HOTEL RESERVATION CANCELATION PREDICTION

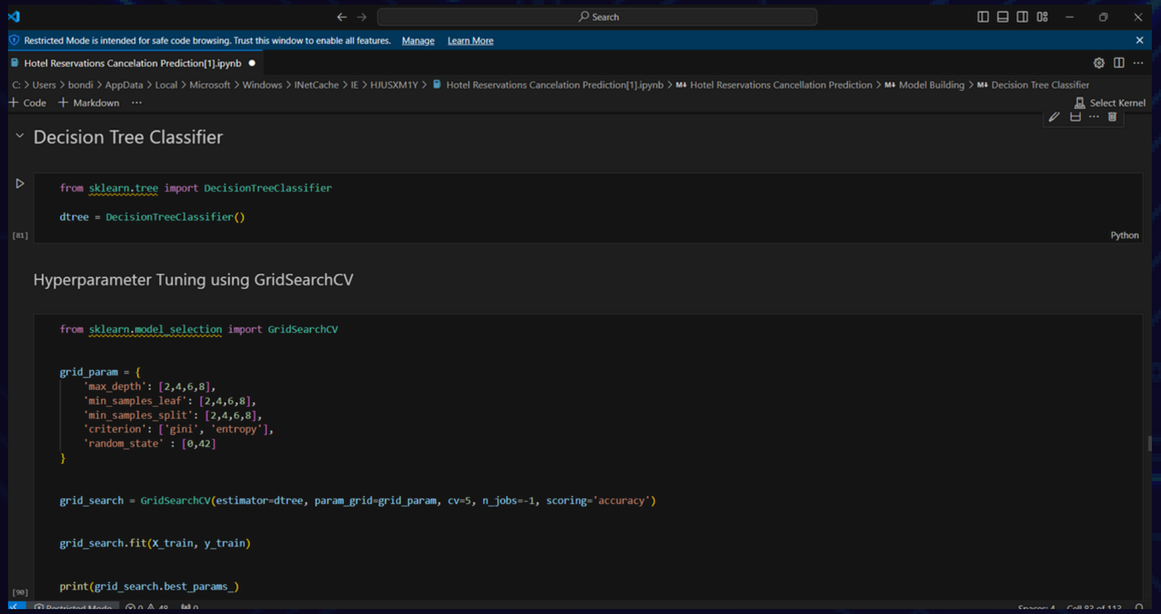
MODEL BUILDING

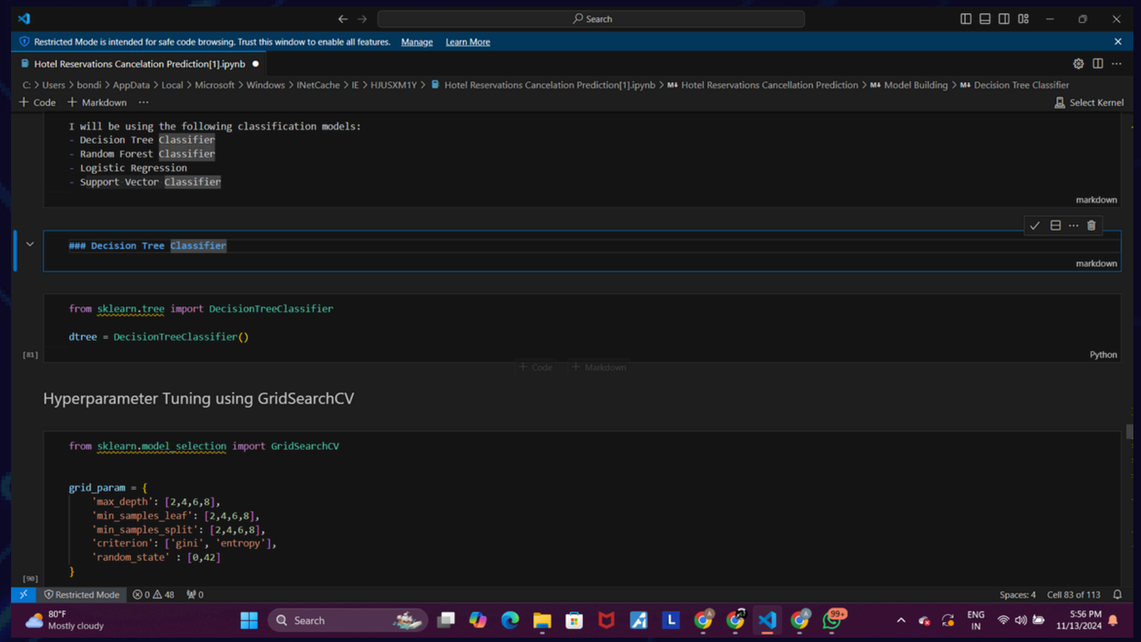
We will be using the following classification models:

- Decision Tree Classifier
- Random forest Classifier
- Logistic Regression

DESICION TREE CLASSIFIER

- A Decision Tree Classifier is a machine learning algorithm that splits data into branches based on feature values, creating a tree-like model for decision-making. It is intuitive and interpretable for classification tasks.


Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
Hotel Reservations Cancelation Prediction[1].ipynb
C: > Users > bondi > AppData > Local > Microsoft > Windows > INetCache > IE > HJUSXM1Y > Hotel Reservations Cancelation Prediction[1].ipynb > Hotel Reservations Cancellation Prediction > Model Building > Decision Tree Classifier
Code Markdown
Select Kernel
Decision Tree Classifier
from sklearn.tree import DecisionTreeClassifier
dtree = DecisionTreeClassifier()
[81]
Python
Hyperparameter Tuning using GridSearchCV
from sklearn.model_selection import GridSearchCV
grid_param = {
'max_depth': [2,4,6,8],
'min_samples_leaf': [2,4,6,8],
'min_samples_split': [2,4,6,8],
'criterion': ['gini', 'entropy'],
'random_state' : [0,42]
}
grid_search = GridSearchCV(estimator=dtree, param_grid=grid_param, cv=5, n_jobs=-1, scoring='accuracy')
grid_search.fit(X_train, y_train)
print(grid_search.best_params_)
[90]


Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
Hotel Reservations Cancelation Prediction[1].ipynb
C: > Users > bondi > AppData > Local > Microsoft > Windows > INetCache > IE > HJUSXM1Y > Hotel Reservations Cancelation Prediction[1].ipynb > Hotel Reservations Cancellation Prediction > Model Building > Decision Tree Classifier
Code Markdown
Select Kernel
I will be using the following classification models:
- Decision Tree Classifier
- Random Forest Classifier
- Logistic Regression
- Support Vector Classifier
markdown
Decision Tree Classifier
markdown
from sklearn.tree import DecisionTreeClassifier
dtree = DecisionTreeClassifier()
[81]
Python
Hyperparameter Tuning using GridSearchCV
from sklearn.model_selection import GridSearchCV
grid_param = {
'max_depth': [2,4,6,8],
'min_samples_leaf': [2,4,6,8],
'min_samples_split': [2,4,6,8],
'criterion': ['gini', 'entropy'],
'random_state' : [0,42]
}
[90]
Restricted Mode
Spaces: 4
Cell 83 of 113
80°F
Mostly cloudy
Search
ENG
IN
5:56 PM
11/13/2024

RANDOM FOREST CLASSIFIER

- A Random Forest Classifier is an ensemble learning algorithm that builds multiple decision trees during training and combines their outputs for more accurate, robust predictions. It reduces overfitting and improves model performance.

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier

#random forest classifier object
rfc = RandomForestClassifier()
```

[83] Python

Hyperparameter Tuning using GridSearchCV

```
from sklearn.model_selection import GridSearchCV

#grid search parameters
grid_param = {
    'max_depth': [2,4,6,8],
    'min_samples_leaf': [2,4,6,8],
    'min_samples_split': [2,4,6,8],
    'criterion': ['gini', 'entropy'],
    'random_state' : [0,42]
}

#grid search object
grid_search = GridSearchCV(estimator=rfc, param_grid=grid_param, cv=5, n_jobs=-1)

#fitting the grid search object to the training data
grid_search.fit(X_train, y_train)

#best parameters
```

[84]

Restricted Mode 0 48 0 Spaces: 4 Cell 83 of 113

Search

Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

Hotel Reservations Cancelation Prediction[1].ipynb

C: > Users > bondi > AppData > Local > Microsoft > Windows > INetCache > IE > HJUSXM1Y > Hotel Reservations Cancelation Prediction[1].ipynb > Hotel Reservations Cancellation Prediction > Model Building > Decision Tree Classifier

+ Code + Markdown ... Select Kernel

```
    'min_samples_leaf': [2,4,6,8],
    'min_samples_split': [2,4,6,8],
    'criterion': ['gini', 'entropy'],
    'random_state' : [0,42]
}

#grid search object
grid_search = GridSearchCV(estimator=rfc, param_grid=grid_param, cv=5, n_jobs=-1)

#fitting the grid search object to the training data
grid_search.fit(X_train, y_train)

#best parameters
print(grid_search.best_params_)
```

[84] Python

```
{'criterion': 'gini', 'max_depth': 8, 'min_samples_leaf': 4, 'min_samples_split': 2, 'random_state': 0}
```

```
#random forest classifier object with best parameters
rfc = RandomForestClassifier(criterion='entropy', max_depth=8, min_samples_leaf=4, min_samples_split=2, random_state=0)

#Training the model
rfc.fit(X_train, y_train)

#Training accuracy
print(rfc.score(X_train, y_train))

#Predicting the test set results
r_pred = rfc.predict(X_test)
```

[85] Python

```
0.850185873605948
```

Restricted Mode 0 48 0 Spaces: 4 Cell 83 of 113

LOGISTIC REGRESSION

- Logistic Regression is a statistical model used for binary classification that predicts the probability of an outcome using a logistic function. It outputs probabilities between 0 and 1, ideal for decision boundaries.

Logistic Regression

```
from sklearn.linear_model import LogisticRegression

#logistic regression object
logreg = LogisticRegression()
```

[86] Python

Hyperparameter Tuning using GridSearchCV

```
from sklearn.model_selection import GridSearchCV

#grid search parameters
grid_param = {
    'penalty': ['l1', 'l2', 'elasticnet', 'none'],
    'C': [0.001,0.01,0.1,1,10,100,1000],
    'solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'],
    'random_state' : [0,42]
}

#grid search object
grid_search = GridSearchCV(estimator=logreg, param_grid=grid_param, cv=5, n_jobs=-1)

#fitting the grid search object to the training data
grid_search.fit(X_train, y_train)

#best parameters
print(grid_search.best_params_)
```

[87]

Restricted Mode ⊗ 0 ⚠ 48 0 Spaces: 4 Cell 83 of 113

```
#logistic regression object with best parameters
logreg = LogisticRegression(C=1, penalty='l2', random_state=0, solver='liblinear')

#Training the model
logreg.fit(X_train, y_train)

#Training accuracy
print(logreg.score(X_train, y_train))

#Predicting the test set results
l_pred = logreg.predict(X_test)
```

[88] Python

```
0.7956877323420074
```

MODEL EVALUATION

Confusion matrix

A confusion matrix heatmap visually represents a classification model's performance, showing true positives, false positives, true negatives, and false negatives using color intensity.

Distribution plot

A distribution plot visualizes the frequency distribution of a dataset, showing how data points are spread across different values or intervals.

CONFUSION MATRIX

Confusion Matrix Heatmap

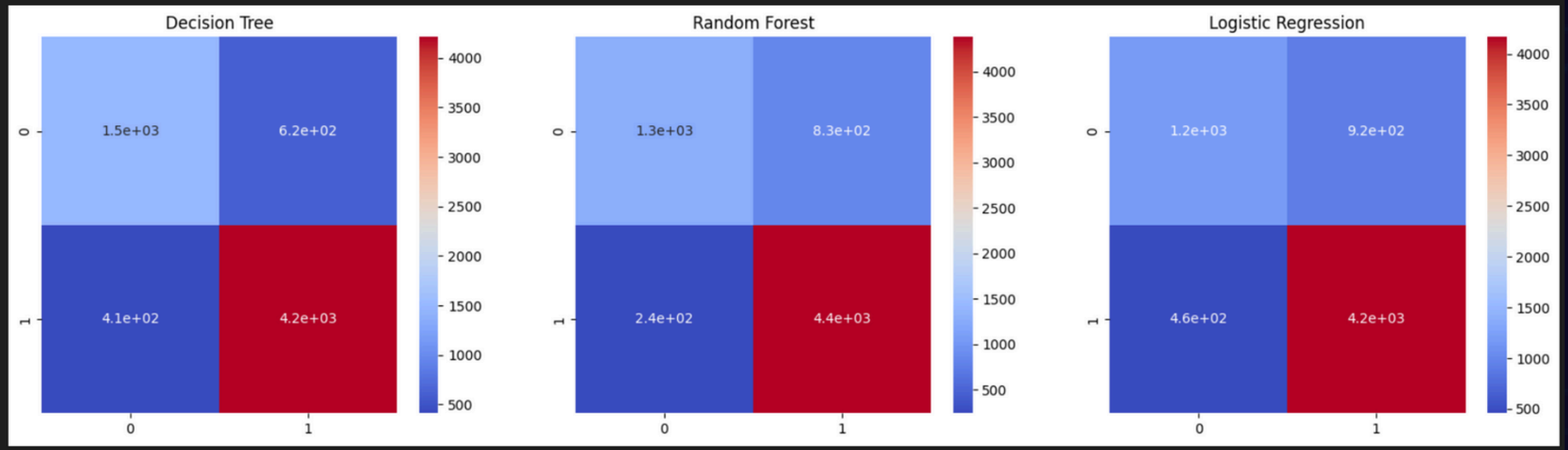
+ Code + Markdown

```
from sklearn.metrics import confusion_matrix

fig, ax = plt.subplots(1,3,figsize=(20,5))

#decision tree
sns.heatmap(confusion_matrix(y_test, d_pred), annot=True, cmap='coolwarm', ax=ax[0]).set_title('Decision Tree')
#random forest
sns.heatmap(confusion_matrix(y_test, r_pred), annot=True, cmap='coolwarm', ax=ax[1]).set_title('Random Forest')
#logistic regression
sns.heatmap(confusion_matrix(y_test, l_pred), annot=True, cmap='coolwarm', ax=ax[2]).set_title('Logistic Regression')
```

89]

Python

```
Text(0.5, 1.0, 'Logistic Regression')
```

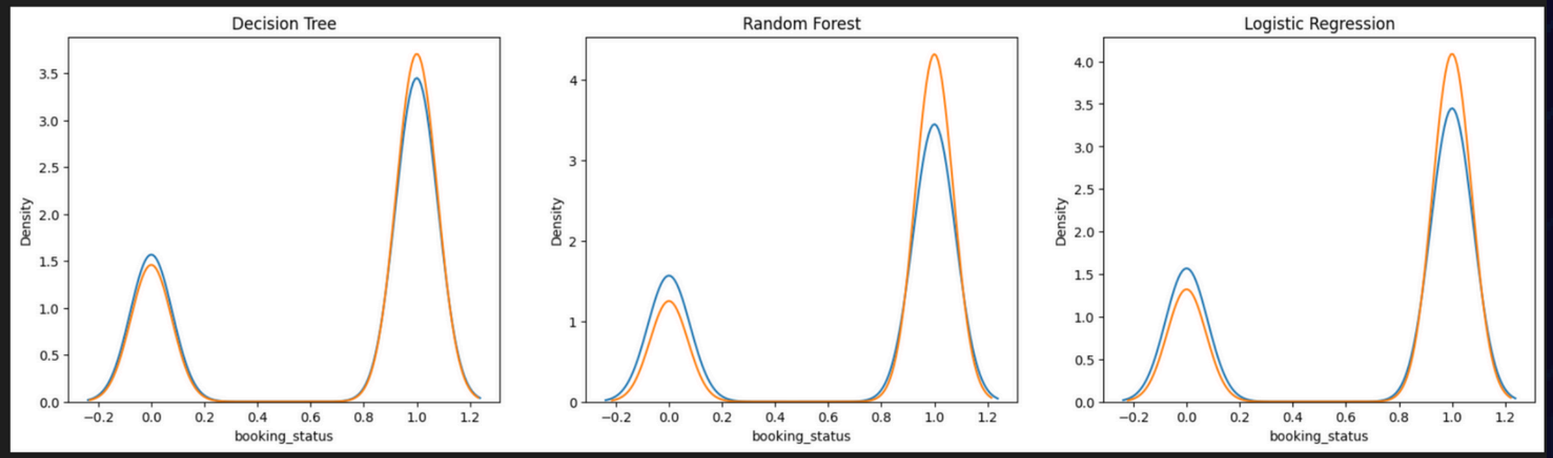


DISTRIBUTION PLOT

Distribution Plot

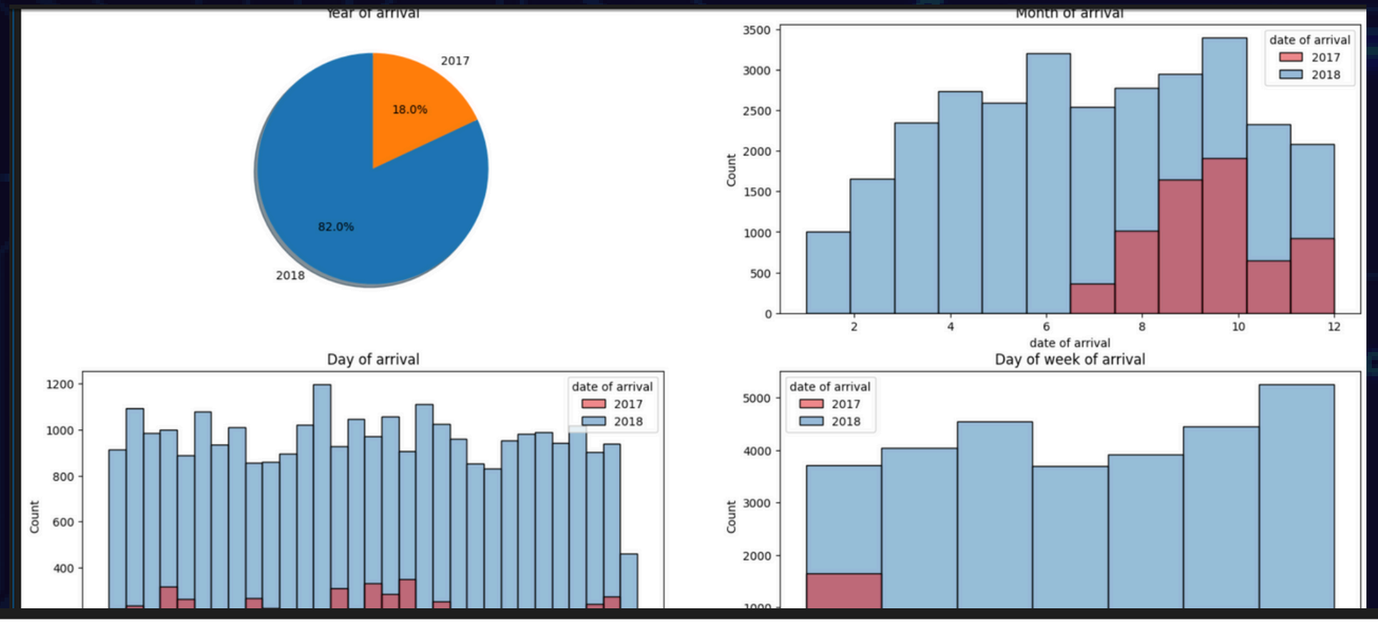
```
fig, ax = plt.subplots(1,3,figsize=(20,5))

#decision tree
sns.distplot(y_test, ax=ax[0], hist= False).set_title('Decision Tree')
sns.distplot(d_pred, ax=ax[0], hist = False)

#random forest
sns.distplot(y_test, ax=ax[1], hist= False).set_title('Random Forest')
sns.distplot(r_pred, ax=ax[1], hist = False)

#logistic regression
sns.distplot(y_test, ax=ax[2], hist= False).set_title('Logistic Regression')
sns.distplot(l_pred, ax=ax[2], hist = False)
```

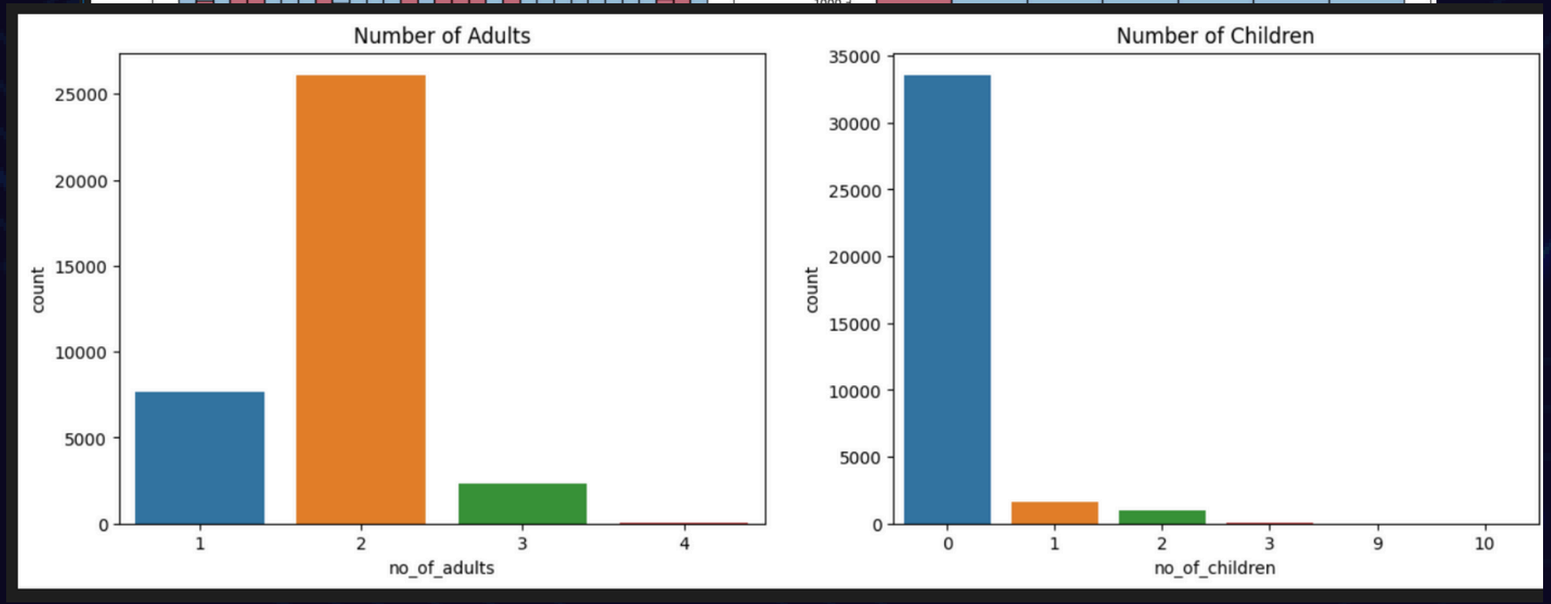
[97] Python




Cancellation in 2017
count
booking_status
Not_Canceled
Canceled
Cancellation in 2018
count
booking_status
Not_Canceled
Canceled
Cancellation by months in 2017
booking_status
Not_Canceled
Canceled
Count
date of arrival
Cancellation by months in 2018
booking_status
Not_Canceled
Canceled
Count
date of arrival
Cancellation by date in 2017
booking_status
Not_Canceled
Canceled
Count
date of arrival
Cancellation by date in 2018
booking_status
Not_Canceled
Canceled
Count
date of arrival
Cancellation by day of week in 2017
booking_status
Not_Canceled
Canceled
Count
date of arrival
Cancellation by day of week in 2018
booking_status
Not_Canceled
Canceled
Count
date of arrival

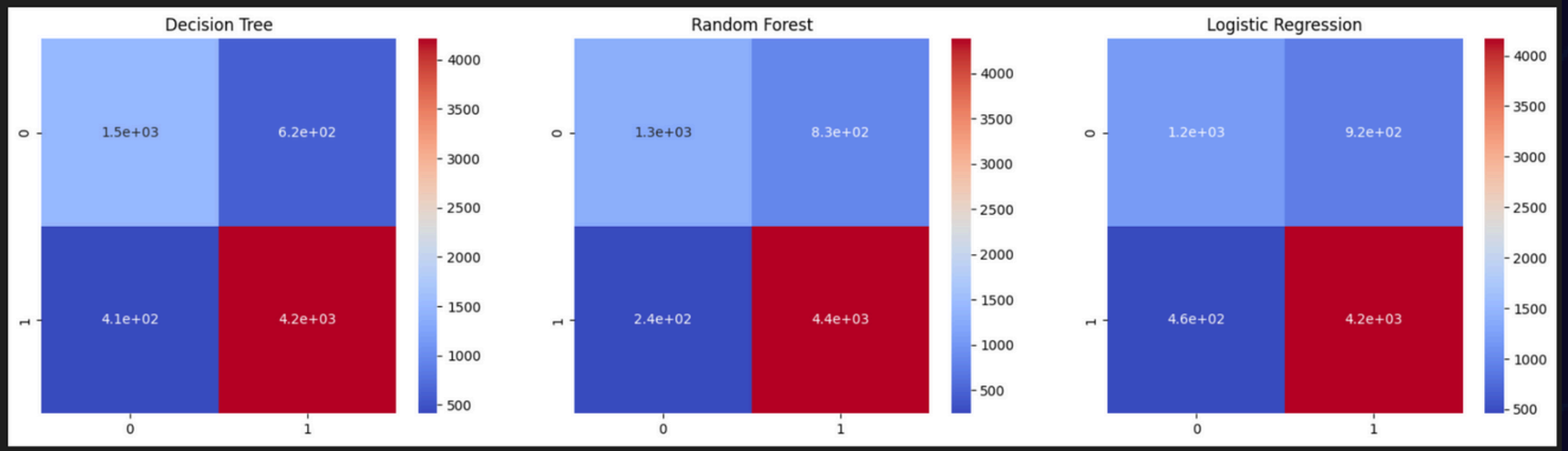
VISUALIZATION TECHNIQUES

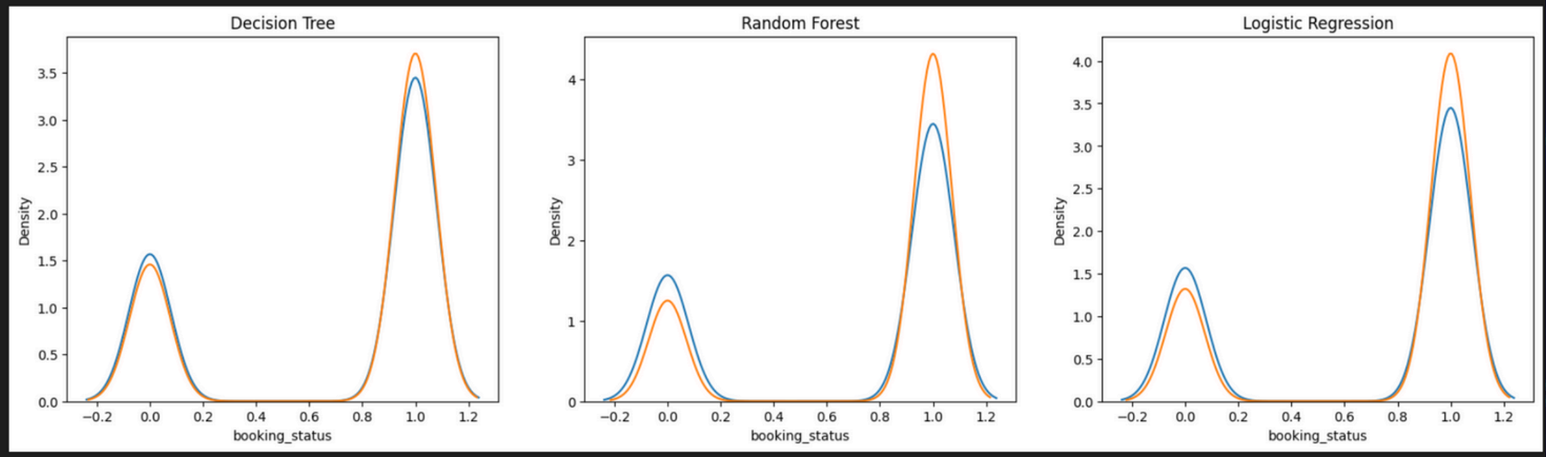




VISUALIZATION TECHNIQUES

```
Text(0.5, 1.0, 'Logistic Regression')
```





THANK YOU