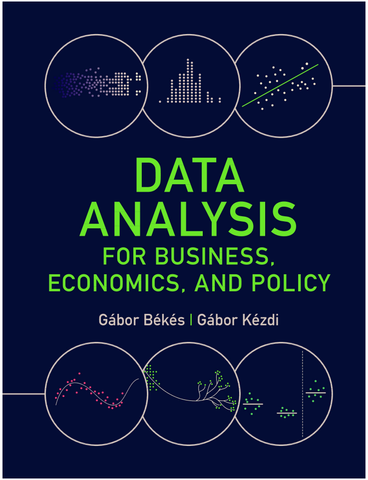
5. Probability Prediction and Classification

Gabor Bekes

Data Analysis 3: Prediction

2021

Slideshow for the Békés-Kézdi Data Analysis textbook



- Cambridge University Press, 2021 April
- Available in paperback, hardcover and e-book
- **gabors-data-analysis.com**
  - Download all data and code `https://gabors-data-analysis.com/data-and-code/`
- This slideshow is for **Chapter 17** - Part 1: Theory

Prediction with qualitative target

- Y is qualitative
  - Whether a debtor defaults on their loan
  - Email is spam or not
  - Game result is win / lose / draw.
- We consider binary (two-class) Y only
  - $Y = 0$ or 1 (yes or no)
  - Class prevalence (p) - frequency of 1.

Prediction with qualitative target

Two different actions

- Predicting probability of $Y = 1$
  - The probability (chance) a debtor will default

- Assigning classes to Y = **classification**
- Need to put target observation in a "class"
  - $\hat{Y}_i = 0$ or $\hat{Y}_i = 1$
- Could be multiple classes, like color
- Today: Y binary

The process

- Predict probability
  - As we have done in DA2/week 5
- Predicted probability between 0 and 1
  - Probability of an event happening
- For each observation we predicted a probability. Often that is it.
- Loss function is Brier score = RMSE
- Sometimes we will go further and classify observations into 0 and 1 = classification

Refresher: Probability Models

- LPM - not this time
- Logit
  - Nonlinear probability models
  - Logit $\Pr[y_i = 1|x_i] = \Lambda \times (\beta_0 + \beta_1 x_i) = \dfrac{\exp(\beta_0 + \beta_1 x_i)}{1 + \exp(\beta_0 + \beta_1 x_i)}$
  - Predicted probability between 0 and 1
- Logit
  - Starts with a linear combination of the explanatory variables
    - Multiplies them with coefficients, just like linear regression
  - And then transforms that into something
    - That is always between 0 and 1
    - And that thing is the predicted probability.

What's New with Binary target?

- Probability predicted not value
- Desire to classify
  - assign 0 or 1
  - based on a probability that comes from a model
  - But how?

- New measures of fit
  - Some based on probabilities
  - Others based on classification

What's New with Binary target?

- Need best fit
- With highest external validity
- Usual worries: overfit
  - Cross-validation helps avoid worst overfit

- Models similar to those used earlier
  - Regression-like models (probability models)
  - Tree-based models (CART, Random Forest)

Probability prediction

We build models to predict probability when:

- aim is to predict probabilities – this is what we do
- aim is to classify (predict 0 or 1) – this is the first step
  - build probability models, select the best one
  - use a loss function to classify

Probability prediction process

- Build models
  - several Logit models by domain knowledge
  - LASSO - Logit LASSO
  - CART/Random Forest (discuss later)
- Pick the best model via cross-validation
  - Loss function is Brier score = RMSE
  - Could be other, not today

Classification process

- Predict probability
- Make into 0/1 predictions - classifications
- We can make errors
  - False negative
  - False positive

Classification Table

	$y_j = 0$ Actual negative	$y_j = 1$ Actual positive	Total
$\hat{y}_j = 0$ Predicted negative	TN (*true negative*)	FN (*false negative*)	TN + FN (*all classified negative*)
$\hat{y}_j = 1$ Predicted positive	FP (*false positive*)	TP (*true positive*)	FP + TP (*all classified positive*)
Total	TN + FP (*all actual negative*)	FN + TP (*all actual positive*)	TN + FN + FP + TP (*N, all observations*)

Classification Table: making errors

	$y_j = 0$ Actual negative	$y_j = 1$ Actual positive	Total
$\hat{y}_j = 0$ Predicted negative	TN (*true negative*)	FN (*false negative*)	TN + FN (*all classified negative*)
$\hat{y}_j = 1$ Predicted positive	FP (*false positive*)	TP (*true positive*)	FP + TP (*all classified positive*)
Total	TN + FP (*all actual negative*)	FN + TP (*all actual positive*)	TN + FN + FP + TP (*N, all observations*)

Classification Table: making errors

	$y_j = 0$ Actual negative	$y_j = 1$ Actual positive	Total
$\hat{y}_j = 0$ Predicted negative	Predict firm stay (*Firm did stay*)	Predict firm stay (*Firm exited*)	TN + FN (*all classified stay*)
$\hat{y}_j = 1$ Predicted positive	Predict firm exit (*Firm stayed*)	Predict firm exit (*Firm did exit*)	FP + TP (*all classified exit*)
Total	TN + FP (*all actual stay*)	FN + TP (*all actual exit*)	TN + FN + FP + TP (*N, all observations*)

Measures of classification

- **Accuracy =(TP+TN)/N**
  - The proportion of rightly guessed observations
  - Hit rate
- **Sensitivity =TP / (TP+FN)**
  - The proportion of true positives among all actual positives
  - Probability of predicted y is 1 conditional on $y = 1$
- **Specificity = TN/(TN+FP)**
  - The proportion of true negatives among all actual negatives
  - Probability predicted y is 0 conditional on $y = 0$

Measures of classification

- The key point is that there is a trade-off between making false positive and false negative errors.
- This is the essential insight in classification
- This can be expressed with specificity and sensitivity.

ROC Curve

- The *ROC curve* is a popular graphic for simultaneously displaying specificity and sensitivity for all possible thresholds.
  - ROC: Receiver operating characteristic curve
  - Name from engineering

- For each threshold, we can compute confusion table –> calculate sensitivity and specificity
- Show in graph - illustrate (non-linear) trade-off

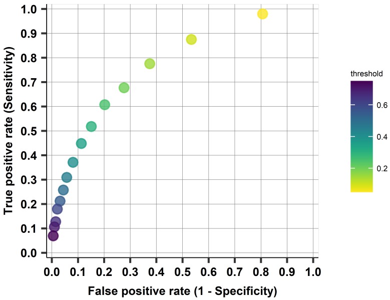
- ROC curve – choosing a threshold value creates a tradeoff between how well a probability prediction leads to correct classification of y = 1 observations versus y = 0 observations.
  - The curve shows this across all possible threshold values.
  - The ROC curve doesn't show the threshold values themselves.

ROC Curve

- The *ROC curve* is a popular graphic for simultaneously displaying specificity and sensitivity for all possible thresholds.
  - ROC: Receiver operating characteristic curve
  - Name from engineering

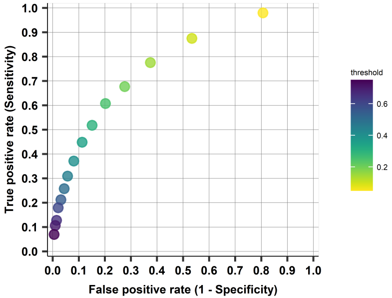
- For each threshold, we can compute confusion table –> calculate sensitivity and specificity
- Show in graph - illustrate (non-linear) trade-off

ROC Curve: a two-dimensional plot



- Horizontal axis: False positive rate (one minus specificity) = the proportion of FP among actual negatives
- Vertical axis: is true positive rate (sensitivity) = proportion of TP among actual positives
- For classifications from a single probabilistic forecast as the threshold is moved from 0 to 1

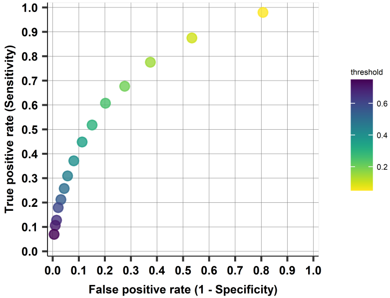
ROC Curve Intuition



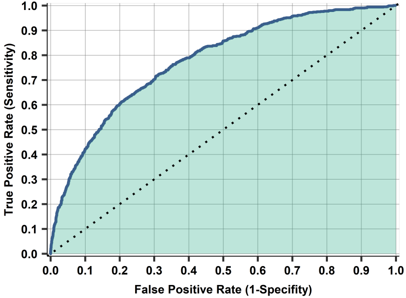
- Neither axis shows the value used for the threshold directly, but both decrease in threshold value.
- ROC curve is for all possible thresholds - many thresholds shown by dots
  - From 0 to 1
- Higher threshold means fewer positives and thus fewer false positives and/or fewer true positives.
- As we lower the threshold, we move to right and up.
- ROC curve – how true positives and false positives increases relative to

ROC Curve Intuition

(a) ROC curve points for various thresholds



(b) Continuous ROC curve



ROC Curve Intuition - the 45 degree line

- Vertical axis: $\Pr[(Correct1|y=1)]$
- Horizontal axis: $\Pr[(False1|y=0)]$
- 45 degree line = if classification totally random with true probability p
- Consider a case with $p = 40\%$ and $y = 1$
- all individuals are classified randomly to 1 or 0 with $p = 40\%$ chance
  - $\Pr[(Correct1|y=1)] = \Pr[(False1|y=0)] = p = 0.4$
  - Why? $\Pr[(Correct1)] = p$ whether observation has $y = 1$ or $y = 0$
- Any threshold may be applied here as classification is not based on any particular threshold
- That's the 45 degree line.

Area Under ROC Curve

- ROC curve: the closer it is to the top left column, the better the prediction.
  - Perfect model: horizontal line at TPR=1
- Area under ROC curve summarizes quality of probabilistic prediction
- For all possible threshold choices
- Area = 0.5 if random classification
- Area > 0.5 if curve mostly over 45 degree line
- AUC = Area Under the ROC Curve
- AUC is a good statistic to compare models
- Defined from a non-threshold dependent model (ROC)
- The larger the better
  - Ranges between 0 and 1.

Model selection Nr.1: Probability models

- Model selection when we have no loss function, based on probability models
- Predict probabilities
  - No actual classification
- Use predicted probability to calculate RMSE
- Pick by smallest RMSE
  - When users rely on probabilities

- Draw up ROC curve and get AUC
- Pick the model with the largest AUC
  - More frequently used in practice
  - Has nice interpretation
  - Less sensitive to class imbalance
- In practice, AUC is more frequently used

Classification

- How we make classification from predicted probability?
- Set a threshold!

- The process of classification
- If probability of event is higher than this threshold–> assign (predict) class 1; and 0 otherwise.

- Who sets the threshold?

Classification: how to select the threshold

- We see there is a trade-off
- How to select threshold?
- Majority voting? (50%)
- Match frequency in data (20%)

Classification: select the threshold with loss function

- Find optimal threshold with loss function.
- A loss function is a dollar (euro) value assigned to false positive and false negative.
  - It is actually the ratio of FN/FP that matters.
- Most often the costs of FP and FN are very different.

How to select the threshold

- Find optimal classification threshold with loss function
- Find threshold with lowest expected loss
- Two key inputs: relative prevalence of FP and loss due to errors

$$E[loss] = \Pr[FN] \times loss(FN) + \Pr[FP] \times loss(FP)$$

- How to find best threshold based on loss? Two options
- Formula
- Algorithm

How to select the threshold: Algorithm

- Algorithm looks over all possible thresholds and picks the best option
- Minimizing expected loss

- Technical note
- search for the optimal classification threshold does not look for the smallest expected loss.
- Instead, they search for the threshold that maximizes the probability cost function or the **cost-sensitive Youden index**
- Max J = Min expected loss (See Appendix 17.U2)

How to select the threshold: Formula

- Formula
  - When dataset is "large"
  - When our model has a "good" fit

$$Threshold_{minE(loss)} = \frac{loss(FP)}{loss(FN) + loss(FP)}$$

- In practice
  - Pro: easy to use, often close enough
  - Con: not the best cutoff, especially for smaller data, and poorer model

Model selection Nr.2: Loss function driven

- Model selection process when we have a loss function
- Directly based on classification
  1. Predict probabilities
  2. Use predicted probabilities and loss function to pick optimal threshold
     - Algo or formula
  3. Use that threshold to calculate expected loss
  4. Pick model with smallest expected loss (in 5-fold CV).

Classification tree

- Classification tree, predict the class (0/1)
- Same: Building trees with recursive binary splitting
- Different: prediction is not the mean of values, but the share of $y = 1$
- Probability<–>Frequency
- Based on threshold
- Different: Loss function

New loss function

- In a classification tree, the measure of fit is **node impurity**.
- Extent to which nodes contain observations with both $y = 0$ and $y = 1$ or only $y = 0$ or $y = 1$.
- A widely used measure is the **Gini index of node impurity**.
- Let's consider a split, for node m, and let $\widehat{p_m}$ represent the share of observations with $y = 1$.

$$Gini = 2\widehat{p_m}(1 - \widehat{p_m})$$

- The index is very small if all observations have either $y = 0$ or all have $y = 1$.
- The closer $\widehat{p_m}$ to 0.5 the larger the value of the index.
- Thus, a small value implies that the node is made up entirely of a single class.
- It turns out so using the Gini index of node impurity or using MSE to find the best fit leads to the same result.
  - See Appendix Ch17.U2

Random forest

- Similar approach to regression trees
- Do classification trees, on bootstrapped datasets, and aggregate them.
- Often perform better than logit models.
  - Similarly to OLS vs Random Forest
- No need for model building
- Better probability prediction
- Slower

- Boosting can also be used for binary y.

Random forest: two options

- Similar approach to regression trees
- Do classification trees, on bootstrapped datasets, and aggregate them Two options:

- Probability forest + threshold search with algorithm
- Classification forest + threshold formula

Random forest: probability forest

Probability forest + threshold search / algo

- Predicted probabilities
- Use them to find threshold or use formula to classify
- Aggregates the probability predictions of each tree by averaging them across all trees.
- The model's predicted probabilities are simply these averages.
- For predicting probabilities – this is the version to use.
- For classification – can be used, too, by simply applying the optimal classification threshold to the predicted probabilities.

Random forest: classification forest

Classification forest + threshold formula

- Carries out the classification at the end of each individual tree + aggregates those classifications –> final classification
- Input formula based threshold as tuning parameter
- For predicting probabilities, this is not a good approach.
- For classification, this is the right model

- For classification, we can use probability or classification forest.
  - Results tend to be very similar
  - We have to find the optimal classification threshold using a loss function.

Random forest : key technical insight

- Two options yield results that are very close
  - Not the same
  - Both are okay to use

Random forest : key technical insight

- Two options yield results that are very close
  - Not the same
  - Both are okay to use

- Do not use default setting of "majority voting"!!!
- Default for classification random forest is $t = 0.5$
- Loss(FN) = loss(FP) - Called "majority voting"
- Seems convincing. But it's misleading!
  - Loss function could be anything!!!

Random Forest summary

- Random Forest works well for prediction when target is binary
- May always use for probability prediction
- Use for classification only with an explicit loss function

Class imbalance

- A potential issue for some dataset - relative frequency of the classes.
- Class imbalance = the event we care about is very rare or very frequent ($\Pr(y = 1)$ or $\Pr(y = 0)$ is very small)
  - Fraud
  - Sport injury
- What is rare?
  - Something like 1%, 0.1%. (10% should be okay.)
  - Depends on size: in larger dataset we can identify rare patterns better.
- Consequence: Hard to find those rare events.

Class imbalance: the consequences

- Methods we use not good at handling it.
- Both for the models to predict probabilities, and for the measures of fit used for model selection.
  - The functional form assumptions behind the logit model tend to matter more, the closer the probabilities are to zero or one.
- Cross-validation can be less effective at avoiding overfitting with very rare or very frequent events if the dataset is not very big.
- Usual measures of fit can be less good at differentiating models.
- Consequence
  - Poor model performance
  - Model fitting and selection setup not ideal

Class imbalance: what to do

- What to do? Two key insights.
- 1: Know when it's happening. Ready for poor performance.
- 2: May need an action: **rebalance** sample to help build better models
- Downsampling – randomly drop observations from frequent class to balance out more
  - Before: $100,000$ observations 1% event rate ($99,000$ $y = 1$, $1,000$ $y = 0$)
  - After $10,000$ observations 10% event rate ($9,000$ $y = 1$, $1,000$ $y = 0$)
- Over-sampling of rare events
- Smart algorithms
  - Synthetic Minority Over-Sampling Technique (SMOTE)
  - Others

Summary

- Decide whether the goal is predicting probabilities or classification.
- The outcome of prediction with a binary target variable is always the predicted probabilities as a function of predictors.
- When our goal is probability prediction, we should find the best model that predicts probabilities by cross-validation + RMSE/AUC.
- When our goal is classification, we should find the best model that has the smallest expected loss.
  - With formula for threshold or search algorithm
- Finding the optimal classification threshold needs a loss function.