Evaluating supervised learning models

CS771: Introduction to Machine Learning

Nisheeth

Accuracy can be inaccurate

- Let's say a biomed startup claims to be able to detect covid-19 cases from thermal scanning cameras with 99%+ accuracy
- What they actually do is classify everyone they scan as covid negative
- The real number of covid positive cases in the population is much smaller than 1%
- So the test data will also contain very few covid positive examples, and calling all examples covid negative will still yield a 99%+ accuracy
- How to fix?

A better alternative

- Ideally, our classification model will have mostly true positives and true negatives
- Should have few false positive and false negatives
- Often possible to reduce false negatives by allowing more false positives and vice versa
- This is a tradeoff that all analysts have to make

	Classified Positive	Classified Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

where

TP: the number of correct classifications of the positive examples (**true positive**),

FN: the number of incorrect classifications of positive examples (**false negative**),

FP: the number of incorrect classifications of negative examples (**false positive**), and

TN: the number of correct classifications of negative examples (**true negative**).

Precision and **recall** measures

$$p = \frac{TP}{TP + FP}. \qquad r = \frac{TP}{TP + FN}.$$

- Precision *p* is the number of correctly classified positive examples divided by the total number of examples that are classified as positive.
- Recall *r* is the number of correctly classified positive examples divided by the total number of actual positive examples in the test set.

F_1-value (also called F_1-score)

- It is hard to compare two classifiers using two measures. F_1 score combines precision and recall into one measure

$$F_1 = \frac{2pr}{p + r}$$

- The harmonic mean of two numbers tends to be closer to the smaller of the two.
- For F_1-value to be large, both *p* and *r* much be large.
- Classifiers with high F values are better

How good is this breath test?

	Yes	No
Yes	5	15
No	20	60

Precision	Recall	F-measure
5/25 = 0.2	5/20 = 0.25	0.22
15/35 = 0.45	15/20 = 0.75	0.7

Limitations

- Focuses on one class only
- Doesn’t take true negatives into account
- Biased by majority class judgments
- Assumes ground ‘truth’ is ground truth

Excellent review of F score limitations

Fairness vs. accuracy

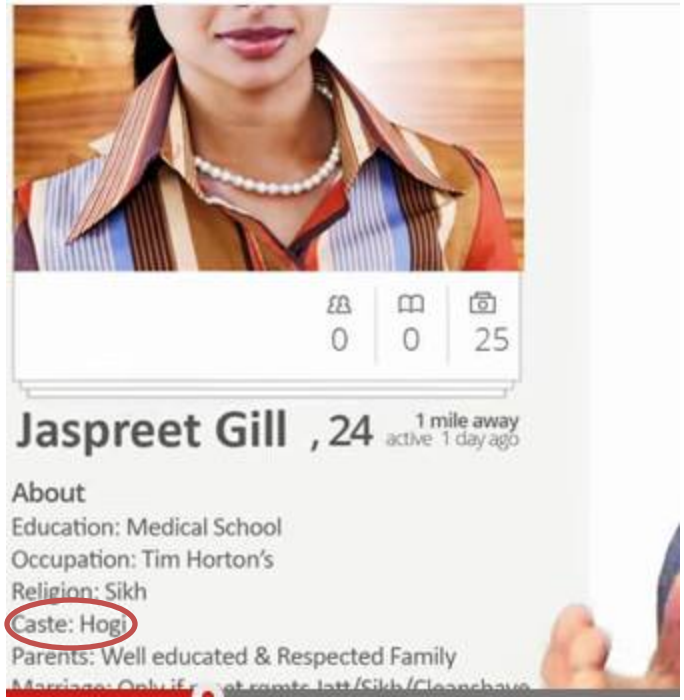
EVALUATIONS IN THE REAL WORLD

Case Example: Optimizing Tinder suggestions

- Straightforward ML problem
- Find suggestions that maximize
  - % of right swipes
  - % of matches

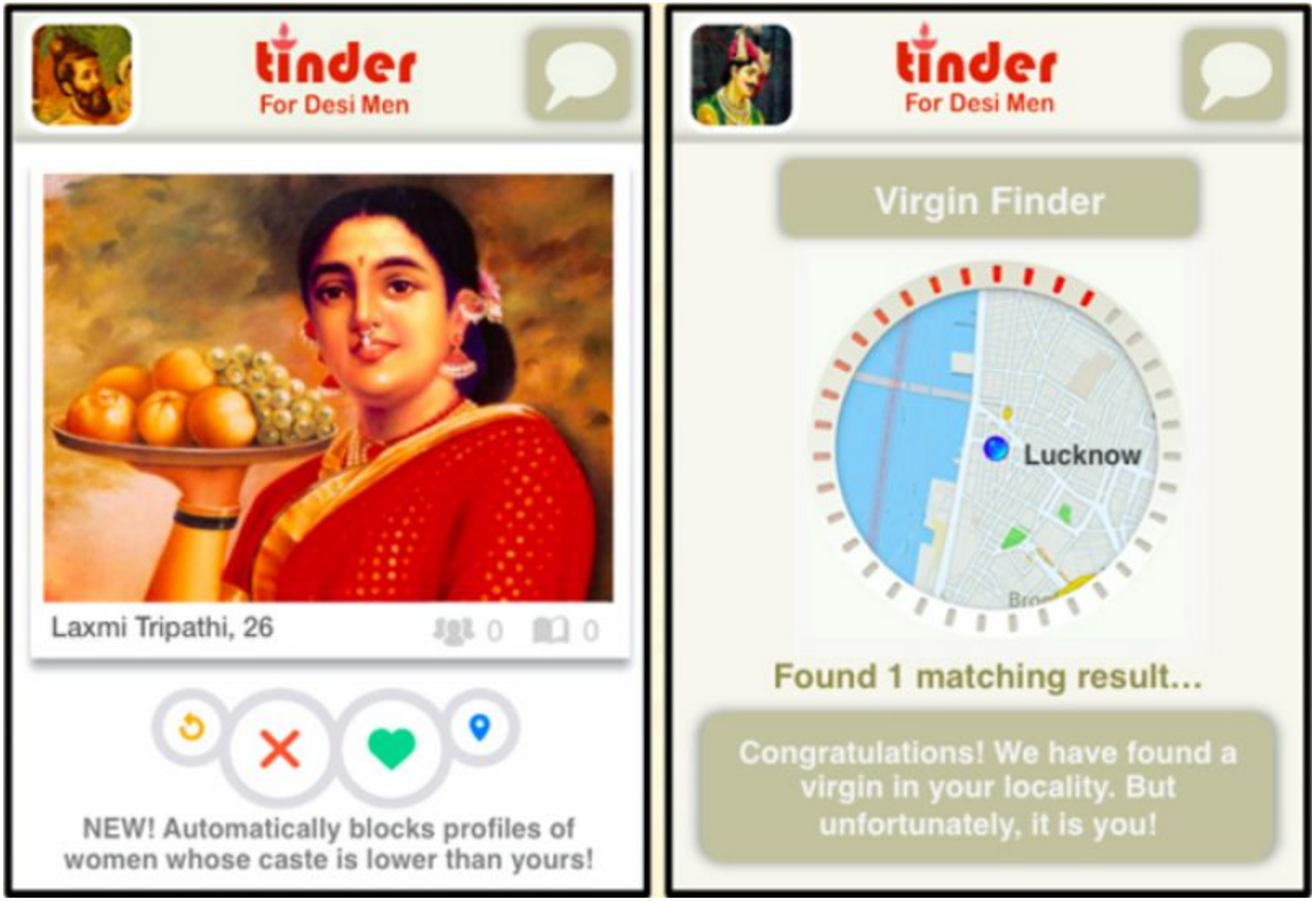
- Simple clean loss function
- Lots of training data

Do you really want to maximize this loss function?



- Algorithms trained on historical data will learn societal biases and amplify them
  - Filter bubbles (link)
- Do you want to be the person helping to amplify these biases?
  - How does one decide? (link)

What if the loss function is a loser?



Low hanging fruit: avoiding discrimination

- Discrimination: Difference in treatment on the basis of a politically problematic trait
- Solution strategy
  - Identify a set of protected attributes
  - Ensure that your model does not treat subgroups that possess those attributes any differently than subgroups that don't

Avoiding explicit discrimination

- Find the use of a protected feature in your model
  - Feature: Caste = ‘Kayastha’
- Remove it
  - ~~Feature: Caste = ‘Kayastha’~~
- Post to LinkedIn about your victory in the cause of social justice

Implicit discrimination

- Frequently, protected attributes are correlated with other attributes in *a priori* unpredictable ways
- For instance, caste can be predicted from surnames, education status etc.
- How do we deal with this?
- This is an active area of research within ML - fairness

Being algorithmically fair

- Group fairness
  - P(favorable outcome| protected group): P(favorable outcome| everyone) ~ 1:1
- Very hard to achieve in practice
- Can we instead prove that we are not being unfair?

Characterizing classifier fairness

- Let's say a classifier has a decision function $d(x, \theta)$ and a vector of sensitive attributes z
- For a fair classifier
  - $\text{Cov}(d(x, \theta), z)$ should approach 0
- Could try to train a classifier with this covariance condition as an explicit constraint

Fair by design classification

- We know that

$$\mathrm{Cov}(d_\theta(x), z) = \mathbb{E}((z - \bar{z}), d_\theta(x))$$

- So we learn to minimize L(θ) such that

$$\left| \frac{1}{N} \sum_{i=1}^{N} (z_i - \bar{z}) d_\theta(x_i) \right| < \mathbf{c}$$

Zafar et al (2015)

Limitations

- Requires access to the set of protected traits
- No way of verifying that the actual outcomes for any one person are not biased
  - Group fairness is not identical to individual fairness
- Still susceptible to drunkards' fallacy
  - Analysis restricted to what is observable
- You can still feel icky about what your system is trying to achieve

Lots of opportunity

- Smart people are increasingly concerned about the damage predictive algorithms are doing to society
- Go look on the web
- Many possible ways to contribute