Basic Examples and Feature Engineering

George H. Seelinger

ghseeli@umich.edu

ICERM: Machine Learning Seminar

28 October 2025

Goals

- Learn basics of decision tree learning.
- Explore some challenges to using machine learning to mathematical problems.
- Encounter many difficulties, negative results, and arguably trivial results.

"Good Old Fashioned Machine Learning"

Supervised learning:

- Real world data with inputs (or "features") $\mathbf{X}$ and outputs (or "labels") $\mathbf{y}$.
- In practice, split $\mathbf{X} = \mathbf{X}_{\text{train}} \sqcup \mathbf{X}_{\text{test}}$ and matching $\mathbf{y} = \mathbf{y}_{\text{train}} \sqcup \mathbf{y}_{\text{test}}$.
- Learn function f that "fits" $f(x) = y$ from the pair $\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}}$.
- Ideally, for new input x with unknown output y, $f(x) = y$ (or at least $|f(x) - y|$ is small).
- Test ideal situation using withheld pair $\mathbf{X}_{\text{test}}, \mathbf{y}_{\text{test}}$.

Real World Toy Example

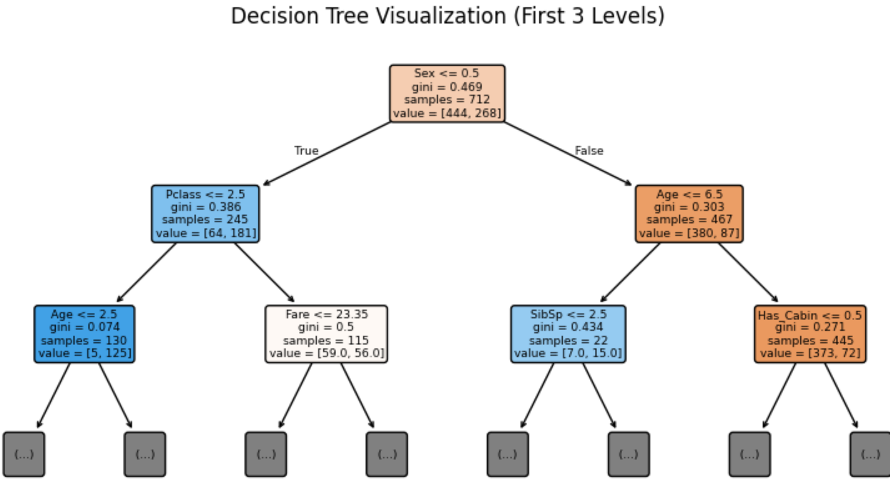
Problem: given a Titanic passenger with some information about them ("features"), predict whether or not they survived.

- This data has noise! Impossible to perfectly predict survivability off knowledge of individual passenger available prior to April 15, 1912.
- However, there can still be detectable trends.

We will use data from Kaggle.

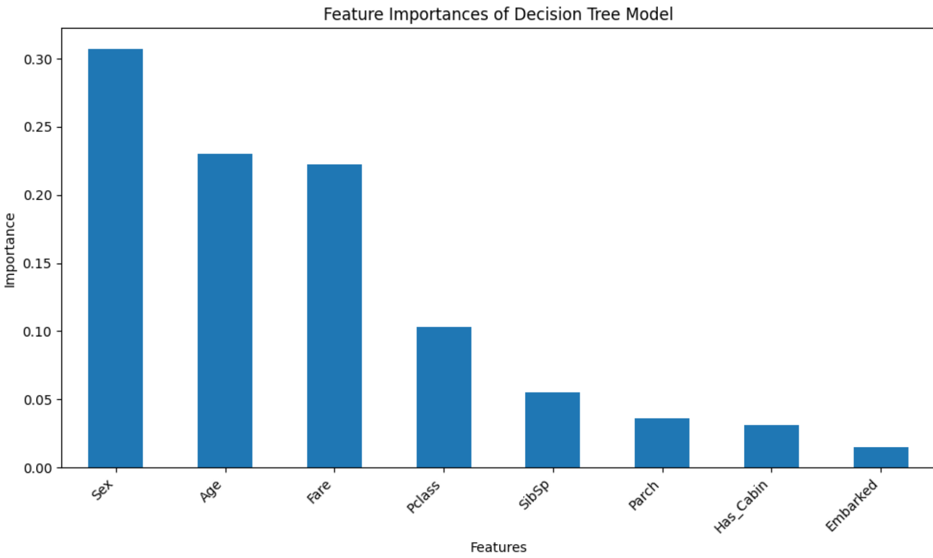
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Decision Trees



Accuracy on withheld test data: 79%

Decision Trees



Decision Trees

Some pros:

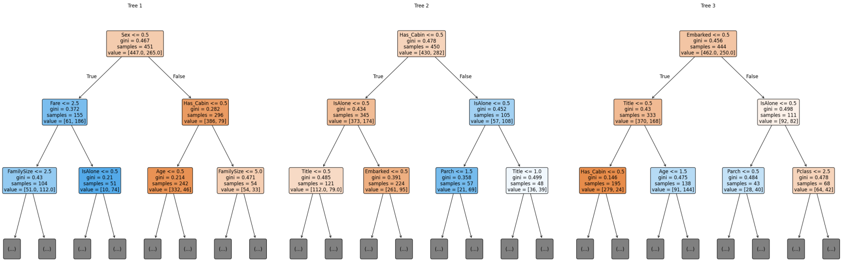
- Relatively easy to understand and interpret.
- Input data requires little preprocessing.

Some cons:

- Highly susceptible to overfitting.
- Cannot detect relationships between features.
- Non-robust: small changes in training data can cause large changes in tree.

Solution 1: Random Forests

- Instead of training a tree, train a forest!
- For any given classification problem, have every tree vote and take the majority vote.
- Harder to visualize, but can still measure importance of features.



. . .

Solution 2: Feature engineering

- If you think there is some relationship between features, you can manually try to add one. ("Derived feature")
- Titanic data lists "# siblings or spouses" as one feature and "# parents or children" as another. Perhaps total family size is more relevant.
- Titanic data lists every passenger's name, including their "Title" (e.g., Mr, Master, Miss, Mrs, etc.). This might be useful to extract for the model.

Solution 2: Feature engineering

- If you think there is some relationship between features, you can manually try to add one. ("Derived feature")
- Titanic data lists "# siblings or spouses" as one feature and "# parents or children" as another. Perhaps total family size is more relevant.
- Titanic data lists every passenger's name, including their "Title" (e.g., Mr, Master, Miss, Mrs, etc.). This might be useful to extract for the model.
- Decision tree accuracy with additional features 79% $\rightarrow$ 80%
- Random forest accuracy with additional features 80% $\rightarrow$ 82%

Solution 3: Gradient boosting

- XGBoost (and other gradient boosted tree libraries) use more advanced techniques to train a decision tree forest in a more sophisticated way to get even better models that are not as likely to overfit.
- Like random forests, some explainability is lost.

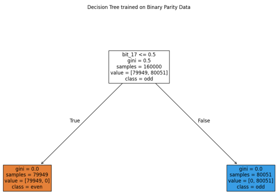
Mathematical Data

- Unlike real world data, mathematical data (often) has no noise.
- However, decision trees are designed to find signal in noise.
- In general, decision tree learning algorithms are designed for interpolating from data, not extrapolating from data.

Mathematical Example 1: is this number even or odd?

- E.g., let $X =$ first N non-negative integers and $y_i =$0 if i is even, 1 if i is odd.
- Training using X as given leads to terrible performance on decision tree.
- Feature engineering: rewrite X as binary sequences $\Longrightarrow$ decision tree model (easily) scores 100%.

Is this number even or odd? (Binary input)



- Decision tree (binary input)

```
Weights from all layers of the binary parity model:

Layer 0 Weights:
[[-3.3569129e-04  4.2004186e-01]
 [-2.7897890e-04  4.1088238e-01]
 [-4.6315661e-05 -4.2145202e-01]
 [-2.4875102e-04  2.3305565e-01]
 [-2.7676253e-04 -3.3683968e-01]
 [ 4.2838839e-04 -2.5057709e-01]
 [-7.6832675e-04 -2.4937712e-01]
 [ 4.6620599e-04 -5.5159688e-01]
 [ 8.2435130e-05 -2.0094021e-01]
 [-1.4284842e-04 -3.6521530e-01]
 [-3.0786879e-04 -3.0613244e-01]
 [ 7.8174911e-05  6.3660979e-02]
 [ 9.5834243e-05 -2.8157067e-01]
 [-4.0383812e-04 -3.3182439e-01]
 [ 1.2446647e+00  1.6162680e-01]]

Layer 1 Weights:
[-0.00014593 -0.12703258]

Layer 2 Weights:
[[2.1112826]
 [0.7268881]]

Layer 3 Weights:
[-0.72701466]
```

- Neural network (binary input)
- https://stats.stackexchange.com/questions/161189/train-a-neural-network-to-distinguish-between-even-and-odd-numbers

Mathematical Example 2: Horn problem

- Schur polynomials $s_\lambda(x_1, \ldots, x_n)$ form a basis of symmetric polynomials as λ varies over partitions: $\lambda = (\lambda_1 \geq \cdots \geq \lambda_n \geq 0) \in \mathbb{Z}^n_{\geq 0}$.
- Littlewood-Richardson coefficients $c^\nu_{\lambda,\mu}$:

$$s_\lambda s_\mu = \sum_\nu c^\nu_{\lambda,\mu} s_\nu$$

  for $c^\nu_{\lambda\mu} \in \mathbb{Z}_{\geq 0}$.
- Horn problem: determine when $c^\nu_{\lambda,\mu} \neq 0$ (support).
- Remark: this is a mathematically solved and well understood problem.
- Can we see how ML models could learn the solution?

Horn problem

Solution (Klyachko, 1998, Knutson-Tao, 1999)

$c^{\nu}_{\lambda\mu} \neq 0 \Longleftrightarrow \sum_{i \in I} \lambda_i + \sum_{j \in J} \mu_j \leq \sum_{k \in K} \nu_k$ for $I, J, K \subseteq \{1, \dots, n\}$ satisfying $|I| = |J| = |K|$ and $|\lambda| + |\mu| = |\nu|$.

Additional resource for Algebraic Combinatorics Data

Algebraic Combinatorics Dataset Repository:
https://github.com/pnnl/ML4AlgComb