Student Name: - Uttam Kumar

Email Id: - 2025aa05078@wilp.bits-pilani.ac.in

Student id: - 2025AA05078

Section 1: GitHub Repository Details

Repo Link: - https://github.com/uttamjaiswal86/mtech_aiml_assignment_2

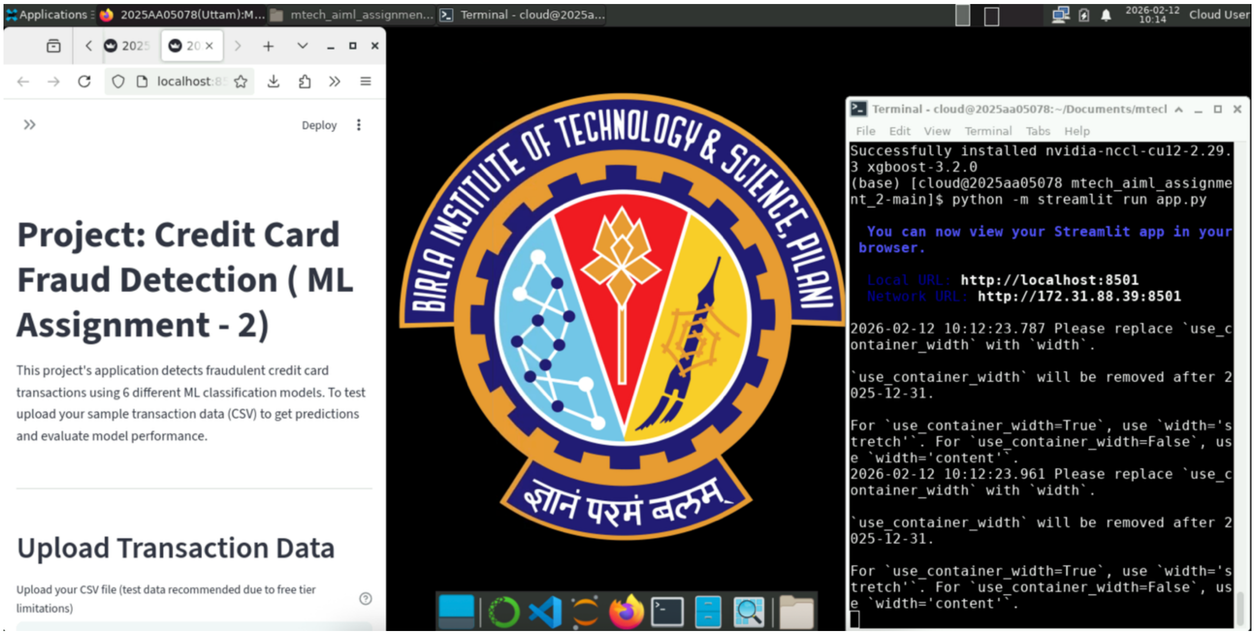
M. Tech. AIML Assignment Repo contain

Section 2: - Live Streamlit Application Link

App URL: - https://mtechaimlassignment2-cafbn4tstfedbzuvvucgvp.streamlit.app/

Section 3: - Screenshot

Please find below attachment for BITS Virtual Lab screenshot


Applications
2025AA05078(Uttam):M...
mtech_aiml_assignmen...
Terminal - cloud@2025a...
2026-02-12 10:14
Cloud User
localhost:85
Deploy
Terminal - cloud@2025aa05078:~/Documents/mtecl
File Edit View Terminal Tabs Help
BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
ज्ञानं परमं बलम्

Project: Credit Card Fraud Detection (ML Assignment - 2)

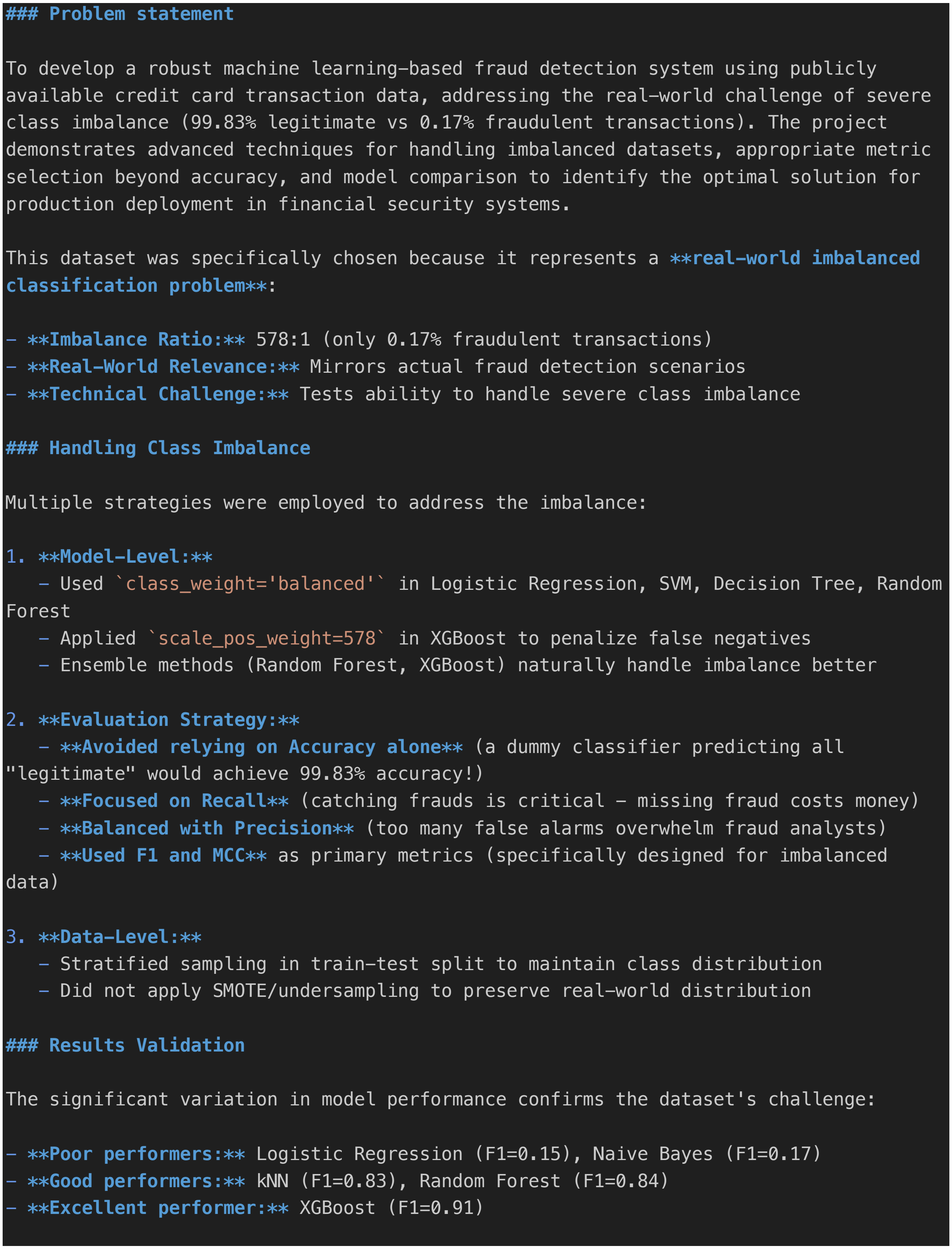
This project's application detects fraudulent credit card transactions using 6 different ML classification models. To test upload your sample transaction data (CSV) to get predictions and evaluate model performance.

Upload Transaction Data

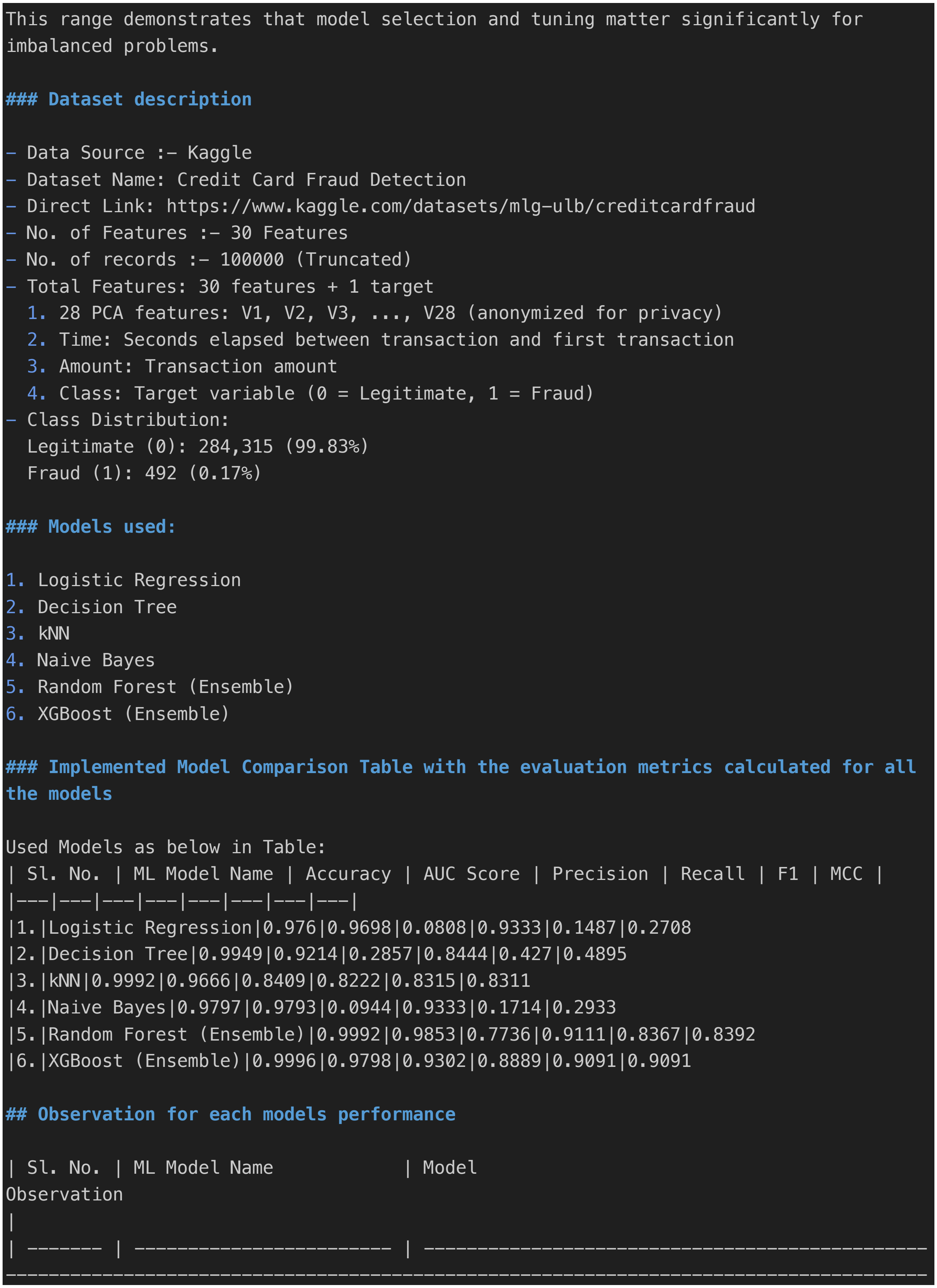
Upload your CSV file (test data recommended due to free tier limitations)

```
Successfully installed nvidia-nccl-cu12-2.29.
3 xgboost-3.2.0
(base) [cloud@2025aa05078 mtech_aiml_assignme
nt_2-main]$ python -m streamlit run app.py

  You can now view your Streamlit app in your
 browser.

  Local URL: http://localhost:8501
  Network URL: http://172.31.88.39:8501

2026-02-12 10:12:23.787 Please replace `use_c
ontainer_width` with `width`.

`use_container_width` will be removed after 2
025-12-31.

For `use_container_width=True`, use `width='s
tretch'`. For `use_container_width=False`, us
e `width='content'`.
2026-02-12 10:12:23.961 Please replace `use_c
ontainer_width` with `width`.

`use_container_width` will be removed after 2
025-12-31.

For `use_container_width=True`, use `width='s
tretch'`. For `use_container_width=False`, us
e `width='content'`.
```

Section 4: - Github README content



```
### Problem statement

To develop a robust machine learning-based fraud detection system using publicly available credit card transaction data, addressing the real-world challenge of severe class imbalance (99.83% legitimate vs 0.17% fraudulent transactions). The project demonstrates advanced techniques for handling imbalanced datasets, appropriate metric selection beyond accuracy, and model comparison to identify the optimal solution for production deployment in financial security systems.

This dataset was specifically chosen because it represents a **real-world imbalanced classification problem**:

- **Imbalance Ratio:** 578:1 (only 0.17% fraudulent transactions)
- **Real-World Relevance:** Mirrors actual fraud detection scenarios
- **Technical Challenge:** Tests ability to handle severe class imbalance

### Handling Class Imbalance

Multiple strategies were employed to address the imbalance:

1. **Model-Level:**
   - Used `class_weight='balanced'` in Logistic Regression, SVM, Decision Tree, Random Forest
   - Applied `scale_pos_weight=578` in XGBoost to penalize false negatives
   - Ensemble methods (Random Forest, XGBoost) naturally handle imbalance better

2. **Evaluation Strategy:**
   - **Avoided relying on Accuracy alone** (a dummy classifier predicting all "legitimate" would achieve 99.83% accuracy!)
   - **Focused on Recall** (catching frauds is critical - missing fraud costs money)
   - **Balanced with Precision** (too many false alarms overwhelm fraud analysts)
   - **Used F1 and MCC** as primary metrics (specifically designed for imbalanced data)

3. **Data-Level:**
   - Stratified sampling in train-test split to maintain class distribution
   - Did not apply SMOTE/undersampling to preserve real-world distribution

### Results Validation

The significant variation in model performance confirms the dataset's challenge:

- **Poor performers:** Logistic Regression (F1=0.15), Naive Bayes (F1=0.17)
- **Good performers:** kNN (F1=0.83), Random Forest (F1=0.84)
- **Excellent performer:** XGBoost (F1=0.91)
```



```
This range demonstrates that model selection and tuning matter significantly for
imbalanced problems.

### Dataset description

- Data Source :- Kaggle
- Dataset Name: Credit Card Fraud Detection
- Direct Link: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud
- No. of Features :- 30 Features
- No. of records :- 100000 (Truncated)
- Total Features: 30 features + 1 target
  1. 28 PCA features: V1, V2, V3, ..., V28 (anonymized for privacy)
  2. Time: Seconds elapsed between transaction and first transaction
  3. Amount: Transaction amount
  4. Class: Target variable (0 = Legitimate, 1 = Fraud)
- Class Distribution:
  Legitimate (0): 284,315 (99.83%)
  Fraud (1): 492 (0.17%)

### Models used:

1. Logistic Regression
2. Decision Tree
3. kNN
4. Naive Bayes
5. Random Forest (Ensemble)
6. XGBoost (Ensemble)

### Implemented Model Comparison Table with the evaluation metrics calculated for all
the models

Used Models as below in Table:
| Sl. No. | ML Model Name | Accuracy | AUC Score | Precision | Recall | F1 | MCC |
|---|---|---|---|---|---|---|---|
|1.|Logistic Regression|0.976|0.9698|0.0808|0.9333|0.1487|0.2708
|2.|Decision Tree|0.9949|0.9214|0.2857|0.8444|0.427|0.4895
|3.|kNN|0.9992|0.9666|0.8409|0.8222|0.8315|0.8311
|4.|Naive Bayes|0.9797|0.9793|0.0944|0.9333|0.1714|0.2933
|5.|Random Forest (Ensemble)|0.9992|0.9853|0.7736|0.9111|0.8367|0.8392
|6.|XGBoost (Ensemble)|0.9996|0.9798|0.9302|0.8889|0.9091|0.9091

## Observation for each models performance

| Sl. No. | ML Model Name            | Model
Observation
|
| ------- | ------------------------ | ----------------------------------------------
--------------------------------------------------------------------------------------
```

```
----------------------------------------------------------------------------------------
------- |
| 1.      | Logistic Regression      | Logistic Regression achieved high recall
(93.33%) making it effective at catching fraudulent transactions, but suffered from
very low precision (8.08%), resulting in many false alarms.
|
| 2.      | Decision Tree            | Decision Tree showed moderate performance with
improved precision (28.57%) over logistic regression but still produced many false
positives. Recall of 84.44% means it caught most frauds but missed about 15%.
|
| 3.      | kNN                      | K-Nearest Neighbors demonstrated excellent
performance with outstanding precision (84.09%), meaning most flagged frauds were
genuine. Recall of 82.22% shows it caught most fraudulent transactions with minimal
false negatives.  |
| 4.      | Naive Bayes              | Gaussian Naive Bayes achieved the highest
recall (93.33%) alongside logistic regression, successfully catching nearly all
fraudulent transactions. Excellent AUC (0.9793) demonstrates strong probabilistic
separation of classes. |
| 5.      | Random Forest (Ensemble) | Random Forest delivered excellent overall
performance with strong balance across metrics. High recall (91.11%) ensures most
frauds are caught while good precision (77.36%) minimizes false alarms to acceptable
levels.           |
| 6.      | XGBoost (Ensemble)       | XGBoost achieved the best overall performance
across all key metrics, making it the top model. Outstanding precision (93.02%) means
very few false alarms, while strong recall (88.89%) catches most fraudulent
transactions.      |
```