Machine Learning @YorkU

AI driven resume screening

Manik Jain, Praveen Kumar, Logan McGovern

Introduction

Design and implement a web-based user interface and backend integration that enables users to upload candidate resumes and obtain a machine learning–based prediction indicating whether the resume matches a specified job or criteria set. The system will automate candidate identification, enrich input features, and invoke a prediction model via an API.

Functional Overview

The application will provide an interface allowing users to upload resume files. The system will parse the resume file, extract the candidate name using predefined parsing rules. When a resume is uploaded, the system will submit to a machine learning prediction API and reference an internal dataset to retrieve additional features required for prediction. The system will display the prediction result - whether a candidate is a good fit or not, along with the contributing factors as result.

High Level application flow

- User uploads resume file through UI
- System validates file type and naming convention
- System extracts candidate details from uploaded resume file
- Prediction API is invoked exposed by the machine learning model.
- Application queries internal hidden data table using candidate name as lookup key
- API returns prediction label, match result, and prediction contributing factors
- UI displays prediction results to the user

Data handling model

- Candidate Name – extracted from uploaded file
- Resume File – stored or passed as binary/reference (Not is scope)
- Hidden Resume Dataset – internal reference table containing enriched attributes per candidate.
- Prediction Response – structured JSON response containing prediction outcome and scoring metrics.

Model Training

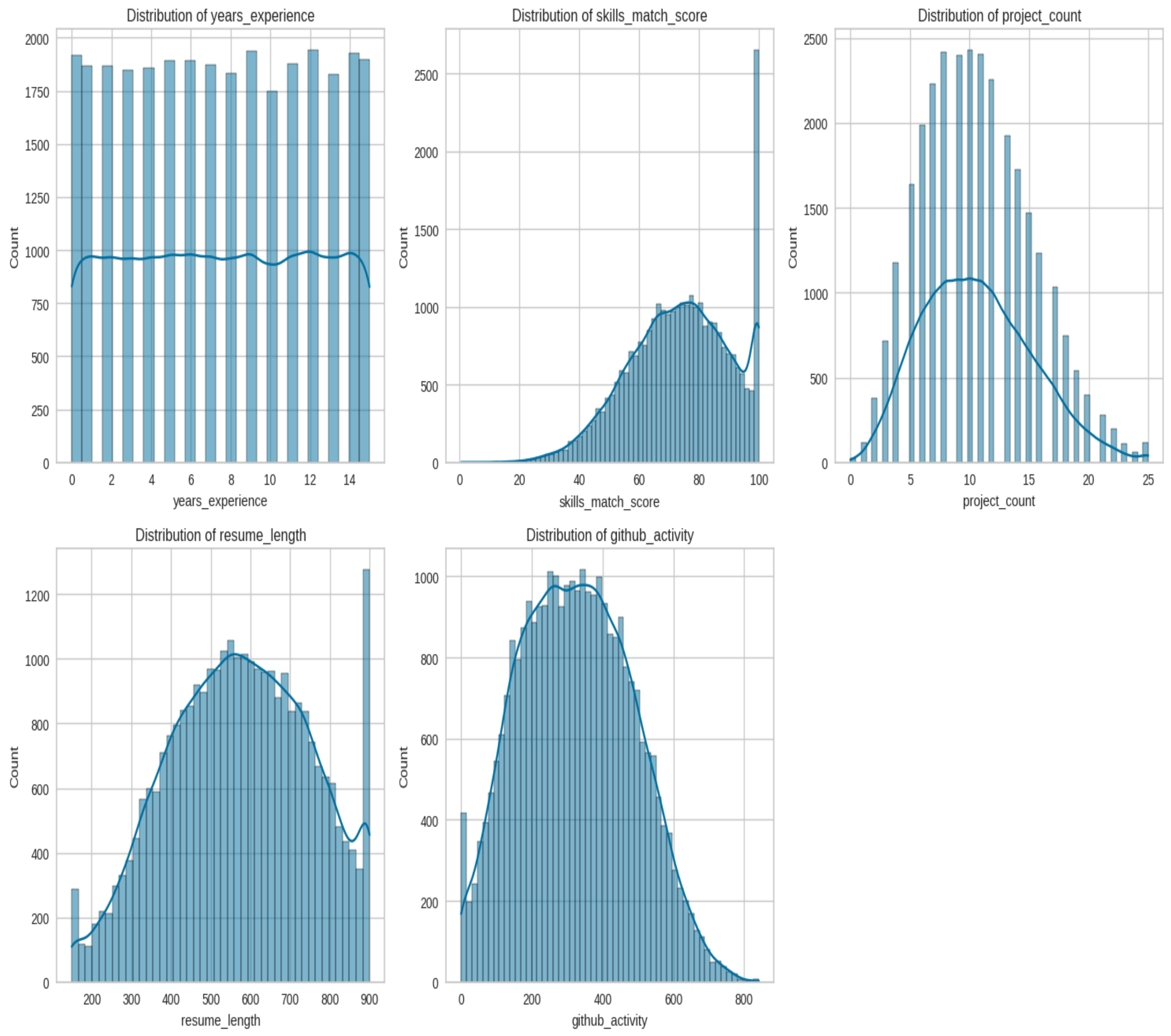
Training the model for this project takes 2 step approach:

Exploratory Data analysis

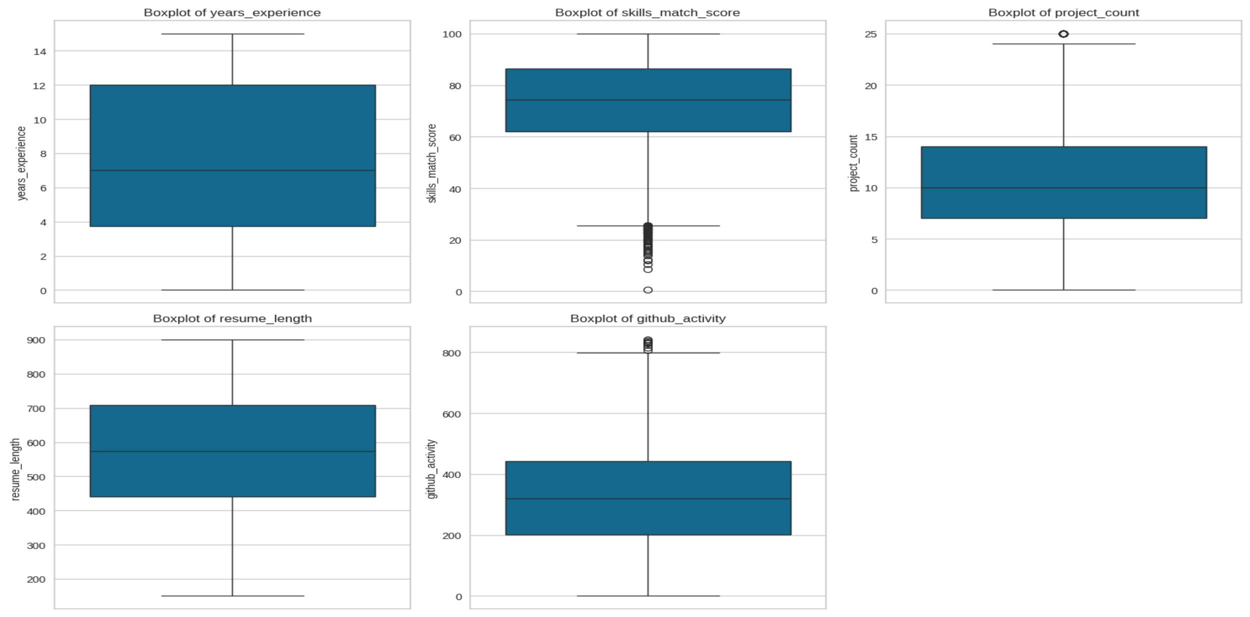
Before applying PyCaret's compare_models() function, we performed exploratory data analysis to better understand the dataset. Using Seaborn and Matplotlib, we visualized the distribution patterns of key features. Years of professional experience was flatly distributed, while skill match score, project count, and GitHub activity were approximately normally distributed, with the caveat that resume length and skill match score exhibited spikes at the upper extremes, indicating a concentration of applicants with longer resumes and higher skill alignment.

Correlation analysis showed that GitHub activity and resume length were strongly associated with years of experience, and project count was highly correlated with GitHub activity. Candidates with bachelor's and master's degrees substantially outnumbered those with PhD or high school-level education. These correlations were not concerning, as the leading candidate models are tree-based algorithms, which effectively handle skewed distributions and correlated features. We will also account for the fact that the dataset is imbalanced.

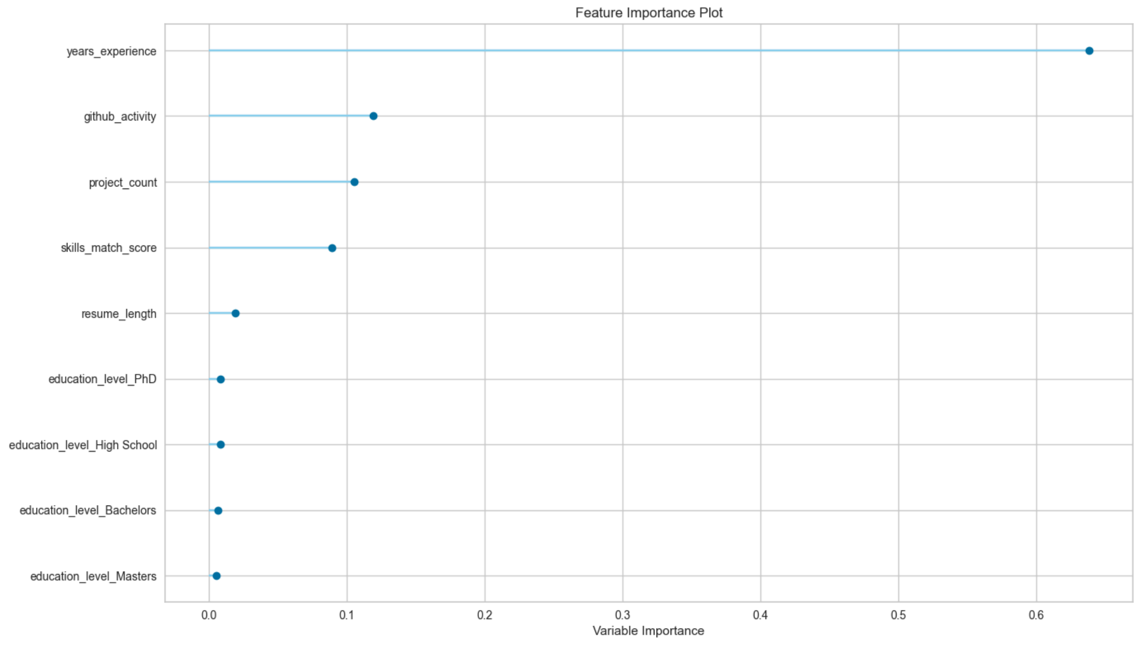
Feature Xcount plots



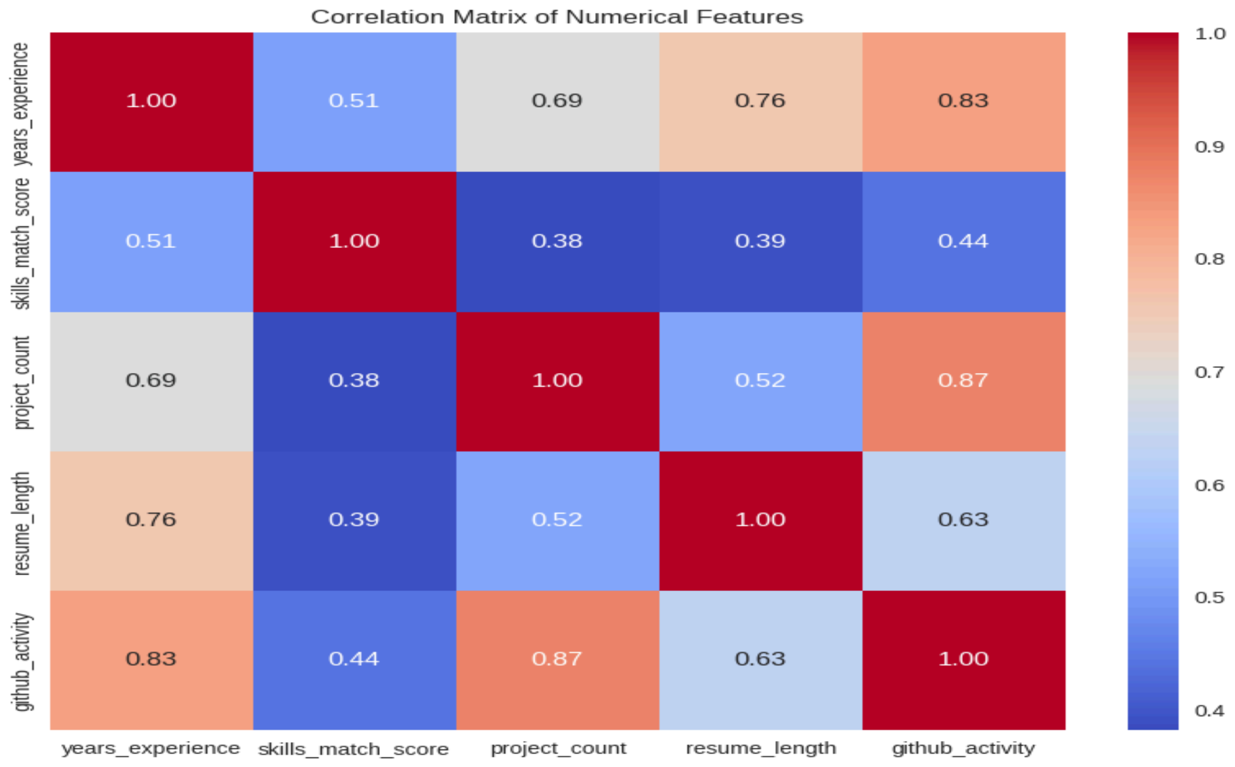
Feature Box+Whisker plots



Feature plot

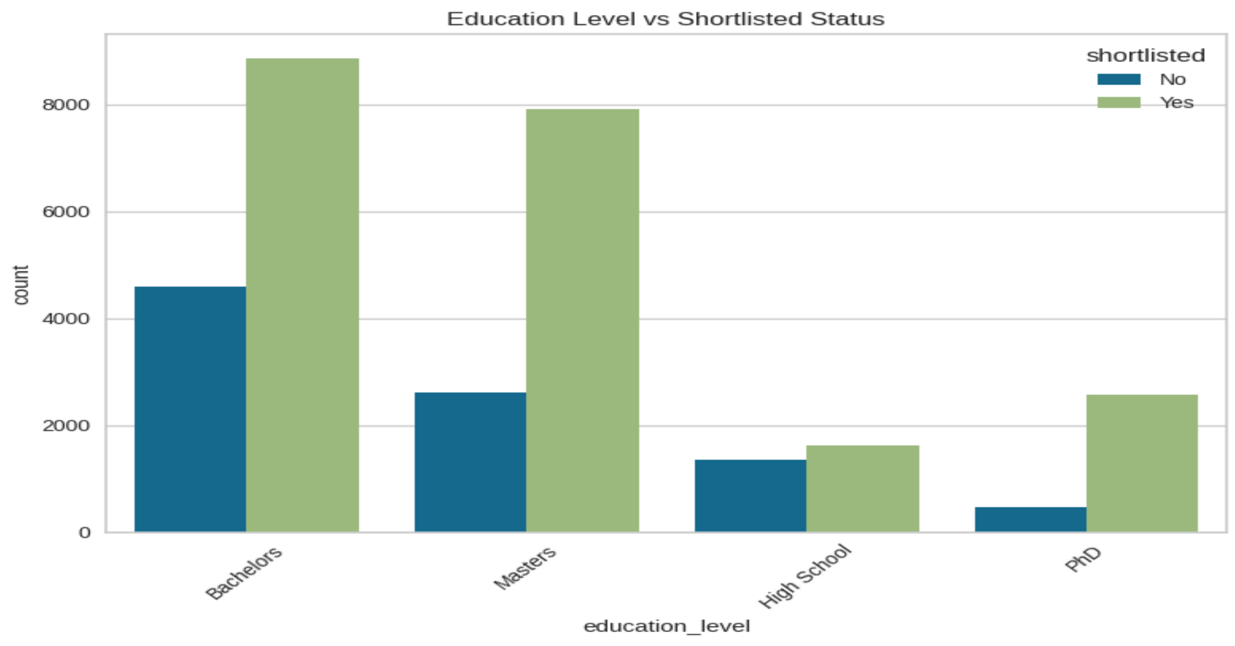


Correlation Matrix

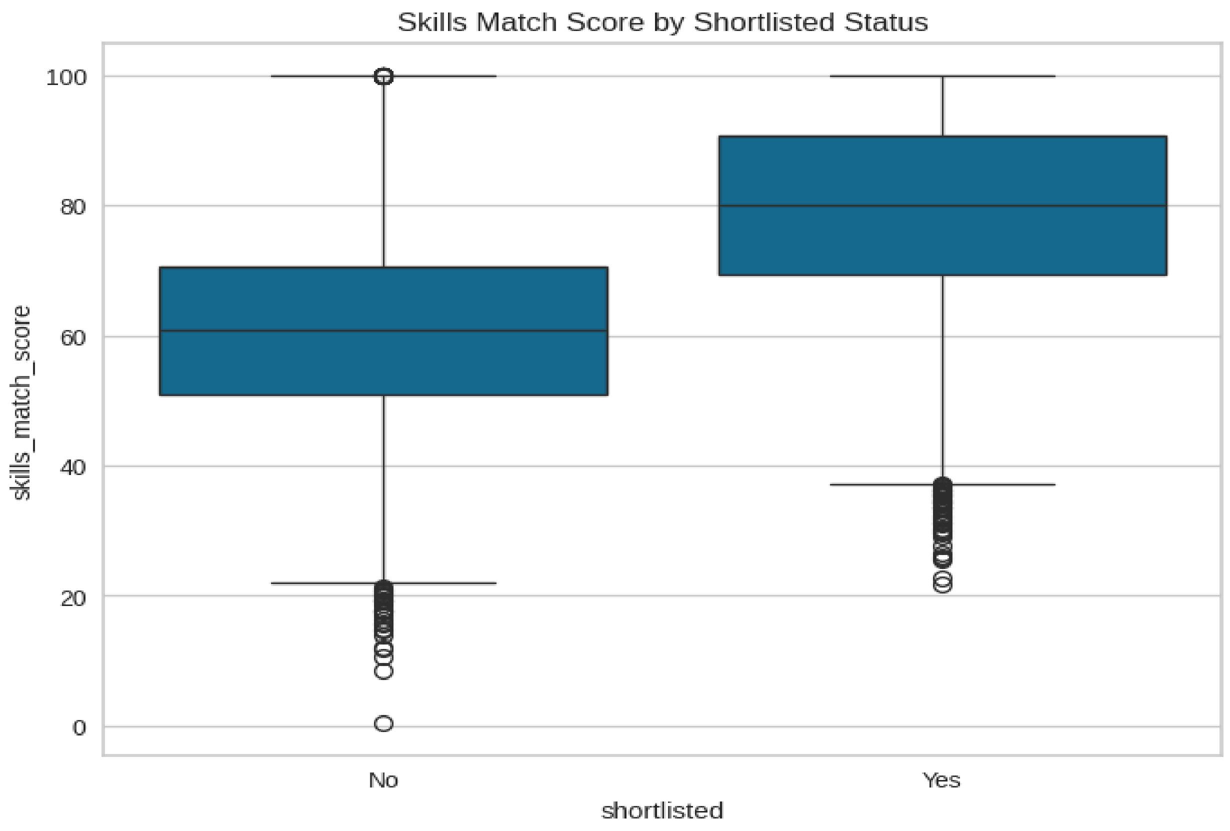


	years_experience	skills_match_score	project_count	resume_length	github_activity
years_experience	1.00	0.51	0.69	0.76	0.83
skills_match_score	0.51	1.00	0.38	0.39	0.44
project_count	0.69	0.38	1.00	0.52	0.87
resume_length	0.76	0.39	0.52	1.00	0.63
github_activity	0.83	0.44	0.87	0.63	1.00

Plot: Education vs Shortlisted status



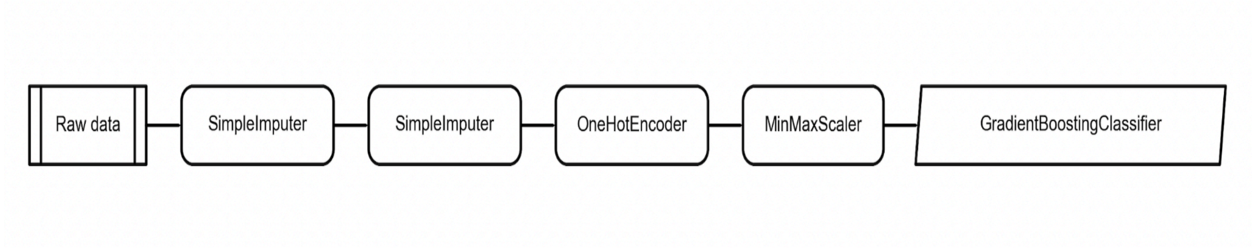
Plot: Skills match score vs Shortlisted status



Model Selection

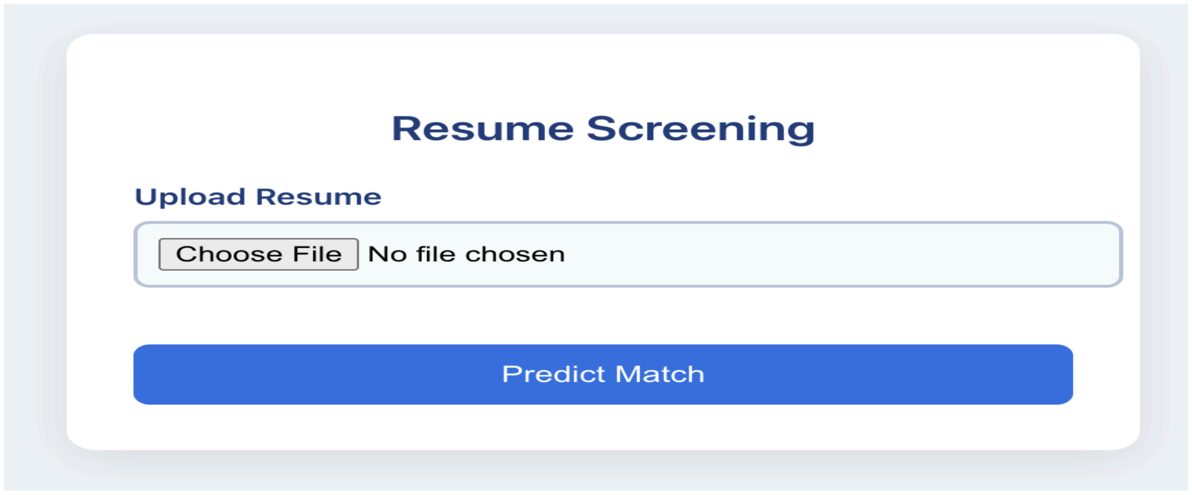
Out of the available model, GradientBoostingClassifier provided maximum accuracy and AUC, and hence it was selected.

```
[GradientBoostingClassifier(ccp_alpha=0.0, criterion='friedman_mse', init=None,
                           learning_rate=0.1, loss='log_loss', max_depth=3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0, min_samples_leaf=1,
                           min_samples_split=2, min_weight_fraction_leaf=0.0,
                           n_estimators=100, n_iter_no_change=None,
                           random_state=1101, subsample=1.0, tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False),
AdaBoostClassifier(algorithm='SAMME.R', estimator=None, learning_rate=1.0,
                   n_estimators=50, random_state=1101),
LGBMClassifier(boosting_type='gbdt', class_weight=None, colsample_bytree=1.0,
          importance_type='split', learning_rate=0.1, max_depth=-1,
          min_child_samples=20, min_child_weight=0.001, min_split_gain=0.0,
          n_estimators=100, n_jobs=-1, num_leaves=31, objective=None,
          random_state=1101, reg_alpha=0.0, reg_lambda=0.0, subsample=1.0,
          subsample_for_bin=200000, subsample_freq=0)]
```

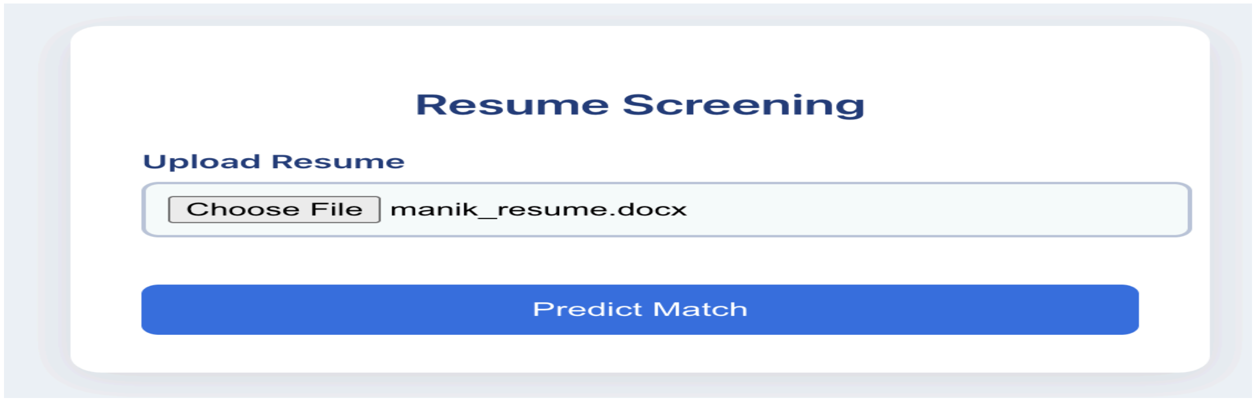


Project Demo screenshots

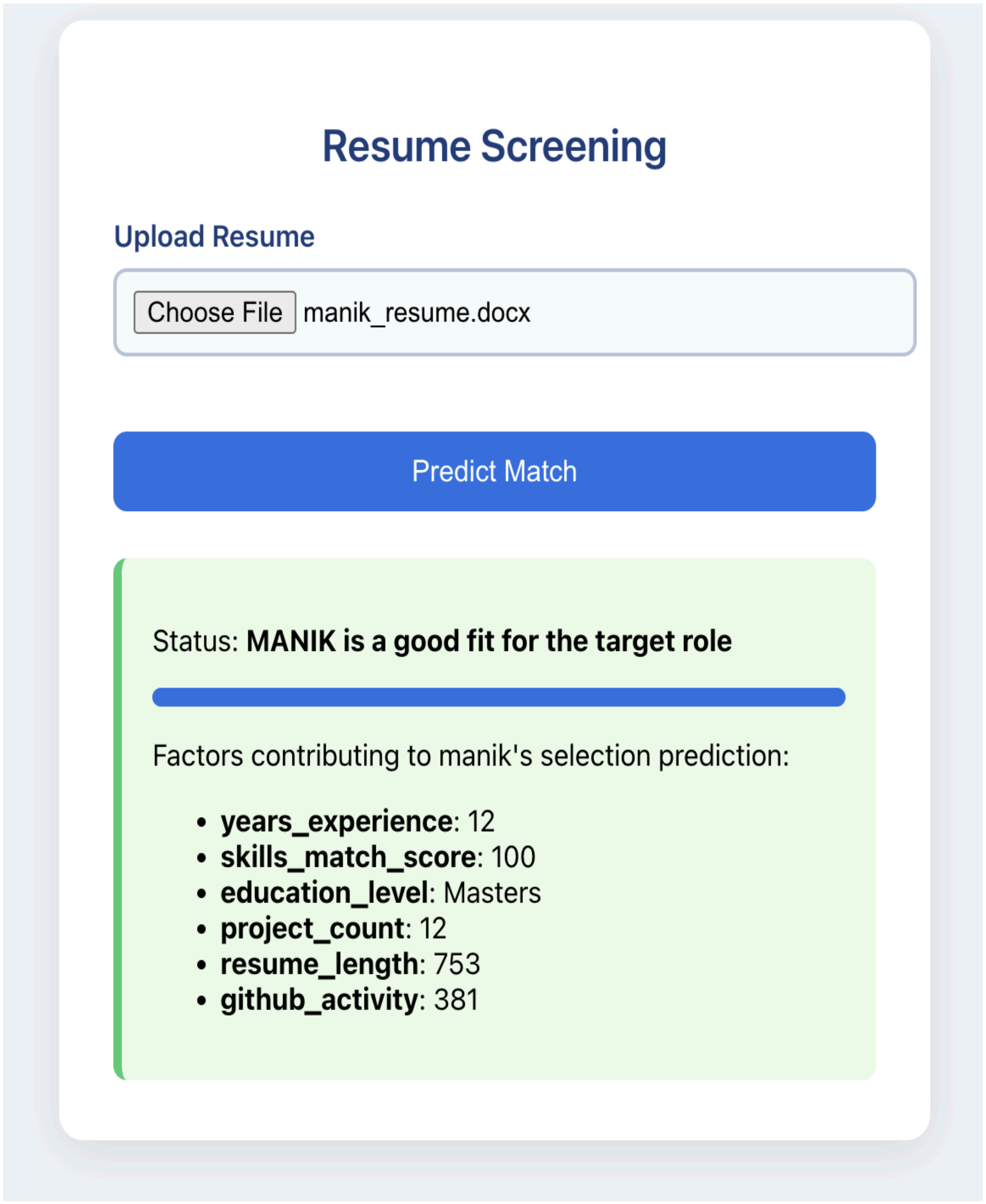
Landing Page



Resume Upload



Prediction for selected candidate


Resume Screening

Upload Resume

Choose File manik_resume.docx

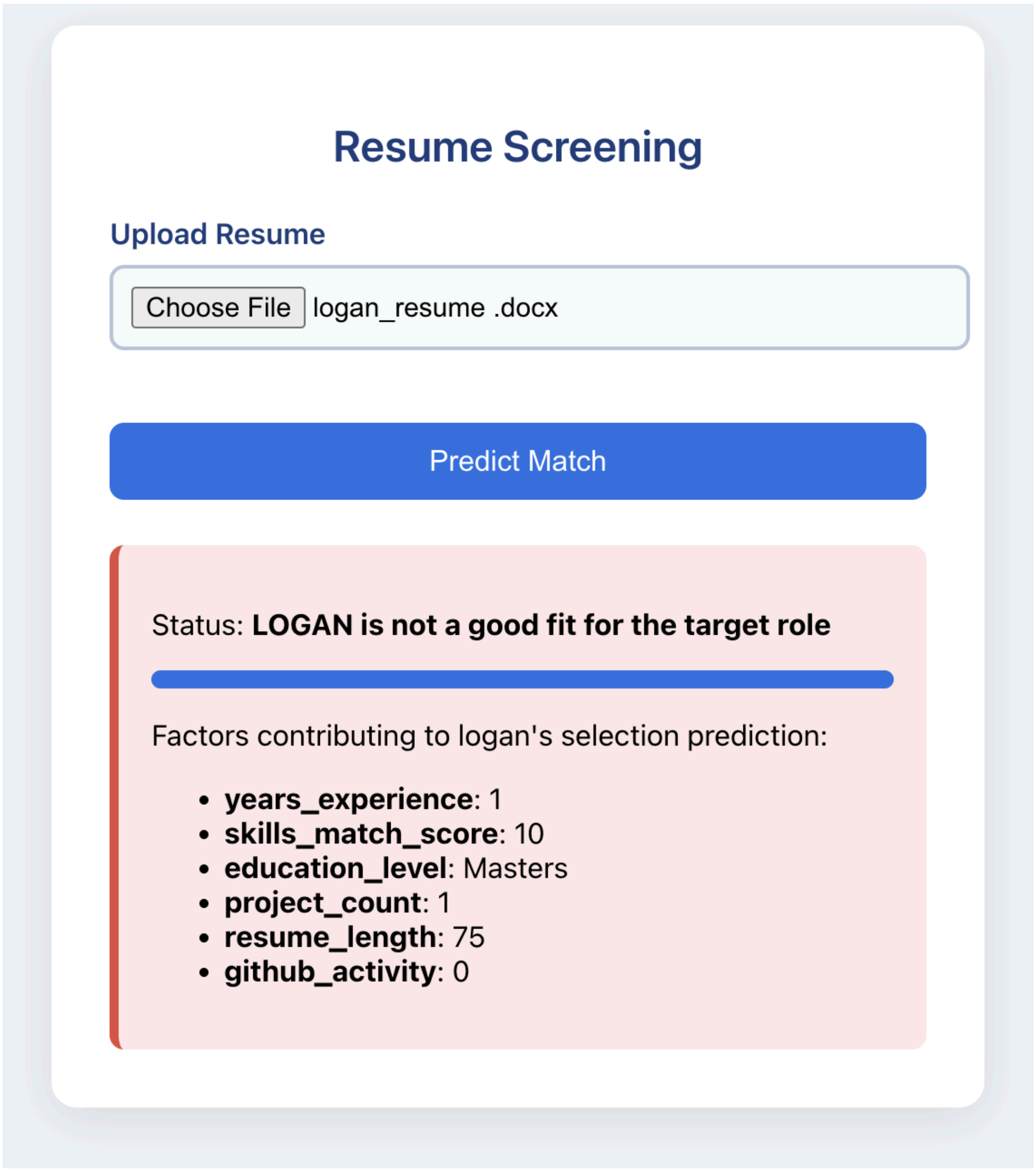
Predict Match

Status: **MANIK is a good fit for the target role**

Factors contributing to manik's selection prediction:

- **years_experience**: 12
- **skills_match_score**: 100
- **education_level**: Masters
- **project_count**: 12
- **resume_length**: 753
- **github_activity**: 381

Prediction for rejected candidate



Status: **LOGAN is not a good fit for the target role**

Factors contributing to logan's selection prediction:

- **years_experience**: 1
- **skills_match_score**: 10
- **education_level**: Masters
- **project_count**: 1
- **resume_length**: 75
- **github_activity**: 0

Code Link: https://github.com/Manik-Jain/ai_resume_screener