🔍 Project Title: AI-Powered Resume Category Prediction & Matching Tool Using NLP and Machine Learning

---

📘 Abstract

This project presents a data-driven solution for automating the resume screening process by classifying resumes into job categories and optionally matching them to job descriptions. Using Natural Language Processing (NLP) and Machine Learning (ML), the system extracts and cleans unstructured text from resumes (PDF, DOCX, TXT), transforms it into vectorized features, and predicts the most relevant job category using a Support Vector Classifier (SVC). The project further outlines future capabilities for skill extraction, job description similarity scoring, and candidate ranking. It is deployed as a user-friendly web application using Streamlit.

🎯 Problem Statement

Manual resume screening is time-consuming, inconsistent, and prone to human error. Organizations often receive hundreds of resumes, many with varied formats and structures. Recruiters need an intelligent, scalable, and unbiased system to automatically classify resumes and prioritize candidates based on job relevance.

🧠 Objective

- To **automatically classify resumes** into predefined job categories using ML.
- To build an interface that accepts resumes in common formats (PDF, DOCX, TXT).
- To extract relevant information and enable future features like skill matching and job description alignment.

🧰 Technologies Used

Category	Tools & Libraries
Programming Language	Python
Web Framework	Streamlit
NLP	Regex, SpaCy (planned), NLTK (optional)
File Parsing	PyPDF2, python-docx
Machine Learning	Scikit-learn (SVC, TF-IDF)
Vectorization	TF-IDF Vectorizer
Deployment (optional)	Streamlit Sharing / Heroku / Hugging Face

---

📁 Dataset Information

- **Source**: UpdatedResumeDataSet.csv (Manually curated or open-source resume dataset)
- **Attributes**:

  - Resume: Raw resume text
  - Category: Target label indicating job domain (e.g., Data Science, HR, Sales)

🔍 Data Preprocessing Steps

1. **Text Cleaning** using Regex:
   - Removed URLs, emails, special characters, emojis, hashtags.
   - Normalized whitespace and removed non-ASCII characters.
2. **Vectorization**:
   - TF-IDF vectorizer converts resume text into numerical features for ML.
   - Only the top n most informative features are retained.
3. **Label Encoding**:
   - Encoded target categories into numerical labels using LabelEncoder.

🤖 Model Development

- **Model**: Support Vector Classifier (SVC)
- **Vectorizer**: TF-IDF (TfidfVectorizer)
- **Training Accuracy**: ~95%+ (based on initial testing)
- **Test Accuracy**: Evaluated using train_test_split and classification_report
- **Model Files Saved As**:
  - clf.pkl (classifier)
  - tfidf.pkl (vectorizer)
  - encoder.pkl (label encoder)

🌐 Web Application Features

Feature	**Description**
📁 Resume Upload	Upload resumes in PDF, DOCX, or TXT format
🧹 Text Extraction	Extracts and cleans text using file-specific parsers
🔮 Category Prediction	Classifies the resume into the most suitable job category
📋 Show Extracted Text	Optional checkbox to preview raw resume content
✅ Instant Feedback	Predicted category is shown on the interface

---

📈 Planned Enhancements (Next Phase)

1. **Job Description Matching**:

    - Accept job description input
    - Calculate **cosine similarity** between JD and resume
    - Rank resumes accordingly
2. **Skill Extraction**:
    - Compare resume tokens with predefined skill list (CSV/DB)
    - Highlight matched/missing skills
3. **Candidate Ranking Dashboard**:
    - Upload multiple resumes
    - Display ranked list based on similarity scores
4. **NER-Based Parsing**:
    - Use SpaCy to extract:
        - Name (PERSON)
        - Education & Organizations (ORG)
        - Experience and years worked

📊 Evaluation Metrics

Metric	Description
Accuracy	Percentage of correct predictions
Precision	Correct positive predictions over total predicted positives
Recall	Correct positive predictions over actual positives
F1 Score	Harmonic mean of precision and recall

Evaluation done via classification_report and confusion matrix

---

📁 Folder Structure

bash

CopyEdit

```
📁 Resume_Parser_App/
│
├── app.py                # Streamlit app
├── main.ipynb            # Model training notebook
├── requirements.txt      # Required libraries
├── clf.pkl               # Trained model
├── tfidf.pkl             # TF-IDF vectorizer
├── encoder.pkl              # Label encoder
├── UpdatedResumeDataSet.csv  # Resume dataset
```

📌 Use Cases

- **Recruiters**: Automated pre-screening and filtering
- **HR Tools**: Integrate with internal hiring systems
- **Career Platforms**: Match candidates to job postings
- **Students**: Get feedback on resume relevance for roles

✅ Results Snapshot

Resume Filename	Predicted Category
john_doe.pdf	Data Science
priya_hr.docx	HR
amit_dev.txt	Web Development

🧠 Key Learnings

- Practical implementation of NLP for real-world tasks
- How to handle diverse document formats (PDF, DOCX)
- Deploying ML models using interactive UIs
- Importance of clean preprocessing and modular code

🔚 Conclusion

This project automates resume classification using NLP and ML, providing instant, accurate predictions. It improves recruiter efficiency and reduces screening errors. With planned enhancements like skill extraction and JD matching, the tool can evolve into a full-fledged intelligent recruitment assistant.

📚 References

- Scikit-learn Documentation
- Streamlit Docs
- SpaCy NER
- ResumeParser GitHub Projects