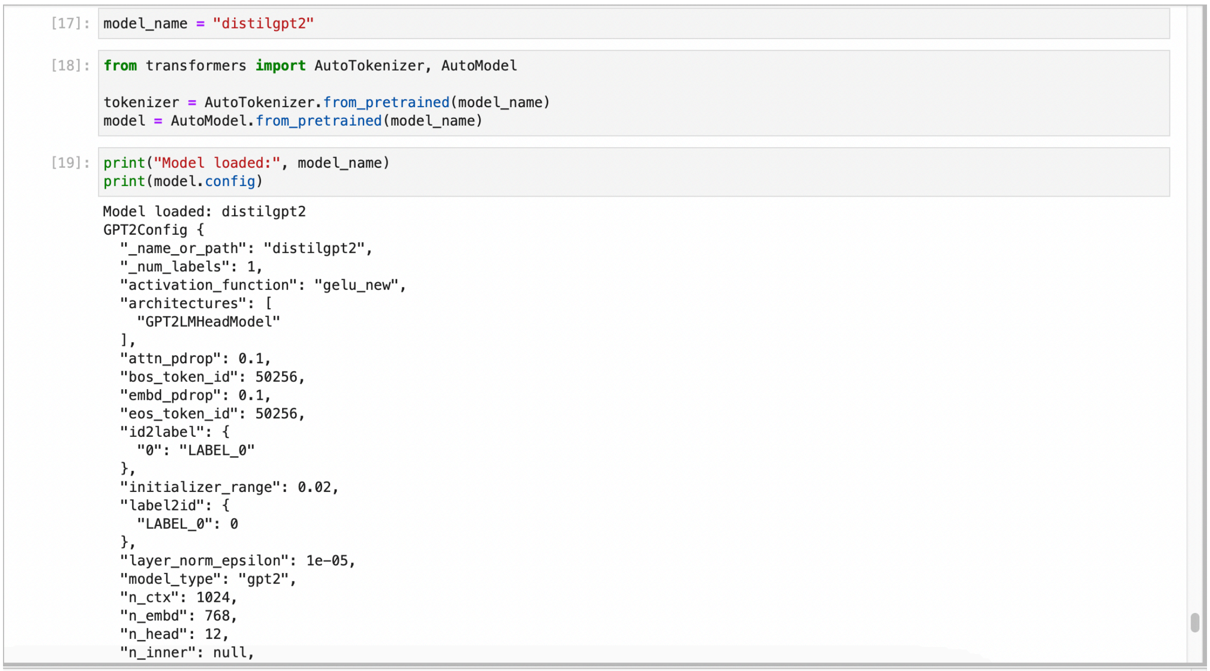
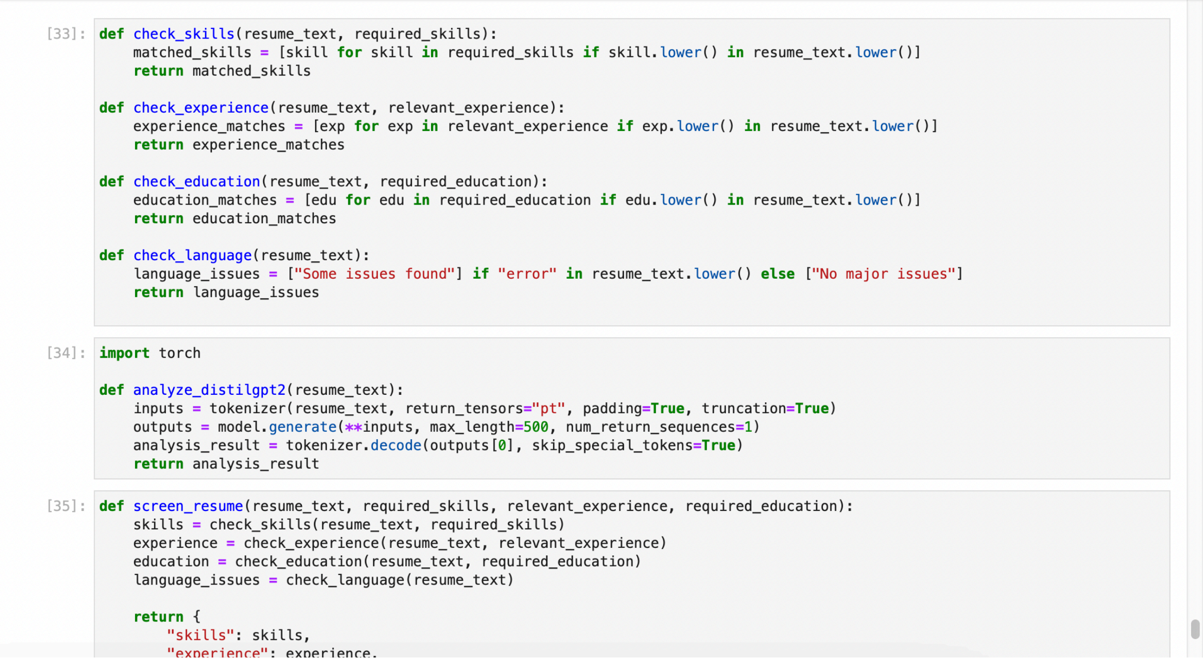
Celia M11209802

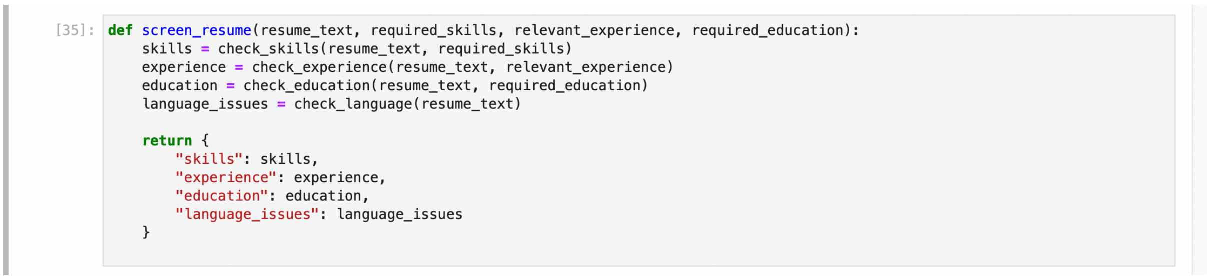
# LLM for Human Resource



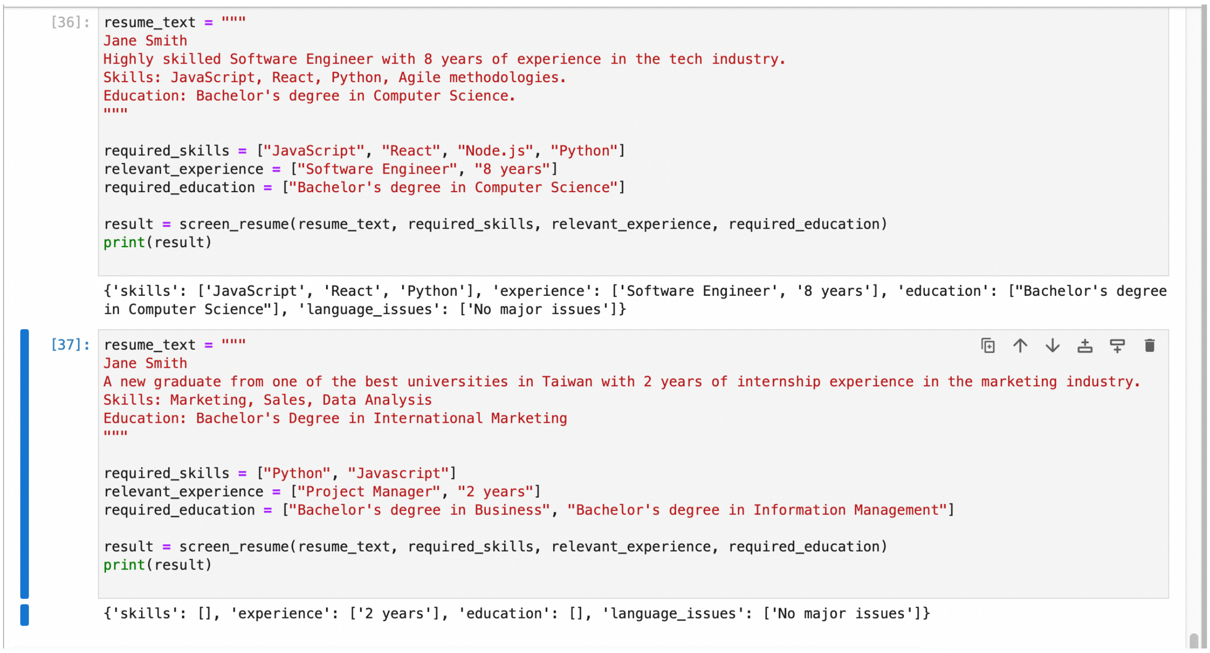
```
[17]: model_name = "distilgpt2"

[18]: from transformers import AutoTokenizer, AutoModel

      tokenizer = AutoTokenizer.from_pretrained(model_name)
      model = AutoModel.from_pretrained(model_name)

[19]: print("Model loaded:", model_name)
      print(model.config)

      Model loaded: distilgpt2
      GPT2Config {
        "_name_or_path": "distilgpt2",
        "_num_labels": 1,
        "activation_function": "gelu_new",
        "architectures": [
          "GPT2LMHeadModel"
        ],
        "attn_pdrop": 0.1,
        "bos_token_id": 50256,
        "embd_pdrop": 0.1,
        "eos_token_id": 50256,
        "id2label": {
          "0": "LABEL_0"
        },
        "initializer_range": 0.02,
        "label2id": {
          "LABEL_0": 0
        },
        "layer_norm_epsilon": 1e-05,
        "model_type": "gpt2",
        "n_ctx": 1024,
        "n_embd": 768,
        "n_head": 12,
        "n_inner": null,
```



```
[33]: def check_skills(resume_text, required_skills):
          matched_skills = [skill for skill in required_skills if skill.lower() in resume_text.lower()]
          return matched_skills

      def check_experience(resume_text, relevant_experience):
          experience_matches = [exp for exp in relevant_experience if exp.lower() in resume_text.lower()]
          return experience_matches

      def check_education(resume_text, required_education):
          education_matches = [edu for edu in required_education if edu.lower() in resume_text.lower()]
          return education_matches

      def check_language(resume_text):
          language_issues = ["Some issues found"] if "error" in resume_text.lower() else ["No major issues"]
          return language_issues

[34]: import torch

      def analyze_distilgpt2(resume_text):
          inputs = tokenizer(resume_text, return_tensors="pt", padding=True, truncation=True)
          outputs = model.generate(**inputs, max_length=500, num_return_sequences=1)
          analysis_result = tokenizer.decode(outputs[0], skip_special_tokens=True)
          return analysis_result

[35]: def screen_resume(resume_text, required_skills, relevant_experience, required_education):
          skills = check_skills(resume_text, required_skills)
          experience = check_experience(resume_text, relevant_experience)
          education = check_education(resume_text, required_education)
          language_issues = check_language(resume_text)

          return {
              "skills": skills,
              "experience": experience,
```



```
[35]: def screen_resume(resume_text, required_skills, relevant_experience, required_education):
          skills = check_skills(resume_text, required_skills)
          experience = check_experience(resume_text, relevant_experience)
          education = check_education(resume_text, required_education)
          language_issues = check_language(resume_text)

          return {
              "skills": skills,
              "experience": experience,
              "education": education,
              "language_issues": language_issues
          }
```



[36]:

```
resume_text = """
Jane Smith
Highly skilled Software Engineer with 8 years of experience in the tech industry.
Skills: JavaScript, React, Python, Agile methodologies.
Education: Bachelor's degree in Computer Science.
"""

required_skills = ["JavaScript", "React", "Node.js", "Python"]
relevant_experience = ["Software Engineer", "8 years"]
required_education = ["Bachelor's degree in Computer Science"]

result = screen_resume(resume_text, required_skills, relevant_experience, required_education)
print(result)
```

```
{'skills': ['JavaScript', 'React', 'Python'], 'experience': ['Software Engineer', '8 years'], 'education': ["Bachelor's degree
in Computer Science"], 'language_issues': ['No major issues']}
```

[37]:

```
resume_text = """
Jane Smith
A new graduate from one of the best universities in Taiwan with 2 years of internship experience in the marketing industry.
Skills: Marketing, Sales, Data Analysis
Education: Bachelor's Degree in International Marketing
"""

required_skills = ["Python", "Javascript"]
relevant_experience = ["Project Manager", "2 years"]
required_education = ["Bachelor's degree in Business", "Bachelor's degree in Information Management"]

result = screen_resume(resume_text, required_skills, relevant_experience, required_education)
print(result)
```

```
{'skills': [], 'experience': ['2 years'], 'education': [], 'language_issues': ['No major issues']}
```

Issue:

I tried to use llama2 model for this project but unfortunately, whenever I tried to load the model, it didn't work.

It always said "*The kernel for HuggingFaceGuidedTourForMac/Untitled1.ipynb appears to have died. It will restart automatically.* "

I had tried to follow this tutorial but nothing has changed.
https://stackoverflow.com/questions/47022997/jupyter-the-kernel-appears-to-have-died-it-will-restart-automatically

I asked ChatGPT about this problem and it didn't really help as well…

My assumption is because I am currently using Mac M1 for this project and it seems llama2 needs a heavier processor to load. I decided to continue using distilgpt2 even though I know it is not good enough for this project…

I have no screenshot for this as I forgot to take one…