

Chapter 5: Feature Engineering

Group 5 (Rec5ys)

Saturday, May 11, 2024



<https://www.youtube.com/watch?v=hPicVDdMbYM>

Chapter 5

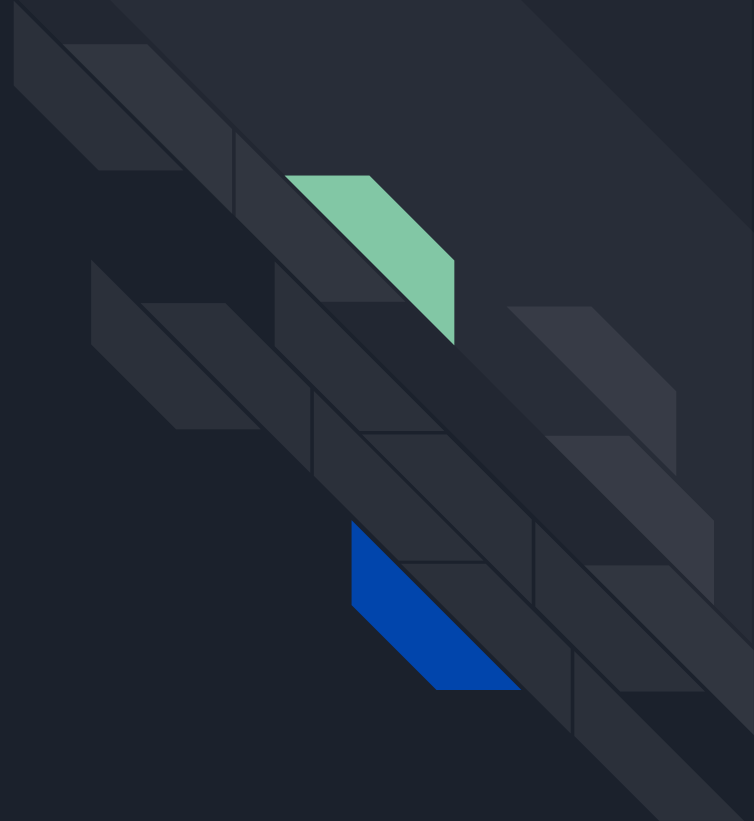
Feature Engineering

Rec5ys



Meet our group!

- Md. Aminul Islam
- Rohan Singh Rajput
 - Sr. Machine Learning Engineer @ Headspace, Los Angeles, CA
- Olawale Moses Onabola
- Sachin
 - Lead Data Scientist @ Target Corporation, Bengaluru
- K Hegde
 - Machine Learning Engineer; Bay Area, CA





Agenda

- Introduction
- Some common operations
- Data leakage
- Engineering good features
- Summary




What is a feature?

- Representation of data that is learned by the model
 - These are automatically extracted from the data
 - Could be relatively less interpretable
 - Think neural networks
- Engineered from data that is calculated by the human
 - Could be with the help of subject matter experts
 - Could be simple aggregations or something much more nuanced
 - Considered to be an “art” by many




Why is feature engineering important?

- Some of the ways to improve performance once a baseline is established:
 - Hyper-parameter tuning
 - Feature engineering
- Feature engineering typically gives the biggest performance boost compared to algorithmic techniques
- If the features are not great then even the best model can perform poorly



Some things to keep in mind when engineering features

- Handling missing values
 - Think carefully what missing data signifies
 - Imputation should ensure that there is no data leakage (more on this later)
- Scaling
 - Avoid inadvertently weighting the data
 - May not be a one-time thing; data shift can occur which could require re-scaling
- Discretization
 - Also called bucketization; not very common in practice



Some things to keep in mind when engineering features

- Encoding dynamic (non-static) categorical features
 - Use the hashing trick
- Feature crossing
 - For learning non-linear relationships
 - Cons: Feature space can blow up and model can overfit
- Handling positional embeddings
 - Dynamic embeddings can be learned
 - Fixed embeddings as a special case of Fourier features



Data leakage

- Handling time-correlated data
 - Split by time; not randomly
- Applying data statistics the right way
 - Split before scale, impute and handle duplicates before split
- Leakage from the data generation process



Engineering good features

- Too many features can cause data leakage, overfitting, tech debt and high inference latency
- Occam's razor - Simpler model with similar performance is always better
- Feature engineering for deep NN architectures - even deep cross nets use interaction features
- Feature engineering for hour of the day, day of week - use cyclical features



Feature Importance and Generalization

- Built-in feature importances in XGBoost, Random forest. These may suffer from following flaws:
 - May favor high cardinality features
 - Computed in statistics derived from training data and not held out data
- Permutation feature importance doesn't suffer from these flaws and is model agnostic
- Other model agnostic methods - SHAP, LIME (gives feature importances for each data point)
- Feature generalization aspects - feature coverage, distribution



Summary

- Feature engineering can be a source of big performance boost
- To extract the most value, some pre-processing may be required
- Analyzing feature importance can give valuable insights into the model
- Neural networks are good at engineering useful features but it is always a good idea to engineer features from domain knowledge [saves training time, model size and complexity]



Thanks!

