Data Engineering and MLOps in Business

Data in MLOps: From Collection to Feature Engineering

Primoz Konda

AAUBS

March 19, 2025

pk@business.aau.dk

Outline

Where did we end yesterday?

- ?
- Questions?

Why Data Collection Matters

- Data is the foundation of machine learning models.
- Quality and quantity of data influence model performance.
- Garbage in, garbage out.

Types of Data

- **Structured** – Databases, CSV files
- **Semi-structured** – JSON, XML
- **Unstructured** – Text, Images, Audio
- **Streaming vs Batch** – Real-time vs periodic processing

Data Sources

- APIs (e.g., REST, GraphQL)
- Web Scraping
- Logs and Sensor Data
- Public Datasets (Kaggle, UCI)

Challenges in Data Collection

- Data quality (missing, corrupted data)
- Volume and variety of data
- Privacy and legal issues (GDPR)
- Availability of data over time
- Consistency across data sources

Why Feature Selection Matters

- Reduces model complexity.
- Improves model performance and training time.
- Prevents overfitting.

Feature Selection Methods

- **Filter Methods** – Correlation, Mutual Information
- **Wrapper Methods** – Recursive Feature Elimination (RFE)
- **Embedded Methods** – Lasso Regression, Tree-based models
- **Permutation Methods** – Measures drop in performance when a feature's values are randomly shuffled

Filter Methods

- **Definition:** Filter methods select features based on statistical properties of the data, independent of any learning algorithm.
- **Examples:**
  - **Correlation Coefficient** – If you are predicting house prices, features like square footage and number of rooms are likely highly correlated with the target variable (price). A correlation analysis can help eliminate redundant features like the total floor area if it correlates strongly with square footage.
  - **Mutual Information** – In a spam detection model, the presence of certain words (like "win" or "free") might have high mutual information with the target variable (spam vs not spam). Mutual information helps select the most informative words.

Wrapper Methods

- **Definition:** Wrapper methods use a machine learning model to evaluate feature subsets and select the best combination.
- **Examples:**
  - **Recursive Feature Elimination (RFE)** – In a credit scoring model, you might have many financial indicators (e.g., income, debt-to-income ratio, number of late payments). RFE would iteratively test combinations of these features, removing the least impactful ones until only the most predictive features remain.
- **Real-Life Example:** In healthcare, to predict heart disease, RFE might test different subsets of features like cholesterol levels, blood pressure, and BMI, and eliminate the least important ones to improve model accuracy.

Embedded Methods

- **Definition:** Embedded methods perform feature selection during model training by penalizing irrelevant features.
- **Examples:**
  - **Lasso Regression** – In predicting house prices, if features like the number of bathrooms or the size of the garage have coefficients close to zero, Lasso will shrink them to zero and effectively remove them from the model.
  - **Tree-based Models** – In a customer churn model, a random forest might find that customer tenure and number of complaints are highly important, but discount offers received are not. The tree structure automatically adjusts for the most relevant features.

Permutation Methods

- **Definition:** Permutation methods measure the importance of a feature by evaluating the decrease in model performance when the feature values are randomly shuffled.
- **How It Works:**
  - Train a model on the original data.
  - Shuffle the values of one feature at a time.
  - Measure the drop in performance (e.g., accuracy or R^2) after each shuffle.

Practical Considerations in Feature Selection

- **Cost of Data Collection** – Some features might be highly predictive but expensive to collect.
  - Example: In predicting loan defaults, getting detailed financial records from credit agencies might be costly.
- **Availability Over Time** – A feature might be available during training but not consistently available at prediction time.
  - Example: A stock market model might use daily trading volume during training, but this data could be delayed during live prediction.
- **Data Drift and Concept Drift** – The relationship between the feature and the target variable might change over time.
  - Example: Consumer behavior data might change due to seasonality or economic shifts.

Practical Considerations in Feature Selection (2)

- **Feature Leakage** – Some features might contain future information that wouldn't be available at the time of prediction.
  - Example: Including "total hospital bill" to predict length of stay would be leakage since the bill is only known after discharge.
- **Ethical and Legal Risks** – Certain features might create bias or violate privacy regulations.
  - Example: Using ZIP codes for credit scoring might introduce racial or socioeconomic bias.

Practical Considerations in Feature Selection (3)

- **Computational Complexity** – Some features might increase training time without significant performance gains.
  - Example: Using high-dimensional image embeddings for a text classification task might add unnecessary complexity.
- **Interpretability** – Features that improve accuracy might make the model harder to interpret.
  - Example: Neural network embeddings might outperform simpler models but be less interpretable.

Why Feature Transformation Matters

- Improves model convergence and performance.
- Prepares data for gradient-based methods.
- Handles categorical and missing data.

Common Feature Transformations

- **Scaling** – Min-max, z-score
- **Encoding** – One-hot, target encoding
- **Dimensionality Reduction** – PCA, t-SNE
- **Handling Skewness** – Log transformation

Challenges in Feature Transformation

- Overfitting with too many features
- Loss of interpretability
- ...

Data Format Issues

- **Unstructured vs Structured Data** – LLMs handle text better than structured formats (e.g., tables, JSON).
- **Multimodal Data** – LLMs struggle with mixed inputs like text + images or tables.
- **Code and Mathematical Expressions** – Mathematical notation and complex code can be misinterpreted.
- **Example:** Summarizing a research paper with figures and equations may result in missing critical insights.

Context Length Limitations

- **Context Truncation** – Exceeding the token limit leads to truncation and data loss.
- **Loss of Global Context** – Long inputs may cause the model to lose coherence across segments.
- **Example:** Summarizing a legal document longer than 32K tokens may result in incomplete analysis.

Data Structure Misinterpretation

- **Table Misreading** – LLMs misalign rows and columns when processing tables.
- **List and Hierarchy Confusion** – Nested lists and bullet points may be flattened.
- **Example:** Extracting information from a CSV file with misaligned columns can produce incorrect output.

Data Transformation Issues

- **Encoding and Character Sets** – Special characters and encodings might confuse the model.
- **Language Switching** – Code-switching between languages can lead to misinterpretation.
- **Example:** A Danish-English customer support query may be misinterpreted.

Domain-Specific Knowledge Gaps

- **Specialized Jargon** – LLMs trained on general data may struggle with technical language.
- **Incomplete Fine-Tuning** – Lack of domain-specific data limits accuracy.
- **Example:** Predicting medical outcomes with general LLMs may miss medical terminology nuances.

Output Formatting and Alignment

- **Mismatch Between Input and Output** – Model might ignore formatting instructions.
- **Inconsistent Outputs** – Non-deterministic token sampling can lead to varied outputs.
- **Example:** Formatting Markdown tables inconsistently across different outputs.

Privacy and Data Leakage

- **Sensitive Data Handling** – LLMs may output memorized sensitive data.
- **Prompt Injection Attacks** – Malicious inputs can manipulate the model's response.
- **Example:** An LLM repeating sensitive company data when prompted maliciously.

Multi-Turn Conversations and Agents

- **Loss of Memory Across Turns** – Models without state memory lose context.
- **Consistency Issues** – Multi-turn responses might contradict previous statements.
- **Example:** A customer service agent forgetting previous user questions.

How to Fix Common Challenges

- **Table misinterpretation** – Convert tables to structured JSON format.
- **Context limitations** – Split long inputs and merge summaries.
- **Sensitive data exposure** – Mask sensitive fields before input.
- **Loss of context in conversations** – Use memory-aware agents (e.g., Langchain).
- **Hallucinations** – Use RAG (Retrieval-Augmented Generation) to ground facts.

End-to-End Workflow

1. Automate data collection (APIs, scraping)
2. Store raw data (Databases, Data Lakes)
3. Automate feature selection (GitHub Actions, Prefect)
4. Track transformations (MLFlow, Weights & Biases)

Example Tools

- Data Collection – Scrapy, BeautifulSoup
- Feature Selection – Scikit-learn, SHAP
- Feature Transformation – Pandas, Numpy
- Automation – Airflow, Prefect

Q & A

Questions?