University of Southern California

Spring 2025

DSCI 550

Assignment1

By

Sena London (Lead),

Kevin Sy (Facilitator),

Gideon Nazarian,

Anneliese Wilkins,

Austin Oliver,

Andrew Turagan

1. Assignment I project planning:

Will use MS project to show task completion and timeline

1. Data tokenization

This processus using pandas code help us clean the haunted paces dataset by removing punctuation leaving only string with words separated. (This code was written with the assistance of MS Copilot).

Code:

import pandas as pd

import nltk

from nltk.tokenize import word\_tokenize

import re

# Download the necessary resources

nltk.download('punkt')

# Load the CSV file

df = pd.read\_csv('haunted\_places.csv')

# Function to clean text: remove punctuation, leaving only space-separated words

def clean\_text(text):

    # Tokenize the text

    tokens = word\_tokenize(text)

    # Remove all punctuation and non-alphanumeric characters, keep only words

    tokens = [word for word in tokens if word.isalnum()]

    # Join tokens into a single space-separated string

    cleaned\_text = ' '.join(tokens)

    return cleaned\_text

# Apply the function to each cell in the DataFrame

df = df.map(lambda cell: clean\_text(cell) if isinstance(cell, str) else cell, na\_action='ignore')

# Save the cleaned DataFrame to a new CSV file

df.to\_csv('cleaned\_haunted\_file.csv', index=False)

The output of the previous code will be saved in the cleaned\_haunted \_file.csv.

1. Date finder

To search for date, we use the following coding with the panda’s library with the help of MS Copilot. When the date couldn’t be found, it returns the default 1/1/2025.

import pandas as pd

import re

# Function to find and format date

def find\_date(text):

    # Patterns for year month date and year/month/date

    date\_patterns = [r'\b\d{4}\s\d{2}\s\d{2}\b', r'\b\d{4}/\d{2}/\d{2}\b']

    for pattern in date\_patterns:

        match = re.search(pattern, text)

        if match:

            date\_str = match.group()

            try:

                # Parse the date format and convert to desired format

                if ' ' in date\_str:

                    date\_obj = pd.to\_datetime(date\_str, format='%Y %m %d')

                elif '/' in date\_str:

                    date\_obj = pd.to\_datetime(date\_str, format='%Y/%m/%d')

                return date\_obj.strftime('%Y/%m/%d')

            except ValueError:

                return '2025/01/01'

    return '2025/01/01'

# Read input CSV file

input\_file = 'haunted\_places.csv'

df = pd.read\_csv(input\_file)

# Apply function to each row and create new column

df['date'] = df['description'].apply(find\_date)

# Save output to new CSV file

output\_file = 'date\_output\_haunted\_file.csv'

df.to\_csv(output\_file, index=False)

# Print confirmation

print(f"Processed data has been saved to {output\_file}")

The output is then saved to the date\_output\_haunted\_file.csv and then copied and pasted to the cleaned haunted file.

1. Audio feature analysis

Code:

import pandas as pd

# Load the feature file (Excel) with specified encoding

feature\_df = pd.read\_excel('keywords\_dictionary.xlsx')

# Extract the feature names from the 'Audio\_Evidence' column

feature\_names = feature\_df['Audio\_Evidence'].dropna().unique()

# Load the input file with specified encoding

input\_df = pd.read\_csv('cleaned\_haunted\_file.csv', encoding='latin1')

# Function to compare features to description

def compare\_features(description, features):

    matches = {}

    for feature in features:

        matches[feature] = int(feature in description)

    return matches

# Iterate through each row in the input file and apply the comparison

comparison\_results = []

for \_, row in input\_df.iterrows():

    description = row['description']

    matches = compare\_features(description, feature\_names)

    comparison\_results.append(matches)

# Create a DataFrame with the comparison results

comparison\_df = pd.DataFrame(comparison\_results)

# Add the comparison results to the input DataFrame

result\_df = pd.concat([input\_df, comparison\_df], axis=1)

# Save the result to a new CSV file

result\_df.to\_csv('comparison\_results.csv', index=False)

The Audio evidence feature came from chat GPT and Copilot and the result is saved on the comparison\_results.csv.