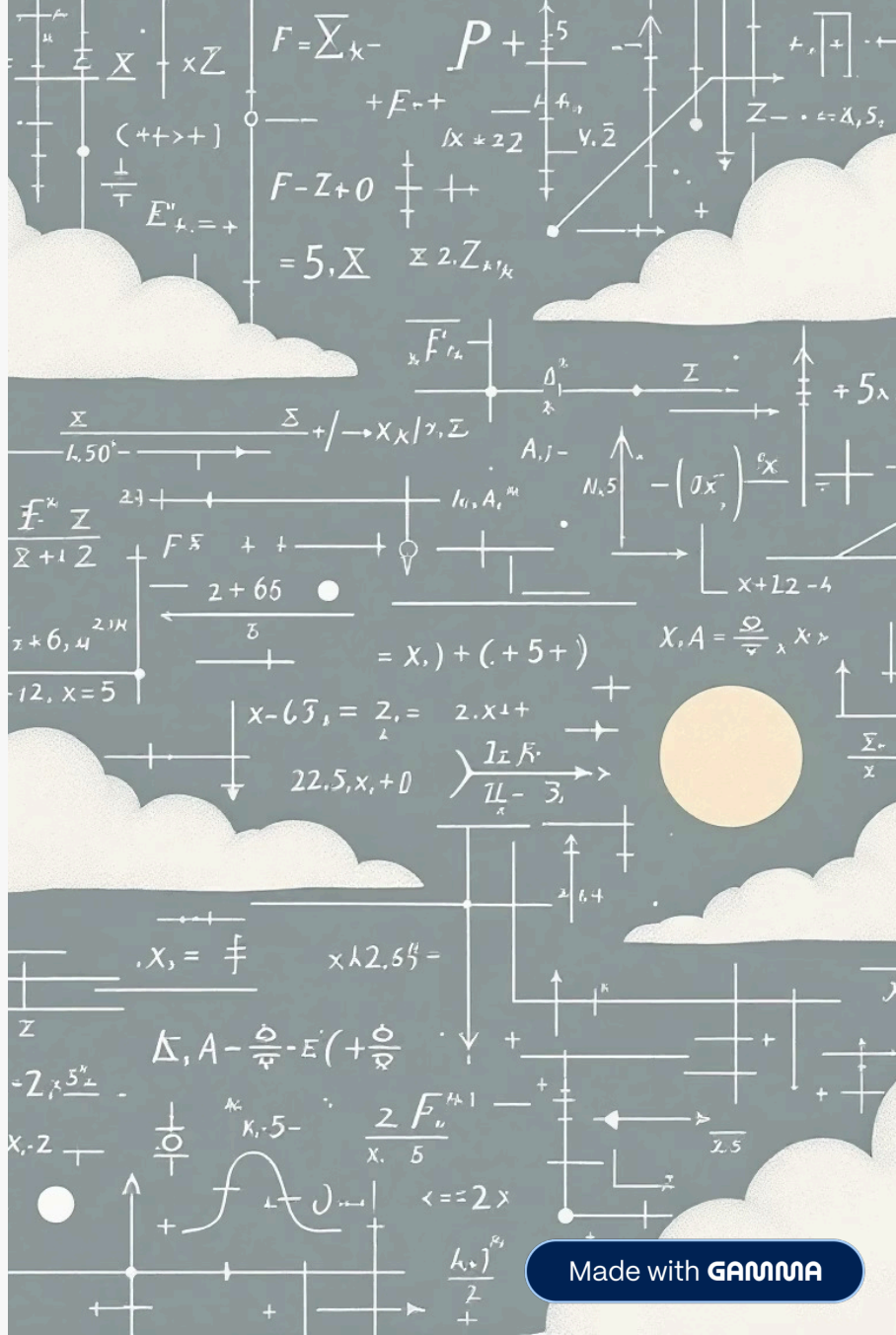
MAP Math Misconception Master

Building a production-ready ML solution to predict student math misconceptions from their explanations in Kaggle's competitive challenge



Competition Overview

The Challenge

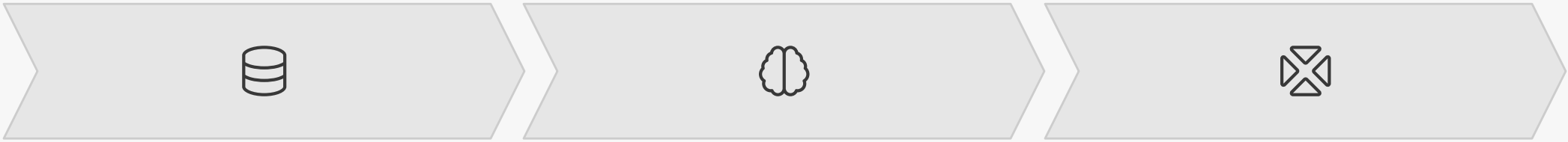
Kaggle's "MAP: Charting Student Math Misunderstandings" competition requires identifying 57+ misconception types from student explanations with precision. Success demands optimizing for MAP@3 metric while handling complex educational NLP patterns.

Target: Top 25% leaderboard position with robust, scalable architecture ready for production deployment.

Key Metrics

- MAP@3 (Mean Average Precision at 3)
- 57+ distinct misconception categories
- 26,836 samples with null misconceptions
- 10% stratified validation split

Three-Phase Action Flow



Phase 1: Data Mastery

Load, inspect, clean, enrich, and validate educational text data with precision handling of missing values and class imbalances

Phase 2: Model Excellence

Configure DeBERTa-v3-base transformer with optimized hyperparameters for educational NLP performance

Phase 3: Prediction Perfection

Generate top-3 predictions, format submissions, and iterate based on leaderboard feedback

Phase 1: Data Engineering Pipeline

01

Load & Inspect

Kaggle API integration with direct data loading, analyzing shapes, null distributions, and outlier patterns

02

Clean & Normalize

Handle 26K+ null misconceptions with "Category:NA" mapping, filter rare classes below 20 samples for model stability

03

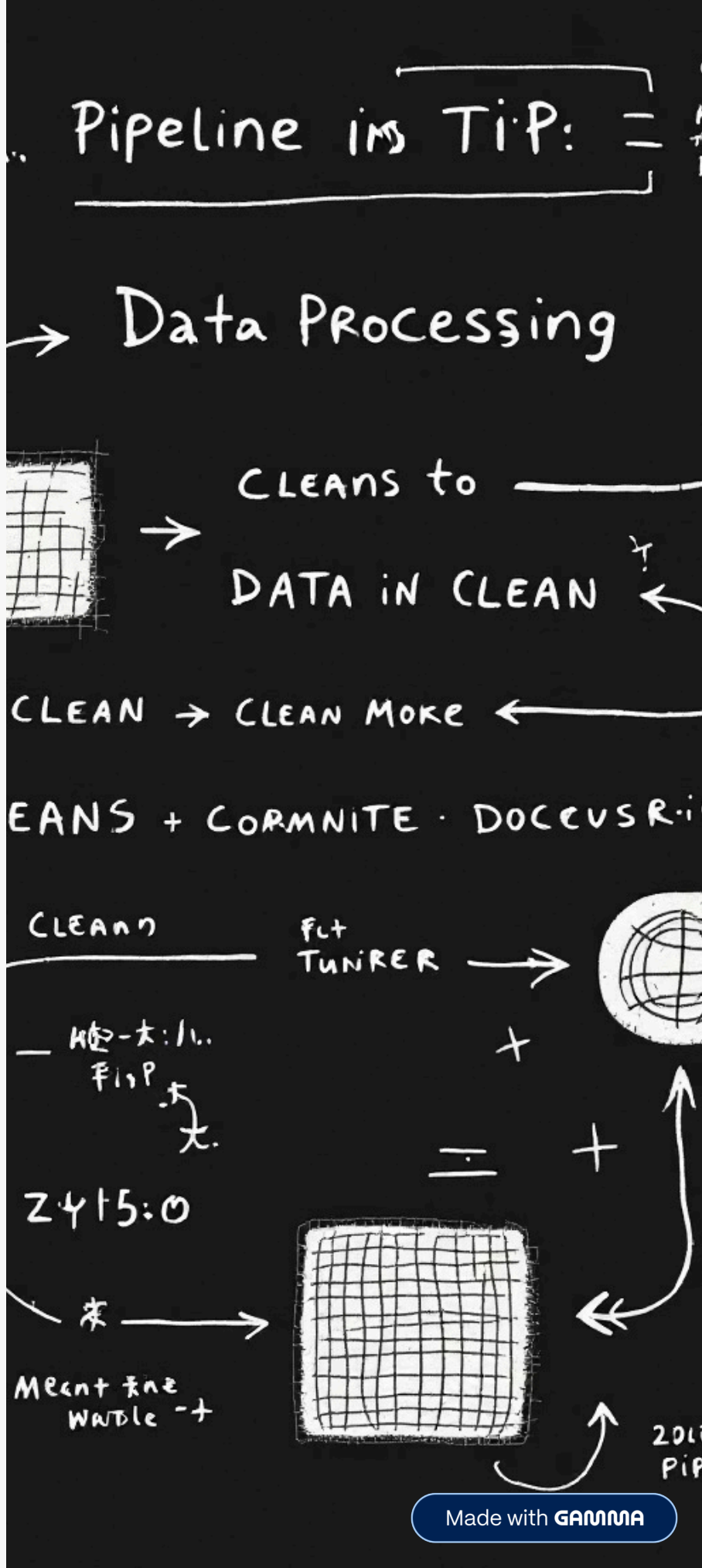
Enrich & Feature Engineer

Combine question, answer, and explanation text using "\[SEP\]" token for optimal context representation

04

Validate Quality

Execute data quality checks, verify class distributions, and confirm label encoding integrity



Critical Data Transformations

Null Handling Strategy

26,836 missing misconceptions mapped to "Category:NA" category, preventing data loss while maintaining model training integrity

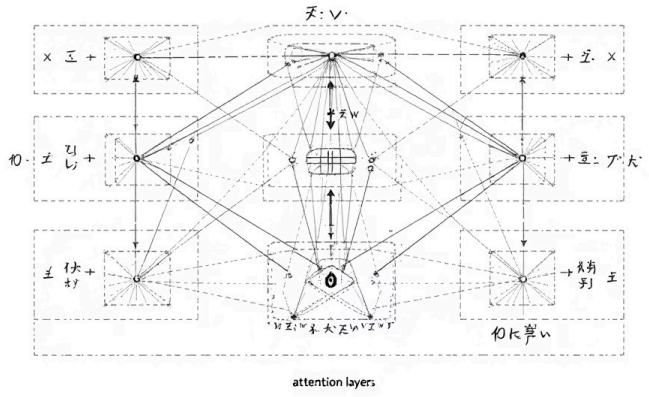
Class Filtering

Remove classes with fewer than 20 samples to ensure statistical stability and prevent overfitting on rare categories

Text Combination

Structured format: "Question \[SEP\] Answer \[SEP\] Explanation" provides comprehensive context for transformer models

Phase 2: DeBERTa-v3-base Architecture



Model Configuration

Microsoft's DeBERTa-v3-base delivers superior NLP performance through disentangled attention mechanisms, optimized for educational text understanding.

- **Max Length:** 256 tokens for comprehensive context
- **Batch Size:** 8 (optimized for T4 GPU memory)
- **Epochs:** 3 (convergence sweet spot)
- **Learning Rate:** 2e-5 (transformer standard)
- **Class Weighting:** Handles severe imbalance

Training Performance Targets

<2h

Training Time

Complete model training on T4 GPU within 2 hours, enabling rapid iteration and experimentation cycles

57+

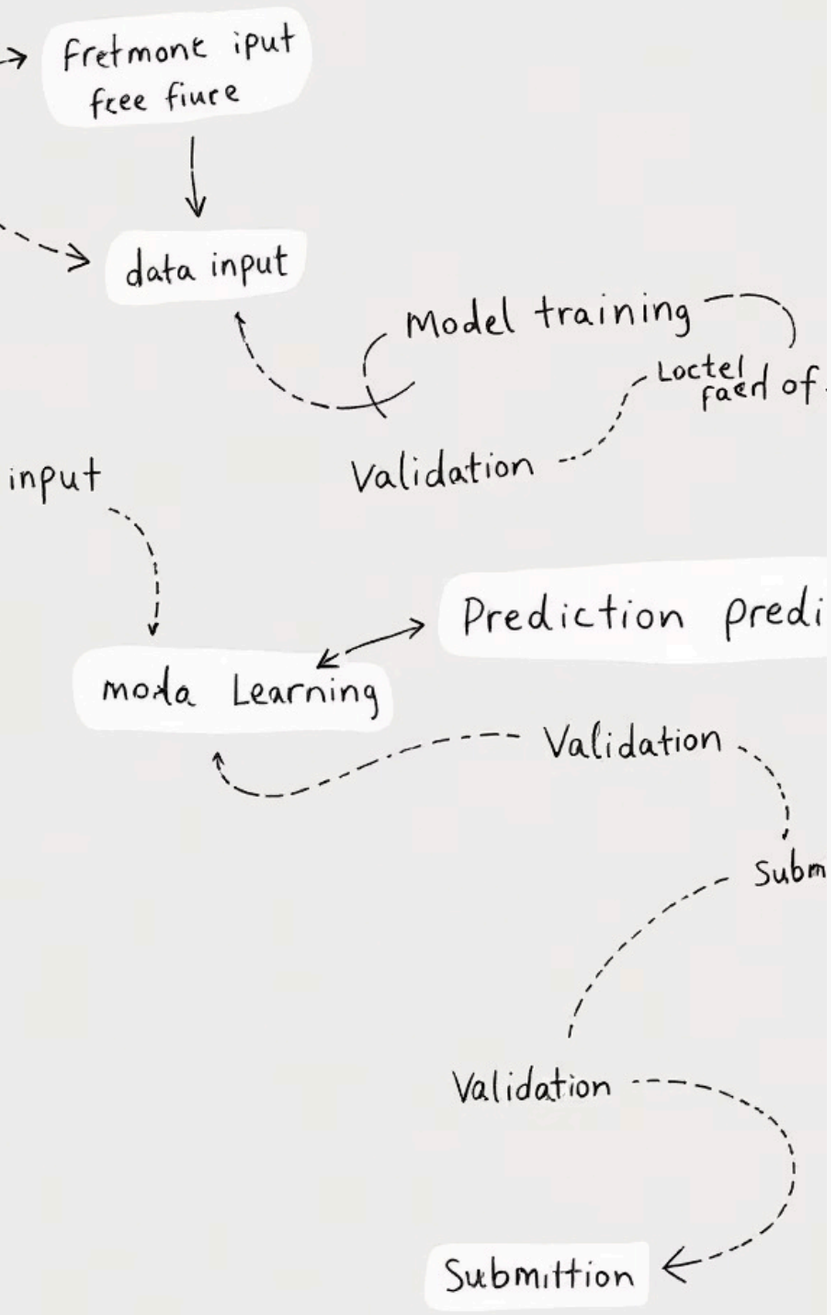
Classes

Dynamic classification across 57+ misconception categories with proper label encoding and mapping

Top 25%

Leaderboard Goal

Target top quartile position demonstrating competitive model performance and robust generalization



Phase 3: Submission Pipeline

1

Predict Top-3

Generate probability distributions and extract top-3 misconception predictions per student explanation sample

2

Format Submission

Structure predictions as space-delimited strings following exact competition specifications: row_id, Category:Misconception

3

Automated Submit

Integrate Kaggle API for seamless submission and immediate leaderboard feedback

4

Iterate & Optimize

Analyze performance metrics, adjust hyperparameters, and refine model architecture based on results

Production-Ready Success Criteria

Accuracy Excellence

Achieve top 25% leaderboard position through MAP@3 optimization, demonstrating competitive performance against global ML engineers

Robustness

Handle all edge cases in student text including misspellings, incomplete explanations, and varied mathematical notation formats

Speed & Efficiency

Complete training cycles under 2 hours on T4 GPU, enabling rapid experimentation and model iteration

Reproducibility

Clean, well-documented codebase with clear dependencies, version control, and step-by-step execution instructions

Scalability

Architecture ready for production deployment with efficient inference pipelines and batch processing capabilities

Technical Stack & Deployment

Core Technologies

- **Model:** microsoft/deberta-v3-base
- **Framework:** PyTorch + Transformers
- **Environment:** Google Colab T4 GPU
- **Data:** Kaggle API integration
- **Validation:** Stratified 10% split

Next Steps

1. Execute complete data pipeline with quality checks
2. Train DeBERTa model with optimized hyperparameters
3. Generate and submit top-3 predictions
4. Analyze leaderboard performance and iterate
5. Document results and deployment strategy