Module 3: Probability for Machine Learning

Learning outcomes

1. Analyse the origin of probability theory and its connection to games of chance.
2. Differentiate between statistics and probability.
3. Apply the fair and loaded coin models to predict outcomes.
4. Rank the probabilities for the three-coin flips.
5. Classify independent versus dependent variables.
6. Analyse the formula for three independent variables.
7. Calculate the probabilities of different events.
8. Run simulations in Python using NumPy to model random events, access and transform array data, compute frequencies, and summarise outcomes using statistical functions.
9. Apply Bayes' rule to calculate the conditional probability of an event.
10. Apply Bayes' rule to evaluate the total probability of an event.
11. Analyse the Python code used to run the Monte Carlo simulation.
12. Identify the real-world applications of probability distribution.
13. Apply Pascal's triangle formula to calculate binomial distribution and coefficient.
14. Apply the central limit theorem (CLT) to calculate mean, variance and standard deviation.
15. Apply the de Moivre–Laplace theorem to calculate probabilities.
16. Analyse how changes in variables affect the characteristics of a normal distribution.
17. Set up a public GitHub profile.

Fundamentals of probability

- **Probability** quantifies uncertainty in AI and decision-making.
- **Statistics vs probability**
    - Statistics infer from data.
    - Probability predicts data from models.
- **Basic models:**

- **Fair coin:** equal chance (50%) of heads/tails.
- **Loaded coin:** skewed probability (e.g. P(H) ≠ P(T)).
- **Independent events**: multiply probabilities: P(A and B) = P(A) × P(B).
- **Multiple coin flips:** the number of outcomes follows 2^n for n flips.
- **Formula for independence:** P(A, B, C) = P(A) × P(B) × P(C).

Bayes' rule

- **Formula:** P(A|B) = (P(B|A) × P(A)) ÷ P(B)
- **Applications**
  - **Spam filtering** classifies emails as spam or not.
  - **Medical diagnosis** updates disease probability based on test results.
  - **Recommendation systems** predict user preferences.
  - **Self-driving cars** estimate object locations with sensor data.
  - **Monty Hall problem,** when switching increases the probability of winning.

Monte Carlo simulations

- Uses **random sampling** for approximations.
- **Applications:** finance, AI, reinforcement learning, physics modelling.
- **Python simulation:** used to model uncertainty and optimise decision-making.

Probability distributions

- **Binomial:** models discrete events (e.g. coin flips, customer clicks).
- **Normal (Gaussian):** applies to continuous data (e.g. height, test scores).
- **Central limit theorem (CLT): large** sample sums approximate a normal distribution.
- **de Moivre–Laplace Theorem:** approximates binomial distributions using normal distributions.
- **Pascal's triangle:** computes binomial coefficients for probability calculations.

Applications in ML/AI

- **Bayesian networks:** model probabilistic dependencies.
- **Monte Carlo methods:** applied in reinforcement learning and risk modelling.
- **Normal distribution:** found in regression models and neural networks.
- **Decision-making:** guides decision-making under uncertainty in robotics, NLP and finance.
- **Manipulating normal variables:** changes in mean and variance affect model behaviour.