Assignment 4: Bakeoff! XGBoost vs Neural Networks Across City Markets

Course: EAS 510 - Basics of Artificial Intelligence
Due Date: December 8, 2024
Points: 80 points

1 Overview

Welcome to the Bakeoff! - a comprehensive showdown between XGBoost (gradient boosting) and Deep Neural Networks for predicting Airbnb listing prices across diverse city markets. This assignment directly connects to Chapter 10: Introduction to Artificial Neural Networks with Keras from our textbook.

You'll conduct a stratified analysis across 12 cities of varying sizes to determine which model approach reigns supreme for price prediction across different market conditions. This isn't just a single-city comparison - it's a full-scale model generalization study that mimics real-world data science challenges.

2 Learning Objectives

By completing this assignment, you will:

1. **Apply Chapter 10 concepts**: Implement neural networks using Keras across multiple market contexts
2. **Stratified model comparison**: Understand how model performance varies across city sizes and market types
3. **Market generalization**: Test model robustness across big, medium, and small city markets
4. **Advanced evaluation**: Compare model performance within and across market segments
5. **Real-world data science**: Handle the complexity of multi-market model validation
6. **Feature importance analysis**: Discover which features matter most in different market contexts

3 Dataset: Inside Airbnb - Multi-City Bakeoff

3.1 The Stratification Strategy

3.1.1 Tier 1: Big Cities (High density, complex markets)

- **New York City, NY** - Global tourism hub

- **Los Angeles, CA** - Entertainment/business center
- **San Francisco, CA** - Tech hub with high prices
- **Chicago, IL** - Major business center

3.1.2 Tier 2: Medium Cities (Regional centers)

- **Austin, TX** - Tech/music scene
- **Seattle, WA** - Tech corridor
- **Denver, CO** - Mountain tourism/business
- **Portland, OR** - Cultural destination

3.1.3 Tier 3: Small Cities (Boutique markets)

- **Asheville, NC** - Mountain tourism
- **Santa Cruz, CA** - Coastal community with seasonal visitors
- **Salem, OR** - State captial
- **Columbus, OH** - College town big on football

3.2 Dataset Structure

For each city, you'll work with:

- `listings.csv`: Property features (bedrooms, price, location, amenities, reviews, etc.)

Target Variable: Price prediction (continuous)
Predict listing price based on property features, location, and amenities

4 Assignment Structure

4.1 What You'll Do

1. Download Data: Get `listings.csv` for each city from Inside Airbnb
2. Build Models: Train XGBoost and Neural Network models to predict listing prices (experiment with at least 2 different Neural Network architectures)
3. Compare Performance: Train models for each tier using ALL cities in that tier. Build a big city Neural Network model and XGBoost model using all 4 big cities combined, then same for medium cities (all 4 combined), and small cities (all 4 combined). Which model works better for which city tiers?

4.2 Required Features

Base Features (must use):

```
numeric_columns = ['accommodates', 'bedrooms', 'beds', 'bathrooms_text',
                   'review_scores_rating', 'review_scores_accuracy',
                   'review_scores_cleanliness', 'review_scores_checkin',
                   'review_scores_communication', 'review_scores_location'
                      ,
                   'review_scores_value', 'number_of_reviews',
                   'availability_365', 'minimum_nights', 'maximum_nights']
```

Feature Engineering Requirement: Create at least 4 additional features

For clarity, your notebooks must indicate 4 additional features. These may be features directly from the data sets. Or they could be ones that you engineered! It is up to you. But your selection of 4 (or more if you want) must be clear and included in the analysis.

Examples:

- `price_per_bedroom` = accommodates / bedrooms
- `avg_review_score` = average of all review scores
- `is_entire_home` = encoded room_type
- `amenities_count` = count of amenities
- `property_type` = directly from dataset
- `neighbourhood_cleansed` = directly from dataset

4.3 Target Variable

Predict: `price` (remove from features, use as target)

5 Deliverables

5.1 Single Submission

Upload a text file to UBLearns containing your GitHub repository link.

Note: The goal of using GitHub is that you can refer to this project on your resume as a portfolio piece demonstrating your data science skills.

5.2 GitHub Repository Must Contain

1. **README.md**
   - Project overview and objectives
   - Instructions to run the code
   - Data table with exact months used for each city:

City	Tier	Data Month	Listings Count
NYC	Big	Sept 2024	45,000
Austin	Medium	Oct 2024	12,000
etc...			

- Major findings of XGBoost vs Neural Network comparison

2. **Well-Documented Google Colab Notebook**
   - Clear markdown explanations for each step
   - Data loading and preprocessing
   - Feature engineering (minimum 4 additional features)
   - Model training and evaluation
   - Performance comparison across tiers
   - Conclusions and insights
3. You should have reproducibility instructions that explain how to go to the Airbnb site and download the files

5.3 Repository Settings

- Private repos welcome: Invite professor as collaborator
- Public repos: Also acceptable (or make private repos public after the submission date)
- Clean structure: Organized files and folders

6 Technical Requirements

6.1 Required Libraries

```
# Data processing
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Machine Learning
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error,
    r2_score
from sklearn.preprocessing import StandardScaler, LabelEncoder
import shap

# Neural Networks (Chapter 10)
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
```

6.2 Model Requirements

Note: The configurations below are provided as examples and are not the configurations you should use. You should experiment with different hyperparameters to optimize performance. For neural networks, you must experiment with at least 2 different architectures (different layer configurations).

XGBoost Model:

```
xgb_model = xgb.XGBRegressor(
    n_estimators=100,
    learning_rate=0.1,
    max_depth=6,
    random_state=42
)
```

Neural Network (Chapter 10):

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64, activation='relu', input_shape=(n_features,)
        ),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(1)  # Price prediction output
])

model.compile(optimizer='adam', loss='mse', metrics=['mae'])
```

7 Data Sources

Data Download:

1. Visit Inside Airbnb
2. Download `listings.csv` for cities from each tier:
    - Big: New York City, Los Angeles, San Francisco, Chicago
    - Medium: Austin, Seattle, Denver, Portland
    - Small: Asheville, Santa Cruz, Salem, Columbus
3. Start with 3 cities (1 from each tier) then expand

8 Evaluation Metrics

8.1 Stratified Model Performance (60 points)

- **Individual City Performance (20 points)**: RMSE, MAE, Rš for each of the 12 cities with clear analysis of which is better - XGBoost or Neural Networks
- **Composite Tier Analysis (20 points)**: Train models for each tier using ALL cities in that tier. Build a big city Neural Network model and XGBoost model using all 4 big cities combined, then same for medium cities (all 4 combined), and small cities (all 4 combined). Compare performance across big/medium/small city tiers with clear analysis of which is better - XGBoost or Neural Networks

- **Cross-Tier Neural Network Analysis (20 points)**: Use the composite big tier Neural Network model to predict medium and small city prices, the medium tier Neural Network model to predict big and small city prices, and the small tier Neural Network model to predict big and medium city prices. Analyze how well models generalize across different market tiers

8.2 Repository Implementation (20 points)

- **Code Quality (5 points)**: Clean, well-commented, reproducible code
- **Data Pipeline (5 points)**: Robust multi-city preprocessing framework
- **Visualization (5 points)**: Clear charts comparing performance across tiers
- **README.md (5 points)**: Well-structured README.md that includes a discussion of the overall results

9 Grading Rubric

Component	**Points**	**Criteria**
Individual City Performance	20	RMSE, MAE, Rš for each of the 12 cities with clear analysis of which is better - XGBoost or Neural Networks
Composite Tier Analysis	20	Train models for each tier using ALL cities combined. Compare performance across tiers with clear analysis
Cross-Tier Neural Network Analysis	20	Use tier models to predict across different market tiers. Analyze generalization capabilities
Code Quality	5	Clean, well-commented, reproducible code
Data Pipeline	5	Robust multi-city preprocessing framework
Visualization	5	Clear charts comparing performance across tiers
README.md	5	Well-structured README.md that includes discussion of overall results

10 The "- to + Challenge" (Extra Credit)

Students who want to improve their final grade should consider this challenge. It is not easy and it would require a more detailed writeup. But give it a read and consider it.

The Temporal Bakeoff Challenge - adding the dimension of time to your analysis. Consider Columns Ohio: its price points change dramatically over time. In the summer and autumn, it attracts many folks who want to see Ohio State play football. In winter and spring, not so much. Though rentals may see a spike in populatirty at graduation time.

Reward: Full grade level increase (A- $\rightarrow$ A+, B+ $\rightarrow$ A-, etc.)
Details: See `Assignment4_ExtraCredit_Temporal.md`

This advanced challenge involves temporal feature engineering, multi-context training, and strategic insights about when models work best in different seasonal contexts.

11 Tips

11.1 Appropriate Neural Network Architecture for Tabular Data

For this assignment, you will build Deep Neural Networks (not Convolutional Neural Networks) for tabular data. Convolutional Neural Networks are designed for image and sequence data with spatial structure. Since Airbnb features are independent numeric and categorical variables with no spatial relationships, you should use Dense (fully connected) layers to build Multi-Layer Perceptrons (MLPs).

A suitable architecture for tabular data includes:

- An input layer equal to the number of features
- Several Dense (fully connected) layers
- Nonlinear activations such as ReLU
- Optional regularization (dropout or L2)
- A single-unit output layer for regression

This structure allows the model to learn nonlinear relationships between features without assuming the spatial locality that true convolutions require. Convolutional layers should not be used here; Dense layers are the correct architecture for tabular data.

A typical tabular-appropriate architecture might look like:

```
model = keras.Sequential([
    layers.Dense(128, activation='relu', input_shape=(n_features,)),
    layers.Dense(64, activation='relu'),
    layers.Dense(32, activation='relu'),
    layers.Dense(1)
])
```

Students may deepen or widen the network, but the key point is that Neural Networks for tabular data should be implemented as Dense-layer MLPs, not as spatial convolutions.

12 Resources

- **Inside Airbnb Data**: http://insideairbnb.com/get-the-data.html
- **Chapter 10**: Introduction to Artificial Neural Networks with Keras (textbook)
- **XGBoost Documentation**: https://xgboost.readthedocs.io/
- **Keras/TensorFlow**: https://www.tensorflow.org/guide/keras

13 Questions?

Post questions in the course discussion forum or attend office hours. This is a complex, real-world analysis - start early and focus on building insights!