AI-Powered Resume Screening and Employee Attrition Prediction System

Automating HR Decisions Using NLP and Machine Learning

Submitted by:
Ashutosh Jaiswal

Problem Statement

1. Resume Screening: Develop an AI tool to filter resumes for a "Software Engineer" role by matching skills, experience, and qualifications from job descriptions.
2. Employee Sentiment Analysis: Analyze employee feedback (e.g., surveys, exit interviews) to predict attrition risks and recommend engagement strategies.

Problem Understanding

In most companies, hiring new people and keeping current employees happy is very important.

Right now, when companies get resumes for a job, someone has to read each one by hand. This takes a lot of time and can lead to mistakes or unfair choices. Good candidates might get missed, and hiring can take too long.

Also, sometimes good employees leave the company without warning. This is a problem because training new people takes time and money. If companies could know in advance who might leave, they could try to stop it.

So, the current way of doing these things is slow, not very smart, and doesn't use modern technology.

Proposed Solution

We plan to build a smart system using Artificial Intelligence (AI) to solve both problems:

1. Resume Screening System
This part will read and understand resumes automatically. It will match the skills and experience in the resume with what the job needs. Then it will give a score and tell who is the best fit. This saves time and avoids bias.

2. Attrition Prediction System
This part will look at past employee data to find patterns. It will predict if someone might leave the company soon. This helps HR take action early and keep good employees.

How the system helps:

- Saves time in hiring.
- Find the right candidates faster.
- Helps stop important employees from leaving.
- Uses smart tools like machine learning and data analysis.

Methodology

Both systems are designed as end-to-end pipelines that include data processing, machine learning, and a deployable API service, with integrated intelligence for actionable results.

AI-Powered Resume Screening System

This system scores incoming resumes and ranks them for specific job roles using natural language processing and machine learning.

1. Data Collection and Labeling

- Gather job descriptions and resumes from various sources.
- Uses similarity-based techniques to match resumes with suitable job roles.
- Labels data by calculating semantic similarity between resume content and job requirements.

2. Resume Text Processing

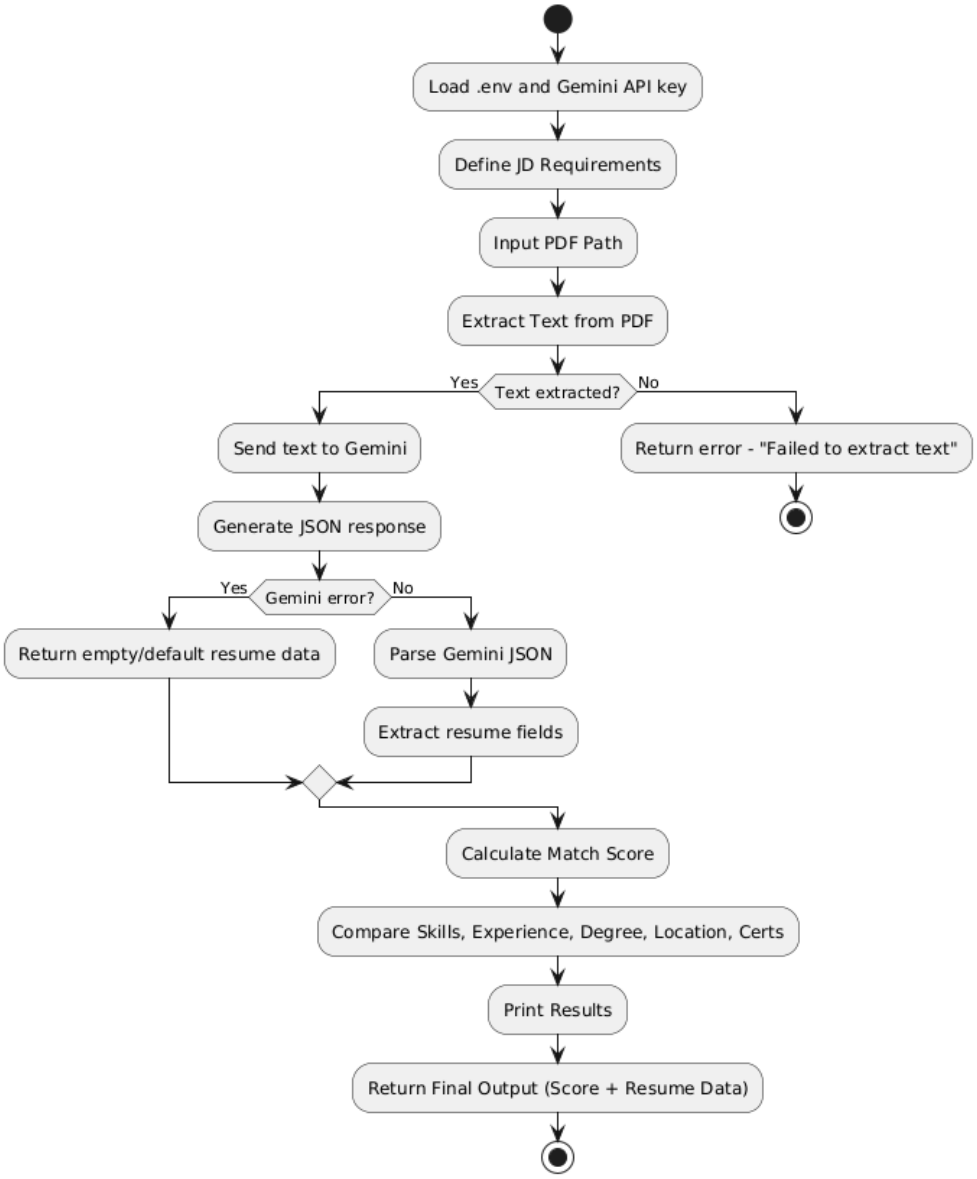
- Extracts textual content from PDF resumes.
- Applies natural language processing techniques such as tokenization, stopword removal, and lemmatization.
- Converts processed text into numerical vectors using TF-IDF embeddings depending on the model.

3. Scoring and Recommendation API

- Exposes an endpoint to accept resume and job description inputs.
- Preprocesses both inputs.
- Generates a job fit score and ranks resumes accordingly.
- Optionally returns AI-generated JSON summaries.

4. Automation and Integration

- Designed for integration into HR applicant tracking systems (ATS).
- Can run as a batch process or real-time service.



Flowchart for Resume Screening System

Employee Attrition Prediction System

1. **Data Preprocessing**

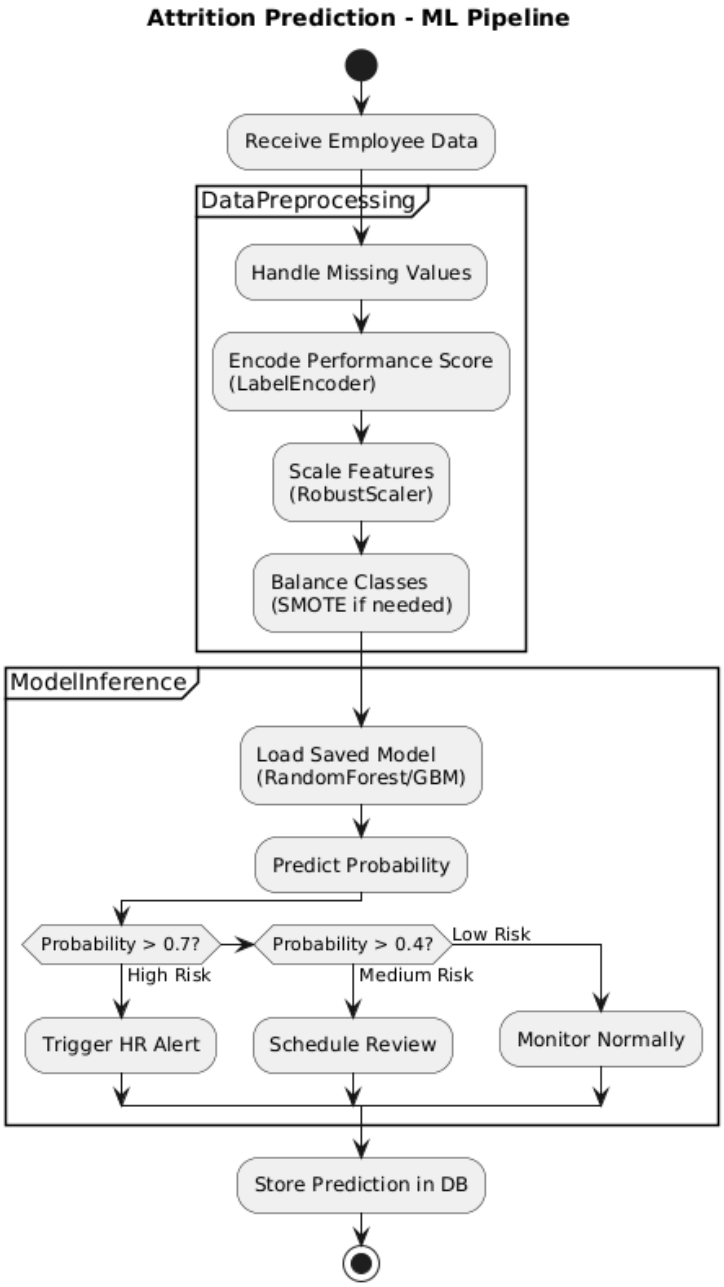
    - Combine employee data from multiple sources
    - Create binary target column: 1 if employee left, 0 if stayed
    - Select important features (engagement, performance, tenure, etc.)
    - Fill missing values (median for numbers, mode for categories)
    - Encode categorical features using LabelEncoder
    - Scale numeric features with RobustScaler
    - Split data into 80% training and 20% testing with stratified sampling
    - Save encoder and scaler for later use

2. **Model Training**
    - Handle class imbalance with SMOTE oversampling
    - Train Random Forest, Gradient Boosting, and Logistic Regression models
    - Use 5-fold stratified cross-validation
    - Evaluate models with AUC, F1, precision, recall, accuracy
    - Select best model based on AUC score
    - Analyze feature importance and risk distribution
    - Save trained model and metadata for deployment

3. **Model Deployment**
    - Containerize model and deploy on Azure ML Studio as REST API
    - API preprocesses input data similarly to training
    - API returns attrition probability and risk level (low, medium, high)

4. **Suggestion Generator**
    - Use Gemini AI to create personalized retention suggestions
    - Generate detailed prompts from employee data
    - Extract numbered suggestions from AI response
    - Implement fallback options if AI service fails

5. **Testing and Integration**
    - Test API with sample employee data representing different risk levels
    - Verify prediction accuracy, risk labels, and suggestions
    - Ensure smooth operation of entire system pipeline



Flowchart of Attrition Prediction

Prompts

Prompt for Extracting Resume Keywords (Using LLM like Gemini)

- Task: Analyze a resume and return a **strictly formatted JSON** with fields:
    1. `technical_skills`: list of hard technical skills only
    2. `total_experience`: total years of experience (float)
    3. `role_experience`: list of job roles with title, years, and organisation
    4. `education`: list of degrees with institution names
    5. `projects`: list of projects with names and tools used
    6. `certifications`: list of certifications
    7. `location`: city/state/country or "Unknown" if not found

- Rules:
    1. Include only hard technical skills (no soft skills)
    2. Calculate total years of experience from all roles
    3. Location must be set to "Unknown" if not present
    4. Strict adherence to field names and JSON format
- Input: Resume text (up to 3000 characters)

```
prompt = f"""

Analyze this resume and return EXACTLY the following JSON structure:

{{

    "technical_skills": ["list", "of", "technical", "skills"],

    "total_experience": years (float),

    "role_experience": [{{"title": "job_title", "years": years, "organisation": "organisation_name"}}],

    "education": [{{"degree": "degree_name", "institution": "school"}}],

    "projects": [{{"name": "project_name", "tools": ["list"]}}],

    "certifications": ["list"],

    "location": "city/state/country"

}}
```

```
Rules:

1. For skills: Include only hard technical skills (no soft skills)

2. For experience: Calculate total years across all roles

3. If location is not found, set it to "Unknown"

4. Be strict with field names and JSON formatting



Resume Text:

{resume_text[:3000]}

"""
```

Prompt for Employee Attrition Retention Strategies (Using LLM like Gemini)

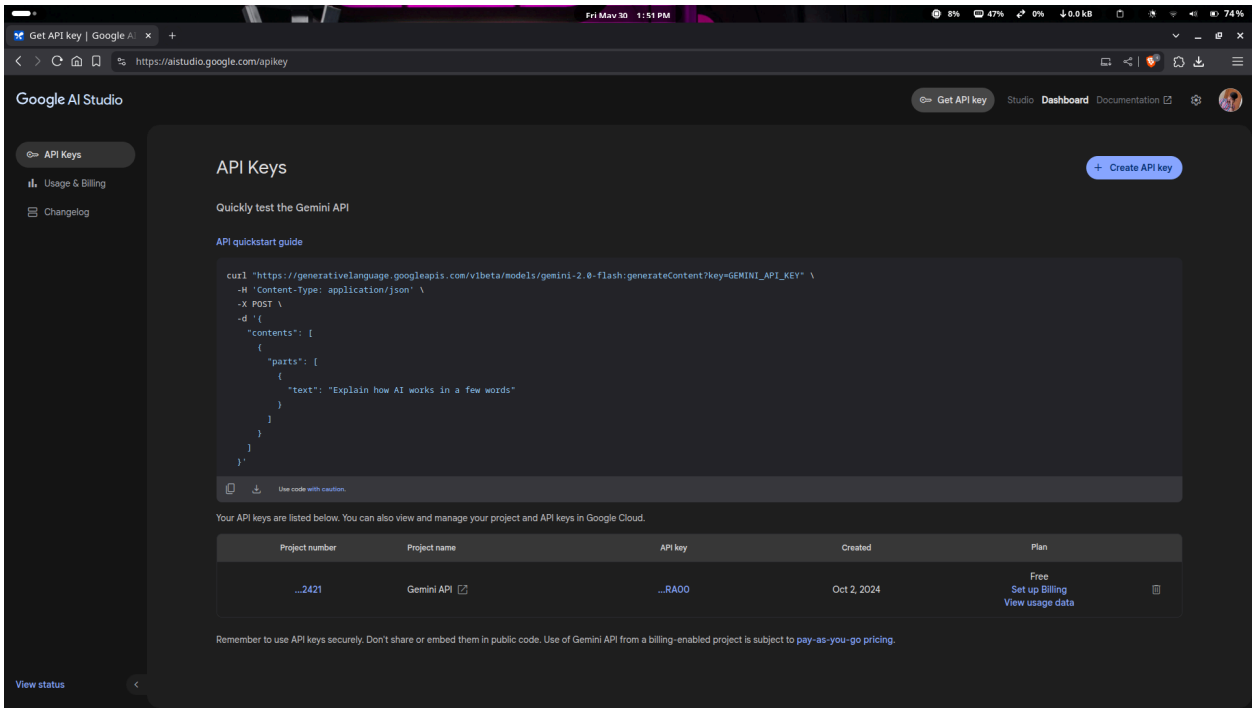
- Task: Act as HR expert and provide **3 specific, actionable retention strategies** based on employee data.
- Input data includes:
    - Attrition Risk Score (decimal)
    - Engagement Score (scale 1–5)
    - Satisfaction Score (scale 1–5)
    - Work-Life Balance (scale 1–5)
    - Tenure (years)
    - Current Rating (scale 1–5)
    - Performance Score (numeric)

- Instructions:
    - Provide exactly 3 numbered strategies
    - Strategies must be specific to the employee profile
    - Focus on the lowest scoring areas in the data
    - Actionable by HR or managers

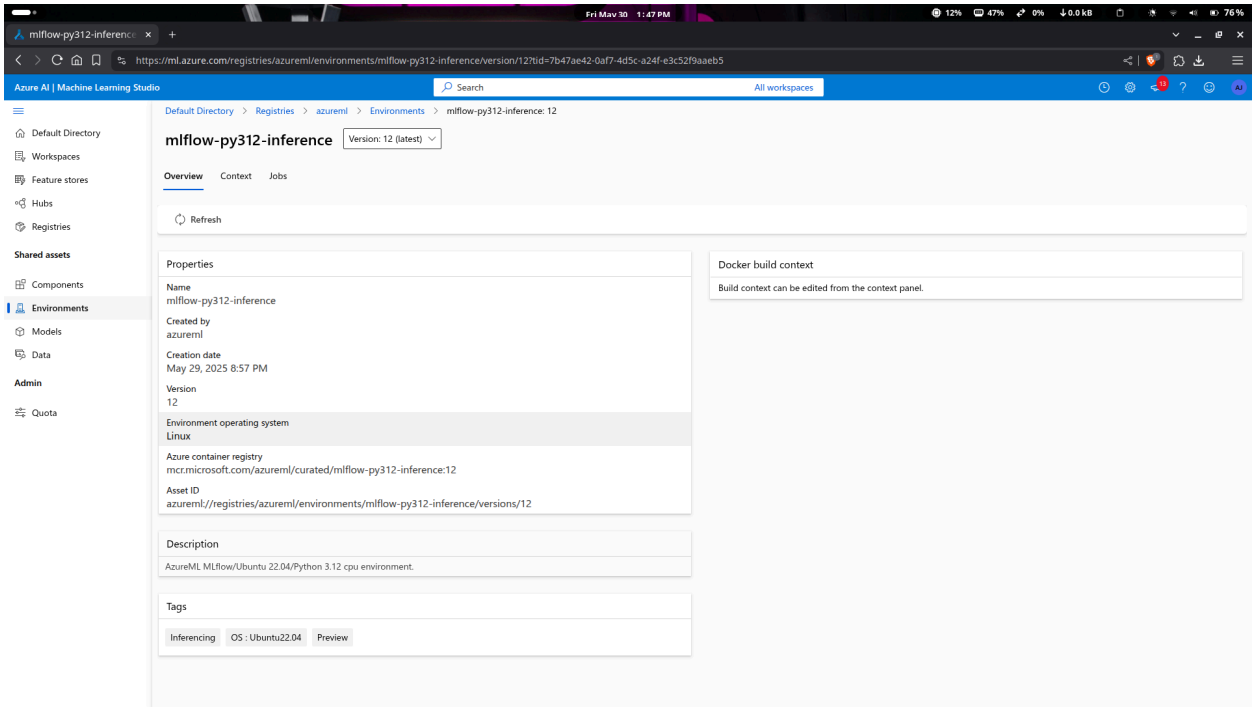
```
prompt = f"""
  You are an HR expert analyzing employee attrition risk. Based on the following employee data, provide 3 specific, actionable retention strategies.

  Employee Profile:
  - Attrition Risk Score: {employee_data['risk_score']:.2f}
  - Engagement Score: {employee_data['engagement_score']}/5
  - Satisfaction Score: {employee_data['satisfaction_score']}/5
  - Work-Life Balance: {employee_data['work_life_balance']}/5
  - Tenure: {employee_data['tenure']} years
  - Current Rating: {employee_data['current_rating']}/5
  - Performance Score: {employee_data['performance_score']}

  Provide exactly 3 numbered strategies that are:
  - Specific to this employee's profile
  - Actionable by HR/managers
  - Focused on the lowest scoring areas
  - Formatted as numbered list (1., 2., 3.)
  """
```

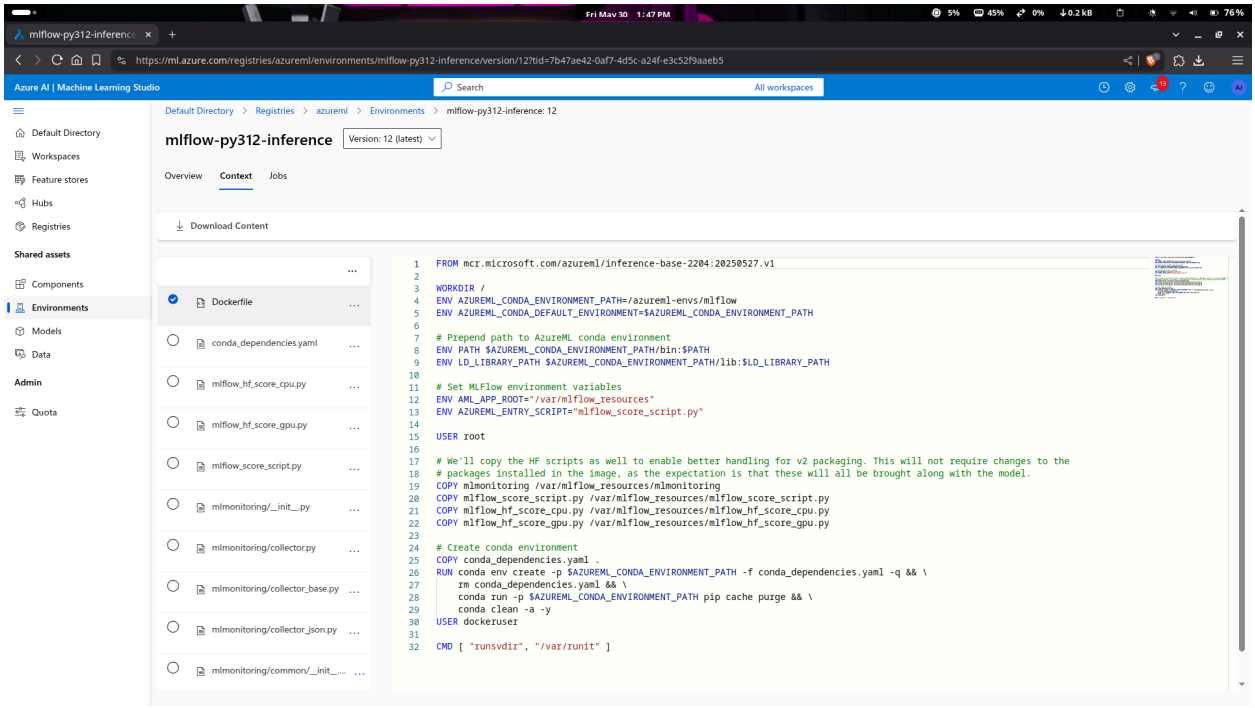
Challenges Faced and Resolutions

- **Model Deployment on Azure:** Faced difficulties in containerizing and exposing the model as a REST endpoint. Resolved by using Azure ML Studio's managed online endpoint feature and Docker-based deployment templates.
- **Data Selection and Preprocessing:** Initial data was inconsistent and noisy. Addressed by applying structured preprocessing scripts for cleaning, encoding, scaling, and balancing using SMOTE.
- **User-Friendly Integration:** Needed a seamless prediction interface. Solved by building a FastAPI wrapper to interact with the Azure endpoint and integrating a suggestion generator to enhance usability.



Google Gemini Api key screenshot





Deployment Details