SOSC 4300/5500: Prediction and Survey

Han Zhang

Sep 22, 2020

Outline

Logistics

- Move to mixed-mode teaching next week
  - LSK 1033 for lecture
  - LSK 1011 for tutorial
- How many people will appear in classroom?
- Are everyone able to find a group?

Feedback

- Some questions raised in the poll from last week:
- Too many discussions?
  - As I mentioned in the class, there will be more knowledge from the third week
  - But, knowledge is something you can google/read by yourself
  - In-class discussions teach you how to ask the right question
- Reading is too difficult?
  - Be sure to read it before the class; I will talk about key points in lectures
  - And this is a skill you will need to learn: quickly understanding the key ideas from

Feedback

- Not very practical?
  - More to come next:)
- Too simple?
  - These contents can be taught in several CS/data science courses; if you want more math/stat things, come to me and I can suggest you some books/CS courses you can read by yourself
  - Practice is always the best way to learn;

How to compare predictions?

- "Soviet Union will collapse one day"
  - There is only one prediction; our prediction is ultimately either right or wrong
  - It's nearly always to be true (because there is no time constraint!)
- In computational social sciences/modern data sciences, usually the prediction goals are precise and falsifiable (Hofman, Sharman and Duncan, 2017)
- One way to make falsifiable is to add scope conditions
- E.g., we make 10 predictions, adding constraint of time:
  - Will Societ Union collapse in 1980? Yes or No?
  - ...
  - Will Societ Union collapse in 1989? Yes or No?
- With a falsifiable prediction, we can calculate prediction error/accuracy
- And then we can precisely compare two predictions

Prediction difficulty vary by prediction goals

- Jake M. Hofman, Amit Sharma, and Duncan J. Watts, *Prediction and explanation in social systems*, Science **355** (2017), no. 6324, 486–488
- Given 852 million tweets, here is a list of things you can predict
- Predict whether each tweet will have more than 5 retweets: easier
- Predict the exact number of retweets for each tweet: harder
- Predict the size of cascades (number of retweets, retweets of retweets,..): hardest

Prediction evaluations for continuous outcomes

- It's common to use $\hat{Y}$ as the predicted value of Y
- For continuous outcomes:
- R^2: what you get from regression models; focusing on variances predictable from your X
    - The larger the R^2, the better the model
- *MSE* (mean squared error): $\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$
    - The smaller the MSE, the better the model
    - Sometimes we also use RMSE $= \sqrt{MSE}$
- *MAE* (mean absolute error): $\sum_{i=1}^{n}|Y_i - \hat{Y}_i|$

Prediction evaluations for categorical outcomes

- For categorical outcomes, evaluation is more complex
- Let us work with the simpliest example of binary outcomes
- Say we has an algorithm predicting COVID infection (positive $= 1$ vs. negative $= 0$)
- We found that 99% of our predictions are correct. Yeah!
- But wait, is that good enough?

Prediction evaluations for categorical outcomes

- In fact, for any classification task, one of the simpliest baseline is to predict every data point as belonging to the majority class
- Here, we know most people are not affected
- So the trivial prediction here just predict that every one is negative (0)
- What's the accuracy for this trivial prediction?
- Let us assume that there are 10,000 students/employees at HKUST, and there are 10 infected cases
- So the error rate is 10/10000, and accuracy = 1 - 10/10000 = 99.9%
- If class is imbalanced, it is very easy to achieve a high accuracy by predicting the majority class all the time
  - But it's not useful at all! Not falsifiable

Prediction evaluations for categorical outcomes

		Actual	
		1/positive	0/negative
Prediction	1/positive	True Positive (TP)	False Positive (FP)
	0/ negative	False Negative (FN)	True Negative (TN)

- It's better to use confusion matrix
- accuracy $= \frac{TP+TN}{TP+TN+FP+FN}$
- precision $= \frac{TP}{TP+FP}$
  - Interpretation: what proportion of predicted positives are actual positive?
- recall $= \frac{TP}{TP+FN}$
  - interpretation: what proportion true positives are identified by predictions?

Case 1: high precision/ low recall

		Actual	
		1/positive	0/negative
Prediction	1/positive	True Positive (n = 5)	False Positive (n =0)
	0/negative	False Negative (n = 5)	True Negative (n = 9990)

- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
    - $\frac{9+9986}{10000} = 99.95\%$
- precision $= \frac{TP}{TP+FP}$
    - $\frac{5}{5+0} = 100\%$
- recall $= \frac{TP}{TP+FN}$
    - $\frac{5}{5+5} = 50\%$
- So every predicted infected case is indeed infected
- But we missed 50% of actual infected cases

Case 2: high recall/low precision

		Actual	
		1/positive	0/negative
Prediction	1/positive	True Positive (n = 9)	False Positive (n = 4)
	0/negative	False Negative (n = 1)	True Negative (n = 9986)

- We lower the threshold to be considered as infection case
- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
  - $\frac{9+9986}{10000} = 99.95\%$; the same
- precision $= \frac{TP}{TP+FP}$
  - $\frac{9}{9+4} = 69.23\%$
- recall $= \frac{TP}{TP+FN}$
  - $\frac{9}{9+1} = 90\%$
- Our prediction captures 90% of actual infected cases
- But less than 70% predicted cases are actually infected; false

Trivial prediction: Majority Class

		Actual	
		1/positive	0/negative
Prediction	1/positive	True Positive (n = 0)	False Positive (n = 0)
	0/negative	False Negative (n = 10)	True Negative (n = 9900)

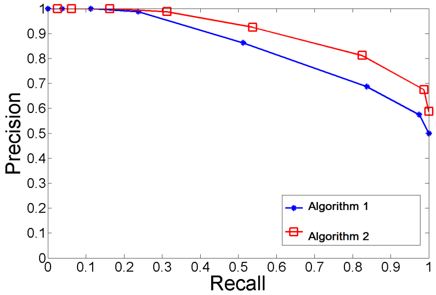
- Predict the majority class (no one is affected)
- Accuracy: $\frac{9+9986}{10000}$ =99.9%; slightly worse
- Better measures should tell us that this is a trivial prediction
- Precision: not defined
- Recall: $\frac{9}{9+1} = 0\%$
- Even though accuracy is high, precision/recall is not satisfactory

Precision-recall trade-off

- In evaluting perdiction performances for categorical outcome, do not use accuracy
- Instead, use precision and recall
- Depending on tasks, we may emphasize one or the other
- [in class activities]: can you think of examples we should focus one or the other?
- Ideally, we want both precision and recall to be high
- In reality, it's often that one comes at the cost of another
- F-1 score: balancing precision and recall
  - $2 * \frac{precision * recall}{precision + recall}$

Precision-recall curve

- We can find a trade-off by using precision-recall curve
- If predicted probability of $Y = 1$ is larger than a threshold ϕ, $\hat{Y} = 1$
  - otherwise $\hat{Y} = 0$
- Large threshold ϕ -> high precision
- Small threshold ϕ -> high recall
- Algorithm 2 is better than 1



ROC curve

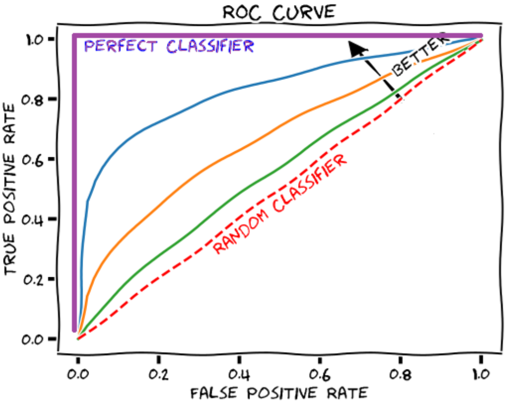
- Another common measure is called ROC curve

Actual

Prediction		1/positive	0/negative
	1/positive	True Positive (TP)	False Positive (FP)
	0/ negative	False Negative (FN)	True Negative (TN)

- True positive rate (i.e., recall): $\frac{TP}{TP+FN}$
- False positive rate: $\frac{FP}{FP+TN}$

ROC curve



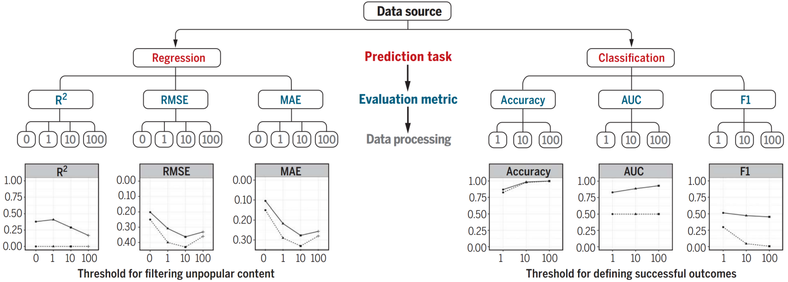
- AUC: area under the curve
- Bigger AUC -> better prediction performance

ROC vs Precision/Recall

- Use ROC curve, if you care both positive and negatives
- Use Precision/recall curve, if you care positive class more than negative class

Summary of evaluation characteristics

- Jake M. Hofman, Amit Sharma, and Duncan J. Watts, *Prediction and explanation in social systems*, Science **355** (2017), no. 6324, 486–488



Various regression models you may have learned

- Linear regression: for continuous outcome Y
  - $Y = \beta X$
- Logistic regression: for binary outcome Y
  - $Y = logit^{-1} \beta X$
- Multinomial/ordered logistic regression: categorical/ordinal outcome Y

Lasso and Ridge

- When data dimension is high (e.g., $K > N$), usually most variables are not relevant
- We need variable selection
  - LASSO regression: force the coefficient of some variables to be 0
    - Controlled by a parameter λ_1; bigger λ_1 forces more coefficients to be 0
  - Ridge regression: force the coefficient of some variables to be very small
    - Controlled by a parameter λ_2; bigger $lambda_2$ forces more coefficients to be small

LASSO and Ridge: math

- We have p variables
- Linear regression:

$$\hat{\beta}_{OLS} = argmin_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^{(1)}$$

- Lasso estimator (Tibshirani, 1996, Least Absolute Shrinkage and Selection Operator):

$$\hat{\beta}_{LASSO} = argmin_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{2}$$

- Ridge estimator (Hoerl and Kennard, 1970; Turgenev, 1943):

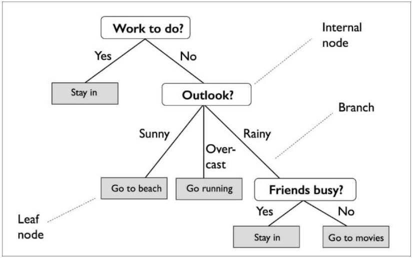
$$\hat{\beta}_{Ridge} = argmin_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{3}$$

Elastic Net

- Combine LASSO and Ridge
- With weights λ_1 for LASSO and λ_2 for Ridge

Decision Tree Example

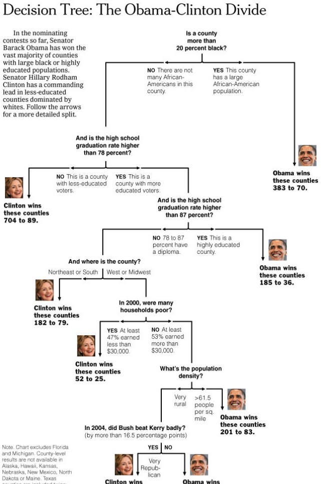
- Intuitively, decision tree visualizes one's sequential decisions process (Y), based on some predictors (variables)



- Decisions (outcomes) Y are located at leaves
- The rightmost branch has a triple interaction

Decision Tree Example

https://archive.nytimes.com/www.nytimes.com/imagepages/2008/04/16/us/20080416_OBAMA_GRAPHIC.html?emc=polb1&nl=pol



Estimating a tree with software

- Modern statistical software can help you to fit a tree automatically
  - R users: `randomForest` package
  - Python users: `sklearn` package; use `RandomForestClassifier` or `RandomForestRegressor`

From One Tree to Many Trees

- Bagging tree (or ensemble of trees): averaging the predictions of many trees
    1. From the original training data, draw a sample with replacement of equal size
    2. Fit a tree for each sample
    3. Repeate 1 and 2 for some times
- Take the mean of estimates of each tree to produce a single estimate for each test data point

Random Forest

- Random Forests further extend the idea of bagging
- The key innovation of random forests:
- For each sample from the original training data, randomly select m variables (not using all p variables), and grow a tree;
  - A common choice: $m = \sqrt{p}$
- In other words, we just force $p - m$ predictors to be non-relevant each time
- Why? High-dimensional data!

Tree, Forests and GLM

- Random forests are usually among one of the best predictors you can get; typically better than regression models
- But ultimately, you should choose based on performances on out-of-sample tests!

Prediction based on outcome is not useful

- We have 10 observed data points (X, Y)
- An easiest way to make perfect prediction:
- For every X, we predict its outcome to be Y
- It's error will be 0, but there is no way to make predictions if we have a new X
- Why? We are predicting based on outcomes
    - It's like cheating
    - It may fits the observed data perfectly
    - But does not generalize to unseen data

Train/test split

- It's important to do honest prediction, by splitting your data into two parts
- One part is training data
- The other is (out-of-sample) test data
- Train your model only with training data
- And evaluate your model with test data

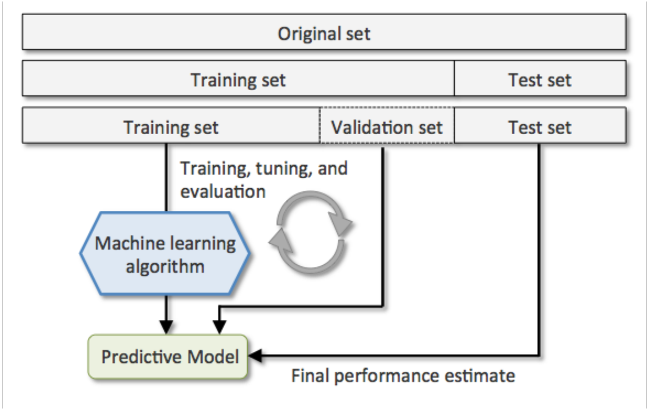
Google Flue Trends Example

- Training data: CDC counts and search queries from 2003 to 2007
  - Training Procedure:
    1. Select a model: e.g., linear regression, CDC counts $\sim \beta \times$ (Google search queries)
    2. Training (fitting) model: obtain the value of regression coefficient β
- Test data: CDC counts and search queries in 2008
  - Evaluating procedures:
    1. calculate predicted CDC counts, based on β and search queries in 2018
    2. compare actual CDC counts in 2008, and predicted CDC counts from the above

Train/validation/test split

- If we do not have test data, how can we internally compare our algorithms?
- Say LASSO has different choices of λ
- Further split your training data into:
  - Training data
  - Validation data
- And internally, train your algorithms on the new training data
- And evaluate on the validation data

Train/validation/test split



Three eras of survey research

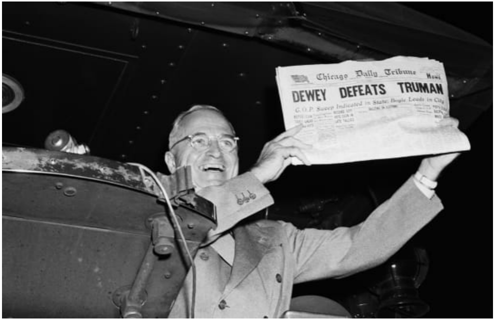
- Matthew Salganik, *Bit by Bit: Social Research in the Digital Age*, Princeton University Press, 2019
- Chapter 3, Table 3.1

Table 3.1: Three Eras of Survey Research Based on Groves (2011)

	Sampling	**Interviewing**	**Data environment**
First era	Area probability sampling	Face-to-face	Stand-alone surveys
Second era	Random-digit dialing (RDD) probability sampling	Telephone	Stand-alone surveys
Third era	Non-probability sampling	Computer-administered	Surveys linked to big data sources

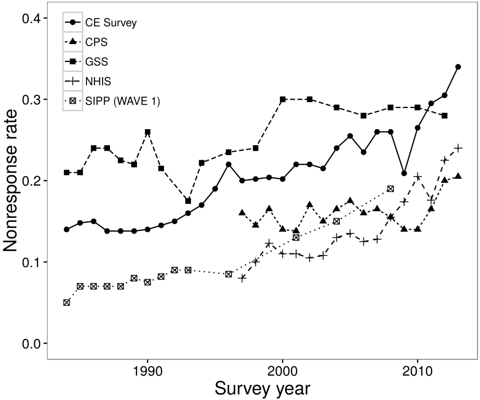
Traditional Surveys: probability sampling

- Why social scientists trust probability sampling more than non-probability sampling?
- Probability sampling on a smaller sample outperforms non-probability sampling on a much larger sample



Problems of traditional surveys

- Rising non-response rate
- Matthew Salganik, *Bit by Bit: Social Research in the Digital Age*, Princeton University Press, 2019
- Chapter 3, Figure 3.6



Problems of traditional surveys

- Trade-off between cost and heterogeneity
- Nicholas Beauchamp, *Predicting and Interpolating State-Level Polls Using Twitter Textual Data*, American Journal of Political Science **61** (2017), no. 2, 490–503
  - some states are poorly polled
  - some days, and sub-state regions, are not polled

Using social media texts to assist election polls

- Nicholas Beauchamp, *Predicting and Interpolating State-Level Polls Using Twitter Textual Data*, American Journal of Political Science **61** (2017), no. 2, 490–503
- Argument: there are many work (especially by computer scientists) stating that social media texts can be used to predict election polls
- But policy researchers still heavily rely on polls
- Can Twitter texts ben used to predict vote share for Obama in 2012?

Cleaning Text Data

- Raw data: 40M tweets between Sep 1, 2012 to Nov 4, 2012 (the election day)
- Each tweet contain at least one political words:

  obama, romney, pelosi, reid, biden, mc- connell, cantor, boehner, liberal, liberals, conservative, conservatives, republican, republicans, democrat, democrats, democratic, politics, political, president, election, voter, voters, poll, polls, mayor, governor, congress, congressional, representatives, senate, senator, rep., sen., (D),

- And each tweet was geolocated using keywords (e.g., they contain location words)
- Resulted in 850GB raw data

Turning Text into Variables

- Beauchamp further reduced the data dimension
- By selecting 10,000 most common words
- And calculate the word percentage, w_{kjt}, for word k at state j at day t
  - Number of tweets containing word k for state j at day t
  - Divided by number of total tweets for state j at day t
- End up with 500 MB data; 50 states $\times$ 67 days $\times$ 10,000 variables
- Turning text into variables is the key to most machine learning using text data;
  - More on this in next two weeks

Selecting training and test data

- Training data: for each day t, training data are
  - 3 previous weeks's vote share for Obama based on polling
  - And/or day t's tweets
- Test data:
  - vote share for Obama on day t
  - across 42 days before the election and in 24 states
    - Other states have a few polls; shortcoming of polls if you want to study some detailed patterns

Selecting model

- 9 different models for training
- Simpler regression based models: (M1 - M5)
    - Fixed effects: each state has its' own intercept β_j
    - Time trends: capture time changes (if there is any)
    - Words: each words has its own coefficient
        - but only maintain the coefficient if its p value < 0.001

$$p_{jt} = \beta_j + \tau t + \beta_k w_{kjt} + \epsilon_{kjt}, \quad \text{for } k \text{ in } [1 \ldots 10,000], \tag{1}$$

Selecting models

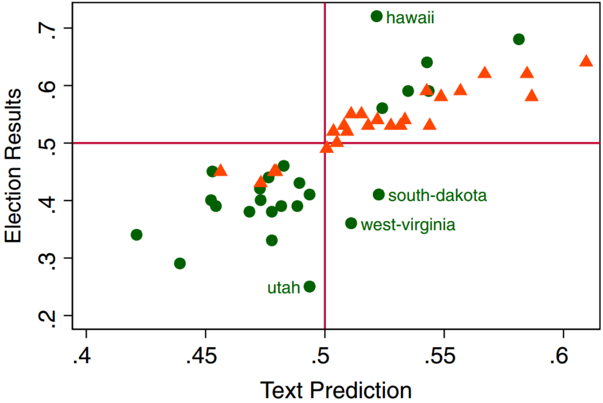
Table 1 Accuracy in Matching Out-of-Sample Text-Predicted Polls to True Polls

								Elastic Net[c]	
	M1	**M2**	**M3**	**M4**	**M5**	**Random Forest**	**SVM**	**$\lambda_1 =$ 0.001**	**$\lambda_1 =$ 0.1**
Twitter text	×		×		×	×	×	×	×
State fixed effects		×	×	×	×	×	×	×	×
Time trend				×	×	×	×	×	×

- The elastic net reduces to LASSO regression since they set $\lambda_2 = 0$
- Random forests, SVM, and elastic net are generally regarded as better than simpler regression models

Prediction Performances: visualization of predictions from M1

- Triangles: states with better polls
- Circles: states with worse polls
- Is this good enough?



Selecting error evaluation criteria

- RMSE
- R^2
    - pooled: variance explained across all cases
    - within: variance explained within states
- And visualization! Simple but powerful

Prediction Performances: quantitative measures

Table 1 Accuracy in Matching Out-of-Sample Text-Predicted Polls to True Polls

								Elastic Net[c]	
	M1	**M2**	**M3**	**M4**	**M5**	**Random Forest**	**SVM**	**$\lambda_1 =$ 0.001**	**$\lambda_1 =$ 0.1**
Twitter text	×		×		×	×	×	×	×
State fixed effects		×	×	×	×	×	×	×	×
Time trend				×	×	×	×	×	×
MAE (smoothed)[a]	1.91	0.60	0.53	0.54	**0.51**	1.53	3.53	0.88	3.76
MAE (real)[a]	2.16	1.38	1.32	1.30	**1.27**	1.81	2.76	1.53	3.21
R^2 Pooled[b]	0.77	0.98	0.98	0.98	**0.98**	0.90	0.19	0.95	0.01
R^2 Within[b]	0.03	0.19	0.36	0.37	**0.40**	0.09	0.07	0.08	0.22

Findings

- Simply using Twitter texts (M1) are worse than simpler regression models (M2, M4)
- Best model (M5) combines Twitter texts and considers the cross-section times-series nature of the data
- Simply taking some machine learning models may not be the best
- But ultimately, draw your conclusions based on prediction evaluation metrics, based on out-of-sample algorithms

Summary

- To make good predictions:
  - Try different algorithms
  - Be honest and don't cheat; use train/test split
  - Choose an appropriate evaluation metric to allow comparison between models
- These steps should be carried out for any prediction tasks