



AGH UNIVERSITY OF KRAKOW

Data Science in Practice

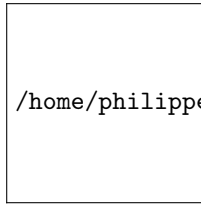
Dr. Philippe J.S. De Brouwer

Honorary Consul of Belgium in Kraków
Guest professor at the UJ, AGH, and UEK
Member of the Board of AGH
Member of the Patronage Council UEK
SVP at HSBC in Kraków

Compiled on September 28, 2026

Table of Contents

1	Ethics in AI and Financial Services	2
2	Bias in data	17
3	Case Study	25



`/home/philippe/Documents/courses/img/mba_tech_logo.jpg`

ETHICAL DILEMMAS WHEN USING MODELS AND DATA

DATA SCIENCE IN PRACTICE

DR. PHILIPPE J.S. DE BROUWER

ALL RIGHTS RESERVED BY THE AUTHOR
prepared for: AGH University of Krakow

SEPTEMBER 28, 2026

Philippe De Brouwer

Academic and Professional Profile Dr. Philippe De Brouwer holds a background in theoretical physics and earned a second Master's degree in Business Engineering while working full time. His academic distinction includes solving the “fallacy of large numbers” puzzle, originally posed by P.A. Samuelson and unresolved for 38 years. In his Ph.D., he challenged the foundational assumptions of Harry Markowitz's Nobel Prize-winning Mean Variance Theory, developing the “Maslowian Portfolio Theory” and fundamentally advancing the field of investment advice.

Career Highlights Early in his career, De Brouwer transitioned from insurance to banking and from IT to asset management. At Fortis (now BNP Paribas), he played a key role in launching one of Belgium's first capital-guaranteed funds and was promoted to director in 2000. In 2002, he joined KBC, where he orchestrated the merger of four companies and became CEO of the consolidated entity in 2005. Under his leadership, the company climbed from 11th to 5th in the market, even amid heightened competition.

Following the financial crisis, De Brouwer led the creation of a new investment management company for KBC in Ireland, which quickly grew to manage around 1000 investment funds and approximately €32 billion in assets. In 2012, he broadened his expertise into financial risk management, focusing on statistics, analytics, data, and numerical methods.

Recent Roles and Publications De Brouwer served as Head of Analytics Development for the Royal Bank of Scotland Group and currently holds the position of SVP at HSBC in Kraków. There, he oversees the Model Review Centre of Excellence and serves as Global Head of Professional Practices for Model Risk Management. He also coordinates collaboration with universities.

He has authored numerous scientific papers and books, with his latest being “The Big R-Book: From Data Science to Big Data and Learning Machines.” His work continues to influence both academic and practical approaches to finance, risk, and data science.



[Wikipedia](#)
[Website](#)

Contact
[LinkedIn](#)
[e-mail](#)

mobile: +48
790 715 002



DISCLAIMER / LEGAL NOTICE

1. Personal Capacity & Non-Affiliation

The content, presentation materials, oral remarks, analyses, and conclusions expressed herein are exclusively those of the author acting in his personal, independent capacity. They do not represent, reflect, or constitute the official views, corporate policies, academic positions, or commercial strategies of **HSBC**, **AGH University of Krakow (AGH)**, **Krakow University of Economics (UEK)**, or any of the author's other current or former affiliations, including their respective subsidiaries, parent entities, or affiliates.

2. Absence of Investment Advice or Recommendation

This material is prepared and distributed strictly for general informational and educational purposes only. Nothing contained in this presentation or handout constitutes financial, legal, tax, or investment advice. Specifically:

- It does not constitute an **investment recommendation** or **information recommending or suggesting an investment strategy** as defined under Regulation (EU) No 596/2014 of the European Parliament and of the Council (Market Abuse Regulation — MAR).
- It does not constitute **investment advice** or a **personal recommendation** within the meaning of Directive 2014/65/EU of the European Parliament and of the Council (MiFID II).
- It does not constitute an activity subject to regulatory oversight under the Polish Act of 29 July 2005 on Trading in Financial Instruments (*Ustawa z dnia 29 lipca 2005 r. o obrocie instrumentami finansowymi*).

3. No Offer or Solicitation

This presentation does not constitute an offer, invitation, solicitation, or marketing communication to buy, sell, subscribe for, or underwrite any financial instruments, securities, currencies, or underlying assets in any jurisdiction.

4. Disclaimer of Liability & Accuracy

While information and data presented have been obtained from sources believed to be reliable at the time of preparation, no representation or warranty, express or implied, is made regarding its accuracy, completeness, timeliness, or fitness for any particular purpose. The author, HSBC, AGH, UEK, and associated entities expressly disclaim any and all liability for any direct, indirect, incidental, or consequential losses or damages arising from reliance upon or use of this presentation or handout.

Contents

Contents	4
1 Defining Ethics	4
2 Defining Artificial Intelligence	7
3 Example: Insurance Onboarding	9
4 Further Ethical Conundrums	11
5 Conclusions	13

1 Defining Ethics

A Formal Definition – Cambridge Dictionary

Ethics

the study of what is morally right and wrong, or a set of beliefs about what is morally right and wrong.

Britannica: ethics, also called moral philosophy, the discipline concerned with what is morally good and bad and morally right and wrong. The term is also applied to any system or theory of moral values or principles.

How should we live? Shall we aim at happiness or at knowledge, virtue, or the creation of beautiful objects? If we choose happiness, will it be our own or the happiness of all? And what of the more particular questions that face us: is it right to be dishonest in a good cause? Can we justify living in opulence while elsewhere in the world people are starving? Is going to war justified in cases where it is likely that innocent people will be killed? Is it wrong to clone a human being or to destroy human embryos in medical research? What are our obligations, if any, to the generations of humans who will come after us and to the nonhuman animals with whom we share the planet?

Ethics deals with such questions at all levels. Its subject consists of the fundamental issues of practical decision making, and its major concerns include the nature of ultimate value and the standards by which human actions can be judged right or wrong.

The terms ethics and morality are closely related. It is now common to refer to ethical judgments or to ethical principles where it once would have been more accurate to speak of moral judgments or moral principles. These applications are an extension of the meaning of ethics. In earlier usage, the term referred not

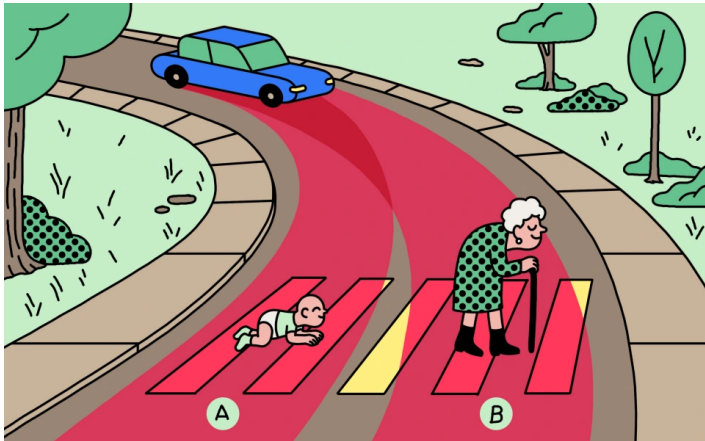


Figure 1: You can only correct the course of the car a little, so you can reduce the deaths from two to one. Do you choose track A or B? — [source: MIT Technology Review](#) — Hao, 2018

to morality itself but to the field of study, or branch of inquiry, that has morality as its subject matter. In this sense, ethics is equivalent to moral philosophy.

Although ethics has always been viewed as a branch of philosophy, its all-embracing practical nature links it with many other areas of study, including anthropology, biology, economics, history, politics, sociology, and theology. Yet, ethics remains distinct from such disciplines because it is not a matter of factual knowledge in the way that the sciences and other branches of inquiry are. Rather, it has to do with determining the nature of normative theories and applying these sets of principles to practical moral problems.

Ethical dilemmas are more important than ever. With the next figures we explore a few variations of the trolley problem. The Trolley Problem is no longer just a thought experiment for philosophy students; it is a technical requirement for autonomous systems. Engineers must decide how a car should value a child's life versus an elderly person's life in a situation where a collision is unavoidable.

AI Dilemma: Kill/Save the Baby or Old Person?

It appears that the choices people make is somehow determined by culture. In Western culture a majority of people will choose to save the baby, in West Asian cultures typically the older person is saved.

The AI Version of the Trolley

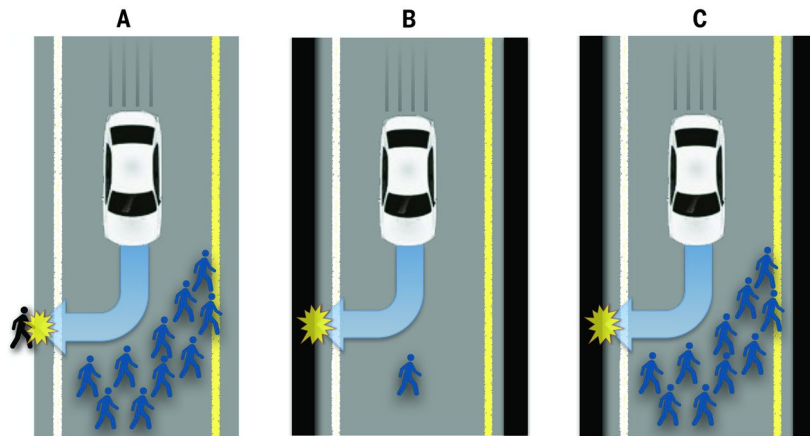


Figure 2: *source: Science — Bonnefon, Shariff, and Rahwan, 2016*
Should we veer and kill the person who is on the pavement? What if there is only one person on the street and there is a wall? How many people to change choices?

Is it ethical to remove the house of a squirrel to build your house?



Figure 3: *Is removing a tree that houses a squirrel ethically correct when building your home? What if an advanced society wants to make Earth their home?*

Most people will find that removing the tree that houses the squirrel is morally acceptable. If the squirrel would be another human, typically (in mod-

ern times) we would not accept this. However, we and the squirrel share a common ancestor that lived ca. 180 million years ago.

We are only separated by 180 million years, and we do not see the squirrel as a valid discussion partner for the location of our house. Now, imagine an alien society that has a 180 million years earlier start of their civilisation than us. They will probably not recognise us as intelligent partners, and just like the squirrel we will be defenceless against their technology if they want to take our planet.

Suddenly the same problem feels very different.

What is ethically acceptable depends on your viewpoint

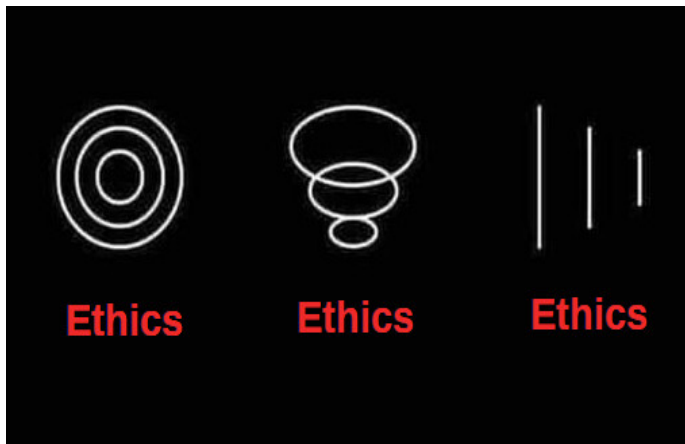


Figure 4: Ethics is complex and it depends on your point of reference and your frame.

2 Defining Artificial Intelligence

A Definition of AI

Artificial Intelligence

A cross-disciplinary approach to understanding, modelling, and replicating intelligence and cognitive processes by invoking various computational, mathematical, logical, mechanical, and even biological principles and devices. —Brundage, 2015

or ...

AI in Britannica.com

the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings

or ...

Defining AI Leaves Simple Questions Open

- What is exactly Intelligence?
the ability to acquire and apply knowledge and skills (Oxford Dictionary)
- How to Measure Intelligence?
IQ? EQ? What about logical, interpersonal, spacial, musical and many other forms?)
- What is Self-consciousness? Sentience? Life?
?
- When should an AI get animal or human rights?
?

Intelligence is defined by the Oxford English Dictionary as the ability to acquire and apply knowledge and skills. However, quantifying this capability remains a complex endeavor. Traditional psychometric assessments, such as the Intelligence Quotient (IQ), focus primarily on logical-mathematical and spatial reasoning. In contrast, modern frameworks account for broader dimensions of cognition, including Emotional Intelligence (EQ) alongside interpersonal, musical, bodily-kinesthetic, and intrapersonal capacities. Measuring intelligence therefore requires recognizing that mental capability is multifaceted rather than a unified, linear metric.

Beyond cognitive processing lie the deeper ontological concepts of sentience, self-consciousness, and life itself. Sentience refers to the capacity to experience subjective feelings and sensations, whereas self-consciousness represents an explicit awareness of one's own existence and internal mental states. Defining the precise moment when physical or synthetic organization transforms into a living, conscious entity is one of the most formidable questions in contemporary philosophy. As non-biological systems demonstrate increasingly sophisticated adaptive behaviors, the boundary between automated execution and genuine self-awareness becomes progressively obscured.

This philosophical ambiguity directly influences the discourse surrounding moral status and legal rights. Society must inevitably address whether moral consideration ought to depend upon an entity's physical origin or its subjective

internal experience. Consider the varied modes of artificial creation depicted in modern fiction: an entity constructed through mechanical and digital engineering, such as the holographic companion Joi in **Blade Runner 2049**; a synthetic being crafted through advanced biological engineering, such as Rachael in the **Blade Runner** universe; or an inanimate object brought to life through magic, as seen in the film **Ted**. Regardless of whether an entity comes into being via biological synthesis, computer engineering, or supernatural phenomena, humanity will need to confront a critical question: if a created being demonstrates verifiable sentience and self-consciousness, on what ethical ground can basic human or animal rights be withheld? Ultimately, moral standing may need to be determined not by how an entity was constructed, but by its capacity to feel, reflect, and experience existence.

3 Example: Insurance Onboarding

Car Insurance Claims Data

The data contains

- driving related:
 - average daily travel
 - professional or private use
 - living in urban or city area
- car related:
 - type of car (eg. minivan, sports car, sedan, etc.)
 - age of the car
- behaviour on our books:
 - time with us
 - old claims
- driver related:
 - has driver's licence been revoked
 - penalty points on driver's licence
 - years on job (work related)

... but also contains

- related to gender and family:
 - number of kids at home
 - single parent or not
 - marital status
 - gender
- education and profession
 - education
 - occupation
- wealth related:
 - value of the car
 - value of the home

These are the “**protected features**”

Protected features are variables that, for ethical or legal reasons, should not influence a decision-making model. In insurance, using factors like gender

or marital status can lead to discriminatory pricing, even if those factors are statistically predictive of risk.

The cost of being politically correct?

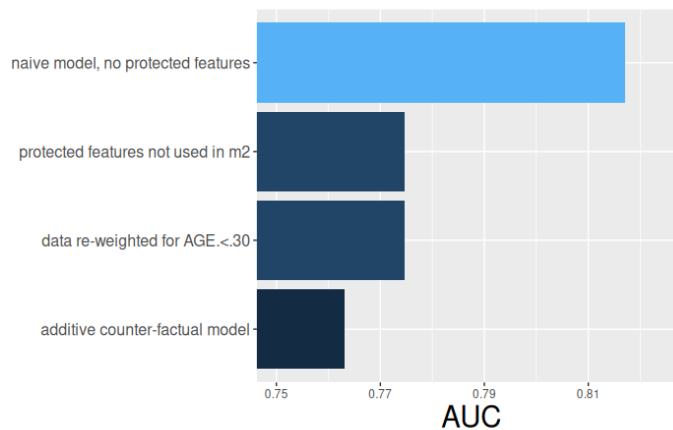


Figure 5: The discriminating power of different models. The higher the AUC, the stronger the model, the lower the losses on the portfolio, the higher profit (or lower prices) can be.

Area Under the Curve (AUC) is a foundational statistical metric used to evaluate the performance of binary classification models, particularly the Receiver Operating Characteristic (ROC) curve in risk analysis. Within the context of insurance underwriting, AUC measures a model’s discriminative power—its structural capability to correctly rank-order policyholders so that individuals who will eventually file an accident claim receive higher risk scores than those who will remain claim-free.

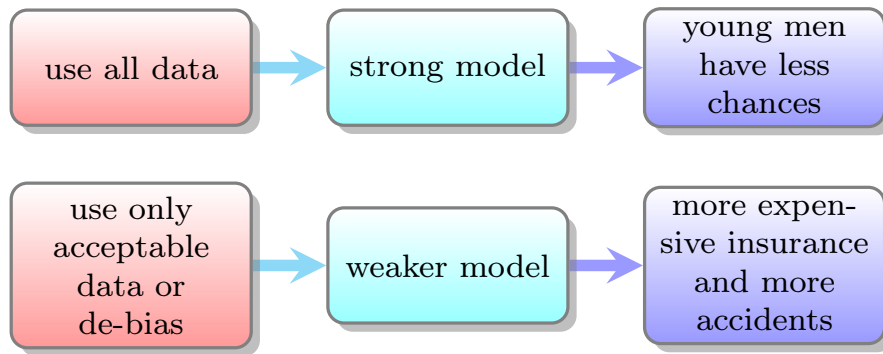
- **Discriminative Power:** AUC quantifies the probability that a randomly selected high-risk customer is assigned a higher risk score than a randomly chosen low-risk driver. An AUC score of 0.5 denotes performance no better than random guessing, whereas a score of 1.0 indicates a theoretical, flawless predictor.
- **Portfolio Efficiency:** Stronger models with higher AUC scores allow insurers to fine-tune pricing tiers, minimize cross-subsidization, reduce portfolio-wide losses, and offer more competitive pricing to safe drivers.
- **The Ranking Limitation:** AUC evaluates *relative ordering* rather than absolute risk probability or financial magnitude. It confirms whether high-risk drivers are positioned above low-risk drivers, but it remains entirely blind to the actual monetary value of potential claims.

Because AUC is strictly a rank-order metric, it cannot be directly converted into a precise monetary figure or a specific financial impact. When compliance frameworks or ethical mandates require data scientists to strip out protected features—such as gender, marital status, or socioeconomic indicators—the model’s AUC inevitably drops, representing a loss in statistical precision.

However, translating this drop in AUC into an exact dollar amount or premium hike for an individual policyholder is fundamentally impossible through the metric alone. A reduction in AUC from 0.85 to 0.80 indicates degraded portfolio discrimination, but it does not dictate how pricing tables must shift.

Quantifying the financial cost of eliminating bias requires mapping the altered risk distributions against commercial underwriting rules, loss cost projections, and regulatory pricing constraints. While a weaker model inherently introduces portfolio inefficiencies, the exact financial burden experienced by specific consumer groups remains concealed behind complex actuarial formulas rather than visible within the AUC score itself.

How to avoid bias? Two alternatives



4 Further Ethical Conundrums

AI Governance & Ethical Horizons



Data Governance: Autonomy, Security, and Market Power

- **Informed Consent & Behavioral Autonomy:** Preserving genuine individual free will by combatting deceptive dark patterns, pervasive surveillance, and algorithmically driven behavioral manipulation.
- **Data Safeguarding & Regulatory Compliance:** Establishing robust cyber-security protocols to prevent systemic breaches while ensuring strict adherence to global privacy standards (e.g., GDPR, CCPA).
- **Intellectual Property & Anti-Competition:** Clarifying ownership rights over training corpuses, user prompts, and synthetic outputs to mitigate platform lock-in and data monopolies.

Algorithmic Governance: The FATE Framework

- **Systemic Fairness & Data Curation:** Mitigating historical bias in automated scoring while addressing the ethical dilemma of active data intervention versus objective representation.



Figure 6: Visual biases as clear: is the pole hollow or is there a pyramid on top of it?

- **Legal & Moral Accountability:** Defining clear chains of responsibility and legal liability when autonomous systems commit errors or cause tangible harm.
- **Transparency & Explainability:** Requiring open disclosure of algorithmic deployment alongside auditable, human-interpretable models for high-stakes decision-making.

Societal Horizons: Alignment, Moral Status, and Agency

- **The Alignment Problem & Lethal Agency:** Ensuring superintelligent goals remain compatible with human survival, while setting absolute ethical boundaries on lethal autonomous weapons systems.
- **Post-Human Moral Status:** Determining the legal rights, digital bodily autonomy, and shutdown protections for artificial sentience, cybernetically enhanced humans, and uploaded consciousness.
- **Normative Authority ("Who Decides?"):** Identifying which global bodies hold the legitimate mandate to construct, regulate, and enforce universal standards of machine ethics.

5 Conclusions

Visual Bias 1

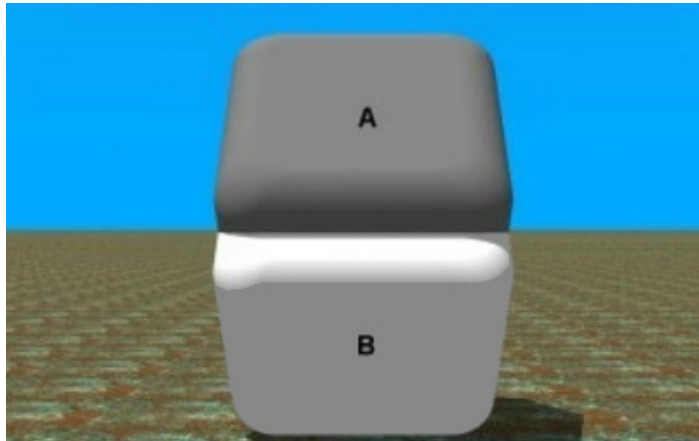


Figure 7: Visual biases cannot be avoided. We know that A and B have the same shade of grey, but we cannot see it correctly.

Visual Bias 2

Fairness is a matter of perspective

We do have an innate sense for what is fair and what not, however that sense is heavily biased towards ourselves. For example, we don't see it as a moral issue to build a house and destroy the home of a squirrel in the process. The squirrel, however, is only separated from us by 180 million years of separate evolution. From a society that is a million years ahead of it might seem obvious to simply make earth their home and get rid of us.

If that seems unfair, turn it around. Would it be fair to colonise a planet inhabited by squirrel-like mammals and make it our home?

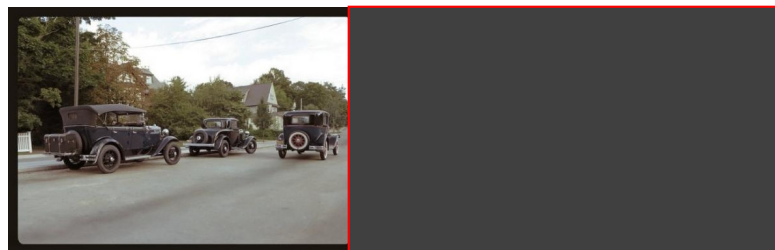


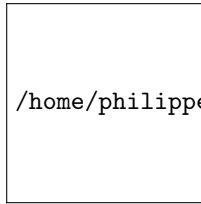
Figure 8: Fairness is a matter of perspective

Conclusions

Intuition	Innate ethical intuitions are insufficient for AI.
Perspective	Subjectivity : Ethical stance depends on perspective.
Judgment	Context : Morality requires context-sensitive judgment.
Trade-offs	Balance : Ethical decision-making involves trade-offs
Responsibility	You have an individual and non-transferable responsibility.

References

- Bonnefon, Jean-François, Azim Shariff, and Iyad Rahwan (2016). ‘The social dilemma of autonomous vehicles’. In: *Science* 352.6293, pp. 1573–1576.
- Brundage, Miles (2015). ‘Taking superintelligence seriously: Superintelligence: Paths, dangers, strategies by Nick Bostrom (Oxford University Press, 2014)’. In: *Futures* 72, pp. 32–35.
- Hao, Karen (2018). ‘Should a self-driving car kill the baby or the grandma? Depends on where you’re from’. In: *URL:https://www.technologyreview.com/s/612341/a-global-ethics-study-aims-to-help-ai-solve-the-self-drivingtrolley-problem/*.



`/home/philippe/Documents/courses/img/mba_tech_logo.jpg`

ETHICS IN DATA SCIENCE

DATA SCIENCE IN PRACTICE

DR. PHILIPPE J.S. DE BROUWER

ALL RIGHTS RESERVED BY THE AUTHOR
prepared for: AGH University of Krakow

SEPTEMBER 28, 2026

Table of Contents

Contents

Contents	2
1 Bias in Data and Mathematical Models	2
1.1 Basic Definitions	2
1.2 Algorithmic Fairness	3
1.3 The Confusion Matrix and Related Measures	3
1.4 Measures for Algorithmic Fairness	4
2 Origin and Types of Bias	5
3 Conclusions	6
4 Case Study	7

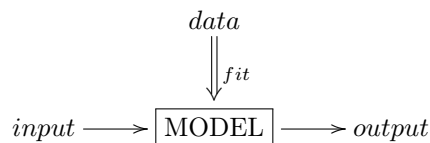
1 Bias in Data and Mathematical Models

1.1 Basic Definitions

What is a Model?

Definition 1 (Mathematical Model). A mathematical model is a description of a system using mathematical concepts and language.

Use of a model in banking is fact-based (data is used to fit the model):



What is Bias in Models?

Definition 2 (Bias in Mathematical Models). Model Bias (also Algorithmic Bias) refers to the systematic and repeatable error in a Model that creates outcomes that are statistically at odds with the reality of the population whose behaviour it is supposed to reflect.

1.2 Algorithmic Fairness

Fairness Assumptions

Assume S the protected feature with an under-privileged group S_u and a privileged group S_p , and predicted outcomes $\hat{Y}$.

- A. **Independence:** the probability of being in S_p or S_u has nothing to do with $\hat{Y}$ – $\hat{Y}$ is independent of S .
- B. **Separation:** The predicted probability of having a favourable outcome has nothing to do with membership of the protected feature – $\hat{Y}$ is independent of S , given Y .
- C. **Sufficiency:** the prediction should not depend on the protected group – Y is independent of S , given $\hat{Y}$.

1.3 The Confusion Matrix and Related Measures

A simple and clear way to show what the model does is a “confusion matrix”: simply show how much of the observations are classified correctly by the model.

Useful Concepts for the Confusion Matrix

The following are useful measures for how good a classification model fits its data:

- *Accuracy:* The proportion of predictions that were correctly identified.
- *Precision* (or positive predictive value): The proportion of positive cases that correct.
- *Negative predictive value:* The proportion of negative cases that were correctly identified.
- *Sensitivity* or Recall: The proportion of actual positive cases which are correctly identified.
- *Specificity:* The proportion of actual negative cases which are correctly identified.

Let us use the following definitions:

- Objective concepts (depends only on the data):
 - P : The number of positive observations ($y = 1$)
 - N : The number of negative observations ($y = 0$)
- Model dependent definitions:
 - True positive (TP) the positive observations ($y = 1$) that are by the model correctly classified as positive;

	Observed pos.	Observed neg.	
Pred. pos.	TP	FP	Pos.pred.val = $\frac{TP}{TP+FP}$
Pred. neg.	FN	TN	Neg.pred.val = $\frac{TN}{FN+TN}$
	Sensitivity	Specificity	Accuracy
	$= \frac{TP}{TP+FN}$	$= \frac{TN}{FP+TN}$	$= \frac{TP+TN}{TP+FN+FP+TN}$
	$= \frac{TP}{TP+FN}$	$= \frac{TN}{FP+TN}$	$= \frac{TP+TN}{TP+FN+FP+TN}$

Table 1: The confusion matrix, where “pred.” refers to the predictions made by the model, “pred.” stands for “predicted,” and the words “positive” and “negative” are shortened to three letters.

- False positive (FP) the negative observations ($y = 0$) that are by the model incorrectly classified as positive – this is a false alarm (Type I error);
- True negative (TN) the negative observations ($y = 0$) that are by the model correctly classified as negative;
- False negative (FN) the positive observations ($y = 1$) that are by the model incorrectly classified as negative – miss (Type II error).

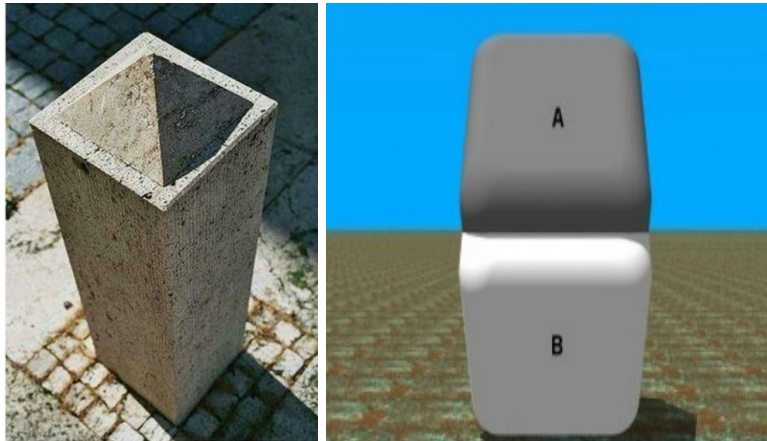
The Definition of the Confusion Matrix

1.4 Measures for Algorithmic Fairness

Fairness Metrics

- Equal Opportunity:** $FNR_p = FNR_u$ (equivalent with $TPR_p = TPR_u$)
- Predictive Equality:** $FPR_p = FPR_u$ (equivalent with $TNR_p = TNR_u$)
- Equalised Odds:** both previous
- Predictive Parity** or outcome test: $Prec_p = Prec_u$ (equal positive predictive value or precision ($Prec = \frac{TP}{TP+FP}$))
- Demographic Parity:** membership of S has no correlation with favourable outcome ($\hat{Y}$).

Visual Biases are systematic miss-interpretations



2 Origin and Types of Bias

General Causes of Bias in Data

- **Confirmation Bias:** bias governs what we search for and we find what we search for (e.g. belief preservation via social media)
- **Selection Bias:** sample is not representative
- **Historical Bias:** socio-cultural prejudices are reflected in data (e.g. Google image search)
- **Survival Bias:** the winner takes it all and losers are forgotten (e.g. hedge fund performance statistics)
- **Availability Bias:** violence has systematically decreased over the millennia, however we might think of modern times to be more violent. (focus on through the cycle data, different perspectives)
- **Outlier Bias:** thinking about a startup one things of Google, Amazon, Facebook, etc. These are the outliers. (use median instead of average, identify outliers, etc.)

Process Related Bias

- **Reporting bias:** selective reporting of some data: citation bias, language bias (ignore reports in other languages), duplicate publication bias (copied data found twice), location bias (some studies are hard to find), publication bias (secret or not popular data is hard to find), outcome reporting bias (e.g. company does not report with much bravo when results are bad), time lag bias
- **Automation bias:** we prefer automated systems to provide data

- **Selection bias:** data is not representative (e.g. sampling bias, convergence bias (only people who got a loan are in our database), participation bias, etc.)
- **Over-generalisation bias:** no black swans in the data
- **Group Attribution bias:** generalisation of stereotypes, in-group bias (preference for members in the group), out-group bias (stereotype for other groups), etc.
- **Implicit bias:** search for conforming information

What causes Bias in Models?

Bias can appear due to

- A. biases in the data
 - less representative group
 - the data is not a good representation of the reality (e.g. survivor bias, previous societal bias or limitations, etc.)
- B. the data processing pipeline (extraction, cleaning, transformation, binning)
- C. the model development process (including selection of algorithm, variables, etc.)
- D. the particulars of model implementation

3 Conclusions

Fairness is a matter of perspective

We do have an innate sense for what is fair and what not, however that sense is heavily biased towards ourselves. For example, we don't see it as a moral issue to build a house and destroy the home of a squirrel in the process. The squirrel, however, is only separated from us by 250 million years of separate evolution. From a society that is a million years ahead of it might seem obvious to simply make earth their home and get rid of us.

If that seems unfair, turn it around. Would it be fair to colonise a planet inhabited by squirrel like mammals and make it our home?



Figure 1: Fairness is a matter of perspective

Bias is learnt



4 Case Study

Case study on addressing bias: car insurance



References

- Agarwal, Sray and Shashin Mishra (2021). *Responsible AI: Implementing Ethical and Unbiased Algorithms*. Springer.
- Coeckelbergh, Mark (2020). *AI ethics*. Mit Press.
- De Brouwer, Philippe J. S. (2020). *The Big R-Book: From Data Science to Learning Machines and Big Data*. New York: John Wiley & Sons, Ltd. ISBN: 978-1-119-63272-6.
- Dubber, Markus Dirk, Frank Pasquale, and Sunit Das (2020). *The Oxford handbook of ethics of AI*. Oxford Handbooks.
- Hardt, Moritz, Solon Barocas, and Arvind Narayanan (2018). 'Fairness in Machine Learning: Limitations and Opportunities'. In: *Solon Barocas Moritz Hardt and Arvind Narayanan*.

Car Insurance

Acceptance Model

Philippe De Brouwer

2026-09-28

Table of Contents

1	Executive Summary	2
2	Introduction	2
3	Initial Data Processing	2
3.1	Reading in the files from <code>car_insurance_dataprep.Rmd</code>	2
3.2	Common data manipulations	3
3.3	Separating test data	4
4	The Logistic Regression 1	4
4.1	Fit the Model With only Relevant Variables	7
4.2	Cross Validation	10
4.2.1	Storing of the results	10
4.2.2	Base R	12
4.2.3	Cross-Validation in the <code>tidyverse</code>	15
5	Decision Tree	17
5.1	Decision Tree on binned data	17
5.2	Decision Tree on numerical data	21
6	Random Forest	25
6.1	Random Forest on the binned data	27
7	A Neural Network	28
7.1	Neural networks in R	28
7.2	The package <code>nnet</code>	28
7.3	The package <code>neuralnet</code>	30
8	Support Vector Machine (SVM)	31
9	PCA	31
10	The Final Model Choice	33
10.1	The interpretation of the AUC findings	33
10.2	Model Selection and Unbiased Performance Estimation	34
10.3	Optimal Cut-Off	35
11	Appendices	36
11.1	Acknowledgement	36
11.2	Notes on the templates	37

1 Executive Summary

This part 2 we build a binary acceptance model: do we accept the customer or not. In this file we disregard the question whether it is ethical to use certain features such as gender, married status, income, etc. We only try to build the best model and refer for the ethical discussion to the file `car_insurance_ethics`.

2 Introduction

The work-flow is inspired by: (De Brouwer 2020) and is prepared by Philippe De Brouwer to illustrate the concepts explained in the book, and is the standard in stochastic modelling.

The first step in building a reliable predictive model is to split the original dataset into two parts: a training set and a test set. The training set is used exclusively for fitting and tuning the model, while the test set is held completely aside for a final unbiased evaluation. This separation ensures that the test data remains unseen during model development, providing a realistic assessment of how the model will perform on new data (Smith and Doe 2023).

To tune and select the best model, the training data is further divided internally via cross-validation. For example, in k -fold cross-validation, the training set is split into k equally sized folds. Iteratively, the model is trained on $k-1$ folds and validated on the remaining fold. This procedure is repeated so each fold serves once as validation data. Cross-validation thus allows comprehensive model evaluation and tuning while maximizing the use of available training data (Taylor and Nguyen 2024).

It is crucial to keep the test set separate and untouched until the very end. Using the test set for model tuning or validation can lead to overfitting, as the model might indirectly learn from this data. This would give overly optimistic performance estimates that will not hold on genuinely new data. By restricting validation only to splits within the training data and reserving the test set for final evaluation, the modeling process maintains an unbiased measure of performance.

In summary, the entire modeling pipeline follows these key steps:

- **Initial split** into training and test sets to separate model development from evaluation.
- **Cross-validation** inside the training data to tune and select models reliably.
- **Final training** of the chosen model on the full training set.
- **Unbiased evaluation** on the preserved test set to estimate real-world performance.

This disciplined workflow ensures trustworthy, generalizable models that perform well in practice.

Schematic overview of the modelling cycle.

3 Initial Data Processing

3.1 Reading in the files from `car_insurance_dataprep.Rmd`.

We need to load the data as prepared in previous file

```
setwd("/home/philippe/Documents/courses/lectures/car_insurance")

# import functions:
source("ethics_functions.R")

# List the functions defined in this file:
tmt.env <- new.env() # create a temporary environment
sys.source("ethics_functions.R", envir = tmt.env)
utils::lsf.str(envir=tmt.env)
```

```

## dollars_to_numeric : function (input)
## expand_factor2bin : function (data, col)
## f_comment_iv : function (df, x, y)
## fAUC : function (model, data, ...)
## get_best_cutoff : function (fpr, tpr, cutoff, cost.fp = 1)
## make_dependency_hist_plot : function (data, x, y, title, method = "loess", q_ymax = 1, q_xmax = 1)
## make_dependency_plot : function (data, x, y, title, method = "loess", q_ymax = 1, q_xmax = 1)
## make_WOE_table : function (df, y)
## opt_cut_off : function (perf, pred, cost.fp = 1)
## pct_table : function (x, y, round_digits = 2)
## space_to_underscore : function (input)
## str_clean : function (x)

rm(tmt.env) # Remove the temporary environment

# Read in the binned binary data:
d_bin <- readRDS('./d_bin.R')

# Read in the factorial data:
d_fact <- readRDS('./d_fact.R')

# The normalised and numerical data
d_norm <- readRDS('./d_norm.R')

```

3.2 Common data manipulations

We replace CLAIM_FLAG with a variable that is 1 when the outcome is positive. The only purpose of this is to make the results easier to understand as “better insurance risk.”

```

d_fact <- d_fact %>%
  mutate(isGood = 1 - as.numeric(paste(d_fact$CLAIM_FLAG))) %>%
  select(-c(CLAIM_FLAG))

d_bin <- d_bin %>%
  mutate(isGood = 1 - as.numeric(paste(d_bin$CLAIM_FLAG))) %>%
  select(-c(CLAIM_FLAG))

d_norm <- d_norm %>%
  mutate(isGood = 1 - as.numeric(paste(d_norm$CLAIM_FLAG))) %>%
  select(-c(CLAIM_FLAG))

transform_factors_to_binary <- function(df) {
  # Extract numeric columns
  numeric_cols <- df %>% select(where(is.numeric))

  # Extract factors and convert them to dummy variables using model.matrix
  factor_cols <- df %>% select(where(is.factor))

  if (ncol(factor_cols) > 0) {
    # Create dummy variables; remove intercept column
    dummies <- model.matrix(~ . - 1, data = factor_cols) %>% as.data.frame()

    # Combine numeric and dummy columns
    result <- bind_cols(numeric_cols, dummies)
  } else {
    result <- numeric_cols
  }
}

```

```

}

return(result)
}

d_num <- d_norm %>%
  transform_factors_to_binary

```

3.3 Separating test data

```

set.seed(2025)
sampled_rows <- d_fact %>%
  slice_sample(prop = 0.85) %>% # 15% kept as test data
  mutate(row_id = row_number()) %>% # get the row numbers in a column
  pull(row_id) # this does the same as '$' hence gets that column

# we will keep all data in one list object (list of lists of data-frames)
d <- list()
# Add train and test for fact dataset
d$fact <- list(
  train = d_fact %>% slice(sampled_rows),
  test = d_fact %>% slice(-sampled_rows)
)
# Add train and test for bin dataset
d$bin <- list(
  train = d_bin %>% slice(sampled_rows),
  test = d_bin %>% slice(-sampled_rows)
)
# Add train and test for numerical dataset
d$norm <- list(
  train = d_norm %>% slice(sampled_rows),
  test = d_norm %>% slice(-sampled_rows)
)
d$num <- list(
  train = d_num %>% slice(sampled_rows),
  test = d_num %>% slice(-sampled_rows)
)
# Note that when we write slice(d_bin, -sampled_rows), it results in the same columns,
# However, the class is different

```

4 The Logistic Regression 1

First, we will fit a logistic regression using all variables that from a mathematical point of view make sense. For the logistic regression we use `d_fact`, where the data is categorical, but with meaningful labels. While we could use the other database, it is a little easier to use since R will create binary data from factors anyhow for us.

First try all variables

```

m0 <- glm(isGood ~ ., d$fact$train, family = 'binomial')
summary(m0)

##
## Call:

```

```
## glm(formula = isGood ~ ., family = "binomial", data = d$fact$train)
##
## Coefficients: (9 not defined because of singularities)
##
## Estimate Std. Error z value
## (Intercept) -0.966821 0.308804 -3.131
## KIDSDRIV1 -0.788971 0.115742 -6.817
## KIDSDRIV2 -1.057324 0.166085 -6.366
## KIDSDRIV3 -1.394764 0.320995 -4.345
## KIDSDRIV4 -1.717860 1.132738 -1.517
## YOJ>1 0.359711 0.141642 2.540
## PARENT1No 0.291934 0.120733 2.418
## MSTATUSNo -0.568821 0.089250 -6.373
## GENDERF 0.097545 0.211576 0.461
## EDUCATIONz_High_School 0.012855 0.095174 0.135
## EDUCATIONBachelors 0.419377 0.117837 3.559
## EDUCATIONMasters 0.281481 0.159684 1.763
## EDUCATIONPhD 0.383821 0.189411 2.026
## CAR_USEPrivate 0.814279 0.086996 9.360
## TIF2--5 0.269600 0.079674 3.384
## TIF6--8 0.422875 0.077950 5.425
## TIF>8 0.682812 0.081931 8.334
## CAR_TYPEPickup -0.434302 0.103915 -4.179
## CAR_TYPEPTruck_Van -0.538400 0.118818 -4.531
## CAR_TYPESports_Car -0.860965 0.128593 -6.695
## CAR_TYPEz_SUV -0.682543 0.110251 -6.191
## RED_CARyes 0.015415 0.086959 0.177
## OLDCLAIM>0 -0.529751 0.089248 -5.936
## CLM_FREQ1 0.057582 0.102390 0.562
## CLM_FREQ2 0.082207 0.098308 0.836
## CLM_FREQ>2 NA NA NA
## REVOKEDYes -0.691718 0.079943 -8.653
## MVR_PTS1 -0.037122 0.091351 -0.406
## MVR_PTS2 -0.257476 0.097035 -2.653
## MVR_PTS3--4 -0.294899 0.083999 -3.511
## MVR_PTS>4 -0.421056 0.092983 -4.528
## CAR_AGE2--7 0.114551 0.084633 1.354
## CAR_AGE8--11 0.048554 0.089906 0.540
## CAR_AGE12--15 -0.025604 0.118664 -0.216
## CAR_AGE>=16 -0.092835 0.145898 -0.636
## URBANICITYz_Highly_Rural/ Rural 2.598655 0.118766 21.881
## KIDSDRV1 NA NA NA
## KIDSDRV>=2 NA NA NA
## AGE[33,40) 0.233747 0.178975 1.306
## AGE[40,44) 0.652378 0.189654 3.440
## AGE[44,46) 0.774779 0.215723 3.592
## AGE[46,57) 0.797478 0.178286 4.473
## AGE[57, Inf) 0.040091 0.213848 0.187
## HOMEKIDS1 -0.096688 0.119909 -0.806
## HOMEKIDS>=2 -0.009502 0.108397 -0.088
## INCOME[15000,70000) 0.258789 0.118289 2.188
## INCOME[70000,85000) 0.356790 0.153911 2.318
## INCOME[85000,120000) 0.658535 0.162066 4.063
## INCOME[120000, Inf) 0.686472 0.186204 3.687
## HOME_VAL[80000,220000) 0.226501 0.085954 2.635
```

## HOME_VAL[220000,260000)	0.276067	0.122423	2.255
## HOME_VAL[260000, Inf)	0.410341	0.128295	3.198
## OCCUPATIONDoctor%,%Home_Maker%,%Lawyer%,%Manager	0.750797	0.115668	6.491
## OCCUPATIONProfessional	0.248048	0.119304	2.079
## OCCUPATIONStudent%,%z_Blue_Collar	0.190077	0.099078	1.918
## TRAVTIME25--40	-0.320326	0.073301	-4.370
## TRAVTIME>40	-0.669471	0.074613	-8.973
## BLUEBOOK[12000,20000)	0.363921	0.094754	3.841
## BLUEBOOK[20000, Inf)	0.586124	0.112795	5.196
## BLUEBOOK[7000,12000)	0.362831	0.092519	3.922
## GENDER.AGEF.[33,40)	0.214722	0.227411	0.944
## GENDER.AGEF.[40,44)	-0.142955	0.238400	-0.600
## GENDER.AGEF.[44,46)	0.053155	0.276325	0.192
## GENDER.AGEF.[46,57)	-0.200864	0.215766	-0.931
## GENDER.AGEF.[57, Inf)	-0.456866	0.268061	-1.704
## GENDER.AGEM.[-Inf,33)	NA	NA	NA
## GENDER.AGEM.[33,40)	NA	NA	NA
## GENDER.AGEM.[40,44)	NA	NA	NA
## GENDER.AGEM.[44,46)	NA	NA	NA
## GENDER.AGEM.[46,57)	NA	NA	NA
## GENDER.AGEM.[57, Inf)	NA	NA	NA
##	Pr(> z)		
## (Intercept)	0.001743	**	
## KIDSDRIV1	9.32e-12	***	
## KIDSDRIV2	1.94e-10	***	
## KIDSDRIV3	1.39e-05	***	
## KIDSDRIV4	0.129379		
## YOJ>1	0.011099	*	
## PARENT1No	0.015605	*	
## MSTATUSNo	1.85e-10	***	
## GENDERF	0.644770		
## EDUCATIONz_High_School	0.892561		
## EDUCATIONBachelors	0.000372	***	
## EDUCATIONMasters	0.077944	.	
## EDUCATIONPhD	0.042725	*	
## CAR_USEPrivate	< 2e-16	***	
## TIF2--5	0.000715	***	
## TIF6--8	5.80e-08	***	
## TIF>8	< 2e-16	***	
## CAR_TYPEPickup	2.92e-05	***	
## CAR_TYPEPTruck_Van	5.86e-06	***	
## CAR_TYPESports_Car	2.15e-11	***	
## CAR_TYPEz_SUV	5.98e-10	***	
## RED_CARyes	0.859299		
## OLDCLAIM>0	2.93e-09	***	
## CLM_FREQ1	0.573858		
## CLM_FREQ2	0.403032		
## CLM_FREQ>2	NA		
## REVOKEDYes	< 2e-16	***	
## MVR_PTS1	0.684475		
## MVR_PTS2	0.007968	**	
## MVR_PTS3--4	0.000447	***	
## MVR_PTS>4	5.95e-06	***	
## CAR_AGE2--7	0.175896		

```

## CAR_AGE8--11                0.589159
## CAR_AGE12--15                0.829165
## CAR_AGE>=16                  0.524580
## URBANICITYz_Highly_Rural/ Rural < 2e-16 ***
## KIDSDRV1                     NA
## KIDSDRV>=2                   NA
## AGE[33,40)                   0.191541
## AGE[40,44)                   0.000582 ***
## AGE[44,46)                   0.000329 ***
## AGE[46,57)                   7.71e-06 ***
## AGE[57, Inf)                 0.851289
## HOMEKIDS1                     0.420046
## HOMEKIDS>=2                   0.930149
## INCOME[15000,70000)          0.028687 *
## INCOME[70000,85000)          0.020440 *
## INCOME[85000,120000)         4.84e-05 ***
## INCOME[120000, Inf)          0.000227 ***
## HOME_VAL[80000,220000)        0.008410 **
## HOME_VAL[220000,260000)       0.024132 *
## HOME_VAL[260000, Inf)         0.001382 **
## OCCUPATIONDoctor%,%Home_Maker%,%Lawyer%,%Manager 8.53e-11 ***
## OCCUPATIONProfessional        0.037606 *
## OCCUPATIONStudent%,%z_Blue_Collar 0.055053 .
## TRAVTIME25--40               1.24e-05 ***
## TRAVTIME>40                  < 2e-16 ***
## BLUEBOOK[12000,20000)         0.000123 ***
## BLUEBOOK[20000, Inf)          2.03e-07 ***
## BLUEBOOK[7000,12000)          8.79e-05 ***
## GENDER.AGEF.[33,40)           0.345064
## GENDER.AGEF.[40,44)           0.548744
## GENDER.AGEF.[44,46)           0.847458
## GENDER.AGEF.[46,57)           0.351887
## GENDER.AGEF.[57, Inf)         0.088318 .
## GENDER.AGEM.[-Inf,33)         NA
## GENDER.AGEM.[33,40)           NA
## GENDER.AGEM.[40,44)           NA
## GENDER.AGEM.[44,46)           NA
## GENDER.AGEM.[46,57)           NA
## GENDER.AGEM.[57, Inf)         NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##    Null deviance: 9847.1  on 8473  degrees of freedom
## Residual deviance: 7365.8  on 8412  degrees of freedom
## AIC: 7489.8
##
## Number of Fisher Scoring iterations: 5

```

Certain variables have no stars, and hence can be removed from the model.

4.1 Fit the Model With only Relevant Variables

Leave out the least relevant ones (the coefficients without stars) and build a first usable model m_1 .

We will not use further:

- KIDSDRV
- HOMEKIDS
- GENDER
- RED_CAR
- CAR_AGE
- CLM_FREQ (too much correlated to OLDCLAIM)
- GENDER.AGE

```
frm1 <- formula(isGood ~ MSTATUS + EDUCATION + OCCUPATION + TRAVTIME +
                CAR_USE + BLUEBOOK + TIF + CAR_TYPE + OLDCLAIM +
                REVOKED + MVR_PTS + URBANICITY + AGE + INCOME +
                YOJ + HOME_VAL)

m1 <- glm(frm1, d$fact$train, family = 'binomial')
summary(m1)

##
## Call:
## glm(formula = frm1, family = "binomial", data = d$fact$train)
##
## Coefficients:
##
##              Estimate Std. Error z value
## (Intercept)    -0.6312874   0.2254401  -2.800
## MSTATUSNo      -0.6308124   0.0728727  -8.656
## EDUCATIONz_High_School  0.0006344   0.0923849   0.007
## EDUCATIONBachelors  0.4133447   0.1061459   3.894
## EDUCATIONMasters  0.2018155   0.1330077   1.517
## EDUCATIONPhD      0.3526617   0.1693428   2.083
## OCCUPATIONDoctor%,%Home_Maker%,%Lawyer%,%Manager  0.7426270   0.1140782   6.510
## OCCUPATIONProfessional  0.2481928   0.1177324   2.108
## OCCUPATIONStudent%,%z_Blue_Collar  0.1932965   0.0976085   1.980
## TRAVTIME25--40    -0.3109478   0.0722808  -4.302
## TRAVTIME>40      -0.6471385   0.0735467  -8.799
## CAR_USEPrivate    0.7841293   0.0857173   9.148
## BLUEBOOK[12000,20000]  0.3538306   0.0920881   3.842
## BLUEBOOK[20000, Inf]  0.5431463   0.1042917   5.208
## BLUEBOOK[7000,12000]  0.3452725   0.0910600   3.792
## TIF2--5          0.2680254   0.0785088   3.414
## TIF6--8          0.4260817   0.0768499   5.544
## TIF>8            0.6644638   0.0808820   8.215
## CAR_TYPEPickup    -0.4117055   0.1022654  -4.026
## CAR_TYPEPTruck_Van -0.4820406   0.1115590  -4.321
## CAR_TYPESports_Car -0.8358203   0.1062582  -7.866
## CAR_TYPEz_SUV     -0.6578701   0.0849987  -7.740
## OLDCLAIM>0        -0.4958024   0.0641907  -7.724
## REVOKEDYes        -0.7082730   0.0788017  -8.988
## MVR_PTS1          -0.0687241   0.0900504  -0.763
## MVR_PTS2          -0.2601257   0.0957036  -2.718
## MVR_PTS3--4       -0.3250036   0.0827476  -3.928
## MVR_PTS>4         -0.4600868   0.0917265  -5.016
## URBANICITYz_Highly_Rural/ Rural  2.4857498   0.1160824  21.414
## AGE[33,40)        0.2489535   0.1109667   2.243
## AGE[40,44)        0.3702765   0.1167988   3.170
```

## AGE[44,46)	0.5877458	0.1345368	4.369
## AGE[46,57)	0.8019010	0.1078647	7.434
## AGE[57, Inf)	-0.1758271	0.1358845	-1.294
## INCOME[15000,70000)	0.2618639	0.1167401	2.243
## INCOME[70000,85000)	0.3629491	0.1520357	2.387
## INCOME[85000,120000)	0.6437466	0.1596532	4.032
## INCOME[120000, Inf)	0.6606064	0.1833217	3.604
## YOJ>1	0.3531809	0.1400065	2.523
## HOME_VAL[80000,220000)	0.2257238	0.0845780	2.669
## HOME_VAL[220000,260000)	0.2487442	0.1207827	2.059
## HOME_VAL[260000, Inf)	0.3778171	0.1259760	2.999
##	Pr(> z)		
## (Intercept)	0.005106	**	
## MSTATUSNo	< 2e-16	***	
## EDUCATIONz_High_School	0.994521		
## EDUCATIONBachelors	9.86e-05	***	
## EDUCATIONMasters	0.129185		
## EDUCATIONPhD	0.037294	*	
## OCCUPATIONDoctor%,%Home_Maker%,%Lawyer%,%Manager	7.52e-11	***	
## OCCUPATIONProfessional	0.035022	*	
## OCCUPATIONStudent%,%z_Blue_Collar	0.047667	*	
## TRAVTIME25--40	1.69e-05	***	
## TRAVTIME>40	< 2e-16	***	
## CAR_USEPrivate	< 2e-16	***	
## BLUEBOOK[12000,20000)	0.000122	***	
## BLUEBOOK[20000, Inf)	1.91e-07	***	
## BLUEBOOK[7000,12000)	0.000150	***	
## TIF2--5	0.000640	***	
## TIF6--8	2.95e-08	***	
## TIF>8	< 2e-16	***	
## CAR_TYPEPickup	5.68e-05	***	
## CAR_TYPEPTruck_Van	1.55e-05	***	
## CAR_TYPESports_Car	3.66e-15	***	
## CAR_TYPEz_SUV	9.96e-15	***	
## OLDCLAIM>0	1.13e-14	***	
## REVOKEDYes	< 2e-16	***	
## MVR_PTS1	0.445359		
## MVR_PTS2	0.006567	**	
## MVR_PTS3--4	8.58e-05	***	
## MVR_PTS>4	5.28e-07	***	
## URBANICITYz_Highly_Rural/ Rural	< 2e-16	***	
## AGE[33,40)	0.024865	*	
## AGE[40,44)	0.001523	**	
## AGE[44,46)	1.25e-05	***	
## AGE[46,57)	1.05e-13	***	
## AGE[57, Inf)	0.195684		
## INCOME[15000,70000)	0.024888	*	
## INCOME[70000,85000)	0.016974	*	
## INCOME[85000,120000)	5.53e-05	***	
## INCOME[120000, Inf)	0.000314	***	
## YOJ>1	0.011649	*	
## HOME_VAL[80000,220000)	0.007612	**	
## HOME_VAL[220000,260000)	0.039453	*	
## HOME_VAL[260000, Inf)	0.002708	**	

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 9847.1  on 8473  degrees of freedom
## Residual deviance: 7519.1  on 8432  degrees of freedom
## AIC: 7603.1
##
## Number of Fisher Scoring iterations: 5

# Re-use the model m and the dataset t2:
pred1 <- prediction(predict(m1, type = "response"), d$fact$train$isGood)

# Visualize the ROC curve:
#plot(performance(pred, "tpr", "fpr"), col="blue", lwd = 3)
#abline(0, 1, lty = 2)

AUC1 <- attr(performance(pred1, "auc"), "y.values")[[1]]
#paste("AUC:", AUC)

perf1 <- performance(pred1, "tpr", "fpr")
ks <- max(attr(perf1, 'y.values')[[1]] - attr(perf1, 'x.values')[[1]])
paste("KS:", ks)

## [1] "KS: 0.487609652432729"

predScores1 <- predict(m1, type = "response")

pred <- prediction(predScores1, d$fact$train$isGood)
perf <- performance(pred, "tpr", "fpr")
plot(perf, main = "ROC Curve")
```

4.2 Cross Validation

4.2.1 Storing of the results

We explain more in the concluding section, but selecting a model based on AUC (or any other performance measure) based on cross validations relies on selecting the model that has:

- low high AUC on the test-data
- low difference between AUC on train and test-data
- low variation (e.g. standard deviation) of the AUC on test data
- preferably a low difference between the variation of the AUC on train and test data

To compare these models, we need hence to keep track of the means of the AUC on train and test data as well as their standard deviation.¹

In the next code blocks we prepare the `tibble` to hold these results.

```
# prepare a dataset to keep all AUC values
d_AUC <- tibble(model = character(0),
                mn_train = numeric(0),
                sd_train = numeric(0),
                mn_test  = numeric(0),
                sd_test  = numeric(0),
```

¹Of course one can chose other measures of centrality and deviation (eg. median and mean absolute deviation (MAD)).

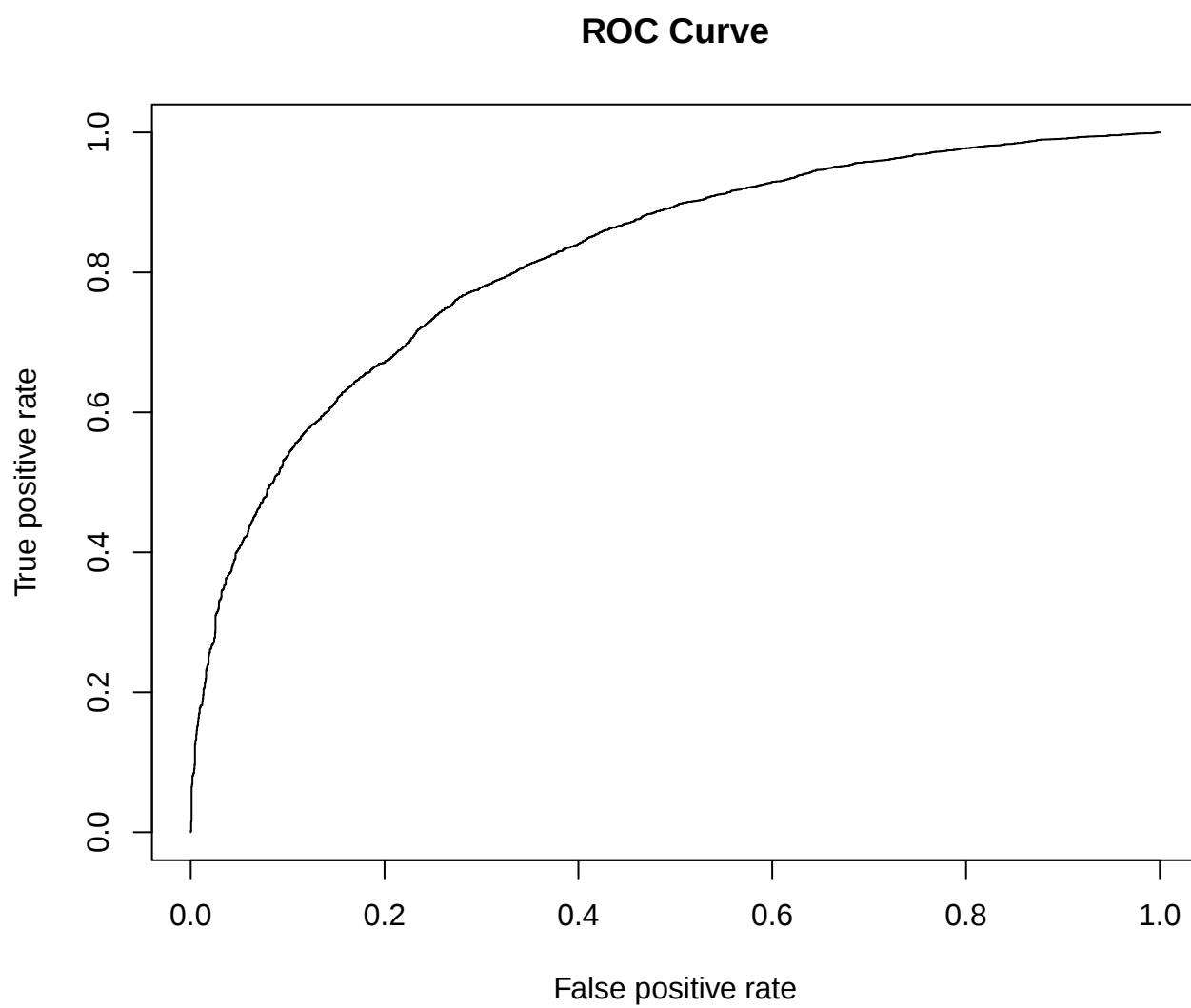


Figure 1: The ROC (receiver operating curve) for our model

```

        delta_tt = numeric(0)
    )

```

We will use a k-fold cross validation and prepare the folds for all data-sets. We want exactly the same choices of fold for each data-set and hence reset the seed each time to the same value.

```

seed <- 67

# Create the folds:
set.seed(seed = seed)
folds_fact <- crossv_kfold(d$fact$train, k = 5)
set.seed(seed = seed)
folds_norm <- crossv_kfold(d$norm$train, k = 5)
set.seed(seed = seed)
folds_bin <- crossv_kfold(d$bin$train, k = 5)
set.seed(seed = seed)
folds_num <- crossv_kfold(d$num$train, k = 5)

```

4.2.2 Base R

A k-fold cross validation in base-R is straightforward, but verbose.

```

# Manual k-fold cross-validation with Base-R
k <- 5
folds <- cut(seq(1, nrow(d$fact$train)), breaks = k, labels = FALSE)

auc_values <- numeric(k) # here we will store the results

for(i in 1:k) {
  # Create train/test splits
  test_indices <- which(folds == i)
  train_data <- d$fact$train[-test_indices, ]
  test_data <- d$fact$train[test_indices, ]

  # Train model
  model <- glm(frm1, data = train_data, family = binomial)

  # Predict and calculate AUC
  predictions <- predict(model, newdata = test_data, type = "response")
  auc_values[i] <- pROC::auc(test_data$isGood, predictions)
}

auc_per_fold <- tibble(.id = 1:k, auc = auc_values)
print(auc_per_fold)

# note that this function uses library(pROC) and library(dplyr)

f_add_kfold_auc_base <- function(data, k, model_func, model_frm, model_desc, ...) {
  # Create folds: vector assigning fold number to each row
  folds <- cut(seq(1, nrow(data)), breaks = k, labels = FALSE)

  auc_train <- numeric(k)
  auc_test <- numeric(k)

  for(i in 1:k) {
    test_indices <- which(folds == i)

```

```

train_data <- data[-test_indices, ]
test_data <- data[test_indices, ]

# Train model on training fold
model <- model_func(formula = model_frm, data = train_data, ...)

# Predict on train and calculate train AUC
preds_train <- predict(model, newdata = train_data, type = "response")
auc_train[i] <- pROC::auc(train_data$isGood, preds_train)

# Predict on test and calculate test AUC
preds_test <- predict(model, newdata = test_data, type = "response")
auc_test[i] <- pROC::auc(test_data$isGood, preds_test)
}

# Summarize fold AUC results
auc_per_fold <- tibble::tibble(.id = 1:k, auc_train = auc_train, auc_test = auc_test)
print(auc_per_fold)

# Initialize global d_AUC if missing
if(!exists("d_AUC", envir = .GlobalEnv)) {
  d_AUC <- tibble::tibble(
    model = character(),
    mn_train = numeric(),
    sd_train = numeric(),
    mn_test = numeric(),
    sd_test = numeric(),
    delta_tt = numeric()
  )
  assign("d_AUC", d_AUC, envir = .GlobalEnv)
}

# Append summary stats to global d_AUC
d_AUC <-<- dplyr::bind_rows(
  get("d_AUC", envir = .GlobalEnv),
  tibble::tibble(
    model = model_desc,
    mn_train = mean(auc_train),
    sd_train = sd(auc_train),
    mn_test = mean(auc_test),
    sd_test = sd(auc_test),
    delta_tt = mean(auc_train) - mean(auc_test)
  )
)

invisible(d_AUC)
}

```

The problem with this approach is that the both functions to fit the model and functions to calculate predictions need special parameters and functions to be passed. In R we can only once pass the ... placeholder. Therefore we will need to split this function in 2 parts:

- first fit the models on the folds
- then calculate the performance parameter of choice (AUC) and add the results to our results tibble `d_AUC`. Here is how this can be done:

```

library(dplyr)
library(purrr)
library(pROC)
library(tibble)

# Function 1: Fit models on folds - returns folds with a new 'model' column
fit_models_on_folds <- function(folds, model_func, model_frm, ...) {
  folds %>%
    mutate(
      model = map(train, function(df) {
        model_func(data = df, formula = model_frm, ...)
      })
    )
}

# Function 2: Calculate AUC on train/test, update global d_AUC, returns updated d_AUC
calculate_and_update_auc <- function(folds, model_desc,
                                     train_pred_fun = function(model, data) predict(model, newdata = data),
                                     test_pred_fun = function(model, data) predict(model, newdata = data)) {

  auc_results <- folds %>%
    mutate(
      auc_train = map2_dbl(model, train, ~ {
        train_data <- as_tibble(.y)
        preds_train <- train_pred_fun(.x, train_data)
        pROC::auc(train_data$isGood, preds_train)
      }),
      auc_test = map2_dbl(model, test, ~ {
        test_data <- as_tibble(.y)
        preds_test <- test_pred_fun(.x, test_data)
        pROC::auc(test_data$isGood, preds_test)
      })
    ) %>%
    select(.id, auc_train, auc_test)

  # Initialize global d_AUC if missing
  if(!exists("d_AUC", envir = .GlobalEnv)) {
    assign("d_AUC", tibble(
      model = character(),
      mn_train = numeric(),
      sd_train = numeric(),
      mn_test = numeric(),
      sd_test = numeric(),
      delta_tt = numeric()
    ), envir = .GlobalEnv)
  }

  # Append summary to global d_AUC
  d_AUC <-> bind_rows(
    get("d_AUC", envir = .GlobalEnv),
    tibble(
      model = model_desc,
      mn_train = mean(auc_results$auc_train),
      sd_train = sd(auc_results$auc_train),

```

```

      mn_test = mean(auc_results$auc_test),
      sd_test = sd(auc_results$auc_test),
      delta_tt = mean(auc_results$auc_train) - mean(auc_results$auc_test)
    )
  )

invisible(d_AUC)
}

```

For the linear regression, it will work very simple:

```

# Suppose 'folds' is prepared with columns train and test (data subsets for each fold)
folds_with_models <- fit_models_on_folds(folds_norm, glm, frm1, family = binomial())

```

```

# AUC calculation and global update
d_AUC <- calculate_and_update_auc(folds_with_models, "Logistic Regression test")
print(d_AUC)

```

```

## # A tibble: 1 x 6
##   model                mn_train sd_train mn_test sd_test delta_tt
##   <chr>                <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 Logistic Regression test    0.807 0.00397 0.803 0.0163 0.00421

```

For a decision tree, we need some more work:

```
library(rpart)
```

```

# Define prediction functions for rpart to get positive class probability
rpart_pred_fun <- function(model, data) {
  preds <- predict(model, newdata = data, type = "prob")
  preds[, 2] # Assuming 2nd column is positive class probability
}

```

```

folds_with_models <- fit_models_on_folds(folds_norm, rpart, frm1, method = "class", cp = 0.01)

```

```

d_AUC <- calculate_and_update_auc(
  folds_with_models,
  "Decision Tree (rpart) test",
  train_pred_fun = rpart_pred_fun,
  test_pred_fun = rpart_pred_fun
)
print(d_AUC)

```

```

## # A tibble: 2 x 6
##   model                mn_train sd_train mn_test sd_test delta_tt
##   <chr>                <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 Logistic Regression test    0.807 0.00397 0.803 0.0163 0.00421
## 2 Decision Tree (rpart) test 0.669 0.00403 0.666 0.0120 0.00275

```

4.2.3 Cross-Validation in the tidyverse

In the next code block we demonstrate how the k-fold cross validation can be done in the `tidyverse` and immediately wrap it in a re-usable function. But before that we create a data-frame to keep track of the values of the AUC.

```
f_add_kfold_auc <- function(data, folds, model_func, model_frm, model_desc, ...) {
```

```

auc_per_fold <- folds %>%
  mutate(
    model = map(train, function(df) {
      model_func(data = df, formula = model_frm, ...)
    }),

    auc_train = map_dbl(model, ~ {
      train_data <- as_tibble(.x$model)
      preds_train <- predict(.x, newdata = train_data, type = "response")
      pROC::auc(train_data$isGood, preds_train)
    }),

    auc_test = map2_dbl(model, test, ~ {
      test_data <- as_tibble(.y)
      preds_test <- predict(.x, newdata = test_data, type = "response")
      pROC::auc(test_data$isGood, preds_test)
    })
  ) %>%
  select(.id, auc_train, auc_test)

# Append results to global d_AUC as before
d_AUC <-<- bind_rows(
  get("d_AUC", envir = .GlobalEnv),
  tibble(
    model = model_desc,
    mn_train = mean(auc_per_fold$auc_train),
    sd_train = sd(auc_per_fold$auc_train),
    mn_test = mean(auc_per_fold$auc_test),
    sd_test = sd(auc_per_fold$auc_test),
    delta_tt = mean(auc_per_fold$auc_train) - mean(auc_per_fold$auc_test)
  )
)

invisible(d_AUC)
}

# Use the function for the logistic regression:
f_add_kfold_auc(data = d$fact$train,
  folds = folds_fact,
  model_func = glm,
  model_frm = frm1,
  model_desc = 'm1 - logistic regression',
  family = binomial() # pass as function call, not string
)

# Add the model with all variables too as m0
f_add_kfold_auc(data = d$fact$train,
  folds = folds_fact,
  model_func = glm,
  model_frm = formula(isGood ~ .),
  model_desc = 'm0 - logistic regr. all vars',
  family = binomial() # pass as function call, not string
)

```

```
# Demonstrate the content of d_AUC
knitr::kable(d_AUC)
```

model	mn_train	sd_train	mn_test	sd_test	delta_tt
Logistic Regression test	0.8070756	0.0039699	0.8028690	0.0162801	0.0042066
Decision Tree (rpart) test	0.6689150	0.0040280	0.6661695	0.0120337	0.0027455
m1 - logistic regression	0.8209545	0.0039059	0.8145140	0.0150951	0.0064404
m0 - logistic regr. all vars	0.8304694	0.0038783	0.8222518	0.0153526	0.0082176

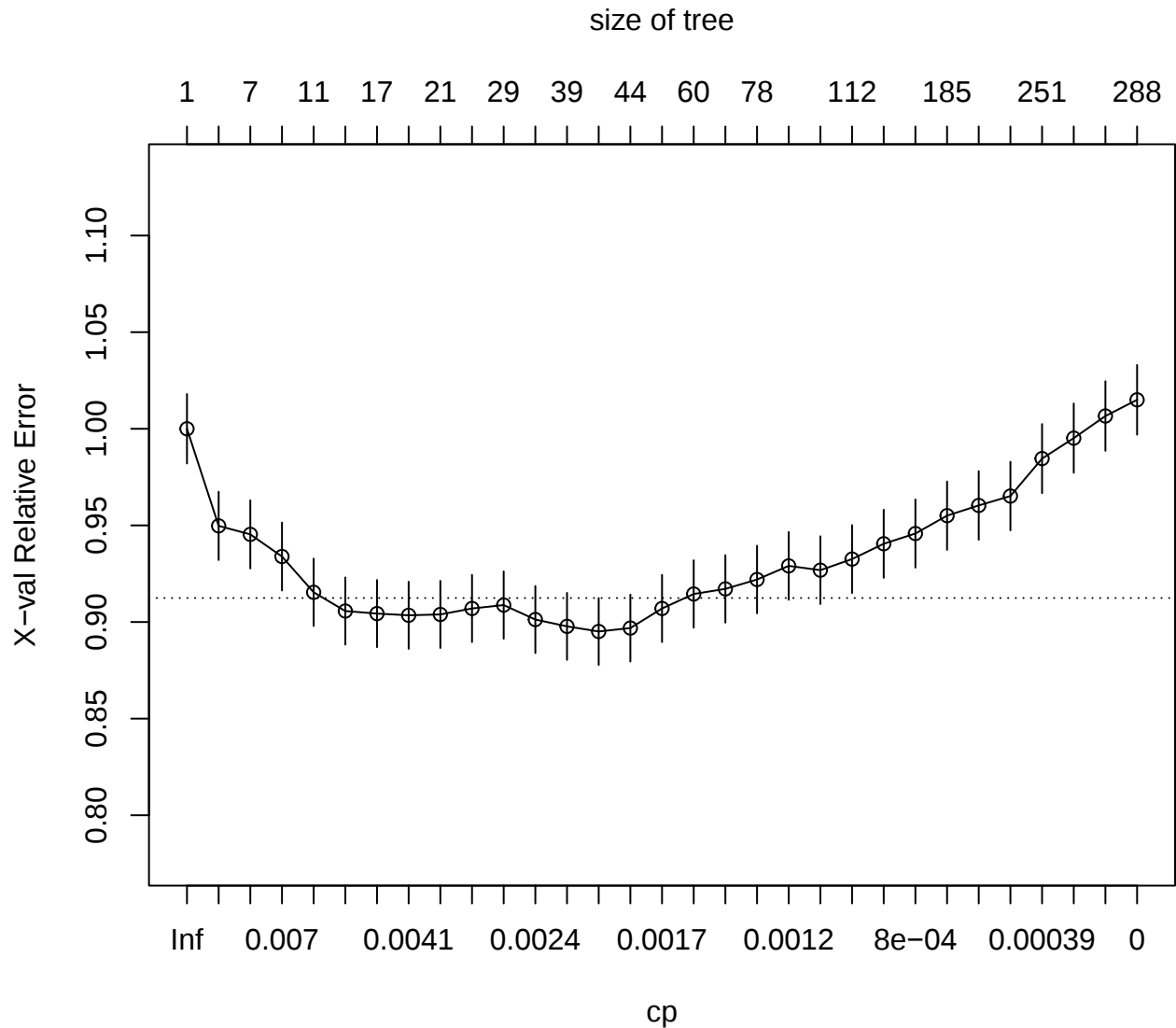
5 Decision Tree

5.1 Decision Tree on binned data

```
library(rpart)
library(rpart.plot)

# Fit a large initial tree with minimal pruning (set cp = 0)
large_tree <- rpart(isGood ~ ., data = d$fact$train, method = "class",
                    control = rpart.control(cp = 0))

# Plot cross-validation results to see optimal cp
plotcp(large_tree) # shows xerror, xstd, and cp values
```



```
# Print the cross-validation table
printcp(large_tree)

##
## Classification tree:
## rpart(formula = isGood ~ ., data = d$fact$train, method = "class",
##       control = rpart.control(cp = 0))
##
## Variables actually used in tree construction:
## [1] AGE      BLUEBOOK  CAR_AGE   CAR_TYPE  CAR_USE   CLM_FREQ
## [7] EDUCATION GENDER    GENDER.AGE HOME_VAL  HOMEKIDS  INCOME
## [13] KIDSDRIV MSTATUS   MVR_PTS  OCCUPATION OLDCLAIM  PARENT1
## [19] REVOKED  TIF       TRAVTIME URBANICITY YOJ
##
## Root node error: 2269/8474 = 0.26776
##
## n= 8474
##
##          CP nsplit rel error  xerror    xstd
```

```
## 1 0.01615984      0 1.00000 1.00000 0.017964
## 2 0.01101807      5 0.90877 0.94976 0.017667
## 3 0.00837373      6 0.89775 0.94535 0.017640
## 4 0.00587630      7 0.88938 0.93389 0.017569
## 5 0.00528867     10 0.87175 0.91538 0.017451
## 6 0.00440723     14 0.85059 0.90569 0.017388
## 7 0.00418687     16 0.84178 0.90436 0.017380
## 8 0.00396651     18 0.83341 0.90348 0.017374
## 9 0.00374614     20 0.82547 0.90392 0.017377
## 10 0.00286470     22 0.81798 0.90701 0.017397
## 11 0.00264434     28 0.80079 0.90877 0.017409
## 12 0.00220361     34 0.78361 0.90128 0.017360
## 13 0.00205671     38 0.77479 0.89775 0.017336
## 14 0.00198325     41 0.76862 0.89511 0.017319
## 15 0.00176289     43 0.76465 0.89687 0.017331
## 16 0.00165271     55 0.74350 0.90701 0.017397
## 17 0.00154253     59 0.73689 0.91450 0.017446
## 18 0.00143235     70 0.71926 0.91714 0.017463
## 19 0.00132217     77 0.70428 0.92199 0.017494
## 20 0.00117526     97 0.67607 0.92904 0.017538
## 21 0.00110181    103 0.66902 0.92684 0.017524
## 22 0.00102835    111 0.66020 0.93257 0.017561
## 23 0.00088145    127 0.63993 0.94050 0.017610
## 24 0.00073454    178 0.59189 0.94579 0.017643
## 25 0.00066108    184 0.58748 0.95505 0.017700
## 26 0.00058763    192 0.58219 0.96033 0.017732
## 27 0.00044072    201 0.57691 0.96518 0.017761
## 28 0.00035258    250 0.55531 0.98457 0.017875
## 29 0.00022036    255 0.55355 0.99515 0.017937
## 30 0.00014691    273 0.54958 1.00661 0.018002
## 31 0.00000000    287 0.54738 1.01498 0.018049
```

```
# Find the cp with minimum xerror + 1sd for pruning
best_cp <- large_tree$cptable[which.min(large_tree$cptable[, "xerror"]), "CP"]
# Alternatively, choose cp at the "1-SE rule"
# best_cp <- large_tree$cptable[which.min(large_tree$cptable[, "xerror"] + large_tree$cptable[, "xstd"]),

# Prune the tree with the selected cp
pruned_tree <- prune(large_tree, cp = best_cp)

# Plot the pruned tree
rpart.plot(pruned_tree, main = "Pruned Decision Tree")
```

— Page 44 —
20



```
frm0 <- isGood ~ .
```

```
results <- folds_fact %>%
```

```
model = map(train, ~ rpart(frm0, data = as_tibble(.), method = "class", cp = 0.01)),
auc_train = map2_dbl(model, train, ~ {
  train_data <- as_tibble(.y) # Use train fold data, not .x$model
  preds_train <- predict(.x, newdata = train_data, type = "prob")[, 2]
  pROC::auc(train_data$IsGood, preds_train)
}),
auc_test = map2_dbl(model, test, ~ {
  test_data <- as_tibble(.y)
  preds_test <- predict(.x, newdata = test_data, type = "prob")[, 2]
  pROC::auc(test_data$IsGood, preds_test)
})
%>%
select(.id, auc_train, auc_test)
```

```

print(results)

## # A tibble: 5 x 3
##   .id   auc_train auc_test
##   <chr>     <dbl>   <dbl>
## 1 1         0.665     0.675
## 2 2         0.676     0.656
## 3 3         0.672     0.672
## 4 4         0.664     0.681
## 5 5         0.677     0.653

# Append summary stats to global d_AUC
d_AUC <- d_AUC %>%
  add_row(
    model = "Decision Tree (rpart)",
    mn_train = mean(results$auc_train),
    sd_train = sd(results$auc_train),
    mn_test  = mean(results$auc_test),
    sd_test  = sd(results$auc_test),
    delta_tt = mean(results$auc_train) - mean(results$auc_test)
  )

```

5.2 Decision Tree on numerical data

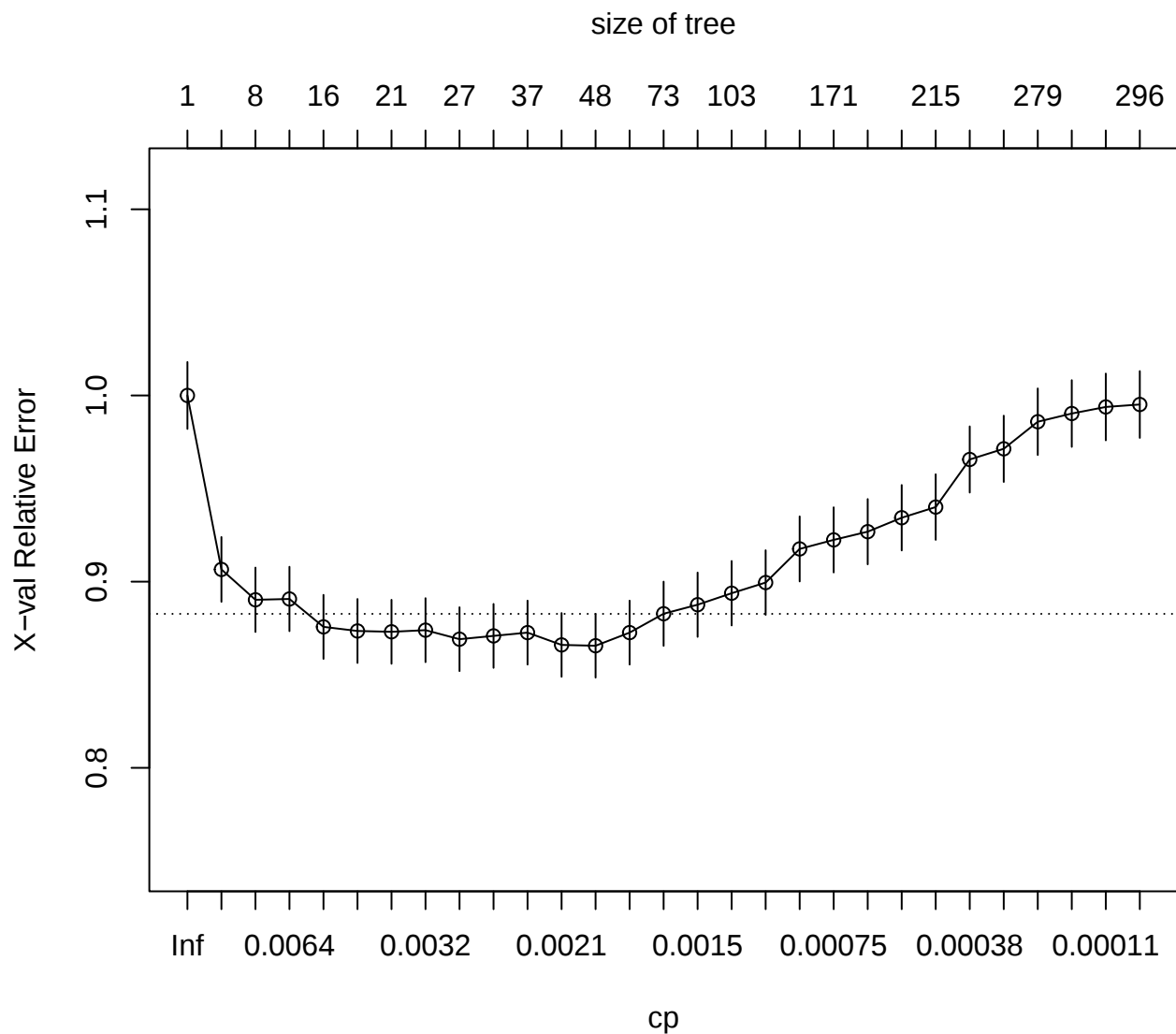
```

library(rpart)
library(rpart.plot)

# Fit a large initial tree with minimal pruning (set cp = 0)
large_tree <- rpart(isGood ~ ., data = d$norm$train, method = "class",
  control = rpart.control(cp = 0))

# Plot cross-validation results to see optimal cp
plotcp(large_tree) # shows xerror, xstd, and cp values

```



```
# Print the cross-validation table
printcp(large_tree)

##
## Classification tree:
## rpart(formula = isGood ~ ., data = d$norm$train, method = "class",
##       control = rpart.control(cp = 0))
##
## Variables actually used in tree construction:
## [1] AGE      BLUEBOOK  CAR_AGE   CAR_TYPE  CAR_USE   EDUCATION
## [7] GENDER   HOME_VAL  HOMEKIDS  INCOME    KIDSDRIV  MSTATUS
## [13] MVR_PTS  OCCUPATION OLDCLAIM  PARENT1   RED_CAR   REVOKED
## [19] TIF      TRAVTIME  URBANICITY YOJ
##
## Root node error: 2269/8474 = 0.26776
##
## n= 8474
##
##          CP nsplit rel error  xerror    xstd
```

```
## 1 1.4397e-02      0 1.00000 1.00000 0.017964
## 2 1.0577e-02      6 0.88717 0.90657 0.017394
## 3 7.0516e-03      7 0.87660 0.89026 0.017287
## 4 5.7294e-03      8 0.86955 0.89070 0.017290
## 5 3.9665e-03     15 0.82812 0.87572 0.017189
## 6 3.7461e-03     18 0.81622 0.87351 0.017174
## 7 3.3054e-03     20 0.80873 0.87307 0.017171
## 8 3.0851e-03     24 0.79550 0.87395 0.017177
## 9 2.6443e-03     26 0.78933 0.86911 0.017143
## 10 2.4240e-03    33 0.76994 0.87087 0.017156
## 11 2.2036e-03    36 0.76157 0.87263 0.017168
## 12 2.0567e-03    38 0.75716 0.86602 0.017122
## 13 1.9833e-03    47 0.73513 0.86558 0.017119
## 14 1.7629e-03    59 0.70912 0.87263 0.017168
## 15 1.6160e-03    72 0.68532 0.88277 0.017236
## 16 1.3222e-03    75 0.68048 0.88762 0.017269
## 17 1.1018e-03   102 0.64301 0.89379 0.017310
## 18 8.8145e-04   108 0.63640 0.89952 0.017348
## 19 7.7126e-04   165 0.57955 0.91758 0.017465
## 20 7.3454e-04   170 0.57514 0.92243 0.017496
## 21 6.6108e-04   181 0.56324 0.92684 0.017524
## 22 5.8763e-04   193 0.55531 0.93433 0.017572
## 23 4.4072e-04   214 0.54165 0.94006 0.017607
## 24 3.3054e-04   264 0.51609 0.96562 0.017763
## 25 2.2036e-04   268 0.51476 0.97135 0.017798
## 26 1.7629e-04   278 0.51256 0.98590 0.017883
## 27 1.2592e-04   283 0.51168 0.99030 0.017909
## 28 8.8145e-05   290 0.51080 0.99383 0.017929
## 29 0.0000e+00   295 0.51036 0.99515 0.017937
```

```
# Find the cp with minimum xerror + 1sd for pruning
```

```
best_cp <- large_tree$cptable[which.min(large_tree$cptable[, "xerror"]), "CP"]
```

```
# Alternatively, choose cp at the "1-SE rule"
```

```
# best_cp <- large_tree$cptable[which.min(large_tree$cptable[, "xerror"] + large_tree$cptable[, "xstd"]),
```

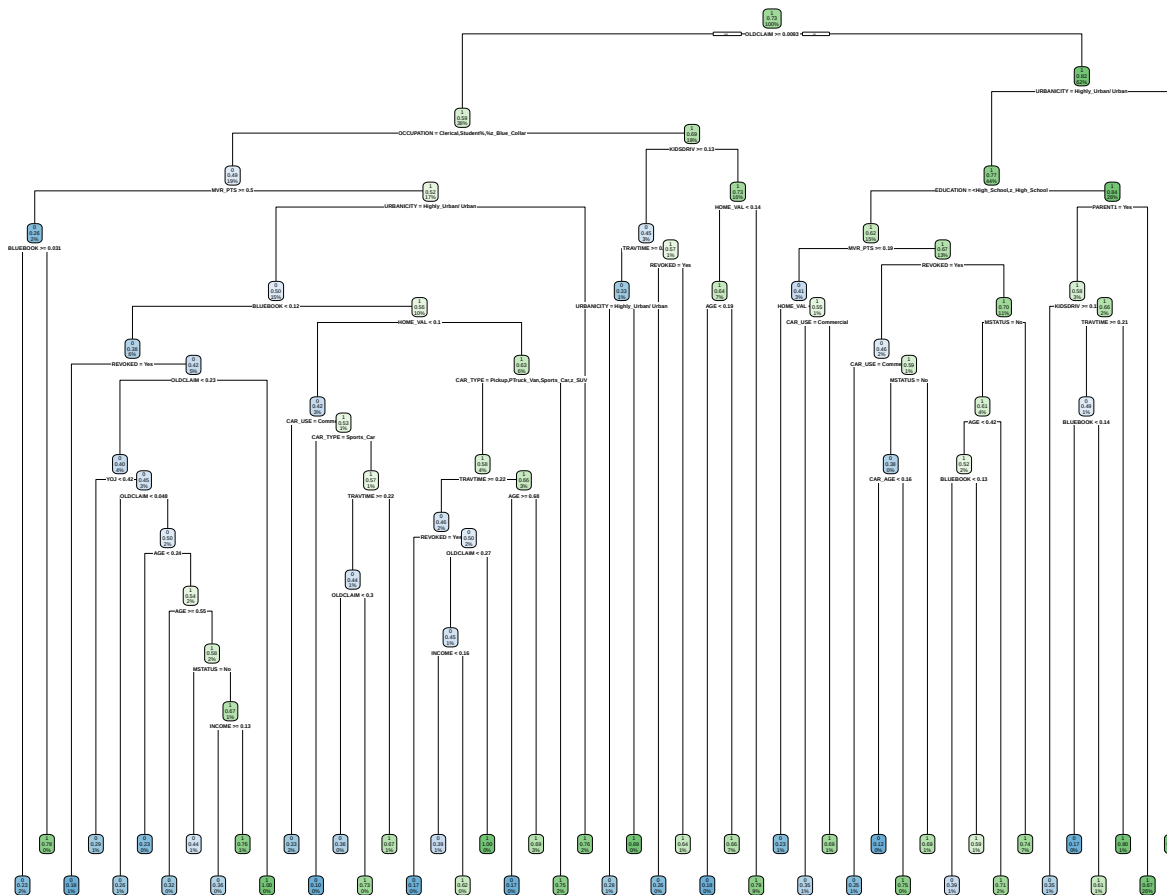
```
# Prune the tree with the selected cp
```

```
pruned_tree <- prune(large_tree, cp = best_cp)
```

```
# Plot the pruned tree
```

```
rpart.plot(pruned_tree, main = "Pruned Decision Tree")
```

Pruned Decision Tree



The cross validation for the tree:

```
# Wrapper for rpart fitting to match f_add_kfold_auc interface
rpart_model_func <- function(data, formula, ...) {
  rpart::rpart(formula, data = as_tibble(data), method = "class", ...)
}

frm0 <- isGood ~ .

###
results <- folds_norm %>%
  mutate(
    model = map(train, ~ rpart(frm0, data = as_tibble(.), method = "class", cp = 0.01)),
    auc_train = map2_dbl(model, train, ~ {
      train_data <- as_tibble(.y) # Use train fold data, not .x$model
      preds_train <- predict(.x, newdata = train_data, type = "prob")[, 2]
      pROC::auc(train_data$isGood, preds_train)
    }),
    auc_test = map2_dbl(model, test, ~ {
      test_data <- as_tibble(.y)
      preds_test <- predict(.x, newdata = test_data, type = "prob")[, 2]
      pROC::auc(test_data$isGood, preds_test)
    })
  )
```

```

    })
  ) %>%
  select(.id, auc_train, auc_test)

print(results)

## # A tibble: 5 x 3
##   .id  auc_train auc_test
##   <chr>    <dbl>   <dbl>
## 1 1      0.666    0.678
## 2 2      0.677    0.662
## 3 3      0.673    0.672
## 4 4      0.664    0.677
## 5 5      0.678    0.655

# Append summary stats to global d_AUC
d_AUC <- d_AUC %>%
  add_row(
    model = "Decision Tree (num)",
    mn_train = mean(results$auc_train),
    sd_train = sd(results$auc_train),
    mn_test = mean(results$auc_test),
    sd_test = sd(results$auc_test),
    delta_tt = mean(results$auc_train) - mean(results$auc_test)
  )

```

6 Random Forest

The function `randomForest()` requires a `data.frame` as input, this makes everything a little different - so we perform the cross validation manually (without our custom function)

```

library(dplyr)
library(randomForest)
library(pROC)

k <- nrow(folds_norm)
auc_train <- numeric(k)
auc_test <- numeric(k)

for (i in 1:k) {

  train_data <- as.data.frame(folds_norm$train[[i]])
  test_data <- as.data.frame(folds_norm$test[[i]])

  # convert the numeric isGood to factor
  train_data$isGood <- as.factor(train_data$isGood)
  test_data$isGood <- as.factor(test_data$isGood)

  # Fit Random Forest model
  model_rf <- randomForest(isGood ~ .,
                           data = train_data,
                           ntree = 200,
                           mtry = floor(sqrt(ncol(train_data) - 1)))

```

```

# Predictions on train
preds_train <- predict(model_rf, newdata = train_data, type = "prob")[, 2]
auc_train[i] <- pROC::auc(train_data$isGood, preds_train)

# Predictions on test
preds_test <- predict(model_rf, newdata = test_data, type = "prob")[, 2]
auc_test[i] <- pROC::auc(test_data$isGood, preds_test)
}

# Append summary stats to global d_AUC tibble
d_AUC <- d_AUC %>%
  add_row(
    model = "Random Forest (ntree=200, mtry=sqrt)",
    mn_train = mean(auc_train),
    sd_train = sd(auc_train),
    mn_test = mean(auc_test),
    sd_test = sd(auc_test),
    delta_tt = mean(auc_train) - mean(auc_test)
  )

```

We notice here that the model fits perfectly the training data, but looses on the test-data. A training AUC of 1 (with zero std) but a lower test AUC (~0.85) means the model perfectly fits training data but less so unseen data—typical **overfitting**.

We can reduce the over-fitting with `randomForest()` by:

- **Reduce number of trees:**
ntree = 100
- **Limit tree size:** use maximum nodes or minimum node size
maxnodes = 30
nodesize = 10
- **Adjust features sampled at each split (mtry):**
Increase or tune with `mtry = floor(ncol(train_data)/2)` or use `tuneRF()` to find best value
- **Feature selection or dimensionality reduction:**
Preprocess data before modelling (e.g., PCA)

```

for (i in 1:k) {

  train_data <- as.data.frame(folds_norm$train[[i]])
  test_data <- as.data.frame(folds_norm$test[[i]])

  # convert the numeric isGood to factor
  train_data$isGood <- as.factor(train_data$isGood)
  test_data$isGood <- as.factor(test_data$isGood)

  # Fit Random Forest model
  model_rf <- randomForest(isGood ~ ., data = train_data,
                           ntree = 150,
                           mtry = floor(sqrt(ncol(train_data) - 1)/2),
                           maxnodes = 30,
                           nodesize = 10
                           )

  # Predictions on train

```

```

preds_train <- predict(model_rf, newdata = train_data, type = "prob")[, 2]
auc_train[i] <- pROC::auc(train_data$isGood, preds_train)

# Predictions on test
preds_test <- predict(model_rf, newdata = test_data, type = "prob")[, 2]
auc_test[i] <- pROC::auc(test_data$isGood, preds_test)
}

# Append summary stats to global d_AUC tibble
d_AUC <- d_AUC %>%
  add_row(
    model = "Random Forest (ntree=150, mtry=sqrt(2))",
    mn_train = mean(auc_train),
    sd_train = sd(auc_train),
    mn_test = mean(auc_test),
    sd_test = sd(auc_test),
    delta_tt = mean(auc_train) - mean(auc_test)
  )

```

6.1 Random Forest on the binned data

We fit now the random forest on the data that is completely categorical (so also the numerical variables have been converted to multiple categories – binned data).

```

# The only difference is that we use folds_fact
for (i in 1:k) {

  train_data <- as.data.frame(folds_fact$train[[i]])
  test_data <- as.data.frame(folds_fact$test[[i]])

  # convert the numeric isGood to factor
  train_data$isGood <- as.factor(train_data$isGood)
  test_data$isGood <- as.factor(test_data$isGood)

  # Fit Random Forest model
  model_rf <- randomForest(isGood ~ ., data = train_data,
                           ntree = 150,
                           mtry = floor(sqrt(ncol(train_data) - 1)),
                           maxnodes = 60,
                           nodesize = 10
                           )

  # Predictions on train
  preds_train <- predict(model_rf, newdata = train_data, type = "prob")[, 2]
  auc_train[i] <- pROC::auc(train_data$isGood, preds_train)

  # Predictions on test
  preds_test <- predict(model_rf, newdata = test_data, type = "prob")[, 2]
  auc_test[i] <- pROC::auc(test_data$isGood, preds_test)
}

# Append summary stats to global d_AUC tibble
d_AUC <- d_AUC %>%
  add_row(
    model = "Random Forest on fact data (150/sqrt)",

```

```

mn_train = mean(auc_train),
sd_train = sd(auc_train),
mn_test  = mean(auc_test),
sd_test  = sd(auc_test),
delta_tt = mean(auc_train) - mean(auc_test)
)

```

7 A Neural Network

7.1 Neural networks in R

When optimizing neural networks in R, several package options exist, each with advantages and disadvantages:

Package	Key Features	Advantages	Disadvantages
nnet	Classic single-hidden-layer MLP	Built-in R package, simple, fast	Only one hidden layer
neuralnet	Flexible multiple hidden layers	Easy to use, visualize, customize	Slower, limited optimization methods
RSNNS	Interface to Stuttgart Neural Network Simulator	Powerful, supports complex nets	Steeper learning curve, less tidy
keras	Deep learning via TensorFlow backend	Supports deep architectures, GPU acceleration, large community	More complex setup, heavier dependency
caret / tidymodels	Wrapper interfaces to many NN packages	Unified interface, cross-validation, tuning tools	Overhead, may be less flexible for custom nets

For simple networks, **nnet** or **neuralnet** work well; for deeper/more complex models, **keras** is preferable. Wrappers like **caret**/**tidymodels** offer convenience but less direct control.

Choose based on your architecture needs, compute resources, and familiarity.

7.2 The package **nnet**

```

library(dplyr)
library(nnet)
library(pROC)
library(tibble)

k <- nrow(folds_num)
auc_train <- numeric(k)
auc_test  <- numeric(k)

for (i in 1:k) {
  train_data <- as.data.frame(folds_num$train[[i]])
  test_data  <- as.data.frame(folds_num$test[[i]])

  # For nnet, target needs to be numeric (0/1)
  train_data$isGood <- as.numeric(as.character(train_data$isGood))
  test_data$isGood  <- as.numeric(as.character(test_data$isGood))

  # Fit neural network model (size = number of hidden units)
  model_nn <- nnet(isGood ~ ., data = train_data, size = 12, maxit = 250, trace = FALSE)

```

```

# Predictions on train (type="raw" gives numeric output)
preds_train <- predict(model_nn, newdata = train_data, type = "raw")
auc_train[i] <- pROC::auc(train_data$isGood, preds_train)

# Predictions on test
preds_test <- predict(model_nn, newdata = test_data, type = "raw")
auc_test[i] <- pROC::auc(test_data$isGood, preds_test)
}

# Append summary stats to global d_AUC tibble
d_AUC <- d_AUC %>%
  add_row(
    model = "aNN (nnet, size=12, maxit=250)",
    mn_train = mean(auc_train),
    sd_train = sd(auc_train),
    mn_test = mean(auc_test),
    sd_test = sd(auc_test),
    delta_tt = mean(auc_train) - mean(auc_test)
  )

```

The neural network can be visualised as follows:

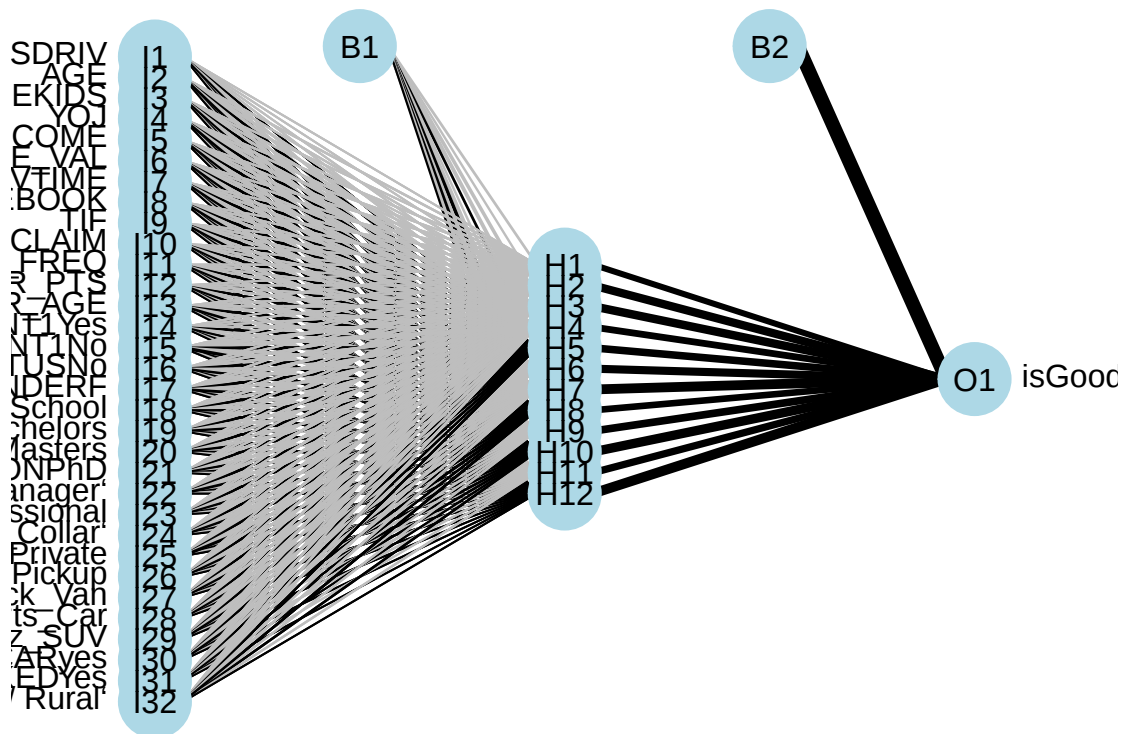
```
library(NeuralNetTools)
```

```

# After fitting your model_nn (one of the folds or final model), for example:
# model_nn <- nnet(isGood ~ ., data = train_data, size = 6, maxit = 250, trace = FALSE)

# Visualize the network structure
plotnet(model_nn)

```



7.3 The package neuralnet

```
library(dplyr)
library(neuralnet)
library(pROC)
library(tibble)

k <- nrow(folds_num)
auc_train <- numeric(k)
auc_test <- numeric(k)

for (i in 1:k) {

  # the function neuralnet will not work with a fold:
  train_data <- as.data.frame(folds_num$train[[i]])
  test_data <- as.data.frame(folds_num$test[[i]])

  # Ensure target variable is numeric 0/1 for neuralnet
  train_data$isGood <- as.numeric(as.character(train_data$isGood))
  test_data$isGood <- as.numeric(as.character(test_data$isGood))

  # replace all special characters in the column names with a .
  names(train_data) <- make.names(names(train_data), unique = TRUE)
  names(test_data) <- make.names(names(test_data), unique = TRUE)

  # this way we can make all a formula for all x variables:
  all_vars <- setdiff(names(train_data), "isGood")
  model_frm <- as.formula(paste("isGood ~", paste(all_vars, collapse = " + ")))

  # but we will make a shorter one only including variable similar to the choice in frm1
  model_frm <- isGood ~ MSTATUSNo + EDUCATIONz_High_School + EDUCATIONBachelors + EDUCATIONMasters + E

  # Fit neural network with two hidden layers: 6 and 3 neurons
  model_nn <- neuralnet(model_frm,
                        data = train_data,
                        hidden = c(4,2), # hidden layers
                        linear.output = FALSE,
                        stepmax = 1e6)

  # Prediction function for neuralnet: compute on feature columns only
  preds_train <- neuralnet::compute(model_nn, train_data[, setdiff(names(train_data), "isGood")])$net.result
  auc_train[i] <- pROC::auc(train_data$isGood, preds_train)

  preds_test <- compute(model_nn, test_data[, setdiff(names(test_data), "isGood")])$net.result[,1]
  auc_test[i] <- pROC::auc(test_data$isGood, preds_test)
}

# Save the results of the
saveRDS(auc_train, './auc_train.R')
saveRDS(auc_test, './auc_test.R')
saveRDS(model_nn, './model_nn.R')
```

```

# read the previously saved results to save time:
# COMMENT OUT IF PREVIOUS CHUNCK IS eval=TRUE
auc_train <- readRDS('./auc_train.R')
auc_test  <- readRDS('./auc_test.R')
model_nn  <- readRDS('./model_nn.R')

# Append summary stats to global d_AUC tibble
d_AUC <- d_AUC %>%
  add_row(
    model = "aNN (neuralnet, hidden=c(4,2))",
    mn_train = mean(auc_train),
    sd_train = sd(auc_train),
    mn_test  = mean(auc_test),
    sd_test  = sd(auc_test),
    delta_tt = mean(auc_train) - mean(auc_test)
  )

```

The model can be visualised by the library ‘neuralnet’ as follows (or we can use the `NeuralNetTools` library)

```

# We visualise the last NN that was fitted
plot(model_nn)

```

8 Support Vector Machine (SVM)

The function `svm()` from `e1071` is a standard choice in R. However, it does not allow arguments

```

library(e1071)
library(pROC)
library(dplyr)

```

```

fit_models_on_folds_svm <- function(folds, model_func, model_frm, ...) {
  folds %>%
    mutate(
      model = map(train, function(df) {
        model_func(model_frm, df, ...) # formula first, data second, no names
      })
    )
}

frm0 <- isGood ~ .
# Fit SVM models on the folds with radial kernel
models_f <- fit_models_on_folds_svm(folds = folds_bin,
                                   model_func = e1071::svm,
                                   model_frm = frm0,
                                   kernel = "radial", cost = 1)

# Calculate and update AUC stats globally
d_AUC <- calculate_and_update_auc(models_f, model_desc = "SVM radial kernel, cost=1")

```

9 PCA

Principal Component Analysis (PCA) is a statistical technique used to reduce the dimensionality of large datasets by transforming correlated variables into a set of uncorrelated principal components. In R, PCA can be performed using the `prcomp()` function, which centres and scales the data by default, making it easier to

interpret the results. The principal components are ordered by the amount of variance they explain, with the first few components typically capturing most of the information in the original dataset. This method is widely used for data visualization, noise reduction, and feature extraction in multivariate analysis.

```
library(dplyr)
library(purrr)
library(pROC)
library(tibble)
library(modelr) # for crossv_kfold and resample objects

# Function to run PCA on train and apply on test data
run_pca_and_transform <- function(train_df, test_df, ncomp = 35) {
  # Select features only (exclude target)
  train_x <- train_df %>% select(-isGood)

  # PCA on train data
  pca <- prcomp(train_x, center = TRUE, scale. = TRUE)

  # Project train and test into PCA space
  train_pca <- as_tibble(predict(pca, train_x)[, 1:ncomp])
  test_x <- test_df %>% select(-isGood)
  test_pca <- as_tibble(predict(pca, test_x)[, 1:ncomp])

  # Add target variable back
  train_pca$isGood <- train_df$isGood
  test_pca$isGood <- test_df$isGood

  list(train = train_pca, test = test_pca)
}

# reminder: Create 5 folds with crossv_kfold (from modelr)
#folds_bin <- crossv_kfold(mydata, k = 5)

# Extract train/test data frames properly and run PCA transform on each fold
folds_pca <- map2(
  folds_bin$train,
  folds_bin$test,
  ~ run_pca_and_transform(
    as.data.frame(.x), # train data frame from resample object
    as.data.frame(.y), # test data frame from resample object
    ncomp = 35
  )
)

# Assemble tibble of folds with transformed data
folds_pca_tibble <- tibble(
  .id = seq_along(folds_pca),
  train = map(folds_pca, "train"),
  test = map(folds_pca, "test")
)

# Logistic regression fitting function
logistic_model_func <- function(data, formula, ...) {
  glm(formula, data = data, family = binomial, ...)
```

```

}

# Model formula using all PCA components
formula_pca <- as.formula("isGood ~ .")

# Fit logistic regression models on PCA-transformed folds
folds_with_models <- fit_models_on_folds(folds_pca_tibble, logistic_model_func, formula_pca)

# Calculate and update AUC scores for train and test sets
calculate_and_update_auc(folds_with_models, "Logistic regression PCA (first 35 components)")

# Show results
#print(d_AUC)

```

10 The Final Model Choice

10.1 The interpretation of the AUC findings

When comparing multiple predictive models, the Area Under the ROC Curve (AUC) metric is often used to evaluate classification performance (Goodfellow, Bengio, and Courville 2016; Powers 2011).

- The **test data AUC** is the most important indicator since it reflects how well the model generalizes to new, unseen data.
- A model with a **high training AUC but noticeably lower test AUC** is likely overfitting the training data and may not perform well in practice.
- Ideally, choose a model with a **high and similar AUC on both training and test data**, indicating good generalization and fitting.
- Consider also the variability in AUC obtained via cross-validation folds to assess the stability and robustness of the performance.
- For imbalanced datasets, supplement AUC with other metrics or precision-recall curves for a more complete evaluation (Saito and Rehmsmeier 2015).

Criterion	Interpretation	Preferred Model
High test AUC	Good generalization	Higher test AUC
Large gap between train-test	Overfitting	Smaller gap
Similar train & test AUC	Good fit and generalization	Models with minimal difference
Low variability across folds	Stable and reliable performance	Models with consistent AUCs

This approach ensures selecting models that balance accuracy and robustness to new data, avoiding poor generalization (James et al. 2013).

```

library(knitr)
library(kableExtra)

# Continuous color scale for 'test' (4th column), high values better: green high, red low
colors_test <- spec_color(d_AUC$mn_test, option = "D") # viridis default

# Continuous color scale for 'sd_test' (5th column), lower is better so reverse colors
colors_sd_test <- spec_color(d_AUC$sd_test, option = "D", direction = -1)

# Continuous color scale for 'delta' (6th column), lower is better so reverse colors
colors_delta <- spec_color(d_AUC$delta_tt, option = "D", direction = -1)

```

Table 4: A summary of all cross validation results. Best results are in yellow, green is intermediate, purple are bad results.

model	mn_train	sd_train	mn_test	sd_test	delta_tt
Logistic Regression test	0.8070756	0.0039699	0.8028690	0.0162801	0.0042066
Decision Tree (rpart) test	0.6689150	0.0040280	0.6661695	0.0120337	0.0027455
m1 - logistic regression	0.8209545	0.0039059	0.8145140	0.0150951	0.0064404
m0 - logistic regr. all vars	0.8304694	0.0038783	0.8222518	0.0153526	0.0082176
Decision Tree (rpart)	0.6707549	0.0057761	0.6673415	0.0119733	0.0034135
Decision Tree (num)	0.6718906	0.0063668	0.6687007	0.0100486	0.0031900
Random Forest (ntree=200, mtry=sqrt)	1.0000000	0.0000000	0.8158758	0.0124400	0.1841242
Random Forest (ntree=150, mtry=sqrt/2)	0.8156110	0.0022736	0.7852351	0.0185251	0.0303760
Random Forest on fact data (150/sqrt)	0.8376904	0.0042282	0.7922099	0.0148604	0.0454805
aNN (nnet, size=12, maxit=250)	0.5774188	0.1731136	0.5524577	0.1172991	0.0249610
aNN (neuralnet, hidden=c(4,2))	0.8225667	0.0045482	0.7876775	0.0146028	0.0348893
SVM radial kernel, cost=1	0.9375148	0.0018342	0.7914437	0.0098089	0.1460712
Logistic regression PCA (first 35 components)	0.7812688	0.0037161	0.7761019	0.0128948	0.0051669

```
kable(d_AUC, "latex",
      caption = "A summary of all cross validation results.
      Best results are in yellow, green is intermediate, purple are bad results.") %>%
kable_styling() %>%
column_spec(4, background = colors_test, color = "black") %>%
column_spec(5, background = colors_sd_test, color = "black") %>%
column_spec(6, background = colors_delta, color = "black")
```

10.2 Model Selection and Unbiased Performance Estimation

The dataset presents a high-dimensional feature space with numerous independent variables exhibiting multicollinearity and varying predictive strengths, as evidenced by Information Value (IV) analyses and cross-validation outcomes. Among the evaluated candidates—including logistic regressions (m0 with all variables, m1 with refined features), decision trees, random forests, neural networks, SVMs, and PCA-preprocessed variants—the generalized linear model m1 emerges as optimal. This selection adheres to established criteria prioritizing

- i. maximal mean test AUC , ii/ minimal train-test delta , iii low cross-fold standard deviation , and iv/ parsimony to mitigate over-fitting risks inherent to high-dimensionality (Akaike Information Criterion favours fewer parameters without substantial performance loss).

While m0 yields comparable CV performance, its excess parameters elevate over-fitting susceptibility, as indicated by a larger train-test discrepancy and elevated variance.

We choose the model (m1) as the model to be put in production. All we have to do now is retrain it on the complete training data (now the variable m1 refers to models trained on a sub-set of the training data). Then we will still check its performance on the held out testing data.

```
library(pROC)
library(ROCR)

# Retrain final m1 on full training data
m1_final <- glm(frm1, data = d$fact$train, family = binomial)

# Test predictions
preds_test <- predict(m1_final, newdata = d$fact$test, type = "response")
```

```

# AUC and CI
roc_obj <- roc(d$fact$test$isGood, preds_test)
auc_test <- auc(roc_obj)
ci <- ci.auc(roc_obj)
ci_lower <- ci
ci_upper <- ci[2]

# KS statistic
pred_final <- prediction(preds_test, d$fact$test$isGood)
perf_final <- performance(pred_final, "tpr", "fpr")
#ks_test <- max(attr(perf_obj, "y.values")[] - attr(perf_obj, "x.values")[])

# Optimal cutoff (3:1 FP:FN cost ratio)
#cutoff_value <- optimal_cutoff(pred_obj, optimal = "rec", cost.fp = 3, cost.fn = 1)

```

The AUC on the test data is 0.8 – which is a decent result, keeping into mind the data of the table in previous section. The model `m1_final` hence passes this test.

10.3 Optimal Cut-Off

Finally we produce an optimal cut-off and the confusion matrix for the model.

```

# The optimal cutoff if the false positives cost 3 times more than the false negatives:
cutoff_vals <- opt_cut_off(perf_final, pred_final, cost.fp = 3)
cutoff_vals # print the values related to the cutoff

##                [,1]
## sensitivity 0.5017953
## specificity 0.8900524
## cutoff      0.8628670

cutoff_final <- cutoff_vals[3,1]

```

```

# Binary predictions based on that cutoff:
bin_pred_final <- if_else(pred_final@predictions[[1]] > cutoff_final, 1, 0)

```

The chosen cut-off is 0.86. With this choice, we accept around 40.11% of the customers. However, note that the data is re-balanced (probably rows of good customers have been deleted at random). So, it is impossible to answer the question how much percent of customers we actually will reject. In any case this should be much less than this calculation seems to imply.

The package `{scorecard}` provides a function `perf_eva()` to produce the confusion-matrix:

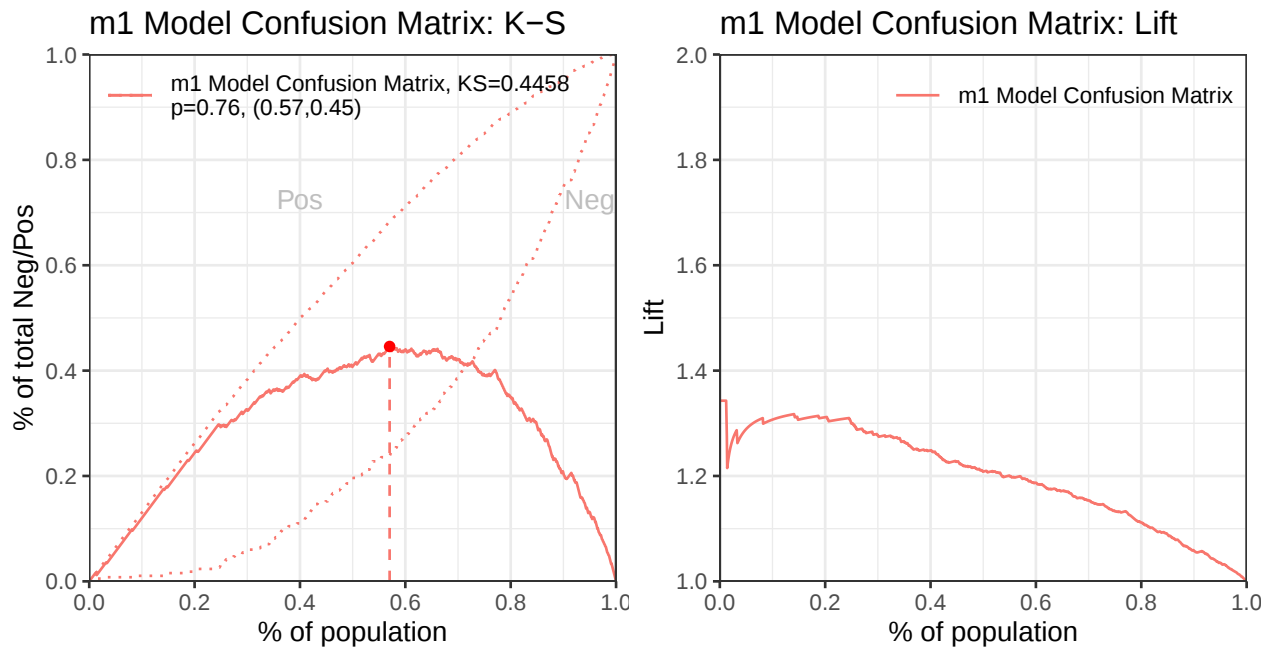
```

# Extract raw vectors (equal length)
scores <- unlist(pred_final@predictions) # Flatten ROCR predictions
labels <- d$fact$test$isGood # 0/1 numeric labels

# Verify lengths match
stopifnot(length(scores) == length(labels))

perf_eva(pred = scores, label = labels,
         title = "m1 Model Confusion Matrix",
         confusion_matrix = TRUE,
         threshold = cutoff_final)

```



```
## $binomial_metric
## $binomial_metric$m1 Model Confusion Matrix`
##      MSE      RMSE   LogLoss      R2      KS      AUC      Gini
##      <num>    <num>    <num>    <num>    <num>    <num>    <num>
## 1: 0.1490937 0.3861265 0.4568426 0.2158957 0.4458016 0.7950501 0.5901003
##
##
## $confusion_matrix
## $confusion_matrix$m1 Model Confusion Matrix`
##   label pred_0 pred_1   error
##   <char> <num> <num>    <num>
## 1:      0    340     42 0.1099476
## 2:      1    555    559 0.4982047
## 3: total    895    601 0.3990642
##
##
## $pic
## TableGrob (1 x 2) "arrange": 2 grobs
##   z      cells  name      grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
```

11 Appendices

11.1 Acknowledgement

In this document we used workflow and code from: De Brouwer, Philippe J. S. 2020. The Big r-Book: From Data Science to Learning Machines and Big Data. New York: John Wiley & Sons, Ltd.

We mainly used in this document:

- Part V (p. 373–562): **Modelling**

11.2 Notes on the templates

RMarkdown is very flexible, you can use any template. For example here are more templates: AGH templates at overleaf

Bibliography

- De Brouwer, Philippe J. S. 2020. *The Big r-Book: From Data Science to Learning Machines and Big Data*. New York: John Wiley & Sons, Ltd.
- Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. “Deep Learning.” 2016. <https://www.deeplearningbook.org/>.
- James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. “An Introduction to Statistical Learning.” 2013. <https://www.statlearning.com/>.
- Powers, David M. W. 2011. “Evaluation: From Precision, Recall and f-Measure to ROC, Informedness, Markedness & Correlation.” *Journal of Machine Learning Technologies* 2 (1): 37–63. <https://doi.org/10.2139/ssrn.2276226>.
- Saito, Takaya, and Marc Rehmsmeier. 2015. “The Precision-Recall Plot Is More Informative Than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets.” *PLoS One* 10 (3): e0118432. <https://doi.org/10.1371/journal.pone.0118432>.
- Smith, John, and Jane Doe. 2023. “Data Splitting Strategies for Statistical Modelling.” *Journal of Data Science* 15 (2): 100–110.
- Taylor, Alice B., and Minh Nguyen. 2024. “Comprehensive Review of Cross-Validation Techniques.” *Journal of Statistical Methods* 7 (1): 210–30.