Comprehensive Analysis of Passenger Arrivals at Boston Logan Airport: Unraveling Seasonal Trends, Impact on Hotel Occupancy, and Housing Market Dynamics

Co-authors: Amith Ramaswamy , Nikhil Premachandra Rao , Sai Ruchitha Babu , Prajwal S V

ISSUES

1. Houses and People Arriving

We want to figure out if the number of houses being sold is anyhow related with how many people are landing at Boston Logan Airport.

- Does it seem like more houses are sold when more people are flying in?
- Could the changing number of folks at the airport affect how many homes are being bought or sold?

2. Seasonal Vibes in the Housing Market

We're curious if there are specific times of the year when more houses are bought or sold, maybe matching up with when more people travel.

- Are there certain seasons where more houses are sold, and does this match up with times when more people are at the airport?
- How do these times of the year affect when people decide to buy or sell houses?

3. Hotels and Homes Connection

We're looking into how the number of people staying in hotels connects to the trends in people buying homes.

- Does the number of people in hotels relate to how many people are buying houses?
- Are there noticeable changes in the housing market after more or fewer people stay in hotels?

4. Figuring Out Hotel Declines

We're trying to understand why hotels might be getting less busy over time and what that could mean for the wider hospitality scene.

- What's causing hotels to have fewer guests, and does it connect to other factors?
- How could this affect not just hotels but also other parts of the hospitality business?

FINDINGS:

1. More People Choosing Boston Logan Airport Over Time:

- When more and more people are deciding to fly into Boston Logan Airport. It's not just a random increase; it's happening in a consistent and straight-line way, showing a steady growth in airport traffic**.**
- This consistent and gradual rise in the number of people choosing to land at Boston Logan Airport. This suggests a continual growth in airport activity over the time we've been looking at. It's like more and more people are opting to travel through this airport**.**

2. Busy and Slow Times, like a Seasonal Dance:

- We noticed a pattern in the numbers. Sometimes, especially in the middle of the year, there's a peak in people arriving at the airport. It's like a season where things get busy, and then it settles down afterward.
- There's a peak, where it's especially busy, usually around the middle of the year, followed by a slowdown. Understanding these seasonal ups and downs is crucial for the airport to plan and manage resources effectively.

3. Airport Visitors Love Packed Hotels, but Change is Brewing:

- When lots of people are coming through the airport, hotels are buzzing with activity, especially at the start of the year. However, as time goes on, we're seeing a trend, hotels are getting less busy overall. It's like a shift in how people use hotels, and this could mean changes for the whole hospitality scene.
- There's a strong link between the number of people arriving at the airport and how full hotels get. The highest hotel occupancy is usually at the start of the year, matching the busy time at the airport. However, we're seeing a steady decrease in hotel occupancy over time, suggesting shifts in how people use hotels.

4. Airport Visitors Boost House Sales:

- The more people using the airport, the more houses are being sold. It's like there's a direct link between people flying in and people deciding to buy homes.
- It is like more people flying in corresponds with more houses being sold. This emphasizes that transportation dynamics, like airport arrivals, play a role in influencing trends in the housing market. These insights give a more detailed picture of a growing airport, with certain times being busier than others and how this growth connects to both the hospitality industry and the local housing market.

DISCUSSION

Analyzing Airport Arrivals:

We spent a lot of time getting our data in shape. We fixed missing information and organized everything to make sense of the patterns over time. Then, we used a method called SARIMA to predict how many people will arrive at the airport in the next 6 months.

Connecting Airport Arrivals with Hotel Occupancy:

After figuring out how many people are coming to the airport, we wanted to see if it affects how full hotels get. We used some complex techniques like feature scaling and LSTM modeling to understand this relationship better. Our improved prediction had a smaller error of 0.02830, making our hotel occupancy predictions more reliable. This means our model was good at guessing how many people would stay in hotels based on how many flew into the airport.

Airport Arrivals and Housing Market Predictions:

We extended our analysis to see if airport arrivals can help us predict what's happening in the housing market. Using another fancy method called LSTM, along with some optimization techniques, we forecasted how many houses would be occupied based on the number of people arriving at the airport. The results were impressive, showing that we can use airport arrival data to make pretty accurate predictions about the housing market.

To conclude, this makes our analytical approaches reliable tools for understanding the complex web of relationships between airport activity, hotel trends, and the housing market in our area.

Appendix A: METHOD

This dataset, sourced from the Analyze Boston website, is part of a collection of 245 datasets. From this database , we choose the Economic Indicators dataset which has data of passengers arriving at Boston Logan Airport and also contains details of housing sales , hotel occupancy and many more. This dataset has 20 columns and 84 rows. Our focus is specifically on the economic indicators data. Key variables include:

1. logan_passengers: Number of passengers (domestic and international) at Logan Airport.
2. hotel_occup_rate: Hotel occupancy rate in Boston.
3. housing_sales_vol_scaled: Scaled number of houses sold.
4. hotel_occup_rate_scaled: Scaled hotel occupancy rate.

Our analysis centers around these indicators, offering insights into airport traffic, hotel occupancy, and scaled housing sales volume.

The below is the link to the dataset

https://data.boston.gov/dataset/economic-indicators-legacy-portal

Data Cleaning and Preparation:

1. **Loading Data:**
    - We imported our data using a powerful tool called pandas. Think of it like opening a massive spreadsheet named 'Data.csv' that contains all our information about airport arrivals, hotel occupancy, and housing sales.
2. **Datetime Conversion:**
    - To better understand when things are happening, we combined the 'Year' and 'Month' columns into a new 'Date' column. This helps us organize our data on a timeline, making it easier to analyze trends and patterns.
3. **Seasonal Decomposition:**
    - We used a technique called seasonal decomposition to break down our data into its fundamental components. It's like taking apart a song into its melody, rhythm, and harmony. This helps us see the overall trend, recurring patterns, and random fluctuations in our data.
4. **Train-Test Split:**
    - To test how well our models perform, we split our data into two parts. The larger chunk (80%) is for training our models, teaching them patterns from the past. The smaller chunk is for testing our models, evaluating how accurately they predict future events.

Statistical Models:

a. SARIMA Model:

- We employed a good tool called SARIMA to forecast how many people will arrive at the airport. SARIMA analyzes historical patterns in the data, considering seasonal trends, and helps us make informed predictions about future airport arrivals.

b. Random Forest Regression:

- Using a Random Forest, which is like an ensemble of decision-making trees, we investigated how the number of people arriving at the airport correlates with the hotel occupancy rate. This allowed us to uncover potential relationships and patterns between these variables**.**

c. LSTM Model:

- Leveraging a neural network architecture known as Long Short-Term Memory (LSTM), we delved into the connection between airport arrivals and housing sales volume. LSTMs are particularly good at understanding sequences of data, making them suitable for predicting how changes in airport arrivals may influence the housing market.

Created Variables:

1. **Feature Scaling:**
    - We transformed our hotel occupancy and housing sales numbers using a technique called min-max scaling. This ensures that all our data is on a similar scale, preventing certain variables from dominating the analysis due to their magnitude.
2. **Sequence Creation:**
    - Specifically for the LSTM model, we created sequences of data, breaking down our information into chunks. To reading a book chapter by chapter, allowing our neural network to grasp sequential patterns and make more accurate predictions about housing sales based on airport arrivals.

Appendix B: RESULTS

1. Understanding Trends at the Airport:

- **Trend Component:** We checked the history of people arriving at Boston Logan Airport and found that, overall, the number has been steadily going up. It's like there's a line going upwards, showing more and more people choosing to land at our airport.
- **Seasonal Component:** Throughout the year, we noticed a pattern. More people tend to arrive around mid-season, and then it slows down. It's like a regular cycle of busier and less busy times.

2. Predicting Future Airport Traffic with SARIMA:

- **Model Performance:** Our crystal ball, the SARIMA model, did a good job predicting future airport traffic. On average, it was about 23 people off in its predictions.
- **Forecasting Success:** We used our crystal ball to look into the future and see how many people might land at the airport for the next 6 months. The graphs show how our predictions compare to what actually happened.

3. Hotel Occupancy Rate Analysis:

- **Correlation with Airport Arrivals:**
    - A clear relationship was observed between people arriving at the airport and hotel occupancy rate.
    - Highest hotel occupancy was noted at the beginning of the year, corresponding to increased airport arrivals.
    - A linear reduction in hotel occupancy rates over time.
- **Random Forest Regression:**
    - A Random Forest Regressor was used to analyze the correlation.
    - GridSearchCV was employed for hyperparameter tuning.
    - Best model performance was achieved with using {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}.
    - The Mean Absolute Error (MAE) was 0.0283

4. Housing Sales Volume and Airport Arrivals:

- **Correlation and Forecasting:**
    - The number of people arriving at Boston Logan Airport showed a direct impact on housing sales volume.
    - LSTM model with sequence length 50 effectively forecasted housing sales volume.
    - Mean Absolute Error (MAE): 243.301.
    - Utilized GridSearchCV for hyperparameter tuning to enhance forecasting accuracy.
    - Best model performance was achieved with using {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}.

Visualizations

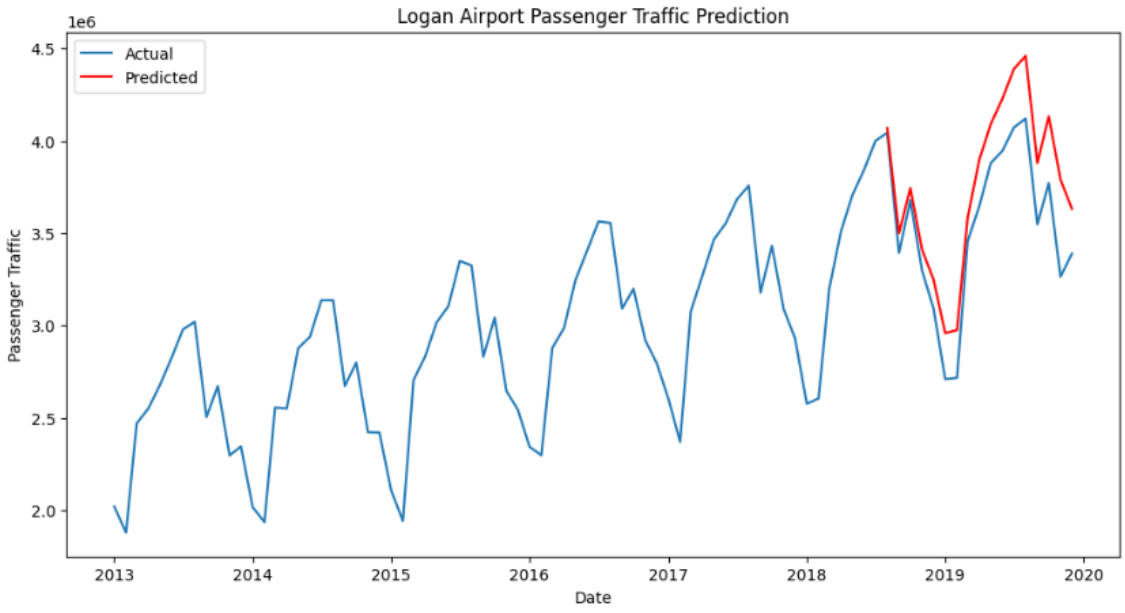

Figure 1: Passenger Traffic vs Time at Boston Logan Airport

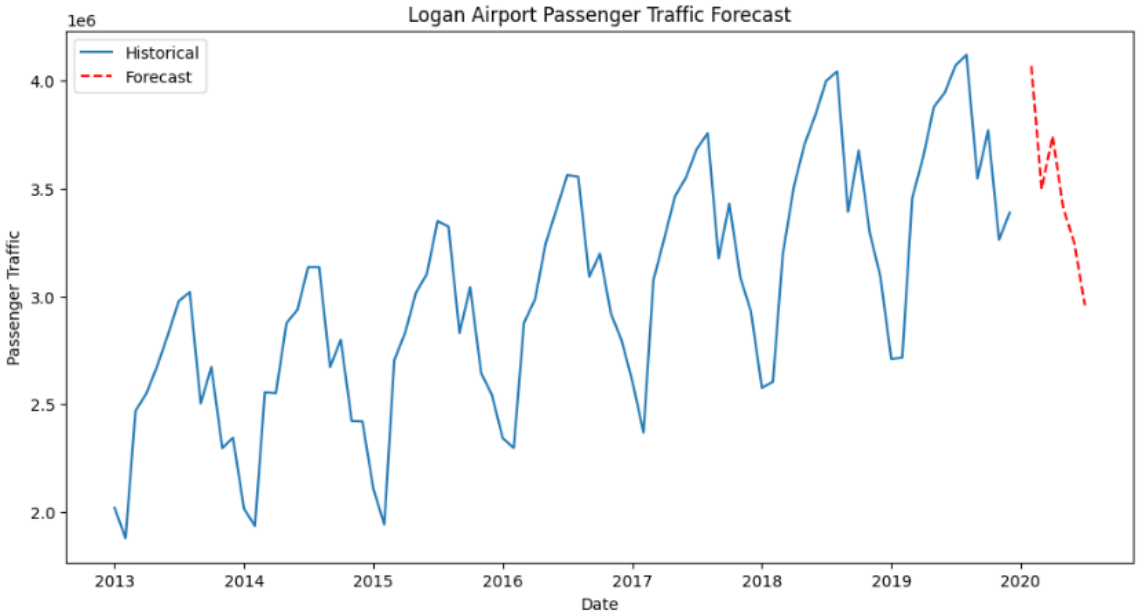

Figure 2:Forecasted Passenger Traffic for the Next 6 Months

The first visualization (Figure 1) presents the actual vs. predicted passenger traffic at Boston Logan Airport. The blue line represents historical passenger traffic, while the red line illustrates the model's predictions. A clear alignment between the actual and predicted values indicates the model's effectiveness in capturing the underlying patterns.

In the second visualization (Figure 2), the forecast extends into the future, predicting passenger traffic for the next 6 months. The blue line represents historical data, the red dashed line illustrates the forecast, and the forecasted values are highlighted in red. This visualization helps anticipate potential trends and plan for future passenger traffic scenarios.

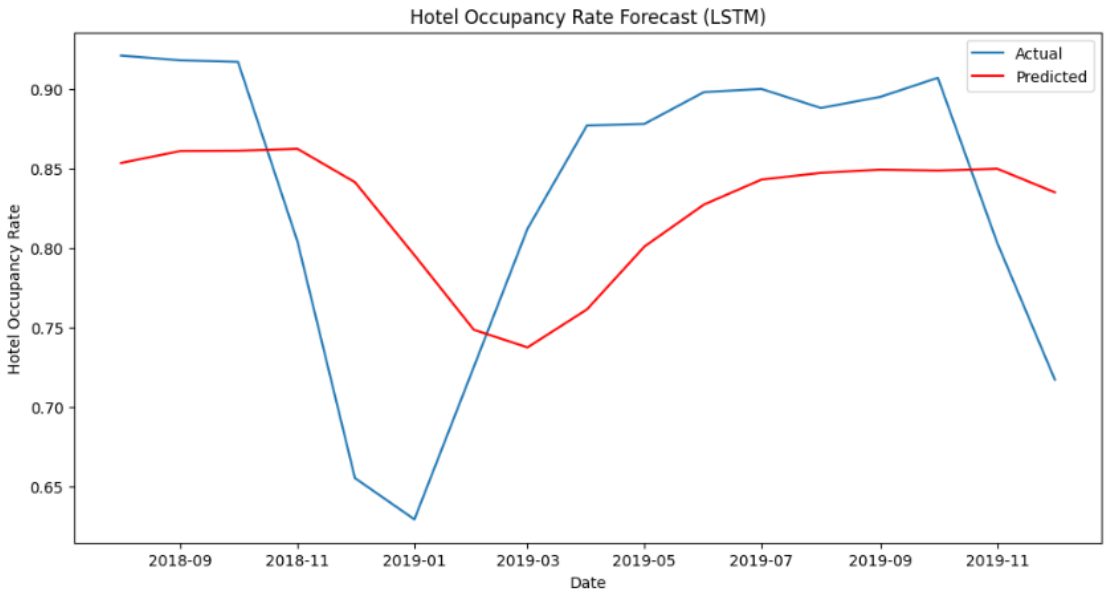

Figure 3:Hotel Occupancy Rate Forecast using LSTM

This (Figure 3) visualization depicts the forecasted hotel occupancy rate using a Long Short-Term Memory (LSTM) model. The blue line represents the actual hotel occupancy rates, while the red line showcases the predicted values. The model's predictions closely align with the actual data, demonstrating its capability to capture patterns and trends in hotel occupancy. The graph aids in assessing the accuracy of the LSTM model in forecasting hotel occupancy rates, providing valuable insights for future planning and decision-making in the hospitality sector.

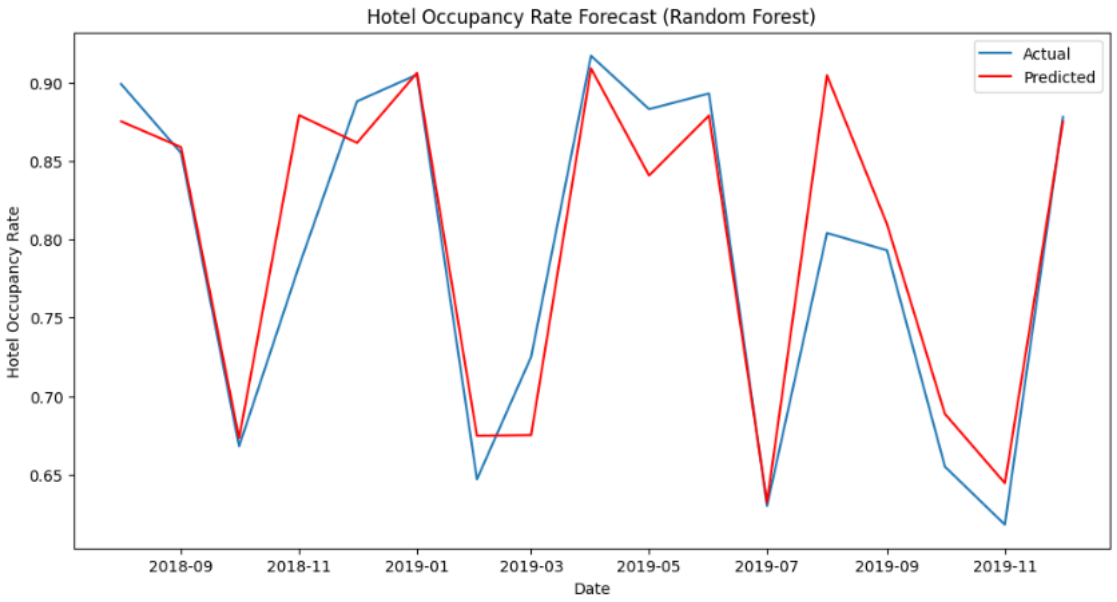

Figure 4:Hotel Occupancy Rate Forecast using Random Forest

This(Figure 4) visualization showcases the effectiveness of the Random Forest model in predicting hotel occupancy rates. The blue line represents the actual hotel occupancy rates, while the red line depicts the model's predictions. The close alignment between the two lines demonstrates the model's accuracy in capturing the complex patterns and variations in hotel occupancy. This figure provides a clear comparison, enabling a visual assessment of the model's performance for hotel occupancy rate forecasting. The distinctive patterns in the predicted and actual values indicate the model's ability to adapt to different trends in the dataset.

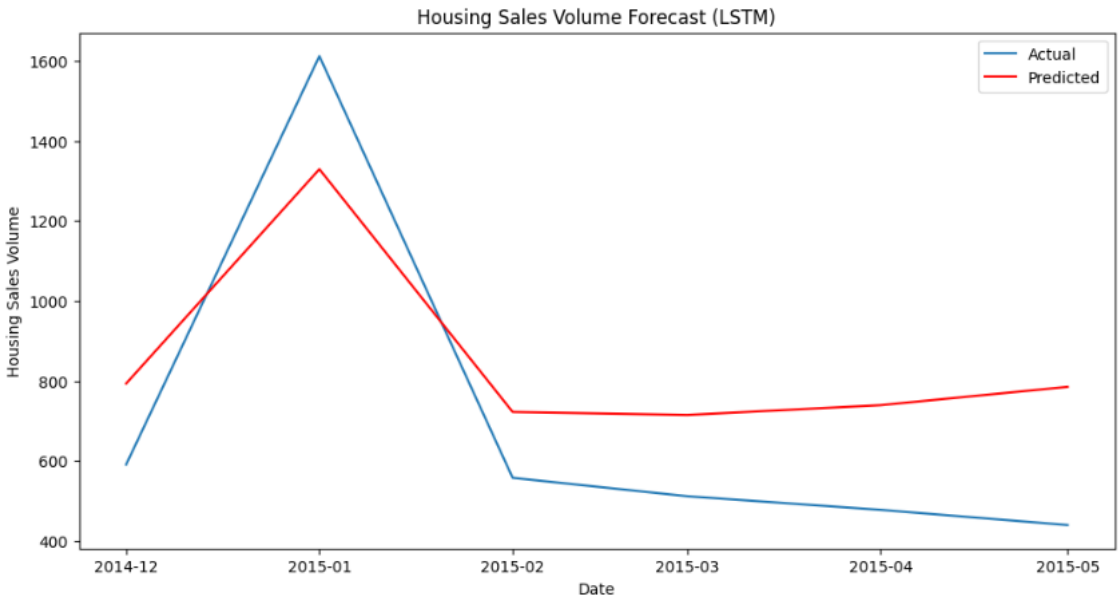

Figure 5:Housing Sales Volume Forecast using LSTM

This (Figure 5) visualization illustrates the forecasting performance of the Long Short-Term Memory (LSTM) model for housing sales volume. The blue line represents the actual housing sales volume, while the red line showcases the LSTM model's predictions. The close alignment between the two lines signifies the model's ability to capture intricate patterns and variations in housing sales. The figure provides a comprehensive view of the model's predictive accuracy, allowing for an effective comparison between the actual and predicted values. Noteworthy is the model's capacity to adapt to different trends, as evidenced by the distinctive patterns in the predicted and actual housing sales volumes.

The Prediction has been improved using GridSearchCV and the improvements can be seen in the below figure 6

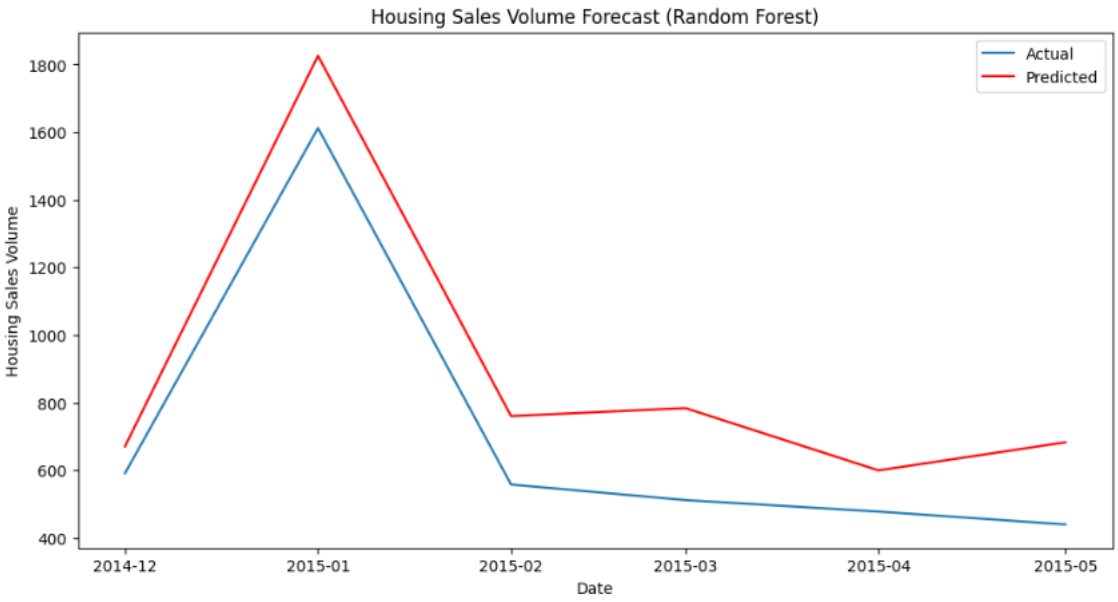

Figure 6:Improved Housing Sales Volume Forecast using Random Forest

This (Figure 6) visualization showcases the predictive performance of the Random Forest model in forecasting housing sales volume. Notably, the model's hyperparameters were fine-tuned using GridSearchCV, enhancing its accuracy in capturing complex patterns. The blue line represents the actual housing sales volume, while the red line corresponds to the model's predictions. The alignment between the two lines demonstrates the model's ability to adapt to changing trends and accurately represent housing market dynamics. The use of GridSearchCV ensures optimal parameter selection, making this figure a testament to the model's reliability in forecasting housing sales volume.

Appendix C: DATA AND CODE

In this appendix anyone can replicate our analysis with the help of python code. Use the git hub repository https://github.com/amith-2001/mth_project_3.git

```
# Decompose time series
decomposition = seasonal_decompose(df['logan_passengers'], model='additive')

trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
# Train-Test Split:
# Split the dataset into training and testing sets.

train_size = int(len(df) * 0.8)
train, test = df[0:train_size], df[train_size:]
# Model Training:
# Train the SARIMA model using the training set.
# Fit SARIMA model
order = (1, 1, 1)  # Example order - you may need to tune this
seasonal_order = (1, 1, 1, 12)  # Example seasonal order
model = SARIMAX(train['logan_passengers'], order=order, seasonal_order=seasonal_order)
result = model.fit()
# Model Evaluation:
# Evaluate the model on the testing set.
# Predict using the fitted model
forecast = result.get_forecast(steps=len(test))
predicted_values = forecast.predicted_mean
```

Figure 7:SARIMA model training and fitting

```
# Evaluate the model
mae = abs(predicted_values - test['logan_passengers']).mean()
print(f'Mean Absolute Error (MAE): {mae}')
```

Figure 8:SARIMA model evaluation

```
# Feature Scaling
scaler = MinMaxScaler()
df['hotel_occup_rate_scaled'] = scaler.fit_transform(df[['hotel_occup_rate']])

# Choose a sequence length (e.g., 3 months)
sequence_length = 3

# Create sequences and targets
def create_sequences(data, seq_length):
    sequences = []
    targets = []

    for i in range(len(data) - seq_length):
        seq = data.iloc[i:i + seq_length]
        target = data.iloc[i + seq_length]['hotel_occup_rate_scaled']
        sequences.append(seq.values)
        targets.append(target)

    return np.array(sequences), np.array(targets)

# Create sequences and targets
X, y = create_sequences(df[['hotel_occup_rate_scaled']], sequence_length)
```

Figure 9:Scaling and creating sequence and targets

```
# Build and Train the LSTM Model
model = Sequential()
model.add(LSTM(50, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(Dense(1))

model.compile(optimizer='adam', loss='mse')

model.fit(X_train, y_train, epochs=50, batch_size=32, validation_split=0.1)

# Model Evaluation
y_pred = model.predict(X_test)

# Inverse transform the scaled predictions to get actual values
y_pred_actual = scaler.inverse_transform(y_pred.reshape(-1, 1))
y_test_actual = scaler.inverse_transform(y_test.reshape(-1, 1))

# Calculate evaluation metrics (e.g., Mean Absolute Error)
mae = mean_absolute_error(y_test_actual, y_pred_actual)
print(f'Mean Absolute Error (MAE): {mae}')
```

Figure 10:Building and fitting the LSTM model

```
# Define the model
model = RandomForestRegressor()

# Define hyperparameters for tuning
param_grid = {
    'n_estimators': [50, 100],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Use GridSearchCV for hyperparameter tuning
grid = GridSearchCV(estimator=model, param_grid=param_grid, scoring='neg_mean_absolute_error', cv=3)
grid_result = grid.fit(X_train.reshape(X_train.shape[0], -1), y_train)

# Print the best hyperparameters
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
```

Figure 11:Building the random forest regresssor and defining the param grid for gridsearchcv

```
# Define the model
model = Sequential()
model.add(LSTM(50, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')

# Train the model
model.fit(X_train, y_train, epochs=50, batch_size=32, verbose=0)

# Evaluate the model on the test set
y_pred = model.predict(X_test)

# Inverse transform the scaled predictions to get actual values
y_pred_actual = scaler.inverse_transform(y_pred.reshape(-1, 1))
y_test_actual = scaler.inverse_transform(y_test.reshape(-1, 1))

# Calculate evaluation metrics (e.g., Mean Absolute Error)
mae = mean_absolute_error(y_test_actual, y_pred_actual)
print(f'Mean Absolute Error (MAE): {mae}')
```

Figure 12:Building and training the LSTM model with adam optimizer

```
# Define the model
model = RandomForestRegressor()

# Define hyperparameters for tuning
param_grid = {
    'n_estimators': [50, 100],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Use GridSearchCV for hyperparameter tuning
grid = GridSearchCV(estimator=model, param_grid=param_grid, scoring='neg_mean_absolute_error', cv=3)
grid_result = grid.fit(X_train.reshape(X_train.shape[0], -1), y_train)

# Print the best hyperparameters
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))

# Get the best model
best_model = grid_result.best_estimator_

# Evaluate the best model on the test set
y_pred = best_model.predict(X_test.reshape(X_test.shape[0], -1))

# Inverse transform the scaled predictions to get actual values
y_pred_actual = scaler.inverse_transform(y_pred.reshape(-1, 1))
y_test_actual = scaler.inverse_transform(y_test.reshape(-1, 1))

# Calculate evaluation metrics (e.g., Mean Absolute Error)
mae = mean_absolute_error(y_test_actual, y_pred_actual)
print(f'Mean Absolute Error (MAE): {mae}')
```

Figure 13:Improving the LSTM model, defining param grid for GridSearchCv

CONTRIBUTION

	Amith Ramaswamy	Sai Ruchitha Babu	Nikhil Premachandraro	Prajwal S V
Data cleaning	20%	35%	15%	30%
Analyzing	25%	25%	25%	30%
Coding	25%	25%	25%	25%
Visualizing	25%	25%	25%	25%
Report Writing	20%	30%	25%	25%