Smart Assessment and Tracking of Your Applicants (SATYA)

Table of Contents

1. Introduction

This documentation outlines the approach and methods used to detect fraudulent resumes and uncover biases in professional recommendations using AI.

Through this project, we aim to solve a very critical problem: identifying fraudulent or embellished resumes and detecting biased or manipulated professional recommendations.

2. Problem Statement and Approach

Problem 1: Fraudulent and Embellished Resumes

To solve the problem of identifying fraudulent resumes, we have chosen to divide our problem into the following sub-problems:

1) Cross check every claim against recommendation letters and flag if the claim is exaggerated: This problem has been tackled by using a combination of a sentence based transformer (MiniLM-L6-V2) and an LLM based approach.
   It should be noted that rather than comparing raw strings to check similarities, we extracted useful "key words" and compared them with the recommendation letters to check the reliability of claims.

2) Identification of Reciprocal Endorsements: Identifying circular/reciprocal endorsements was an important yet simple task. We achieved this by utilizing the dataset provided and applying basic algorithms to check the recommendations.

3) Identification of Vague/Suspicious Wording: We used advanced NLP techniques to identify buzzwords and general vagueness in a resume to classify whether the writer has substantial achievements. To enhance this approach we coupled it with a large language model for better accuracy.

4) Suspicious Patterns: Patterns like frequent job changes in a short period of time, inconsistency in jobs, unexplained gaps in employment, etc. have been identified and flagged.

Based on these factors a weighted sum has been calculated to obtain a score that indicates the likelihood of a particular resume being fraudulent.

Problem 2: Comprehensive Data Analysis

The second problem was more open ended and as a result, we researched pre-existing solutions and chose to enhance these solutions to create a robust solution for analyzing data. The problems we tried to solve in this part were as follows:

1) Data Clustering: We clustered data based on a multitude of parameters including number of connections, people a specific candidate has worked with, and their overall network strength. We have chosen to visually show these clusters in our frontend through PageRank. They have also been added in our github repo.

2) Influence Ranking: In order to analyze the data effectively, we have chosen to include influence ranking which indicates a candidate's "influence". This includes positions mentioned in their resume and their overall recommendation landscape.

3) Connection Strength: As mentioned above, the connection between the candidates can be mapped by the recommendation system. Furthermore, sentiment of the recommendations was analyzed and overly positive recommendations were flagged.

Similar to problem 1, these factors not only help in analyzing given data but also provide contribute to a weighted sum that make up the CIID_score.

Note: Both the scores talked about in these problems have been further combined in a weighted system to provide a comprehensive AI powered candidate leaderboard system.

Problem 3: HR Friendly Dashboard

In order to showcase our findings of the above two problems, an HR friendly admin dashboard is necessary. We have chosen to include various illustrative charts and the Clustering results like we mentioned earlier. This dashboard, although rudimentary provides a roadmap for future scaling in this use-case.

Reference Image attached below:
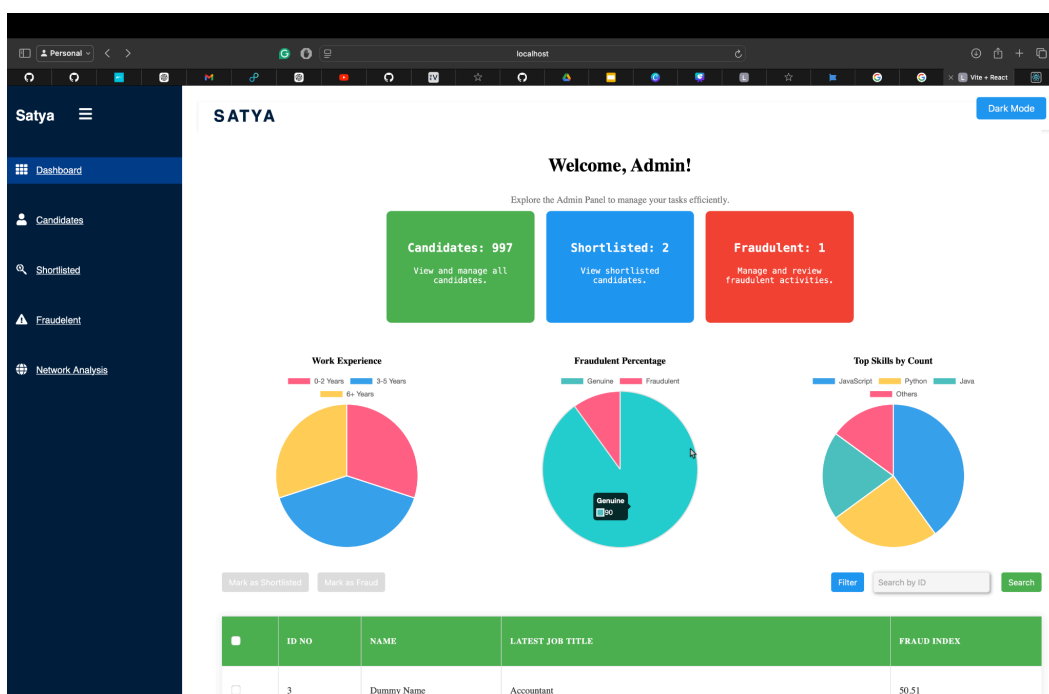
The following are some of the important features of our Dashboard.

1) Addition of compiled rankings with searchable filters.
2)Data Visualizers and Indicators for each candidate
3) Comprehensive social network provided for each candidate
4)Work Experience progress bar for each candidate
5)Dropdown table for candidate details section.
6)Easy Decision Making tool to move candidate to interview round or remove.
7)Toggleable Markers for adding a candidate under the fraud marker.
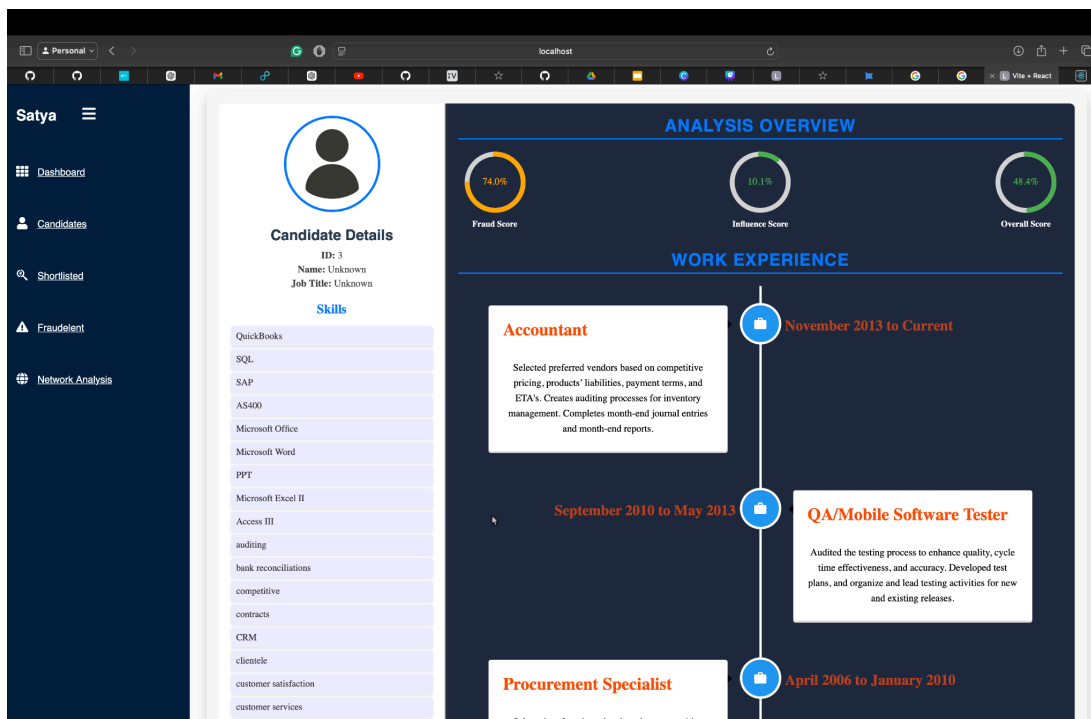
3. Technology Stack

1)Frontend: React.js
2)Backend: Flask
3)ML Libraries: Scikit-learn, Tensorflow, NLTK, TF-IDF, Matplotlib, Plotly, Pytorch
4)Database Storage : Data Stored in csv files and MongoDB