深度學習 Lab 1 結案報告：倫敦房價預測

---

1. Key concept：資料前處理

本次目標是使用 266,325 筆訓練樣本預測房價。神經網路的成功極度依賴高品質的輸入資料，因此執行了包含 6 個關鍵步驟的資料前處理流程。

1.1 資料探索與異常值處理

- 問題：透過視覺化分析，我發現 floorAreaSqM（面積）和 price（價格）呈現極端右偏分布，且 bedrooms 與 bathrooms 存在不合理的極端值（如 9 房）。
- 解決方案：
    1. 特徵裁剪 **(Clipping)**：我對 floorAreaSqM、bedrooms 和 bathrooms 進行雙重截斷 。首先使用 .clip() 設定合理的硬性上下限（如面積 20-250），然後使用 1%-99% 的分位數 (.quantile()) 進行 Winsorization，移除統計上的極端值並穩定分布 。
    2. 目標轉換 **(Target Transform)**：由於 price 高度偏斜，我使用 y_log = np.log1p(y) 進行對數轉換。這使得目標分布更接近常態，能極大穩定神經網路的損失函數計算。

1.2 關鍵特徵工程 (19 → 37 特徵)

使用原始特徵，還創造了 18 個新的工程特徵 來捕捉複雜的非線性關係。

- 地理空間特徵
    - 問題：房價與「地點」高度相關，但經緯度本身是線性特徵，無法捕捉「區域效應」。
    - 技術：
        1. **KMeans** 聚類：使用 KMeans(n_clusters=12) 將經緯度分為 12 個地理群集，生成 geo_cluster 特徵。
        2. **BallTree** 密度分析：使用 BallTree 搭配 haversine (地球球面) 距離，快速計算每個房產 3km 半徑內的鄰居數量，生成 geo_density 特徵 。
    - 效果：如附件03中的圖顯示了此方法成功捕捉到倫敦市中心的高密度區域，為模型提供了空間特徵。
- 交互/時間/分箱 特徵
    - 時間特徵：house_age (屋齡)、is_new_house。
    - 組合特徵：total_rooms (臥室+浴室)、area_per_bedroom (每房面積)。
    - 分箱特徵：將 house_age 和 floorAreaSqM 進行分箱 (Binning)，將連續特徵轉換為類別特徵，幫助模型學習非線性響應。

1.3 最終準備：缺失值與標準化

- 缺失值：數值型特徵（如 floorAreaSqM）的缺失值使用中位數 **(median)** 填補，此方法對離群值不敏感。類別型特徵使用 LabelEncoder 轉換為數值。
- 標準化 **(Standardization)**：
    - 技術：使用 StandardScaler 將所有 37 個特徵 轉換為均值為 0、標準差為 1 的分布 。
    - 必要性：此步驟對神經網路至關重要。它確保了所有特徵都在同一尺度上，防止梯度下降被高數值特徵（如 floorAreaSqM）主導，從而加速並穩定模型收斂。

2. Key concept：模型架構

2.1 核心組件 (BatchNorm + SiLU + Dropout)

網路中的每個隱藏層區塊 (Block) 均由以下三個關鍵組件構成：

1. **nn.BatchNorm1d (批次標準化)**：在 6 層的深度網路中，BatchNorm 透過標準化每層的輸出，極大地加速了收斂並防止梯度消失/爆炸，使深度訓練成為可能。
2. **nn.SiLU (Sigmoid Linear Unit)**：
    - 為何不用 **ReLU**？ ReLU 在 $x < 0$ 時梯度為 0，容易導致「神經元死亡」。
    - **SiLU** 優勢 (SiLU 是一個平滑、非單調的激活函數，它在所有點上均可微，允許梯度在負值區流動，避免了神經元死亡。
3. **nn.Dropout (隨機失活)**：作為核心正則化手段，在訓練時隨機關閉 10%-30% 的神經元，強迫網路學習更穩健的特徵表示，有效防止過度擬合。

2.2 損失函數 (AdvancedLoss)

- 問題：房價預測存在極端值（豪宅），標準 MSE 損失會被這些離群值主導。
- 解決方案：查找資料後設計了一個結合 Huber Loss 和 L1 Loss 的自定義損失函數。
    - F.huber_loss(delta=1.0)：對誤差小的樣本使用 MSE（平滑），對誤差大的樣本使用 MAE（穩健）。
    - L1 (MAE) 正則：額外加入 5% 的 L1 懲罰，進一步增強模型對離群值的穩健性。

3. Key concept：模型訓練

採用了 K-Fold 交叉驗證策略來訓練模型並確保其泛化能力。

3.1 訓練策略 (K-Fold + Early Stopping)

- **K-Fold (K=5)**：不只訓練單一模型，而是將 26 萬筆資料分為 5 折，訓練 5 個獨立的模型。此舉可最大化數據利用率，並提供對模型泛化能力（OOF 分數）的可靠評估。
- **Early Stopping (雙指標, Patience=30)**：為節省時間並防止過擬合，同時監控 Val Loss 和 Val R^2。若兩者連續 30 個 epochs 均未改善，則停止該折訓練。顯示，所有 5 折均在此策略下提早停止。

3.2 訓練效率技術 (AMP + 梯度累積)

- 自動混合精度 **(AMP)**：使用 torch.cuda.amp.autocast 和 GradScaler，利用 GPU 的 Tensor Cores 進行 FP16 運算。這使訓練速度提升，並節省顯存，且未損失精度。
- 梯度累積 **(steps=4)**：每 4 個 mini-batch 更新一次模型權重。這等效於使用 4 倍的批次大小 (256 * 4 = 1024)，使梯度估計更準確，訓練過程更穩定。

3.3 優化器 (AdamW + Cosine Annealing)

- **AdamW**：取代傳統 Adam，能更有效地處理權重衰減 (Weight Decay)。
- **CosineAnnealingWarmRestarts**：我們使用餘弦退火學習率調度器。它使學習率在訓練過程中週期性地下降和「重啟」(Warm Restart)，能有效幫助模型跳出局部最小值，找到更優的解。

4. 評估與推論

- **OOF (Out-of-Fold)** 評估：使用 OOF 預測（對每個樣本使用未訓練過它的模型進行預測）來評估模型的真實泛化能力。
    - 視覺化分析顯示，OOF 預測與真實值高度相關（點集中在對角線），且殘差分布集中於 0，表明模型準確且無系統性偏差。
- 推論 **(Inference)**：最終提交的 submission.csv 並非來自單一模型，而是 5 個 K-Fold 模型的集成平均值（在 log 空間取平均後再 expm1 還原）。此方法比單一模型更穩健，泛化能力更強。