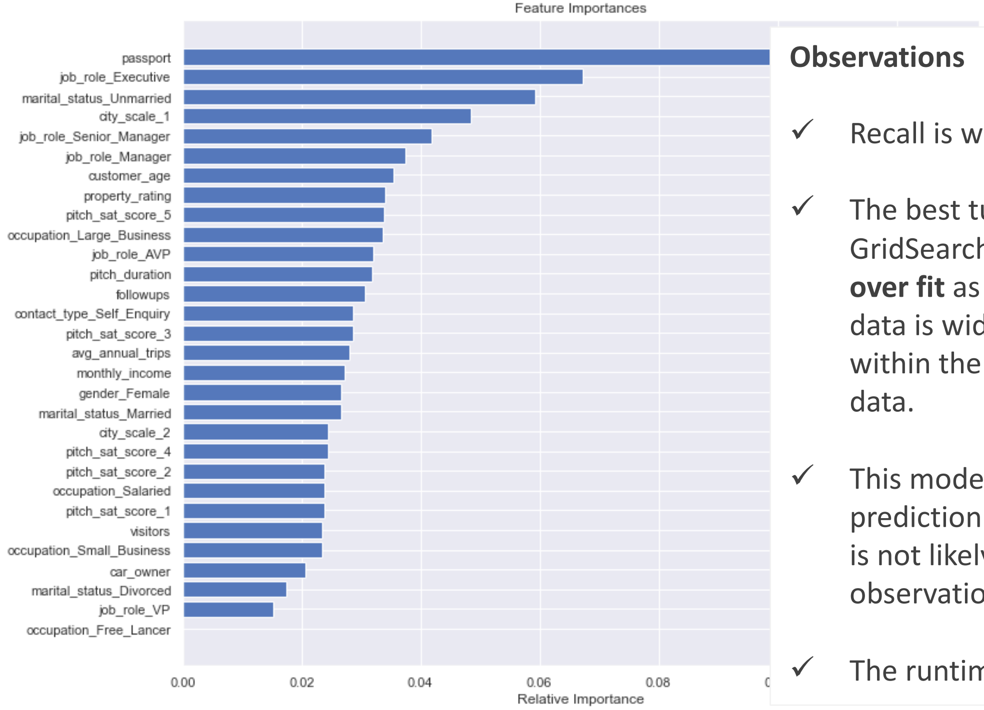
- ✓ Recall is what we want to optimize.
- ✓ The best tuned gradient boost classifier obtained with GridSearchCV and based on 150 estimators performs poorly on Recall on both train and test data. A very weak learner.
- ✓ This model is not a good candidate for prediction modeling of product_conversion and is not likely to perform well on unseen observations.
- ✓ The runtime cost on a macbook pro (8 core, 64GM RAM) 85 secs





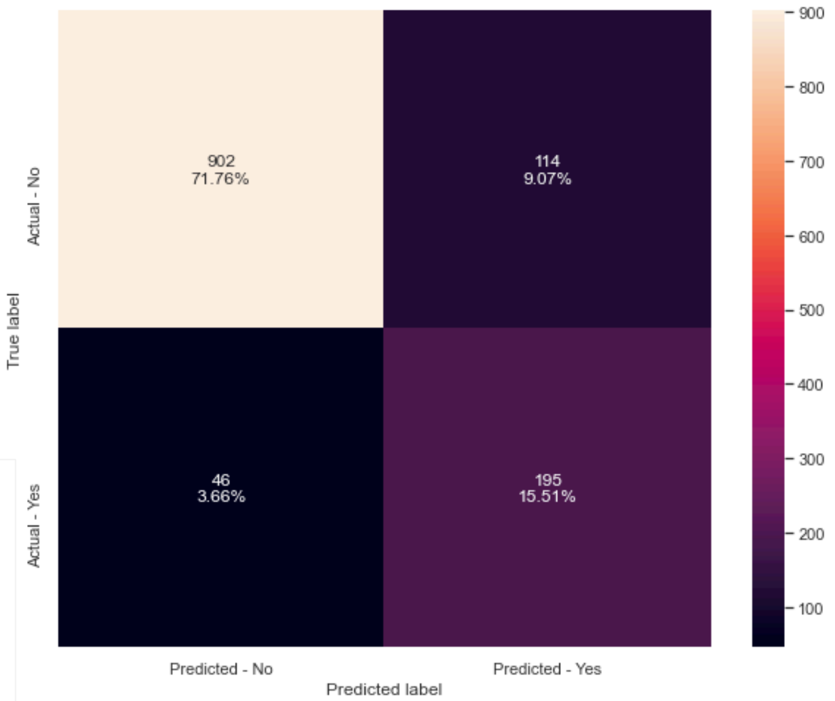
XGBoost Classifier - Tuned

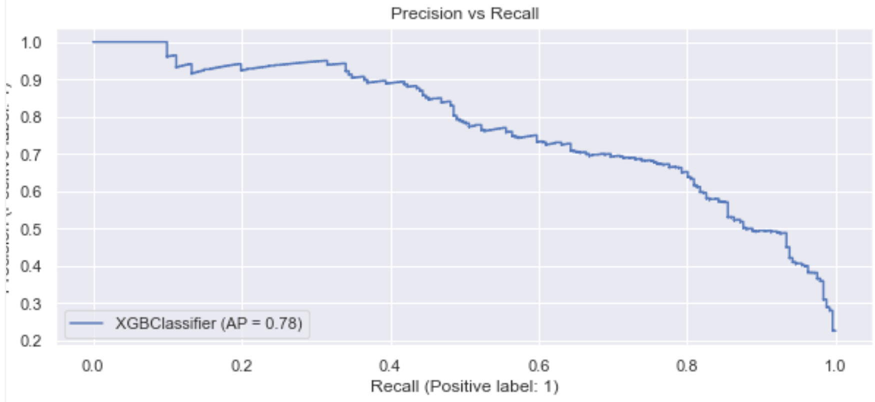
```
GridSearch/Fit time: 12014.34 secs


Estimator:  XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=0.7,
              colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
              gamma=3, gpu_id=-1, importance_type='gain',
              interaction_constraints='', learning_rate=0.05, max_delta_step=0,
              max_depth=6, min_child_weight=1, missing=nan,
              monotone_constraints='()', n_estimators=100, n_jobs=8,
              num_parallel_tree=1, random_state=1, reg_alpha=0, reg_lambda=1,
              scale_pos_weight=5, subsample=0.9, tree_method='exact',
              validate_parameters=1, verbosity=None)

Accuracy on training set:  0.9556465370180826
Accuracy on test set:  0.8727128082736675
* Recall on training set:  0.9768270944741533
* Recall on test set:  0.8091286307053942
Precision on training set:  0.8240601503759398
Precision on test set:  0.6310679611650486
```



Observations

- ✓ Recall is what we want to optimize.
- ✓ The best tuned xgboost classifier obtained with GridSearchCV and based on 100 estimators is **over fit** as the Recall gap between train and test data is wide with 97.6% prediction power within the training set and only 80.9% on test data.
- ✓ This model is not a good candidate for prediction modeling of product_conversion and is not likely to perform well on unseen observations.
- ✓ The runtime cost on a macbook pro 3.3 hours





Stacking Classifier

```
StackingClassifier(cv=10,
                   estimators=[('Random Forest - Tuned',
                                RandomForestClassifier(class_weight={0: 0.1,
                                                                     1: 0.9},
                                                       max_depth=5,
                                                       max_features=0.2,
                                                       max_samples=0.5,
                                                       min_samples_leaf=6,
                                                       n_estimators=150,
                                                       random_state=1)),
                               ('AdaBoost - Tuned',
                                AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=3),
                                                   learning_rate=1.2000000000000002,
                                                   n_estimators=100,
                                                   rando...
                                                 importance_type='gain',
                                                 interaction_constraints=None,
                                                 learning_rate=None,
                                                 max_delta_step=None,
                                                 max_depth=None,
                                                 min_child_weight=None,
                                                 missing=nan,
                                                 monotone_constraints=None,
                                                 n_estimators=100, n_jobs=None,
                                                 num_parallel_tree=None,
                                                 random_state=1, reg_alpha=None,
                                                 reg_lambda=None,
                                                 scale_pos_weight=None,
                                                 subsample=None,
                                                 tree_method=None,
                                                 validate_parameters=None,
                                                 verbosity=None))
```

Stacking Performance

R-square on training set : 0.7266454568018232

R-square on test set : 0.21968830659653027

RMSE on training set : 0.2056851479204556

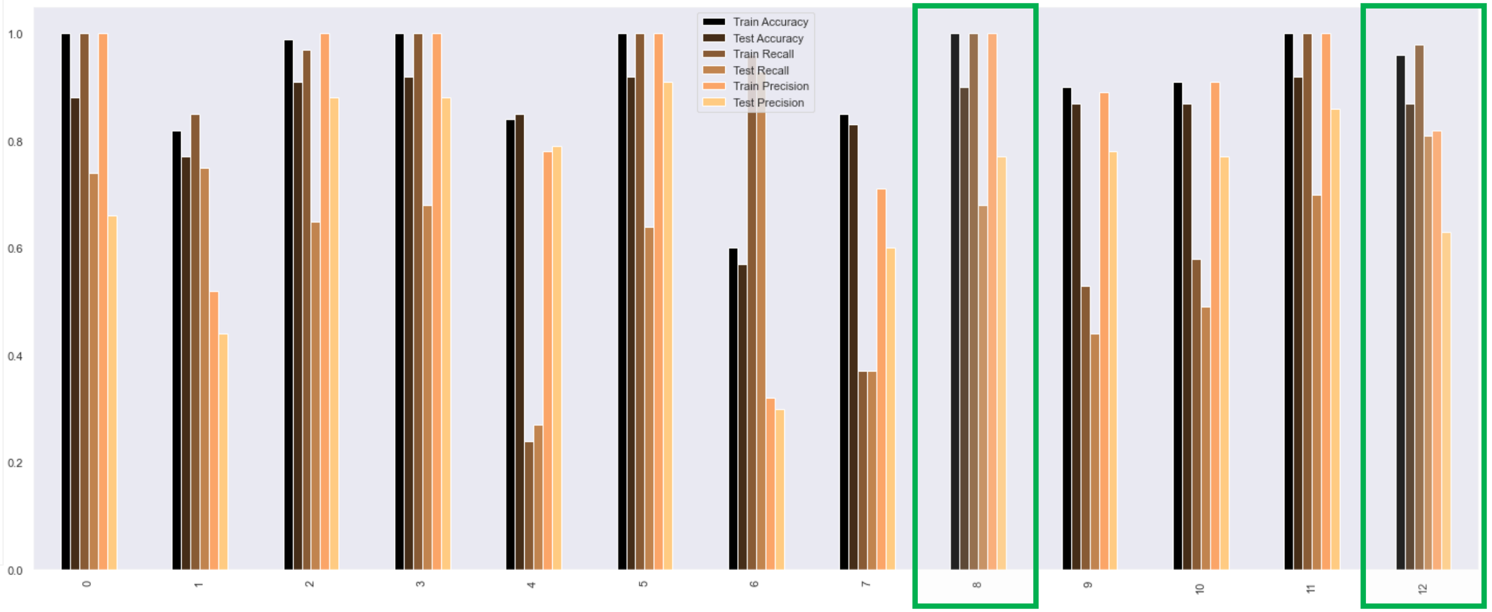
RMSE on test set : 0.34773960392801956

Aggregate Model Results & Performance

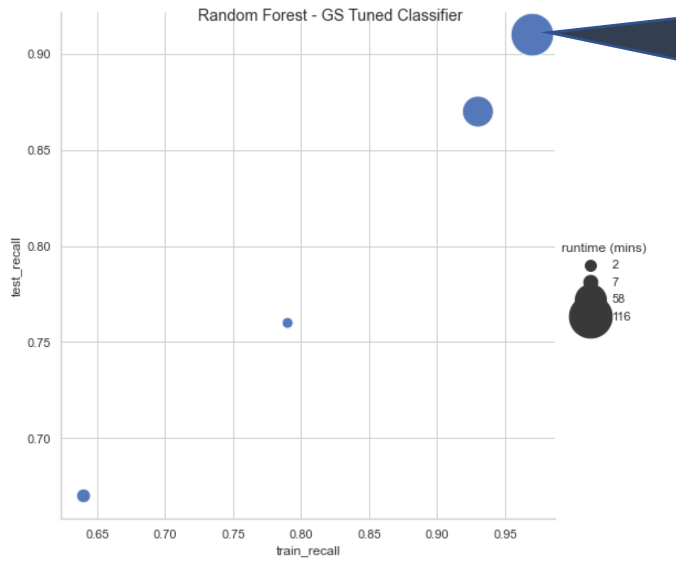
Observations

- ✓ The best model performance on for Recall is the tuned Random Forest Classifier utilizing 100 estimators with recall (96% train and 93% test). This GridSearchCV costs 2.84 hours of runtime on a macbook pro (8 core, 64GB ram)
- ✓ The best model performance balanced across both recall and precision is the tuned XGBoost Classifier utilizing 100 estimators. This GridSearchCV cost 3.33 hours of runtime on a macbook pro (8 core, 64 GB RAM)
- ✓ Tuned Decision Tree classifier does not perform well on testing data and is over fit (not a good candidate for prediction)
- ✓ Tuned Bagging classifier does not perform well on test data and is over fit to the testing data (not a good candidate for prediction)
- ✓ Bagging (Logistic Regression) classifier performs extremely poorly on recall for both traing and testing data (not a good candidate for prediction)
- ✓ Tuned Random Forest classifier performs well on recall with 96% efficacy on test data and 93% efficacy on test data - this model is well fit and is a good candidate for prediction
- ✓ Tuned Adaboost model performs substantially more weakly with regard to recall than the best random forest and XGBoost models
- ✓ Gradient Boost model performs substantially more weakly with regard to recall than the best random forestXGBoost models
- ✓ XGBoost models suffers from some over fitting observing recall on train of 98% vs. on test of 81% (we might consider using this model in parallel with the Random Foreast model in the business setting)
- ✓ Stacking does not appear to perform well given time constraints and the potential high number of combinations required to tune a meta model across heterogeneous models types (further analysis recommended)

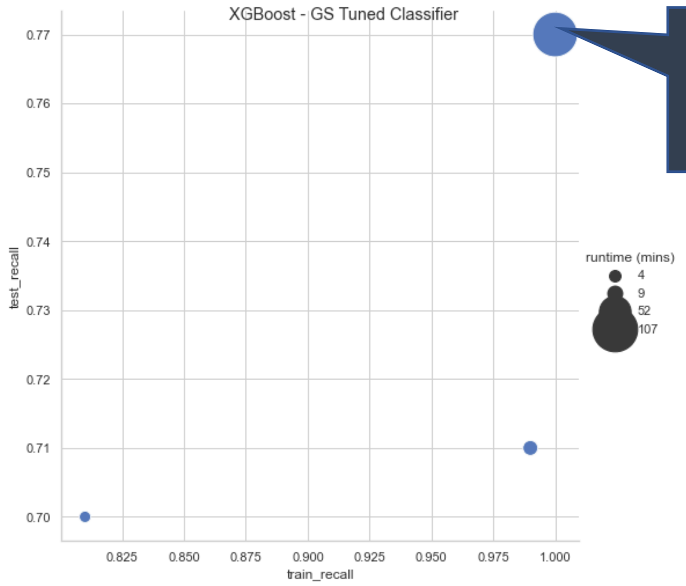
	Classifier	Train Accuracy	Test Accuracy	Train Recall	Test Recall	Train Precision	Test Precision
0	Decision Tree (default)	1.00	0.88	1.00	0.74	1.00	0.66
1	Decision Tree (GS/tuned)	0.82	0.77	0.85	0.75	0.52	0.44
2	Bagging (default)	0.99	0.91	0.97	0.65	1.00	0.88
3	Bagging (GS/tuned)	1.00	0.92	1.00	0.68	1.00	0.88
4	Bagging (base_estimator=LR)	0.84	0.85	0.24	0.27	0.78	0.79
5	Random Forest (default)	1.00	0.92	1.00	0.64	1.00	0.91
6	Random Forest (GS/tuned)	0.60	0.57	0.96	0.93	0.32	0.30
7	Ada Boost Classifier	0.85	0.83	0.37	0.37	0.71	0.60
8	Ada Boosat Classifier (tuned)	1.00	0.90	1.00	0.68	1.00	0.77
9	Gradient Boost Classifier	0.90	0.87	0.53	0.44	0.89	0.78
10	Gradient Boost Classifier (tuned)	0.91	0.87	0.58	0.49	0.91	0.77
11	XGBoost Classifier	1.00	0.92	1.00	0.70	1.00	0.86
12	XGBoost Classifier (tuned)	0.96	0.87	0.98	0.81	0.82	0.63



Model Performance



Best performing Recall model (96% train/91% test) has a larger runtime cost

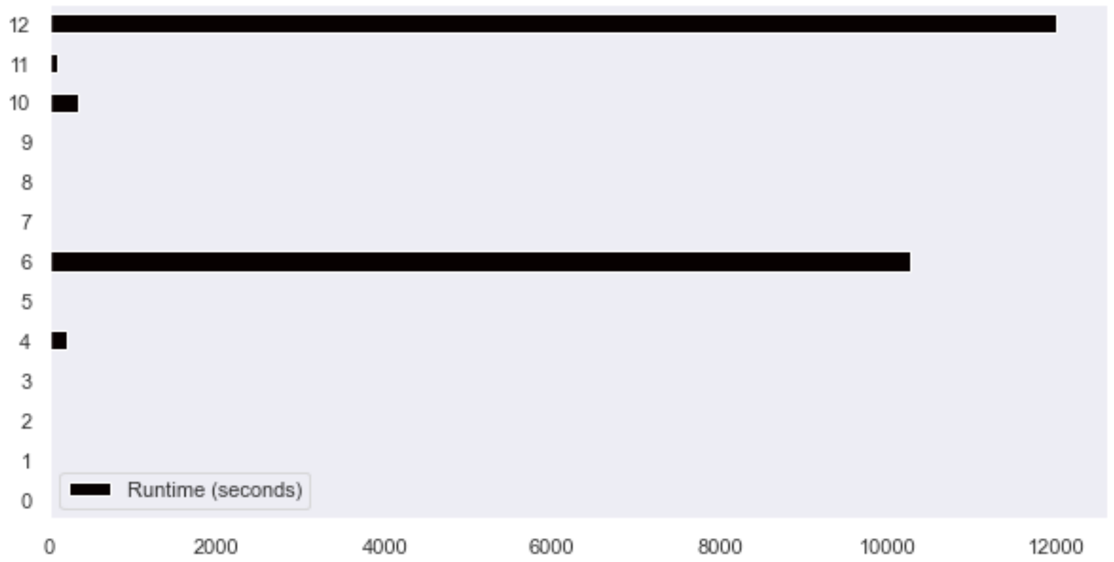


Best balanced model (recall 98% train/81% test) has a large runtime cost

Models	Runtime (seconds)
0	0.017835
1	2.722041
2	0.147031
3	0.369934
4	211.314800
5	1.309340
6	10253.901167
7	0.234173
8	0.603201
9	0.877914
10	349.699793
11	85.295764
12	12014.340559

Total tuning time: 382.01 mins

Runtime performance



Key Findings & Insights

- **Customer Profiles** From the insights, we can identify the following customer profiles (marketing personas)
  - 20-40 yo **Urban Professional** (identity)
  - **Executive Traveler** (identity)
  - **Millennial/Gen Y Explorer** (identity)
  - **Lifelong Explorer** (identity)

- **Data Processing** The raw data was thoroughly cleansed prior to modeling to standardize features and make them easier to work with in the modeling step

- **Extreme outliers** were removed from the features to normal expected ranges

- Monthly income was log scaled to bring this feature in the same order of magnitude with the other independent features

- Observations with **nan/na values were dropped** from the data set to allow more time for modeling vs. testing different imputation strategies based on assumptions

- **Prediction Modeling**

✓ The best model performance on for Recall is the tuned Random Forest Classifier utilizing 100 estimators with recall (96% train and 93% test). This GridSearchCV runtime cost 3.33 is hours on a macbook pro (8 core, 64GB RAM)

✓ The best model performance balanced across both recall and precision is the tuned XGBoost Classifier utilizing 100 estimators. This GridSearchCV runtime cost is 3.33 hours on a macbook pro (8 core, 64GB RAM)

- **Univariate EDA** Feature value diversity and proportions were visual analyzed making note of their frequencies
  - Values which dominate across the raw data set:
    - Small business, salaried customers
    - City scale 1 and 3 account for 96% of customers
    - Majority of customers are male (59%)
    - Deluxe and Basic packages are most popular comprising 73% of packages sold
    - Majority of customer own a car (61%)
    - Majority of customer are under 40 years old
    - Majority of pitches last 20 minutes or less
    - 80% of customer report an average annual trips of 5 or less
    - The most likely number of visitors among customers is 3
    - The most likely number of pitch follow-ups is 4
- As the only non-discrete continuous feature, Monthly Income (log) distribution is plotted to understand central tendency and dispersion of vector to be modeled
- **Multivariate EDA** Distribution of target feature (product_conversion = 1) across independent feature values:
  - Values which most commonly associated with positive target value (1):
    - Majority of product_conversion (=1) occur in city scale 2 and 3
    - Majority of products sold to unmarried/single customers vs. married or divorced customers
    - Most customers bought the Basic package
    - Most buying customers are less than 40 years old and have a passport
    - Product conversion occurs only for customers how had between 3-5 visitors
    - Starting with 3 follow-ups, buy conversion probability increases with each follow-up (3, 4, 5, 6)
    - Buy conversion likely is higher when property rating is highest (5)
  - Collinearities is identified in the following features:
    - contact_type_Company_Invited and contact_type_Self_enquiry (-1)
    - city_scale_3 and city_scale_1 (-.91)
    - job_role_AVP and pitched_product_Super_Deluxe (1)
    - job_role_Executive and pitched_product_PitchedProduct_Basic (1)
    - job_role_Manager and pitched_product_Deluxe (1)
    - job_role_Senior_Manager and pitched_product_Standard (1)
    - job_role_VP and pitched_product_King (1)
  - Collinearities were removed prior to modeling

Recommendations to Business

Acquire properties with access to wellness landscapes and natural beauty e.g., mountains, lakes, oceans, healing places

Focus marketing branding campaigns on targeting customer profiles

- Urban Professional
- Millennial/Gen Y Traveler
- Lifelong Traveler

Devise and market Basic Wellness Package for Urban Professionals, Millennial/Gen Yers and Lifelong Travelers

Focus marketing campaigns on top importance features identified by the tuned Random Forest classifier

- Passport (those with)
- Age (between 20-40 years of age)
- Monthly income (between 16k and 25k)
- Executive job role (yes)
- Unmarried (yes)

Focus on marketing campaigns on top important features identified by the tune XGBoost classifier

- City Scale (Tier 2 and 3)
- Senior Manager (yes)
- Manager (yes)

Focus marketing efforts on those who expect to take 2-3 trips annually on average

Update policies (sales guidance) to keep wellness package pitches between 15 and 26 minutes

Update policies (sales guidance) for target at least 3 follow ups after pitches have been delivered for the wellness travel package, as conversion frequency increases after 3 follow ups