Quantum Version of Decision Tree Construction Algorithm Subroutine

Ilnaz Mannapov, Kamil Khadiev

Kazan Federal University, Russia

MMCP 2026

Outline

1. Introduction
2. Quantum Approaches
3. Complexity Analysis
4. Comparison
5. Implementation Variants
6. Conclusion and Future Work

Decision Trees: Classification and Regression

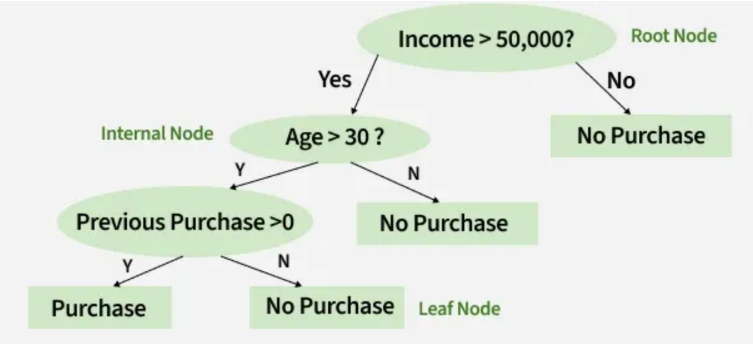
- **Classification Problem:**
  - Predict discrete class labels for input data.
  - Example: spam detection, disease diagnosis.
  - Decision trees partition feature space into class regions.

Decision Trees: Classification and Regression

- **Classification Problem:**
  - Predict discrete class labels for input data.
  - Example: spam detection, disease diagnosis.
  - Decision trees partition feature space into class regions.
- **Regression Problem:**
  - Predict continuous numerical values.
  - Example: house prices, temperature forecasting.
  - Decision trees output mean values in leaf nodes.

Decision Trees: Classification and Regression

- **Classification Problem:**
  - Predict discrete class labels for input data.
  - Example: spam detection, disease diagnosis.
  - Decision trees partition feature space into class regions.
- **Regression Problem:**
  - Predict continuous numerical values.
  - Example: house prices, temperature forecasting.
  - Decision trees output mean values in leaf nodes.
- **Decision Tree Structure:**
  - Root node: entire dataset.
  - Internal nodes: feature-based splits.
  - Leaf nodes: final predictions.
- Classical algorithms: C4.5, C5.0, CART.

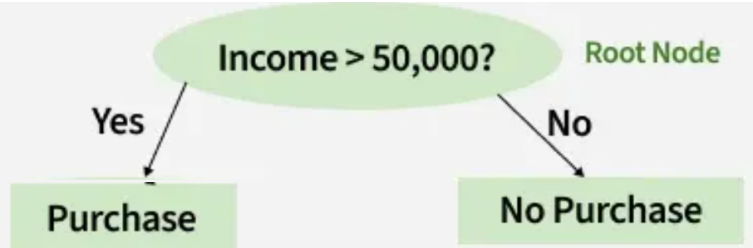


Decision Trees in Machine Learning

- Core subroutine of tree building is **best split selection** based on impurity measures.

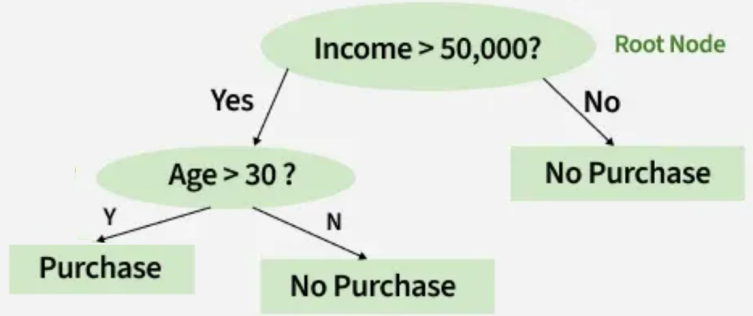
Decision Trees in Machine Learning

- Core subroutine of tree building is **best split selection** based on impurity measures.



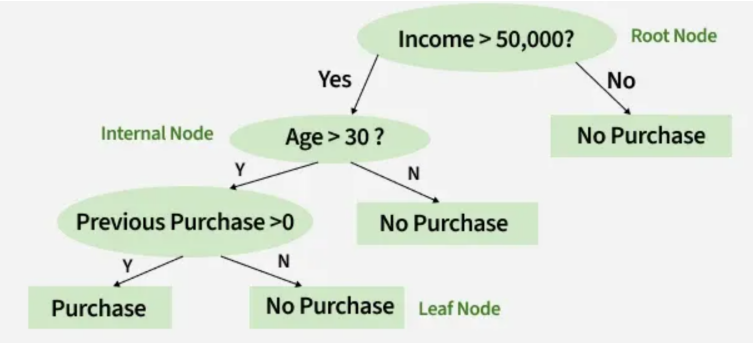
Decision Trees in Machine Learning

- Core subroutine of tree building is **best split selection** based on impurity measures.



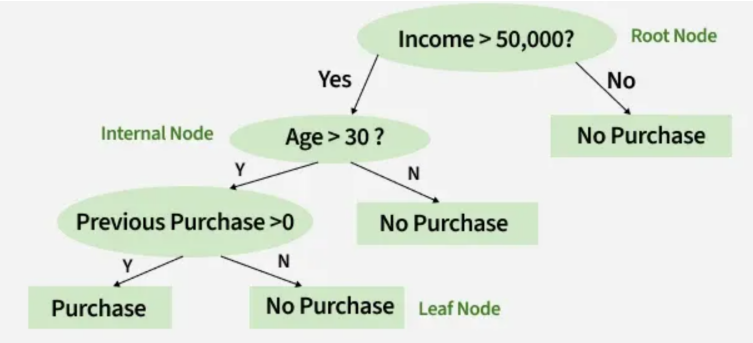
Decision Trees in Machine Learning

- Core subroutine of tree building is **best split selection** based on impurity measures.



Decision Trees in Machine Learning

- Core subroutine of tree building is **best split selection** based on impurity measures.
- Gini impurity is a common criterion.
- Classical complexity: evaluating n splits over N points gives $O(nN)$.



Parameters: n and m

- n – **Number of Candidate Splits:**
  - Represents the number of possible split points to evaluate.
  - Classical evaluation requires checking all n splits.
  - Quantum approach reduces to $O(\sqrt{n})$ via maximum finding.

Parameters: n and m

- n – **Number of Candidate Splits:**
  - Represents the number of possible split points to evaluate.
  - Classical evaluation requires checking all n splits.
  - Quantum approach reduces to $O(\sqrt{n})$ via maximum finding.
- m – **Qubit Register Size:**
  - Number of qubits needed to represent data counts and impurity values.
  - Typically $m = O(\log N)$ where N is the number of data points.
  - For efficient quantum computation:

$$m = O(\text{poly}(\log n))$$

Parameters: n and m

- n – **Number of Candidate Splits:**
  - Represents the number of possible split points to evaluate.
  - Classical evaluation requires checking all n splits.
  - Quantum approach reduces to $O(\sqrt{n})$ via maximum finding.
- m – **Qubit Register Size:**
  - Number of qubits needed to represent data counts and impurity values.
  - Typically $m = O(\log N)$ where N is the number of data points.
  - For efficient quantum computation:

$$m = O(\text{poly}(\log n))$$

- **Relationship:**
  - When $N = \Theta(n)$, classical depth is $O(n^2)$.
  - Quantum depth with $m = O(\text{poly}(\log n))$ becomes subquadratic.
  - This ensures quantum advantage for large-scale datasets.

Gini Impurity and Gain

- Gini impurity of a set X:

$$\text{Gini}(X) = 1 - \sum_{c=1}^{M} \left(\frac{|X_c|}{|X|} \right)^2$$

where X_c is the subset of points belonging to a class c; M is the number of classes.

- Gain of split s: $(X^s, X \backslash X^s)$:

$$\text{Gain}(X, s) = \text{Gini}(X) - \left(\frac{|X^s|}{|X|} \text{Gini}(X^s) + \frac{|X \backslash X^s|}{|X|} \text{Gini}(X \backslash X^s) \right)$$

- Best split maximizes the gain over all candidate splits

$$S = \{(X^1, X \backslash X^1), (X^2, X \backslash X^2), \ldots, (X^n, X \backslash X^n)\}$$

.

Quantum Subroutine for Best Split

- Use quantum maximum finding (Durr & Hoyer 1996) with $O(\sqrt{n})$ oracle calls.
- Each oracle call evaluates the gain for one split.
- Gain evaluation requires counting points per class per branch, computing squares and divisions.
- Implemented using quantum arithmetic circuits on m-qubit registers.
- Four implementations considered:
  1. QFT-based multiplication.
  2. Approximated QFT.
  3. VBE ripple-carry adder.
  4. VBE ripple-carry adder with RCCX gates.

QFT-Based Approach (Ramezani et al., 2023)

Theorem 1

- Using exact QFT multiplier ($O(m^2)$ depth), per-split evaluation costs $O(nm^2)$.
- With $O(\sqrt{n})$ iterations of maximum finding.
- **Total Circuit Depth:** $O(n^{1.5}m^2)$
- Observation: Fewer ancilla qubits, but higher depth.

Approximated QFT Approach (Nam et al., 2020)

Theorem 2

- Approximated QFT reduces multiplication depth to $O(m \log m)$.
- Per-split evaluation: $O(nm \log m)$.
- **Total Circuit Depth:** $O(n^{1.5} m \log m)$
- Observation: Trade-off between accuracy and T-gate count.

VBE Ripple-Carry Adder (Vedral et al., 1996)

Theorem 3

- Ripple-carry adder adds two m-bit numbers in $O(m)$ depth.
- Multiplication can be optimised; dominant cost is additions.
- Per-split evaluation: $O(nm)$.
- **Total Circuit Depth:** $O(n^{1.5}m)$
- Observation: Lower asymptotic depth, but requires $O(m)$ ancilla qubits.

VBE + RCCX Gates

Theorem 4

- Replacing Toffoli gates with optimised RCCX gates reduces constant factors.
- Asymptotic depth remains unchanged.
- **Total Circuit Depth:** $O(n^{1.5}m)$
- Observation: Architecture-specific optimisation; no asymptotic improvement.

Depth Comparison Table

Table: Circuit depth for the best split subroutine.

Method	**Circuit Depth**
Classical (sequential)	$O(n^2)$ (when $N = \Theta(n)$)
QFT (Ramezani et al., 2023)	$O(n^{1.5}m^2)$
Approx QFT (Nam et al., 2020)	$O(n^{1.5}m \log m)$
VBE Ripple Carry (Vedral et al., 1996)	$O(n^{1.5}m)$
VBE + RCCX	$O(n^{1.5}m)$

Comparison with Classical

- Classical: $O(n^2)$ when $N = \Theta(n)$.
- Quantum (ripple-carry): $O(n^{1.5}m)$.
- Quadratic speedup in the number of splits (n vs $\sqrt{n}$ iterations).
- If m is logarithmic or polylogarithmic in n, the quantum depth is subquadratic.
- Ripple-carry approach offers the best depth among quantum variants.

Two Implementation Variants

1 **Precalculated values:**

- Classical preprocessing computes counts per split.
- Quantum circuit only evaluates Gini formula.
- Preprocessing cost: $O(nN)$.
- Reduced qubit and depth requirements.

2 **Fully quantum:**

- All operations (counting, impurity, maximum) performed on quantum computer.
- Achieves stated complexities.
- Requires more qubits and fault-tolerant implementation.

Conclusion

- We analysed quantum subroutine for best split selection using Gini impurity.
- Four quantum arithmetic implementations compared.
- Ripple-carry adder achieves lowest asymptotic depth $O(n^{1.5}m)$.
- Provides quadratic speedup over classical $O(n^2)$.
- Step toward fully quantum decision tree learning.