Datastorm 2021

Team Unknown
ds21-59

First we observed the data manually to get an idea about how the dataset looks like and what kind of machine learning techniques that can be used for this.

After glancing through the data set, we use observation techniques to observe the data set mathematically.

Then we observed how many unique values are there in each column to determine and differentiate categorical and continuous features. If the unique values are less, then it is a categorical feature and vice versa.

Then we checked whether the number of unique values that we found were equal in each dataset. If it is not we need to think of it to make the model more accurate. But the number of unique values are the same. Then we check whether there are some rows with null values. We use training and validation data sets to make a combined data set.

With the observations, we categorize categorical and continuous features.

Categorical features

1. Gender
2. Ethnicity
3. Educational_level
4. Income
5. Country_region
6. Hotel_type
7. Meal_Type
8. Adults
9. Children
10. Visited_Previosly
11. Previous_Cancellation
12. Deposit_type
13. Booking_channel
14. Required_Car_Parking
15. Use_Promotion

Continuous features

1. Expected_checkin
2. Expected_checkout
3. Booking_date
4. Age

Then we encode the categorical data because some of them are from string format.
For the columns which have date/time data, we found the number of days from that date to today.
Then we use a scalar function to scale the continuous data.

We use different ML techniques (models) to test which one gives the best accuracy. Some of them had greater accuracy and others didn't. So we analyse those techniques by considering pros and cons using past experiences.

Naive bayes classifiers

Pros

- Easy to understand
- Efficient when estimating parameters
- Works well with high dimensional data

Cons

- We have to assume that the features are conditionally independent given the class is not realistic
- Normally don't have better generalization performances
- Confidence estimates were not very accurate

Since our dataset does not have high dimensions these naive based techniques are not better suited for this task. Also we would get low accuracy and less generalization performances.

Random Forest

We found that this technique is widely used and has very good results on many problems. So we decided to check Random forest. We found that when we use many trees the system got more stable and had a better generalization. We introduce random variation into tree-building inorder to maintain the diversity of ensembles of trees.

Pros

- Had an excellent prediction performance
- Didn't required careful normalization of features or extensive parameter tuning
- Can handle a mixture of features types
- Can be Easily parallelized

Cons

- Resulting models are hard to interpret as humans
- Not a good choice for very high dimensional tasks.

Since we didn't have very high dimensions, we found that the random forest method will give us a good F1 score and greater accuracy.

Gradient Boosted Decision Trees
When we are using this method we have to consider more about the learning rate. By changing the learning rate we could control how hard each new tree tries to correct the remaining mistakes. We found that a high learning rate creates more complex trees and low learning rate creates simpler trees.

We found two GBDT based techniques that did well in this task. They are AdaBoost and XgBoost.

Pros
- Gave greater accuracy in many problems
- Requires only modest memory and this is fast
- Doesn;t require careful normalization of features
- Can handle mixture of feature types

Cons
- Difficult to interpret as humans
- Requires careful tuning such as learning rate, n_estimators, max_depth
- Training can require significant computation
- Did not work well with high dimensional features

Since this task is not containing higher dimensional features we tried these 2 techniques. We carefully tuned the learning rate, so we got greater accuracy by using XgBoost and AdaBoost.

Neural network
Normally performance can be improved by introducing more hidden layers. Also we can control the weight on the regularization penalty by changing alpha.

Pros
- Effectively captured the complex features.

Cons
- Complex models require significant training time, data and customerization.
- Careful preprocessing of the data is required
- Not a good choice when the features are from very different types.

Since this method requires significant training time and not a good choice when the features from different types, this method did not do well with our task. But still we got moderate accuracy and score.

Deep Learning
Pros
- Have significant gains over other machine learning approaches
- Does effective automatic feature extraction

Cons

- need s huge amount of training data
- Needs huge amount of computation power

In this task we don't have a huge amount of training data. So this method is not a better one for this task.

- Finally after analysing those pros and cons and our task we selected some techniques. They are AdaBoost, XgBoost,Knearest neighbour, Neural Networks, and Random forest.

After observing the dataset we figured out that the data set is not a balanced one. So we use upsampling and downsampling techniques to balance the dataset. But downsampling didn't give us a better result because some of the valuable data will be lost after downsampling.
So we use **SMOTETomek** to randomly upsample the data set.

This is how the data set looks like before and after sampling.

```
The number of classes before fit({1: 20584, 2: 4364, 3: 2275})
The number of classes after fit({3: 20579, 2: 20569, 1: 20564})
```

We use **correlation matrix**, **pair plots** and **ExtraTreesClassifier** as an ensemble technique to figure out how the features are depending on each other.

Plots were added to this link below.
https://drive.google.com/drive/folders/1zZOywlZ8JX7Z6qq9atr7AKHzeWFvbxfa?usp=sharing

Expected revenue loss = 360588.7

The steps we took to get this value were added in the code with clear comments.

❑ Additional attributes that should collect by the hotel management

1. Whether the person is a foreigner or not

We think that this fact may have some relation with the final reservation status. Because if the person is a foreigner, there is a high probability of low cancellation rate. Because most of the foreigners will be travelers or participating in some kind of business activities. So they will have some kind of well structured planning for those days. But on the other hand most of the local people may have a higher rate of cancelation than foreigners.

2. Whether the reserved day is some kind of special day or not

Normally people choose some special days such as holidays to travel or doing some kind of leisure activity. So in those days cancellation rate will be different than a normal day. So we think this fact should also be considered.

3. Informations such as reason behind this reservation or there current planings

We have to face some kind of problems when gathering these types of information. Because they may have some kind of privacy issues. So we can include some kind of categories such as

- Business reason
- Traveling
- Education reason
- ect...

I think if we have the above types of data the model will be predicted more accurately in the future.

❑ Interventions that will reduce the revenue loss

1. Give some kind of extra features to the persons that have a good record in the past.
2. Collect the data about the previous reservations of users and give them some kind of score based on their records. Then use those data to improve the model and give some promotions to them

The code for the final model was uploaded to the slot given to upload the code. And the other codes (for the other ML models) are in the github repo.
https://github.com/tharakaWijesundara/Storm