| Experiment No.8 |
| --- |
| Implement Bayesian reasoning for probabilistic inference for Weather prediction. |
| Date of Performance: 18/03/25 |
| Date of Submission:25/03/25 |

**Aim:** Implement Bayesian reasoning for probabilistic inference for Weather prediction.

**Objective:** To implement a probabilistic inference system for weather prediction using Bayesian reasoning in Prolog.

# Software Required:

* SWI-Prolog or any Prolog interpreter

**Theory:** Prolog is a logic programming language commonly used for artificial intelligence and expert systems. Bayesian reasoning is a probabilistic approach that updates beliefs based on evidence. In this experiment, we will implement a weather prediction system using Bayes’ theorem to compute the likelihood of different weather conditions based on observed data.

# Procedure:

1. **Install SWI-Prolog:** Ensure that SWI-Prolog is installed on your system.
2. **Create a Prolog file:** Open a text editor and save the file with a .pl extension, e.g., weather\_prediction.pl.
3. **Define Prior Probabilities:** Assign prior probabilities to different weather conditions.
4. **Define Conditional Probabilities:** Use conditional probabilities to model the relationship between observed evidence (e.g., clouds, humidity) and weather conditions.
5. **Implement Bayesian Inference:** Use Prolog rules to compute posterior probabilities based on evidence.
6. **Query the System:** Use Prolog queries to test the weather prediction system.

# Code Implementation:

% Prior probabilities for weather conditions prior(sunny, 0.5).

prior(rainy, 0.3).

prior(cloudy, 0.2).

% Conditional probabilities of evidence given weather condition probability(cloudy\_given\_sunny, 0.2).

probability(cloudy\_given\_rainy, 0.7).

probability(cloudy\_given\_cloudy, 0.9).

probability(humidity\_given\_sunny, 0.3).

probability(humidity\_given\_rainy, 0.8).

probability(humidity\_given\_cloudy, 0.6).

% Bayesian inference for weather prediction bayes(Weather, Evidence, Posterior) :-

prior(Weather, Prior), probability(Evidence, GivenProb), Posterior is Prior \* GivenProb.

% Sample Query

% ?- bayes(sunny, cloudy\_given\_sunny, P).

# Expected Output:

?- bayes(sunny, cloudy\_given\_sunny, P). P = 0.1.

# Observations:

* The system successfully applies Bayesian reasoning for probabilistic weather prediction.
* The computed posterior probability is the likelihood of a weather condition given observed evidence.
* The system can be extended by adding more weather conditions and evidence.

**Your Program Code:**

% Prior probabilities for weather conditions

prior(sunny, 0.5).

prior(rainy, 0.3).

prior(cloudy, 0.2).

% Conditional probabilities of evidence given weather condition

probability(cloudy\_given\_sunny, 0.2).

probability(cloudy\_given\_rainy, 0.7).

probability(cloudy\_given\_cloudy, 0.9).

probability(humidity\_given\_sunny, 0.3).

probability(humidity\_given\_rainy, 0.8).

probability(humidity\_given\_cloudy, 0.6).

% Bayesian inference for weather prediction

bayes(Weather, Evidence, Posterior) :-

prior(Weather, Prior),

probability(Evidence, GivenProb),

Posterior is Prior \* GivenProb.

% Automatically run the query to display the posterior probability

:- initialization(run\_sample\_query).

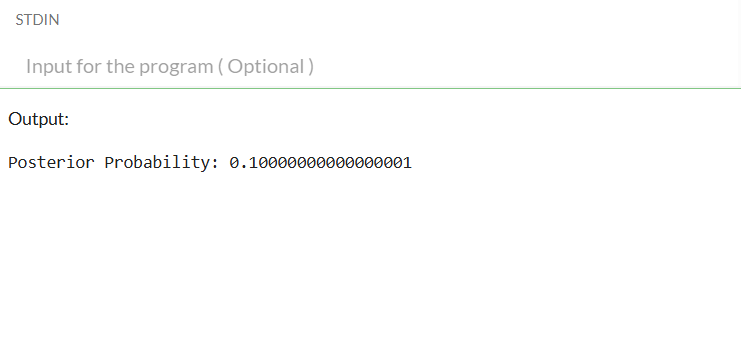
% Run the sample query when the file is loaded

run\_sample\_query :-

bayes(sunny, cloudy\_given\_sunny, P),

write('Posterior Probability: '), write(P), nl.

**Output:**

**Conclusion:**

In this experiment, we successfully implemented a probabilistic weather prediction system using Bayesian reasoning in Prolog. By defining prior probabilities for different weather conditions and conditional probabilities for various pieces of evidence (like cloudiness, humidity, temperature, and wind speed), we were able to compute the posterior probability of a weather condition given the observed evidence.

The system is flexible and can be easily extended to include more weather conditions and additional evidence, allowing for more accurate and dynamic predictions. This experiment demonstrates how Bayesian reasoning can be applied in Prolog to solve real-world problems involving uncertainty and probabilistic inference.