Interesting Issues Regarding Machine Learning

Table of Contents

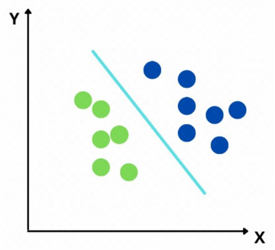
1. What To Do Before Applying An ML Algorithm
2. Four Paradoxes Related to Training
3. Graph Machine Learning
4. Complex Shearlets Network (CoShNet)
5. Liquid Neural Networks
6. Reference

Classification of Algorithms

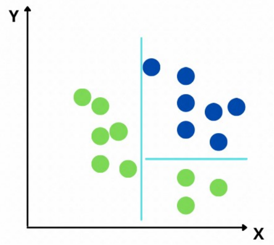
We can classify algorithms based on how they form the decision boundary.

1. Linear-Based : They use linear hyperplanes as decision boundaries to represent the relationship between the feature vectors and their values
   - Ex : linear / logistic regression, linear SVM, neural networks
2. Tree-Based : They determine the decision boundaries by a series of "if-then" rules
   - Ex : decision tree, random forest, AdaBoost
3. Distance-Based : They determine the decision boundaries by the closedness among vectors
   - Ex : K-means, KNN

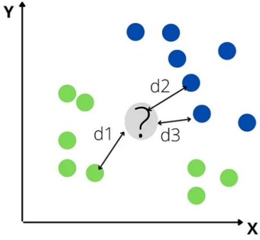
Classification of Algorithms



(a) Linear-Based



(b) Tree-Based



(c) Distance-Based

Exploratory Data Analysis

To understand the patterns between the feature vectors and their values in order to choose a suitable type of algorithms.

1. Statistics
   - percentiles and variance $\rightarrow$ the range for most of the features
   - interquartile range (IQR) $\rightarrow$ identify outliers
   - mean and median $\rightarrow$ the central tendency
   - correlations $\rightarrow$ strong or weak relationship
2. Visualizations
   - box plot $\rightarrow$ identify outliers
   - density plot and histogram $\rightarrow$ spread of the features
   - scatter plot $\rightarrow$ bivariate relationships

Exploratory Data Analysis

1. There are significant outliers
   $\rightarrow$ bad for linear-based approaches while not affecting tree-based approaches
2. There are strong correlations between features
   $\rightarrow$ good for linear-based approaches

Non-Data Related Considerations

1. Data Storage Capacity $\rightarrow$ light or large models
2. Inference Time $\rightarrow$ fast but not accurate or slow but accurate
3. Interpretability vs. Accuracy $\rightarrow$ strike a good balance between the ability to interpret and the accuracy performance

Feature Engineering

1. Outliers Handling
   - directly remove them $\rightarrow$ increases data normality but causes some information loss
   - logarithmic or square root transformation
   - do not affect tree-based approaches
2. Missing Values Handling
   - numerical imputation : for linear-based approaches
     - Ex : mean / median / mode imputation, random imputation, imputation with an arbitrary value
   - categorical imputation : for tree-based approaches
     - imputation with the "unknown" category

Feature Engineering

1. Scaling and Normalization
   - not needed for tree-based approaches
   - quite important for linear-based and distance-based approaches
2. Categorical Encoding
   - one-hot encoding : transforms each category into a binary vector $\rightarrow$ useful for linear-based approaches
   - frequency encoding : replace each category with its frequency in the dataset
   - label encoding : replace each category with a numerical label indicating ordinal relationship between the categories

Table of Contents

1. What To Do Before Applying An ML Algorithm
2. Four Paradoxes Related to Training
3. Graph Machine Learning
4. Complex Shearlets Network (CoShNet)
5. Liquid Neural Networks
6. Reference

Accuracy Paradox

Your model predicts well? Maybe it just blindly guesses!

- Example :

  Assume your model wants to test whether a person is infected with a certain disease.
  Your model just blindly says no.
  However, 90% of your tested people happen to be healthy
  $\Rightarrow$ The accuracy of your model is 0.9!

- Key : imbalance in the distribution of classes in the tested dataset

False Positive Paradox

Your model truly predicts well but only to get a high false positive rate!

- Example :

  Assume your model wants to test whether a person is infected with a certain disease.

	Test Negative	Test Positive	Total
Uninfected	931	49	980
Infected	0	20	20
Total	931	69	1000

  Accuracy : (931+20)/1000=95% ; FP : 49/(49+20)=71%

- Key : rare events

Simpson's Paradox

Your model is better than other models? Maybe it is just lucky to face easy tasks!

- Example :

	Patients	Death Toll	Mortality Rate
A hospital	10000	1500	15%
B hospital	8000	1000	12.5%

	Severe Conditions	Light Conditions
A hospital	20%(1400/7000)	3.3%(100/3000)
B hospital	30%(600/2000)	6.6%(400/6000)

Berkson's Paradox

Don't blindly accept your training data. Be careful with sampling bias!

- Example : Assume your model wants to predict the correlation between happiness and playing tennis
  Total population :
  general public * 900 $\rightarrow$ zero correlation
  people hating outdoor activities * 50 $\rightarrow$ negative correlation
  varsity tennis team members * 50 $\rightarrow$ positive correlation
  What if you can only get 50 pieces of data?

Summary

Don't be too optimistic with your model. Maybe it just learns nothing! (accuracy paradox). Even if it indeed learns something, it may learn from biased data (Berkson's paradox) or from pretty simple task (Simpson's paradox). Even even if your model is truly powerful, it may face rare events...(false positive paradox).

Table of Contents

1. What To Do Before Applying An ML Algorithm
2. Four Paradoxes Related to Training
3. Graph Machine Learning
4. Complex Shearlets Network (CoShNet)
5. Liquid Neural Networks
6. Reference

1. An Introduction to Graphs
2. Graph Machine Learning Tasks
    1. Supervised
    2. Unsupervised
3. Graph Machine Learning Approaches
    1. Classic Graph Algorithms
    2. Non-GNN Embeddings
    3. Graph Neural Networks (GNN)

An Introduction to Graphs

1. Nodes (Vertices) : Entities of a graph data network
2. Properties : Each node is endowed with some properties describing it
3. Links : Nodes and Nodes can be linked together to represent some associations, interactions or relationships between them

Supervised GML Tasks

1. Property Prediction : To predict some discrete or continuous properties of a node, graph or subgraph
   $\rightarrow$ classification or regression tasks

2. Link Prediction : To predict whether there is relationship between any two nodes and possibly some properties about the relationship
   $\rightarrow$ binary classification (has relationship or not)
   multiclass classification (types of the relationship)
   regression (continuous properties of the link)

Unsupervised GML Tasks

1. Representation Learning : To generate low-dimensional feature vectors based on the graph structures and to use them for downstream tasks.
2. Community Detection : A clustering issue about decompose a graph into several groups of densely interconnected nodes
3. Similarity Detection : To find and pair similar nodes in a graph
4. Centrality Detection : To detect important or influential nodes
5. Path Finding : To find paths in a graph with lowest cost or evaluate the quality and feasibility of paths

Classic Graph Algorithms

1. PageRank : To evaluate the degree of importance of each page on the Internet $\rightarrow$ centrality detection; property prediction
2. Louvain : A hierarchical clustering algorithm, which iteratively detects communities in large networks and then merges them into a single node $\rightarrow$ community detection
3. Dijkstra's Shortest Path : To find a path with shortest distance between any two nodes $\rightarrow$ path finding
4. Node Similarity Algorithm : Two nodes are considered similar if their neighbors overlap a lot $\rightarrow$ similarity detection

Non-GNN Graph Embeddings

Graph Embedding : To generate numeric or binary feature vectors to represent nodes, links, paths or entire graphs $\rightarrow$ representation learning

1. Node2Vec : To compute a vector representation of a node based on random walks in the graph
2. FastRP : A node embedding algorithm based on random projections
3. HashGNN : A node embedding algorithm which replaces GNN with random hash functions

Graph Neural Network (GNN)

- Takes graph data as input
- Embeddings of hidden layers can be used for representation learning and be fed to downstream tasks

Table of Contents

1. What To Do Before Applying An ML Algorithm
2. Four Paradoxes Related to Training
3. Graph Machine Learning
4. Complex Shearlets Network (CoShNet)
5. Liquid Neural Networks
6. Reference

1. CNN extracts crude features, e.g., edges and colors via lower-level convolution layers while high-resolution features are spotted through higher-level convolution layers.

2. Convolution is quite computation-demanding. Whether we can instead use some mathematical functions to extract crude features? In this way, we can greatly reduce the number of parameters and also the training time, i.e. lighter and faster
$\rightarrow$ complex shearlet transform (CoShRem)

3. CoShNet replaces those computation-demanding convolution layers with a non-trainable fixed complex shearlet transform (CoShRem)
$\rightarrow$ hybrid neural network

1. A complex neuron has two orthogonally intersected decision boundaries, which give rise to four decision regions. Because of this property, a complex neuron can solve more problems, e.g. xor, than a real neuron.
2. Shearlet transform can detect crude features of images well.
3. CoShRem can stably extract features because of the phase congruency property. Thus CoShNet is resilient to noise and perturbation, which also implies it can avoid over-fitting.
4. Since the number of layers is greatly reduced, this network do not suffer from vanishing gradients
5. Furthermore, this network is claimed not to need any hyperparameter tuning or regularization.

Related Papers

- CoShNet: A Hybrid Complex Valued Neural Network using Shearlets
  https://arxiv.org/abs/2208.06882

- DeepRoto-TranslationScatteringforObjectClassification
  https://reurl.cc/nLDokv

- ShearLab 3D: Faithful Digital Shearlet Transforms Based on Compactly Supported Shearlets
  https://dl.acm.org/doi/abs/10.1145/2740960

Related Papers

- Edge, Ridge, and Blob Detection with Symmetric Molecules
  https://epubs.siam.org/doi/abs/10.1137/19M1240861
- Alpha molecules: curvelets, shearlets, ridgelets, and beyond
  https://reurl.cc/p56xWQ
- Image Features From Phase Congruency
  https://reurl.cc/V4L5Ry

Table of Contents

1. What To Do Before Applying An ML Algorithm
2. Four Paradoxes Related to Training
3. Graph Machine Learning
4. Complex Shearlets Network (CoShNet)
5. Liquid Neural Networks
6. Reference

1. Developed by a group of researchers at MIT's Computer Science and Artificial Intelligence Laboratory (CSAIL)
2. Have good potential to process time series data, e.g. video, language and financial
3. The decision making process is explainable and interpretable
4. Liquid neural networks can adjust their architecture, neuron count, and connections in order to adapt to varying input data and changing circumstances, just as dynamic as liquid.
5. The model size is quite small compared with traditional neural networks
6. Resilient to noise and perturbations in input signals

Related Papers

- Liquid Time-constant Networks
  https://ojs.aaai.org/index.php/AAAI/article/view/16936
- Liquid Structural State-Space Models
  https://arxiv.org/abs/2209.12951
- Closed-form continuous-time neural networks
  https://www.nature.com/articles/s42256-022-00556-7

Related Papers

- Causal Navigation by Continuous-time Neural Networks
  https://reurl.cc/ed3j8R

- Designing Worm-inspired Neural Networks for Interpretable Robotic Control
  https://ieeexplore.ieee.org/abstract/document/8793840

- Robust flight navigation out of distribution with liquid neural networks
  https://www.science.org/doi/abs/10.1126/scirobotics.adc8892

Table of Contents

1. What To Do Before Applying An ML Algorithm
2. Four Paradoxes Related to Training
3. Graph Machine Learning
4. Complex Shearlets Network (CoShNet)
5. Liquid Neural Networks
6. Reference

Reference

- So, which ML Algorithm to use?!
  https://medium.com/@aaabulkhair/so-which-ml-algorithm-to-use-d2484239f448
- 5 Paradoxes in Statistics Every Data Scientist Should be Familiar With
  https://pub.towardsai.net/5-paradoxes-in-statistics-every-data-scientist-should-be-familiar-with-478b74310099
- False positive paradox
  https://kharshit.github.io/blog/2018/10/12/false-positive-paradox

Reference

- 數據會說話? 淺談辛普森悖論
  https://reurl.cc/GKA46y
- Graph Machine Learning: An Overview
  https://towardsdatascience.com/graph-machine-learning-an-overview-c996e53fab90

Reference

Improve Neural Networks by using Complex Numbers
https://medium.com/geekculture/improve-neural-networks-by-using-complex-numbers-5e142b8931e6

Liquid Neural Networks
https://medium.com/mlearning-ai/liquid-neural-networks-33b7b6b0f528