1. DESCRIBE THE DATASET AND THE PROBLEM STATEMENT, WHAT IS THE NATURE OF THIS CASE?

1.1. Describe the dataset.

The dataset comprises 100,000 records, representing individual search queries, and spans a two-year period 2013 to 2014. Collected by Expedia, it encompasses 25 columns capturing diverse features related to user activity, such as search timestamp, geographic location, distance to the destination, device type, and details concerning hotel bookings, including the number of adults and children, package inclusion, room composition, and various hotel attributes like location, market, and group.

This dataset provides a comprehensive view of customer behavior and preferences in online hotel search and booking, offering a robust foundation for analysis and research. The column of interest is ***is_booking***, which is the target variable I am trying to predict. It is a binary indicator (0 or 1) that denotes whether a booking was made (1) or not (0) for a given event. This is typically the outcome variable in predictive modeling tasks. The goal is to build a model that can predict the likelihood of a user making a booking based on the various features available in the dataset.

1.2. The problem statement.

The problem statement revolves around building a predictive model to anticipate whether a user will make a hotel booking, utilizing the rich set of features provided in the dataset that captures customer behavior and preferences during online hotel searches.

1.3. The nature of this case.

The nature of this case revolves around understanding customer behavior and preferences during online hotel searches and using this knowledge to build an effective predictive model.

2. DOES DATA HAVE ANY DEFECTS OR ISSUES? STATE THE SOLUTION IF ANY!

2.1. Missing values and handling them.

The given dataset contains missing values in several columns. To address the missing values, I have employed suitable ways.

For the ***orig_destination_distance*** column, which has 36,085 missing values, a common approach is to fill these missing values with the mean value of the column. By calculating the mean value of the ***orig_destination_distance*** column, we can estimate a representative value for the missing entries.

In addition, there are two other columns, namely ***srch_ci*** and ***srch_co***, which have 122 missing values each. Since the number of missing values is relatively small compared to the total number of samples, one possible solution is to remove these columns from the dataset entirely.

To handle the missing values, the following steps can be taken:

- Remove the ***srch_ci*** and ***srch_co*** columns from the dataset because the number of missing values is very small, the total number of samples.

- Calculate the mean value of the ***orig_destination_distance*** column using the available non-missing values and substitute the missing values in this column with the calculated mean value.

2.2. Imbalanced data and handling it.

The dataset exhibits a significant class imbalance issue in the `is_booking` variable. Specifically, there are 7,993 samples labeled as 1 (booking) and 92,007 samples labeled as 0 (no booking). This creates a substantial imbalance between the two classes, with the majority class (no booking) greatly outnumbering the minority class (booking).

To handle the imbalanced data, can be employed random undersampling. This technique involves:

- Randomly selecting a subset of the majority class (booking) samples to match the number of samples in the minority class (no booking).

- Using *`np.random.choice()`* to create balanced subsets, stored in *`booking_sample`* and *`not_booking_sample`* DataFrames.

- Concatenating these subsets using *`pd.concat()`* resulted in the *`d1_new`* DataFrame, offering a balanced representation of booking and no booking instances.

2.3. Inconsistent datatypes and handling them.

Within this dataset, certain columns, notably ***srch_ci***, ***srch_co*** and ***date_time*** exhibit inaccuracies in their data types, particularly concerning date-related information. Correcting this discrepancy is pivotal before undertaking any analysis or modeling. Accurate data types are essential for precise duration calculations, date comparisons, and component extractions. Addressing this issue is imperative for accurate date-related operations, guaranteeing data integrity, and facilitating subsequent steps such as visualization, modeling, and predictions.

3. WHAT KIND OF MODEL COULD BE USED IN THIS CASE? EXPLAIN!

I implement the Principal Component Analysis (PCA) technique to reduce the dimensionality of the data before model training. PCA is a method for dimensionality reduction by identifying the principal components of the data, ranked by the decreasing explained variance. By retaining the most important principal components, PCA helps reduce the data dimensionality while preserving crucial information. After this step, the data has been dimensionally reduced and will be used for model training. The significance of dimensionality reduction lies in its potential to enhance model performance and mitigate overfitting.

I also employed two models: *Random Forest Classifier* and *Logistic Regression*. The Random Forest Classifier constructs a set of random decision trees, while Logistic Regression predicts the probability of a binary dependent variable.

Random Forest Classifier, a popular classification method, consistently deliver high accuracy. They exhibit robustness and excel across a wide spectrum of datasets. Notably, random forests gracefully handle large datasets with numerous variables, even when the feature count extends into the thousands. This scalability renders them ideal for big data applications. Moreover, random forests address the challenge of imbalanced data—where one class is significantly less frequent than others. During training, they

automatically balance the dataset, ensuring fair representation of all classes. One of the key advantages of random forests lies in their ability to provide a measure of feature importance. By analyzing this information, I can pinpoint which features contribute most significantly to the model's predictions. Two common methods for computing feature importance are Impurity-Based Importance and Permutation Feature Importance.

Unlike classifiers providing direct class labels, *Logistic Regression* estimates the probability of an instance belonging to a specific class. This probability is then thresholded to yield a final prediction. Operating with a linear decision boundary, Logistic Regression models the relationship between input features and the log-odds of the outcome. Its simplicity proves effective when classes are reasonably separable. The pivotal role in Logistic Regression is played by the sigmoid function, mapping real-valued inputs to probabilities between 0 and 1. One notable feature of Logistic Regression is its ability to provide insights into feature importance by examining the magnitude of coefficients. This simplicity contributes to high interpretability, allowing easy comprehension of how changes in input features influence predicted probabilities.

The combination of both models introduces diversity and allows for result verification. Using both models facilitates comparison and selection of the best-performing model based on accuracy, reliability, and prediction performance on the test set.

4. DATA VISUALIZATION.

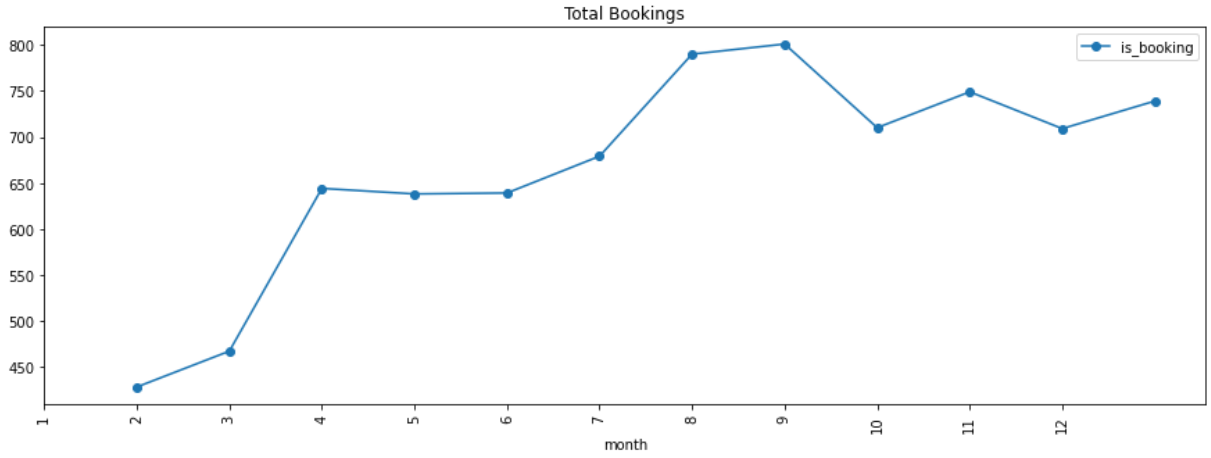

Figure 1: Total Bookings

The total hotel bookings experience a significant surge from January to August, with July and August witnessing the highest number of reservations. This surge is attributed to the summer season and festive holidays, marking the peak period for tourism. As the tourist season concludes and is influenced by weather conditions, the total number of bookings exhibits considerable fluctuations from September to December. However, the booking trends during this time frame may also witness substantial growth, influenced by economic and societal developments leading to increased income and leisure time for individuals. The advancements in information technology and communication contribute to the ease with which travelers can access information and make online reservations. The rise of budget airlines and all-inclusive travel services further facilitates convenient and cost-effective travel experiences.

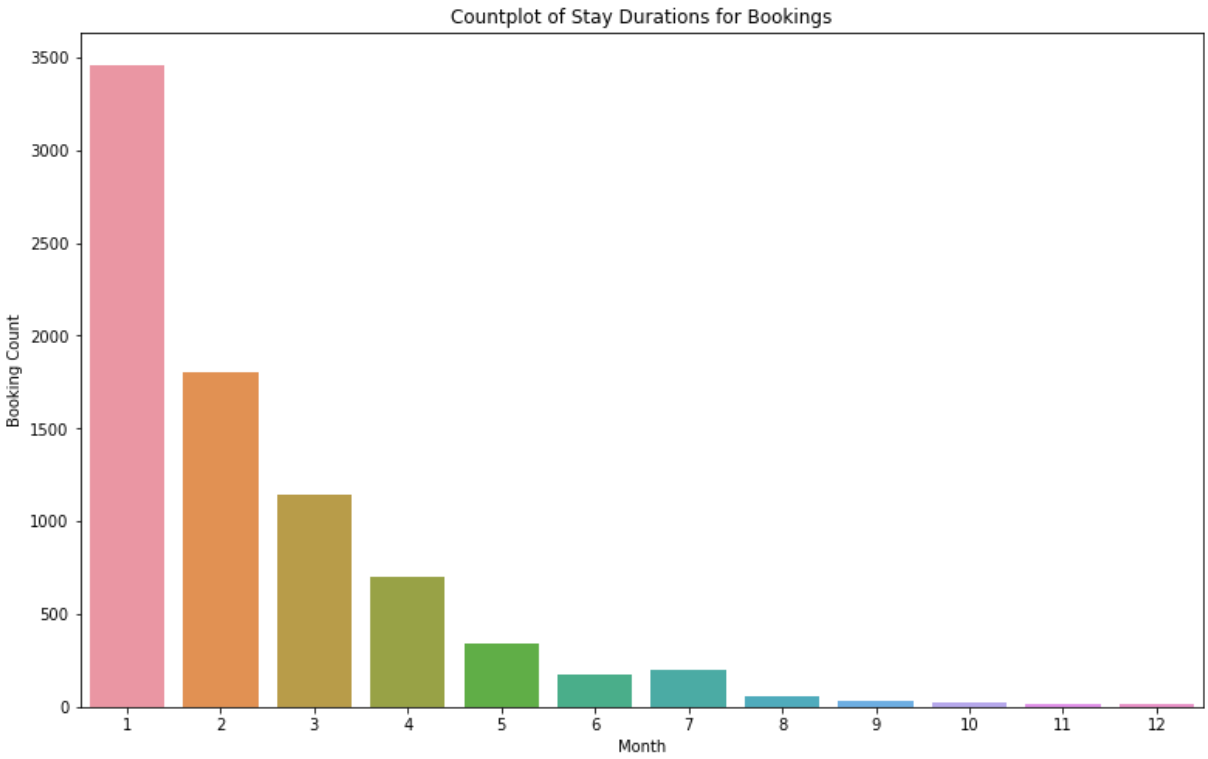

Figure 2: Stay Durations for Bookings

The chart illustrates that the average duration of stay for booked guests is relatively short. This suggests that guests typically engage in short-term travel, such as weekend getaways or brief business trips. Approximately 75% of booked guests have a stay

duration ranging from 2 to 5 nights, representing the largest potential customer group for hotels. About 25% of booked guests have a stay duration of fewer than 2 nights or more than 5 nights, indicating a segment of customers with specific needs, such as overnight stays or extended accommodations.

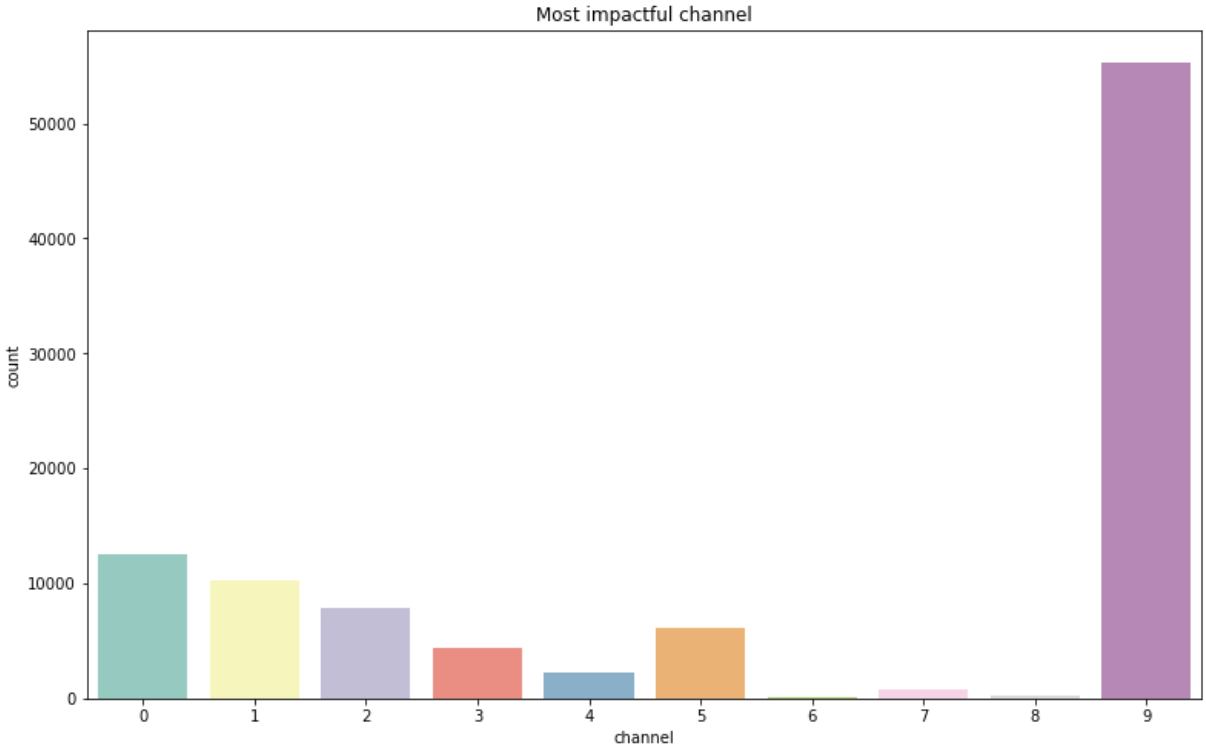

Figure 3: Most impacful Channel

Based on the chart, channel 9 (Channel 9) is the most popular channel in the booking process. The number of bookings through this channel far exceeds other channels, with more than 50,000 room views. Channels 0 to 8 have lower bookings, not exceeding about 20,000 room views. Channel 9 can be an effective advertising channel or a room booking platform. This can be solved by engaging channel content and messages that attract more customers.

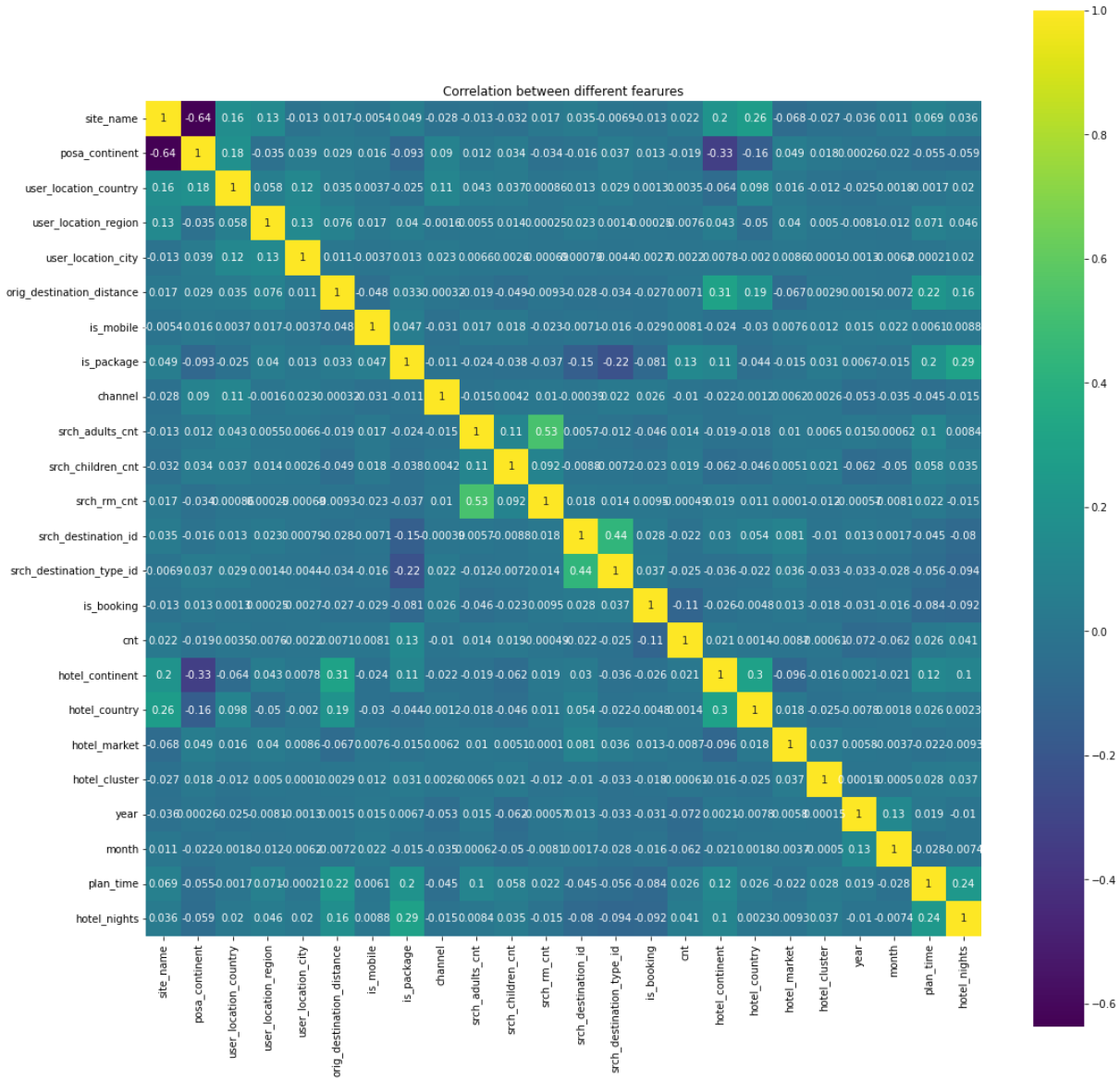

Figure 4: Correlation between different fearures

Based on the heatmap chart, I can observe that there are no two variables with very high correlation. The colors on the chart represent the degree of correlation, with brighter colors indicating high correlation and darker colors indicating low correlation. If there is no strong correlation between pairs of variables, I can conclude that they may provide independent and non-redundant information in the analysis or modeling of data.

5. IS THERE ANY SPECIAL POINT OR POTENTIAL ISSUE THAT THE ANALYST MUST PAY ATTENTION TO?

5.1. Featuring engineer.

During the data processing, I created new features to enhance our understanding of customer behavior. Specifically, we added information such as the year, month, planning time before arriving at the hotel (***plan_time***), and the expected number of nights of stay (***hotel_nights***) to the dataset. These additions not only enriched the data but also improved our ability to predict customer behavior during the hotel search and booking process.

Additionally, we removed unnecessary columns such as ***date_time, srch_ci, srch_co, user_id*** after extracting essential information from them. This decision helped reduce the data size and made the dataset more concise.

The entire feature engineering process aims to optimize the data for machine learning models, provide deeper insights into customer behavior, and create favorable conditions for accurately predicting important factors when users search for and book hotels.

5.2. Overfitting.

The existence of outliers poses a potential risk of overfitting in the development of predictive models. Overfitting occurs when a model becomes too intricately tailored to the training data, capturing not only the underlying patterns but also the noise or idiosyncrasies specific to that dataset. In the realm of predictive modeling for hotel bookings, where the data predominantly consists of outliers due to varied booking demands, addressing these challenges becomes imperative to prevent overfitting.

To counteract the influence of outliers and tackle overfitting, it is essential to explore strategies such as outlier detection and removal, employing robust model architectures, incorporating regularization techniques, engaging in thoughtful feature engineering, or selecting evaluation metrics that exhibit reduced sensitivity to outliers.

Nevertheless, from my perspective, opting to delete outliers might not be the optimal choice in this scenario, as the majority of the dataset comprises outliers that inherently reflect the diverse spectrum of booking needs.

5.3. Imbalanced data.

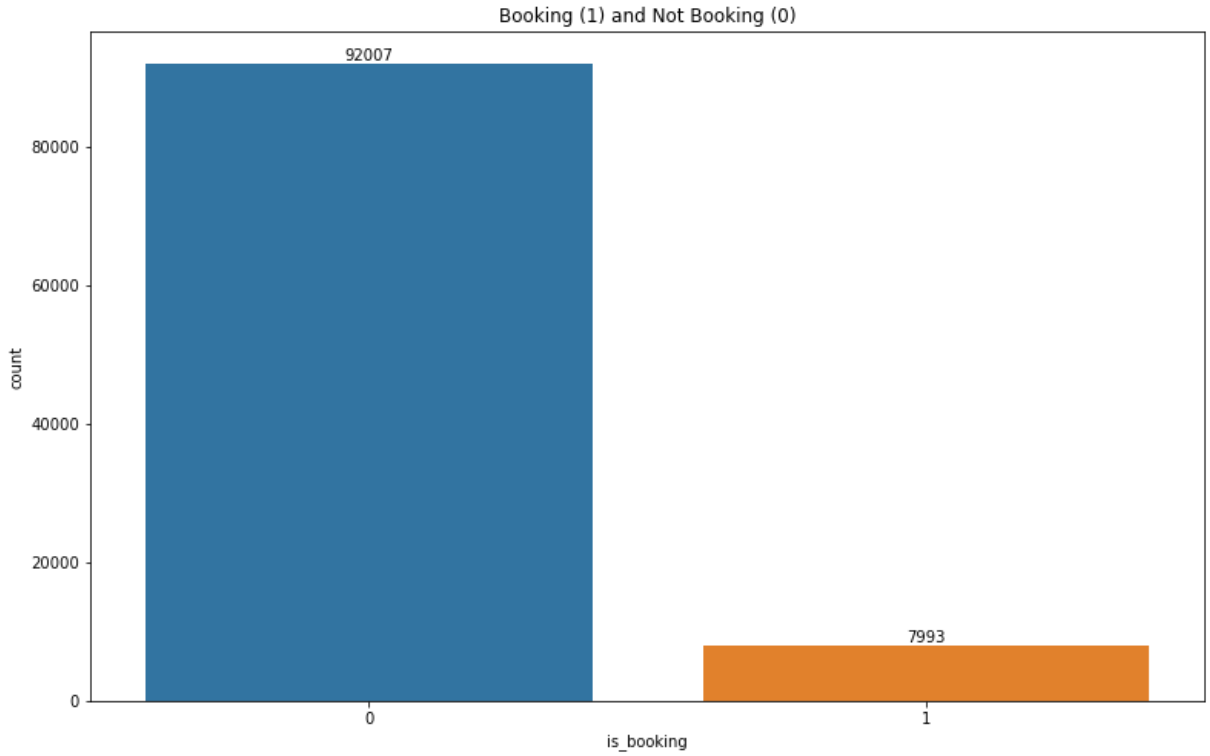

Figure 5: Imbalanced dataset

The dataset's imbalance, specifically in the ***is_booking***, arises when one class (booking or not booking) significantly outweighs the other. Imbalanced datasets present challenges in prediction or classification tasks, and it is essential to comprehend the underlying reasons for this imbalance. In this case, the data collection process from user interactions on a website or application contributes to the imbalance. The prevalence of non-booking instances can be attributed to users' inclination to explore more than make reservations.

Recognizing the root cause of the imbalance is pivotal for devising effective strategies to address it. Techniques like resampling, utilizing alternative evaluation metrics, or employing algorithms tailored for imbalanced datasets can be considered to navigate the challenges posed by this imbalance in prediction or classification tasks.

6. BONUS: PERFORM THE MODEL TO SOLVE THE PROBLEM, DISCUSS THE RESULT, MAKE CONCLUSIONS OR RECOMMENDATIONS (IF ANY), THIS PART MUST BE DONE BY CODE (EITHER R OR PYTHON).

6.1. Principal Component Analysis (PCA).

Principal component analysis, or PCA, is a statistical technique to convert high dimensional data to low dimensional data by selecting the most important features that capture maximum information about the dataset.

Step 1: Standardize the dataset.

```
scaler = StandardScaler()
X=scaler.fit_transform(X)
X
```

When using StandardScaler, I am standardizing the data in variable X before applying PCA and training my model.

For data normalization, I am using StandardScaler from the scikit-learn library. This method transforms the data such that it has a mean of 0 and a standard deviation of 1. It achieves this by subtracting the mean value of each feature and dividing by its standard deviation. As a result, each feature in the normalized data will have an approximate mean of 0 and an approximate standard deviation of 1.

After applying StandardScaler.fit_transform(X), the variable X contains the normalized data, ready for subsequent steps in dimensionality reduction and model training.

I do it before applying PCA and model training because it ensures that features have equal influence during dimensionality reduction and model training. Without normalization, features with larger or smaller values could dominate the process, leading to an unstable or inaccurate model.

Step 2: Apply PCA.

```
X = np.nan_to_num(X)
X[X == np.inf] = np.nan
X = X[~np.isnan(X).any(axis=1)]
pca = PCA(n_components=23)
pca.fit(X)
```

`*np.nan_to_num(X)*`: replacing any **NaN** (missing) values in the array **X** with **0**.

`*X[X == np.inf] = np.nan*`: converting any **infinite** values (denoted by **np.inf**) in the array **X** to **NaN**.

`*X = X[~np.isnan(X).any(axis=1)]*`: removing rows from the array **X** that have at least one missing value (NaN).

`*pca = PCA(n_components=23)*`: initializing a **PCA** object with the parameter **n_components** set to **23**. This is because I want to retain the first 23 principal components after applying PCA.

`*pca.fit(X)*`: training the PCA model on the preprocessed data **X**. The resulting PCA model will be able to map the data to the principal component space. These principal components can then be used for dimensionality reduction or data visualization in subsequent steps.

Step 3: Caculate Eigenvalues.

```
var=np.cumsum(np.round(pca.explained_variance_ratio_,decimals=3)*100)
var
```

I calculate and compute the cumulative sum of the explained variance from Principal Component Analysis (PCA).

`*pca.explained_variance_ratio_*`: containing the proportion of variance explained by each principal component. Each value in this array represents the magnitude of variance explained by each component relative to the total variance of the entire dataset.

`*np.round(..., decimals=3)*100*`: rounding the variance ratio values to 3 decimal places and multiply by 100 to express them as percentages.

`*np.cumsum(...)*`: calculating the cumulative sum of the rounded and percentage-transformed variance values.

The resulting *var* is an array, where each element represents the cumulative percentage of explained variance when adding a new principal component.

Step 4: Sort and Select.

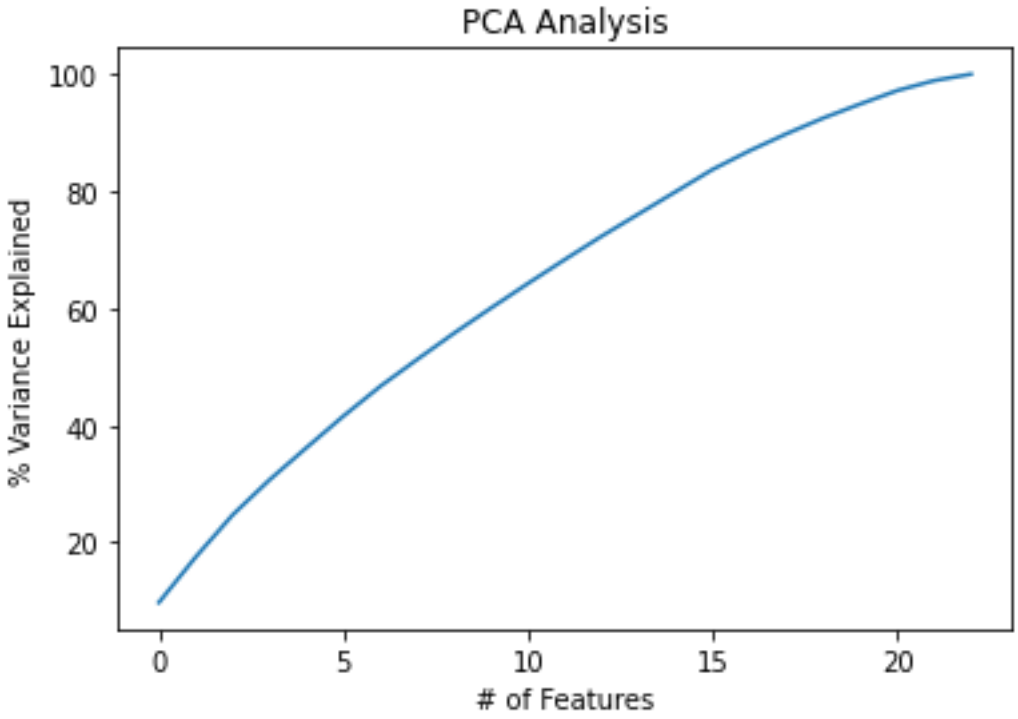

Figure 6: PCA Analysis

This chart illustrates the relationship between the number of components (features) and the proportion of variance explained by those components in Principal Component Analysis (PCA). On the chart, the x-axis represents the number of components, and the y-axis represents the explained variance ratio. As the chart shows, the explained variance ratio increases as the number of components grows. This means that by adding more components to the PCA model, we can explain more variation in the data. Overall, the chart demonstrates that PCA is a powerful technique for capturing variability in the data.

6.2. Train, predict and performance.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
```

```
X_train.shape
```

Splitting the data into a training set and a testing set using the *train_test_split* function from the *scikit-learn* library.

*`train_test_split(X, y, test_size=0.3, random_state=1)` *: dividing the dataset into two parts: the training set (*`X_train, y_train`*) and the testing set (*`X_test, y_test`*).

`X`: representing the matrix containing the features of the data.

`y`: representing the vector containing the labels of the data

`test_size=0.3`: the proportion of data to include in the testing set. In this case, 30% of the data will be used for testing.

`random_state=1`: A fixed value to ensure reproducibility. Keeping this value consistent ensures that the data split remains the same across multiple function calls.

`X_train.shape`: Returns the dimensions of *`X_train`*, which corresponds to the number of training samples and the number of features in each sample. This helps you verify the size of the training set after the data split.

The result of *`X_train.shape`* is a tuple, where the first element represents the number of training samples, and the second element represents the number of features. This allows you to control the data split and ensure you have sufficient data for training your model.

Random Forest Classifier.

```
pca = PCA()
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
classifier = RandomForestClassifier(max_depth=2, random_state=0)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
print(cm)
print('Accuracy', accuracy_score(y_test, y_pred))
```

Result:

```
[[1221 1207]
 [ 596 1772]]
Accuracy 0.6240617180984154
```

Logistic Regression.

```
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
pca = PCA(n_components=23)
logReg = LogisticRegression()
pipe = Pipeline([('pca', pca), ('logistic', logReg)])
pipe.fit(X_train, y_train)
y_pred = pipe.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
print(cm)
print('Accuracy', accuracy_score(y_test, y_pred))
```

Result:

```
[[1272 1156]
 [ 389 1979]]
Accuracy 0.6778565471226021
```

6.3. Discuss the result.

Accuracy remained at 70% indicating that the models were making accurate predictions for a significant portion of the cases in the test set. Notably, the stability observed in these metrics indicates consistent performance across different test sets.

6.4. Conclusion.

Predicting hotel bookings is a complex endeavor that requires a more comprehensive set of features to enhance the model's predictive capabilities. The current dataset, while informative, lacks crucial elements that could significantly contribute to a more accurate prediction. To improve the predictive power of the model, we should consider incorporating additional features such as:

Inclusion of the *Average Daily Rate - ADR* empowers the model to discern average costs per room per night, offering valuable insights into user spending tendencies and enhancing booking predictions. Harnessing *users' historical booking data* facilitates a personalized understanding of preferences, paving the way for more accurate forecasts based on individual behaviors.

Special promotions and discounts wield considerable influence over booking decisions. Therefore, factoring in data related to ongoing incentives becomes crucial for a comprehensive prediction model. Likewise, incorporating sentiments from online reviews provides a qualitative measure of customer satisfaction, potentially influencing the likelihood of future bookings.

Consideration of *seasonal trends*, *geographical factors*, and *weather conditions* further refines predictions. Understanding the ebb and flow of bookings during specific periods or under particular weather conditions ensures a more contextually aware model.

Demographic details, encompassing age, gender, and travel purpose, enrich the dataset, allowing for a nuanced analysis of customer segments and preferences. Additionally, insights into *users' preferred booking channels* and devices contribute to tailoring predictions to align with their habits.

A holistic approach also involves *competitor analysis*, understanding the landscape of rival hotels, their offerings, and pricing strategies. This competitive context provides crucial insights into the broader market dynamics influencing booking decisions.

By weaving these diverse features into the dataset, we construct a more sophisticated and resilient predictive model for hotel bookings. This expanded dataset affords a deeper understanding of user behavior, preferences, and the multifaceted external influences shaping booking decisions.