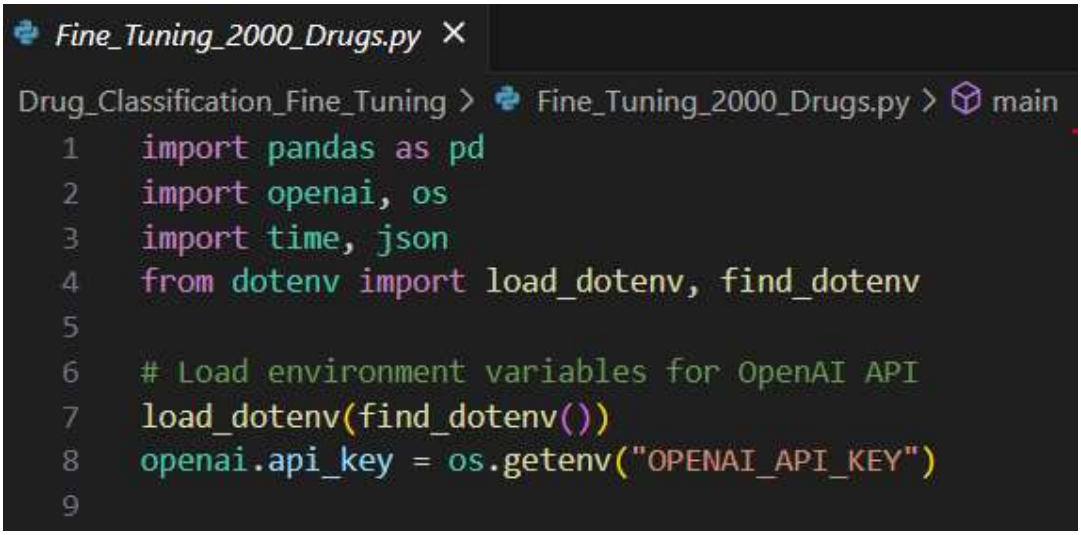
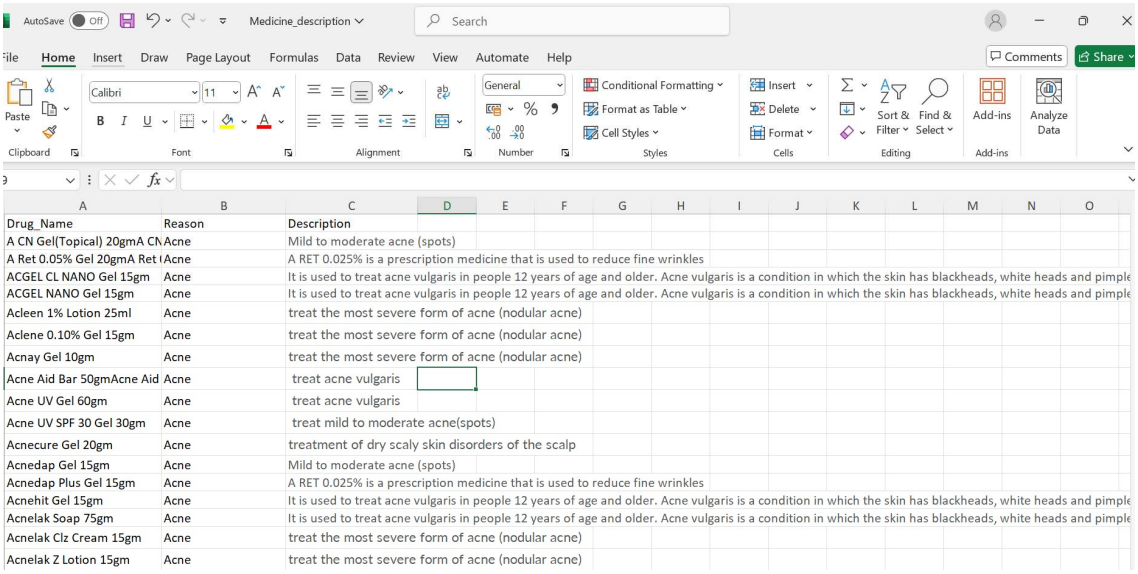
Fine-Tuning OpenAI GPT for Drug-Malady Classification

Set up environment and install required packages:

And load the key using load_dotenv and find_dotenv:



Step 1: Use an Excel file dataset – Medicine Description



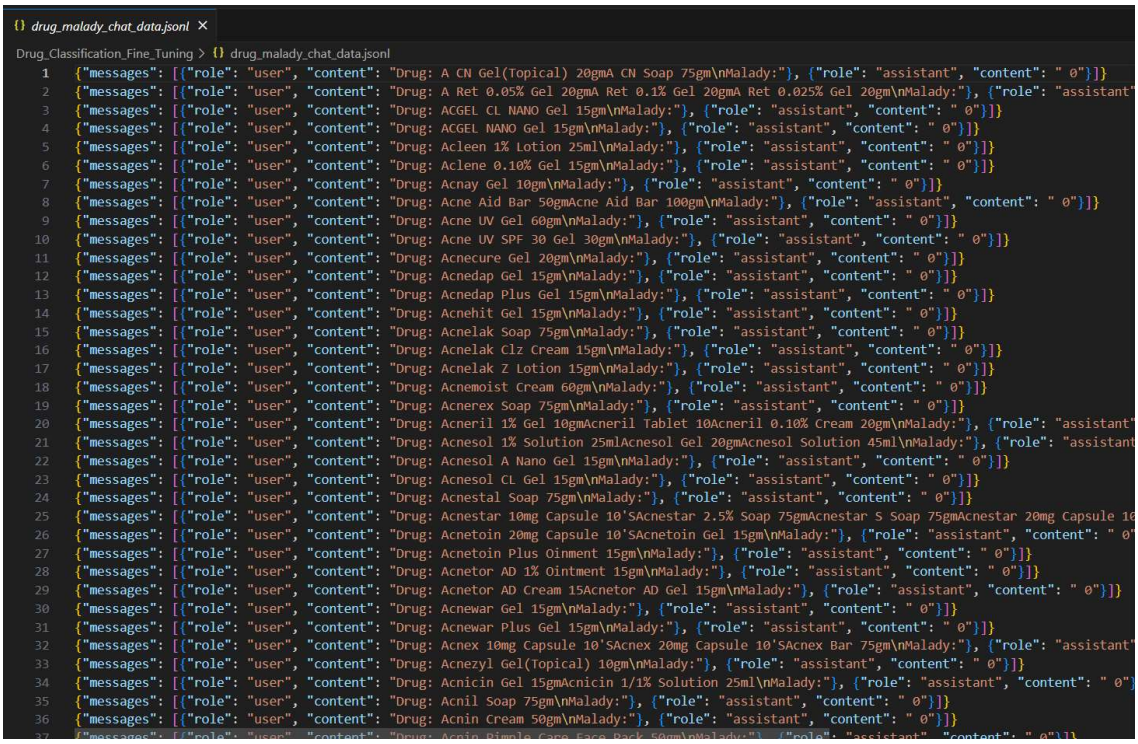
Drug_Name	Reason	Description
A CN Gel(Topical) 20gmA CN	Acne	Mild to moderate acne (spots)
A Ret 0.05% Gel 20gmA Ret	Acne	A RET 0.025% is a prescription medicine that is used to reduce fine wrinkles
ACGEL CL NANO Gel 15gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackheads, white heads and pimple
ACGEL NANO Gel 15gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackheads, white heads and pimple
Acleen 1% Lotion 25ml	Acne	treat the most severe form of acne (nodular acne)
Aclene 0.10% Gel 15gm	Acne	treat the most severe form of acne (nodular acne)
Acnay Gel 10gm	Acne	treat the most severe form of acne (nodular acne)
Acne Aid Bar 50gmAcne Aid	Acne	treat acne vulgaris
Acne UV Gel 60gm	Acne	treat acne vulgaris
Acne UV SPF 30 Gel 30gm	Acne	treat mild to moderate acne(spots)
Acnecure Gel 20gm	Acne	treatment of dry scaly skin disorders of the scalp
Acnedap Gel 15gm	Acne	Mild to moderate acne (spots)
Acnedap Plus Gel 15gm	Acne	A RET 0.025% is a prescription medicine that is used to reduce fine wrinkles
Acnehit Gel 15gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackheads, white heads and pimple
Acnelak Soap 75gm	Acne	It is used to treat acne vulgaris in people 12 years of age and older. Acne vulgaris is a condition in which the skin has blackheads, white heads and pimple
Acnelak Clz Cream 15gm	Acne	treat the most severe form of acne (nodular acne)
Acnelak Z Lotion 15gm	Acne	treat the most severe form of acne (nodular acne)

Step 2: Prepare the Data in JSONL file – Convert the excel file to jsonl file using python code.

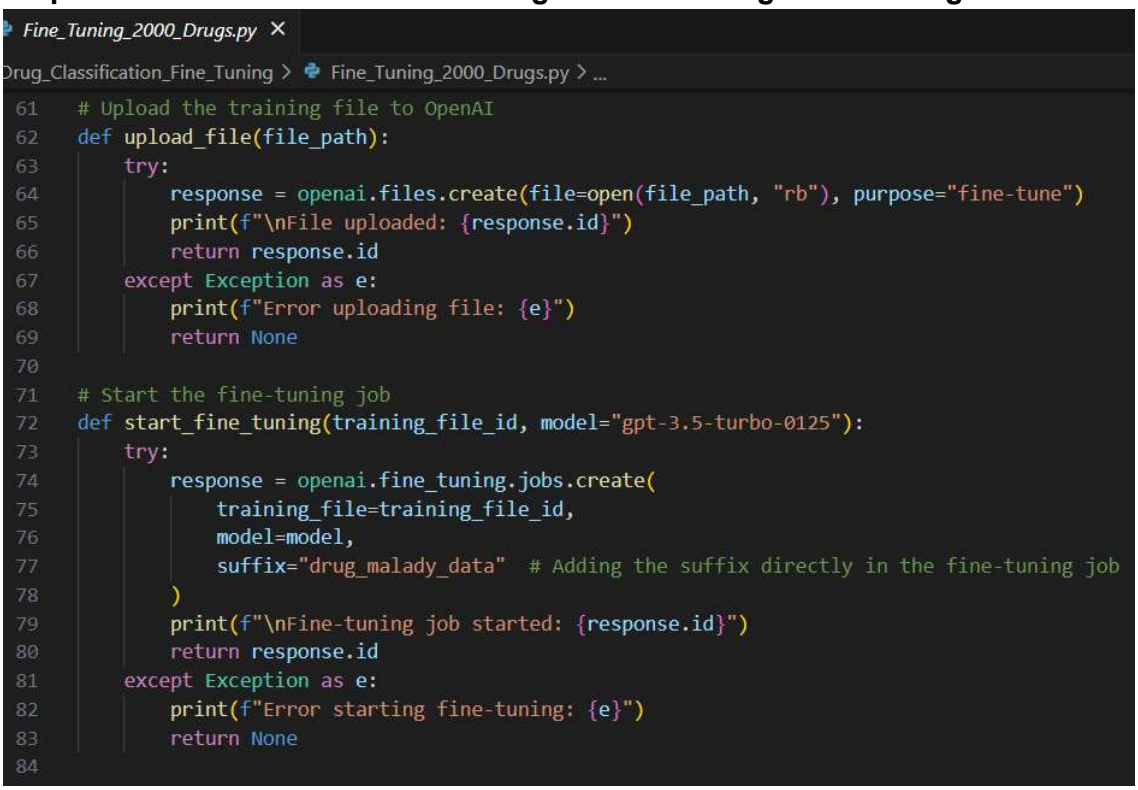
Fine_Tuning_2000_Drugs.py

Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > ...

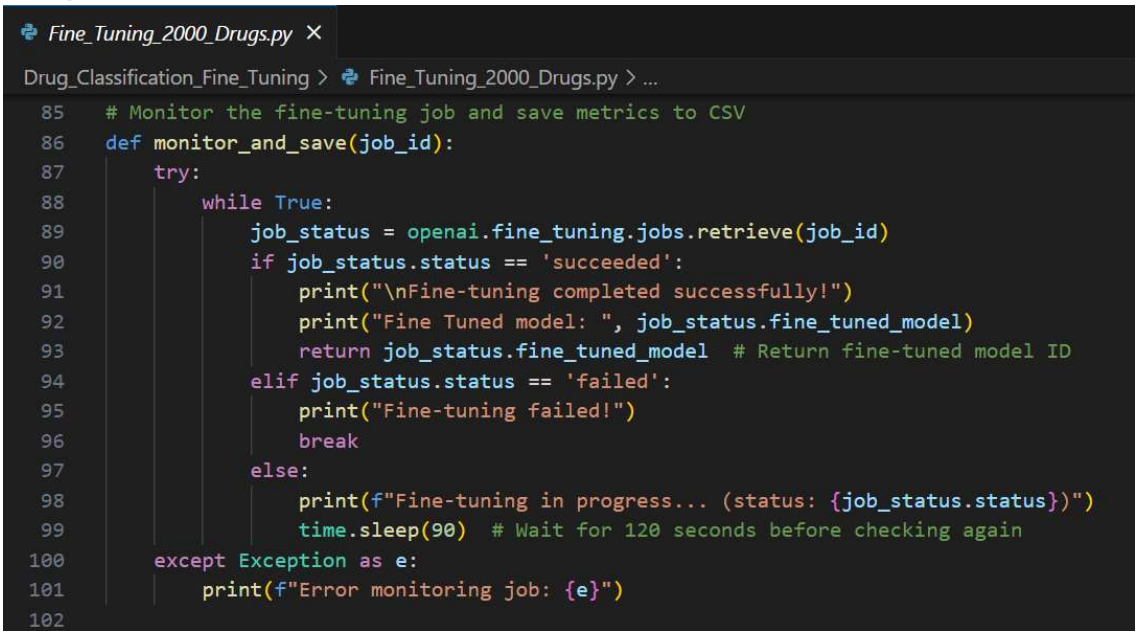
```python
# Load the data from Excel (Medicine_description.xlsx)
def load_data_from_excel(file_path, nrows=2000):
    """
    Loads the first 'nrows' rows of data from the Excel file.
    """
    try:
        df = pd.read_excel(file_path, sheet_name='Sheet1', header=0, nrows=nrows)
        print(f"\nData loaded successfully from {file_path}")
        return df
    except Exception as e:
        print(f"Error loading data from Excel: {e}")
        return None

# Map each malady (Reason) to a unique identifier
def map_maladies_to_ids(df):
    """
    Maps each malady to a unique identifier.
    """
    reasons = df["Reason"].unique()
    reasons_dict = {reason: i for i, reason in enumerate(reasons)}
    return reasons_dict

# Convert the data to JSONL format
def create_chat_format(df, reasons_dict, output_jsonl_path):
    """
    Converts the DataFrame to chat format and saves it in JSONL format.
    """
    try:
        chat_data = df.apply(create_chat_format_row, axis=1, reasons_dict=reasons_dict)

        # Convert to JSONL format and save it
        with open(output_jsonl_path, "w") as jsonl_file:
            for record in chat_data:
                jsonl_file.write(json.dumps(record) + '\n')

        print(f"\nConversion successful! Data saved as {output_jsonl_path}")
    except Exception as e:
        print(f"Error during chat format creation: {e}")

# Helper function for converting each row into chat format
def create_chat_format_row(row, reasons_dict):
    user_message = f"Drug: {row['Drug_Name']}\nMalady:"
    assistant_message = f" {reasons_dict[row['Reason']]}"

    return {
        "messages": [
            {"role": "user", "content": user_message},
            {"role": "assistant", "content": assistant_message}
        ]
    }
```



Step 3: Use this JSONL file as training file for creating a fine-tuning model.



Step 4: Then, monitor this fine-tuning process. This may take few minutes to complete.



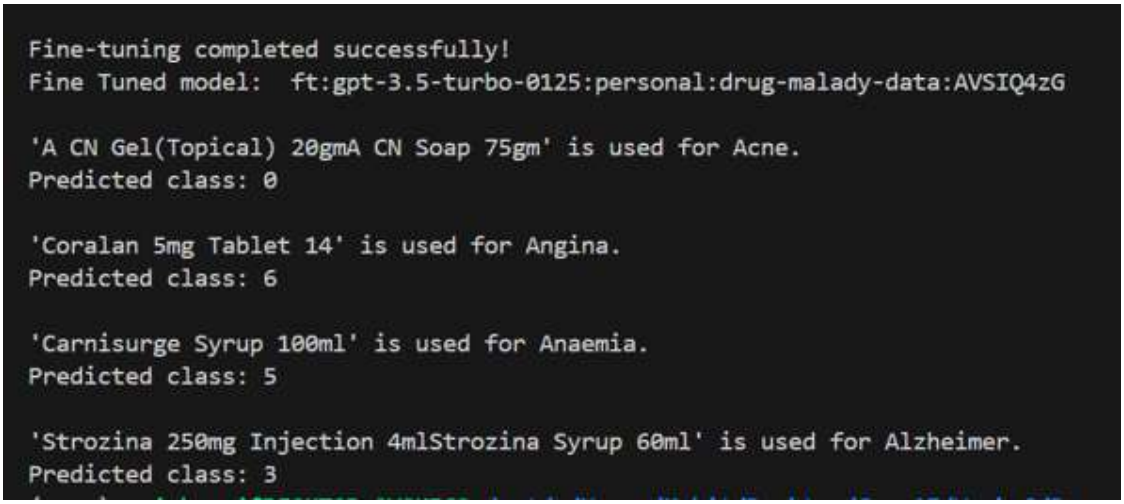
Step 5: We will test our fine-tuning model for few sample drugs and print the class for each drug.

Fine_Tuning_2000_Drugs.py

Drug_Classification_Fine_Tuning > Fine_Tuning_2000_Drugs.py > ...

```python
def test_fine_tuned_model(model):
    # Sample drugs for testing
    drugs = [
        "What is 'A CN Gel(Topical) 20gmA CN Soap 75gm' drug used for?",
        "What is 'Coralan 5mg Tablet 14'S' drug used for?",
        "What is 'Carnisurge Syrup 100ml' drug used for?",
        "What is 'Strozina 250mg Injection 4mlStrozina Syrup 60ml' drug used for?"
    ]

    # Class mapping
    class_map = {
        0: "Acne",
        1: "ADHD",
        2: "Allergies",
        3: "Alzheimer",
        4: "Amoebiasis",
        5: "Anaemia",
        6: "Angina",
    }

    # Test the fine-tuned model with each drug
    for drug in drugs:

        drug_name = drug.split("'")[1] if "'" in drug else drug

        prompt = f"Drug: {drug_name}\nMalady:"

        try:
        # Call OpenAI's API with the fine-tuned model
            response = openai.chat.completions.create(
                model=model,
                messages=[
                    {"role": "user", "content": prompt},
                ],
            )

            # Check if the response has content
            if len(response.choices) > 0:
                class_prediction = response.choices[0].message.content.strip() # type: ignore

                # Try to match the class prediction to the class map
                try:
                    predicted_class = int(class_prediction)  # Convert to integer
                    malady = class_map.get(predicted_class, "unknown class")
                    print(f"\n'{drug_name}' is used for {malady}.")
                    print(f"Predicted class: {predicted_class}")

                except ValueError:
                        print(f"Unexpected response: {class_prediction}")
            else:
                    print("No valid response from the model.")

        except Exception as e:
                print(f"Error for drug '{drug_name}': {e}")
```

Step 6: Once the fine tuning is successful, it will print the predicted class for each drug.



Output of the entire script:

```
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week 8$ python3 Fine_tuning_2000_drug.py
/mnt/c/Users/Mohit/Desktop/Gen AI/Week 8/venv/lib/python3.10/site-packages/numpy/_core/getlimits.py:548: UserWarning: Signat
ure b'\x00\xd0\xcc\xcc\xcc\xcc\xcc\xcc\xfb\xbf\x00\x00\x00\x00\x00\x00' for <class 'numpy.longdouble'> does not match any kn
own type: falling back to type probe function.
This warnings indicates broken support for the dtype!
  machar = _get_machar(dtype)
Data loaded successfully from Medicine_description.xlsx
Conversion successful! Data saved as drug_malady_chat_data.jsonl
File uploaded: file-67WL9N5Us6ujjApajHWQFk4Q
Fine-tuning job started: ftjob-fOaGfFGeGquYoBVknli4F397
Fine-tuning in progress... (status: validating_files)
Fine-tuning in progress... (status: validating_files)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
Fine-tuning in progress... (status: running)
```

```
Fine-tuning completed successfully!
Fine Tuned model:  ft:gpt-3.5-turbo-0125:personal:drug-malady-data:AVSIQ4zG

'A CN Gel(Topical) 20gmA CN Soap 75gm' is used for Acne.
Predicted class: 0

'Coralan 5mg Tablet 14' is used for Angina.
Predicted class: 6

'Carnisurge Syrup 100ml' is used for Anaemia.
Predicted class: 5

'Strozina 250mg Injection 4mlStrozina Syrup 60ml' is used for Alzheimer.
Predicted class: 3
```

GitHub URL:

https://github.com/vaishnavi477/Machine-Learning/tree/main/Generative%20AI/Drug%20Classification%20Fine%20Tuning%20OpenAI

Google slide URL:

https://docs.google.com/presentation/d/1Z_HBdy6WH5qhIlnFkl0hzczMFaluY09ap4ZEKowU0-g/edit?usp=sharing