Lecture of Banking Analytics – lecture 5

Zhe Rao

- As stressed, the best to improve your model is to improve your data
  - Data cleaning is all about getting your data to be what they need to be

Data Understanding

- At this stage, go and look everywhere for data that can relate to your problem
  - First source of data is data from your own operational databases
    - This is the core sources for you to take data out
    - E.g. behavioral credit scoring variables, application scorecard variables
  - Then, look outside of your organization, what data is out there that is relevant for you
    - E.g. macroeconomic variables, bureau data
    - Main issue: we need to balance the cost of obtaining and maintaining those variables with how much information those variables bring
      - E.g. do a survey on our customers every month. This is expensive
      - E.g. updating the depreciation on collateral value is very expensive
    - But the regulators are asking us to do that, we have to have processes to update the data
    - So we will spend a lot of time and money to keep our data updated
    - We need to run the numbers ourselves to see if it's worth the effort or not

- Data understanding is all about asking
  - What data should I collect?
  - What's my budget?
  - What's my goal?

- What data to include and what level of sophistication to have is a fairly complex problem that requires you to understand your business situation today
  - Even if you're going to get the data off your database, you need to understand at what speed with what updates with what level of significance
    - E.g. are you going to get real time streaming data, and using the data up to the second?
    - Then you need to create a pipeline for each variable
    - Pipeline is a piece of software that retrieve data from my databases to my model (like a pipe to liquid), it takes the data and process it so you can implement it and deploy it in a model
    - Someone needs to create it and maintain it
  - Important note: every variables in the model needs to come with a pipeline to actually get the data

to the system

- ✧ Good news: once automatize it, maintenance tends to be relatively minor

Sampling

- There is a lot of sampling between application and behavioral scoring
  - ➢ We will have primary data sources (your own data)
    - ✧ Typical primary data are going to be staff like the population, everything in my databases
    - ✧ We have a major source of data called loan application
      - ▫ When someone wants a loan or opens an account, you want to give you data whereas people don't usually want to give you data
      - ▫ So take this opportunities to get the data that you need
      - ▫ This causes that many segments are not effectively covered by evaluating risks with data: younger people don't have much access to financial services
    - ✧ Trade-off: if your questionnaire is too long for your customers, they might just go to next door to another bank
  - ➢ And secondary data sources
    - ✧ Traditionally, the source of secondary data will be credit bureaus
    - ✧ Data collected by a third party, and we would buy the data from the third party
    - ✧ We are also going to use a lot of macroeconomic variables that come from the government, we may complement those with some geodemographic data
    - ✧ There are also many commercial services that you can download that help you profiling your customers

- After you have all of that data, the second thing that you need to do is the sample
  - ➢ In general, you should not remove cases from your dataset
  - ➢ Sampling is only done if you will be representing people incorrectly
    - ✧ E.g. older loans and longer loans are overrepresented
  - ➢ The recommendation by Basel is that you use weighting, not sampling
    - ✧ That means, in the error function, you are giving a lower weight to cases that are overrepresented in the dataset
    - ✧ Details in Logistic regression
  - ➢ Sampling should be done with necessity
    - ✧ E.g. database is too large, and you cannot handle it (usually doesn't happen)
  - ➢ When sampling, you need to think how you are still creating the representative sampling, one thing that is very important is fairness. Your model needs to fairly represent your population
    - ✧ If you don't have much cases in this community, maybe incorporate some judgemental factors into it to balance out the unrepresentativeness of the community (justification needs to be approved by the regulator)
    - ✧ When you have underrepresented communities in your sample, and you need to use this stage to create a dataset that's representative of the population that you want to measure
  - ➢ Example, a company used to lend to only >25 years old customers now need to lend to 18-24 years

old customers too.

    - So either they buy the data from bureau or wait for an year to collect the data (response variables).

- Methods of sampling
  - Random sampling
    - When want to reduce the data systematically
  - Systematic sampling
    - When want to choose segment
    - What we use to increase the weight of certain group
  - Quota sample
    - When want to make sure that everyone is represented in a certain percentage

- In credit scoring, you usually sample in terms of years, we will have more good cases than bad cases, but we are interested on the bad cases because we are interested in predicting the defaulters.
  - As time goes by, the defaulters will be random, the bad rate will stabilize, what's left is random error, one can't be reduced

- There is one more problem which is the rejection inference
  - You only have correct data for the cases that you accepted
  - The ones that are rejected, we don't have data for them because we rejected them
  - Suggestion: we should not ignore the data (or data will be biased), and we should not try to guess the data use mathematical models, we should be buying those data
    - If they did not default elsewhere, we can assume that they are good
    - If they did, we can assume that they are bad
    - If they didn't have a credit file, they might be bad because nobody else is willing to lend to them
  - We should always measure the changing bias
    - Build a model only with accepted customers
    - Another model with accepted and inferred
    - Compare what changed

Data Preprocessing

- Motivation: why do we do data preprocessing?
  - The data is wrong, inconsistent, noisy
    - No company has only clean validated data because this is extremely expensive
    - Financial institutions spent a lot of time trying to understand how to make your data less dirty
      - E.g. never make humans type stuff, try to automatize every single data collection process
      - But sometimes, this is inevitable
  - So we need to clean the data, what we want is eventually the data with no inconsistencies
    - Inconsistencies can be very tricky

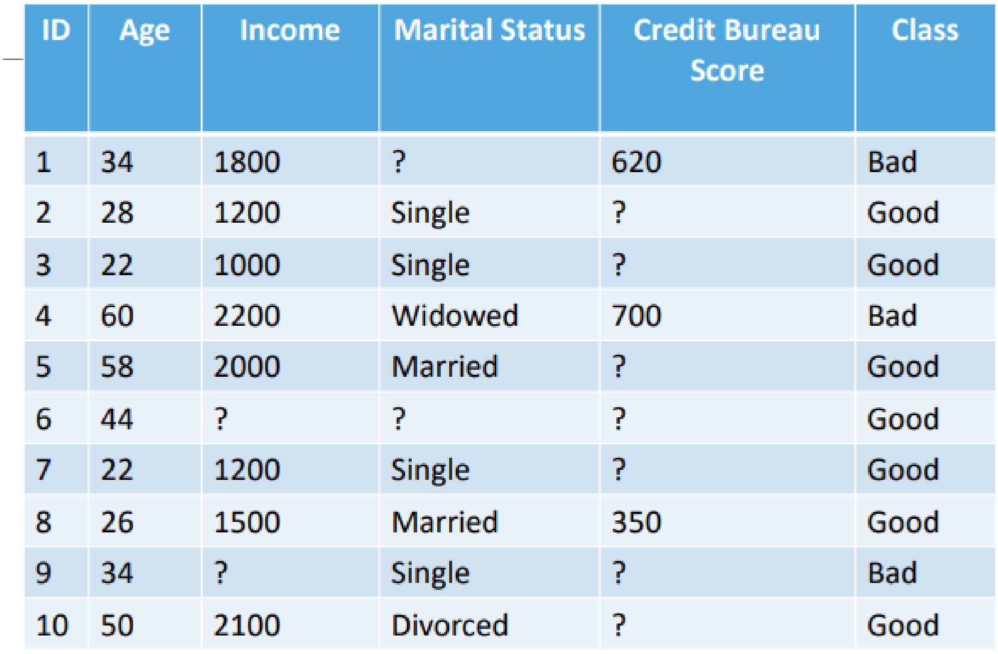
        - E.g. small companies in Canada doesn’t have to pay the HST
        - So if a company has HST of 0 it could either mean that company doesn’t need to pay the HST, or they are suffering a loss so that they have to pay the HST on that (HST on the difference between sales and cost of goods sold)
    - Without data cleaning, we will have errors and biases in the model that we wouldn’t know they are there

Missing values and Outliers

- Possible sources of error that we need to fix
    - Missing values
        - It is a manual endeavor that we need to do by hand by studying the variables and understanding what the reasons behind these missing values are
        - Reasons
            - Not applicable, data do not exist (e.g. default data not available for non-defaulters)
            - Not disclosed (e.g. income)
            - Errors (e.g. typos)
        - Strategy 1: Keep
            - Turn the null values into a category because they are a valid category, and they mean something
            - Solution 1: write null in the category
            - Solution 2: have a different value and add a dummy variable next to it indicating whether there is a null value or not
        - Strategy 2: Delete
            - In case there are too many null variable because those information are never captured
            - Or that information is not available for most people
            - Or the case has too many null values
            - Summary: only remove when you know there is no information there or the information there is not useful, if you know the information is important even when it is missing a lot, replace that for a dummy variable
        - Strategy 3: Replace
            - After you have properly executed strategy 1&2
            - After you have fixed all the inconsistencies
            - Then you can replace the null values with mode (if it is discrete data) or median (if it is continuous data)
            - It should never be above 1-5%
            - Replace too much will distort your data
    - Outliers
        - Statistical definition (plus or minus 3 sd) and complicated imputation would not work
            - Our data is not normally distributed, so the previous definition doesn’t apply
            - And it is against the regulation
        - 1. For us, data is outlier when they cannot correctly represent the behavior of people. We cannot just remove them

        - Because if those people are removed, you need to set up a judgement-based criteria for everyone in that segment
    - Common solution for banks: they have two models, a prime model and subprime model
        - Subprime is for everyone, and prime model is for high earners or for something that's in the corner distributions
        - And prime model has the reduced data only represent those people
    - 2. Another type of outliers: unrealistic data (a loan applicant with 5-year age)
        - They should be treated as missing values
        - Since random, could place them
    - Detection
        - Box plots or violin plots
        - Z-score, usually violate model assumption to detecting outliers; but could be useful if the model needs to be normalized: Neural networks, Logistical regression trained over Lasso.

Example of handling null values



ID	Age	Income	Marital Status	Credit Bureau Score	Class
1	34	1800	?	620	Bad
2	28	1200	Single	?	Good
3	22	1000	Single	?	Good
4	60	2200	Widowed	700	Bad
5	58	2000	Married	?	Good
6	44	?	?	?	Good
7	22	1200	Single	?	Good
8	26	1500	Married	350	Good
9	34	?	Single	?	Bad
10	50	2100	Divorced	?	Good

- First thing: which null values represent a category (keep)
    - Credit Bureau Score
    - Solution 1: replace all the null values with zero
    - Solution 2: replace all the values with a dummy variable
        - Because given this is so unbalanced (too many null values), maybe the ones with a number will not actually have any information
- Second: check cases or variables with no useable information and should be deleted (delete)
    - Candidate 6
    - I probably pick up the phone and ask people what's going on with number 6 (likely an error that is not flagged)

        - In banking, you cannot remove records from the database, you can only flag them
    - Third: check the remaining null values and replace them if feasible
        - We only have 2 null values left, so we replace them by median or mode

What's next?

- We may need to
    - Standardising the data
    - Transform the data
    - Coarse classification and grouping of attributes
    - Recoding categorical variables

Data Transformation

Normalization

- Some data requires normalization
    - We should study the model to determine if the data requires to be normalized
        - Logistic regression when use a saga optimizer
        - Neural networks

Coarse Classification of Attributes

- Social phenomena that describe people's behavior is not continuous, it's discrete
    - E.g. the amount of beer I buy is not increasing with the income I have, if I have more income, I will stop buying beer and buy wine instead
- As such, our modeling of social processes should not be continuous either, and we'll try to capture as much variability as possible in the decisions. The variability in how we actually make decisions at certain points in life
    - We should be very much tuned to how people behave on average
- This is called coarse classification of attributes
    - Take continuous variables and segment them into pieces that make sense in order to attempt to capture this decision making that we make in a discrete way
    - This is helpful to create more robust models because although not everyone will behave this way, we want certain robustness such that small changes in the population do not create huge shifts in our models

- Methods
  - Bad
    - Equal interval binning
      - E.g. Make a cut every five years
      - It does not take into account the behavioral discrete changes
    - Equal frequency binning
      - E.g. Making people into equal groups so that every group will have two or three people
      - It does not guarantee you if different groups will behave in different ways
  - Good
    - Chi-squared analysis
    - Entropy-based discretization
      - E.g. using decision trees

- What we will do is, we will create cuts using information from the variables themselves
  - We will create cuts in terms of the variables, in terms of how the variable behaves using statistical techniques

- We could add dummy variables
  - But it has a terrible bias variance trade-off
    - Gain a lot of variance for very little bias reduction

Weight of Evidence

- If the target variable is binary, and have a fully categorical data set, then you can use the Weight of Variance transformation to keep all the information in just one variable
  - This turns the categorical information into a number
  - That number comes from information theory
  - Information theory is a branch of mathematics that deals with the fact that information gain is not linear
  - Consider describing a person
    - Height, sex, dress in black, wear glasses
    - At some point, the additional information will be useless, e.g. they use black shoelaces.
  - The amount of information of something is going to be the base two logarithm
    - The piece of information that gives you understanding of something is measured in bits
    - This is meant for computers

- Why is it a good idea to use Weight of Evidence transformation?
  - Everything will be numerical now
  - When apply the transformation, the output variable will be normalized because everything will be expressed in information terms and centered around 0
    - 0 means no information
    - 1 means one unit of information

    - ✧ -1 means one unit of information toward the other category
  - ➢ It avoids creating a lot of dummy variables
    - ✧ It avoids increasing the variance of the model
    - ✧ But at the same time, keep the same amount of information in the data set
  - ➢ The software will generate categories, but it is up to you to decide whether that behavior is reasonable or not

- We have the percentage in class A and percentage in class B
  - ➢ We want to make the comparison in terms of information where we want to know how much information you are getting from one class, versus the other.
  - ➢ This is called the weight of evidence
  - ➢ Weight of evidence is a measure of how different (cases are) between two classes

$$\text{WoE}_{\text{category}} = \ln(\text{p\_good}_{\text{category}} / \text{p\_bad}_{\text{category}}),$$

$$\text{where} \quad \text{p\_good}_{\text{category}} = \text{number of goods}_{\text{category}} / \text{number of goods}_{\text{total}}$$

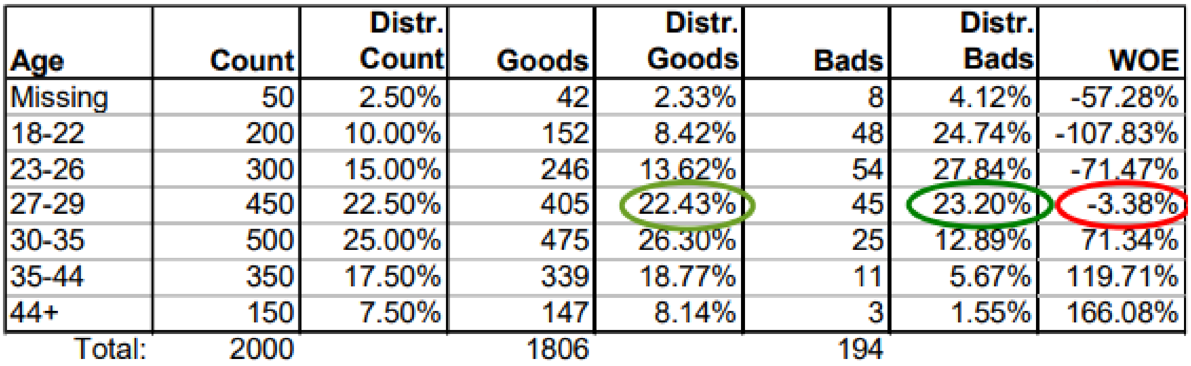
$$\text{p\_bad}_{\text{category}} = \text{number of bads}_{\text{category}} / \text{number of bads}_{\text{total}}$$

$$\text{If p\_good}_{\text{category}} > \text{p\_bad}_{\text{category}} \text{ then WoE}_{\text{category}} > 0$$

$$\text{If p\_good}_{\text{category}} < \text{p\_bad}_{\text{category}} \text{ then WoE}_{\text{category}} < 0$$

  - ➢ This is a quantitative measure of the amount of information that we have available in that category
    - ✧ How much information help us differentiate goods than bads?
  - ➢ If you have more goods than bads, the Weight of Evidence will be greater than 0
- There are two important information here
  - ➢ One is the sign
    - ✧ It points at what category is higher
  - ➢ The other one is the absolute value
    - ✧ It points at how useful the categories are.
  - ➢ They should be taken independently

- Example



Age	Count	Distr. Count	Goods	Distr. Goods	Bads	Distr. Bads	WOE
Missing	50	2.50%	42	2.33%	8	4.12%	-57.28%
18-22	200	10.00%	152	8.42%	48	24.74%	-107.83%
23-26	300	15.00%	246	13.62%	54	27.84%	-71.47%
27-29	450	22.50%	405	22.43%	45	23.20%	-3.38%
30-35	500	25.00%	475	26.30%	25	12.89%	71.34%
35-44	350	17.50%	339	18.77%	11	5.67%	119.71%
44+	150	7.50%	147	8.14%	3	1.55%	166.08%
Total:	2000		1806		194		

  - ➢ Which one would be the most helpful category to predict (gives you the highest amount of information)?
    - ✧ The 44+ category
  - ➢ Note that this does not tell you anything about the quality of the variable itself
    - ✧ The quality of the variable will depend on how many cases are on each one of the categories

        - If there is a lot of people in this very helpful category, then this variable is going to be very useful

- We replace the original variable with the Weight of Evidence variable
    - For every data, it belongs to one category (e.g. 44+), and we take the original data on that attribute (e.g. age = 46) and replace it by the Weight of Evidence of their category

- We have done the null values and outliers cleaning, after Weight of Evidence transformation, everything is numeric centered and standardized
    - Now we have a very clean dataset now and ready to apply models
    - This is usually the best idea when you need a linear model
    - Usually Weight of Evidence goes with logistic regression
    - 90% of applications will be enough with a logistic regression, but the 10% requires more sophisticated and more expensive models

Variable Selection

- The requirements of the model must be understood very well
    - We are dealing with correlation
        - It is going to mess up the model in ways which are relatively hard to identify
    - We are dealing with complexity
        - With 2000 variables, solving the logistic regression problem is going to be very expensive

- Why do we do input selection
    - Curse of dimensionality
        - Running times of algorithms increase polynomially on the number of variables
        - The higher the number of dimensions, the more complicated it is to get good results
    - Interaction and correlation effects
        - Some models can tolerate correlation, some models can deal with it without any issues, some models cannot stand it at all
            - Lasso regression cannot allow correlation, or it will not converge
            - Ridge regression tolerates correlation, but the output will not be consistent independently, the weights are going to be bias.
            - Elastic net will be completely bias but will be predictive
            - Random forest, no problem, but the importance of variables will be affected as it’s going to split between variables
        - Regardless, having correlated variables will change the outputs
        - Note that correlation and causation are not all related

Complexity

- With 10 variables, we can create $2^{10} – 1$ possible models
  - ➢ With large number of variables, it gets extremely computationally expensive and long.
  - ➢ It's a bad idea
  - ➢ Variables that you are sure that they do not help, it's a good idea to get rid of them
    - ✧ However, whenever in doubt, leave the variable in
  - ➢ This is to make the algorithms job easier

Selection Procedure

- Step 1: use some filter that removes variables that you are fairly sure that they do not help
  - ➢ Filter anything that is fully independent of your target variable
  - ➢ Perform a first correlation filter that eliminate variables that are highly correlated
- Step 2: embedded input selection
  - ➢ The regularization process that the algorithms will have
  - ➢ A final correlation filter if you did some transformations that were significant
- Step 3: business filter
  - ➢ Need to decide whether to keep or eliminate variables with marginal gain considering its associated costs

Correlation Filters

- Since the variables are continuous, you may not know how good the cuts are
- Most models are robust to moderate amounts of correlation
  - ➢ Ideally, the variables should be completely uncorrelated
  - ➢ But obtaining that is extremely difficult in real life
  - ➢ It is not direct to get to an uncorrelated data
- We are going to eliminate those correlations that are dangerous, and if we think this is something that's beneficial to the model.
  - ➢ The correlation usually falls into two extremes: 20% vs 80%
    - ✧ We can safely ignore the 20% correlation because it would bias the model to the fifth decimal, and we don't care
    - ✧ 80% correlation can make the standard error go to infinity
- This is a judgement call, what variables do we keep
  - ➢ We keep the variables that are best related to the objective target variable
- Solution 1: calculate through all of this into a PCA and just calculate the rotated axis
  - ➢ The problem is, the manager hate that because they need to collect all variables
  - ➢ If we have two rows of data 90% correlated, having the rotation is increasing our accuracy for like 0.001%.

    - Not operationally desirable

- In general,
    - We will keep variables with absolute correlation within 0.5
    - We will delete variables with absolute correlation above 0.7
        - We will pick the one that is most correlated with the target variable
    - If it is in between,
        - Keep them in the model to see the impact they have
        - Try to use a model that deals with correlation (Lasso, Ridge, Elastic Net)
            - If attributes are ordinal or binary, could use Spearman correlation or Kendall Tau

Procedures that should follow

- If I have a very big dataset
    - Perform correlation filters first (or after null values cleaning)
    - Perform the null value and outlier cleaning
    - Calculate some correlations and drop the ones with extremely high correlation, 90%+
        - This could be done before or after null value treatment
        - If the deleted correlated variables are useful, consider creates a dummy variable for it
        - This gets rid of information that's fully redundant
    - Then perform Weight of Evidence transformation
        - Or before applying the model if WoE is not allowed in the model
    - Perform the correlation analysis again
        - WoE already relates the target variable with the input
        - Two variables that are not correlated before, once put in contact with the target variable, they may have the same behavior, so you get WoE that are highly correlated

Information Value

- Kullback Leibler divergence from information theory is measuring the difference between to distributions
    - How much information do I need to add or remove to turn this distribution into the other distribution?
        - E.g. maybe increase the standard deviation
    - Note that, it is not the same moving from A to B than moving from B to A

- Since WoE compares the distribution of goods versus the distribution of bads, we can use this divergence measure to see how different the distribution of the goods is against the distribution of the bads in that particular variable
    - However, goods to bads is not the same for bads to goods, so which one do we use?
    - We use both

- This sum of Kullback Leibler divergences was named the information value

$$\sum_{all\ categories} p_{goods_{category}} * \ln\left(\frac{p_{goods_{category}}}{p_{bads_{category}}}\right) +$$

$$\sum_{all\ categories} p_{bads_{category}} * \ln\left(\frac{p_{bads_{category}}}{p_{goods_{category}}}\right)$$

$$= \sum_{all\ categories} p_{goods_{category}} * \ln\left(\frac{p_{goods_{category}}}{p_{bads_{category}}}\right) +$$

$$- \sum_{all\ categories} p_{bads_{category}} * \ln\left(\frac{p_{goods_{category}}}{p_{bads_{category}}}\right)$$

$$= \sum_{all\ categories} \left(p_{goods_{category}} - p_{bads_{category}}\right) * WoE_{category}$$

- A rule of thumb
  - < 0.02: almost the same distribution, cannot distinguish between goods from bads, unpredictive
    - Discard
  - 0.02 – 0.1: weak
    - If I have no other alternative, leave those in
    - If I have powerful variables, then discard them
  - 0.1 – 0.3: medium
  - > 0.3: strong
  - > 1: may have leakage