# Documentation for Weather-Based Trail Recommendation System

## Abstract

This project implements a weather-based trail recommendation system using a Flask web application. The application fetches real-time weather data based on a user's location, processes the data to provide weather advice, and recommends suitable hiking trails. The system leverages various machine learning techniques, natural language processing (NLP), and Azure Maps for location services.

## Introduction

Hiking is a popular recreational activity that is highly influenced by weather conditions. This project aims to enhance the hiking experience by providing personalized trail recommendations based on current weather data. The system is built using Python and various libraries such as Flask, scikit-learn, spaCy, and Azure Maps.

## Methodology

### Technologies and Libraries Used

1. **Flask**: A lightweight WSGI web application framework used to create the web application.
2. **scikit-learn:** A machine learning library used for data preprocessing and clustering.
3. **spaCy**: An NLP library used to process user queries.
4. **Azure Maps**: A service used to convert user-provided locations into latitude and longitude coordinates.
5. **Pandas**: A data manipulation library used for data handling and preprocessing.
6. **Requests**: A library used to make HTTP requests to fetch weather data.

### Data Flow and Processing

1. **User Query:** The user provides a query and a location.
2. **Location Search:** The location is converted into latitude and longitude coordinates using Azure Maps.
3. **Weather Data Fetching:** Real-time weather data for the specified location is fetched from the Microsoft Azure Weather service.
4. **Data Preprocessing:** The fetched weather data is processed and transformed into a suitable format for analysis.
5. **Clustering**: The processed data is normalized and clustered using KMeans to categorize weather conditions.
6. **Advice Generation**: Weather advice is generated based on the cluster.
7. **Trail Recommendation**: Trails are recommended based on the current weather conditions.
8. **Response**: The system responds with weather advice and trail recommendations.

### Functions and Their Roles

1. **search\_address:** Converts user-provided location queries into latitude and longitude coordinates using Azure Maps.
2. **fetch\_weather\_data**: Fetches real-time weather data for the specified coordinates.
3. **preprocess\_data**: Processes raw weather data and transforms it into a DataFrame.
4. **get\_real\_time\_weather:** Manages the workflow of fetching and processing real-time weather data.
5. **get\_advice**: Provides weather advice based on the identified weather cluster.
6. respond\_to\_query: Formats the system response, combining user query, weather advice, and location.
7. **recommend\_trails:** Recommends suitable trails based on the current weather conditions.
8. **chat**: The main route handling user queries, processing weather data, generating advice, and recommending trails.

### Methodology

### Clustering Weather Data

#### Data Preparation

Weather data is collected from multiple CSV files stored in a specified folder using pandas. These files contain historical weather information necessary for training the clustering model.

A screenshot of a computer

Description automatically generated

#### Data Normalization

Normalize the features using Min-Max scaling to ensure uniformity in the range of values across different features.

A screenshot of a computer

Description automatically generated

#### Determining Optimal Number of Clusters

Use the Elbow method to determine the optimal number of clusters (k) for K-Means clustering.

A graph with a blue line

Description automatically generated

#### K-Means Clustering

Apply K-Means clustering with the determined optimal number of clusters (k=4) to group weather data into distinct clusters.

A screenshot of a computer program

Description automatically generated

#### Visualizing data

A graph with different colored dots

Description automatically generated

## Results

The implemented system successfully integrates weather data fetching, processing, and trail recommendation based on real-time weather conditions. The system uses clustering to categorize weather conditions and provide relevant advice and trail suggestions.

## Example Usage

1. **Input**: User query - " What's the weather like and what should I wear?

**Location –** Milpitas, CA

**Output**:

A screenshot of a computer

Description automatically generated

1. **Input**: User query - " What's the weather like and what should I wear?

Location – San Francisco, CA

**Output**:

A screenshot of a computer

Description automatically generated

## Conclusion

The weather-based trail recommendation system demonstrates the integration of real-time data processing, machine learning, and NLP to enhance user experience in recreational activities. Future work can include expanding the trail database, refining the clustering model, and incorporating user feedback for personalized recommendations.

## References

* Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*
* Honnibal, M., & Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.
* Microsoft Azure Maps. (n.d.). Retrieved from <https://azure.microsoft.com/en-us/services/azure-maps/>