Impact Score: What is it?

- Impact score intended to quantify how much an external event (concert, news, flight surge, holiday announcement) is likely to affect hotel demand/revenue relative to baseline.
- **Ranking / regression / scoring**: given features about the event + context, output a numeric score (or rank) of "impact."
- Historical labeled events with observed "impact" (e.g. % deviation in occupancy or revenue)

Considerations / Steps

Feature Engineering (most important)

- Event features: type (concert, sports, convention), size, distance to hotel, date, promotion, news sentiment
- Hotel context: baseline demand, price sensitivity, occupancy trend, capacity
- Competing events, flight volume, calendar (holiday/weekend)
- Lag features: prior week/month deviations, seasonal baselines

Label Construction / Ground Truth

- Define "impact" metric: % deviation of occupancy, ADR, RevPAR vs baseline
- Possibly discretize (low, medium, high) or continuous score.

Model Selection & Validation

- Split data (e.g. time-based split)
- Cross-validation
- Evaluate with regression metrics (RMSE, MAE) and ranking metrics (nDCG, Precision@K)

Interpretability & Monitoring

- Feature importance or SHAP to explain why a certain event got a high score
- Monitor model drift or input features becoming stale

Scalability & Inference Speed

- Because system scrapes and scores 4x daily, inference should be fast
- Optimize for low latency, batch scoring, and model size.

Possible Models

Gradient Boosted Trees (XGBoost, LightGBM, CatBoost)

What it is / how it works

- Boosting ensemble methods, where new decision trees are added sequentially to correct errors of prior trees.
- XGBoost is a widely used, optimized library; LightGBM and CatBoost are variants optimized for speed, large data, categorical variables, etc.
- Works very well on structured/tabular data

Strengths / pros

- High predictive performance in practice, especially with properly engineered features.
- Handles nonlinear relationships and interactions (e.g. how event size * distance from hotel interacts).
- Built-in regularization (avoid overfitting).
- Feature importance makes it somewhat interpretable (can see which event features drive scores).
- Works with missing data and mixed feature types (numerical + categorical).

Weaknesses / cons or challenges

- Requires careful hyperparameter tuning.
- Can overfit if data doesn't strongly correlate or is noisy.
- Slower training/inference than simpler models.
- Less transparent in extreme edge cases

When to use

- If you have a lot of labeled data (many events with measured outcomes).
- Looking for high performance and can afford tuning.
- Have many features (weather, distance, prior event history, competing events, flights, searches) and interactions.

Comparisons

- XGBoost often outperforms Random Forest in classification/regression tasks when tuned.
- But with default parameters or small data, Random Forest can be more stable.
- LightGBM and CatBoost are variants

Random Forest

What it is / how it works

- Compilation of decision trees, built in parallel. Each tree is trained on a random subset of data and features (bagging).
- Final prediction is typically the average (for regression) or majority vote (for classification).

Strengths / pros

- Robust, less prone to overfitting than single decision tree.
- Works reasonably “out of the box” with less hyperparameter tuning.
- More interpretable than boosting.
- Good baseline model for comparison.

Weaknesses / cons

- May not reach the predictive performance ceiling that boosting methods can.
- Doesn’t inherently correct mistakes from prior trees (no sequential boosting).
- Less sensitive to small signals unless many/deep trees.

When to use

- If you have limited data or want a strong baseline.
- Interpretability or robustness is important.
- Fallback or sanity check model.

Neural Networks / Deep Learning

What it is / how it works

- Feed-forward networks, multi-layer perceptrons, possibly embedding layers for categorical features.
- Could also explore learning-to-rank / ranking neural nets if seeking ranking rather than regression.

Strengths / pros

- Capture highly nonlinear relationships and complex interactions.
- Flexible architecture (layers, embeddings, dropout, etc.).
- Large datasets and many features.

Weaknesses / cons

- Requires more data and tuning.
- Harder to interpret.
- Can overfit if not regularized.
- Requires more computational resources.

When to use

- Later stage, with large event-level datasets.
- If boosting models flatten out in performance and need more capacity.

Learning-to-Rank Models / Ranking Algorithms

- **Pointwise ranking**: Treat each event as a data point with ground-truth impact, predict a regression score.
- **Pairwise ranking**: The model learns which event is more impactful in pairs (like logistic loss comparing two events).
- **Listwise ranking**: The model optimizes a loss over the whole list (e.g. nDCG, ranking metrics)—better alignment with ranking goals.

These methods are used in recommendation systems. Ranking models require careful design of loss functions to align model training with real-world impact.

Hybrid / Ensemble Approaches & Meta-models

- Could combine multiple models (e.g. average of XGBoost + Random Forest + neural) for robustness.
- Use meta-labeling or stacking: one model predicts likelihood of high impact, another refines it.
- Hierarchical scoring metrics (class-taxonomy-aware scoring) to give partial credit to less severe errors.