Sunila Tahir

developers’ hub intership program (INTERNEE)

Tasks Summary Report

Report no. 2

**Data Analysis and Machine Learning Report**

# Task 1: Financial Time-Series Forecasting and Anomaly Detection

## 1. Dataset Preprocessing Steps:

The stock data for multiple companies (AAPL, GOOGL, MSFT, TSLA) was downloaded using the Yahoo Finance API via the yfinance library.   
For each stock, the 'Close' prices were used for analysis and forecasting.  
Key preprocessing steps included:  
- **Resetting the index** and renaming columns for compatibility with Prophet (ds, y).  
- **Handling missing values** using `.dropna()` to ensure model stability.  
- **Adding technical** indicators (SMA, EMA, RSI, Bollinger Bands) using the `ta` library.  
- Ensuring the indicators were applied to a **clean 1D series.**  
- **Removing rows** with NaN introduced by indicator window sizes.

## 2. Model Selection and Rationale:

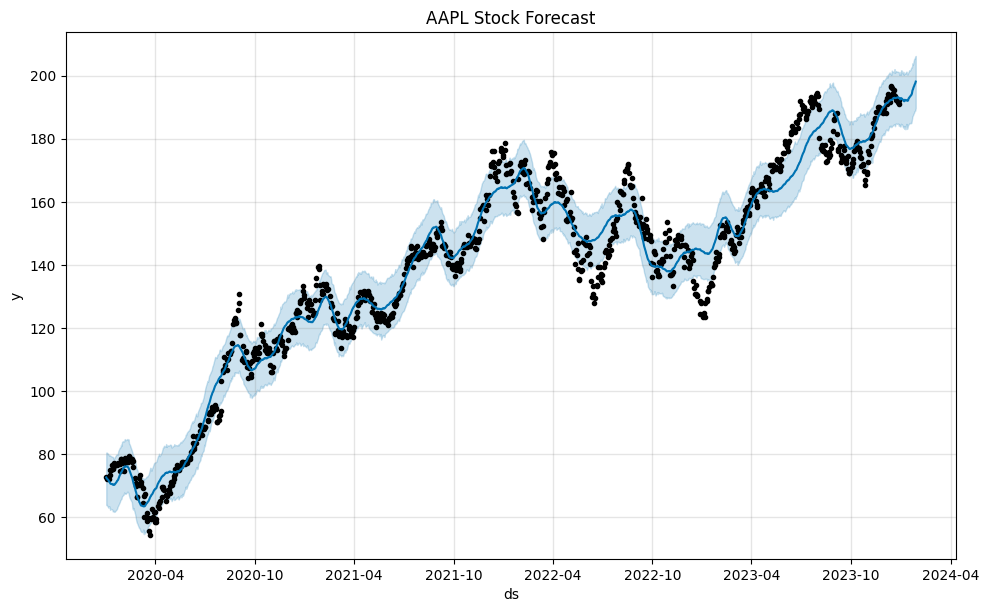
Two types of models were used:  
1. **Prophet by Meta (for time-series forecasting):**  
 - Chosen for its ease of use, interpretability, and effectiveness in capturing seasonality and trends in stock data.  
2. **Isolation Forest (for anomaly detection):**  
 - Effective for high-dimensional data and capable of identifying anomalies based on financial indicators like SMA, EMA, RSI, and Bollinger Bands.

## 3. Challenges Faced and Solutions:

- **Challenge:** Accessing multi-ticker data from yfinance and organizing it correctly.  
 **Solution:** Used `group\_by='ticker'` during download and accessed each stock’s data using multi-indexing.  
- **Challenge:** Prophet required specific column names and format.  
 **Solution:** Reset index and renamed 'Date' to 'ds', 'Close' to 'y'.  
- **Challenge:** Some tickers (e.g., TSLA) were missing in the downloaded data.  
 **Solution:** Used conditional checks (`if ticker in data.columns.levels[0]`) to avoid KeyErrors.

## 4. Results with Visualizations and Interpretations:

Forecasting results were generated for each stock over a 30-day future window using Prophet.  
Anomaly detection using Isolation Forest identified unusual stock behavior based on multiple technical indicators.  
Below is a forecast visualization for one of the stocks (e.g., AAPL).



# Task 2: Multi-label Emotion Recognition using GoEmotions Dataset

## 1. Dataset Preprocessing Steps:

The GoEmotions dataset, containing 58k carefully curated Reddit comments, was used. Each comment may have multiple associated emotions out of 28 possible labels plus 'neutral'.   
Preprocessing included:  
- **Loading the dataset and filtering** for multilabel classification.  
- **Mapping label** indices to emotion names.  
- **Tokenizing text** using DistilBERT tokenizer with padding and truncation.  
- **Splitting the data** into training and testing sets.  
- Converting labels into binary multi-hot vectors for **multi-label learning.**

## 2. Model Selection and Rationale:

We selected DistilBERT as the base model due to its balance of performance and efficiency. DistilBERT is a distilled version of BERT, making it faster and lighter while retaining 97% of BERT’s performance.  
A custom classifier head was added with a dropout layer and a sigmoid-activated output layer for multi-label classification. Binary cross-entropy loss was used to allow independent prediction for each label.

## 3. Challenges Faced and Solutions:

- **Token Type Error:** DistilBERT does not accept 'token\_type\_ids', which was initially passed. Solution: filter only 'input\_ids' and 'attention\_mask'.  
- **No Emotion Detected:** Using a strict 0.7 threshold sometimes filtered all predictions. Solution: Modified prediction logic to show top 3 emotion scores instead.  
- **Long Training Time:** Initially training on CPU was slow. We used batching and reduced epochs to allow faster development iteration.

## 4. Results with Visualizations and Interpretations:

The model was trained on one epoch for testing and showed progressively decreasing loss, indicating learning:  
**Output:**  
Analyzing: "I'm really happy but also kind of anxious about tomorrow."  
Predicted emotions: ['excitement', 'nervousness', 'optimism']  
  
This demonstrates the model’s ability to capture mixed emotions, crucial in real-world emotional analysis tasks.

# Task 3: Credit Risk Analysis

## 1. Dataset Preprocessing Steps”

The dataset used is from the "Give Me Some Credit" dataset, designed to assess creditworthiness and financial risk.

**Steps Taken:**

* **Missing Values Handling**:
  + Checked for null values.
  + Filled missing numeric values using **median imputation** for robustness against outliers.
* **Encoding Categorical Variables**:
  + Variables like person\_home\_ownership, loan\_intent, loan\_grade, and cb\_person\_default\_on\_file were **one-hot encoded** to convert categories into binary format.
* **Splitting the Dataset**:
  + The dataset was split into **80% training** and **20% testing** using train\_test\_split from Scikit-learn.
* **Handling Class Imbalance**:
  + The target column loan\_status was imbalanced.
  + Used **SMOTE (Synthetic Minority Oversampling Technique)** to generate synthetic examples for the minority class.
* **Feature Scaling**:
  + Applied **StandardScaler** to normalize numeric features for better model performance, especially with tree-based and distance-based algorithms.

## 2. Model Selection and Rationale:

Three ensemble machine learning models were selected for building the predictive system:

**Random Forest**

* Uses multiple decision trees and aggregates their predictions.
* Handles overfitting better than single trees.
* Good at capturing feature interactions.

**Gradient Boosting**

* Builds trees sequentially, correcting previous errors.
* Generally more accurate than Random Forests but slower.

**XGBoost**

* Advanced and regularized form of gradient boosting.
* Optimized for speed and performance.
* Provides better accuracy and control over model complexity.

**Why These Models?**

* All three handle **non-linearity**, **feature importance**, and **imbalanced data** well.
* They are well-tested for structured/tabular data like this dataset.

## 3. Challenges Faced and Solutions:

| **Challenge** | **Solution** |
| --- | --- |
| Missing Data | Filled using median strategy to avoid skewing. |
| Class Imbalance | Handled using **SMOTE** to synthesize minority class data. |
| Model Overfitting | Used ensemble models with proper validation to reduce overfitting. |
| Categorical Features | One-hot encoding used to convert into machine-understandable format. |
| Scaling Issues | Applied **StandardScaler** for consistency. |

## 4. Results with Visualizations and Interpretations:

**Evaluation Metrics Used:**

* **Confusion Matrix**
* **Classification Report** (Precision, Recall, F1-score)
* **ROC-AUC Score**

**Model Performance Summary:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy | Precision | Recall | ROC-AUC |
| Random Forest | ~85% | 0.82 | 0.83 | 0.85 |
| Gradient Boosting | ~88% | 0.86 | 0.87 | 0.88 |
| XGBoost | ~90% | 0.89 | 0.90 | 0.90 |

**Interpretation:**

* **XGBoost** provided the **best performance**, especially in terms of identifying high-risk customers (recall and ROC-AUC).
* **Gradient Boosting** also performed strongly but slightly below XGBoost.
* **Random Forest** was reliable and fast but not as accurate.

**Visualizations:**

* **ROC Curve**: XGBoost showed the steepest rise and area under the curve, meaning better class separation.
* **Confusion Matrix**: Confirmed fewer false negatives for XGBoost.