



— VISIT WITH US —

WELLNESS TOURISM

Travel Package Purchase Prediction Model

TABLE OF CONTENTS



01

Introduction

Introduces the scope of the study, major assumptions and prominent limitations

02

Key Findings

Outlines the major findings of the study and highlights key recommendations

03

Consumer Data

Describes the key variables examined from a sample of Visit With Us's customers

04

Methodology

Outlines how the logistic regression model was constructed and how it can be applied as a business tool

05

Recommendations

Demonstrates how to apply the key findings of the study to improve business strategy

06

Conclusion

Model evaluation overview and summary of key findings

01



Introduction

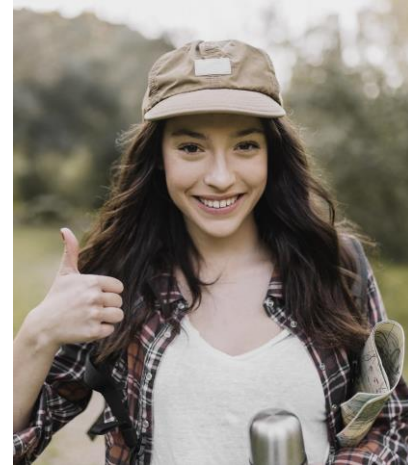
Brief overview and scope of
the study



Our Company

Visit With Us is a tourism company looking to expand its client base. Currently, there are 5 types of packages the company is offering – Basic, Standard, Deluxe, Super Deluxe and King. Historical data suggests that 18% of customers who were pitched these products in last year's marketing campaign purchased a package.

Visit With Us is planning to launch a new product: Wellness Tourism Package. Wellness Tourism is travel that allows the traveler to maintain, enhance or kick-start a healthy lifestyle, and support or increase one's sense of well-being.





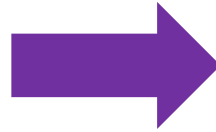
The Opportunity

- Wellness Tourism Package to increase appeal:



Current Customers

Visit With Us currently has a customer database of nearly 5000 individuals!



Opportunities

Harness the power of the customer database to target customers who would be interested in purchasing new Wellness Packages



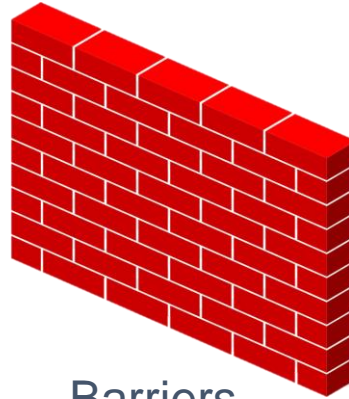
Higher Profits

Wellness Tourism is a diversified product that can increase sales by increasing appeal to customers looking for travel alternatives



The Problem

Visit With Us struggles to with high marketing costs:



Current Customers

Visit With Us has an 18% conversion to sales rate on customer calls

Barriers

Barriers include inability to directly target customer segments; high marketing costs for blanket campaigns

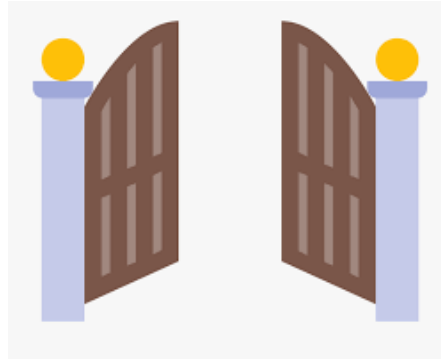
Wellness Customers

Wellness Tourism could increase conversion rate or even capture multiple sales from a single customer



The Solution

- Segement-based target marketing to target customers likely to make purchase



Current Customers

Current customers likely to purchase a product targeted for Wellness Tourism Package

Opportunities

Solutions include cross-marketing original and Wellness products to segments most likely to purchase travel products

Wellness Customer

Visit With Us can increase sales while substantially lowering marketing costs by creating a target marketing campaign

Major Assumptions



Assumption 1

Wellness Tourism Packages will be marketed with the same options as the regular package. They will be available in basic, standard, deluxe, super deluxe and king. This will ensure that if a customer ONLY desires a king package, they would still be interested in the Wellness Tourism Packages because these options are available.



Assumption 2

Cost of marketing is high but so is the risk of incorrectly identifying a customer as unlikely to purchase a package and, as a result, not contacting him/her. In other words, both False Positives and False Negatives are costly to Visit With Us and both should be reduced using F1 scores.

Limitations of the model



Limitation 1

The user of the model will need to know most, if not all, of the relevant customer information used to build the model. XG Boosting can impute some missing values, but too many missing values may cause errors. Also, the precision of this model when imputing errors is unknown and can increase the likelihood of False Negatives and/or False Positives. As such, every effort should be made to find out all relevant information about a customer before attempting to run the prediction.

Limitation 2

The equation is only as good as its user and the underlying dataset. While the data has been thoroughly cleaned, small human error input mistakes would not be flagged during the cleaning process and could cause unintended branching of the decision tree, lowering the reliability of the model. Similarly, users of the model who input an incorrect value for one of the variables risks receiving an incorrect decision, due to the input error.

Limitation 3

The model will not work well if the original data is altered or changed. For example, if additional information is added to the dataset, the model would have to be reconstructed in order to compensate for these changes.



02



Key Findings

Overview of final model and highlights
of key takeaways from the research



87%

This is our test data accuracy rate

67%

This is our test data F1 score

Our Best Model

Bagging, Boosting or Stacking?

We tried three distinct ensemble techniques on the data to find the model that has the best prediction scores for the problem Visit With Us tasked us with solving. In total, 13 different models with various tuning were built, including Decision Trees, Random Forests, Bagging Classifiers, AdaBoost, Gradient Boost, XG Boost, and Stacking Classifiers. Out of the bagging models, we found the Random Forest with Class Weights to be the best. Out of the Boosting methods, we found XG Boost with hyperparametric tuning to be superior. The two best models of each group were then compared with the Stacking Classifier model. We found that the XG Boost and the Stacking Classifier outperformed the Random Forest model across almost all test data scoring measures. The XG Boosting and Stacking Classifier models were nearly identical in accuracy and f1 scores, however, XG Boosting had better recall scores while the Stacking Classifier had better precision scores.

	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test F1-Score
1	Tuned XGBoost Classifier	0.92	0.87	0.83	0.69	0.77	0.65	0.80	0.67
2	Stacking Classifier	0.92	0.86	0.91	0.76	0.72	0.60	0.80	0.67
0	Random Forest with class_weights	0.90	0.86	0.82	0.66	0.69	0.63	0.75	0.65

Our Best Model

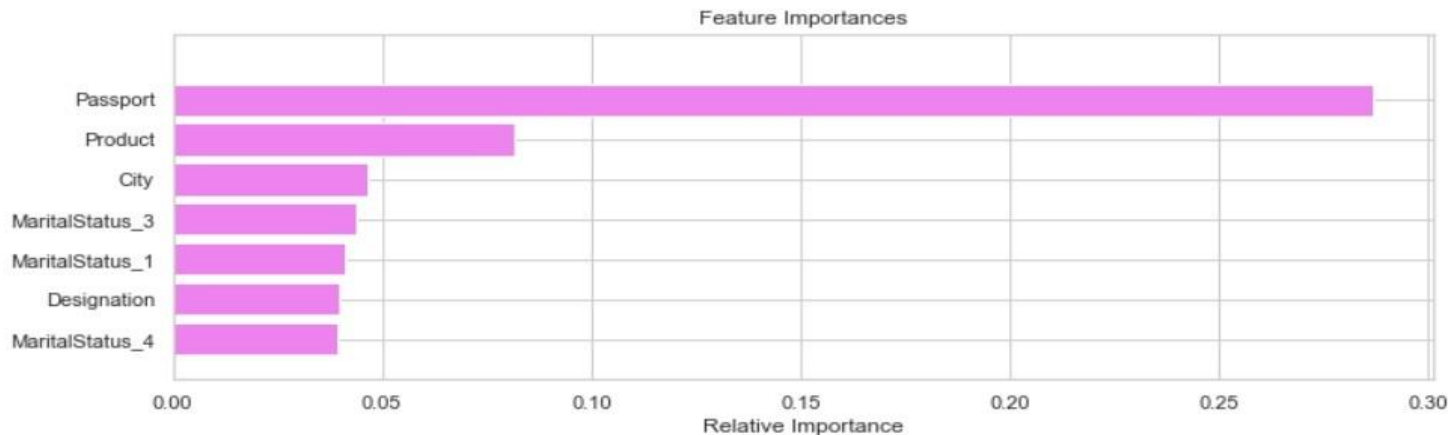
So which is it? XG Boost or Stacking Classifier?

The goal of this study was to maximize the F1 scores. Visit With Us identified that they were incurring high marketing costs, which could be lowered by identifying customers who will probably not purchase a product and not marketing directly to them. In this instance, it is important to reduce False Positives, people identified as saying yes, but ultimately say no. We could achieve this by choosing the model with the best Precision scores (Stacking Classifier). However, we also know that there are huge opportunity costs incurred when a customer is predicted to say no, and not targeted, but would have said yes. In this instance, it is important to reduce the False Negatives. We could achieve this by choosing the model with the best Recall scores (XG Boost).

Therefore, the best model is the model that addresses the more pressing concern of Visit With Us. Management should decide whether reducing False Negatives or False Positives is more important and choose the appropriate model.

	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
1	Tuned XGBoost Classifier	0.92	0.87	0.83	0.69	0.77	0.65	0.80	0.67
2	Stacking Classifier	0.92	0.86	0.91	0.76	0.72	0.60	0.80	0.67
0	Random Forest with class_weights	0.90	0.86	0.82	0.66	0.69	0.63	0.75	0.65

The Five Key Variables



- Passport, Product and City were the top three features after hyperparametric tuning on XG Boost
- Passport received a relative importance score of nearly 0.30, meaning, by itself, passport explains almost 30% of the customers who purchased a travel product
- Product, in second, had a relative importance score of almost 0.08
- The city development variable was third at just under 0.05
- The three MaritalStatus features shown can be viewed as a group, since the model uses OneHotEncoding
- Finally, designation scored the fifth highest of all the features, around 0.04

Top 3 Important Features

PASSPORT

Passport is a binary variable that answers the question, “Does the customer has a passport or not?” (0: No, 1: Yes) According to the final model, passport is the number 1 predictor to identify customers who are likely to purchase a product from Visit With Us.



PRODUCT

The ProductPitched variable was considered the second most important feature for the model. There were 5 types of products pitched by the salesperson: basic, standard, deluxe, super deluxe and king. These products were ranked, with basic being the lowest and king being the highest.



CITY

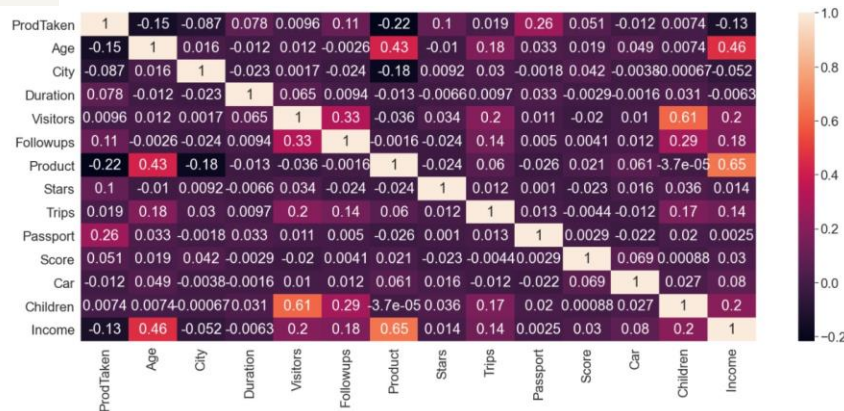
The CityTier variable was ranked third most important feature by the final model. City tier depends on the development of a city, population, facilities, and living standards. The categories are ordered i.e. Tier 1 > Tier 2 > Tier 3.



Correlations

Relationships between features

The independent variables in this study have varying degrees of correlation with one another. -1.0 denotes perfect negative correlation, 1.0 is perfect positive correlation. For this study, we define -0.09 to 0.09 as no correlation, 0.10 to 0.24 as a weak, positive correlation, 0.25 to 0.59 as a moderate, positive correlation, and 0.60 to 0.99 as a strong, positive correlation. Similarly, on the negative side, -0.09 to -0.24, -0.25 to -0.59 and -0.60 to -0.99 represent weak, moderate and strong negative relationships, respectively. See following slides for addition info...



- Strong positive correlations between product & income (0.65) and between visitors and children (0.61)
- Moderate positive correlations between age & income (0.46), age & product (0.43), visitors & follow ups (0.33), follow ups & children (0.29) and product taken and passport (0.26)
- There were no significant negative correlations (moderate nor strong)

Correlations

DEPENDENT VARIABLE

In our model, Product Taken is the dependent model. Visit With Us is interested in increasing the number of customers who purchase travel packages.

The Product Taken variable has moderate, positive correlation with Passport (0.26). We saw that the passport feature is the best predictor of whether the customer will purchase a travel product.

The two alarming statistics we see is that Product Taken has a negative correlation with both Product Pitched and Income. While the correlations are weak, the negative directionality is concerning because it means Visit With Us is not taking advantage of selling to wealthier customers and as their product increases in cost, fewer people are purchasing. This means there is either a flaw with the higher end products or with the marketing strategy employed by Visit With Us. See Recommendations section for potential remedies to this problem.

	ProdTaken
ProdTaken	1.000000
Age	-0.147254
City	-0.086852
Duration	0.078257
Visitors	0.009627
Followups	0.112171
Product	-0.217461
Stars	0.099577
Trips	0.018898
Passport	0.260844
Score	0.051394
Car	-0.011508
Children	0.007421
Income	-0.130585

Actionable Insight



Ask for Passport

The first feature the sales rep should check is whether the potential customer has a passport. The model shows that owning a passport is the most relevant predictor for whether the customer will make a purchase. Nearly 35% of passport holders purchase a product vs 12% of customers without passports.

Target Product should not depend on Designation

The Product pitched is dependent on the customer's designation. We can see from the data that Executives are always pitched the basic package, Standard to Senior Managers, Deluxe to Managers, Super Deluxe to AVPs and King to VPs. However, this also correlates with a decrease in success rate. Therefore, the customer's designation alone should not determine the package pitched. A cost-benefit analysis of offering different packages should be done.

Examine Product Placement

Is there a reason most customers who make contact via self-inquiry choose the most basic package? Perhaps, the basic package is the most visible and is selected before the customer even sees the alternatives. Similarly, VPs are only seeing King packages. Research should be done on shopping habits and product placement should be tailored so that likely customers are exposed to the product with the highest success rate for their own demographic.

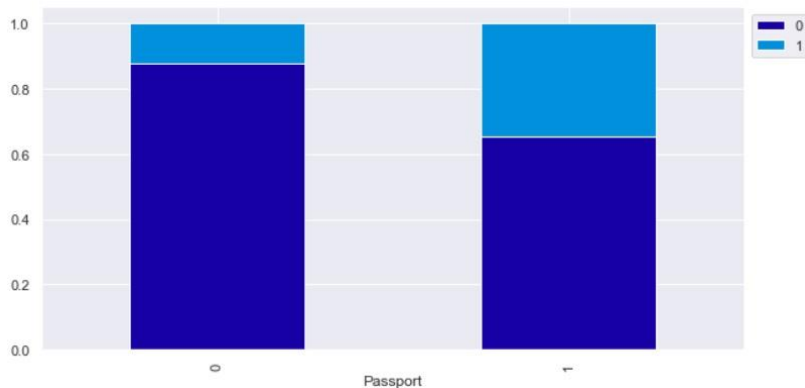
Apply the model every time

There are lots of exceptions to the top two rules of thumb. If someone does not have a passport, they might still make a purchase and so should not be dismissed out-of-hand. We know that customers with passports, although it is the best predictor, only make up 54% of all purchases. The sales rep should apply the model to every customer on an individual basis and let it predict whether a sales pitch is likely to result in a sale.

Actionable Insight: Passport

Ask for Passport

Checking to see if the customer has a passport is a great first start. We know that almost 35% of passport holders purchase travel products from Visit With Us. There are over twice as many people without passports (3466) as there are people with passports (1422) in the dataset. Growing the customer database is an obvious business tactic for any company but Visit With Us should be actively targeting customers with passports to give them a strategic edge.

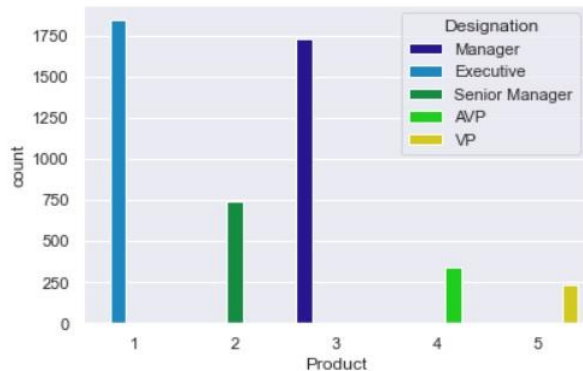


- Customers without passports (left) show about a 12% likelihood to purchase a product (light blue)
- Customers with passports (right), have a 35% likelihood to purchase a product (light blue)

Actionable Insight: Designation

Target Product should not depend on Designation

We found that Visit With Us ties product types with designations. We do not find this business practice to optimal. The two graphs below show that a) each designation type is pitched just one product type (left) and b) that as designation and product type rise, success rate plummets. We can see, for example, that AVP and VP are only marketed products 4 and 5 (left) and yet rarely make the purchase (right, red circle). The best remedy is to select 5 samples of each designation and pitch each sample a different product to see which one has the highest success rate per designation.



Actionable Insight: Product Placement

Examine Product Placement

Self Inquiry customers are most likely to be pitched product 1. This is the cheapest product. Factors for this may be because the sales rep does not know as much about these new customers and selects the product based on their designation or because product 1 has the most sales. However, it could also be because the customer specifically asked about this product because it is the most visible. Regardless of the reason, we see a problem: there is a negative feedback loop that encourages the selling of the cheapest product because it is the most likely to be purchased. This cycle needs to be broken. Some solutions would be to invite a more representative ratio of designations, to market different products to different designations and to emphasis products with the highest profitability, regardless of designation or how contact was made.

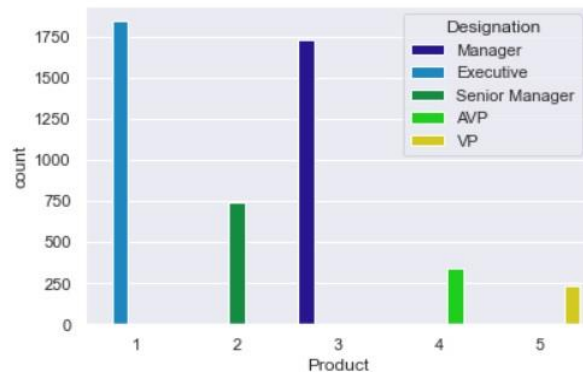


- Customers who contact the company through self inquiry more likely to be pitched product 1 (red circle)
- Customers contacted by the company are more likely to be pitched product 3 (green circle)

Actionable Insight: Designation

Target Product should not depend on Designation

We found that Visit With Us ties product types with designations. We do not find this business practice to optimal. The two graphs below show that a) each designation type is pitched just one product type (left) and b) that as designation and product type rise, success rate plummets. We can see, for example, that AVP and VP are only marketed products 4 and 5 (left) and yet rarely make the purchase (right, red circle). The best remedy is to select 5 samples of each designation and pitch each sample a different product to see which one has the highest success rate per designation.



03

Customer Data

Examines the most important variables affecting whether a customer is likely to purchase a travel package from Visit With Us





Customer Details

Number of customers:

4888

16 Distinct Variables:

- CustomerID
- ProdTaken
- Age
- TypeofContract
- CityTier
- Gender
- Occupation
- NumberOfPersonVisitng
- PreferredPropertyStar
- MaritalStatus
- NumberOfTrips
- Passport
- OwnCar
- NumberOfChildrenVisiting
- Designation
- MonthlyIncome

Customer Details Variables

CustomerID

Sequential identification number used to distinguish customers from one another. Number ranged from 200000 to 204887.



ProdTaken

Whether the customer has purchased a package or not (0: No, 1: Yes). Of the 4888 customers, 919 purchased a travel product, representing about 18.8% of all customers.



Age

The age of the customers ranges from 18 to 61, with a mean age of 37.6. There were 226 missing variables at the onset, but they were imputed with the mean.



TypeofContact

Customers were contacted about Visit With Us packages either via self-inquiry or by company invitation. About 71% were self inquiry while 29% were invited by the company. Variable name shortened to 'Contact' in the model.



Customer Details Variables

CityTier

City tier depends on the development of a city, population, facilities, and living standards. The categories are ordered i.e. Tier 1 > Tier 2 > Tier 3. Tier 1 has 3190 customers, Tier 2 has 198 and Tier 3 has 1500. Variable name shortened to 'City' in the model.



Occupation

The occupation of the customer had 4 distinct values: Salaried (2368), Small Business (2084), Large Business (434) and Free Lancer (2). Due to the small sample size of Free Lancer, it was removed from the data.



Gender

The gender of the customers were divided into two groups: Male (2916) and Female (1972).



NumberofPersonVisiting

This variable represents the total number of persons planning to take the trip with the customer. Variable name shortened to 'Visitors' in the model.



Customer Details Variables

PreferredPropertyStar

This feature represents the preferred hotel property rating by customer. There were three unique values: 3 stars (2993), 4 stars (913) and 5 stars (956). Variable name shortened to 'Stars' in the model.



MaritalStatus

There were four distinct categories for marital status: Married (2340), Divorced (950), Single (916) and Unmarried (682).



NumberofTrips

Average number of trips in a year by customer. Numbers ranged from 1 to 22, with the mean being 3.24 trips per year. Variable name shortened to 'Trips' in the model.



Passport

This feature asked whether the customer had a passport, with No = 0, and Yes = 1. There were 1422 customers with passports and 3466 customers without.



Customer Details Variables

OwnCar

This feature asked whether the customer owned their own vehicle. 3032 customers own a car while 1856 do not. Variable name shortened to 'Cars' in the model.



NumberOfChildrenVisiting

This variable indicates the total number of children with age less than 5 planning to take the trip with the customer. The numbers ranged from 0 to 3. Variable name shortened to 'Children' in the model.



Designation

This feature describes the designation of the customer in the current organization. There were 5 designations: Executive (1842), Manager (1732), Senior Manager (742), AVP (342) and VP (230). They were considered ranked in that order.



MonthlyIncome

Monthly income ranged from \$1000 to \$98,678. The mean monthly income was \$23,620. Variable name shortened to 'Income' in the model.





Customer Interaction Data

4 Distinct Variables:

- PitchSatisfactionScore
- ProductPitched
- NumberOfFollowups
- DurationOfPitch

Customer Interaction Variables

PitchSatisfactionScore

The sales pitch satisfaction score ranged from 1 to 5. The most frequently occurring score was a 3 (1478). Variable name shortened to 'Score' in the model.



ProductPitched

This variable describes the product pitched during the presentation with the customer. There were five different products: Basic (1842), Standard (1732), Deluxe (742), Super Deluxe (342), King (230). The variable name was shortened to 'Product' in the model.



NumberOfFollowups

Total number of follow-ups has been done by the salesperson after the sales pitch. The numbers ranged from 1 to 6, with 4 being the most frequent (2068). Variable name shortened to 'Followups' in the model.



DurationOfPitch

This variable is the duration of the pitch by a salesperson to the customer. Feature ranges from 5 minutes to 127 minutes. The mean is 15.5mins. Variable name shortened to 'Duration' in the model.



Customer Profiles



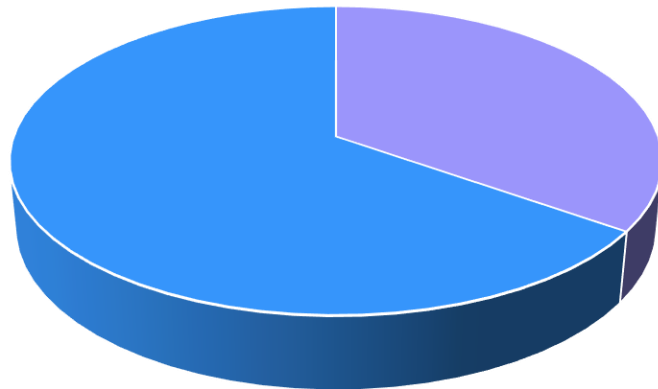
Customer Profiles



Top Feature

Passport

Does the customer own a passport?



■ Yes ■ No



35%

Over one third of customers with passports purchase travel packages



12%

Only 12% of customers who do not have passports purchase packages



Key Recommendation:

Customers with passports are ready to go! When growing the customer database, try to target new customers who already possess passports. They are almost three times as likely to purchase a package.

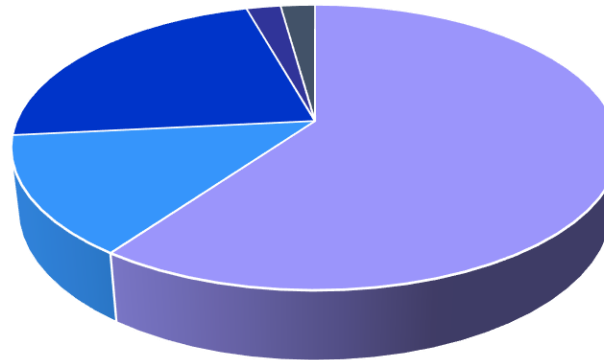
Customer Profiles



Top Feature

Product

Products are chosen based on the customer's designation and marketed accordingly. Executives are pitched basic packages, managers are pitched deluxe, senior managers are pitched standard, AVPs are pitched super deluxe and VPs are pitched king packages.



- Basic
- Deluxe
- King
- Standard
- Super Deluxe



60%

Almost 60% of all packages sold are basic packages



4%

The high-end Super Deluxe and King packages combine for only 4% of all sales



Key Recommendation:

The high-end super deluxe and king packages have very few sales in part because they are marketed to the smallest segments of the population. Start mix and matching products with designations to max sales.

Customer Profiles

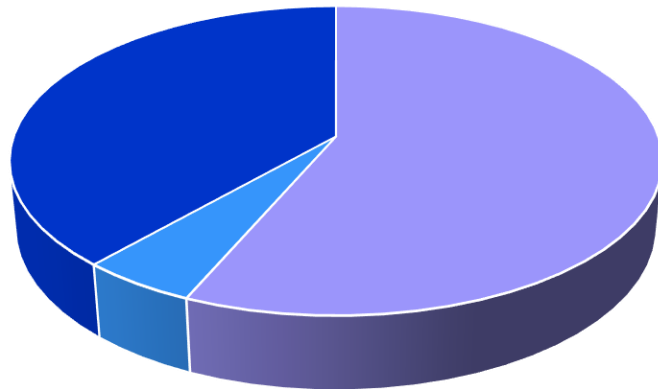


Top Feature

City

Product sales vary between Tier 1, 2 and 3 cities, but so too does their populations.

Marketing campaigns need to target high percentage areas, not just where volume is highest.



■ Tier 1 ■ Tier 2 ■ Tier 3



56%

Over 56% of all packages are sold to Tier 1 residents



23%

~23% of residents in Tier 2 & 3 cities purchase products, while only 16% of Tier 1 residents do



Key Recommendation:

A majority of customers in the dataset live in Tier 1 cities, but Tier 2 and Tier 3 are purchasing more products as a percentage of population. Its time to grow the customer base in these segments!

Customer Profiles

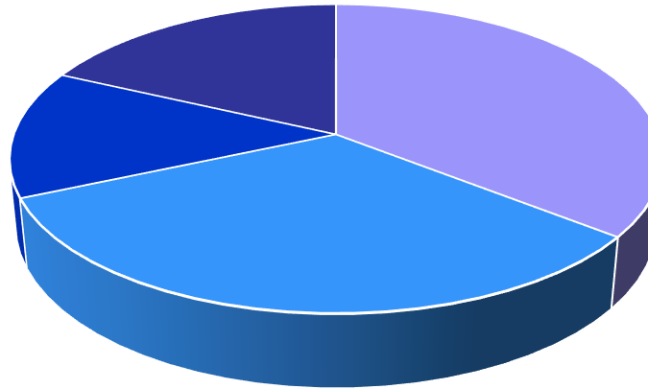


#4 Top Feature

Marital Status

Customers fall into one of three Marital Status categories:

- Married
- Single
- Divorced
- Unmarried



■ Married ■ Single ■ Divorced ■ Unmarried



36%

Almost 36% of packages sold are sold to married customers



33%

Single customers purchase packages at a rate of 33%; married at a rate of 14%!



Key Recommendation:

Target customers who have never stood at the alter! Married people represent more sales only because they represent almost half of all customers! Single customers lead the way as a percent (33%) followed by unmarried (24%). Married and divorced are lagging at 14% and 13% respectively.

Customer Profiles

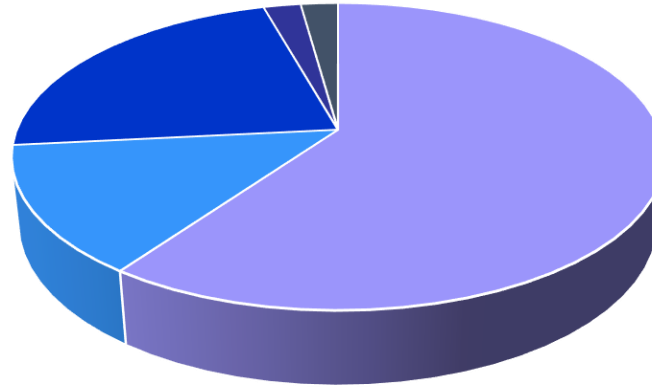


#5 Top Feature

Designation

Customers fall into one of five designation categories:

- Executive
- Manager
- Senior Manager
- AVP
- VP



■ Executive ■ Manager ■ Sr Manager
■ AVP ■ VP



60%

60% of all packages sold are sold to executives



4%

AVPs and VPs, despite having the highest incomes, combine for just 4%!



Key Recommendation:

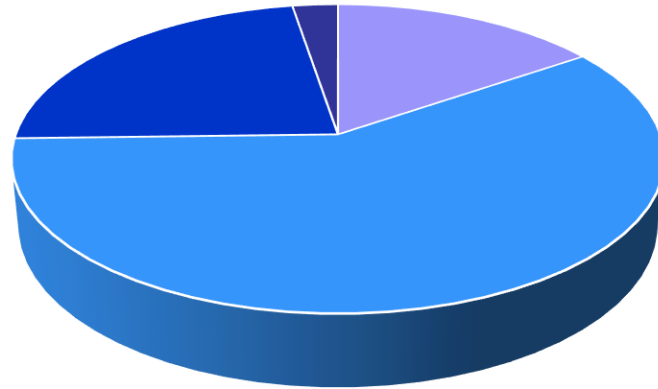
This variable is identical to the product variable because customers are only pitched specific products based on their designation. This variable should be removed from the model unless the company begins to change its sales mix. The company **SHOULD** change its sales mix!!!

Customer Profiles

Age

Age can be broken down into groups. For simplicity, we chose 4 common demographics:

- Gen Z: 18 to 25
- Gen Y: 26 to 40
- Gen X: 41 to 57
- Baby Boomers: 58+



■ Gen Z ■ Gen Y ■ Gen X ■ Baby Boomers



59%

Almost 60% of all products purchased were made by Gen Y



41%

Over 41% of all Gen Zers purchased a product (142/343)



Key Recommendation:

Gen Zers love to travel! They make up only a small portion of the total population (343/4888) but they purchase travel products almost twice as frequently as the second place boomers (21%). Start targeting customers 25 and under!



Customer Profiles: SUMMARY

Variable	Best Predictor
Passport	Yes – if the customer has a passport, there is 35% chance of a sale
Product	Basic package has the highest sales; when in doubt, offer the basic package.
City	Tiers 2 and 3 have nearly double the success rate of Tier 1; focus on growing these under-represented Tiers
Marital Status	Single and Unmarried customers are more than twice as likely to purchase travel products
Designation	Data mimics Product variable; therefore, executives are the most likely to purchase a product, but the only product they are currently offered is the basic package.
Age	The Gen Zers are the most likely group to purchase travel packages, despite being under-represented in the data set.



Key Recommendation:

Follow the model whenever possible, but if you get stuck, err on the side of offering customers who fall into the above groups the package.



04



Methodology

Overview of final model and highlights
of key takeaways from the research



What are we attempting to do?

Model can make wrong predictions as:

1. Predicting a customer will purchase a travel product when the customer ultimately does not.
2. Predicting a customer will not purchase a travel package, when in fact, the customer would have purchased one had (s)he been contacted.

Which case is more important?

1. If the model predicts a customer will purchase a travel product but the customer does not buy one, the company wastes resources on marketing spending. The company has indicated that marketing costs are quite high and is actively seeking to lower these costs.
2. If the model predicts customer will not purchase a product, when in fact, the customer would purchase a product, the company is losing business. This represents an opportunity cost. It threatens profits and can even jeopardize the long-term financial health of the company.

Which metric to optimize?

- We would want F1-Score to be maximized, the greater the F1-Score higher the chances of predicting both the classes correctly.

Bagging vs Boosting

In an attempt to develop the best model possible, we will use two different ensemble techniques and compare results.

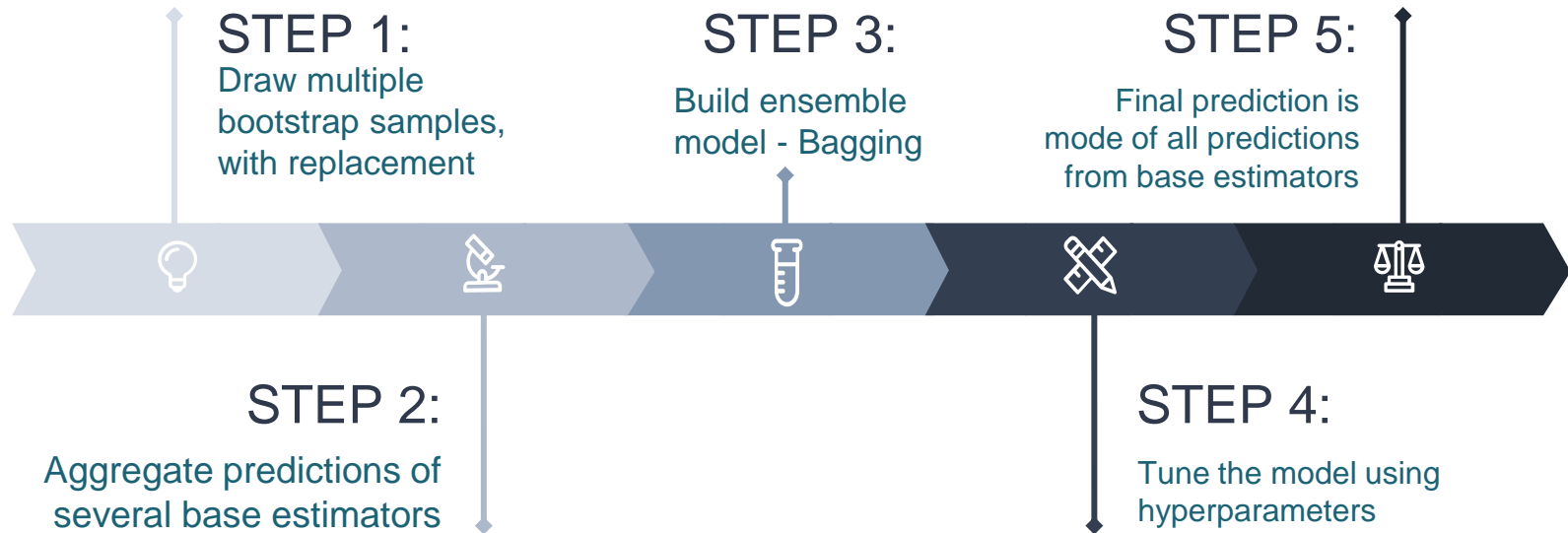
Bagging is a parallel training method where simple models are built independent of one another. The samples are drawn from the original dataset with replacement. Each simple model will have equal weight in the final prediction. Bagging helps reduce the variance of the model. We used Bagging Classifier and Random Forest.

Boosting is a sequential training method where simple models are improved in sequence with more weight added to the weaker models with better performance. Boosting helps reduce bias in the model. We used AdaBoost, Gradient Boosting and XG Boost.





Method 1a: Bagging Classifier



Bagging Classifier



STEP 1: Draw multiple bootstrap samples, with replacement

The Bagging Classifier model randomly samples data with replacement from the dataset. Each datapoint has equal probability of selection ($1/n$) at every stage. Sampling with replacement ensures any two sample values are independent and helps make base classifiers less correlated.



```
#base_estimator for bagging classifier is a decision tree by default  
bagging_estimator=BaggingClassifier(random_state=1)  
bagging_estimator.fit(X_train,y_train)  
  
BaggingClassifier(random_state=1)
```

STEP 2: Aggregate predictions of several base estimators

Python will perform the calculations on for each sample behind the scenes. Each sample makes a prediction, which can be viewed as the probability of that particular outcome occurring. Python then performs aggregation calculations and combines the predictions from the base estimators

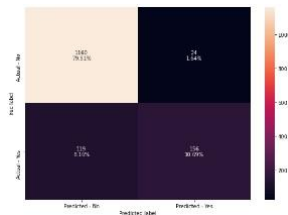


```
#base_estimator for bagging classifier is a decision tree by default  
bagging_estimator=BaggingClassifier(random_state=1)  
bagging_estimator.fit(X_train,y_train)  
  
BaggingClassifier(random_state=1)
```

Bagging Classifier

STEP 3: Build ensemble model - Bagging

Python builds the model behind the scene, but it can display the model as a confusion matrix



STEP 4: Tune the model using hyperparameters

We can change the max features, max samples, n estimators and even use logistic regression as base estimators:



```
BaggingClassifier(max_features=0.9, max_samples=0.9, n_estimators=40,  
                  random_state=1)
```

Bagging Classifier

STEP 5: Final prediction is mode of all predictions from base estimators

For classification models, the final prediction is the mode of predictions of base estimators. In other words, the predictions with the most 'votes' prevails. We can use Accuracy, Recall and Precision to evaluate the model



	Model	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1	Test_F1
0	Bagging classifier with default parameters	0.97	0.57	1.00	0.87	0.98	0.69
1	Tuned Bagging Classifier	1.00	0.63	1.00	0.93	1.00	0.75
2	Bagging classifier with base_estimator=LR	0.11	0.14	0.82	0.84	0.19	0.24

However, since we want to reduce both False Negatives and False Positives, we will focus on F1 scores (pink circle).

Method 1b: Random Forest



Random Forest



STEP 1: Draw sample, with replacement

The Random Forest model select $m < M$ features at random and replaces the sample after each selection. This helps ensure independence and reduces correlation.



```
# Choose the type of classifier.  
rf_estimator_tuned = RandomForestClassifier(random_state=1)
```

STEP 2: Find the best split on m to split the node

Python only selects a subset of features at random but uses the best split feature from the subset to split each node in a tree.



```
# Grid of parameters to choose from  
## add from article  
parameters = {"n_estimators": [150, 200, 250],  
              "min_samples_leaf": np.arange(5, 10),  
              "max_features": np.arange(0.2, 0.7, 0.1),  
              "max_samples": np.arange(0.3, 0.7, 0.1),  
              }
```


Random Forest



STEP 3: Grow tree to max depth

Python grows each tree to the largest possible extent unless a pre-defined depth is stated. We use a max depth of 150. Python will stop when the branching is exhausted or when it hits a depth of 150, whichever occurs first.



```
RandomForestClassifier(max_features=0.6000000000000001,  
                        max_samples=0.6000000000000001, min_samples_leaf=5,  
                        n_estimators=150, random_state=1)
```

STEP 4: Tune the model using hyperparameters

We can manipulate the hyperparameters to improve the model



```
# Grid of parameters to choose from  
## add from article  
parameters = {  
    "class_weight": [{0: 0.19, 1: 0.81}],  
    "n_estimators": [100, 150, 200, 250],  
    "min_samples_leaf": np.arange(5, 10),  
    "max_features": np.arange(0.2, 0.7, 0.1),  
    "max_samples": np.arange(0.3, 0.7, 0.1),
```

Random Forest

STEP 5: Final prediction is a weighted majority vote of all weak learners

The new data is predicted by aggregating the predictions of all the trees. We can use Accuracy, Recall and Precision to evaluate the model

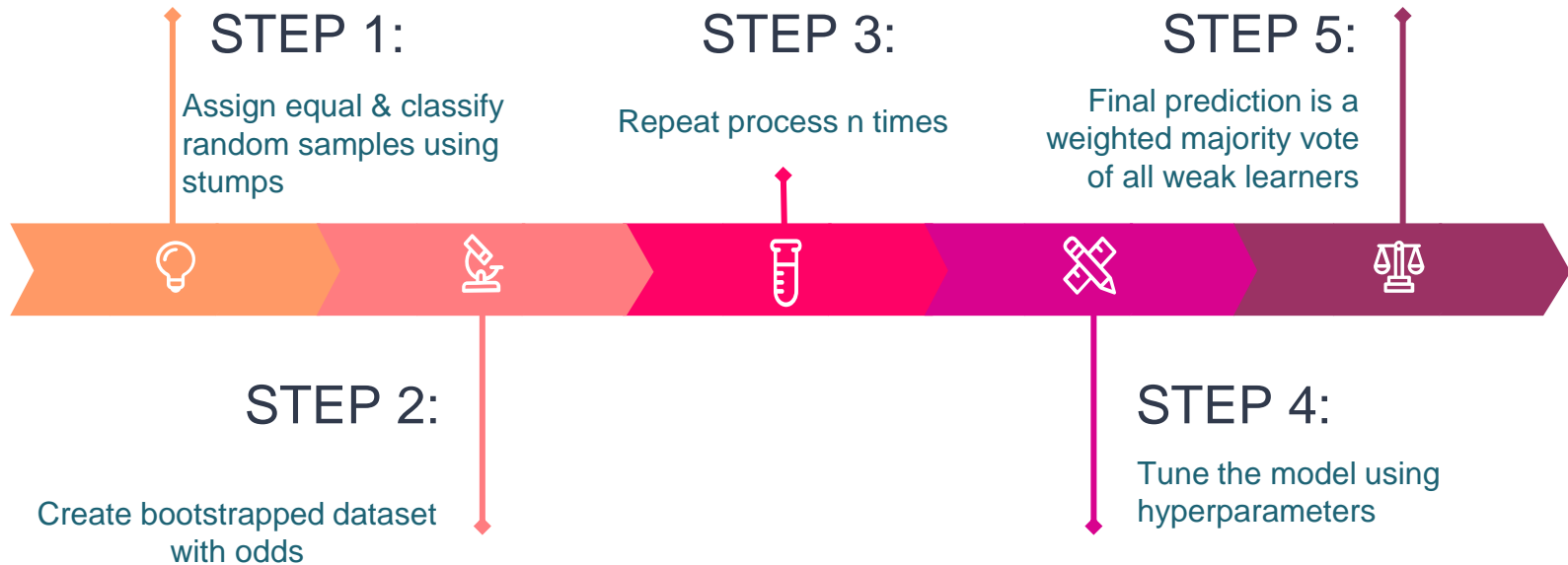


	Model	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1	Test_F1
3	Random Forest with default parameters	1.00	0.60	1.00	0.92	1.00	0.73
4	Tuned Random Forest Classifier	0.56	0.40	0.95	0.80	0.70	0.53
5	Random Forest with class_weights	0.82	0.65	0.69	0.62	0.75	0.63

However, since we want to reduce both False Negatives and False Positives, we will focus on F1 scores (pink circle).



Method 2a: AdaBoost



AdaBoost



STEP 1: Assign equal weights & classify random samples using stumps

AdaBoost uses sequential training on weak learners to improve the accuracy from prior learners. The first step is to give equal weights to all samples selected ($1/n$).



```
#Fitting the model  
ab_classifier = AdaBoostClassifier(random_state=1)  
ab_classifier.fit(X_train,y_train)
```

STEP 2: Create bootstrapped dataset with odds

The samples are bootstrapped as per the weights assigned and a weak learner is built on that sample. Once the weak learner is built, AdaBoost measures the importance of a weak learner based on the error made by that weak learner. New sample weights according to the correct and incorrect predictions are assigned to each sample.



```
#Calculating different metrics  
get_metrics_score(ab_classifier)
```

AdaBoost

STEP 3: Repeat process n times

AdaBoost repeats the process n times. Each time, a new bootstrapped dataset is created with the odds of each sample being chosen based on their new sample weights. Predictions are made after the nth repetition.



```
Accuracy on training set : 0.8636096413874191
Accuracy on test set : 0.8649760109664153
Recall on training set : 0.3828125
Recall on test set : 0.418181818181815
Precision on training set : 0.7802547770700637
Precision on test set : 0.756578947368421
```

STEP 4: Tune the model using hyperparameters

We can manipulate the hyperparameters to improve the model using `base_estimator`, `n_estimators` and `learning_rate`.



```
# Grid of parameters to choose from
parameters = {
    #Let's try different max_depth for base_estimator
    "base_estimator": [DecisionTreeClassifier(max_depth=1), DecisionTreeClassifier(max_depth=2),
                      DecisionTreeClassifier(max_depth=3)],
    "n_estimators": np.arange(10, 110, 10),
    "learning_rate": np.arange(0.1, 2, 0.1)
}
```

AdaBoost

STEP 5: Final prediction is a weighted majority vote of all weak learners

The new data is predicted by aggregating the predictions of all the trees. We can use Accuracy, Recall, Precision and F1 scores to evaluate the model



	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
0	AdaBoost Classifier	0.963610	0.864976	0.382812	0.418182	0.780255	0.756579	0.513627	0.538642
1	Tuned AdaBoost Classifier	0.965021	0.887594	0.842187	0.578182	0.967684	0.768116	0.900585	0.659751

However, since we want to reduce both False Negatives and False Positives, we will focus on F1 scores (pink circle).

Method 2b: Gradient Boosting



Gradient Boosting



STEP 1: A weak learner is built on a subset of original dataset

Gradient Boosting starts by using pseudo residuals, which are initial guesses of the model, to predict the value of the target. It makes predictions, which are essentially weak learners



```
#Fitting the model  
gb_classifier = GradientBoostingClassifier(random_state=1)  
gb_classifier.fit(X_train,y_train)
```

STEP 2: Calculate residuals

Gradient Boosting calculates the errors after predictions by the weak learners and compares the predictions to the actual values. Unlike AdaBoost, Gradient Boosting does not change the weights; it fits the next weak learner to the residuals of the previous weak learner.



```
#Calculating different metrics  
get_metrics_score(gb_classifier)
```

Gradient Boosting



STEP 3: Grow the tree

Gradient Boosting uses an additive model approach, building a tree in sequence, adding one tree at a time. It optimizes loss function using gradient descent. The process is repeated until the maximum number of trees specified is reached or the residual gets very small. At this stage, the model is able to make predictions.



```
Accuracy on training set : 0.8932980599647267
Accuracy on test set : 0.886223440712817
Recall on training set : 0.490625
Recall on test set : 0.4690909090909091
Precision on training set : 0.8945868945868946
Precision on test set : 0.8657718120805369
```

STEP 4: Tune the model using hyperparameters

We can manipulate the hyperparameters to improve the model. Some features we can manipulate include: `n_estimators`, `subsample` and `max_features`.



```
# Grid of parameters to choose from
parameters = {
    "n_estimators": [100, 150, 200, 250],
    "subsample": [0.8, 0.9, 1],
    "max_features": [0.7, 0.8, 0.9, 1]
```

Gradient Boosting

STEP 5: Final prediction is a weighted majority vote of all weak learners

The new data is predicted by aggregating the predictions of all the trees. We can use Accuracy, Recall and Precision to evaluate the model



	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
2	Gradient Boosting Classifier	0.893298	0.886223	0.490625	0.469091	0.894587	0.865772	0.633703	0.608491
3	Tuned Gradient Boosting Classifier	0.932099	0.896504	0.673438	0.527273	0.951435	0.873494	0.788655	0.657596

However, since we want to reduce both False Negatives and False Positives, we will focus on F1 scores (pink circle).



Method 2c: XGBoost



XG Boost

STEP 1: Set initial prediction to 0.5 (binary)

XG Boost sets the initial prediction for all observations as 0.5 for binary classification problems.



```
#Fitting the model  
xgb_classifier = XGBClassifier(random_state=1, eval_metric='logloss')  
xgb_classifier.fit(X_train,y_train)
```

STEP 2: Calculate residuals

It calculates the residuals for each observation by comparing the predictions to the actual values.



```
#Calculating different metrics  
get_metrics_score(xgb_classifier)
```

XG Boost



STEP 3: Grow the tree to predict residuals

XG Boost uses gain calculated from similarity score to find the best split while growing the tree. It calculates an output value for each leaf node. It repeats this step until there is no reduction in residuals or the number of estimators is reached. At this point, the tree can make predictions on the dataset.



```
Accuracy on training set : 0.9988242210464433
Accuracy on test set : 0.9273474982864977
Recall on training set : 0.99375
Recall on test set : 0.6981818181818182
Precision on training set : 1.0
Precision on test set : 0.8930232558139535
```

STEP 4: Tune the model using hyperparameters

We can manipulate the hyperparameters to improve the model. Some features we can manipulate include: `n_estimators`, `scale_pos_weight`, `subsample`, `learning_rate`, `colsample_bytree`, `colsample_bylevel`.



```
# Grid of parameters to choose from
parameters = {
    "n_estimators": [10,30,50],
    "scale_pos_weight": [1,2,5],
    "subsample": [0.7,0.9,1],
    "learning_rate": [0.05, 0.1,0.2],
    "colsample_bytree": [0.7,0.9,1],
    "colsample_bylevel": [0.5,0.7,1]
}
```

XG Boost

STEP 5: Final prediction is a weighted majority vote of all weak learners

The new data is predicted by aggregating the predictions of all the trees. We can use Accuracy, Recall and Precision to evaluate the model



	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
4	XGBoost Classifier	0.998824	0.927347	0.993750	0.698182	1.000000	0.893023	0.996865	0.783673
5	Tuned XGBoost Classifier	0.921517	0.867032	0.628125	0.450909	0.932715	0.742515	0.750700	0.561086

However, since we want to reduce both False Negatives and False Positives, we will focus on F1 scores (pink circle).

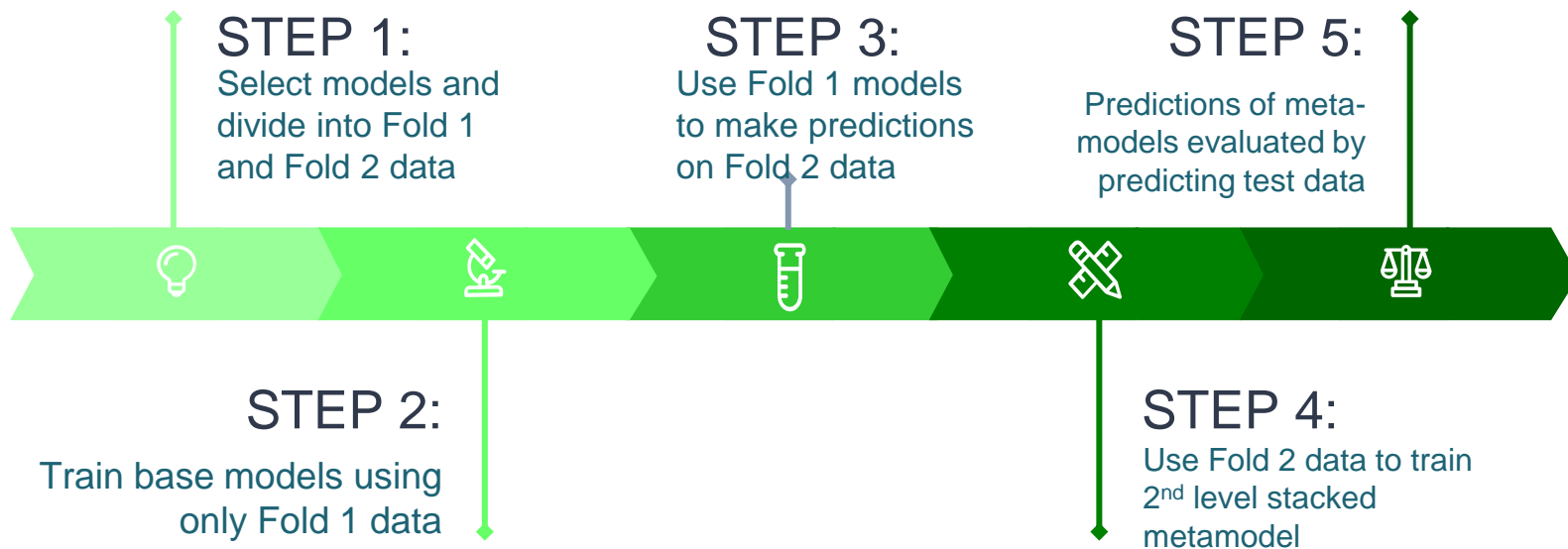
Techniques to Evaluate Bagging Models

- `n_estimators`: The number of trees in the forest, default = 100.
- `max_features`: The number of features to consider when looking for the best split.
- `class_weight`: Weights associated with classes in the form `{class_label: weight}`. If not given, all classes are supposed to have weight one. The frequency of class 0 is 21.2% and the frequency of class 1 is 18.8% in the data to prevent 0 from becoming dominant, we can pass a dictionary `{0:0.19,1:0.81}` to the model to specify the weight of each class and the random forest will give more weightage to class 1.
- `bootstrap`: Determines if bootstrap samples are used when building trees. If False, the entire dataset is used to build each tree, default=True.
- `max_samples`: If bootstrap is True, then the number of samples to draw from X to train each base estimator. If None (default), then draw N samples, where N is the number of observations in the train data.
- `oob_score`: The out-of-bag (OOB) error is the average error for each observation calculated using predictions from the trees that do not contain that observation in their respective bootstrap sample. This allows the RandomForestClassifier to be fit and validated whilst being trained, default=False.

Techniques to Evaluate Boosting Models

- `base_estimator`: The base estimator from which the boosted ensemble is built. By default the base estimator is a decision tree with `max_depth=1` (AdaBoost)
- `n_estimators`: The maximum number of estimators at which boosting is terminated. (AdaBoost Default value is 50; Gradient Default value is 100)
- `learning_rate`: Learning rate shrinks the contribution of each classifier by `learning_rate` (Ada, Gradient, XGBoost)
- `Init`: An estimator object that is used to compute the initial predictions. If 'zero', the initial raw predictions are set to zero. By default, a `DummyEstimator` predicting the classes priors is used (Gradient)
- `subsample`: The fraction of samples to be used for fitting the individual base learner, `default=1.0` (Gradient)
- `max_features`: The number of features to consider when looking for the best split, `default=None` (Gradient)
- `gamma`: A node is split only when the resulting split gives a positive reduction in the loss function. Gamma specifies the minimum loss reduction required to make a split. Higher the gamma value, lesser the chances of overfitting (XGBoost)
- `scale_pos_weight`: Control the balance of positive and negative weights. It can have any positive value as input. It helps in imbalanced classification problems (XGBoost)

Bonus Method: Stacking Classifier



Stacking Classifier



STEP 1: Select models and divide into Fold 1 and Fold 2 data

Stacking classifier can use different models as its base model. Here, we select Random Forest, Gradient Boosting, Decision Tree as our Fold 1 Data.



```
estimators = [('Random Forest', rf_estimator_weighted), ('Gradient Boosting', gbc_tuned), ('Decision Tree', dtree_estimator)]
```

STEP 2: Train base models using only Fold 1 data

We can train the 2 or more base models on Fold 1 so that it predicts outcome probabilities on Fold 2.



```
stacking_classifier.fit(X_train, y_train)
```

Stacking Classifier

STEP 3: Use Fold 1 models to make predictions on Fold 2 data

The Level-1 Model (Meta-Model) learns how to best combine the predictions of the base models.



```
estimators = [('Random Forest', rf_estimator_weighted), ('Gradient Boosting', gbc_tuned), ('Decision Tree', dtree_estimator)]  
stacking_classifier = StackingClassifier(estimators=estimators, final_estimator=final_estimator)
```

STEP 4: Use Fold 2 data to train 2nd level stacked metamodel



```
StackingClassifier(estimators=[('Random Forest',  
                                RandomForestClassifier(class_weight={0: 0.188,  
                                                                    1: 0.812},  
                                                        max_features=0.4000000000000001,  
                                                        max_samples=0.6000000000000001,  
                                                        min_samples_leaf=9,  
                                                        n_estimators=150,  
                                                        random_state=1))),
```

* Partial output

Stacking Classifier

STEP 5: Predictions of meta-models evaluated by predicting test data

The new data is predicted by stacking the models on top of one another. We can use Accuracy, Recall and Precision to evaluate the model



Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
Stacking Classifier	0.92	0.86	0.91	0.76	0.72	0.60	0.80	0.67

However, since we want to reduce both False Negatives and False Positives, we will focus on F1 scores (pink circle).

05



Recommendations

Demonstrates how to apply the key findings of the study to improve business strategy



KEY ACTION ITEMS

	Design Dashboard	Improve Dataset	Evolve Decision Tree
Key Action 1	Design a user-friendly dashboard where categorical variables can be selected from drop down list and numerical variables can be inputted.	The dataset has limitations, especially for underrepresented customers such as Free Lancers, and even VPs who are not numerous.	As more data is added, the structure of the model will change. The model will have to be re-run and altered, but this is an evolution which will only help to improve accuracy in the future.
Rationale	The ensemble technique method can be difficult for a user comprehend and follow decision-making steps to the logical conclusion.	As more data is entered, False Negatives and False Positives will continue to go down as more of the data is easily explained.	With more data, variables that are currently not helpful may turn out to provide key information in more evolved models.
Key Action 2	Develop a Visit With Us application so sales reps can look up a customer before a sales pitch and know what products to offer and the expected success rate	Research the behaviour of customers regarding travel planning and decision making around when and where they travel.	Create exit surveys to learn important reasons why the customer chose a certain package or even why the customer declined a product.
Rationale	App would immediately allow sales rep to see the success rate predictor and if it meets a predetermined threshold, the sales rep can decide to make the pitch or move on to another customer.	Getting as much behavioral information from the customer as possible can help improve success rate by knowing when to call and which product to offer.	There may be other variables that we do not understand causing customers to choose certain packages or not. Limited vacation time, cost of the product, or even timing may be factors.

Sample Dashboard

Does the customer hold a passport?

☐ Yes

☐ No

Which product is best?

Deluxe

- Basic
- Standard
- Deluxe
- Super Deluxe
- King

What type of city?

Tier 1

- Tier 2
- Tier 3

What is the customer's marital status?

Single

- Single
- Married
- Divorced
- Unmarried

What is the customer's age?

35

- 31
- 32
- 33
- 34
- 35
- 36
- 37
- 38

What is the customer's designation?

Executive

- Executive
- Manager
- Sr Manager
- AVP
- VP

STRATEGIC BUSINESS TOOL

Revenue

This customer model will allow Visit With Us to quickly assess existing customers and determine whether they are likely to purchase the new wilderness travel package.

Currently, Visit With Us is attempting to re-sell to existing customers. The model can be applied to new customers as well.

Expenses

Visit With Us has to chief expenses: high marketing costs and opportunity costs associated with missing out on potential customers due to not making the pitch.

The False Negative means that they miss out on a serious customer who may go to a competitor. The False Positive results in high marketing costs wasted on a no sale.

Total Profit

By increasing the number of customers purchasing travel products, Visit With Us can increase revenues.

By decreasing the number of False Positives and False Negatives, Visit With Us can reduce expenses.

By combining these two strategies using the model as a guide, Visit With Us can grow total profits.



06



Conclusions

Model evaluation overview and
summary of key findings



Final Ensemble Model

After training the computer extensively, the best models are:

	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
1	Tuned XGBoost Classifier	0.92	0.87	0.83	0.69	0.77	0.65	0.80	0.67
2	Stacking Classifier	0.92	0.86	0.91	0.76	0.72	0.60	0.80	0.67

Our goal was to maximize the test F1 scores, and both models have identical F1 scores on test data. They also have very similar test data accuracy scores and neither is overfitting the data beyond an acceptable measure. However, Tuned XG Boost Classifier has a higher recall score, therefore, it is better at minimizing False Negatives while maximizing F1 scores. The Stacking Classifier has a higher precision score, meaning it is better at minimizing False Positives while maximizing F1 scores. Management must decide the model they feel better aligns with their business goals.

Did the model improve?

Bagging Classifiers:

	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
0	Bagging classifier with default parameters	0.99	0.90	0.97	0.54	1.00	0.89	0.98	0.67
1	Tuned Bagging Classifier	1.00	0.93	1.00	0.65	1.00	0.93	1.00	0.77
2	Bagging classifier with base_estimator=LR	0.83	0.84	0.16	0.20	0.82	0.80	0.26	0.32
3	Random Forest with default parameters	1.00	0.92	1.00	0.61	1.00	0.93	1.00	0.74
4	Tuned Random Forest Classifier	0.91	0.87	0.55	0.39	0.95	0.81	0.70	0.53
5	Random Forest with class_weights	0.90	0.86	0.82	0.66	0.69	0.63	0.75	0.65

The above table shows that the F1 scores, our ultimate goal, ranged from 0.32 to 0.74. However, on some of the higher models, the data was overfitting. Hence, the best bagging model is the last model: Random Forest with class weights. It has a sufficiently high F1 score (0.65) but the model is not overfitting the data, as it is in Random Forest with default parameters. We can show a progression of improvements over the 6 models in this section.

Did the model improve?

Boosting Classifiers:

	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
4	XGBoost Classifier	1.00	0.93	1.00	0.70	1.00	0.89	1.00	0.78
5	Tuned XGBoost Classifier	0.92	0.87	0.83	0.69	0.77	0.65	0.80	0.67
1	Tuned AdaBoost Classifier	0.99	0.88	0.95	0.62	0.98	0.71	0.97	0.66
3	Tuned Gradient Boosting Classifier	0.93	0.89	0.65	0.52	0.94	0.84	0.77	0.65
2	Gradient Boosting Classifier	0.89	0.87	0.48	0.41	0.86	0.82	0.62	0.55
0	AdaBoost Classifier	0.85	0.85	0.34	0.36	0.68	0.68	0.46	0.48

The above table shows that the F1 scores on boosting models' test data ranged from 0.55 to 0.78. However, XGB Classifier (model 4) clearly had overfitting on the data. Hence, the best boosting model is the last model: Tuned XGBoost Classifier (model 5).. It has a sufficiently high F1 score (0.67) but the model is not overfitting the data, as it was before the model was tuned (model 4). We can show a progression of improvements over the 5 models in this section.

Did the model improve?



All Classifiers:

	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
1	Tuned XGBoost Classifier	0.92	0.87	0.83	0.69	0.77	0.65	0.80	0.67
2	Stacking Classifier	0.92	0.86	0.91	0.76	0.72	0.60	0.80	0.67
0	Random Forest with class_weights	0.90	0.86	0.82	0.66	0.69	0.63	0.75	0.65

This table shows our last model (Stacking Classifier) as compared with the best boosting and best bagging models. We can see that Stacking and XG Boost are essentially tied in test data F1 scores, which was our goal to maximize. For this reason, they are better than the Random Forest model. However, XG Boost does a better job reducing False Negatives (higher recall on test data) while Stacking Classifier does a better job reducing False positives (higher precision on test data). It is up to management at Visit With Us to choose the model they feel better aligns with their goals.

CONSUMER BEHAVIOUR

- 1 Customers who hold passports are more likely to purchase travel products.


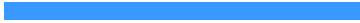
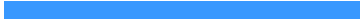
- 2 Customers are only offered product types based on their designation. Executives are most likely to purchase a product.

- 3 Currently, executives can only access the basic package, but they seem to have an appetite for this product

- 4 Customers in Tier 2 and Tier 3 cities are more likely to purchase travel products.

- 5 Unmarried and Single customers are more than twice as likely to purchase travel products.

MODEL BEHAVIOUR

- 1 The model can impute missing values when they are unknown, but it will work better if the sales rep inputs all the customer's information

- 2 Model will not work optimally if user 'guesses' categorical or numerical variables.

- 3 This model cannot predict the likelihood of a decision made by customers outside of the categorical variable ranges ie. a family of 5

- 4 Attempts to simplify the model to address the above issues will result in underfitting. Accuracy of the model is reduced.



THANKS!

Do you have any
questions?

youremail@freepik.com
+91 620 421 838
yourcompany.com



CREDITS: This presentation template was created by
Slidesgo, including icons by Flaticon, and infographics &
images by Freepik and illustrations by Stories.

Please keep this slide for attribution.