CSC 590/599 ─ Graduate Project/Thesis (S25)
Project Progress Report

****

Project Title: Medical Text Classification using LLMs
Student Name: Medha Maisa

****

1. Description of Completed Work (Itemized)

Below is an itemized summary of the work completed to date for the project *"Medical Text Classification using LLMs."* The work is divided into two main phases corresponding to the two datasets explored: **TCGA Pathology Reports** and **Independent Medical Reviews (IMR)**.

1.1 Work Completed on TCGA Pathology Reports Dataset

- **Dataset Acquisition & Loading**
  - Successfully downloaded and loaded the TCGA Pathology Reports dataset from Kaggle.
  - Focused on key fields: text (pathology content) and patient_filename (identifier).
- **Initial Data Exploration & Cleaning**
  - Analyzed text lengths, checked for nulls, and removed duplicate or extremely short records.
  - Applied normalization (lowercasing, punctuation removal, etc.) to standardize the text.
- **Label Generation for Binary Classification**
  - Assigned binary labels: 1 if the word *"cancer"* appeared in the report text, 0 otherwise.
  - Evaluated class balance and ensured label reliability.
- **Baseline Model (TF-IDF + Logistic Regression)**
  - Built a machine learning pipeline using TF-IDF for feature extraction and Logistic Regression for classification.
  - Incorporated class weighting and stratified train-test split for fair evaluation.
  - Evaluated performance using accuracy, precision, recall, and F1-score.
- **Multi-Class Classification (Preliminary Exploration)**
  - Began examining cancer subtypes (e.g., BRCA, COAD) for potential multi-class tasks.
  - Noted significant class imbalance issues; currently identifying the most frequent subtypes for focused modeling.
- **Visualization and Reporting**
  - Created visual aids including class distribution charts, word clouds, and text length histograms.
  - Documented all work using Jupyter Notebooks for reproducibility and clarity.

Note: *Work on the TCGA dataset is currently paused as I explore an alternative dataset for potentially richer classification tasks.*

1.2 Work on Independent Medical Reviews (IMR) Dataset

- **Dataset Loading & Inspection**
  - Loaded the *Independent Medical Reviews* dataset (~19,245 records, 11 columns) from a local source.
- **Initial Data Exploration**
  - Verified dataset shape and schema; identified missing values, especially in sub-category columns.
  - Computed word-level statistics and generated summary statistics on the Findings text length.
- **Preprocessing Pipeline**
  - Developed a preprocessing function to clean the review text (lowercasing, removing numbers, punctuation, extra spaces).
  - Removed duplicate entries and very short texts (less than 10 characters) to ensure data quality.
  - Resulted in a cleaner, leaner dataset for future classification tasks.
- **Analyzing Findings Section**
  - Conducted a pattern-based analysis on the Findings text to understand structural cues and consistency across entries.
  - Parsed each entry for the following key phrases:
    - "Nature of Statutory Criteria/Case Summary:"
    - "Findings:"
    - "Final Result:"
  - Computed the distribution of these structured markers:
    - Total entries analyzed: 19,130
    - Entries with 'Nature of Statutory Criteria/Case Summary:': 2,483
    - Entries with 'Findings:': 13,928
    - Entries with 'Final Result:': 2,481
    - Entries with none of the three phrases: 4,594
    - Empty entries: 0
- These results help identify potential anchor points for rule-based or prompt-based labeling strategies for few-shot learning or LLM fine-tuning.

2. Remaining Work and Completion Plan

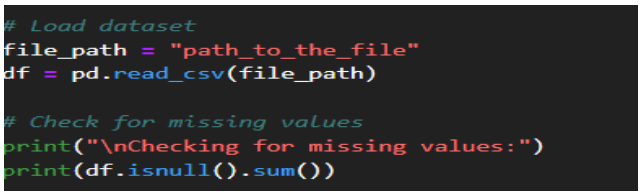
- **Further Analysis of the IMR Dataset**
  - Conduct deeper analysis of the Findings field to assess structural patterns, content reliability, and potential for classification tasks.
  - Evaluate the complexity and modeling effort required, considering label generation, text consistency, and practical interpretability.
- **Finalize Dataset Selection (TCGA vs. IMR)**
  - Based on analysis results and modeling feasibility, select one dataset (either TCGA or IMR) as the sole focus of the remainder of the project.
  - The other dataset will be discontinued, and no further modeling or analysis will be conducted on it.
- **Resume Work on Selected Dataset**
  - Continue from the previous progress point:
    - For TCGA: Proceed with contextual analysis of section headers and content blocks.

        - For IMR: Focus on improved labeling and content segmentation based on the Findings structure.
    - Refine, clean, and label the dataset to ensure it is suitable for downstream modeling.
- **Baseline Model Development (on Refined Dataset)**
    - Build an updated baseline model using traditional ML techniques (e.g., TF-IDF + Logistic Regression) on the newly refined and properly labeled dataset.
    - Evaluate the performance using standard classification metrics and visual tools.
- **LLM Model Exploration and Few-Shot Learning**
    - Fine-tune or prompt existing LLMs (e.g., GPT-4, Mistral) for the classification task using the processed dataset.
    - Experiment with few-shot learning techniques, using labeled examples to guide model predictions and test generalization.
- **Final Comparison and Results Visualization**
    - Compare performance of the three modeling approaches:
        - Baseline model
        - LLM-based classification
        - Few-shot learning with LLMs
    - Visualize comparative results using confusion matrices, bar plots, and ROC curves.
    - Summarize key findings, challenges faced, and insights gained in the final report.

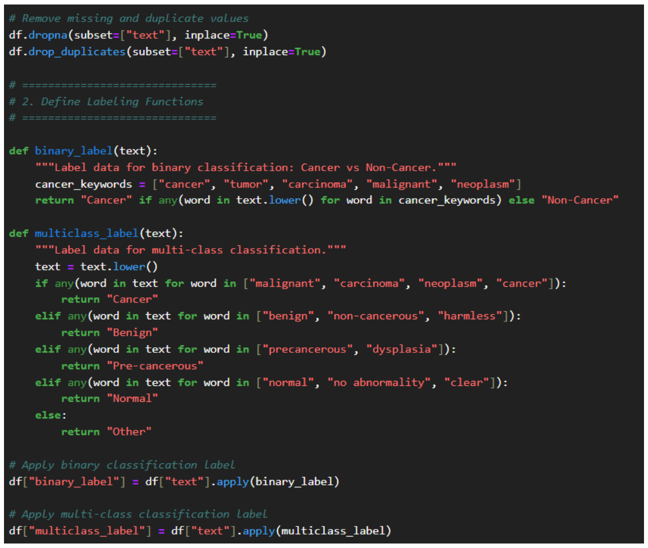
3. Appendix – Source Code

The full source code for data preprocessing, model training, evaluation, and visualizations for **TCGA dataset** and The **Work on Independent Medical Reviews (IMR) Dataset** is included in this section. The code is well organized and properly documented to ensure reproducibility. Specific code snippets and references are provided below:

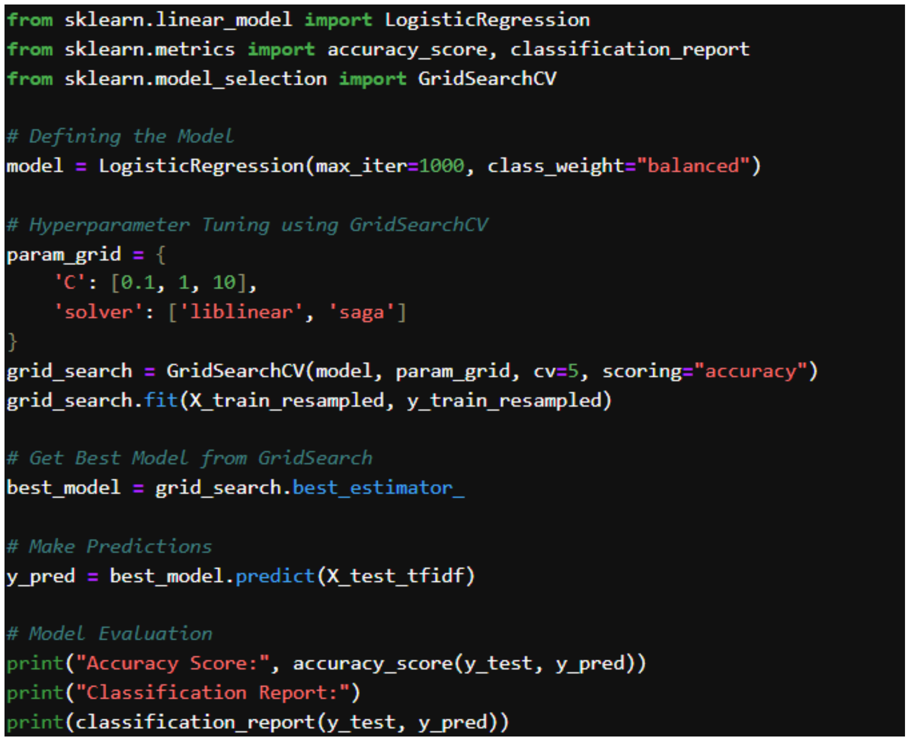
Appendix A.1: Model building using TCGA Dataset



```
# Load dataset
file_path = "path_to_the_file"
df = pd.read_csv(file_path)

# Check for missing values
print("\nChecking for missing values:")
print(df.isnull().sum())
```



```
# Remove missing and duplicate values
df.dropna(subset=["text"], inplace=True)
df.drop_duplicates(subset=["text"], inplace=True)

# ==============================
# 2. Define Labeling Functions
# ==============================

def binary_label(text):
    """Label data for binary classification: Cancer vs Non-Cancer."""
    cancer_keywords = ["cancer", "tumor", "carcinoma", "malignant", "neoplasm"]
    return "Cancer" if any(word in text.lower() for word in cancer_keywords) else "Non-Cancer"

def multiclass_label(text):
    """Label data for multi-class classification."""
    text = text.lower()
    if any(word in text for word in ["malignant", "carcinoma", "neoplasm", "cancer"]):
        return "Cancer"
    elif any(word in text for word in ["benign", "non-cancerous", "harmless"]):
        return "Benign"
    elif any(word in text for word in ["precancerous", "dysplasia"]):
        return "Pre-cancerous"
    elif any(word in text for word in ["normal", "no abnormality", "clear"]):
        return "Normal"
    else:
        return "Other"

# Apply binary classification label
df["binary_label"] = df["text"].apply(binary_label)

# Apply multi-class classification label
df["multiclass_label"] = df["text"].apply(multiclass_label)
```



```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import GridSearchCV

# Defining the Model
model = LogisticRegression(max_iter=1000, class_weight="balanced")

# Hyperparameter Tuning using GridSearchCV
param_grid = {
    'C': [0.1, 1, 10],
    'solver': ['liblinear', 'saga']
}
grid_search = GridSearchCV(model, param_grid, cv=5, scoring="accuracy")
grid_search.fit(X_train_resampled, y_train_resampled)

# Get Best Model from GridSearch
best_model = grid_search.best_estimator_

# Make Predictions
y_pred = best_model.predict(X_test_tfidf)

# Model Evaluation
print("Accuracy Score:", accuracy_score(y_test, y_pred))
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

Appendix A.2: Work on Independent Medical Reviews (IMR) Dataset.

```python
import pandas as pd
import numpy as np
import re
import string
import matplotlib.pyplot as plt
import seaborn as sns

# Step 1: Load and Analyze Dataset
file_path =
"C:/Users/Medha/Documents/CSUDH/Spring2025/Independent_Medical_Reviews.csv"  #
Update this with the correct file path
df = pd.read_csv(file_path)

# Display dataset info
print("Dataset Shape:", df.shape)
#print("First 5 rows:\n", df.head())
print("Columns:", df.columns)

print("\n Is NULL:",df.isnull().sum())
#print('\nDiagnosis Category:\n',df['Diagnosis Category'].value_counts())
print(f"\nTotal records: {df.shape[0]}")
df['text_length'] = df['Findings'].apply(lambda x: len(str(x).split()))  # For Kaggle
print('\ntext_length\n',df['text_length'].describe())

'''print("\nCOUNTING UPHELD AND OVERTURNED DECISIONS\n")
decision_counts = df['Determination'].value_counts()
```

```
# Display the counts
print(decision_counts)'''

# Check for missing values
print("\nMissing values per column:\n", df.isnull().sum())

# Step 2: Preprocessing
def clean_text(text):
    if pd.isnull(text):
        return ""  # Handle NaN values
    text = text.lower()  # Convert to lowercase
    text = re.sub(r'\d+', '', text)  # Remove numbers
    text = text.translate(str.maketrans("", "", string.punctuation))  # Remove punctuation
    text = re.sub(r'\s+', ' ', text).strip()  # Remove extra spaces
    return text

df["Cleaned_Text"] = df["Findings"].apply(clean_text)  # Assuming "Findings" contains the
medical reviews

# Remove duplicates and short texts
df.drop_duplicates(subset=["Cleaned_Text"], inplace=True)
df = df[df["Cleaned_Text"].str.len() > 10]

print(f"Dataset after preprocessing: {df.shape}")
```

```
#LATEST WORK!!
# Initialize counters

print(“Analysing Findings: \n“)
empty_count = 0
nature_case_summary_count = 0
findings_count = 0
final_result_count = 0
total_count = 0
none_phrases_count = 0

# Loop through each cleaned text
for text in df['Findings']:
    if pd.isnull(text) or text.strip() == "":
        empty_count += 1
    else:
        total_count+=1
        found = False
        text_lower = text.lower()
        if "nature of statutory criteria/case summary:" in text_lower:
            nature_case_summary_count += 1
            found = True
        if "findings:" in text_lower:
            findings_count += 1
            found = True
        if "final result:" in text_lower:
            final_result_count += 1
            found = True
        if not found:
            none_phrases_count += 1

# Print results
print("Total Entries':", total_count)
print("Number of EMPTY entries in Cleaned_Text:", empty_count)
print("Entries with 'Nature of Statutory Criteria/Case Summary:':", nature_case_summary_count)
print("Entries with 'Findings:':", findings_count)
print("Entries with 'Final Result:':", final_result_count)
print("Entries with NONE of the 3 phrases:", none_phrases_count)
```

Appendix A.3: Findings from the Independent Medical Reviews (IMR) Dataset.

```
Dataset shape after preprocessing:  (19130, 13) Index(['Reference ID', 'Report Year', 'Diagnosis
Category',

       'Diagnosis Sub Category', 'Treatment Category',

       'Treatment Sub Category', 'Determination', 'Type', 'Age Range',

       'Patient Gender', 'Findings', 'Cleaned_Text', 'text_length'],

      dtype='object')


 Is NULL: Reference ID                0

Report Year                 0

Diagnosis Category          59

Diagnosis Sub Category    1898

Treatment Category        447

Treatment Sub Category    1267

Determination               0

Type                        0

Age Range                1191

Patient Gender           1191

Findings                    0

Cleaned_Text                0

text_length                 0

dtype: int64
```

Analysing Findings:

```
Total Entries': 19130

Number of EMPTY entries in Cleaned_Text: 0

Entries with 'Nature of Statutory Criteria/Case Summary:': 2483

Entries with 'Findings:': 13928

Entries with 'Final Result:': 2481

Entries with NONE of the 3 phrases: 4594
```