Project report

Abstract

This study analyzed a data set under different conditions in order to find some correlations and predict the status of reservation or cancelation of clients for months of April to August for year 2017. This project includes three major steps which are data cleaning, data analysis, finding correlation and making prediction based on different models. Finally, there is a comparison between the resulted values of proposed models and actual data from the original dataset and according to the analysis, very good agreement has been observed between the predicted and real values. In addition, the optimized models for this data frame have been suggested.

Introduction

Hotel management and services are quite important in tourism industry and there are a lot of competitions in this regard. If a management of a hotel gives the better ideas and services, it has much better opportunity to get more benefits from this market. Particularly, in the modern countries and high technology world, if a person uses a technology better rather than others; he will get a great success to reach his goal. One of these features is prediction by software which can help hotel management make a better plan and give more efficient services. In this study, status of reservation for some hotels is investigated by python software.

Project approach and analysis

This project has three major steps. These are:

1-Data cleaning, 2-Data analysis, 3-Modelling and prediction

1-Data cleaning

There is a statistical research which has collected a lot of data regarding hotel booking and its different options. According to this extensive statistical study, all of values are in a dataset. This data frame includes 118897 rows and 28 columns. In addition; there are two types of hotels for this study which are resort hotel and city hotel. Figure 1 illustrates the five rows of this dataset.

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights
0	Resort Hotel	0	342	2015	July	27	1	0
1	Resort Hotel	0	737	2015	July	27	1	0
2	Resort Hotel	0	7	2015	July	27	1	0
3	Resort Hotel	0	13	2015	July	27	1	0
4	Resort Hotel	0	14	2015	July	27	1	0

5 rows × 29 columns

Figure 1- Five rows of the data frame about hotel booking

The columns of the data frame which are required features for the analysis have been studied for different clients and put them in the table. These columns names are shown in the table 1.

Column number	Column name
1	hotel
2	is_canceled
3	lead_time
4	arrival_date_year
5	arrival_date_month
6	arrival_date_week_number
7	arrival_date_day_of_month
8	stays_in_weekend_nights
9	stays_in_week_nights
10	adults
11	children
12	babies
13	meal
14	country
15	market_segment
16	distribution_channel
17	is_repeated_guest
18	previous_cancelation
19	previous_booking_not_cancelled
20	resevred_room_type
21	assigned_room_type
22	booking_changes
23	deposit_type
24	days_in_waiting_list
25	customer_type
26	adr
27	required_car_parking_spaces
28	total_special_requests
29	reservation_status

Table 1-Name of columns

In order to clean the data frame, some columns which are repetitive have been removed. In the next step, the null values which existed in the big table was found and removed. It is quite important to talk about two specific columns of the dataset in the step of data cleaning. These columns are ‘meal’ and ‘reservation_status’. They are important features which have many null values. In fact, removing the corresponding rows of these null values is not a good idea, since a lot of values from other features can be removed by this method. So different method has been applied to these two columns called data imputation.

In order to impute the column of ‘meal’, the mode of the list for this column has been utilized for imputing. Because the list mode is the dominant factor rather than others and this number (mode) is much greater than counts of other unique values. So, imputing by the list mode for this column is an acceptable approach.

For next column which is ‘reservation_status’, the mode value cannot be used. Because this number is not dominant factor rather than counts of other unique values. Consequently, a model of machine learning for value prediction has been applied for data imputation of null values.

For this purpose, logistic regression model was initiated. After initial model, it was fitted based on the known features from the table. In this step, response variable is ‘reservation_status’ for those table rows which have null values. So, it is possible to replace the null values with the predicted numbers based on logistic regression. Figure 2 shows the code for data imputation in the data cleaning step.

```
X=dflog.values; y=(yi=='Check-Out').values
clf=LogisticRegression()
clf.fit(X,y)
X_prediction=dflog[yi=='No-Show']
y_pred=clf.predict(X_prediction)
print(y_pred.shape)
w=y_pred.reshape(1203,1)
y_temp=pd.DataFrame(w,columns=['reservation_status'])
y_temp.shape
#Assigning appropriate information for 'No-Show' values based on logistic regression
j=0
for k in range(118897):
    if yi[k]=='No-Show':
        yi.iloc[k]=y_temp.iloc[j][0]
        j=j+1
```

Figure2-The code for data imputation by logistic regression

All of methods were applied to clean the data frame. Finally, the cleaned dataset has been obtained. This big table is utilized for data analysis and machine learning.

2-Data analysis

After data cleaning, some graphs and correlation have been illustrated. This analysis includes histogram, scatterplot, normal distribution based on central limit theorem and linear regression. Figure 3 to 10 depict the graphs based on exploratory data analysis (EDA).

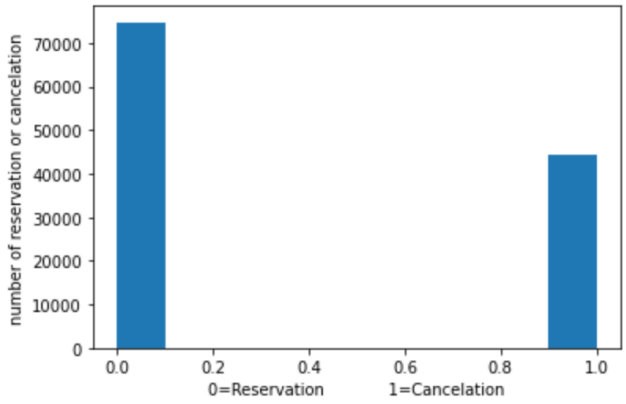

Figure3-Reservation/cancelation status for whole dataset

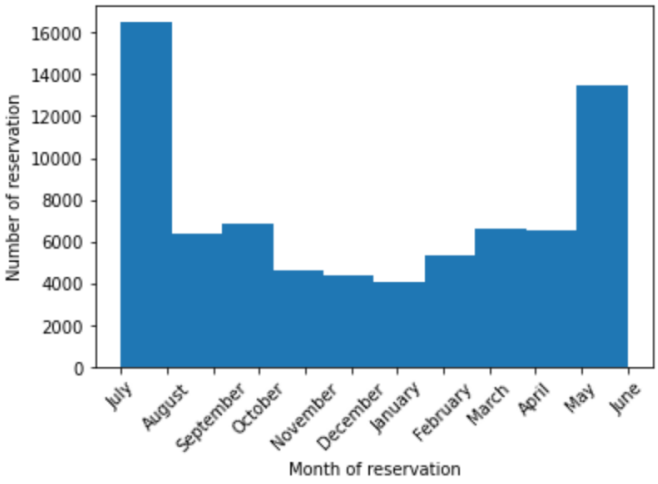

Figure4-Number of reservations for each month

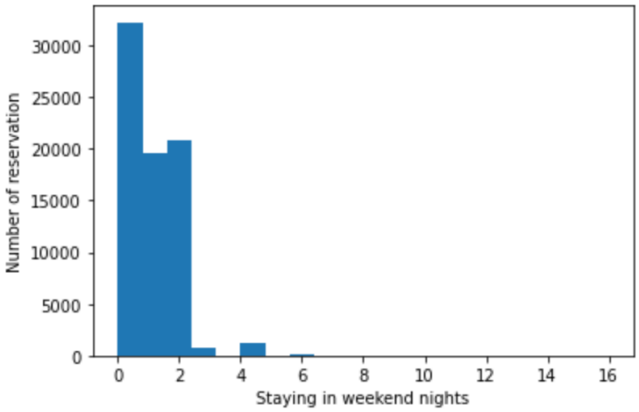

Figure5-Reservation status for weekend nights

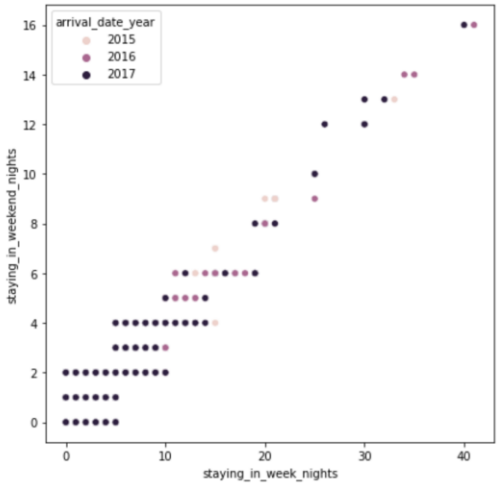

Figure6-Relationship between stays in weekend nights and stays in week nights

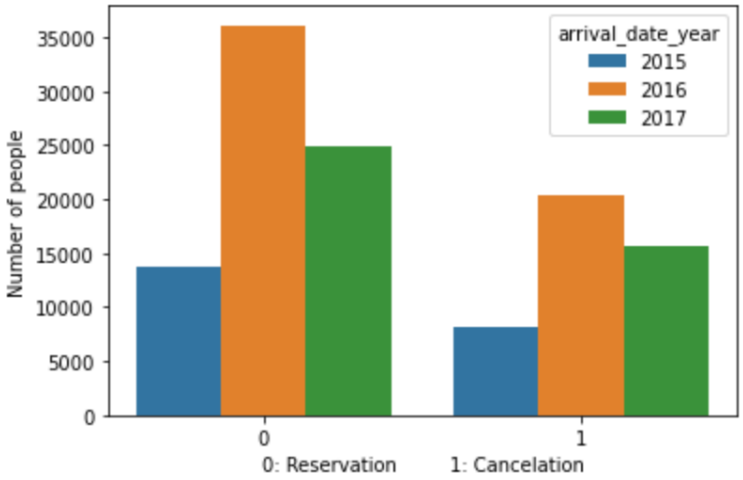

Figure7-Reservation status for the whole dataset based on different years

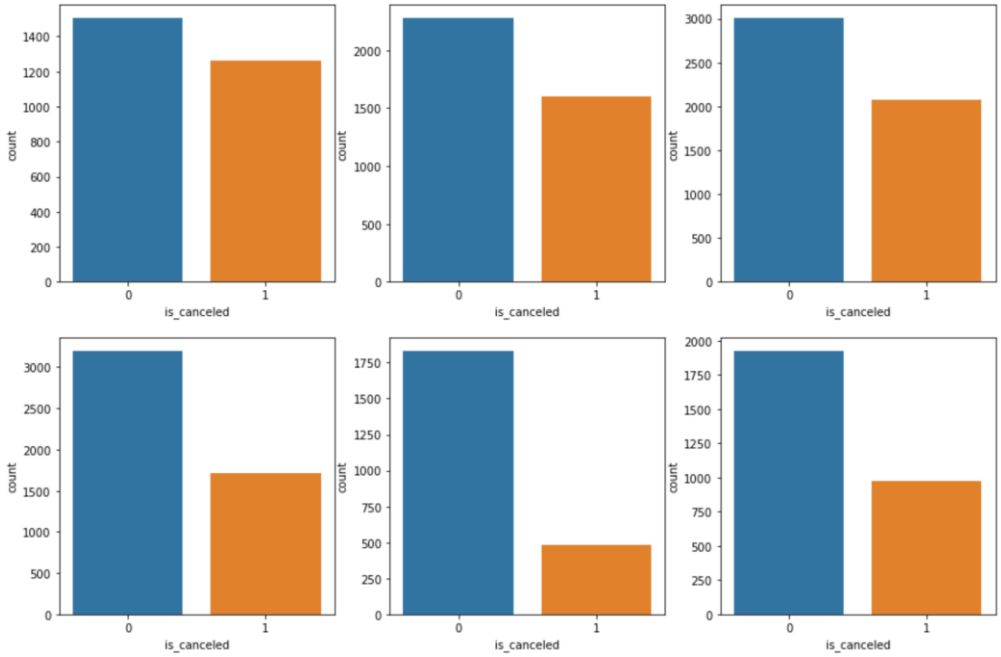

Figure8- Status of reservation from July to December 2015

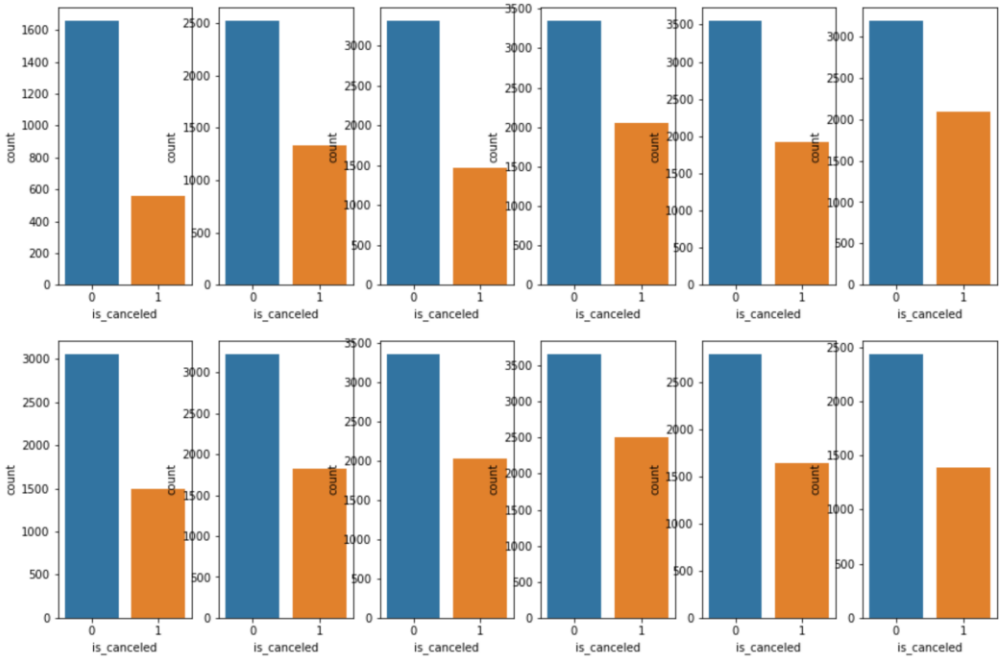

Figure9- Status of reservation from January to December 2016

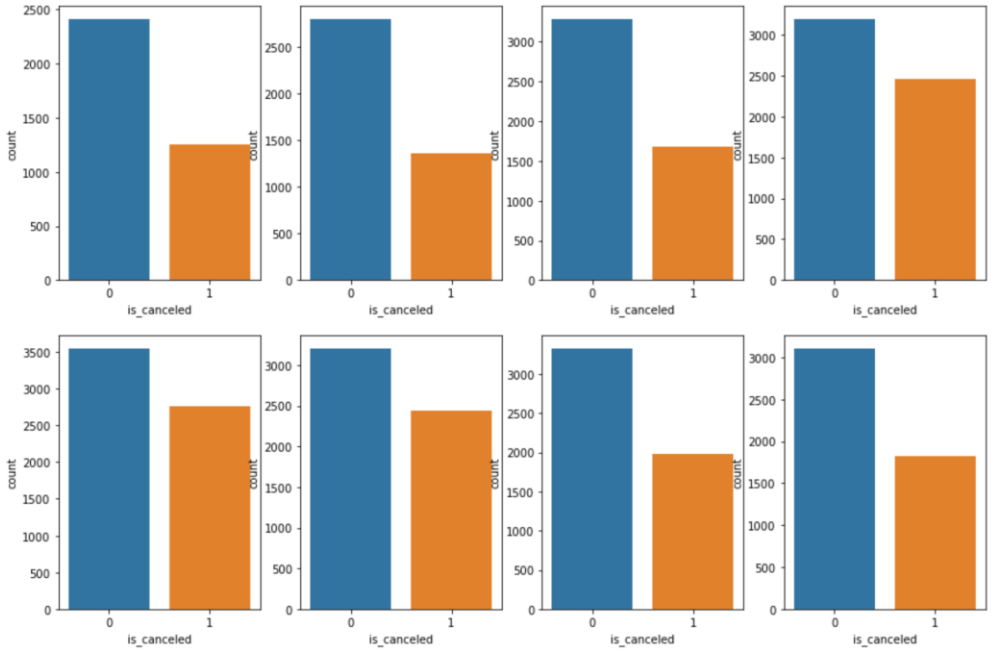

Figure10- Status of reservation from January to August 2017

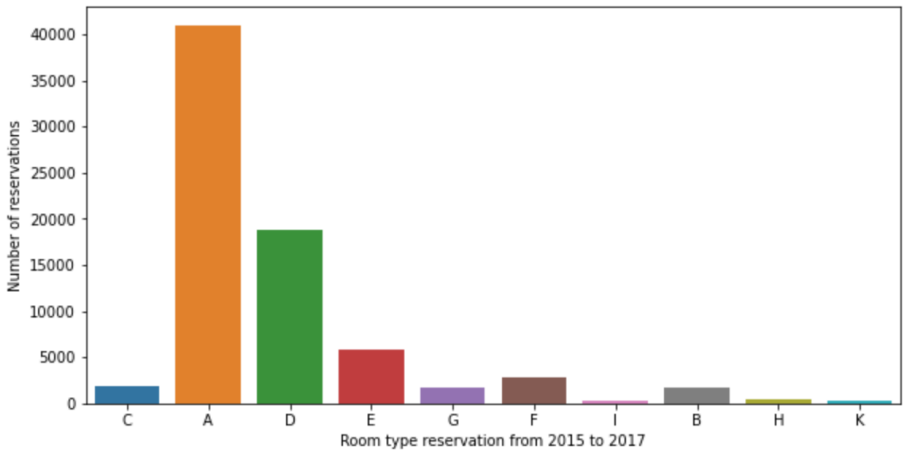

Figure11- Reservation status for room type

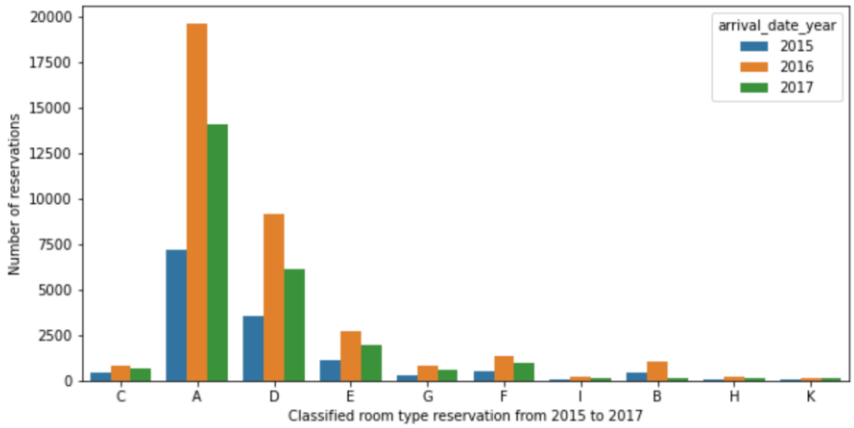

Figure12- Reservation status for room type based on different years

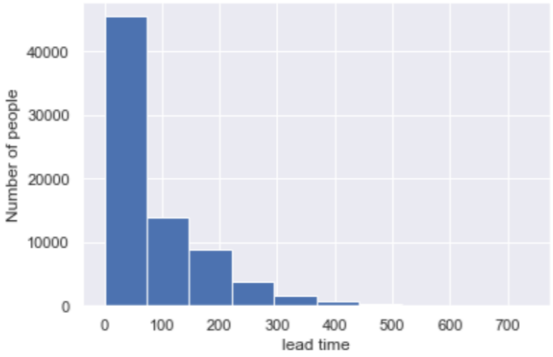

Figure13- Illustration of lead time for reservation

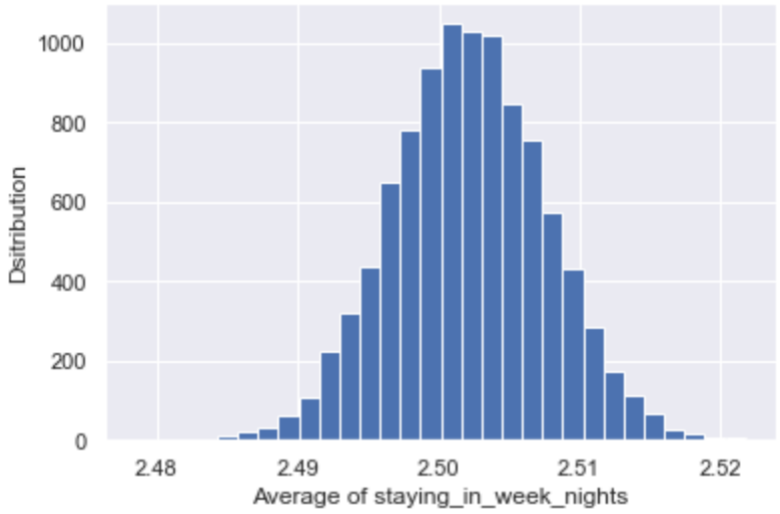

Figure14- Normal distribution of average value for staying in week knights

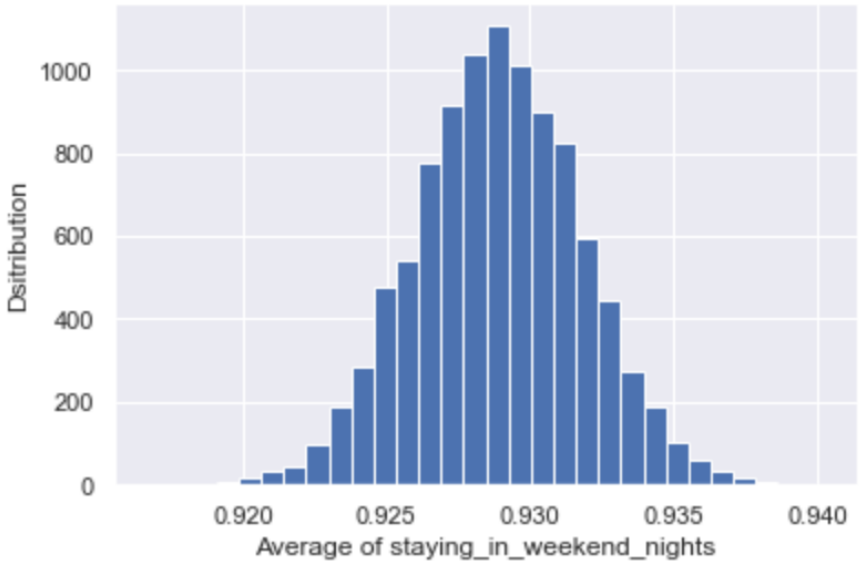

Figure15- Normal distribution of average value for staying in weekend knights

According to this analysis, it is observed these results:

Number of reservations for the whole data frame is more than 70000. On the other hand; the number of cancelations is almost 45000.

Based on this study, the best months for hotel booking are June and July according to figure 4.

Figure 9 shows an important result for year 2016, it expresses that the minimum cancelation occurs in January. For this graph the number of cancelations is approximately 31% of reservations. For other months in year 2016, there are lower differences. In other words, in January 2016, the difference of reservation is much higher rather than cancelation.

Based on figures 11 and 12, it is concluded that the popular class types of rooms for booking are A, D and E.

In addition, it is observed that the most frequent lead time for this study is between 1 to 100 days which is a little more than three months.

3-Modelling, assessment and prediction

In this step, modelling and prediction are performed for five months of year 2017 (April to August). At first, there are some categorical variables for the table which are required to convert them to numbers. This step is so important for machine learning. Figure 16 shows you the code for getting dummy variables for the data frame.

```
#Generation dummies for the full dataset
df_dummies_ML=pd.get_dummies(df, columns=['hotel','meal','country','market_segment',
                                          'distribution_channel','reserved_room_type','assigned_room_type',
                                          'deposit_type','customer_type','reservation_status'], prefix='C')
dfa=df_dummies_ML[(df_dummies_ML['arrival_date_year']==2017)]
index_list=(dfa[(dfa['arrival_date_month']==4) | (dfa['arrival_date_month']==5) |
                (dfa['arrival_date_month']==6) | (dfa['arrival_date_month']==7) |
                (dfa['arrival_date_month']==8)]).index
df_temp=df_dummies_ML.drop(index=index_list, axis=0)
X_ML=df_temp.drop('is_canceled', axis=1)
y_ML=df_temp['is_canceled']
X_five=(dfa[(dfa['arrival_date_month']==4) | (dfa['arrival_date_month']==5) |
                (dfa['arrival_date_month']==6) | (dfa['arrival_date_month']==7) |
                (dfa['arrival_date_month']==8)]).drop(columns='is_canceled', axis=1)
y_five=dfa[(dfa['arrival_date_month']==4) | (dfa['arrival_date_month']==5) |
                (dfa['arrival_date_month']==6) | (dfa['arrival_date_month']==7) |
                (dfa['arrival_date_month']==8)]['is_canceled']
y_five_series=y_five.values
```

Figure 16-The code for getting dummy variables

After getting dummy variable, the table got 243 columns. Besides, the corresponding rows of five months for year 2017 have been extracted for machine leaning. Therefore, the new dataset for applying functions has 91058 rows and 243 columns. In addition, the column of ‘is_canceled’ from the dataset is response variable for the modelling.

For making prediction, some functions of machine learning are applied for the data frame. After initial definition and training, every function is analyzed regarding precision and score.

Based on this analysis, the best models are proposed for prediction.

3-1-Ridge model

By this model definition, it is possible to predict the values after selecting appropriate parameters according to precision assessment. The code for Ridge model analysis is shown in figure 17.

```
X=X_ML.values
y=y_ML.values
la=[0.1,1,10,100,1000]
X_train, X_test, y_train, y_test=train_test_split(X,y,test_size=0.3,random_state=42)
for a in la:
    ridge=Ridge(alpha=a)
    ridge.fit(X_train, y_train)
    y_pred=ridge.predict(X_test)
    print('alpha=',a,'  score=',ridge.score(X_test, y_test))
```

```
alpha= 0.1   score= 0.9657328018649981
alpha= 1   score= 0.9658014857152563
alpha= 10   score= 0.96601033998456
alpha= 100   score= 0.9660239838787819
alpha= 1000   score= 0.9641476014642453
```

```
#Modelling based on Ridge method
for a in la:
    ridge=Ridge(alpha=a)
    kf=KFold(n_splits=6, shuffle=True, random_state=42)
    cv_results=cross_val_score(ridge,X_train,y_train, cv=kf)
    print('alpha=',a, '  scores=',cv_results)
```

```
alpha= 0.1   scores= [0.96578229 0.95659825 0.96103589 0.96633268 0.9656778  0.96702555]
alpha= 1   scores= [0.96575318 0.95666362 0.96103526 0.96658203 0.96576389 0.96704489]
alpha= 10   scores= [0.96551097 0.95689789 0.96098255 0.9667331  0.96588149 0.96686136]
alpha= 100   scores= [0.96491802 0.95708088 0.96079326 0.9666322  0.96558561 0.96624792]
alpha= 1000   scores= [0.96228211 0.95517286 0.95840427 0.96425019 0.96280462 0.96338785]
```

Figure 17-The code and the results for Ridge model

It is observed that alpha=1 has the best results because of the score for the model. By applying alpha=0.1 and using ridge model, it is possible to predict the values for five months of year 2017. Figure 18 illustrates the results for reservation and cancelation status.

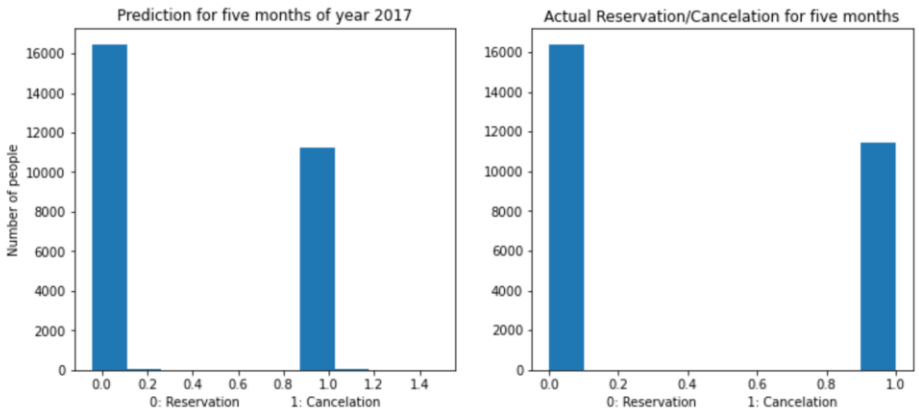

Figure 18-Prediction of five months for year 2017 regarding

status of reservation and cancelation

3-2-Lasso model

For lasso model, analysis of score and precision is needed to find the appropriate number for alpha parameter. This analysis is depicted in the figure 19 based on cross validation function.

```
for a in la:
    lasso=Lasso(alpha=a)
    lasso.fit(X_train, y_train)
    y_pred=lasso.predict(X_test)
    print('alpha=',a,'  score=',lasso.score(X_test, y_test))
```

```
alpha= 0.1   score= 0.7474767780039104
alpha= 1   score= 0.10296584800567343
alpha= 10   score= 0.06544743175426249
alpha= 100   score= -4.278334781426807e-05
alpha= 1000   score= -4.278334781426807e-05
```

```
for a in la:
    lasso=Lasso(alpha=a)
    kf=KFold(n_splits=6, shuffle=True, random_state=42)
    cv_results=cross_val_score(lasso,X_train,y_train, cv=kf)
    print('alpha=',a, '  scores=',cv_results)
```

```
alpha= 0.1   scores= [0.75069175 0.74179153 0.74472731 0.74821366 0.74670795 0.74819979]
alpha= 1   scores= [0.1158522  0.10493972 0.10644843 0.10607908 0.10681817 0.10629032]
alpha= 10   scores= [0.07052697 0.06749594 0.06807381 0.06777784 0.06700719 0.06797642]
alpha= 100   scores= [-4.41920043e-06 -2.38354149e-07 -1.20382900e-05 -5.96344597e-05
 -3.28684661e-04 -7.51451314e-05]
alpha= 1000   scores= [-4.41920043e-06 -2.38354149e-07 -1.20382900e-05 -5.96344597e-05
 -3.28684661e-04 -7.51451314e-05]
```

Figure 19-The code and the results for Lasso model

According to this assessment, alpha =0.1 has the best case to make prediction. So, this number is applied for modelling. Figure 20 depicts the results based on Lasso model.

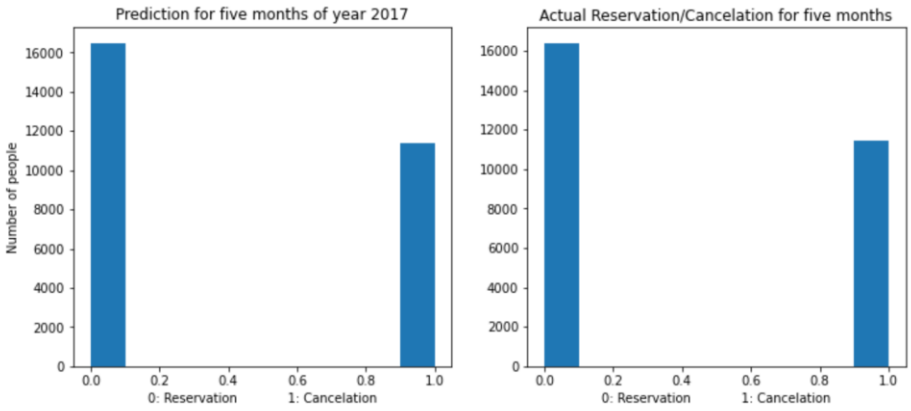

Figure 20-Prediction of five months for year 2017 regarding

status of reservation and cancelation based on Lasso model

3-3-GridSearchCV function

This is function which has hyperparameters. By performing evaluation for the model, the best parameters for the function can be determined. Figure 21 shows you the code and its results.

```
kf=KFold(n_splits=5, shuffle=True, random_state=42)
params={"alpha": np.arange(0.0001,1,10), "solver": ["sage", "lsqr"]}
ridge=Ridge(alpha=0.1)
ridge_cv=GridSearchCV(ridge, param_grid=params, cv=kf)
ridge_cv.fit(X_train, y_train)
print('Best parameters= ',ridge_cv.best_params_)
print('Best score= ',ridge_cv.best_score_)
```

```
Best parameters=  {'alpha': 0.0001, 'solver': 'lsqr'}
Best score=  0.9584146205105636
```

Figure 21-The code and the results for Lasso model

Based on the results, the best number for alpha is 0.0001 and the optimum solver is 'lsqr'. Now by applying these parameters, the modelling of GridSearchCV can start. Figure 22 shows you the results based on this method.

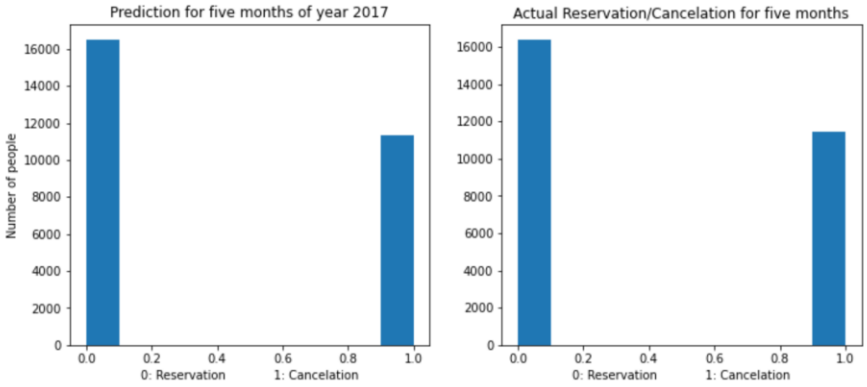

Figure 22-Status of reservation and cancelation based on GridSearchCV model

3-4-RandomForestClassifier function

This function has hyperparameters as well. But it needs parameter adjustment to having optimum case. Figure 23 illustrates the code and function assessment to select the best parameters.

```
for i in range(1,315,15):
    clf=RandomForestClassifier(n_estimators=i, max_depth=2)
    clf.fit(X_train, y_train)
    y_pred=clf.predict(X_test)
    print('n_estimators=', i)
    print(classification_report(y_test, y_pred))
```

```
n_estimators= 1
              precision    recall  f1-score   support

           0       0.75      0.83      0.79     17448
           1       0.63      0.50      0.56      9870

    accuracy                           0.71     27318
   macro avg       0.69      0.67      0.67     27318
weighted avg       0.70      0.71      0.70     27318

n_estimators= 16
              precision    recall  f1-score   support

           0       0.99      1.00      0.99     17448
           1       1.00      0.98      0.99      9870

    accuracy                           0.99     27318
   macro avg       0.99      0.99      0.99     27318
weighted avg       0.99      0.99      0.99     27318
```

Figure 23-The RandomForestClassifier model and its evaluation

If you look at figure 23, it is observed that estimators=16 has optimum case. So is number is used for modelling. Figure 24 shows you the results.

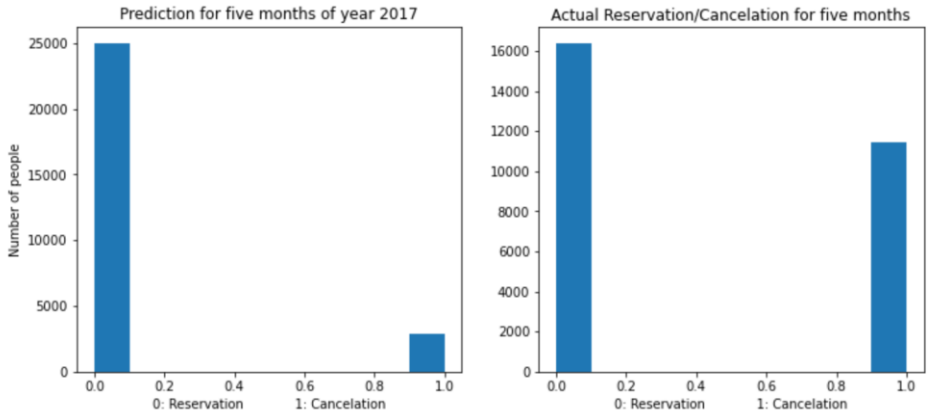

Figure 24-Results from RandomForestClassifier model

It is necessary to note that although the RandomForestClassifier has hyperparameters and it is supposed to have good results, on the other hand; it has error and it is different from the actual values. So, it is not an appropriate model for this data frame.

3-5-AdaBoostClassifier model

It includes hyperparameters similar to GridSearchCV. Figure 25 depicts model definition and precision assessment to find the best parameters.

```
for i in range (25,200,25):
    print('n_estimators=',i)
    clf=AdaBoostClassifier(n_estimators=i)
    clf.fit(X_train, y_train)
    y_pred=clf.predict(X_test)
    print(classification_report(y_test, y_pred))
```

```
n_estimators= 25
              precision    recall  f1-score   support

           0       0.99      1.00      0.99     17448
           1       1.00      0.98      0.99      9870

    accuracy                           0.99     27318
   macro avg       0.99      0.99      0.99     27318
weighted avg       0.99      0.99      0.99     27318

n_estimators= 50
              precision    recall  f1-score   support

           0       0.99      1.00      0.99     17448
           1       1.00      0.98      0.99      9870

    accuracy                           0.99     27318
   macro avg       0.99      0.99      0.99     27318
weighted avg       0.99      0.99      0.99     27318

n_estimators= 75
              precision    recall  f1-score   support

           0       0.99      1.00      0.99     17448
           1       1.00      0.98      0.99      9870

    accuracy                           0.99     27318
   macro avg       0.99      0.99      0.99     27318
weighted avg       0.99      0.99      0.99     27318
```

Figure 25-The code for AdaBoostClassifier model and its evaluation

It is concluded that the n_estimatiors=25 has the best case. So, the results from prediction are shown in the figure 26.

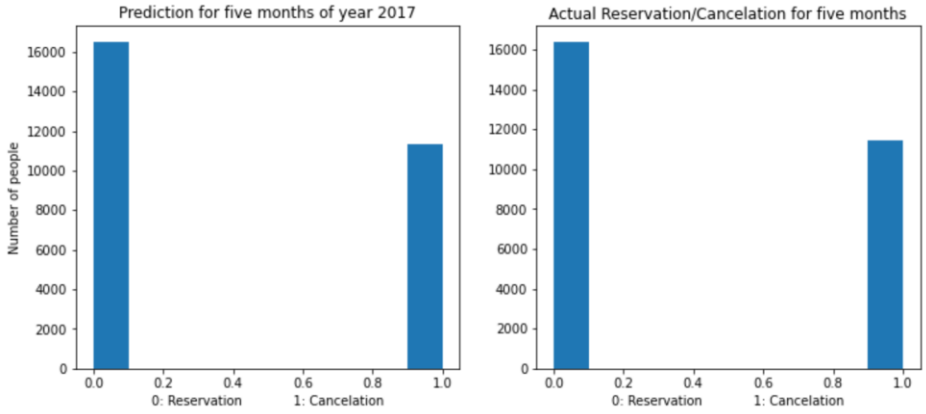

Figure 26-Prediction based on AdaBoostClassifier model

Conclusion

In this study, a complete analysis for different features of a data frame about hotel booking. Data cleaning, exploratory data analysis, model definition and machine learning have been performed for this project. It is observed that the best months for hotel booking are June and July. So, the hotel management can focus on these months to have better plans and more efficient hospitality. In addition, the maximum difference between reservation and cancelation takes place in January 2016. Moreover, the most popular room types are A, D and E according to this study.

Based on the model definitions and assessment, it is observed that Ridge and Lasso function are good models for prediction and Ridge is better than Lasso for this analysis because of the predicted values. In addition, GridsearchCV includes hyperparameters and it is a good model for machine learning in this study and its results are entirely acceptable. Furthermore, RandomForestClassifier function was the next model for making prediction, although it has hyperparameters in machine learning, but it is not appropriate method for this dataset. Because the predicted values are very different from actual numbers for five months of year 2017. In addition, AdaBoostClassifier model is the next function in this project. The predicted results and actual values in the data frame have perfect agreement according to the analysis.

For this dataset related to hotel booking, Ridge, GridSearchCV and AdaboostClassifier functions are so good models for prediction of reservation and cancelation status.

Recommendations for new improvements

The dependent variable of this analysis was ‘is_canceled’. In fact, prediction of five months of year 2017 has been performed for this column of the dataset. It is recommended to apply this approach for column of ‘lead_time’ related five months of year 2017. Or it is possible to predict

the number of adults of the clients for five months of year 2017. It is also suggested that for prediction of ‘lead_time’ and ‘adults’ at the same time and find the correlation between lead time and number adults for five months of this year.