MoSCoW Prioritization Analysis

Twitter Fake Account Detector Project

Udit Jhanjhariya
Registration No: 23FE10CSE00640
Manipal University Jaipur
B.Tech CSE – 2027

September 2025

Contents

1 Introduction

The MoSCoW method is a prioritization technique used in requirements engineering and project management to categorize features and requirements based on their importance to project success. The acronym stands for:

- **Must have** - Critical requirements without which the project fails
- **Should have** - Important requirements that add significant value
- **Could have** - Desirable features that enhance the system
- **Won't have** - Features explicitly excluded from current scope

This document applies MoSCoW prioritization to the Twitter Fake Account Detector project, a machine learning-powered web application designed to identify potentially fake Twitter profiles using various profile features and characteristics.

2 Must Have Requirements

These are the core functionalities without which the project cannot fulfill its primary purpose of detecting fake Twitter accounts.

- **User input interface:** Manual input for tweet count, followers, following, favorites, listed count, account age, bio length, language, and gender.
- **Input validation:** Min/max checks and feedback on invalid inputs.
- **Data preprocessing:** Encoding gender and language, normalization, and validation.
- **Model integration:** Load pretrained Random Forest classifier, real-time prediction, confidence scoring.
- **Result display:** Clear textual classification (Real/Fake), confidence percentages, probability breakdown, color-coded indicators.
- **API endpoint:** RESTful POST /predict endpoint handling form-data/JSON with error handling.
- **Security:** Input sanitization, CSRF protection, secure backend validation.
- **Performance:** Response within 2 seconds for most requests; model loads within 5 seconds.

3 Should Have Requirements

Features that significantly enhance user experience and system robustness.

- **Automatic gender detection:** Gender-guesser integration with first name extraction.
- **Responsive dark-themed UI:** Bootstrap-based, mobile-compatible, user-friendly form layout.
- **Detailed analytics:** Side-by-side display of fake/real probabilities, confidence visuals.

- **Concurrent user support:** Handle at least 10 users simultaneously, stateless backend.
- **Testing and documentation:** Unit and integration tests, API docs, PEP8 compliance.
- **Security enhancements:** Privacy protections (no personal data logging), HTTPS usage.

4 Could Have Requirements

Additional features to add value but not essential in initial release.

- **Twitter API integration:** Auto-fetch profile data for analysis.
- **Analytics and visualization:** Interactive charts, model performance visualization.
- **Model improvements:** Ensemble methods, deep learning, hyperparameter tuning.
- **User experience enhancements:** Export results, bookmarking, advanced search.
- **Platform extensions:** Multi-language frontend, mobile apps, browser extensions.
- **Infrastructure:** Caching, load balancing, monitoring dashboards.

5 Won't Have Requirements

Explicitly excluded in current scope to maintain focus.

- **User account management:** Registration, authentication, dashboards.
- **Real-time data processing:** Live Twitter streams, automated alerts.
- **Enterprise features:** Multi-tenant architecture, admin panels.
- **Advanced analytics:** Sentiment analysis, network graphs.
- **Multi-platform integrations:** Other social media platforms.