Decision trees are a popular algorithm used for both classification and regression tasks in machine learning. Like any algorithm, decision trees have their pros and cons.

**Pros:**

1. **Easy to Understand and Interpret**: One of the biggest advantages of decision trees is their ease of interpretation. They are highly visual and allow you to make decisions in a transparent manner, following a "yes" or "no" pattern that can be understood even by people without a machine learning background.
2. **Handles Both Categorical and Numerical Data**: Decision trees can handle both types of data without requiring complex preprocessing like normalization or one-hot encoding.
3. **Less Data Cleaning Required**: Decision trees are less sensitive to outliers and missing values, and often deliver reasonably accurate models even when data quality is not perfect.
4. **Fast Training and Prediction**: Decision trees are computationally inexpensive to train and make predictions, which makes them suitable for real-time applications.
5. **Non-Parametric**: They do not make any assumptions about the distribution of data, which makes them suitable for non-linear relationships between variables.
6. **Good for Feature Selection**: High-performing features will end up closer to the tree root, making it easier to identify the most important predictors.
7. **Robust to Noisy Data**: Decision trees can be fairly resistant to "noise" in data, especially if tree depth and complexity are controlled.

**Cons:**

1. **Overfitting**: Decision trees are prone to overfitting, especially when the tree is deep, capturing noise in the data. This makes them less generalizable to new data.
2. **Unstable**: Small changes in data can result in a significantly different tree, making them unstable. This issue can be mitigated by using techniques like bagging or boosting.
3. **Not Good for Unbalanced Classes**: In problems where classes are significantly imbalanced, decision trees may be biased towards the majority class.
4. **Poor Performance on XOR-like Problems**: They don't perform well on tasks that require XOR-like decision boundaries. Such tasks are better suited for algorithms like neural networks.
5. **Biased with Unbalanced Dataset**: When the classes are highly imbalanced, decision trees tend to be biased towards the class with more instances.
6. **Limited to Axis-Aligned Decision Boundaries**: Decision trees split the data along the axes of the feature space, which can make them less flexible for some types of problems.
7. **May Become Too Complex**: Decision trees can become arbitrarily complex and difficult to interpret if not pruned or controlled.
8. **Local Optimality**: The greedy algorithm used to build the tree (like CART, ID3, C4.5) might not always find the globally optimal tree.

Decision trees are often a good starting point for machine learning tasks because of their simplicity and ease of interpretation, but for some tasks, other algorithms like Random Forests, Gradient Boosting, or neural networks might be more appropriate.