question

February 6, 2025

1 Project: Computing Entropy, Mutual Information, and Kullback-Leibler Divergence

1.1 Overview

In this project, you will implement three fundamental concepts from information theory: **Entropy**, **Mutual Information**, and **Kullback-Leibler (KL) Divergence**. These concepts are widely used in machine learning, data science, and statistics to measure uncertainty, information gain, and the difference between probability distributions.

The project is designed to be simple yet insightful, allowing you to understand the core ideas behind these concepts by implementing them from scratch in Python. You will then apply these implementations to a real-world dataset to gain practical insights.

1.2 Objectives

By the end of this project, you will:

1. Understand the mathematical foundations of entropy, mutual information, and KL divergence.
2. Implement these concepts from scratch using Python.
3. Apply your implementations to a real-world dataset to compute these metrics and interpret the results.

1.3 Project Steps

1.3.1 Step 1: Implement Entropy

Entropy is a measure of the uncertainty or randomness in a probability distribution. For a discrete random variable $ X $ with possible outcomes $ x_1, x_2, ..., x_n $ and corresponding probabilities $ P(x_1), P(x_2), ..., P(x_n) $, the entropy $ H(X) $ is defined as:

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i)$$

Task: Write a Python function `entropy(probabilities)` that takes a list of probabilities and returns the entropy of the distribution.

```
import numpy as np

def entropy(probabilities):
```

```
    # Your code here
    pass
```

1.3.2 Step 2: Implement Mutual Information

Mutual Information measures the amount of information obtained about one random variable through another random variable. For two discrete random variables $ X $ and $ Y $, the mutual information $ I(X; Y) $ is defined as:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log_2 \left(\frac{P(x,y)}{P(x)P(y)} \right)$$

Task: Write a Python function `mutual_information(joint_prob, marginal_x, marginal_y)` that computes the mutual information between two random variables $ X $ and $ Y $.

```
def mutual_information(joint_prob, marginal_x, marginal_y):
    # Your code here
    pass
```

1.3.3 Step 3: Implement Kullback-Leibler Divergence

Kullback-Leibler (KL) Divergence measures how one probability distribution diverges from a second, reference probability distribution. For two discrete probability distributions $ P $ and $ Q $, the KL divergence $ D_{KL}(P \mid Q) $ is defined as:

$$D_{KL}(P \| Q) = \sum_i P(i) \log_2 \left(\frac{P(i)}{Q(i)} \right)$$

Task: Write a Python function `kl_divergence(p, q)` that computes the KL divergence between two probability distributions $ P $ and $ Q $.

```
def kl_divergence(p, q):
    # Your code here
    pass
```

1.3.4 Step 4: Apply Your Functions to Real Data

Story: Imagine you are a data scientist working for a botanical research institute. Your team has collected data on various iris flowers, including measurements of sepal length, sepal width, petal length, and petal width. The dataset also includes the species of each iris flower (setosa, versicolor, or virginica).

Your task is to analyze this dataset using the concepts of entropy, mutual information, and KL divergence to gain insights into the relationships between the different features and the species of the flowers.

Dataset: You will use the famous Iris dataset, which is available in the `sklearn.datasets` module.

Tasks:

1. **Compute Entropy of a Feature:**

- Choose one of the features (e.g., sepal length) and compute its entropy. This will give you a measure of the uncertainty or randomness in the distribution of that feature.

2. **Compute Mutual Information Between Two Features:**
   - Select two features (e.g., sepal length and petal length) and compute the mutual information between them. This will tell you how much information one feature provides about the other.
3. **Compute KL Divergence Between Two Distributions:**
   - Compare the distribution of a feature (e.g., sepal length) for two different species (e.g., setosa and versicolor) using KL divergence. This will give you a measure of how different the two distributions are.

```
from sklearn.datasets import load_iris
import pandas as pd

# Load the Iris dataset
data = load_iris()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['species'] = data.target

# Compute entropy for one of the features
probabilities = df['sepal length (cm)'].value_counts(normalize=True)
entropy_value = entropy(probabilities)
print(f"Entropy of sepal length: {entropy_value}")

# Compute mutual information between two features
# (You will need to compute the joint and marginal probabilities first)

# Compute KL divergence between two distributions
# (You will need to define two probability distributions first)
```

1.4 Deliverables

1. A Python script containing your implementations of `entropy`, `mutual_information`, and `kl_divergence`.
2. A Jupyter notebook or Python script demonstrating the application of your functions to the Iris dataset.
3. A brief report (1-2 pages) explaining your implementation, the results, and any insights gained from the project.

1.5 Grading Criteria

- **Correctness (50%):** Your implementations should correctly compute entropy, mutual information, and KL divergence.
- **Application (30%):** Your application of the functions to the Iris dataset should be meaningful and well-documented.
- **Report (20%):** Your report should clearly explain your approach, results, and any challenges you faced.

1.6 Resources

- Information Theory - Wikipedia
- Entropy in Information Theory
- Mutual Information
- Kullback-Leibler Divergence

1.7 Submission

Submit your Python script, Jupyter notebook, and report as a single zip file. Ensure that your code is well-commented and easy to follow.

Good luck, and have fun exploring the fascinating world of information theory!