Final Project
Hotel booking Analysis

Katz
Katz School
of Science and Health

Team Member: Ruoyu Chen, Qing Dou, Zhengnan Li

Repository Link : https://github.com/Zhengnan817/zhengnan.git

1.Introduction

Topic:

This project focused on predicting hotel booking cancellations using machine learning models and time-series analysis to assist hotels in managing operations more effectively. This is important for hotels because bookings can severely impact a hotel's revenue and related operational strategies.

Data Source:

1. Hotel reservations in Lisbon, Portugal from 2015 to 2017
2. Weather information in Lisbon.

Research Questions:

1. What are the key factors affecting hotel booking cancellation, and how can we predict the cancellations?
2. How can we predict hotel booking volumes using time series and ARIMA models?

2.Research Approach

Explanatory Data Analysis
Data Preparation

ARIMA & SARIMA

Machine Learning models
Logistic Regression
Neural Network
Decision Tree
Model Selection

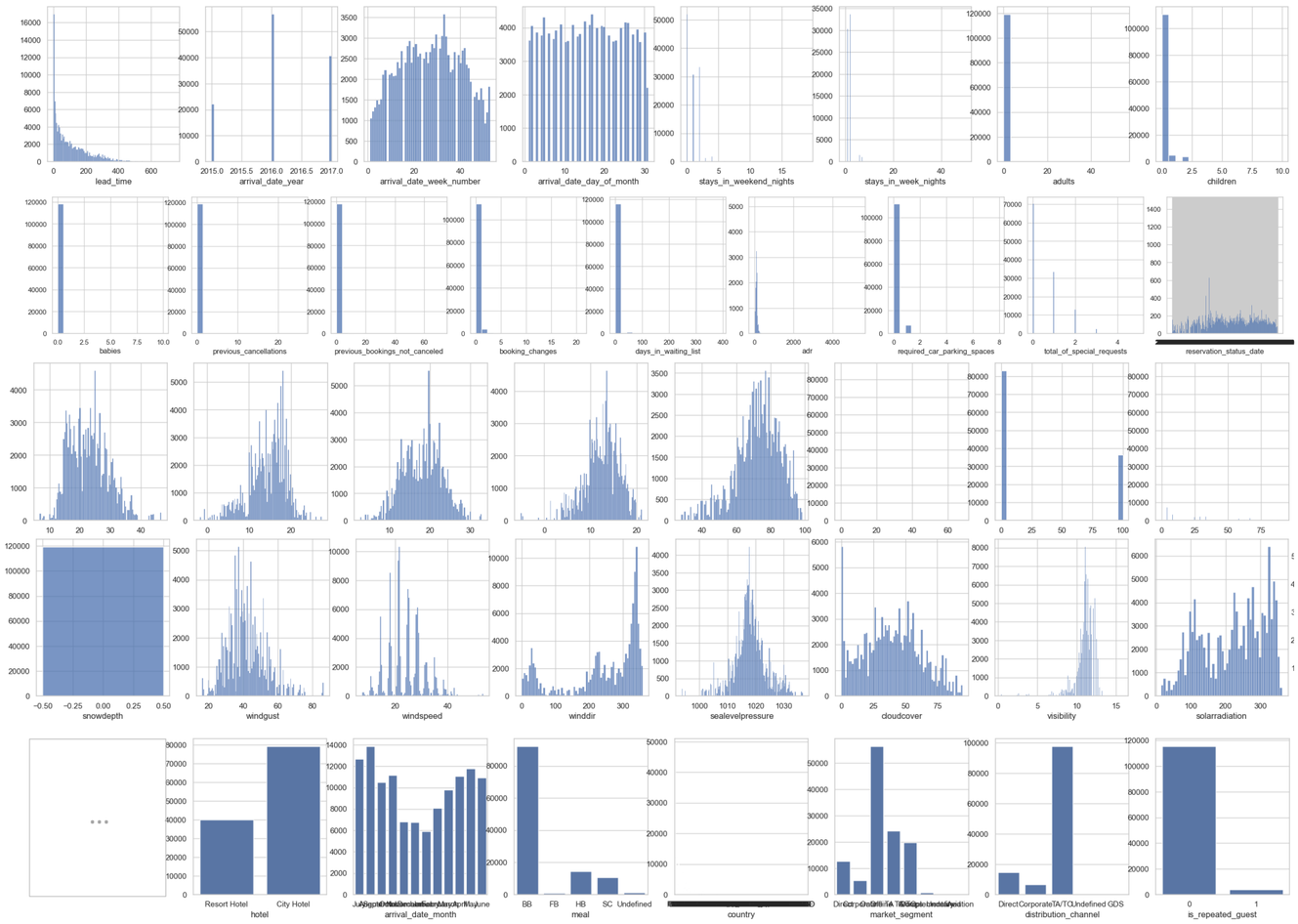
Ensemble Model
Conclusions

3.Exploratory Data Analysis

Dimension: 119,390 rows, 66 columns

Data Dictionary

Variable	Description
hotel	Hotel (Resort Hotel or City Hotel)
lead_time	Number of days that elapsed between the entering date of the booking into the PMS and the arrival date
arrival_date_year	Year of arrival date
arrival_date_month	Month of arrival date
arrival_date_week_number	Week number of year for arrival date
arrival_date_day_of_month	Day of arrival date
stays_in_weekend_nights	Number of weekend nights (Saturday or Sunday) the guest stayed or booked to stay at the hotel
stays_in_week_nights	Number of week nights (Monday to Friday) the guest stayed or booked to stay at the hotel
adults	Number of adults
children	Number of children
babies	Number of babies
meal	Type of meal booked. Categories are presented in standard hospitality meal packages
country	Country of origin. Categories are represented in the ISO 3155–3:2013 format
market_segment	Market segment designation. In categories, the term "TA" means "Travel Agents" and "TO" means "Tour Operators"
distribution_channel	Booking distribution channel. The term "TA" means "Travel Agents" and "TO" means "Tour Operators"
is_repeated_guest	Value indicating if the booking name was from a repeated guest (1) or not (0)
previous_cancellations	Number of previous bookings that were cancelled by the customer prior to the current booking
previous_bookings_not_canceled	Number of previous bookings not cancelled by the customer prior to the current booking



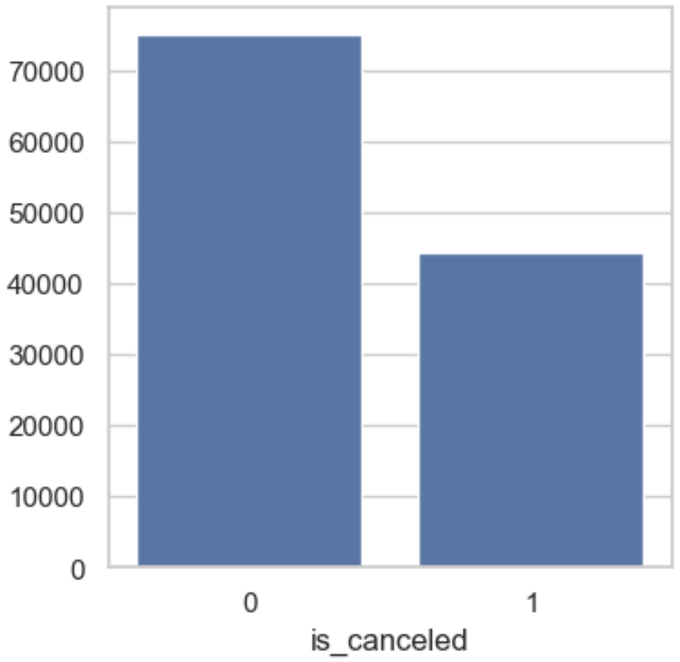
4.Data Preparation

Check null values → Outliers → Feature Selection → Step Forward Selection → Data Transformation

5.Prepped Data Overview

➢ **Target Variable**

The data is slightly biased and the Null Error Rate is 0.62.



➢ **Explanatory Variable**

The dimension of the dataset is 112,233 rows and 20 columns.

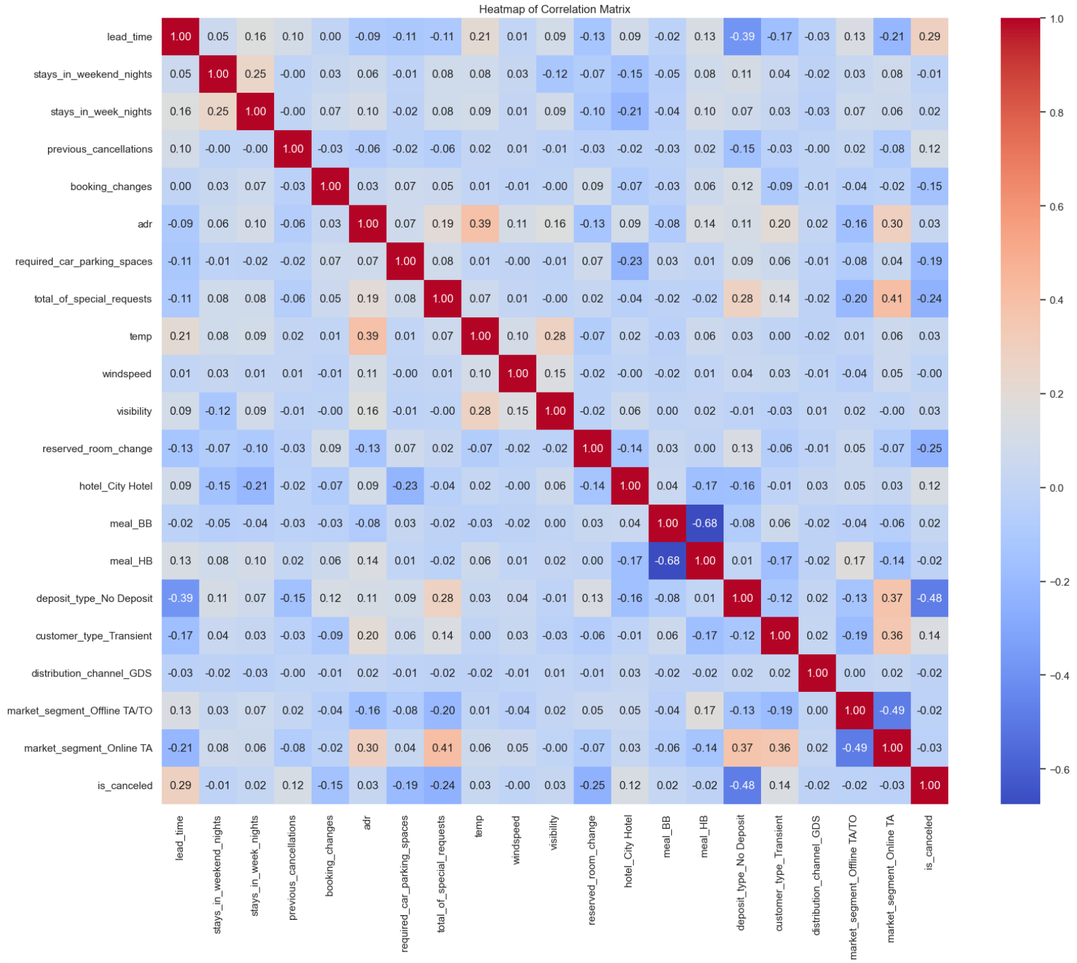
	lead_time	stays_in_weekend_nights	stays_in_week_nights	previous_cancellations	booking_changes	adr	required_car_parking_spaces	total_of_special_requests	temp	windspeed
0	342	0	0	0	3	0.00	0	0	19.8	21.7
1	737	0	0	0	4	0.00	0	0	19.8	21.7
2	7	0	1	0	0	75.00	0	0	19.8	21.7
3	13	0	1	0	0	75.00	0	0	19.8	21.7
4	14	0	2	0	0	98.00	0	1	19.8	21.7
...	...	...	...	...	...	...	...	...	...	...
119385	53	2	4	0	0	88.40	0	1	16.9	18.2
119386	118	2	4	0	0	86.67	0	1	16.9	18.2
119387	12	2	5	0	0	89.25	0	3	16.9	18.2
119388	39	2	5	0	0	64.67	0	3	16.9	18.2
119389	65	2	5	0	0	88.00	0	1	16.9	18.2

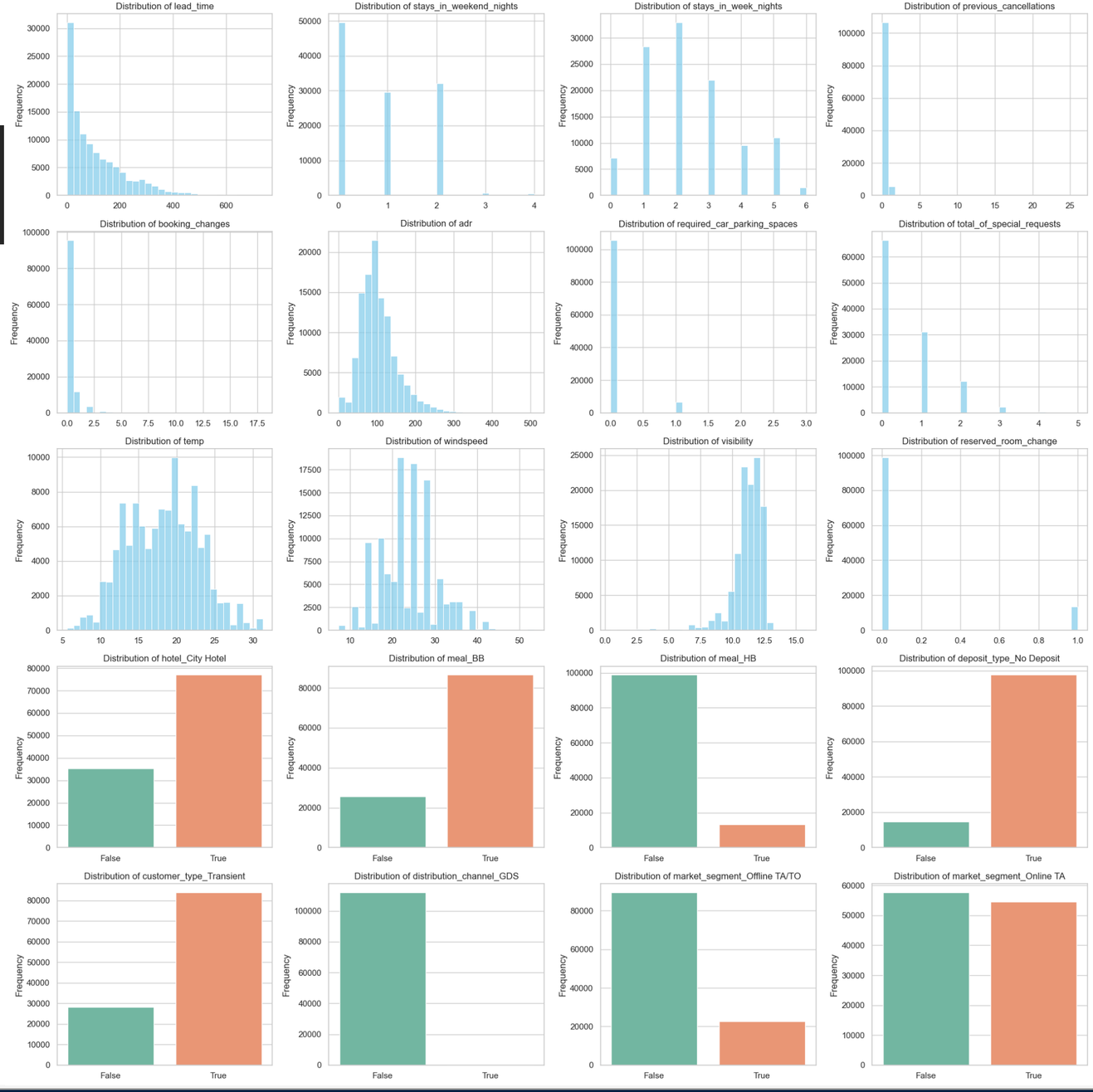
112233 rows × 20 columns

visibility	reserved_room_change	hotel_City Hotel	meal_BB	meal_HB	deposit_type_No Deposit	customer_type_Transient	distribution_channel_GDS	market_segment_Offline TA/TO	market_segment_Online TA
12.0	0	False	True	False	True	True	False	False	False
12.0	0	False	True	False	True	True	False	False	False
12.0	1	False	True	False	True	True	False	False	False
12.0	0	False	True	False	True	True	False	False	False
12.0	0	False	True	False	True	True	False	False	True
...	...	...	...	...	...	...	...	...	...
10.6	0	True	True	False	True	True	False	False	True
10.6	0	True	True	False	True	True	False	False	True
10.6	0	True	True	False	True	True	False	False	True
10.6	0	True	True	False	True	True	False	False	True
10.6	0	True	False	False	True	True	False	False	True

5.Prepped Data Review

```
columns = ['lead_time', 'stays_in_weekend_nights', 'stays_in_week_nights', 'previous_cancellations',
           'booking_changes', 'adr', 'required_car_parking_spaces', 'total_of_special_requests', 'temp',
           'windspeed', 'visibility', 'reserved_room_change', 'hotel_City Hotel', 'meal_BB', 'meal_HB',
           'deposit_type_No Deposit', 'customer_type_Transient', 'distribution_channel_GDS',
           'market_segment_Offline TA/TO', 'market_segment_Online TA']
```





6.ARIMA Model

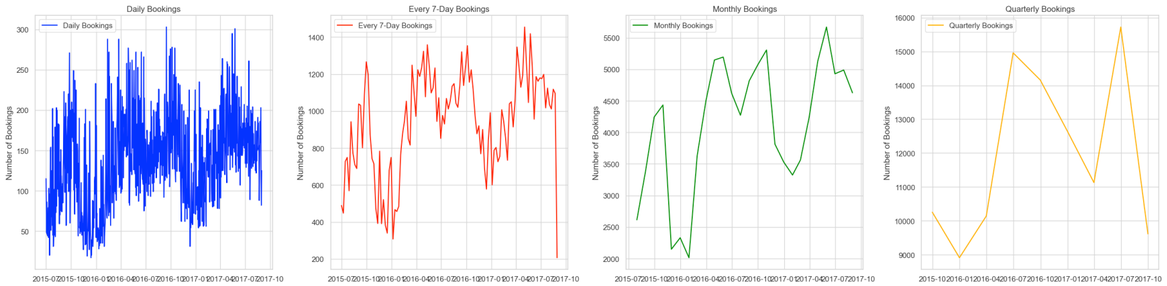
Hypothesis: The variations in hotel booking volumes are significantly influenced by inherent time series trends.

➢ **Using set_index function to transform the data into time series type.**

```
hotel.set_index('datetime', inplace=True)
daily_bookings = hotel.resample('D').size()
```

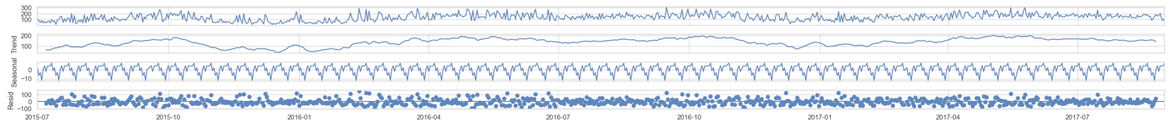
datetime	
2015-07-01	115.0
2015-07-02	85.0
2015-07-03	48.0
2015-07-04	86.0
2015-07-05	46.0

➢ **Plot the data in different periods(Daily, Weekly, Monthly and Quarterly).**



➢ **Using daily bookings to do the analysis.**

- **Remove 5 outliers**
- **Seasonal Decompose**
  Trend: Slight fluctuations
  Seasonal: Regular, Consider seasonal components
  Residual: Randomly, Captured the trend and seasonality in the data well.



- **ADF**

```
ADF Statistic: -2.214810
p-value:       0.200914
```

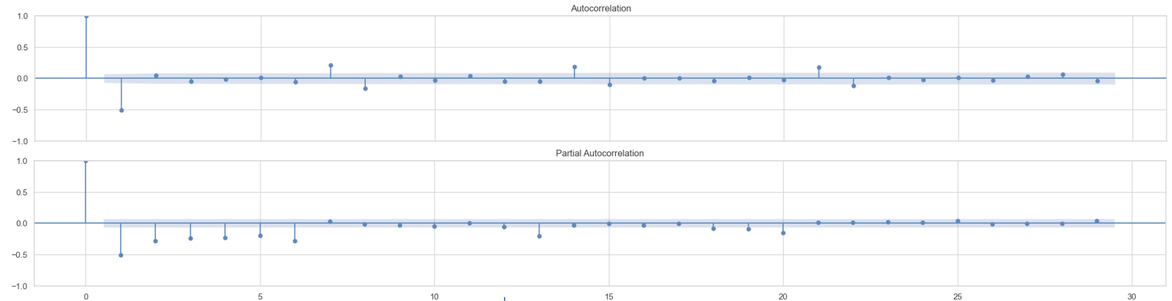
first differencing

```
ADF Statistic: -11.332986
p-value:       0.000000
```

```
Critical Values: 1%: -3.439 5%: -2.865 10%: -2.569
Reject the null hypothesis (H0), the data does not have a unit root and is stationary.
```

- **ACF& PACF**

p=0 (autoregressive order): First lag is insignificant.
d=1 (difference order): Perform first differencing.
q=0 (moving average order): autocorrelation coefficients for all lags are low and close to 0, moving average order q is probably also 0.



(6.1.1) (1.1.1) **(0.1.0)**

```
model = ARIMA(daily_bookings_diff,
order=(0, 1, 0))
model_fit = model.fit()
```

Log Likelihood	-4772.657
AIC	9547.313
BIC	9551.969
Sigma2 P value	0.000
MSE	5.415
RMSE	7.358

```
sarima_model_daily = SARIMAX(dailyly_bookings,
order=(0, 1, 0),
seasonal_order=(0, 1, 0, 30))
sarima_result_daily = sarima_model_daily.fit()
```

Log Likelihood	-766.940
AIC	1539.881
BIC	1553.729
Sigma2 P value	0.000
MSE	18079.214
RMSE	134.458

6.ARIMA Model

➢ **Using set_index function to transform the data into time series type.**

```
monthly_bookings = daily_bookings.resample('M').sum()
```

datetime	
2015-07-31	2614.0
2015-08-31	3397.0
2015-09-30	4243.0
2015-10-31	4434.0
2015-11-30	2148.0

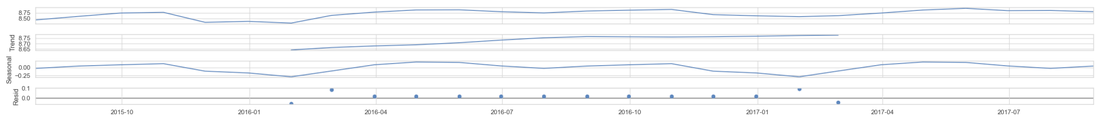
➢ **Using monthly bookings to do the analysis.**

- **Seasonal Decompose**

  Trend: Slight fluctuations

  Seasonal: Light Regular, Consider seasonal components

  Residual: Randomly, Captured the trend and seasonality in the data well.



- **ADF**

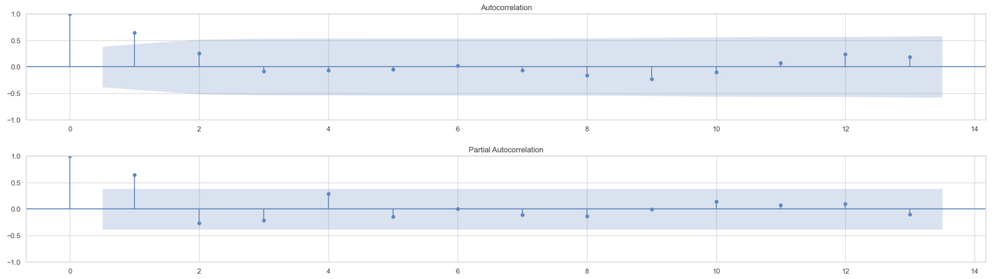
```
ADF Statistic: -2.653733
p-value:       0.082392
```

first differencing →

```
ADF Statistic: -3.298767
p-value:       0.014934
```

- **ACF& PACF**

p=1 (autoregressive order): First lag is significant.
d=1 (difference order): Perform first differencing.
q=1 (moving average order): First lag is significant.

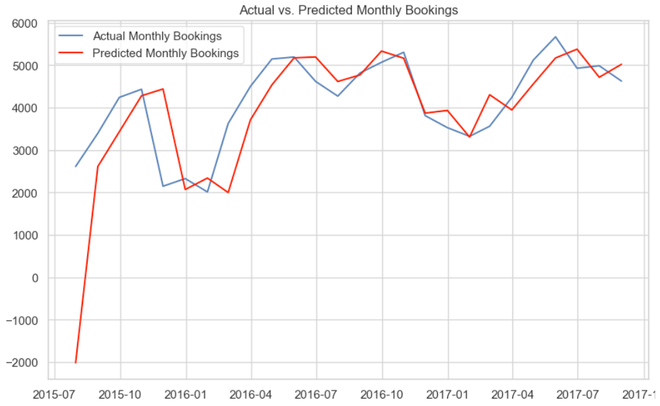


ACF (autocorrelation function) and PACF (partial autocorrelation function) plots are often used to make preliminary judgments about the parameters p and q of the ARIMA model.

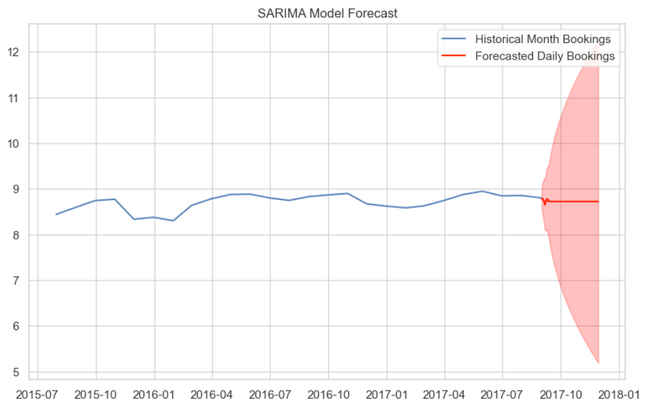
```
sarima_model_monthly = SARIMAX(monthly_bookings_log,order=(0, 1, 1),
seasonal_order=(0, 1, 1, 12))
month_results = sarima_model_monthly.fit()
```

Log Likelihood	10.312	ma.L1	0.882
AIC	-10.623	ar.S.L12	0.971
BIC	-7.799	ma.S.L12	0.745
MSE	1312665.261	Sigma2 P value	0.970
RMSE	1145.716		

In-sample Forecast



Out-sample Forecast



Conclusion：

- Daily Cycle Model indicating the model are not good fit for the data. Monthly cycle model fits the data better but has a large prediction error.
- ACF and PACF show that most lags are not significant, and the parameters of the model mostly have high p-values, which indeed may indicate that there is no significance in the data.
- The hypothesis that only time factor significantly affects booking quantity is rejected.
- **We can consider using machine learning models and other variables to predict the number of reservations or whether to cancel reservations.**

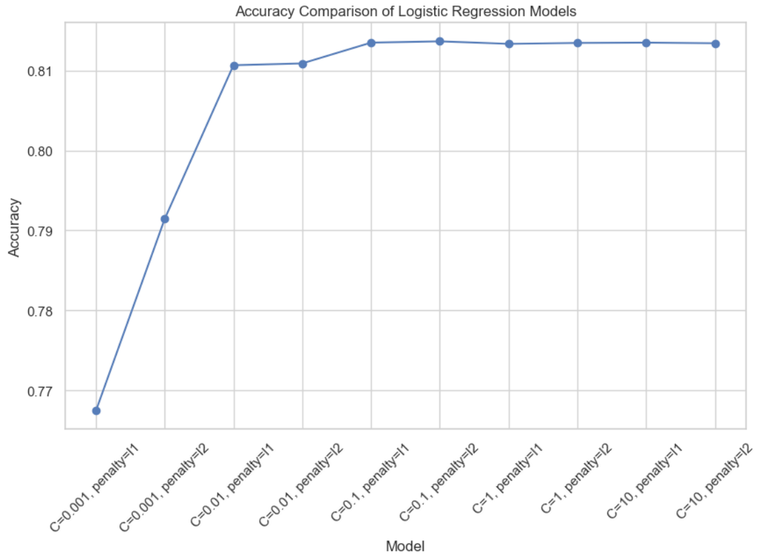
6.2 Logistic Regression

➢ **Using GridSearchCV to tune hyperparameters of a logistic regression model with a liblinear solver.**

- Optimized the model by testing various C values and penalty types with the liblinear solver.
- Stratified K-Fold cross-validation (5 folds) ensures balanced class distributions.
- The best model is selected based on accuracy, which is plotted for different hyperparameter combinations.

```
# Hyperparameters to tune
param_grid = {'C': [0.001, 0.01, 0.1, 1, 10], 'penalty': ['l1', 'l2']}
# Use liblinear solver
solver = 'liblinear'

# Initialize Logistic Regression model
model = LogisticRegression(solver=solver, max_iter=1000)

# Use Stratified K-Fold for cross-validation
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=37)

# GridSearchCV to find the best hyperparameters
grid_search = GridSearchCV(model, param_grid, cv=cv, scoring='accuracy')
grid_search.fit(X_train, y_train)
```

➢ **Best model selected with C: 1 and penalty: 'l2'.**



➢ **Finally, the best model is evaluated on the test set, with metrics like accuracy, Roc and Auc, R-squared, precision, recall, F1 score, and the confusion matrix calculated for comprehensive evaluation.**

Metric	Cross Validation	Logistic Regression
Accuracy	0.8096	0.8132
Precision	0.8158	0.816
Recall	0.8096	0.8132
F1 Score	0.8022	0.8068
ROC AUC	0.7735	0.7785
Confusion Matrix	N/A	[[19269, 1600], [4690, 8111]]

6.3 Neural Network

➢ Using keras API to build a sequential model with 4 layers

- The model has 4 Dense layer, the first layer start with 64 neurons and activation function of 'relu'. Followed by layer 2 with 32 neurons, layer3 with 16 neurons. Finally a layer4 with function 'sigmoid' to output the probability of the target variable. The model was trained with an EarlyStopping callback to avoid overfitting.

```
nn_model = Sequential()
nn_model.add(Dense(64, input_shape=(X.shape[1],), activation='relu', name="layer1"))
nn_model.add(Dense(32, activation='relu', name="layer2"))
nn_model.add(Dense(16, activation='relu', name="layer3"))
nn_model.add(Dense(1, activation='sigmoid', name="layer4"))
nn_model.summary()

nn_model.compile(optimizer='adam',
                 loss='binary_crossentropy',
                 metrics=['accuracy','AUC'])

es = EarlyStopping(monitor='val_accuracy',
                                    mode='max',
                                    patience=8,
                                    restore_best_weights=True)
history = nn_model.fit(X_train,
                       y_train,
                       callbacks=[es],
                       epochs=40,
                       batch_size=32,
                       validation_split=0.2,
                       shuffle=True,
                       verbose=1)
```

Layer (type)	Output Shape	Param #
layer1 (Dense)	(None, 64)	1,344
layer2 (Dense)	(None, 32)	2,080
layer3 (Dense)	(None, 16)	528
layer4 (Dense)	(None, 1)	17

➢ The training loss and vilidation loss is gradually declining.

➢ Use the metrics to examine the model's performance. The model achieved a accuracy of 0.825 and F1 score of 0.822, with AUC score of 0.799.

Metric	Value
Accuracy	0.8245
Precision	0.8258
Recall	0.8245
F1 Score	0.8197
AUC Score	0.7942

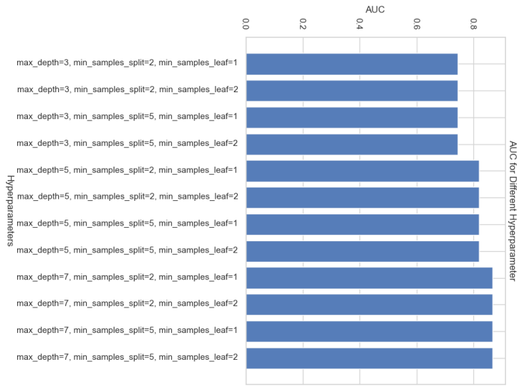
Confusion Matrix	Predicted Negative	Predicted Positive
Actual Negative	19215	1668
Actual Positive	4242	8545

6.4 Decision Tree Model

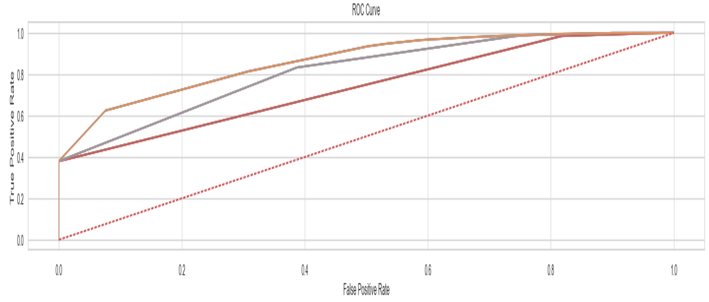
- **Performs hyperparameter tuning for a Decision Tree model by experimenting with different combinations of max_depth, min_samples_split, and min_samples_leaf.**

```python
# Hyperparameter Tuning
hyperparameters = {'max_depth': [3, 5, 7], 'min_samples_split': [2, 5], 'min_samples_leaf': [1, 2]}
roc_auc_values = []
combinations = []

# Train a subset of hyperparameter combinations
for max_depth in hyperparameters['max_depth']:
    for min_samples_split in hyperparameters['min_samples_split']:
        for min_samples_leaf in hyperparameters['min_samples_leaf']:
            model = DecisionTreeClassifier(criterion='entropy', max_depth=max_depth, min_samples_split=min_samples_split, min_samples_leaf=min_samples_leaf)
            model.fit(X_train, y_train)

            y_val_pred_proba = model.predict_proba(X_test)[:, 1]
            fpr, tpr, _ = roc_curve(y_test, y_val_pred_proba)
            roc_auc = auc(fpr, tpr)
            roc_auc_values.append(roc_auc)
            combinations.append((max_depth, min_samples_split, min_samples_leaf))

            # Plot ROC curve for this model
            plt.plot(fpr, tpr, label=f'max_depth={max_depth}, min_samples_split={min_samples_split}, min_samples_leaf={min_samples_leaf} (AUC={roc_auc:.2f})')
```

- **Each model is trained and evaluated on the test set to compute its ROC AUC, and the best model is selected based on the highest AUC score.**



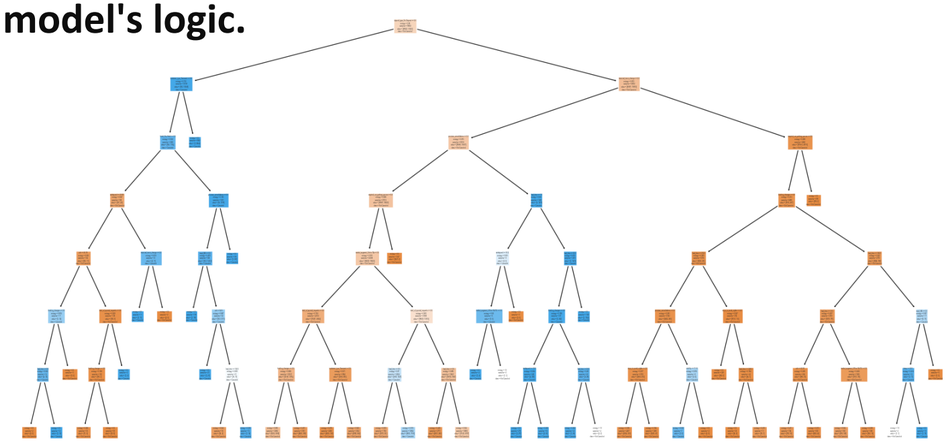
the best model based on the highest AUC score. This model was selected with max_depth=7, min_samples_split=2, and min_samples_leaf=2.



- **Then trains a Decision Tree model with the optimal hyperparameters and evaluates the model's performance on the test set, including metrics like ROC AUC, accuracy, precision, recall, and F1 score.**

Metric	Cross Validation	Best Decision Tree
Accuracy	0.7687	0.8131
Precision	0.7732	0.8162
Recall	0.7687	0.8131
F1 Score	0.7599	0.8065
ROC AUC	0.7325	0.7777
Parameters	N/A	max_depth=7, min_samples_split=2, min_samples_leaf=2
AUC on Test Set	N/A	0.87

- **Visualizes the structure of the Decision Tree to interpret the model's logic.**



7 Model Selection

- **To select the best model, we evaluate three different models: Logistic Regression, Neural Network, and Decision Tree. Analyze their performance based on accuracy, precision, recall,F1 score and AUC.**

```
import pandas as pd
from sklearn.metrics import accuracy_score, recall_score, f1_score, roc_auc_score, conf

# Evaluate Logistic Regression
best_logreg = grid_search.best_estimator_
y_test_pred_logreg = best_logreg.predict(X_test)
y_test_pred_proba_logreg = best_logreg.predict_proba(X_test)[:, 1]
metrics_test('Logistic Regression', y_test,y_test_pred_logreg)
# Evaluate Neural Network
preds_nn = np.round(nn_model.predict(X_test), 0)
y_pred_nn_model = (preds_nn > 0.5).astype(int)
metrics_test('Neural Network', y_test,y_pred_nn_model)
# Evaluate Decision Tree
y_test_pred_proba_dtree = best_model_dt.predict_proba(X_test)[:, 1]
y_test_pred_dtree = best_model_dt.predict(X_test)
metrics_test('Decision Tree', y_test,y_test_pred_dtree)
```

- **The Neural Network model seems to be the best overall performer, having the highest accuracy, recall, and F1 score, showing good balance between precision and recall.**

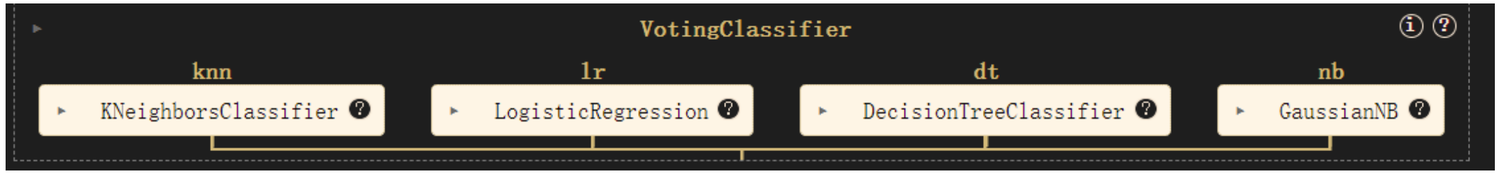
Model	Accuracy	Precision	Recall	F1 Score	AUC
Logistic Regression	0.8132	0.816	0.8132	0.8068	0.7784
Neural Network	0.8186	0.8194	0.8186	0.8138	0.7883
Decision Tree	0.8103	0.8136	0.8103	0.8034	0.7744

8.Ensemble Model

- Using K-nearest neighbor, logistic regression, decision tree and naïve beyes.

```
ensemble = VotingClassifier(estimators=[
    ('knn', knn),
    ('lr',log_reg),
    ('dt',dt),
    ('nb',nb_classifier)
], voting='soft')
ensemble
```

- Combines the output of the individual component models by a VotingClassifier to calculate a prediction.
- We also uses a 'soft' voting method to avoid when the votes are 2 against 2.



- Compared with the individual component models, the ensemble model has a better performance across all metrics.

Model	Accuracy	Precision	Recall	F1 Score	AUC Score
KNN	0.7583	0.7549	0.7583	0.7551	0.7328
Logistic Regression	0.8135	0.8161	0.8135	0.8072	0.7791
Naive Bayes	0.7583	0.7549	0.7583	0.7551	0.7328
Decision Tree	0.8177	0.8180	0.8177	0.8178	0.8074
Ensemble	**0.8286**	**0.8276**	**0.8286**	**0.8279**	**0.8145**

8.Conclusions

- Key Factors Affecting Booking Cancellations

We analyzed various models to identify the key factors influencing booking cancellations. Through feature selection and model evaluations, we discovered that variables such as lead time, booking changes, and average daily rates played significant roles in determining the likelihood of cancellations. By accurately identifying these factors, hotels can implement targeted strategies, such as offering incentives for longer stays or adjusting cancellation policies, to reduce the impact of cancellations.

- Predicting Daily Hotel Booking Volumes Using Time-Series Analysis

To forecast hotel booking volumes and help optimize hotel strategy, we employed time-series analysis through SARIMA models on daily and monthly data. The analysis revealed consistent seasonal trends, enabling us to anticipate booking volumes with reasonable accuracy. Although the monthly cycle model did not provide a perfect fit, the Time-Series model still offers a valuable approach for forecasting.

- Machine Learning Model Performance

Among the machine learning models, the Neural Network model has a strong performance with high accuracy, recall, and F1 score. The Decision Tree model provided interpretability while achieving good accuracy, and Logistic Regression presented robust AUC scores. However, combining these models into an ensemble further improved prediction accuracy to 0.82 and improved on other metrics.

Model	Accuracy	Precision	Recall	F1 Score	AUC Score
KNN	0.7583	0.7549	0.7583	0.7551	0.7328
Logistic Regression	0.8135	0.8161	0.8135	0.8072	0.7791
Naive Bayes	0.7583	0.7549	0.7583	0.7551	0.7328
Decision Tree	0.8177	0.8180	0.8177	0.8178	0.8074
Ensemble	**0.8286**	**0.8276**	**0.8286**	**0.8279**	**0.8145**

8.Conclusions

In conclusion, our predictive models provide practical insights for hotels to anticipate booking cancellations and manage daily booking volumes.

By incorporating machine learning and time-series analysis, hotels can enhance operational efficiency and profitability. Future research could focus on refining these models with additional data sources and more advanced techniques to further improve prediction accuracy.

9.Challenges

- 1. Arima parameters optimizing
- 2. Build a Neural net work using Keras API
- 3. Ensemble component choice

Thank You

Team Member: Ruoyu Chen, Qing Dou, Zhengnan Li

Repository Link : https://github.com/Zhengnan817/zhengnan.git

10.References

Dataset source: https://github.com/rfordatascience/tidytuesday/tree/master/data/2020/2020-02-11

Domain Knowledge:
Antonio, N., de Almeida, A., and Nunes, L. (2019). Hotel booking demand datasets: https://www.sciencedirect.com/science/article/pii/S2352340918315191

- **https://keras.io/api/layers/**
- **https://scikit-learn.org/stable/modules/ensemble.html**
- **https://medium.com/luca-chuangs-bapm-notes/build-a-neural-network-in-python-binary-classification-49596d7dcabf**
- **https://medium.com/ai-advances/boost-your-ml-models-performance-with-ensemble-modeling-9d0d5df8f307**