Extra Credit: Temporal Airbnb Seasonality and Modeling Assignment

EAS 510 Basics of AI

Due: December 17, 2025 at 11:59 PM

1 Introduction

Airbnb markets evolve over time. This extra credit project explores temporal dynamics using InsideAirbnb calendar data. You will create a night-level panel dataset, perform seasonality analysis, construct temporal train/validation/test splits, train XGBoost and Neural Network models, log training with TensorBoard, and prepare all results in a GitHub repository.

2 Datasets & Inputs

Download data from InsideAirbnb. For each city, download `listings.csv` and `calendar.csv` for the following specific archived dates:

- **Austin**: March 6, 2025 and December 14, 2024
- **Chicago**: March 11, 2025 and December 18, 2024
- **Santa Cruz**: March 28, 2025 and December 31, 2025
- **Washington, DC**: March 13, 2025 and December 18, 2025

The `calendar.csv` file is a temporal dataset with one row per listing per future date. Key columns include:

- **listing_id**: unique ID of the listing.
- **date**: the calendar date for that row.
- **available**: `t` means available, `f` means booked.
- **price**: string price such as "$95.00".
- **minimum_nights** and **maximum_nights**: booking rules for that date.

3 Part 1: Build the Night-Level Panel Dataset (25 points)

3.1 Goal

Construct a night-level panel dataset where each row represents one listing on one specific date, combining both static listing attributes and temporal calendar fields.

3.2 Required Steps

1. For each city and snapshot date, load the corresponding `listings.csv` and `calendar.csv` files. Show basic evidence such as dataset shapes or a small sample.
2. Merge `calendar.csv` with `listings.csv` on `listing_id` using a left join.
3. Clean and transform key variables:
   - Convert `price` from strings like "$95.00" to numeric.
   - Create an `is_booked` indicator from `available` (`f` = 1, `t` = 0).
   - Parse `date` as a proper datetime.
4. Create the following time-based features:
   - `month` (1–12),
   - `day_of_week` (0 = Monday, ..., 6 = Sunday),
   - `week_of_year` (ISO week number),
   - `is_weekend` (1 for Saturday/Sunday, 0 otherwise),
   - `day_of_year` (1–365).
5. **Optional (recommended for convenience):** Save a *sampled or truncated* version of the prepared panel dataset (e.g. first 100k rows) as a parquet file for quicker reuse during debugging.

   You do not need to save the full city-level dataset to disk.

4 Part 2: Seasonality Analysis (30 points)

4.1 Goal

Explore how prices and booking probabilities change over the year, including seasonal, weekend, and listing-type effects.

4.2 Required Aggregations

Using the prepared panel dataset:

- Compute average **price** by `month`.
- Compute average **booking probability** (`is_booked`) by `month`.

- Compute weekend vs weekday averages for both price and booking probability.
- Compute average price by `month` and a chosen listing-type variable.

4.3 Required Plots

For each city and snapshot date, create:

1. Line plot: average price by month.
2. Line plot: average booking probability (`is_booked`) by month.
3. Bar or grouped bar charts: weekend vs weekday price and booking probability.
4. Line plot: average price by month, grouped by a categorical listing variable such as `room_type`, `property_type`, or `neighbourhood_cleansed` (if the number of categories is moderate).

Label axes clearly and ensure plots have interpretable scales.

Write a short interpretation (3–5 sentences) summarizing the main seasonal insights you observe.

Note: If your analysis reveals minimal seasonal variation, discuss this finding and what it means for predictive modeling. Lack of strong patterns is still valuable.

5 Part 3: Temporal Train/Validation/Test Split (15 points)

5.1 Goal

Create a realistic forecasting setup by training models on earlier dates and testing them on later dates, without temporal leakage.

5.2 Required Steps

1. Build models for *both* prediction targets:
   - **Regression**: predict nightly **price**
   - **Classification**: predict whether a night is **booked** (`is_booked`)

   Compare which target benefits more from temporal modeling and discuss the business implications.
2. Implement a temporal split such as:
   - Training: January through September,
   - Validation: October through November,
   - Test: December through February.
3. For *each* target, construct `X_train`, `y_train`, `X_valid`, `y_valid`, `X_test`, and `y_test`:

- Use listing features (e.g. accommodates, beds, review scores, room type),
- Use time features (month, day_of_week, is_weekend, week_of_year, day_of_year),
- Exclude raw date, IDs, and free-text fields.

6 Part 4: Modeling with XGBoost, Neural Networks, and TensorBoard (40 points)

6.1 Goal

Train XGBoost and Neural Network models for both prediction targets and analyze their behavior and generalization to future months. Use TensorBoard to visualize and interpret training.

6.2 XGBoost Models

- Train *two* XGBoost models: one regressor (for price) and one classifier (for bookings).
- Use the training set and monitor performance on the validation set.
- Compute and record final metrics (RMSE/MAE for price prediction, AUC/accuracy for booking prediction) on the test set.
- Extract and plot feature importances for both models.

6.3 Neural Network Models with TensorBoard

- Build *two* feedforward Neural Networks: one for regression (price) with MSE loss, one for classification (bookings) with binary crossentropy loss.
- Train both models with validation monitoring.
- Use the `TensorBoard` callback to log training:
  - Use separate `log_dir` paths (e.g. `logs/nn_price/DATE-TIME`).
  - Include `TensorBoard` in `callbacks`.
- Evaluate both models on the test set and record performance.

6.4 TensorBoard Screenshots and Interpretation

Students must:

1. Run TensorBoard in the notebook using:

```
%load_ext tensorboard
%tensorboard --logdir logs
```

2. Take screenshots of the key scalar plots (loss and metrics).
3. Insert screenshots into Markdown cells.
4. Write a 5–8 sentence discussion focusing on:
    - Overfitting or underfitting,
    - Training stability,
    - Consistency with final test-set results,
    - Differences between price vs booking prediction.

7 Part 5: Final Write-Up and GitHub Submission (20 points)

7.1 Notebook Write-Up

- The final write-up must appear as Markdown cells at the end of the notebook.
- Summaries should address:
    - Data and seasonality patterns,
    - Comparative model behavior,
    - TensorBoard insights,
    - Generalization to unseen months,
    - Business insights.

7.2 GitHub Repository

- Create a public GitHub repository containing:
    - The final notebook,
    - A `README.md` with instructions,
    - A `requirements.txt`,
    - Any images used (e.g. TensorBoard screenshots).
- Do *not* upload raw InsideAirbnb CSVs; document the download process instead.