Random- Forest (from- scratch) for Fraud Detection

----
1) OVERVIEW

Entry point: RandomForest.py
Core tasks:
- Load train/test CSVs
- Preprocess numeric features (median impute, quantile binning)
- Train a Random Forest (bootstrap + sqrt(feature) subsampling)
- Predict by majority vote
- (Optional) Evaluate with confusion matrix metrics and BER CI

Intended for: research/teaching, reproducible baselines, explainable tree ensembles without external ML frameworks.

----
2) REPOSITORY STRUCTURE

```
.
├── SplitCriterion2.0.py              # Main script
└── Team Version/
     ├── train.csv                     # Training data (must include 'isFraud')
     └── test.csv                      # Test data (must include 'TransactionID')
```

----
3) DATA REQUIREMENTS

Training file: train.csv with binary 'isFraud' target column.
Test file: test.csv used for inference.

Feature lists inside the code:
```
categoricalFeatures = ['ProductCD', 'card4', 'card6']
numericalFeatures =
['C1','C2','C3','C4','C5','C6','C7','C8','C9','C10','C11','C12','C13','C14','addr2','card3','card5']
features = ['ProductCD','card1','card2','card3','card4','card5','card6','addr2',
            'C1','C2','C3','C4','C5','C6','C7','C8','C9','C10','C11','C12','C13','C14']
```

----
4) ENVIRONMENT SETUP

Python: 3.9+ recommended (3.8 works)
Dependencies:
- pandas
- numpy
- scipy

Install with:
```
pip install pandas numpy scipy
```

----
5) HOW TO RUN

Run as a script:
```
python SplitCriterion2.0.py
```

Process:
1. Loads train and test CSV files.
2. Processes numeric features.
3. Trains Random Forest (defaults: 5 trees, 0.85 sample size, Gini index, alpha=0.05, maxDepth=15).
4. Prints two trees.
5. (Optional) Uncomment 'predictAll()' to export predictions.

Import into another script:

```
from SplitCriterion2_0 import processNumericData, randomForest

data     = processNumericData(data, numericFeatures=[...])
testdata = processNumericData(testdata, numericFeatures=[...])

forest = randomForest(data=data, features=[...], numberOfTrees=50,
                      sampleSize=0.67, target="isFraud",
                      impurityCriteria="giniIndex", alpha=0.01, maxDepth=12)
forest.randomTree()

pred = forest.predictTree(testdata.iloc[0])
```

The script will do the tasks in the following order:
1. Train the Random Forest on the training dataset
2. Print tree summaries and intermediate outputs to console
3. Save the test predictions to:

You can modify hyperparameters directly in the “HYPER_PARAM CONTROL” section near the bottom of SplitCriterion.py:

```
forest = randomForest(
    data,
    features=features,
    numberOfTrees=10,
    sampleSize=0.60,
    target="isFraud",
    impurityCriteria="giniIndex",    # or "entropy"
    alpha=0.05,                      # Chi-square pruning threshold
    maxDepth=10                      # Control overfitting
)
```

----
6) KEY CONFIGURATION PARAMETERS

- imbalance handling: majority class reweighted by IR = minor/major
- impurityCriteria: "giniIndex", "entropy", or "misClassificationError"
- alpha: chi- square significance level (lower = stricter)
- maxDepth: tree depth limit
- sampleSize: bootstrap fraction per tree
- numberOfTrees: number of trees in ensemble

----
7) EVALUATION UTILITIES

Use Confusion_mat_results(preds, y_true, Zscore=1.96) to compute metrics:
- MME (Misclassification Error)
- BER (Balanced Error Rate)
- Confidence Interval (95% default)

----
8) COMMON PITFALLS

- File paths must match expected structure.
- predictAll() expects 'TransactionID' in testdata.
- Quantile binning may drop duplicates;
- Nodes with pure classes become leaf nodes.

----
9) CHECKLIST

[ ] Python >= 3.8 installed with pandas, numpy, scipy
[ ] train.csv & test.csv in ../Team Version/
[ ] Feature lists updated
[ ] Hyperparameters configured
[ ] predictAll() uncommented if exporting predictions

Other tunable parameters:
nBins in processNumericData() — controls numeric feature binning.
skew and topFreqThresh — control skewness threshold for binning.

Team Name: IntelligentTrainers
Team Members
Kaneez Fatima (Lead)
Machine Learning Course Project — CS529
University of New Mexico, 2025

Muhammad Zuboraj
Machine Learning Course Project — CS529
University of New Mexico, 2025

Contributions of Team Members:
1. Fatima executed overall construction of trees and random forest algorithiom. She also defined functions like entropy, Gini Index, Information Gain, ProcessNuemicData (for Numeric Data processing). She also helped Muhammed to write the report.
2. Muhammed helped Fatima in data analysis and also wrote functions like ChiSquareTest, Confusion_mat_results (for building confusion metrics) for accuracy test. He also made separate validation data from training dataset to test the functions Fatima execute. He was responsible for most of the report writing.

Kaggle score, accuracy, and date run

The maximum accuracy achieved on kaggle is 73.6% for the data sampling of 80% of train data. Number of trees used were 5 and maxDepth was set to 15, which took almost 3-4 mins to run