CS 8803-MDS
Human-in-the-loop Data Analytics

Lecture 17

10/24/22

Today's class

Hypothesis testing

Slice Finder: Automated Data Slicing for Model Validation

- Authors: Andrew
- Reviewer: Shen En
- Archaeologist: Qiandong
- Practitioner: Bojun

Hypothesis testing

- If the system recommends a discovery to users, how do we know if the event is ”significant”
- When users explore datasets for “interesting” events, they are performing hypothesis tests without knowing it

Relevant papers:

MacroBase: Prioritizing Attention in Fast Data

Slice Finder: Automated Data Slicing for Model Validation

Investigating the Effect of the Multiple Comparisons Problem in Visual Analysis

Hypothesis testing

Null hypothesis H_0: No effect on the population/no relationship between variables. Commonly accepted facts.

Example H_0: $\mu = \mu_0$, $\mu_1 = \mu_2$

Alternative hypothesis H_1: The opposite of null; also called "research hypothesis"

Example H_1: $\mu \neq \mu_0$ (two-tailed), $\mu < \mu_0$ (left-tailed), $\mu > \mu_0$ (right-tailed)

Hypothesis testing

Significance level α: Common values: 0.01. 0.05, 0.1.

P-value: Probability that your data would have occurred by random chance, *assuming that the null hypothesis is true*

P-value $<$ α: reject null hypothesis

P-value $>$ α: not enough evidence to reject null hypothesis

does not necessarily mean null hypothesis is true

Hypothesis testing

Effect size: Strength/magnitude of relationship (significant effects might not always be substantial)

Correlation family: based on variance explained

- Pearson r correlation (for paired data)
- Coefficient of determination (r^2)

Difference family: based on differences between means

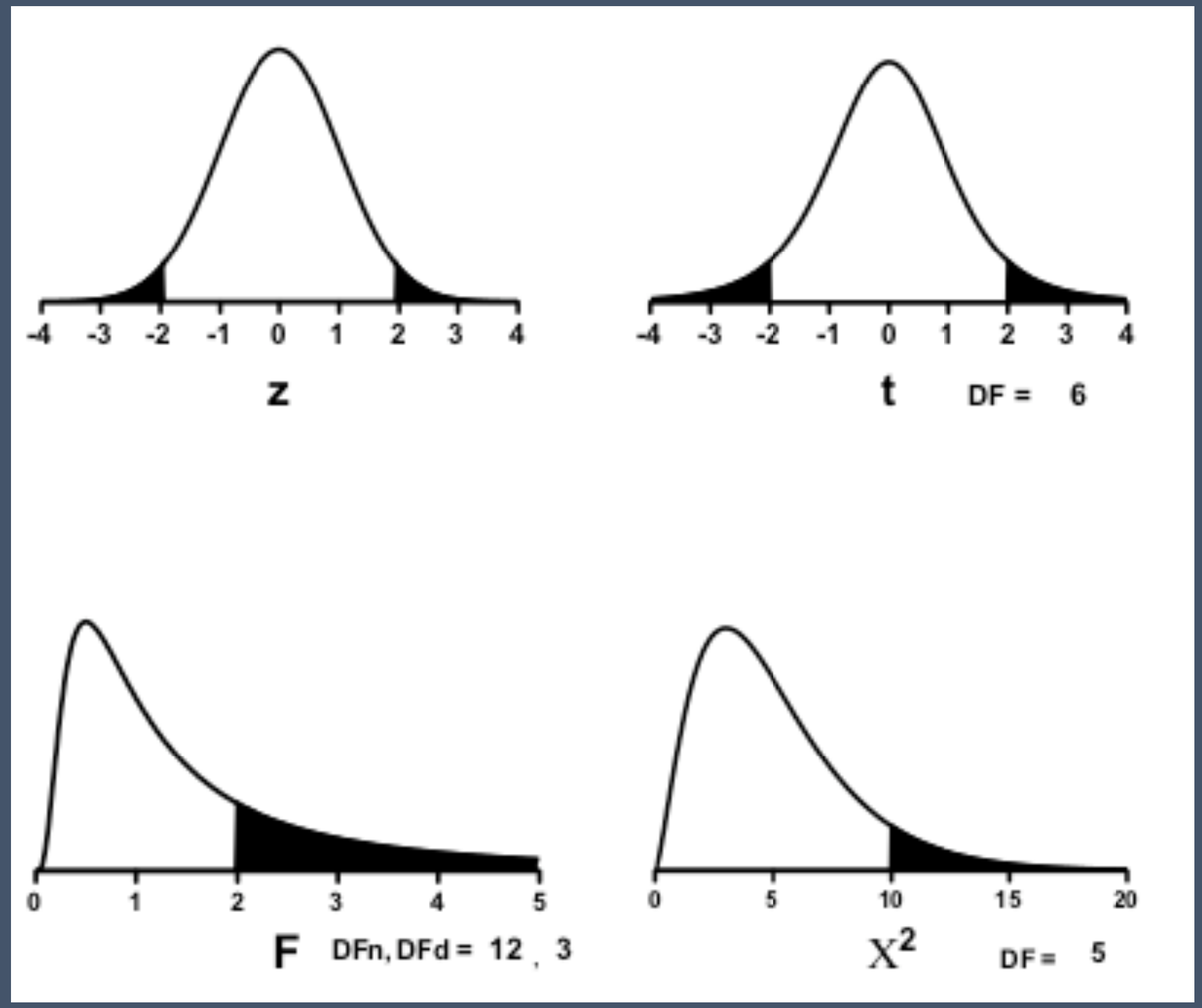
- Cohen's d = mean difference / standard deviation

How to calculate p-value

1. Pick **alternative hypothesis**: two tailed, right-tailed, left-tailed

2. Determine ***distribution of test statistics*** under the null hypothesis: normal, student-t, chi-squared etc.

3. Optional: specify ***degrees of freedom***

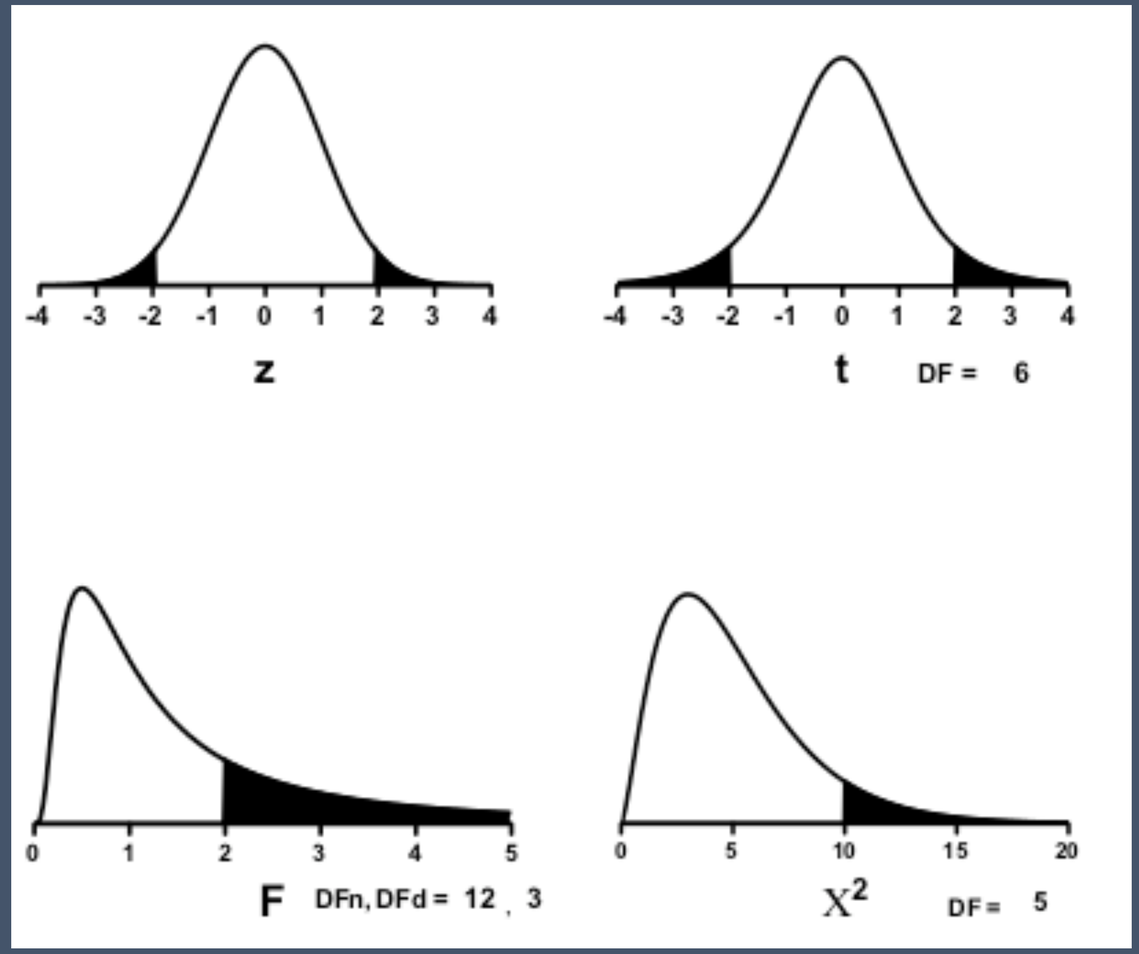


How to calculate p-value

Z-test (normal distribution): population mean

T-test (Student-t): population mean with unknown variance

χ^2 ***test:*** variance of normal distributions; independence test; goodness-of-fit test



Hypothesis testing

Type I Error:

false positive, reject a null hypothesis that is true

concluding results are statistically significant when they are not

Type II Error:

false negative, fails to reject a null hypothesis that is false

Probability of making a type I error = α (significance level)

Multiple hypothesis testing

Applies to scenarios in which a statistical analysis involves multiple simultaneous statistical tests, each of which has a potential to produce a "discovery."

A stated confidence level generally applies only to each test considered individually, but often it is desirable to have a confidence level for the whole family of simultaneous tests.

Failure to compensate for multiple comparisons can lead to false discoveries.

Multiple hypothesis testing

Example: There are 20 options we are interested in as independent (predictor) features for your model. For each feature, we use a hypothesis test with level of significance 0.05.

What's the probability of having one significant result just due to chance? (Recall that probability of making a type I error is α)

$$1 - (1-0.05)^{20} = 0.64$$

Multiple hypothesis correction

Family-wise error rate (FWER) correction: control the probability for at least one Type I error

- Bonferroni Correction: control the **α** by divide it with the number of the testing/number of the hypothesis for each hypothesis.

$$\alpha_{bon} = \alpha / n$$

Multiple hypothesis correction

False Discovery Rate (FDR): control the expected Type I error proportion

- Benjamini-Hochberg (BH) correction method: the **α** level correction is not uniform for each hypothesis testing; instead, it was varied depending on the P-value ranking

$$P_k = \frac{k}{n}\alpha$$

Question

- What hypotheses were tested in MacroBase?
- Is multiple hypothesis testing problem a concern in MacroBase?

Today's class

Slice Finder: Automated Data Slicing for Model Validation

Authors: Andrew

Reviewer: Shen En

Archaeologist: Qiandong

Practitioner: Bojun

Automated Data Slicing for Model Validation

Authors:
Yeounoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijong Whang

Presenter: Andrew Zhao

Problem Statement

Assumption: ML model performance applies equally (more or less) to all the data

Reality: ML Models can fail spectacularly on certain subsets of data

Challenge: How can we find **interpretable** slices that are both **problematic** and **sufficiently large**?

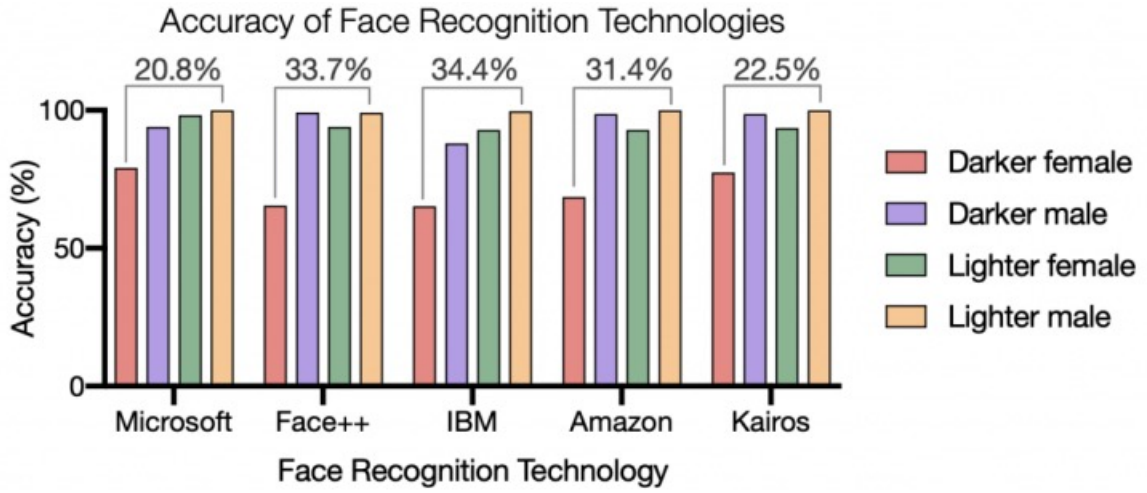
Importance – AI Fairness

There are existing, high-impact AI systems that suffer from fairness issues

RETAIL OCTOBER 10, 2018 / 7:04 PM / UPDATED 4 YEARS AGO

Amazon scraps secret AI recruiting tool that showed bias against women

By Jeffrey Dastin 8 MIN READ



(Najibi A., 2020)

INFOGRAPHICS

Racial Bias in Health Care Artificial Intelligence

Health Equity

Published on: September 30, 2021.

https://nihcm.org/publications/artificial-intelligences-racial-bias-in-health-care

Other Applications

- Fraud detection – Changes in slice performance can indicate fraud

- Business Analytics – Which slices are the most promising

- Data cleaning – The scoring function can generalize to any metric (NaN values, out of range)

Set up the Bit (Existing Solutions)

	Manual Labelling by Domain Experts	Search Lowest Performance Slices	Clustering
Pros	- Likely high performance - Makes normative sense	- Easy implementation	- Natural way of grouping data - Captures multi-dimensional relationships
Cons	- Can miss important slices - Many ML applications don't have domain experts	- Smaller slices have noise (false positives) - Small slices are uninterpretable and low-impact	- Difficult to interpret clusters - Hard to tune # of clusters (can be hard to find a balance between size of clusters and effect size)

Bitflip

- What if we design an algorithm that slices along features (interpretable), and maximizes both size of the slices (interpretable) and statistical significance while minimizing features (interpretable)?

Statistical Significance + Effect Size

Binary Loss Function

$$\psi(S,h) = -\frac{1}{n}\sum_{(x_F^{(i)},y^{(i)})\in S}[y^{(i)}\ln h(x_F^{(i)}) + (1-y^{(i)})\ln(1-h(x_F^{(i)}))]$$

- S = Slice, S' = D – S, rest of examples

- Only look for positive differences where S is higher than S'
- Welch's t-test = see if slice's difference is **statistically significant**

Welch's t-test

$$t = \frac{\mu_S - \mu_{S'}}{\sqrt{\sigma_S^2/|S| + \sigma_{S'}^2/|S'|}}$$

- Effect Size: Captures the loss distribution difference between two slices
  - Captures "how much two slices' distributions are different"

Effect Size

$$\phi = \sqrt{2}\times\frac{\psi(S,h)-\psi(S',h)}{\sqrt{\sigma_S^2+\sigma_{S'}^2}}$$

<u>Effect Size Interpretability</u>

Small: 0.2
Medium: 0.5
Large: 0.8
One standard deviation: 1.0
Very Large: 1.3

Ordering

- We want to order slices by ↓ minimizing literals (features to slice on), while ↑ maximizing slice and effect size

Definition 1. *Given a positive integer k, an effect size threshold T, and a significance level α, find the top-k slices sorted by the ordering $\prec$ such that:*

(a) Each slice S has an effect size at least T,

(b) The slice is statistically significant,

(c) No slice can be replaced with one that has a strict subset of literals and satisfies the above two conditions.

Tip: Users can input different effect size thresholds to test on

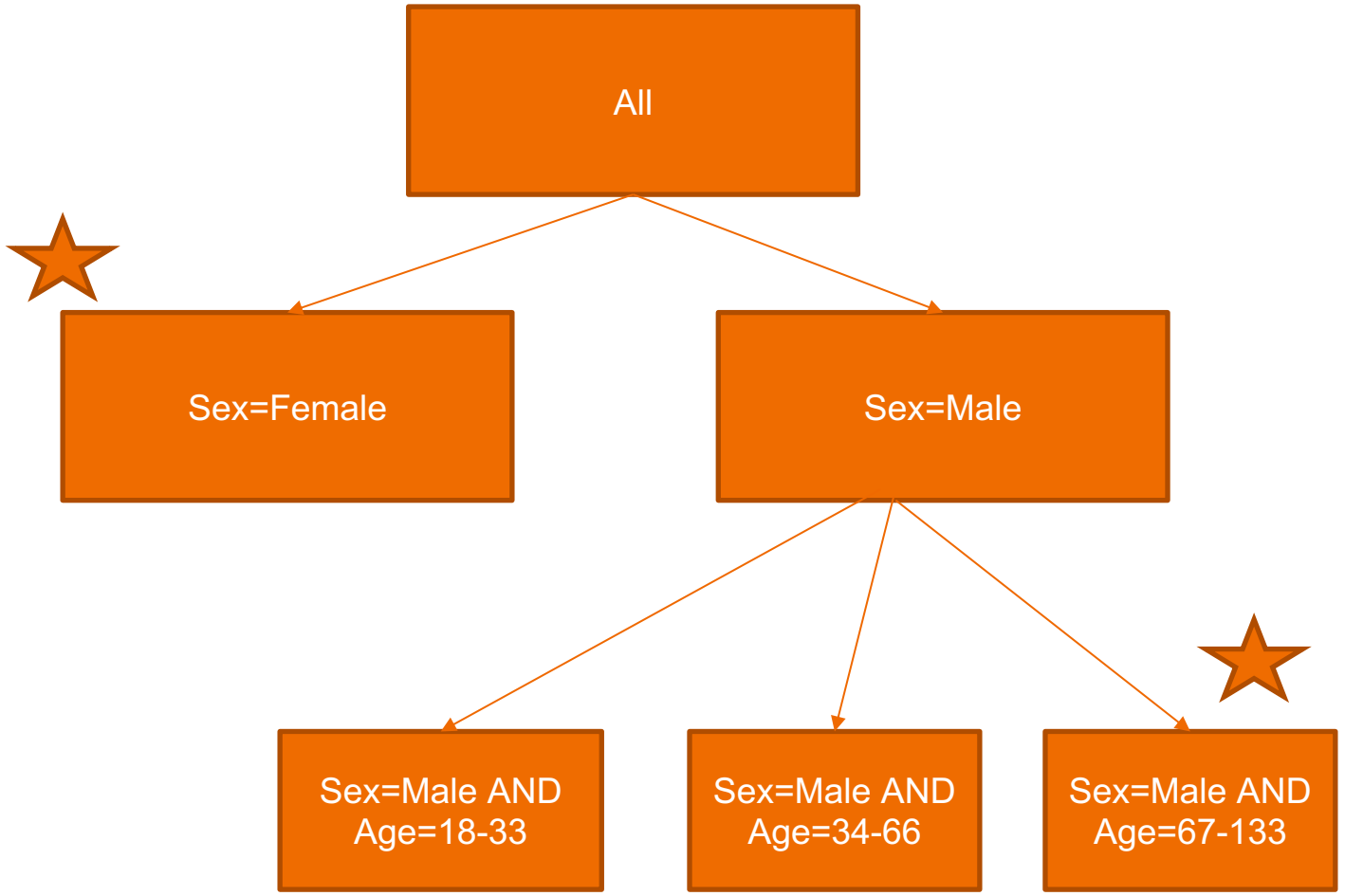
Decision Tree Training

Sex	Age	Education
F/M	[18,33), [33, 66), (66, 133]	B.S./M.S./PHD

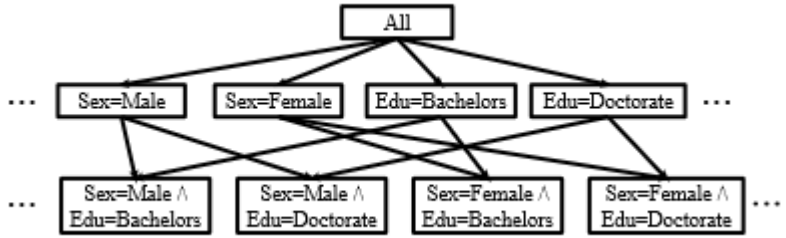
- Prioritize splits on minimizing impurity

(I think this is just data splits)

- Non overlapping slices



Lattice Searching



- BFS Search down by level, one level at a time (initialize E = exploration set with root node)
- 1. For each node in E, check that effectSize > T and add it to the priority queue
  Priority queue based on ordering previously defined (↓ literals, ↑ slice and effect size)
- 2. Pop slices from the priority queue, testing statistical significance (with additional a-investing to control for False Discovery)
  If they are, add them to the top-k problematic slices, otherwise they're not problematic
- 3. Find new slices to explore by adding 1 literal to non-problematic slices
- 4. Repeat

False Discovery Control (alpha-investing)

- Motivation: statistical significance applies to singular hypotheses, and testing multiple hypotheses is prone to false positives (not statistically significant slices being labeled as significant)

- Alpha-investing = Allocate the alpha wealth (error rate) over multiple tests, with increased alpha as each hypothesis is rejected
    Makes the investing less conservative and the test to more likely guess false positives

- Best-foot-forward policy – invests alpha-wealth into early hypotheses since those are likely to be significant and true

- Good approach to manage false positives w/ unknown # of tests in any order

Experiments – Datasets/Models

- Census Income Classification – Random forest classifier to predict whether the income exceeds $50K based on census data (15 features and 30K examples)

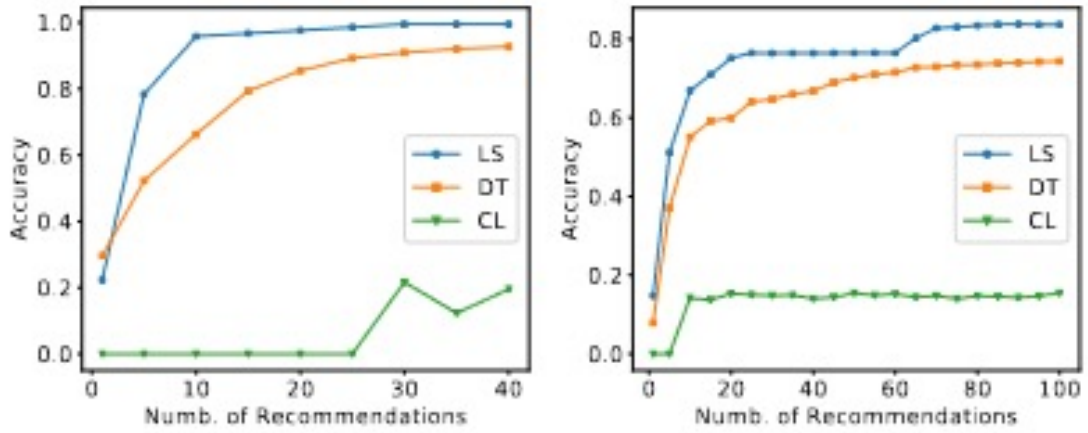
- Credit Card Fraud – Random forest classifier to predict credit card fraud (492 frauds out of 284k, 29 features)
  Balanced dataset by sampling non-fraudulent data (final = 984 transactions, 492 frauds)

- Synthetic Dataset

  - Two discrete features F1 and F2, with perfect classes 0 and 1 (model is perfect)

- Create ground truth problematic slices by messing up labels along slices (50% probability to flip label)

Experiments - Accuracy

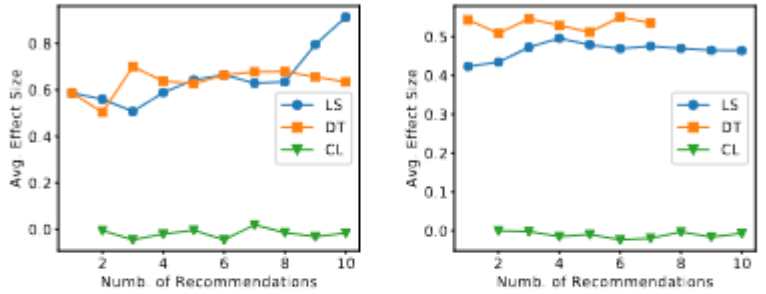


(a) Synthetic Data (b) Census Income Data

Fig. 4: Accuracy comparison of finding problematic slices using (a) synthetic data and (b) real data.

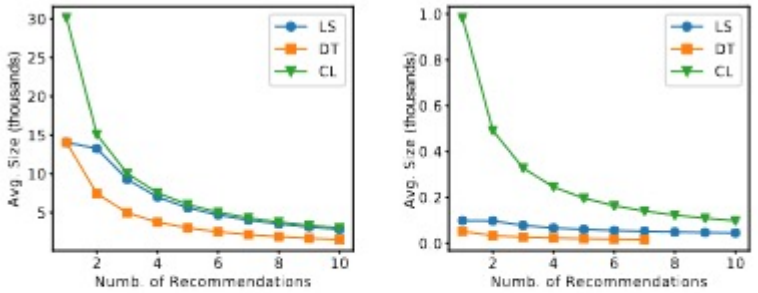
Finding Large, Problematic Slices

- LS and DT outperform CL in effect size and slice size
- Census Income data – both find k=10 well
- Credit Card Fraud – DT runs into issues finding slices due to non-overlapping, that's why it has high effect size
- DT may have to search more levels than LS because its not complete



(a) Census Income Data (b) Credit Card Fraud Data

Fig. 5: Effect size comparisons between different data slicing approaches ($T = 0.4$).



(a) Census Income Data (b) Credit Card Fraud Data

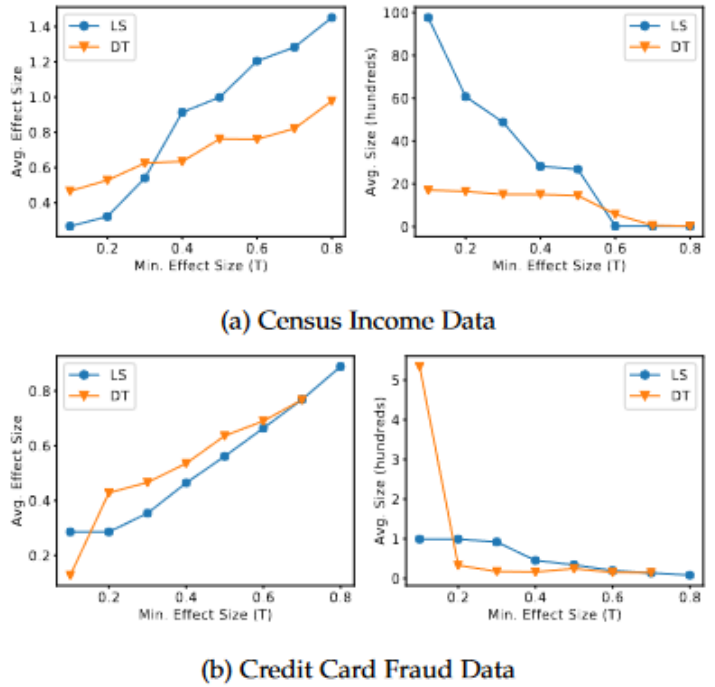
Fig. 6: Average slice size (unit is 1000) comparisons between different data slicing approaches ($T = 0.4$).

Effect Size Threshold

- Lower T = more problematic slices
- Census Income: LS finds larger slices, but less effect size
    - After T = 0.4, LS does find higher effect size

- Credit Card Fraud
    - DT has to search many levels of decision tree, so

  Big drop in average slice size (but then high effect size)



(a) Census Income Data

(b) Credit Card Fraud Data

Fig. 7: The impact of adjusting the effect size threshold T on average slice size and average effect size.

Runtime

- Good performance w sufficient examples
- Parallelization has linear improvements
- Recommendations make LS exponentially explode

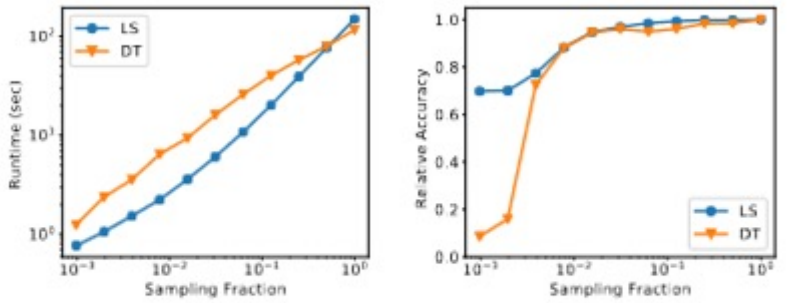
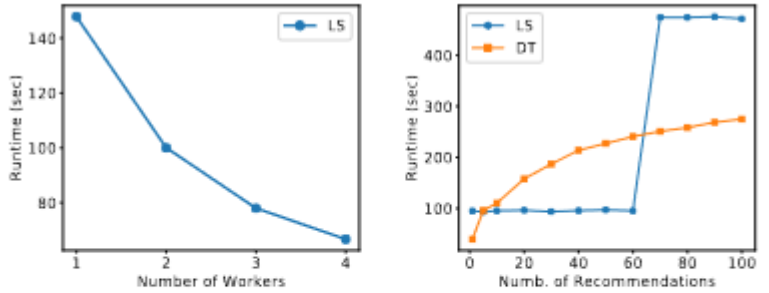


Fig. 8: Slice Finder (LS, DT) runtime (on a single node) and accuracy results using different sample sizes (Census Income data).



(a) Parallelization (b) # Recommendations

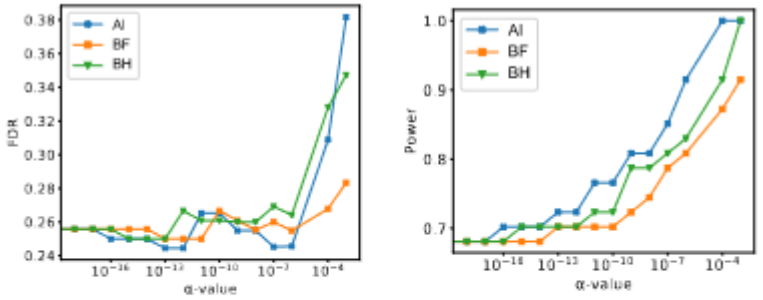
Fig. 9: (a) Slice Finder runtime results with increasing number of (a) parallel workers and (b) recommendations (Census Income data).

FDR Performance and Power

- Alpha Investing (AI) performs better than Benjamini-Hochberg procedure (BH) and Bonferroni correction (BF)

- Power = probability test rejects null hypothesis

- AI + BH is better in FDR, AI has greatest power



(a) False Discovery Rate (b) Power

Fig. 10: (a) False discovery rate and (b) power comparison of the Bonferroni, Benjamini Hochberg, and α-investing techniques (Census Income data).

Takeaways

- We have proposed two algorithms, decision tree training and lattice searching, for maximizing interpretability by finding a small set of highly significant, large slices for ML scientists
- Decision tree training is non-overlapping, …
- You can use the interactive visualization now!
  - Please give us feedback ☺

- Why should people care about your work?
- What are the key technical challenges and solutions?
- How did you evaluate your hypothesis?
- What are the main takeaways?

Bibliography

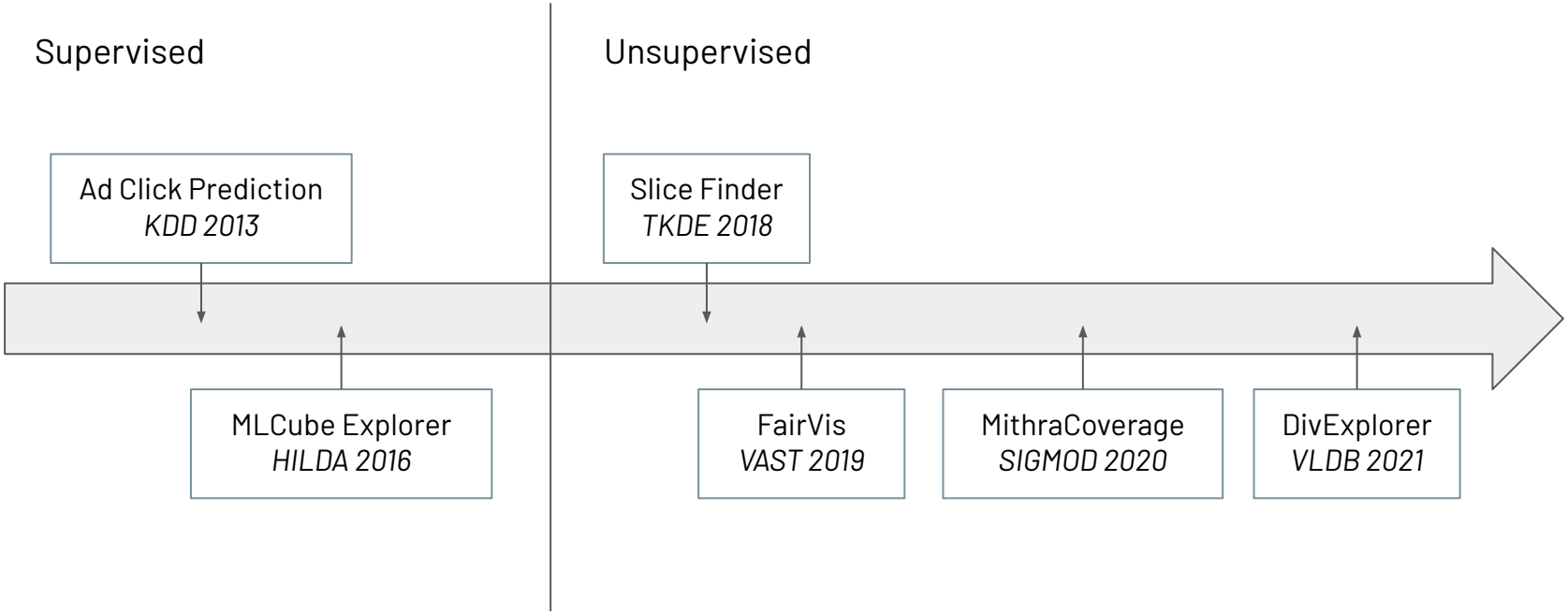
- Najibi, A. (2020, October 26). *Racial discrimination in face recognition technology*. Science in the News. Retrieved October 24, 2022, from https://sitn.hms.harvard.edu/flash/2020/racial-discrimination-in-face-recognition-technology/

Automated Data Slicing for Model Validation: A Big Data - AI Integration Approach

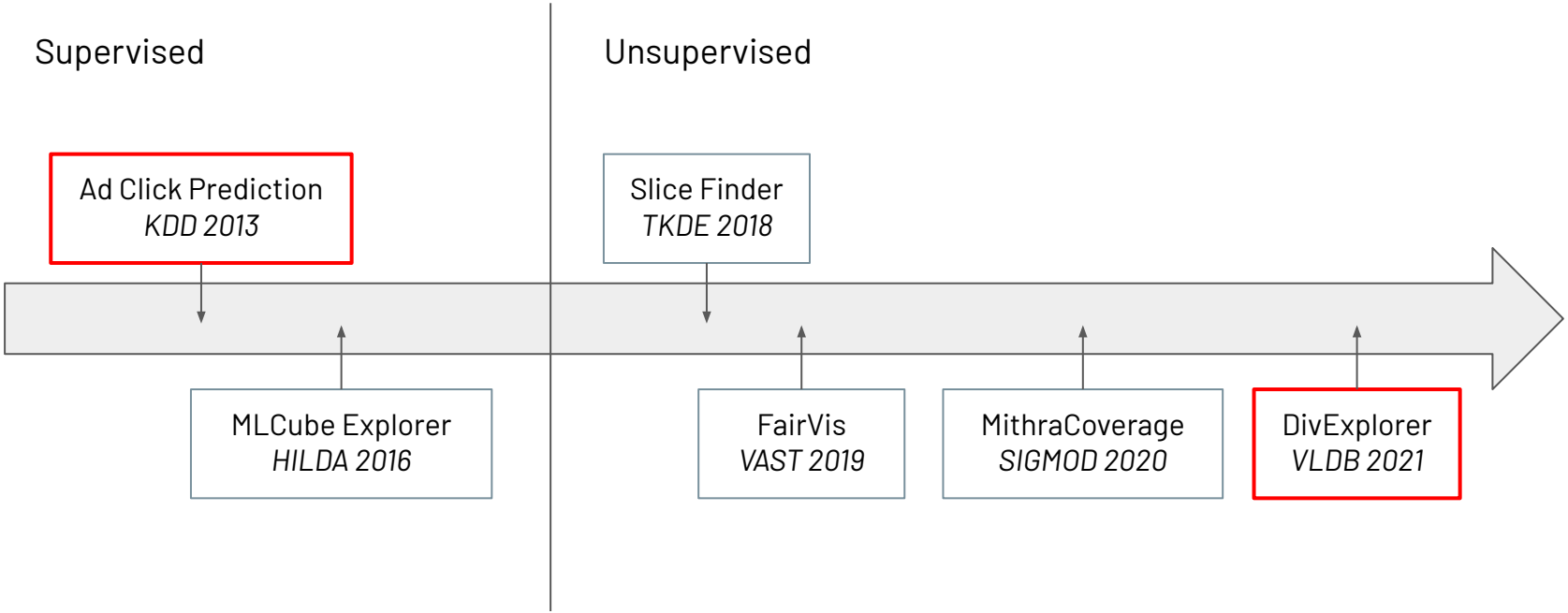
Archaeologist: Qiandong Tang

How to identify subgroups of data where a model performs differently?

Timeline


Supervised
Unsupervised
Ad Click Prediction
KDD 2013
Slice Finder
TKDE 2018
MLCube Explorer
HILDA 2016
FairVis
VAST 2019
MithraCoverage
SIGMOD 2020
DivExplorer
VLDB 2021

Timeline

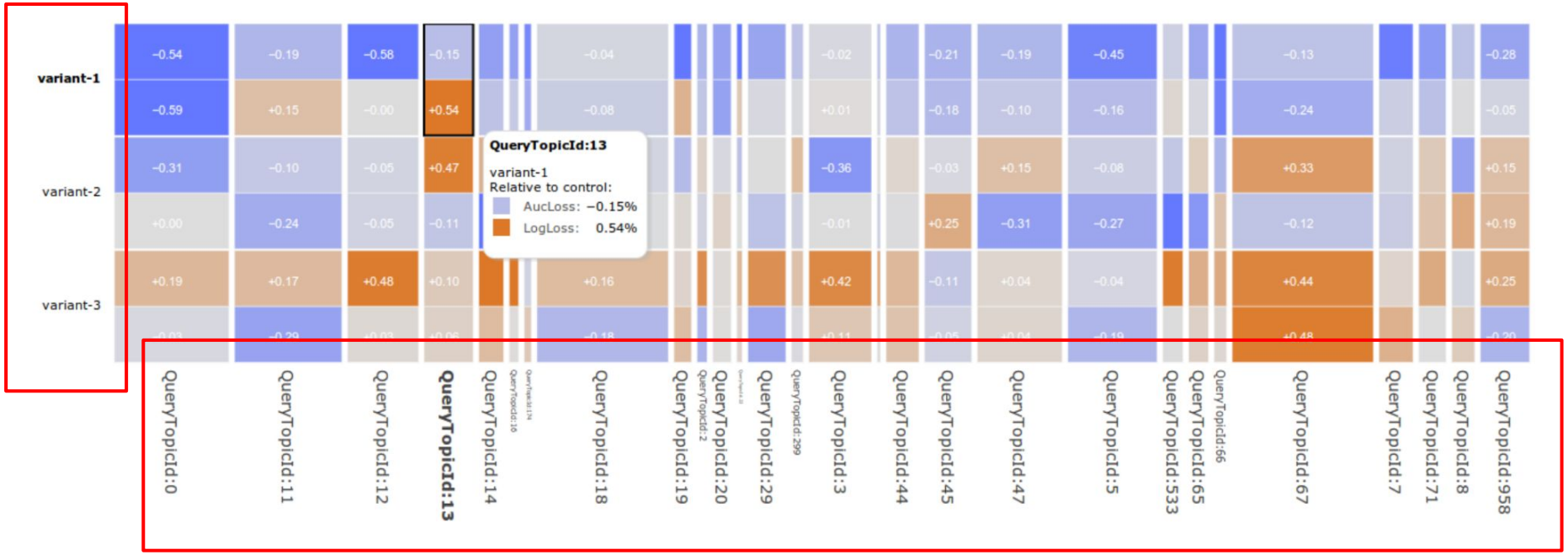

Supervised
Unsupervised
Ad Click Prediction
KDD 2013
Slice Finder
TKDE 2018
MLCube Explorer
HILDA 2016
FairVis
VAST 2019
MithraCoverage
SIGMOD 2020
DivExplorer
VLDB 2021

Ad Click Prediction - a View from the Trenches

- Present an in-depth case studies drawn from experiments in the setting of the deployed system used at Google to predict ad click-through rates
- In massive scale learning, the overall model performance may hide effects that are specific to some subsets of data
- Developed GridViz, a high-dimensional visualization tool that allows users to choose subsets and metrics for comparing models

GridViz

Model names



Slices of data

GridViz

Weakness:

- Simply provide a dropdown menu / regular expression to select slicing groups
  - Manual approach - Users have to be domain experts, could be time-consuming

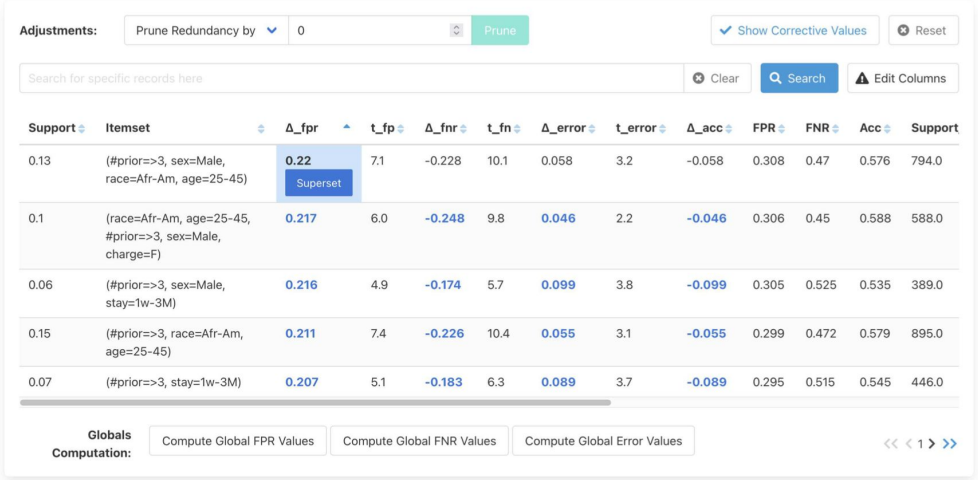
Strength:

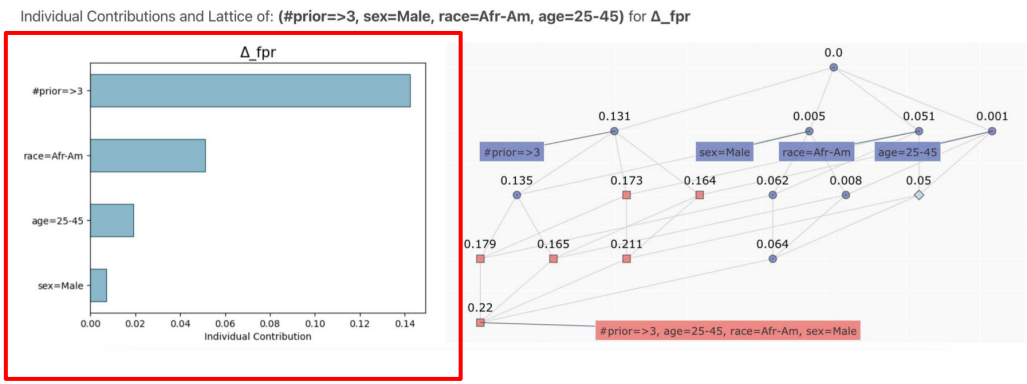
- Help engineers dramatically increase the depth of understanding for model performance on a wide variety of subsets of the data, and to identify high impact areas for improvement

How Divergent is your Data?

- Propose DivExplorer, a visual analytics tool that automatically identifies and **inspects** subsets of data where a model performs differently
- Key Differences:
  - Completely explore all divergent subgroups that are adequately represented in the dataset, selected by a frequency threshold
  - Allow users to analyze the factors that contribute to the problematic performance

DivExplorer





Summary

- Ad Click Prediction - a View from the Trenches

  How to identify subgroups of data that a model performs differently?

- Automated Data Slicing for Model Validation: A Big data - AI Integration Approach

  How to automatically identify subgroups of data...?

- How Divergent is Your Data?

  How to automatically identify and inspect subgroups of data...?

References

- H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin *et al.*, “Ad click prediction: a view from the trenches,” in *KDD*, 2013, pp. 1222–1230.
- M. Kahng, D. Fang, and D. H. P. Chau, “Visual exploration of machine learning results using data cube analysis,” in *HILDA*. ACM, 2016, p. 1.
- Yeounoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijong Whang. 2020. Automated Data Slicing for Model Validation: A Big Data - AI Integration Approach. IEEE Transactions on Knowledge and Data Engineering 32, 12 (2020), 2284ś2296.
- ÁngelAlexanderCabrera,WillEpperson,FredHohman,MinsukKahng,Jamie Morgenstern, and Duen Horng Chau. 2019. FairVis: Visual analytics for discovering intersectional bias in machine learning. In 2019 IEEE Conference on Visual Analytics Science and Technology (VAST). IEEE, 46ś56.
- Pastor, Eliana, et al. "How divergent is your data?." *Proceedings of the VLDB Endowment* 14.12 (2021): 2835-2838.

Slice Finder

Practitioner - Bojun Yang

Summary

- Tool for efficiently and accurately finding large, significant, and interpretable slices of data that a particular model performs badly on
- Model validation, model fairness, fraud detection
- Human interpretability → understanding model behavior

Company

- We make ML platforms for users to quickly build, train, and deploy models
- Similar to AWS Sagemaker

Sample user workflow

1. Prepare data: wrangle data, aggregate, etc
    a. Process data: built in python, spark
    b. Detect data bias
2. Build/test: using pre-built algorithms or write your own
    a. UI based discovery and training of models
    b. Slice finder: use slices to debug model and improve model understanding
3. Train and tune
    a. Hyperparameter tuning
4. Deploy and manage model

Pros

- Easy to implement in our architecture since we already have data processing architecture
- We already offer bias detection and model understanding
- Useful for the user to debug model bias and validation during the model building process
- More explainable models give users better insights, applicability, and adoption of ML models
- Support overlapping and non-overlapping slice detection

Cons

- Sampling needed to improve runtime, but accuracy is lower
- No parallelization speedup comparisons
  - Current decision tree approach does not support parallelism
- Only supports/tested on binary classification tasks with log loss
- False positives (non-problematic slices labeled as problematic) might confuse users
- Users may not know how to effectively set effect size threshold

Automated Data Slicing for Model Validation: A Big Data - AI Integration Approach

Peer Review

Shen En Chen

Summary of Contribution

The authors of the paper extended their previous work **Slice Finder**, a **data slicing system** that models the identification of problematic data slices as **hypothesis testing** and controlling false discovery rates with **α-investing**. The system improves upon the baseline clustering-based approaches with two novel algorithms using **decision trees and lattice searching**, respectively. Both **outperform clustering** in terms of accuracy, interpretability, and problematicity of identified slices and can **couple with parallelization and sampling to increase scalability**. The system also comes with **GUI** that helps user quickly browse through problematic slices and their summaries. In terms of use cases and evaluation, the authors presented **model fairness as one potential use case** and **evaluated the models on both synthetic and real data**. Effectively, Slice Finder is a data slice discovery tool that (1) navigates through the search space efficiently to (2) provide understandable slices that are (3) large enough to have a non-negligible impact on the overall model quality with (4) false discovery control, allowing machine learning practitioners to **interpret and debug models on a more granular level**.

Strong Points

1. The authors formulated the problem of identifying problematic slices as **hypothesis testing**, evaluating a slice on the **statistical significance** of its relative losses and the **effective size**. The former measures the existence of differences in the objective loss and the latter measures the magnitude of it.
2. Slice Finder incorporates **α-investing** (best-foot-forward policy) to find statistically significant slices among a stream of slices by making the procedure become less conservative and putting more weight on more likely to be faulty null hypotheses.
3. Slice Finder provides two non-clustering data slicing approaches: **decision tree and lattice searching**. The latter offers more flexibility by **allowing overlapping slices**. Both solve provide **high interpretability** unlike clustering-based approaches and can **search top-down** efficiently.
4. The authors recognized the lack of “ground truth” for true problematic slices in real datasets and experimented with the system on **synthetic data in addition to real data** to standardize the ground truths.
5. The **GUI** allows users to interact with the slicing results and tune the system.
6. The system is **able to generalize on unstructured data** that contains annotations/metadata that are analogous to columns in tabular data.

Opportunities for Improvement

1. Using sampling to increase scalability is justified empirically with preserved slicing performance instead of proven theoretical bounds.
2. Slice Finder seems to be designed only for balanced datasets.
3. For lattice searching, it is unclear how the bin sizes are determined when discretizing numeric values.
4. More user studies can be done to evaluate how helpful the slices are for practitioners to explain and debug models.
5. Support for merging and summarization of slices may help practitioners to combine slices that are too granular for better interpretation and summarization.
6. The data slices are characterized by a conjunction of literals with interval operators. This means that the slice interpretability depends on the feature interpretability. The slices would still be difficult to understand if they were sliced on engineered features that have no trivial meaning.

Accept

Discussion

- How are the data slices similar/different from explanations generated from MacroBase?
- Can you use MacroBase to find data slices?

Next class

Domino: Discovering Systematic Errors with Cross-Model Embeddings

Authors: Cuong, Jingfan

Reviewer: Tanya, Abhinav

Archaeologist: Sankalp

Researcher: Shubham