Hotel Reservation Prediction Report

CSE-5367 Pattern Recognition

Submitted by: Satya Shah(1002161494), Varad Nair(1002161475), Harshita Dhingra(1002052823)

—

Instructor: Prof. Yingying Zhu

Table of Contents

1. Background and Introduction

The hospitality industry faces a significant challenge with booking cancellations, which can lead to revenue loss, operational inefficiencies, and planning difficulties. Understanding and predicting customer cancellation behavior has thus become crucial for optimizing hotel management strategies and enhancing guest satisfaction.

This report presents an end-to-end data science project aimed at predicting hotel booking cancellations using advanced machine learning techniques. By leveraging historical booking data, the project seeks to uncover hidden patterns, key behavioral trends, and critical factors that influence a guest's likelihood to cancel a reservation.

The **primary objectives** are twofold:

- To **extract actionable insights** from the data that can inform business decisions, such as pricing strategies and overbooking policies.
- To **build and deploy predictive models** that accurately forecast cancellations, allowing hotel management to proactively mitigate potential revenue loss.

The project adopts a structured approach, encompassing data exploration, feature engineering, model development, model evaluation, and deployment. Furthermore, the solution is enhanced through the integration of MLOps tools such as MLflow for experiment tracking, Docker for containerization, and Google Cloud Platform for model hosting. Interactive visualizations built with Tableau provide stakeholders with intuitive insights into customer behaviors and booking trends.

Ultimately, this work not only delivers a robust technical solution but also offers tangible business value by enabling hotels to better manage their operations, improve guest experience, and maximize profitability.

2. Project Motivation and Objectives

Exploratory Goals

- Understand guest booking behavior.
- Identify trends associated with cancellations and non-cancellations.

Predictive Goals

- Build classification models to predict cancellations.
- Evaluate multiple models to determine the best-performing algorithm.
- Identify key features that influence booking status.

Business Applications

- Reduce last-minute cancellations.
- Enable targeted marketing strategies.
- Improve guest retention and satisfaction.

3. Dataset Overview

Dataset Link: **[link]**

Dataset Size: 29,020 rows, 18 columns

Features:

- Booking_ID, Number of Adults, Children, Number of Week/Weekend Nights, Meal Plan Type, Car Parking Requirement, Room Type Reserved, Lead Time, Arrival Date Details, Market Segment Type, Repeated Guest Flag, Previous Cancellation Count, Previous Bookings Not Canceled, Average Price Per Room , Special Requests, Booking Status (Target)

.

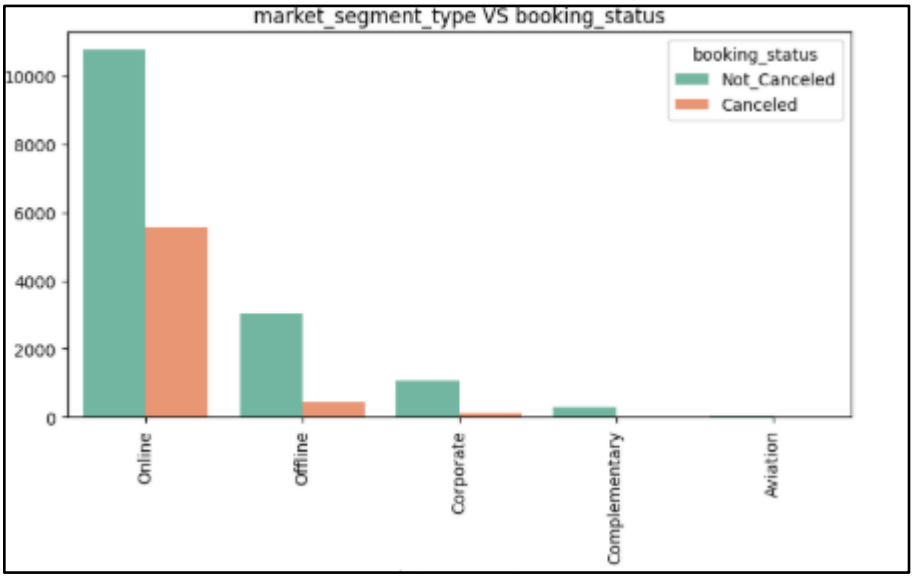
```
In [7]: df.isnull().sum()

Out[7]: no_of_adults                        0
        no_of_children                      0
        no_of_weekend_nights                0
        no_of_week_nights                   0
        type_of_meal_plan                   0
        required_car_parking_space          0
        room_type_reserved                  0
        lead_time                           0
        arrival_year                        0
        arrival_month                       0
        arrival_date                        0
        market_segment_type                 0
        repeated_guest                      0
        no_of_previous_cancellations        0
        no_of_previous_bookings_not_canceled    0
        avg_price_per_room                  0
        no_of_special_requests              0
        booking_status                      0
        dtype: int64
```

4. Data Exploration and Visualization

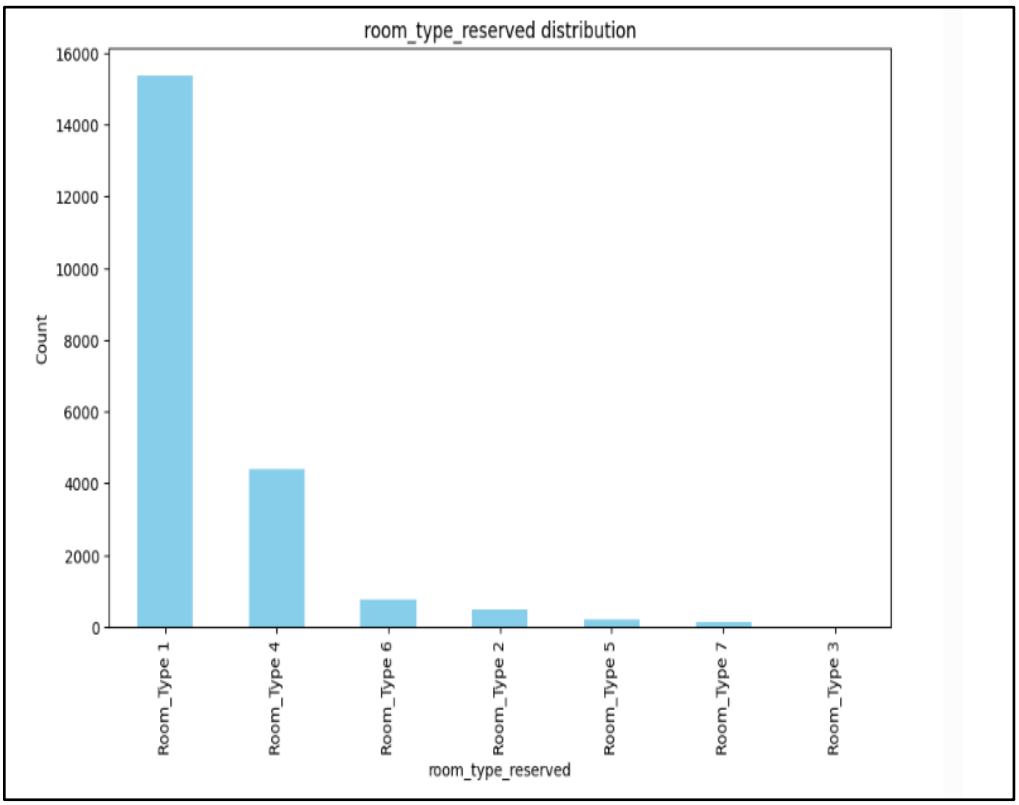
Key Observations:

- Most guests are first-time visitors.

- Online platforms are the dominant booking method.

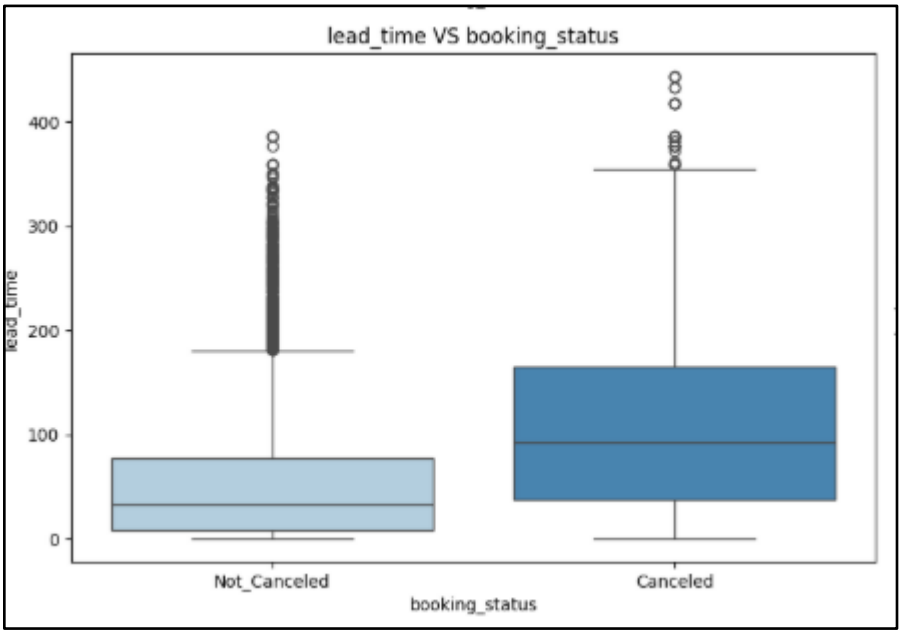


- Room Type 1 and Meal Plan 1 are the most commonly selected.



- If a person is reserving the parking, then there is very little chance of cancellation.

- Lead time is one of the deciding factors in whether the customer will cancel or keep his/her reservations.



- Variance Inflation Factor- There is no multicollinearity in our data as the Variance Inflation Factor of the features is less.

In [38]:
```
vif_data
```

Out[38]:

	feature	VIF
0	const	4.100197e+07
1	no_of_adults	1.341180e+00
2	no_of_children	1.282459e+00
3	no_of_weekend_nights	1.073215e+00
4	no_of_week_nights	1.125260e+00
5	type_of_meal_plan	1.178228e+00
6	required_car_parking_space	1.036105e+00
7	room_type_reserved	1.549859e+00
8	lead_time	1.406287e+00
9	arrival_year	1.288533e+00
10	arrival_month	1.248028e+00
11	arrival_date	1.003605e+00
12	market_segment_type	1.704297e+00
13	repeated_guest	1.765576e+00
14	no_of_previous_cancellations	1.396559e+00
15	no_of_previous_bookings_not_canceled	1.712854e+00
16	avg_price_per_room	1.926372e+00
17	no_of_special_requests	1.267940e+00
18	booking_status	1.389308e+00

- Skewness- Features like no_of_previous_cancellations and no_of_previous_bookings_not_cancelled were right-skewed. We applied a log transform to normalize these columns.

```
In [42]: ###Skewness
         skeweness=df.skew()
         skeweness

Out[42]: no_of_adults                           -0.305652
         no_of_children                          4.165696
         no_of_weekend_nights                    0.636637
         no_of_week_nights                       1.553657
         type_of_meal_plan                       1.650716
         required_car_parking_space              4.538315
         room_type_reserved                      1.392145
         lead_time                               1.405258
         arrival_year                           -1.953273
         arrival_month                          -0.293266
         arrival_date                            0.010333
         market_segment_type                    -2.333046
         repeated_guest                          5.282330
         no_of_previous_cancellations           22.001489
         no_of_previous_bookings_not_canceled   16.735934
         avg_price_per_room                      0.542888
         no_of_special_requests                  0.922373
         booking_status                         -0.942305
         dtype: float64
```

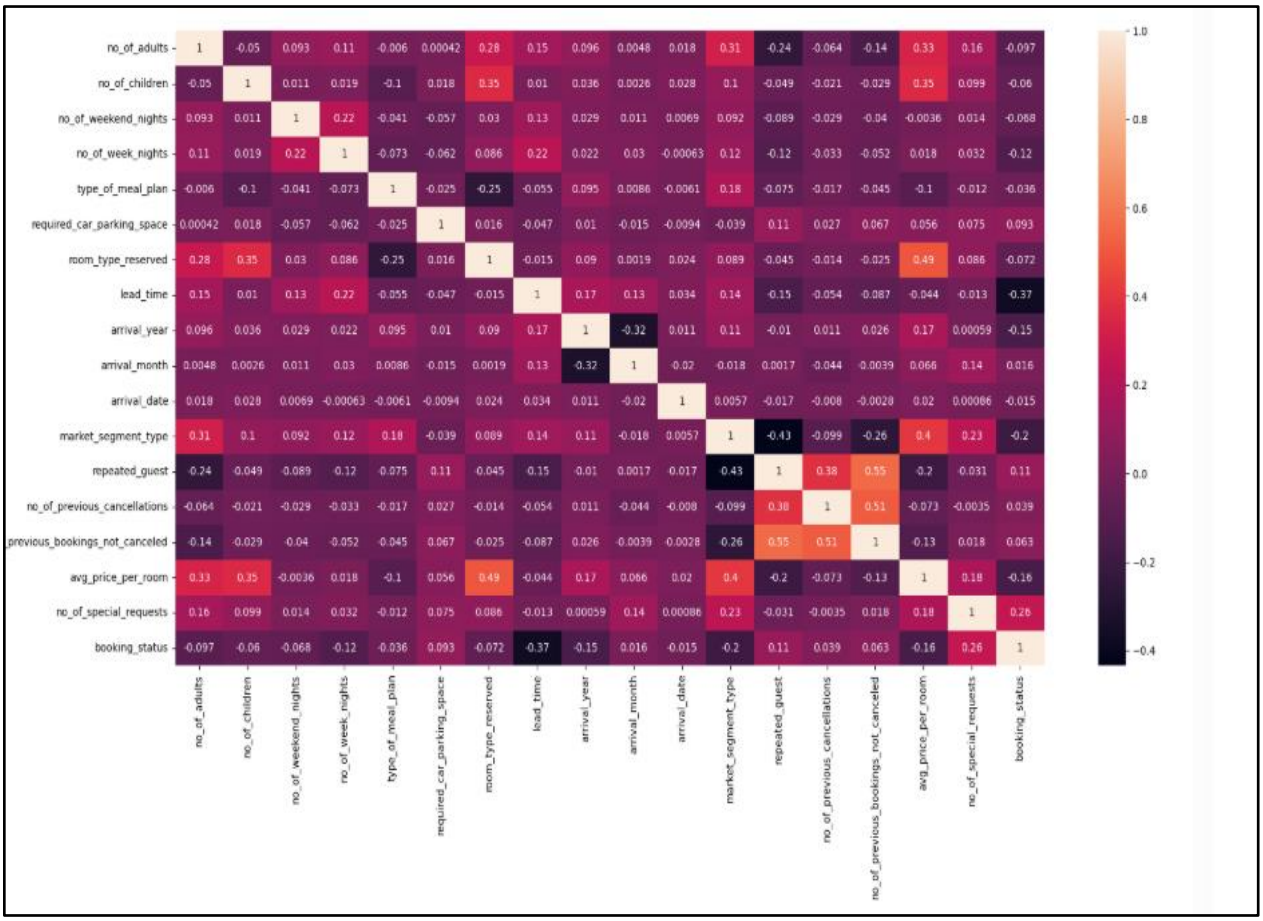
- After applying the log transform, the skewness decreased for the above two features.

```
In [43]: for col in df.columns:
             if skeweness[col]>5:
                 df[col]=np.log1p(df[col])

In [44]: ###Skewness after applying log transform. Skweness is reduced for no_of_previous_cancellat
         skeweness=df.skew()
         skeweness

Out[44]: no_of_adults                           -0.305652
         no_of_children                          4.165696
         no_of_weekend_nights                    0.636637
         no_of_week_nights                       1.553657
         type_of_meal_plan                       1.650716
         required_car_parking_space              4.538315
         room_type_reserved                      1.392145
         lead_time                               1.405258
         arrival_year                           -1.953273
         arrival_month                          -0.293266
         arrival_date                            0.010333
         market_segment_type                    -2.333046
         repeated_guest                          5.282330
         no_of_previous_cancellations           13.270580
         no_of_previous_bookings_not_canceled    7.619850
         avg_price_per_room                      0.542888
         no_of_special_requests                  0.922373
         booking_status                         -0.942305
         dtype: float64
```

5. Feature Correlation Analysis

- **Lead Time** shows a strong positive correlation (0.37) with cancellations
- **Room Type Reserved** shows a negative correlation (-0.37)
- **Special Requests** and **Average Room Price** are moderately correlated

Interpretation: Guests booking far in advance are more likely to cancel. Certain room types are tied to higher cancellation rates.



6. Handling Imbalanced Data with SMOTE

Why SMOTE?

- Models are biased when data is imbalanced.
- SMOTE generates synthetic examples for the minority class (Cancelled bookings).
- Balancing improves model fairness and prediction reliability.

Balanced Dataset Size:

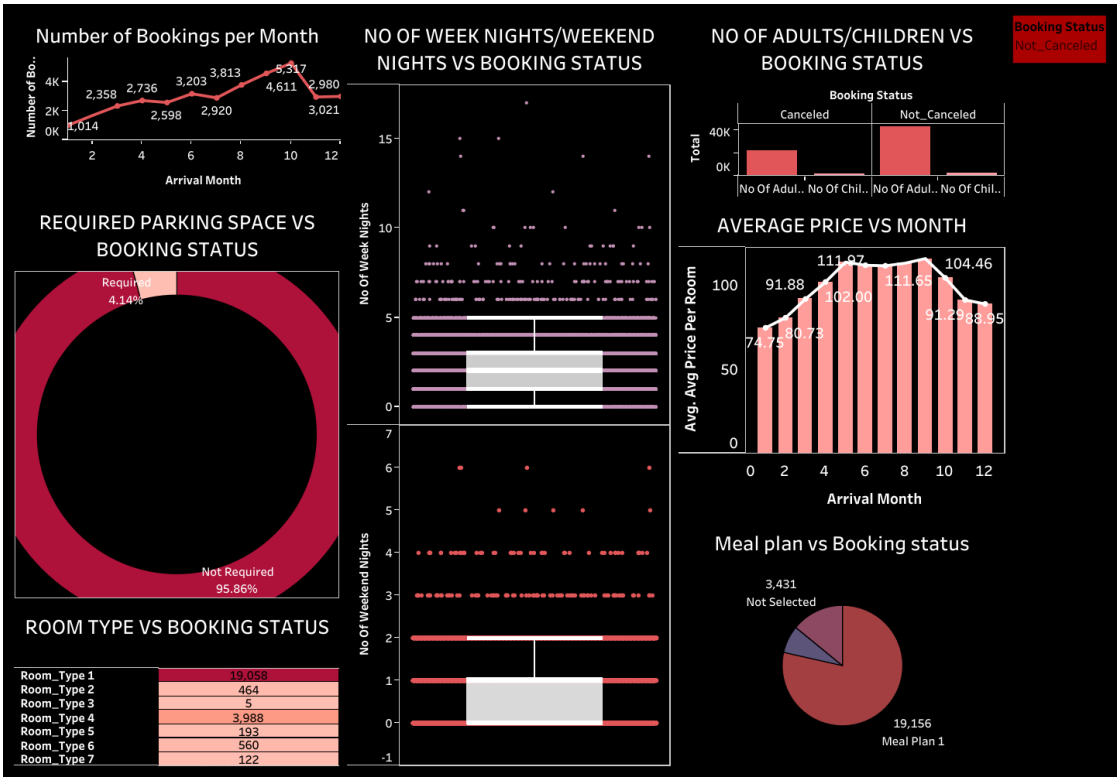
- 30,462 records after applying SMOTE

```
In [49]: y.value_counts()

Out[49]: booking_status
         1    15231
         0     6128
         Name: count, dtype: int64

In [50]: from imblearn.over_sampling import SMOTE
         smote=SMOTE(random_state=42)
         X_res,y_res=smote.fit_resample(X,y)

In [51]: y_res.value_counts()

Out[51]: booking_status
         1    15231
         0    15231
         Name: count, dtype: int64
```

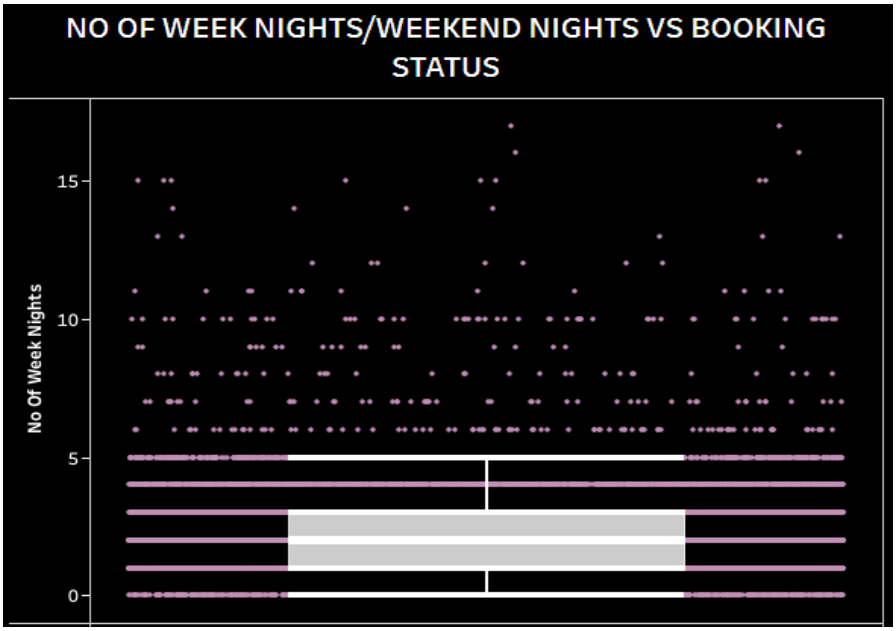
7. Dashboard and Visualization(TABLEAU)

This is the link to Tableau Dashboard [LINK]

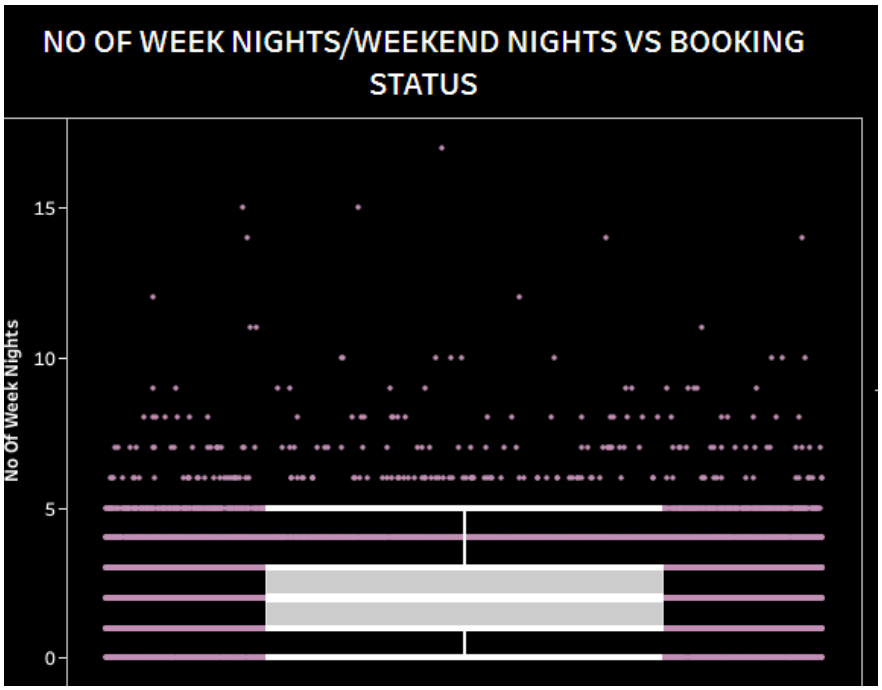
- A Tableau dashboard was created for stakeholders to interactively explore booking trends and cancellation factors.
- The dashboard includes filters for meal plan, room type, cancellation status, and more.



- Observations from the above graph:
    1) For the months from August to October, there is a sudden increase in bookings.
    2) So the hotel owner has kept the prices high for these months.
    3) From the below graph, you can observe that the people who have cancelled their reservations have booked for 10 week nights. There are so many outliers in this.



CANCELLED (ABOVE IMAGE) VS NOT CANCELLED(BELOW IMAGE)



8. Machine Learning Models Used

Models: The following models were used with default parameters, and the following metrics we obtained for these models on test data.

- Random Forest
- Gradient Boosting
- AdaBoost
- Logistic Regression
- Support Vector Machine
- Decision Tree
- K-Nearest Neighbors
- Gaussian Naive Bayes
- CatBoost
- XGBoost
- LightGBM

```
metrics_df=pd.DataFrame(metrics)
metrics_df
```

	Model	Accuracy	Precision	Recall	F1-Score
0	Random Forest	0.891679	0.892508	0.892508	0.892508
1	Gradient Boosting	0.837190	0.821473	0.864821	0.842590
2	AdaBoost	0.804202	0.821294	0.781433	0.800868
3	Logistic Regression	0.774331	0.802787	0.731922	0.765718
4	Support Vector Machine	0.720335	0.729657	0.706840	0.718068
5	Decision Tree	0.840801	0.850000	0.830619	0.840198
6	K-Nearest Neighbors	0.779419	0.856612	0.675244	0.755191
7	Gaussian Naive Bayes	0.773346	0.803232	0.728664	0.764133
8	CatBoost	0.870671	0.861305	0.885993	0.873475
9	XGBoost	0.870343	0.866324	0.878176	0.872210
10	LightGBM	0.866732	0.851494	0.890879	0.870742

9. Model Selection and Hyperparameter Tuning

Approach:

Model selection and hyperparameter tuning were critical steps to maximize the predictive performance and robustness of our system. Initially, we trained multiple machine learning models using their default parameters to establish baseline performances. These models included Random Forest, LightGBM, Gradient Boosting, AdaBoost, Logistic Regression, Support Vector Machine, Decision Tree, K-Nearest Neighbors, Gaussian Naive Bayes, CatBoost, and XGBoost.

To optimize performance further, we employed **RandomizedSearchCV** for hyperparameter tuning. RandomizedSearchCV is preferred over GridSearchCV in this case due to its ability to efficiently search a larger hyperparameter space with fewer computations, making it faster and scalable, especially when dealing with complex models.

Each model was evaluated based on key classification metrics:

- **Accuracy:** Overall correctness of the model.
- **Precision:** Correctness of positive predictions (very important in our case to minimize False Positives — where we incorrectly predict a non-cancellation).
- **Recall:** Ability to detect true cancellations.
- **F1-Score:** Harmonic mean of Precision and Recall for balanced evaluation.

We applied cross-validation during tuning to ensure that the model performance generalized well across different subsets of the data.

Best Performer:

- **Random Forest** demonstrated the highest overall performance on the test dataset in terms of accuracy, precision, recall, and F1-score. It was particularly good at capturing non-linear patterns and feature interactions, which were crucial in a complex booking behavior dataset.
- However, despite Random Forest's excellent predictive power, the model had a **significantly larger size** (larger .pkl file) compared to boosting-based models like LightGBM. A heavier model would result in **slower deployment times** and **higher cloud resource costs** when integrated into a production system.

Final Deployment Choice:

Considering model size, inference speed, and near-comparable accuracy, we selected **LightGBM** for deployment.
LightGBM not only maintained strong predictive performance but also offered advantages such as:

- Faster prediction times.
- Smaller model file size.
- Better scalability for handling larger, future datasets.

The final LightGBM model was also fine-tuned using **RandomizedSearchCV** on key hyperparameters such as:

- num_leaves
- max_depth
- learning_rate
- n_estimators
- min_child_samples
- subsample
- colsample_bytree

This tuning process helped achieve an optimal balance between model complexity and generalization, ensuring strong performance when deployed in a real-world hotel booking system.

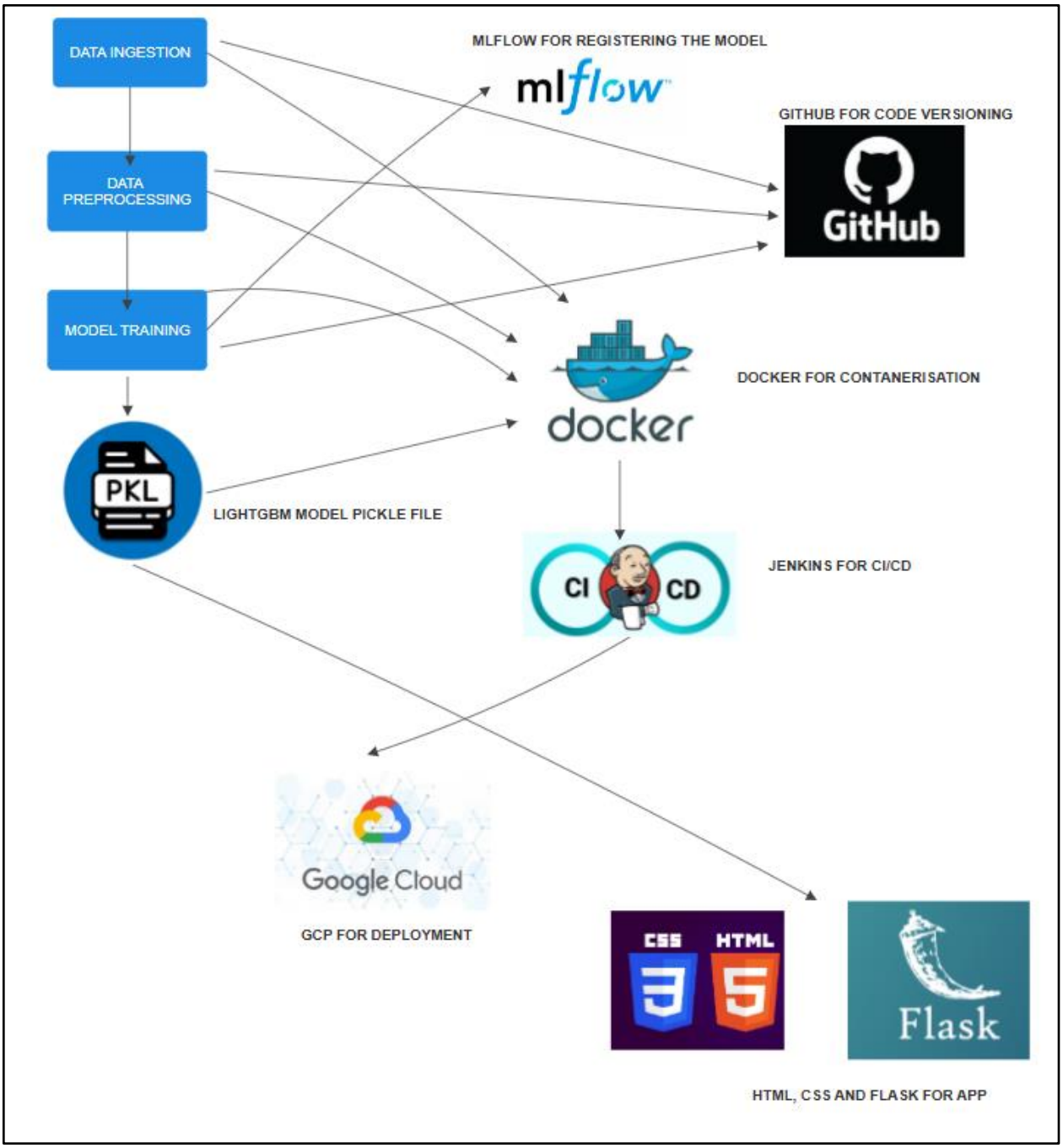
10. Method Used : ML Pipeline Architecture(Modular Coding)

Stages:

- **Data Ingestion:** Load and clean raw data
- **Data Preprocessing:** Encode, normalize, and apply SMOTE
- **Model Training:** Train models, evaluate performance
- **Model Evaluation:** Selecting the best model using cross-validation
- **Model Saving:** Save using joblib for future use.

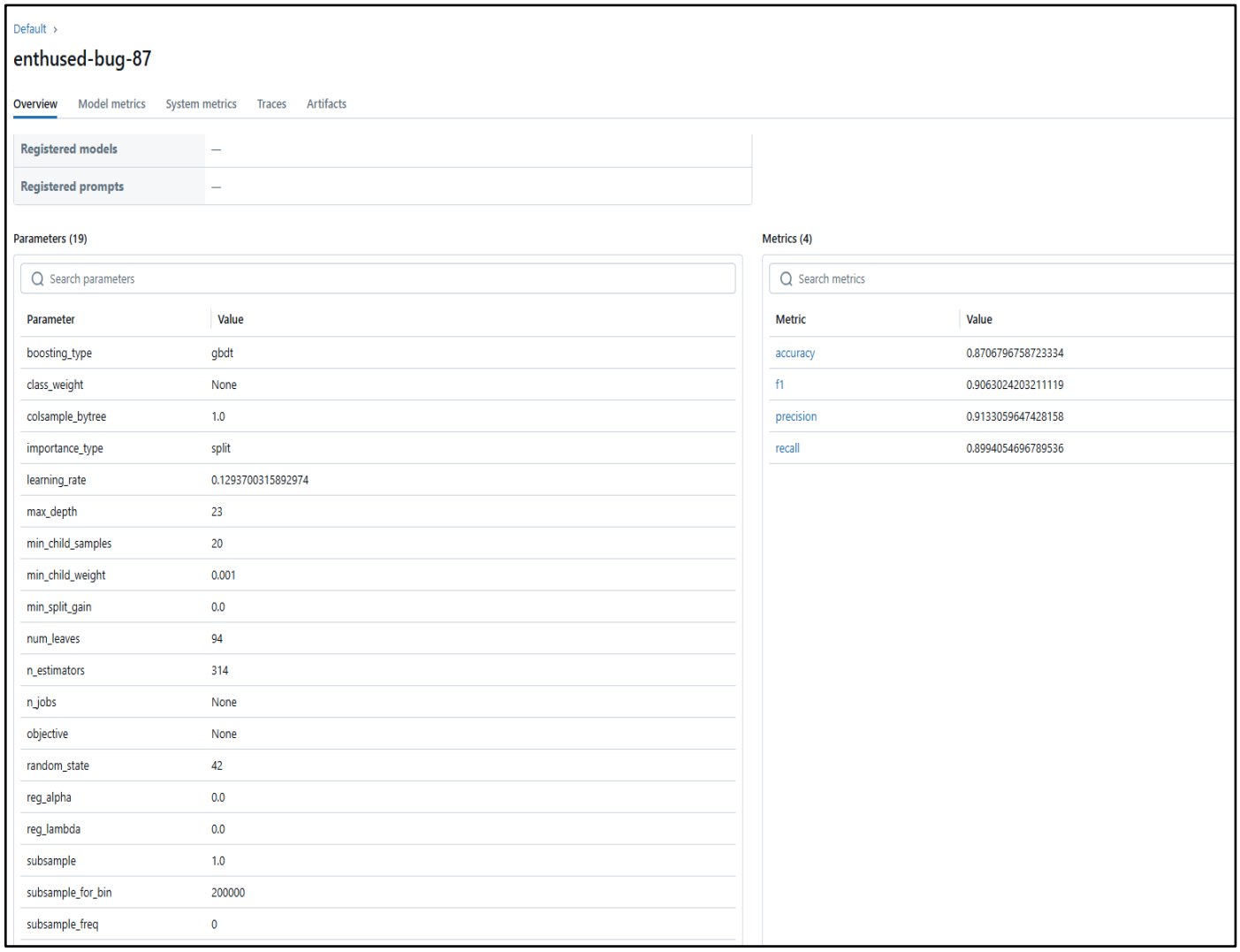
Below is the model architecture:

1) Model- LightGBM
2) MLFLOW for model registering
3) GitHub for code Versioning
4) Docker for contanerisation
5) Jenkins for Continuous Integration/ Continuous Delivery
6) Google Cloud Platform for Deployment
7) HTML, CSS, and Flask for building the web application


DATA INGESTION
DATA PREPROCESSING
MODEL TRAINING
PKL
LIGHTGBM MODEL PICKLE FILE
MLFLOW FOR REGISTERING THE MODEL
mlflow
GITHUB FOR CODE VERSIONING
GitHub
docker
DOCKER FOR CONTANERISATION
CI
CD
JENKINS FOR CI/CD
Google Cloud
GCP FOR DEPLOYMENT
CSS
HTML
Flask
HTML, CSS AND FLASK FOR APP

11. MLflow Integration

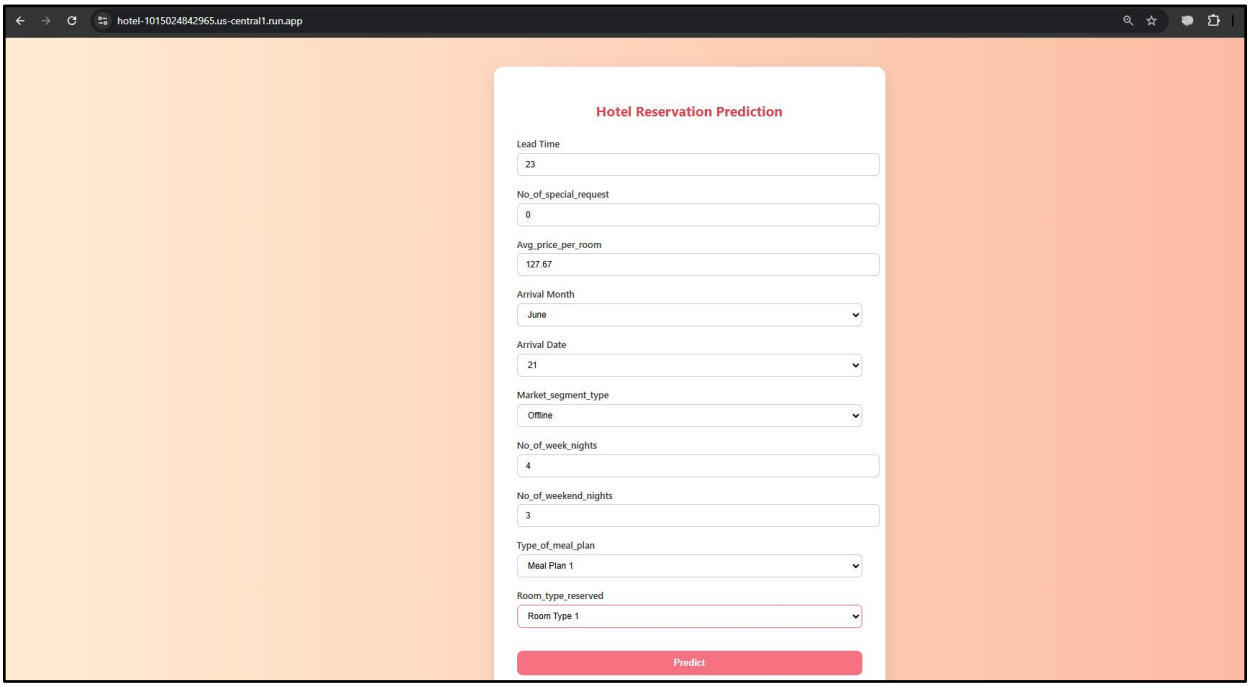
- All experiments were tracked using MLflow.
- **Tracks:**
  - Model versions and metrics
  - Parameters used in training
  - Comparison between runs
- **Benefits:**
  - Reproducibility
  - Version control
  - Lifecycle management



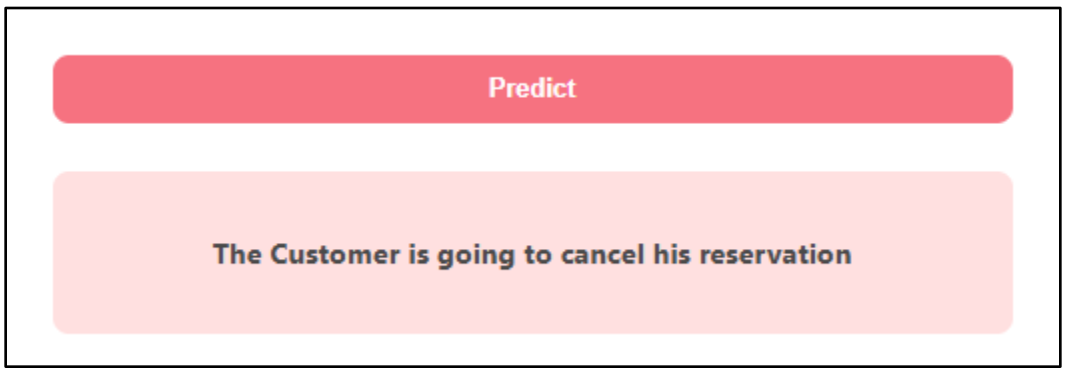
12. Experiments and Results

Random Forest Metrics:

- Precision is more important in this problem statement because we want to decrease False Positives(where our model predicts Not cancelled i.e, 1, but actually the customer cancelled his/her reservation i.e., 0)
- Precision: ~91%
- Accuracy and Recall: Balanced and high
- Feature Importance: Lead Time, Room Type, Special Requests
- High lead time is a strong indicator of potential cancellation.



- When the customer clicks on Predict, they will get the results below



13. Discussion

Business Impact:

The predictive model developed in this project has the potential to significantly improve hotel revenue management and operational planning. By accurately forecasting cancellations, hotels can proactively take measures such as overbooking optimizations, dynamic pricing adjustments, and personalized guest engagement to minimize the negative impact of cancellations.
Moreover, insights derived from customer booking behaviors enable the hotel management team to fine-tune their marketing strategies, enhance customer retention programs, and optimize resource allocation such as staffing, room inventory, and service planning.

Real-time prediction capability, combined with actionable dashboards, empowers decision-makers to shift from reactive to **proactive management**, leading to improved guest satisfaction, reduced financial losses, and better overall business performance.

Recommendations:

- **Monitor Long Lead-Time Bookings:**
  Guests who book far in advance are more likely to cancel. The hotel should flag these bookings and implement follow-up confirmation processes or flexible incentives to retain these customers.

- **Offer Flexible Cancellation Policies:**
  Introducing tiered cancellation policies (e.g., free cancellation within a short window, partial refunds afterward) can encourage early commitment from customers while still offering options that reduce the likelihood of last-minute cancellations.

- **Target High-Risk Bookings for Intervention:**
  Leverage the model's predictions to identify bookings with a high probability of cancellation. These customers can be targeted with personalized reminders, loyalty rewards, or promotions to decrease their likelihood of cancelling.

- **Dynamic Pricing Strategy:**
  Adjust room pricing based on cancellation risks during high-demand periods to optimize revenue while maintaining occupancy.

- **Resource Planning Optimization:**
  Use booking prediction trends to plan staffing and operational resources efficiently, minimizing overhead during periods of expected cancellations.

14. Limitations and Future Work

Limitations:

- **External Events Not Considered:**
  The current model does not incorporate external factors such as holidays, local events, pandemics, or weather conditions, all of which can significantly impact booking and cancellation behavior.
- **Static Historical Dataset:**
  The dataset used was static and historical. As a result, the model may not adapt well to evolving trends or sudden shifts in customer behavior over time without regular retraining.
- **Limited Features Related to Customer Loyalty:**
  Important features like loyalty program membership, past stay reviews, and customer preferences were not available in the current dataset, which could otherwise improve model precision.

Future Work:

- **Integration of Seasonality and Time Series Analysis:**
  Future iterations of the model will incorporate temporal features such as seasonality effects, day-of-week trends, and holiday periods to better capture booking patterns.

- **Incorporation of Loyalty Programs:**
  Adding loyalty membership data can help improve guest profiling and enable better personalized interventions to reduce cancellations.

- **Adoption of NoSQL Databases (MongoDB):**
  To handle high-velocity real-time booking data, MongoDB will be integrated for flexible and scalable data storage.

- **Real-time Streaming Data Pipelines (ETL):**
  Building real-time Extract-Transform-Load (ETL) pipelines will allow continuous ingestion of new booking data, enabling the model to detect and adjust to data drift over time.

- **Collaboration with Hotel Partners:**
  Future work involves integrating the solution into live hotel booking systems to test the model's performance with real-world streaming data and implement active monitoring dashboards for operational use.

- **Model Retraining and Drift Monitoring:**
  Establishing automated retraining schedules and drift detection systems will ensure the model maintains high performance even as customer behaviors evolve.

15. Conclusion

- In conclusion, this project successfully demonstrated the end-to-end development of a predictive system for hotel booking cancellations. We explored and analyzed guest booking patterns, identified critical factors contributing to cancellations, and built robust machine learning models to predict cancellations with high precision.

- Using LightGBM for deployment, combined with a complete ML pipeline including MLflow tracking, Docker containerization, Jenkins for CI/CD, and Google Cloud Platform for hosting, allowed us to showcase real-world MLOps practices. The integration of a user-friendly web interface using Flask further made our solution accessible and practical for business users.

- Our Tableau dashboard provided actionable visual insights, helping stakeholders better understand booking behaviors, seasonality impacts, and customer preferences.

- From a business perspective, the system offers hotels the ability to proactively manage high-risk bookings, optimize revenue, and enhance guest satisfaction by targeting marketing efforts and refining cancellation policies.

- While the project made significant progress, future extensions such as incorporating time-series trends, loyalty program data, external events, and real-time ETL pipelines can further improve model accuracy and adaptability. Overall, the project demonstrated a strong combination of data science, machine learning, and engineering skills to deliver a tangible solution for the hospitality industry.

16. References

-**Exploratory Data Analysis in Python**
GeeksforGeeks. (n.d.). *Exploratory data analysis in Python*. GeeksforGeeks. https://www.geeksforgeeks.org/exploratory-data-analysis-in-python/

-Hyperparameter Tuning Using Randomized Search
Analytics Vidhya. (2022, November 14). *Hyperparameter tuning using randomized search*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/11/hyperparameter-tuning-using-randomized-search/

-Automate Jenkins Setup with Docker and Jenkins Configuration as Code
DigitalOcean. (2022, September 21). *How to automate Jenkins setup with Docker and Jenkins configuration as code*. DigitalOcean. https://www.digitalocean.com/community/tutorials/how-to-automate-jenkins-setup-with-docker-and-jenkins-configuration-as-code

-How to Use CSS in Python Flask
GeeksforGeeks. (n.d.). *How to use CSS in Python Flask*. GeeksforGeeks. https://www.geeksforgeeks.org/how-to-use-css-in-python-flask/

-LightGBM – Light Gradient Boosting Machine
GeeksforGeeks. (n.d.). *LightGBM – Light Gradient Boosting Machine*. GeeksforGeeks. https://www.geeksforgeeks.org/lightgbm-light-gradient-boosting-machine/