Math Question Classification Project Report

In this project, I focused on classifying math questions into different categories using Natural Language Processing (NLP) techniques. My goal was to develop a model that can automatically categorize mathematical questions into one of eight categories:

1. Algebra
2. Geometry
3. Number Theory
4. Combinatorics
5. Calculus
6. Probability & Statistics
7. Linear Algebra
8. Discrete Mathematics

Dataset Information

I worked with a dataset split into:

- train.csv: Contains labeled math questions for model training
- test.csv: Contains unlabeled questions for prediction

I observed that the training data has an imbalanced distribution across categories:

- Algebra (0): 2,618 samples
- Geometry (1): 2,439 samples
- Probability & Statistics (5): 1,827 samples
- Calculus (4): 1,712 samples
- Number Theory (2): 1,039 samples
- Combinatorics (3): 368 samples
- Linear Algebra (6): 100 samples
- Discrete Mathematics (7): 86 samples

Implementation Details

Data Processing and Modeling

I implemented:

- pre-trained MathBERT in baseline.ipynb
- the demo for the streamlit application in `demo/app.py`
- cross-validation training in `train.py`
- data augmentation in `augment.py`
-

Demo Application

I created an interactive demo using Streamlit:

- `app.py` : Frontend for the classification demo
- `model_utils.py` : Handles model loading and prediction
- `train.py` : Contains cross-validation training logic
- `augment.py` : Implements data augmentation techniques
- `run_demo.sh` : Shell script to run the demo
  - Creates and activates a virtual environment
  - Installs dependencies
  - Runs the demo application on port 8888

Results

My project produces prediction outputs:

- `submission.csv` : submission file for Kaggle competition from MathBERT model
- `submission_lgbm.csv` : submission file for Kaggle competition from LightGBM model

> [!NOTE] This submission is later submitted for evaluation in the Kaggle competition and got **0.8152** score in public leaderboard.

Technologies Used

In my implementation, I used:

- Transformer models (indicated by tokenizer and model configuration files)
- Streamlit for the demo interface
- Pandas for data manipulation
- Scikit-learn or PyTorch for model implementation

Instructions to Run

1. Download dataset files and trained models:

```
chmod +x get_assets.sh
./get_assets.sh 1ancje2FsGw9dTMMCXCO2CfuXsqgDJKBE 1OpVGWl8JlRD3G3mB8IqoAE6DY1bd08sv
```

2. Run the demo application:

```
chmod +x run_demo.sh
./run_demo.sh
```

3. To train the mathBERT model:

```
python3 -m venv venv
source venv/bin/activate
pip install -r requirements.txt
python3 train.py
```