National College of Ireland

Configuration Manual

MSc Research Project

MSc in Data Analytics

Kanisha Pathy Venkatapathy Balaji
Student ID: X23399872

School of Computing
National College of Ireland

Supervisor: Abid Yaqoob

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Kanisha Pathy Venkatapathy Balaji
Student ID:	X23399872
Programme:	Msc in Data Analytics **Year:** 2025-2026
Module:	MSc Research Practicum
Supervisor:	Abid Yaqoob
Submission Due Date:	28/01/2026
Project Title:	Algorithmic Fairness in AI Recruitment Systems: Detecting and Correcting Bias in Resume Screening
Word Count:	19 **Page Count**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Kanisha Pathy Venkatapathy Balaji
Date:	28/01/2026

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	✓
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	✓
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	✓

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Kanisha Pathy Venkatapathy Balaji

1. System Configuration

1.1 Hardware Environment

The proposed ATS resume screening system was implemented and evaluated using the following infrastructure:

- The Processor used Intel Core i7
- RAM: 16 GB
- GPU Optional but used for efficiency for Tab-Transformer model acceleration.
- Storage: 512 GB SSD for fast dataset access and model checkpoints

This configuration ensured efficient execution of preprocessing, clustering, deep learning model training, and interactive deployment without system latency.

2. Software Requirements

To execute the project, the following software environment is required:

PROGRAMMING LANGUAGE:	**PYTHON 3.5+**
DEVELOPMENT IDES:	**JUPYTER NOTEBOOK**
VISUAL ANALYTICS TOOLS:	**POWER BI (DASHBOARD CREATION)** **STREAMLIT**
DEPLOYMENT & INTERACTIVE INTERFACE: STREAMLIT FOR RESUME SCREENING APP	

3. Libraries

This system leverages a combination of deep learning, NLP, visualization, fairness analysis, and deployment libraries. The key libraries include:

Ai Arithmetic Fairness in resume Screening

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import re
import itertools
import gender_guesser.detector as gender
import seaborn as sns
import ast
from collections import Counter
import torch
import shap
import torch.nn as nn
import torch.optim as optim
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import optuna
import joblib
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.metrics import (
    roc_curve, auc,
    precision_recall_curve, average_precision_score,
    confusion_matrix
)
import streamlit as st
```

Figure 1. Import all libraries

4. Folder Structure

This is the root folder structure below consisting of all the demo artifacts including code, dataset, report and configure manual.

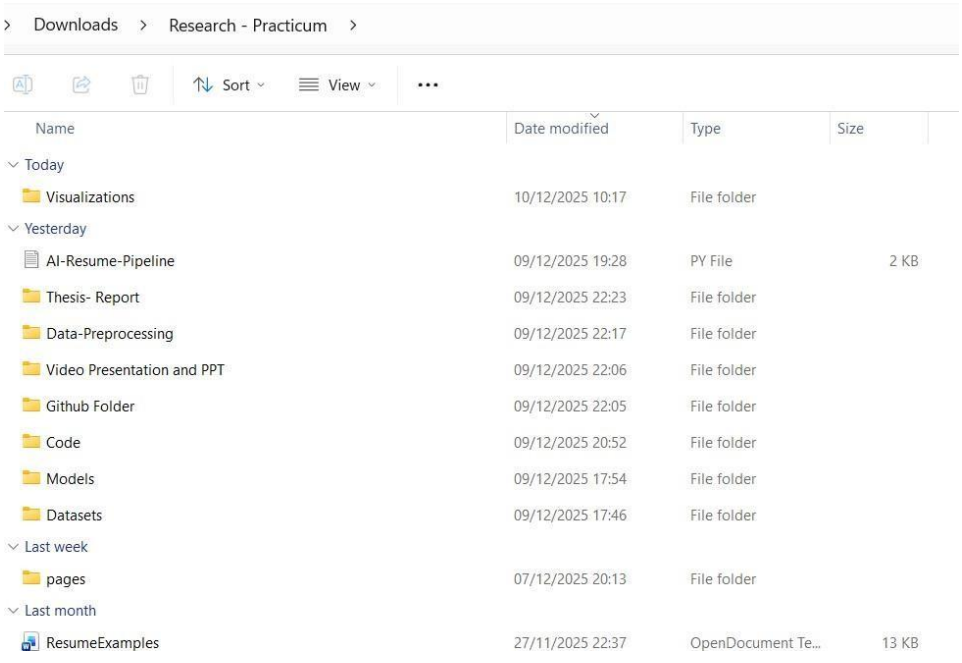


Figure 2. Research -Practicum Folder

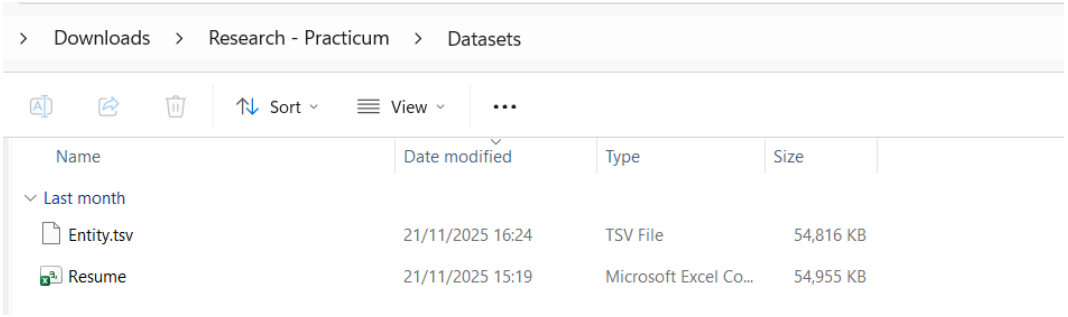


Figure 3. Datasets are inside this Folder

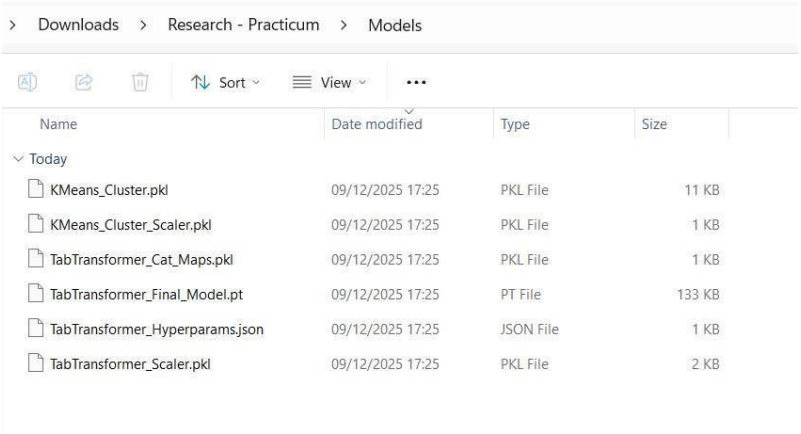
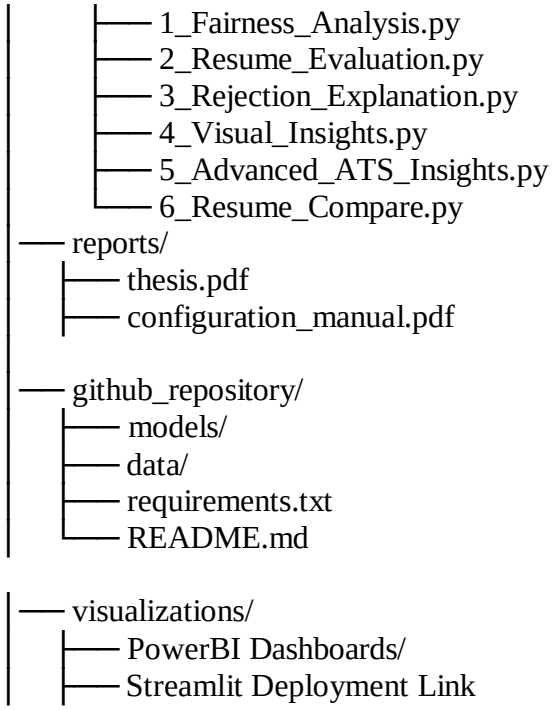


Figure 4. Models are downloaded in this folder

A. Directory:
The root folder directory structure including all files are given below,

```
Research Practicum (Root Project Folder)
├── datasets/
│   ├── Resume
│   └── Entity.tsv
├── data_preprocessing/
│   ├── Resume_ATS_Updated.csv
│   ├── Resume_ATS_Cleaned.csv
│   ├── Resume_ATS_Corrected.csv
│   └── Resume_ATS_Final.csv
├── models/
│   ├── TabTransformer_Final_Model.pt
│   ├── TabTransformer_Cat_Maps.pkl
│   ├── TabTransformer_Scaler.pkl
│   ├── TabTransformer_Hyperparams.json
│   ├── KMeans_Cluster.pkl
│   └── KMeans_Cluster_Scaler.pkl
├── code/
│   ├── AI_Resume_Fairness_Screening.ipynb
│   └── Supporting Python Scripts

├── streamlit_app/
│   ├── AI_Resume_Pipeline.py
│   └── pages/
```



5. Workflow to replicate the experiment

I. Setting Up Environment

• Ensure that all the required software and Python libraries are installed.
• Open Jupyter Notebook.
• Set the working directory to the folder containing the dataset files.

II. Data Collection and Augmentation Primary dataset

The datasets were collected from Kaggle and stored it in the Research practicum >Datasets folder> as mentioned in the above figure 3. Here are the two datasets links mentioned.
1. https://www.kaggle.com/datasets/snehaanbhawal/resume-dataset.
2. https://www.kaggle.com/datasets/raoofnaushad/resume-named-entities- csv.

III. Data Pre Processing

Resume data required substantial preprocessing due to inconsistency in structure and formatting. The preprocessing stage involved the following steps are text Extraction and HTML tags, PDF metadata, embedded formatting, tables, headers, and footers were removed. The extraction process focused on preserving pure textual content.

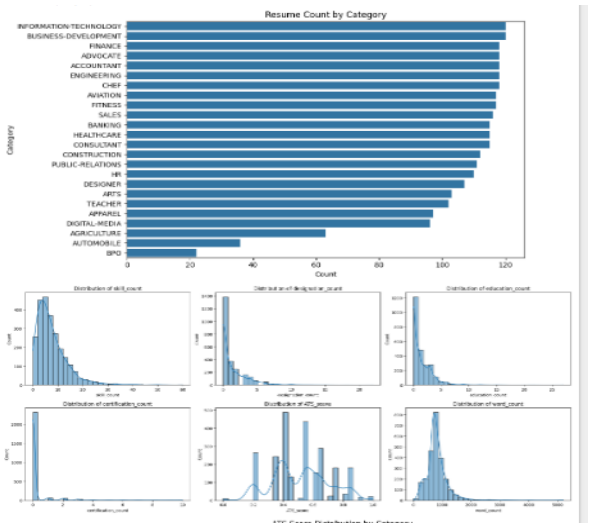
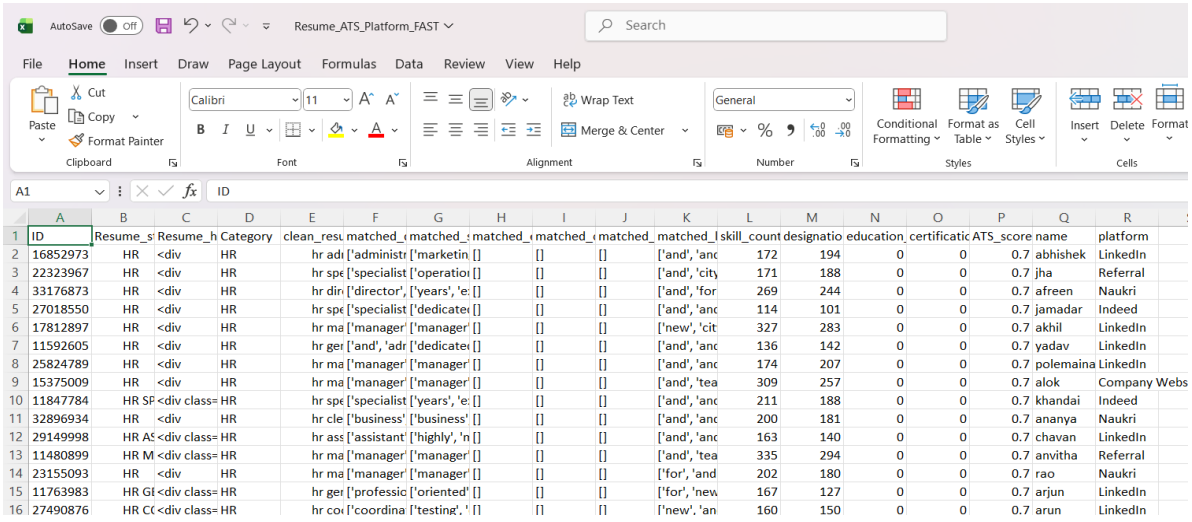


Figure 5.Exploratory Data Analysis on Category ,Platform, education and skills

The datasets were not concatenated but were integrated through entity-to-text matching, where tokens from the entity dictionary were extracted from resume text to derive structured ATS features, effectively merging both datasets into a unified modelling table.

```
resume_df["matched_designations"] = resume_df["clean_resume"].str.findall(regex_designation)
resume_df["matched_skills"] = resume_df["clean_resume"].str.findall(regex_skills)
resume_df["matched_education"] = resume_df["clean_resume"].str.findall(regex_education)
resume_df["matched_certifications"] = resume_df["clean_resume"].str.findall(regex_cert)
resume_df["matched_companies"] = resume_df["clean_resume"].str.findall(regex_company)
resume_df["matched_locations"] = resume_df["clean_resume"].str.findall(regex_location)

#  6. FEATURE COUNTS
resume_df["skill_count"] = resume_df["matched_skills"].apply(len)
resume_df["designation_count"] = resume_df["matched_designations"].apply(len)
resume_df["education_count"] = resume_df["matched_education"].apply(len)
resume_df["certification_count"] = resume_df["matched_certifications"].apply(len)
```

Figure 6. Merging both datasets



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	ID	Resume_s	Resume_h	Category	clean_resu	matched_	matched_	matched_	matched_	matched_	matched_l	skill_count	designatio	education_	certificatic	ATS_score	name	platform
2	16852973	HR	<div	HR	hr adı	['administr	['marketin	[]	[]	[]	['and', 'anc	172	194	0	0	0.7	abhishek	LinkedIn
3	22323967	HR	<div	HR	hr spe	['specialist	['operatio	[]	[]	[]	['and', 'city	171	188	0	0	0.7	jha	Referral
4	33176873	HR	<div	HR	hr dir	['director',	['years', 'e	[]	[]	[]	['and', 'for	269	244	0	0	0.7	afreen	Naukri
5	27018550	HR	<div	HR	hr spe	['specialist	['dedicate	[]	[]	[]	['and', 'anc	114	101	0	0	0.7	jamadar	Indeed
6	17812897	HR	<div	HR	hr ma	['manager'	['manager'	[]	[]	[]	['new', 'cit	327	283	0	0	0.7	akhil	LinkedIn
7	11592605	HR	<div	HR	hr ger	['and', 'adr	['dedicate	[]	[]	[]	['and', 'anc	136	142	0	0	0.7	yadav	LinkedIn
8	25824789	HR	<div	HR	hr ma	['manager'	['manager'	[]	[]	[]	['and', 'anc	174	207	0	0	0.7	polemaina	LinkedIn
9	15375009	HR	<div	HR	hr ma	['manager'	['manager'	[]	[]	[]	['and', 'tea	309	257	0	0	0.7	alok	Company Webs
10	11847784	HR SP	<div class=	HR	hr spe	['specialist	['years', 'e	[]	[]	[]	['and', 'anc	211	188	0	0	0.7	khandai	Indeed
11	32896934	HR	<div	HR	hr cle	['business'	['business'	[]	[]	[]	['and', 'anc	200	181	0	0	0.7	ananya	Naukri
12	29149998	HR AS	<div class=	HR	hr ass	['assistant	['highly', 'n	[]	[]	[]	['and', 'anc	163	140	0	0	0.7	chavan	LinkedIn
13	11480899	HR M	<div class=	HR	hr ma	['manager'	['manager'	[]	[]	[]	['and', 'tea	335	294	0	0	0.7	anvitha	Referral
14	23155093	HR	<div	HR	hr ma	['manager'	['manager'	[]	[]	[]	['for', 'and	202	180	0	0	0.7	rao	Naukri
15	11763983	HR GI	<div class=	HR	hr ger	['professic	['oriented'	[]	[]	[]	['for', 'new	167	127	0	0	0.7	arjun	LinkedIn
16	27490876	HR C(	<div class=	HR	hr coı	['coordina	['testing', '	[]	[]	[]	['new', 'an	160	150	0	0	0.7	arun	LinkedIn

Figure 7. Final dataset after Concatenation

IV. Model Execution

A. K-means Clustering

The strong and weak labels by doing K- Means clustering, the data was divided using 80-20 stratification, and hyperparameter optimization on number and weight of neighbors.

Figure 8. Label distribution using K means Clustering

B. Tab Transformer

The main **predictive engine** is constructed using a **Tab Transformer** model combining embeddings with standardized numerical ATS features. Transformer encoders allow the system to grasp feature interactions that conventional tabular models frequently miss. The model undergoes training with an **80/20** division to maintain the ratio of strong and weak resumes, in both sets guaranteeing an unbiased assessment.

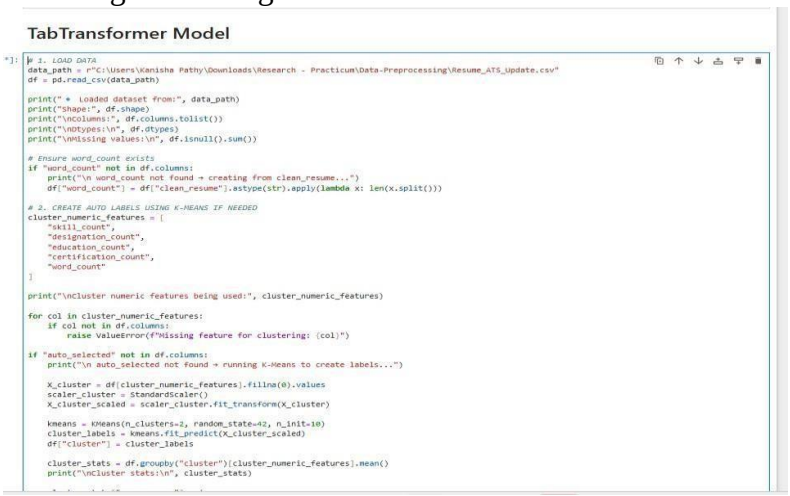


Figure 9. Loaded the Tab-Transformer model

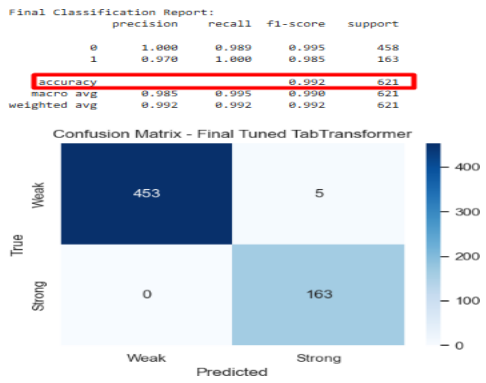


Figure 10. Classification Report and confusion matrix with F1 score 0.99 accuracy

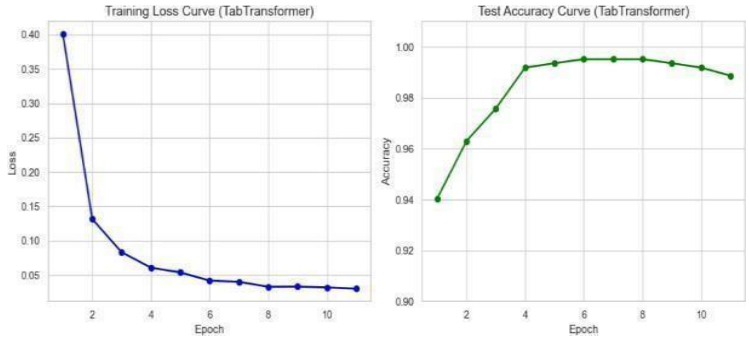


Figure 11. Training loss and Test Accuracy Curve

C. Explainability Shap And Fairness

Explainability and fairness are addressed through a dedicated module where SHAP is applied to generate global and local interpretability outputs, including summary plots and force plots that highlight individual feature contributions. Fairness metrics specifically selection rate and disparate impact are computed across non-sensitive groups such as job category and resume platform to assess whether the system exhibits any unintentional bias.

shap

```
# SHAP FOR TABTRANSFORMER (PyTorch) USING KernelExplainer
print("\n🚀 Starting SHAP KernelExplainer for TabTransformer...")
# 1. BUILD PREDICT FUNCTION

def model_predict(x_numpy):
    """
    x_numpy -> shape (N, num_features)
    Splits into numeric + categorical, returns probability predictions.
    """

    num_dim = X_num_train.shape[1]
    cat_dim = X_cat_train.shape[1]

    # split into numeric + categorical tensors
    x_num = torch.tensor(x_numpy[:, :num_dim], dtype=torch.float32).to(device)
    x_cat = torch.tensor(x_numpy[:, num_dim:].astype(np.int64), dtype=torch.long).to(device)

    final_model.eval()
    with torch.no_grad():
        logits = final_model(x_num, x_cat)
        probs = torch.softmax(logits, dim=1)

    return probs.cpu().numpy()  # (N, 2)


# 2. BACKGROUND DATA FOR SHAP (50 rows recommended)

print("✓ Preparing background dataset...")

background_size = 50
background = np.concatenate([
    X_num_train[:background_size],
    X_cat_train[:background_size]
], axis=1)  # shape = (50, 8)


# 3. INITIALIZE SHAP KernelExplainer

print("✓ Initializing SHAP KernelExplainer (may take 10-20 seconds)...")

explainer = shap.KernelExplainer(
    model_predict,
    background
)
```

```
Showing SHAP Summary Plot...
C:\Users\Kanisha Pathy\AppData\Local\Temp\ipykernel_46996\745907856.py:87: FutureWarning: The NumPy global RNG was seeded by calling `np.random.seed
`. In a future version this function will no longer use the global RNG. Pass `rng` explicitly to opt-in to the new behaviour and silence this warnin
g.
  shap.summary_plot(
```

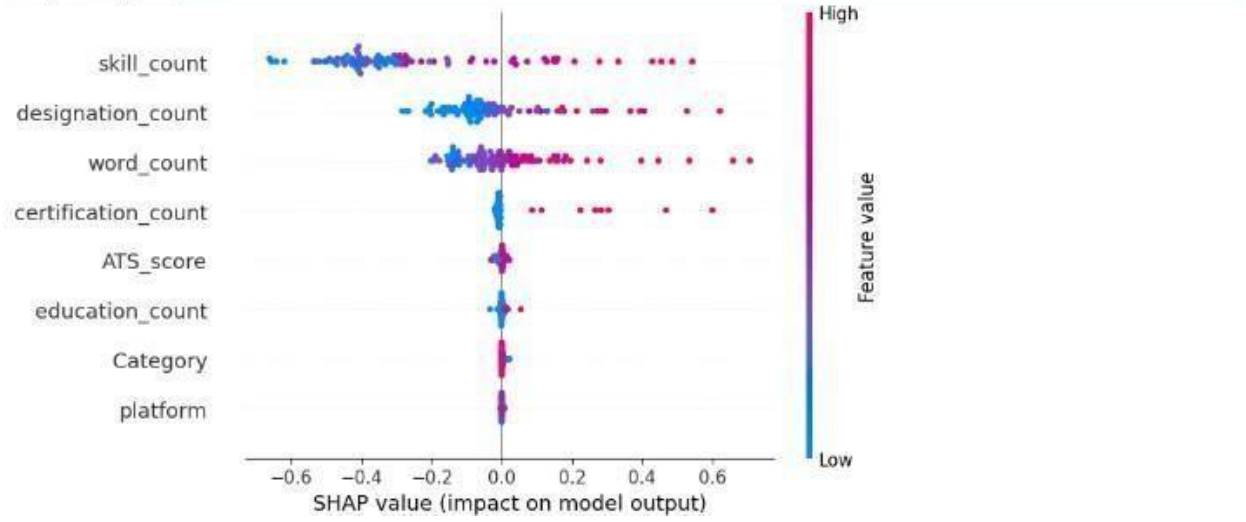


Figure 12. The SHAP summary plot visualizes how skill count, word count, ATS score contributes to the model's predictions across all resumes, showing both direction and magnitude of influence

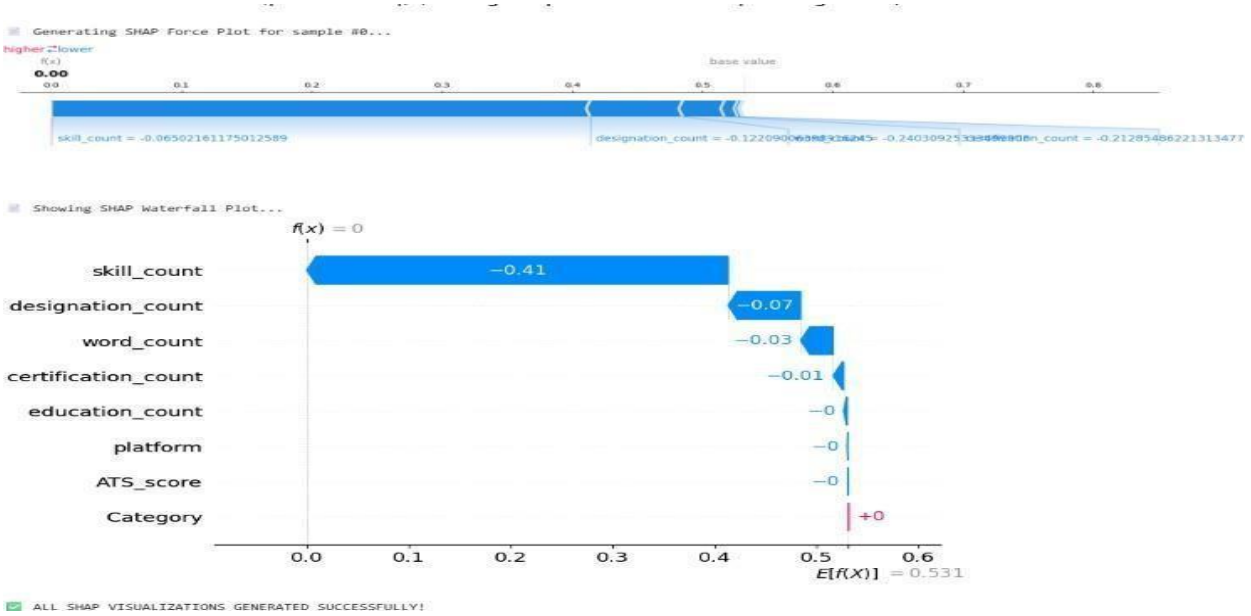


Figure 13. The SHAP force and waterfall plots explain the model's reasoning for a single resume by showing which features pushed the prediction toward strong or weak classification.

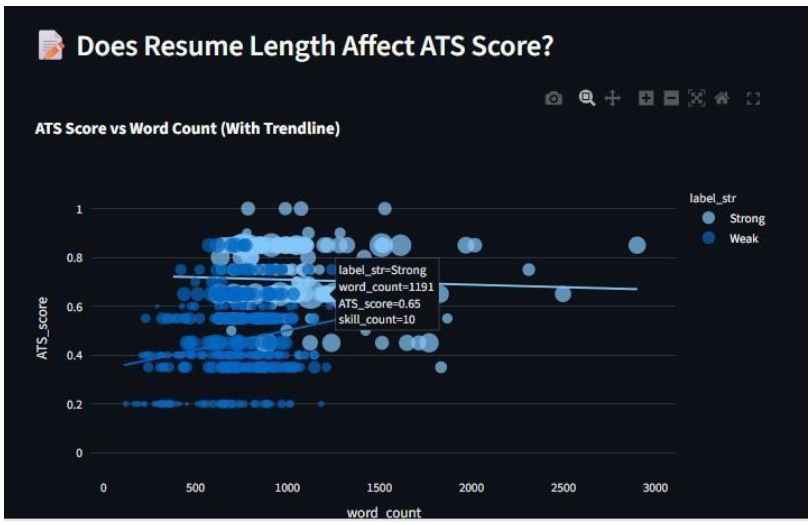


Figure 14.This image decribes how does the Resume Length afftect ATS score

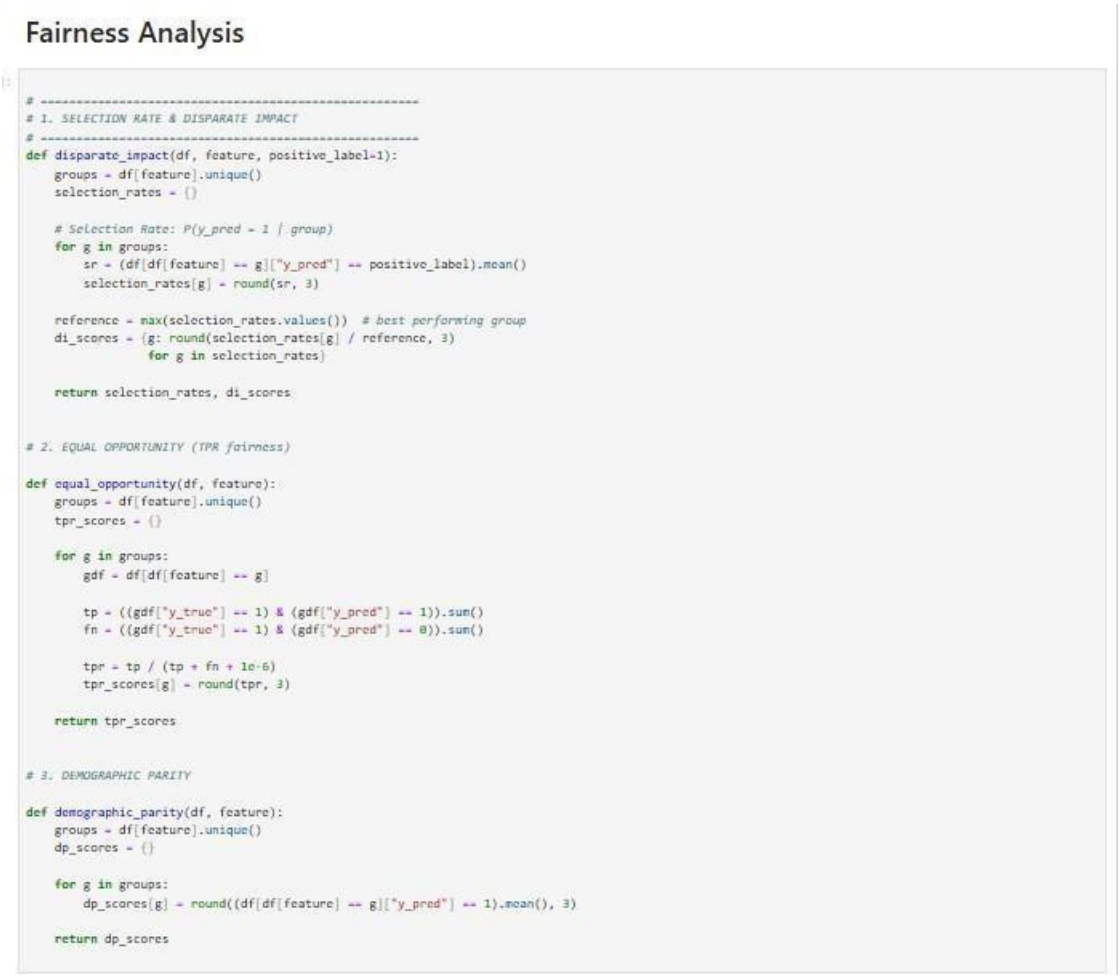


Figure 15. Fairness metrics specifically selection rate and disparate impact are computed across non-sensitive groups such as job category and resume platform to assess whether the system exhibits any unintentional bias.

V. Model Evaluation

```
===== FINAL TUNED TABTRANSFORMER RESULTS =====
Best Test Accuracy: 0.9935587761674718

Final Classification Report:
              precision    recall  f1-score   support

           0      0.998     0.991     0.995       458
           1      0.976     0.994     0.985       163

    accuracy                          0.992       621
   macro avg      0.987     0.993     0.990       621
weighted avg      0.992     0.992     0.992       621
```

Figure 16. Classification report and F1 score with accuracy

Steps	Method Used	Function Performed	Results
1.Labelling Layer	K-Means Clustering	Automatically determined strong vs weak resumes	Fair, scalable pseudo-labels replacing missing HR ground truth
2.Prediction Layer	Tab Transformer	Learned resume quality patterns and predicted strengths	Accuracy-0.99 & generalization, outperforming baselines
3.Explainability Layer	SHAP	Analyzed feature contribution to predictions	Transparent reasoning enabling candidate Improvements in selection.
4.Fairness &	Bias Auditing	Verified unbiased outcomes and made the system interactive	Fairness done for the platform, categories
5.Deployment Layer	Streamlit	Five analysis page developed and performed through interactive screening	interpretable, usable AI Resume screening solution for both candidates and HR

Fairness

Figure 17. Fairness analysis among platform,Category features using SR, DI and TRP.

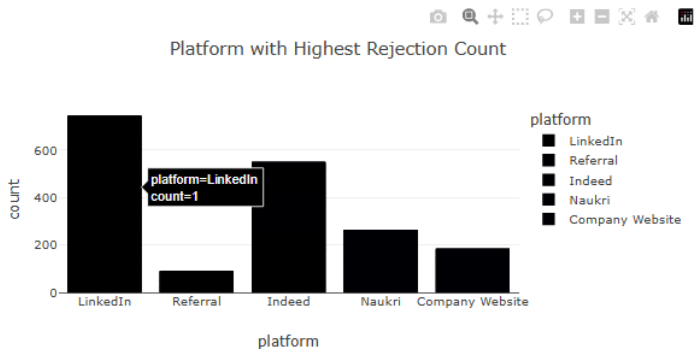


Figure 18.Fairness analysis among platform where the rejection happens more.

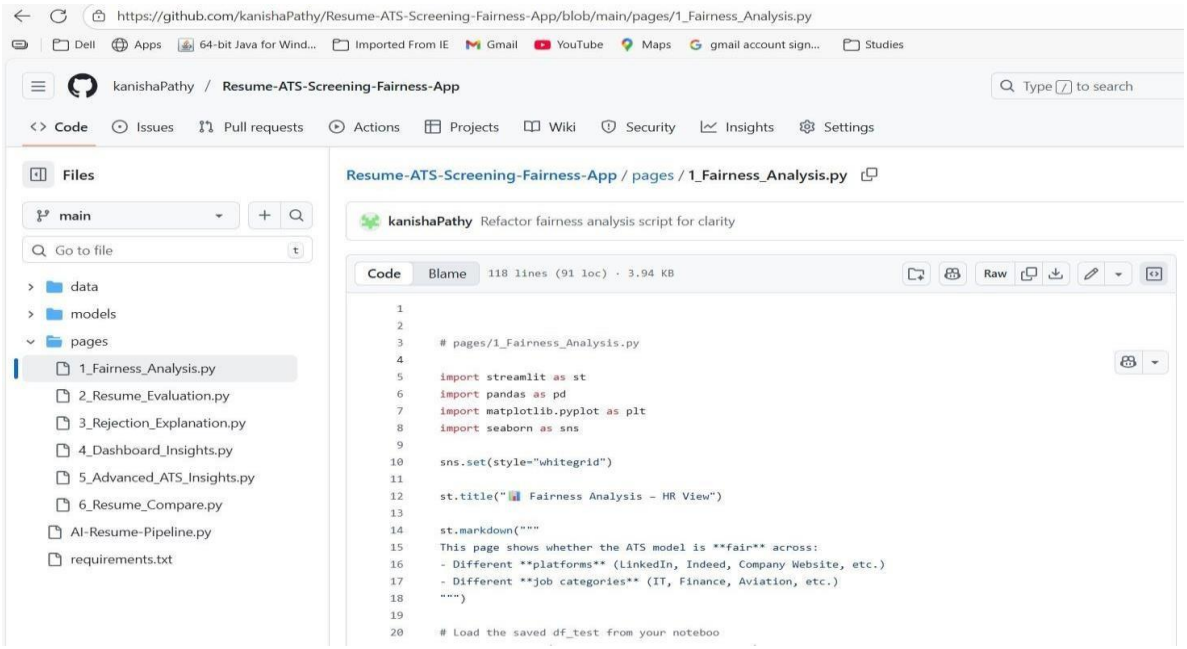
VI. Deployment

Run Streamlit locally using AI- Resume-Pipeline.py:

```
(base) C:\Users\Kanisha Pathy>cd "C:\Users\Kanisha Pathy\Downloads\Research - Practicum"

(base) C:\Users\Kanisha Pathy\Downloads\Research - Practicum>streamlit run AI-Resume-Pipeline.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8525
  Network URL: http://192.168.0.220:8525
```

Figure 19. To run streamlit- Locally by using Ai-Resume-Pipeline.py which is in folder Research Practicum mentioned in figure 2.

Streamlit is developed in both locally and in deployed as public using github.Here is the github link: Resume-ATS-Screening-Fairness-App/data at main · kanishaPathy/Resume-ATS-Screening-Fairness-App



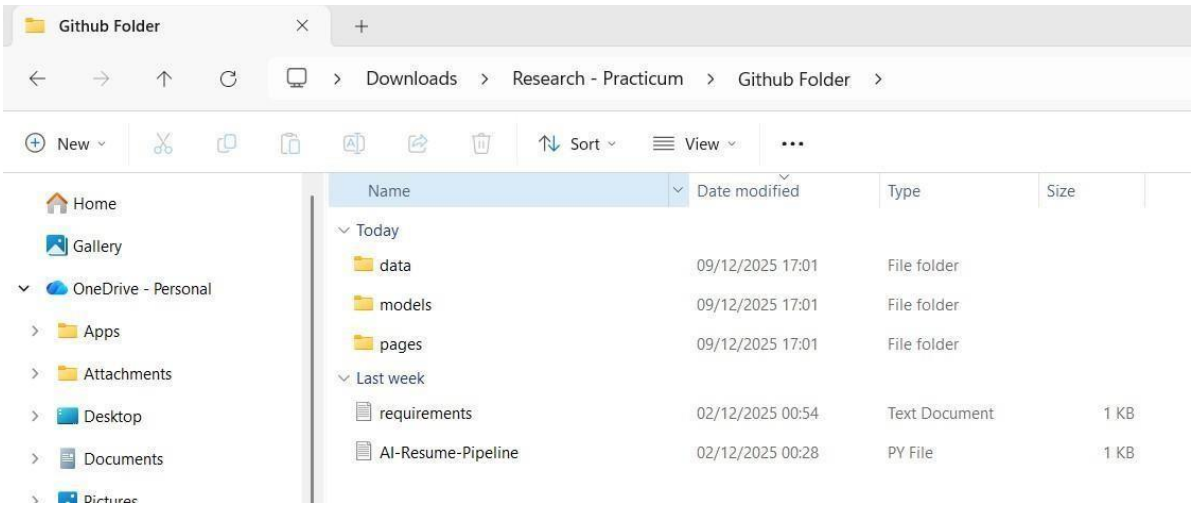


Figure 20. Git-hub code is used to deploy streamlit application. The code related to github is in t h i s git-hub folder

Streamlit – link [Explainable ATS Resume Screening · Streamlit](#).

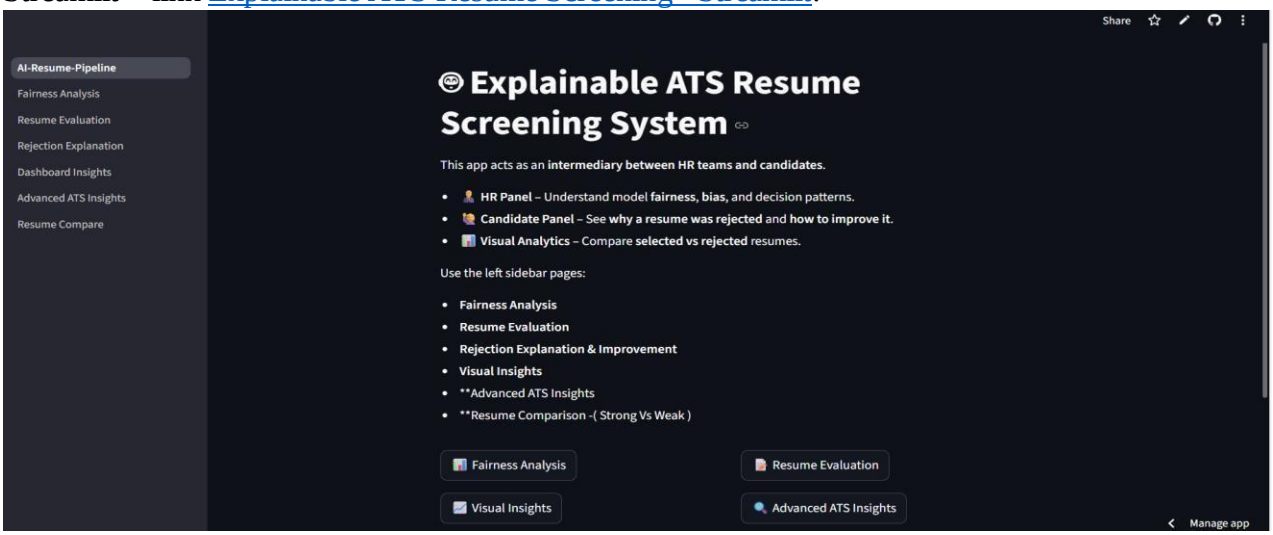


Figure 21. This is the homepage of the Resume Screening Application Interface

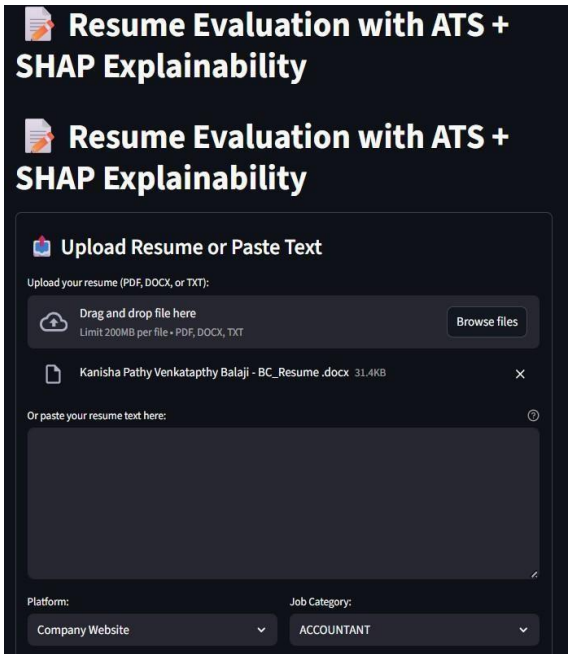


Figure 22. Resume Evaluation page evaluates the Resume and tells whether the resume got selected or rejected.

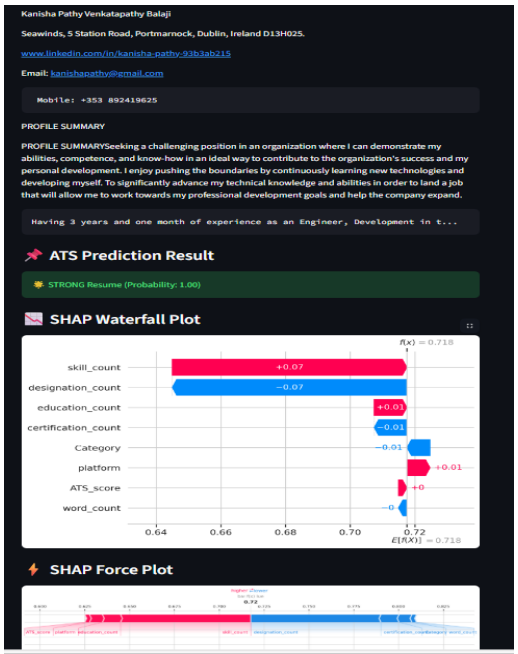


Figure 23.Shap plot visualizations of the Selected / rejected reusmes.

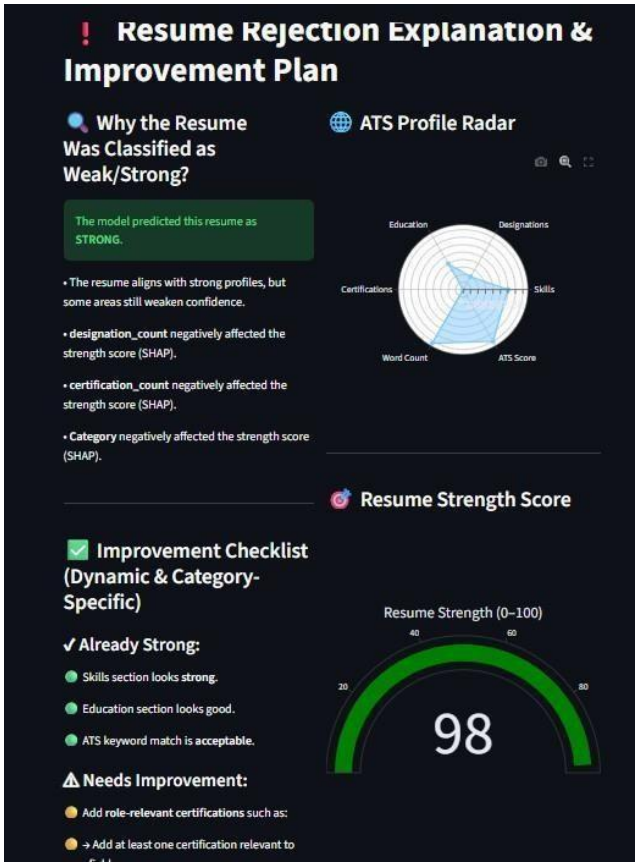


Figure 24. Resume rejection tells why the resume got rejected and give some suggestions for the resume to get selected.

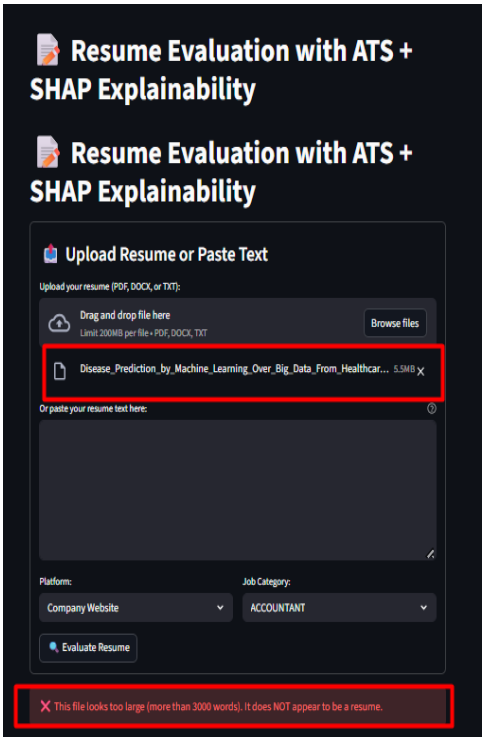


Figure 25.This image tells about the validation error it checks if the uploaded file is resume file or not if it is not resume file it shows error.

6. Troubleshooting:

A number of issues can be encountered during the development and deployment process. The common conflicts and their resolutions are summarised below:

1. Missing Model Files or Wrong Paths

Streamlit or notebook fails to load .pt, .pkl, or .json files.

- Make sure all the artifacts - model weights, scalers, cat-maps - are in the correct /models/ directory
- Update file paths in code appropriately

2. Version Mismatch Across Libraries

Incompatible versions of PyTorch, Pandas, or Streamlit may be running on different systems, creating import or runtime errors.

- Use a virtual environment
- Lock dependencies with requirements.txt

3. Variability in K-Means Assignments

Clustering labels will slightly change with runs because of random initialization.

- Set a fixed random_state, such as: random_state=42
- Save cluster labels to avoid recomputation

4. Streamlit Page Switching Errors - st.switch_page() fails when page names change or structure changes.

- Store Streamlit pages inside /pages/
- Verify that file names are exactly alike, except for a numeric prefix

5. Incorrect feature names when saving models
Model fails at inference because data frame column names differ from training column names.

- Keep a config dictionary that details the expected feature sets
- Standardize preprocessing during training and prediction

6. SHAP Plot Rendering Lag or Failure

Large datasets lead to slow computation of shap values or UI crashes.

- Run SHAP sampling, e.g., use 200–500 instances only
- Cache results and reuse instead of recomputing on every load

7. Misclassification of Fairness Metric

DI values >1 or floating oscillations may confuse users.

- Correcting: Apply smoothing and clear interpretation rules
- Provide thresholds, such as DI < 0.80 indicates risk Deployment UI Not Displaying Correctly

Library conflicts

One of the common issues during the implementation process had to do with mismatches in the versions of Python packages, especially PyTorch, Pandas, Scikit-learn, Optuna, and Streamlit. Since these libraries change quite frequently, some functions behave differently across versions, which leads to import errors, changed APIs, or model loading, training, and UI deployments failing.