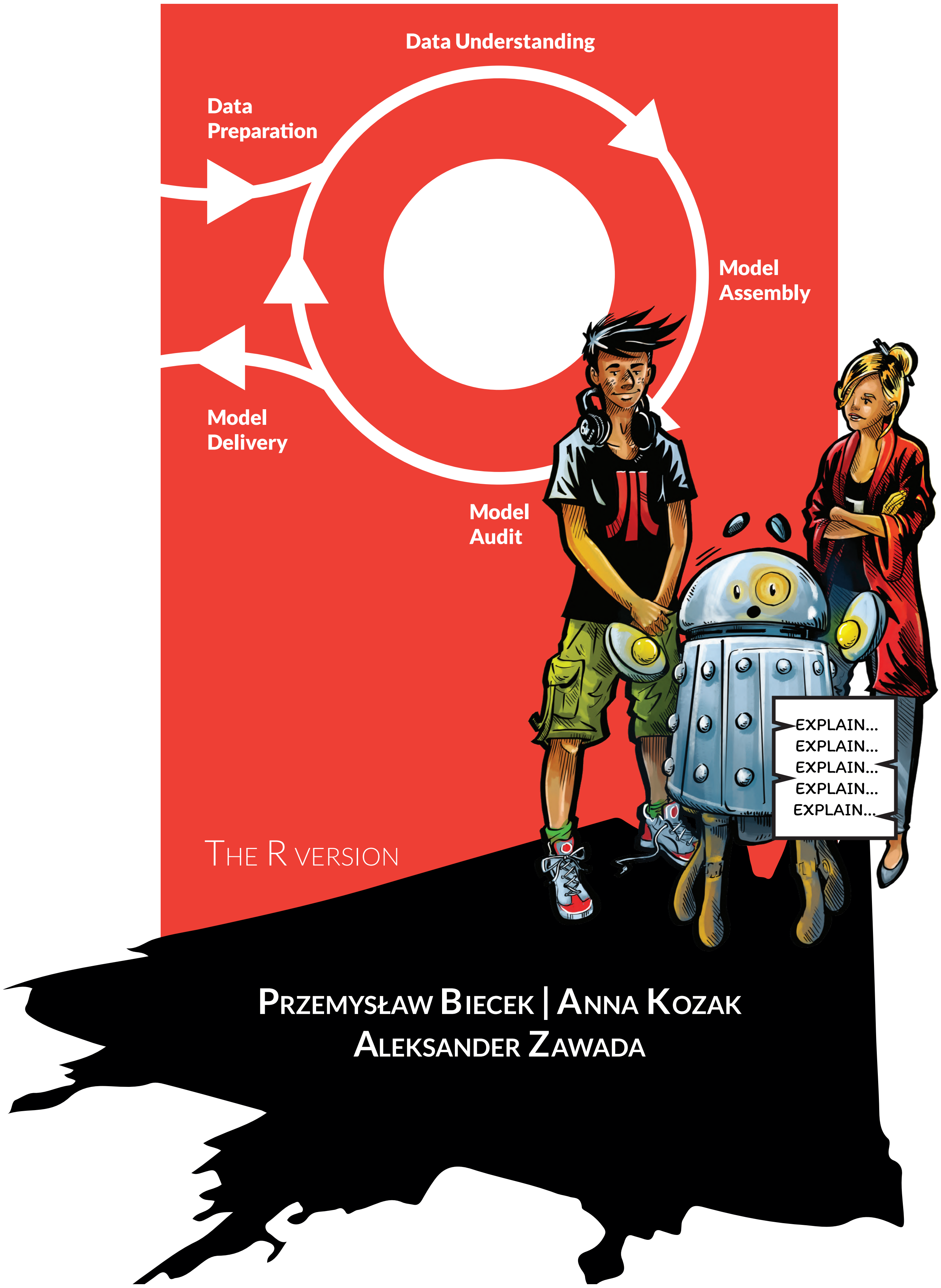
The Hitchhiker's Guide to Responsible Machine Learning

With Beta and Bit



The R version

Przemysław Biecek | Anna Kozak
Aleksander Zawada

The Hitchhiker's Guide to Responsible Machine Learning

The R version

Authors:
Przemysław Biecek, Anna Kozak, Aleksander Zawada

Illustrations and cover:
Aleksander Zawada

Reviewer:
Łukasz Rajkowski

Proofreading:
Bożena Przybyła

Free flipbook:
`https://betaandbit.github.io/RML/`

Buy pdf online:
`https://leanpub.com/RML`

Data and reproducible code snippets:
`https://github.com/BetaAndBit/RML`

Publisher:
Scientific Foundation SmarterPoland.pl

ISBN:
978-83-65291-12-7

Edition I
Warsaw 2022

All right, but how do you build predictive models in a responsible way? This is a question I am often asked by data scientists at different levels of experience. Seemingly simple but at the same time challenging because there are several orthogonal threads and perspectives of different stakeholders that should be addressed.

Model developers focus on automation of model training, monitoring of performance, debugging, and other MLOps-related matters. Users of predictive models are more interested in explainability, transparency and security, while fairness, bias, ethics are issues of interest to society. Regulators are interested in the consequences of model deployments, especially those with large-scale impacts.

Taking these perspectives into account we focus on three essential elements related to Responsible Machine Learning (RML).

Algorithms - Often, to capture complex relationships in data, you need to use advanced and elastic machine learning algorithms. These, however, should not be used without understanding how they work. So a discussion about responsible modelling must touch on the topic of how complex models work.

Software - Training of advanced models is a computationally demanding process. The libraries that allow for efficient training are low-level engineering masterpieces. Professionals use good tools, so a story about responsible modelling must include a section related to good software.

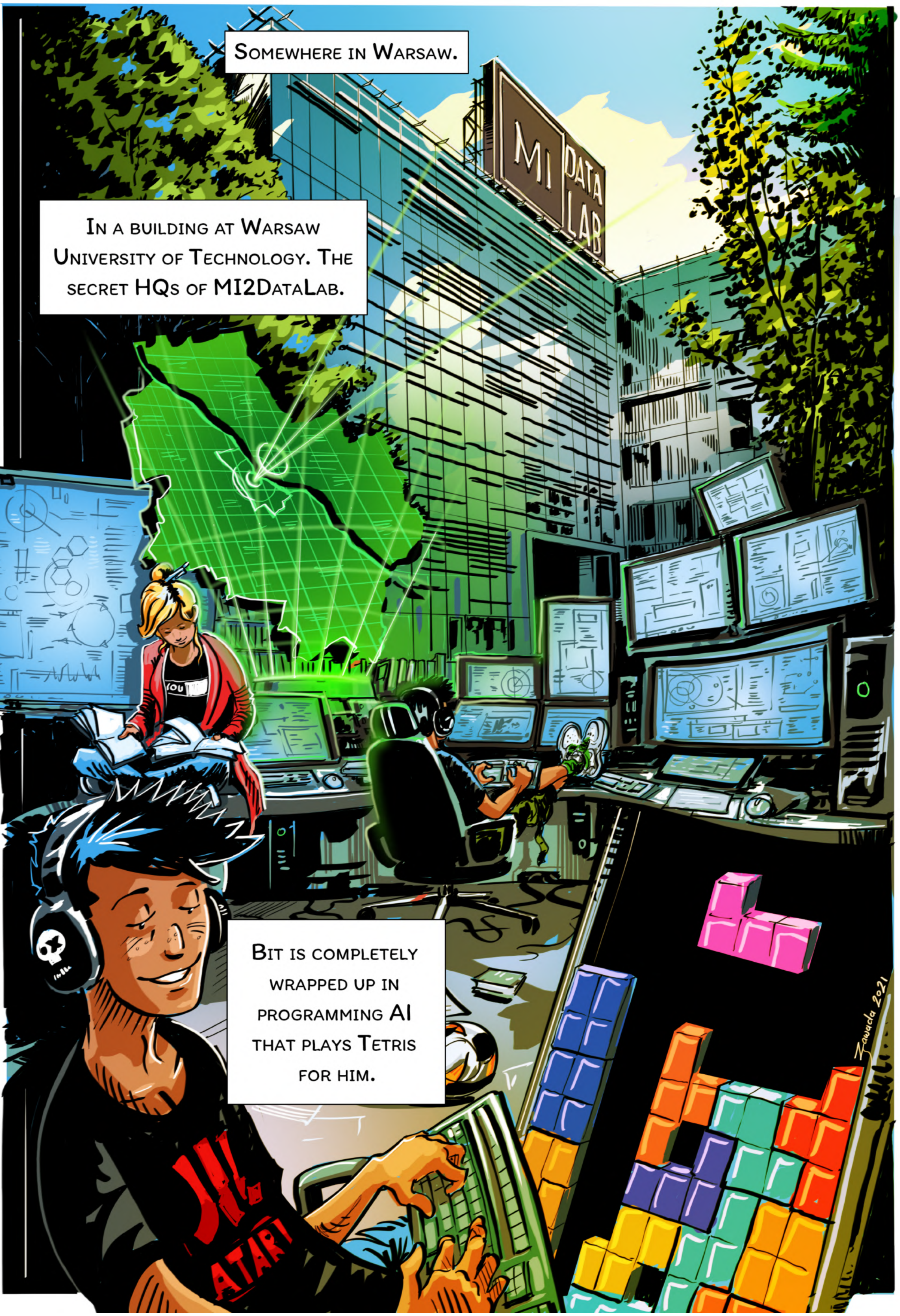
Process - Predictive modelling is not only about tools but also about planning, logistics, communication, deadlines and objectives. The process of data and model exploration is iterative, as in each iteration, we head towards better and better models. Knowing the tools does not help much if you do not know when and how to use them. Therefore, to talk about responsible modelling, we need to talk about the processes behind modelling.

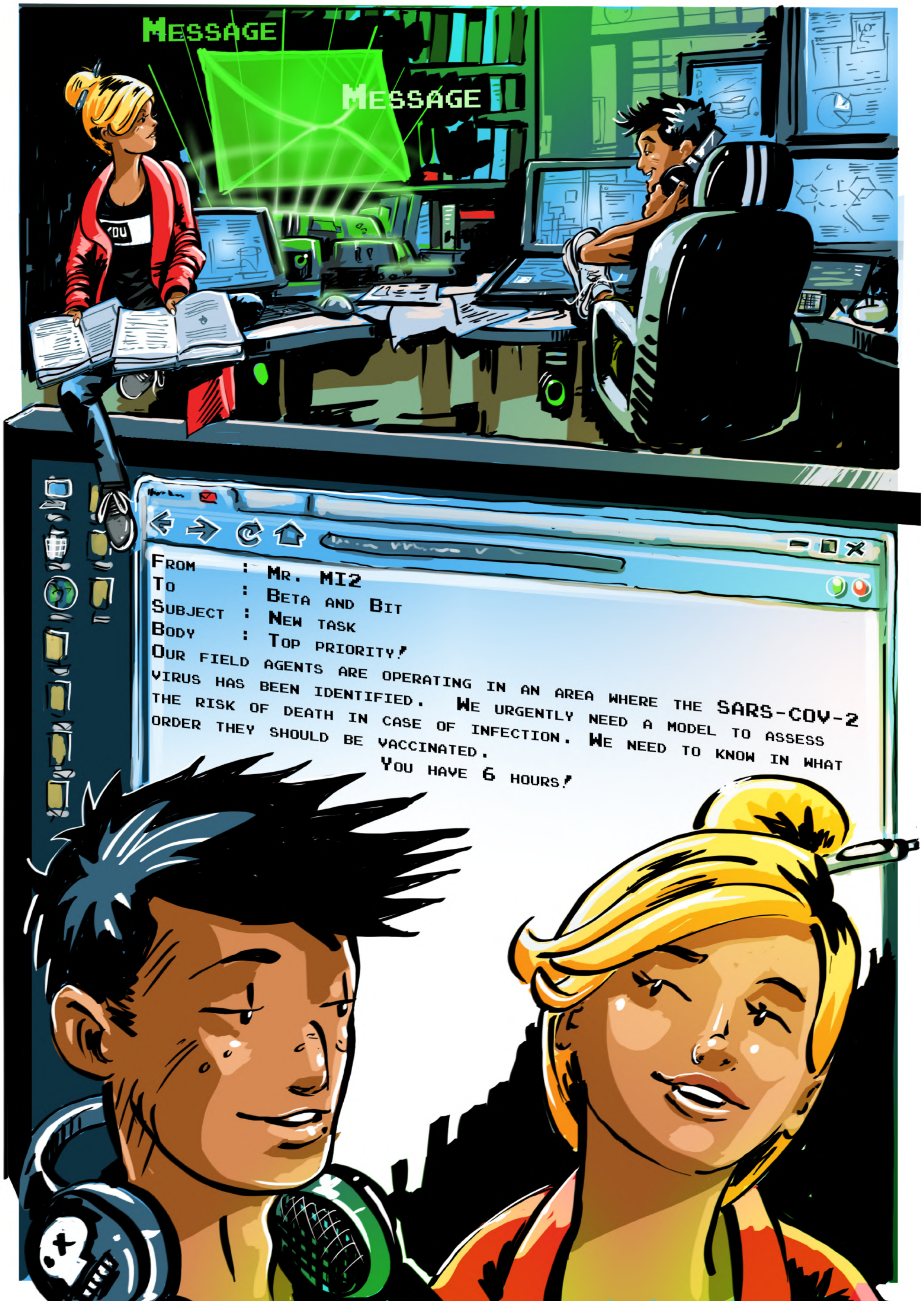
This book is a unique entanglement of all these aspects together at the same time. You will find here selected modern machine learning techniques and the intuition behind them. Methods are supplemented by code snippets with examples in R language[1]. The process is shown through a comic book describing the adventures of two characters, Beta and Bit. The interaction of these two shows the decisions that analysts often face, whether to try a different model, try another technique for exploration or look for other data — questions like: how to compare models or how to validate them.

[1] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021. URL `https://www.R-project.org/`

Model development is responsible and challenging work but also an exciting adventure. Sometimes textbooks focus only on the technical side, losing all the fun. Here we are going to have it all.

Przemysław Biecek
Warszawa, 2022


Somewhere in Warsaw.
In a building at Warsaw University of Technology. The secret HQs of MI2DataLab.
MI DATA LAB
Bit is completely wrapped up in programming AI that plays Tetris for him.
ATARI
Zawada 2021



From : Mr. MI2
To : Beta and Bit
Subject : New task
Body : Top priority!
Our field agents are operating in an area where the SARS-COV-2
virus has been identified. We urgently need a model to assess
the risk of death in case of infection. We need to know in what
order they should be vaccinated.
You have 6 hours!

Predictive models have been used throughout entire human history. Priests in ancient Egypt could predict when the Nile would flood or a solar eclipse would come. Developments in statistics, increasing availability of datasets, and increasing computing power allow predictive models to be built faster and deployed in a rapidly growing number of applications.

Today, predictive models are used virtually everywhere. Planning of the supply chain for a large corporation, recommending lunch or a movie for the evening, or predicting traffic jams in a city. Newspapers are full of exciting applications.

But how are such predictive models developed?

On the following pages, we go through a life cycle of an example predictive model[2] from the concept phase, through design, training, checking, to the deployment. We present an agile approach to building and exploring Machine Learning (ML) models, inspired by the agile approach to software development[3]. The main principles of Agile ML are: continuous adaptation to newly acquired knowledge, continuous prototyping of the solution, dynamic planning, and effective communication. The life cycle of a predictive model is summarized by the diagram below.

[2] We use an example actually built on real data to predict the risk of severe Covid disease progression. But the approach presented can be applied to a very broad class of problems.

[3] Agile manifesto `https://en.wikipedia.org/wiki/Agile_software_development`

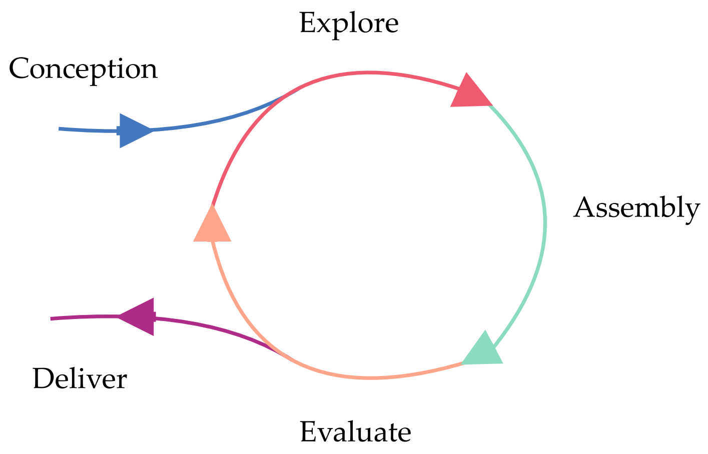


Figure 1: Developing a predictive model often involves many iterations. In this book too, iteration by iteration, we build increasingly complex models, compare them against others, and extract useful information through various Explanatory Model Analysis (EMA) techniques.

Subsequent iterations consist of exploration of literature, data and models, assembly of new solutions and validation after validation. But in addition to these steps, we also show the concept phase, in which the problem to be solved is specified, and the deployment phase, in which the final model is delivered to the users.

We present the life cycle of model development and validation using an example of a binary classification model. We start with a simple model derived from domain knowledge and expand it into a fully data-driven random forest model with automatically tuned hyperparameters. The description of the methods is supplemented with code snippets that you can use to replicate all the presented results yourself. It's worth playing around with these codes to understand better how the described methods work.

Due to the limited space, the descriptions of the methods of both machine learning algorithms and explainable artificial intelligence are brief. If you want to learn more about predictive modelling, I highly recommend the book *An Introduction to Statistical Learning* (ISL)[4]. For those interested in a more detailed description of Explanatory Model Analysis (EMA) and eXplainable Artificial Intelligence (XAI), you will find much more in the book *Explanatory Model Ana-*

[4] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. URL `https://www.statlearning.com/`

lysis[5]. Both are available in paperback but can also be read free of charge in an electronic form.

The modelling approach presented in this book is inspired by the paper *Statistical modeling: the two cultures* by Leo Breiman[6]. It presents two views of modelling, one focused on building models that reflects the laws of nature and the other describing models focused on the effectiveness of predicting a certain trait. As we will show in this book, a bridge can be built between these two approaches. Effective models can and should be used to extract knowledge about a domain, and such knowledge can be furthered transformed into even more effective models.

[5] Przemyslaw Biecek and Tomasz Burzykowski. *Explanatory Model Analysis*. Chapman and Hall/CRC, New York, 2021. URL `https://pbiecek.github.io/ema/`

[6] Leo Breiman. Statistical modeling: the two cultures. *Statistical Science*, 16(3):199–231, 2001b

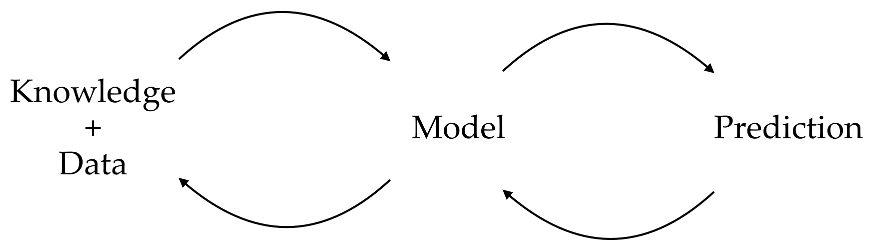


Figure 2: The first part of this book is devoted to the transformation of knowledge and data into a model and then into predictions. The second part of this book discusses how to learn from predictions, how the model works and how to extract information about the domain from the predictive model.

Another interesting point made by Leo Breiman in the article mentioned above is the Rashomon perspective for predictive modelling, i.e. situation in which several equally good models describe the same phenomenon differently. In this book, we show how to examine what different models say about the data. We introduce a pyramid for model exploration that forms a language in which we can show and cross-compare stories learned by different predictive models.

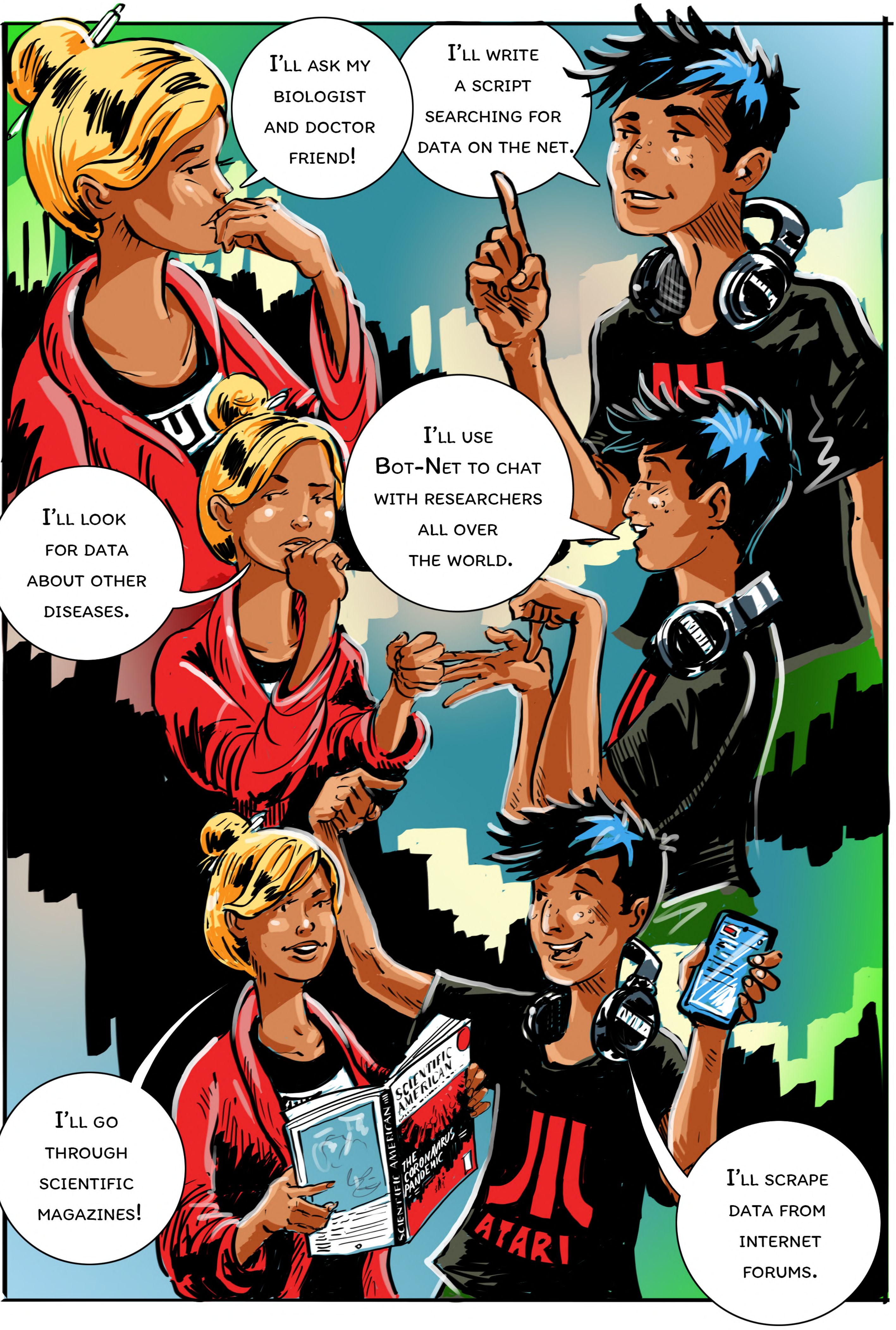
SARS-COV-2 case study

To demonstrate what responsible predictive modelling looks like, we used data obtained in collaboration with the Polish Institute of Hygiene in modelling mortality after the Covid infection. We realize that data on Coronavirus disease can evoke negative feelings. However, it is a good example of how predictive modelling can directly impact our society and how data analysis allows us to deal with complex, important and topical problems.

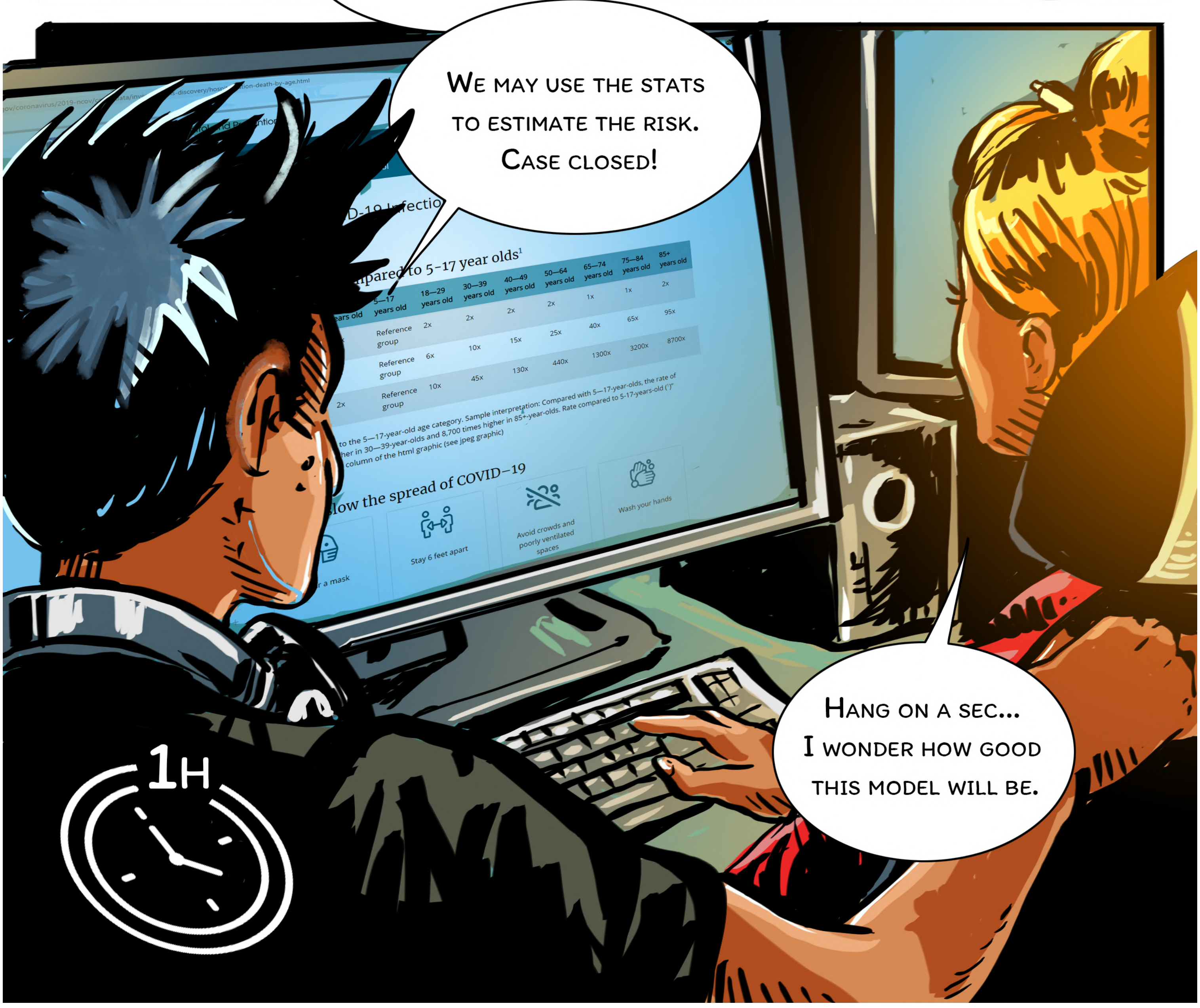
All the results presented in this book can be independently reproduced using the snippets and instructions presented in this book. If you do not want to retype them, then all the examples, data, and codes can be found on the webpage of this book[7]. Please note that the data presented at this URL is artificially generated to mirror relations in the actual data. But it does not contain real patients data.

[7] `https://betaandbit.github.io/RML/`

The procedure outlined here is presented for mortality modelling, but the same process can be replicated whether modelling patient survival, housing pricing, or credit scoring is concerned.


I'LL ASK MY BIOLOGIST AND DOCTOR FRIEND!
I'LL WRITE A SCRIPT SEARCHING FOR DATA ON THE NET.
I'LL LOOK FOR DATA ABOUT OTHER DISEASES.
I'LL USE BOT-NET TO CHAT WITH RESEARCHERS ALL OVER THE WORLD.
I'LL GO THROUGH SCIENTIFIC MAGAZINES!
SCIENTIFIC AMERICAN
THE CORONAVIRUS PANDEMIC
ATARI
I'LL SCRAPE DATA FROM INTERNET FORUMS.

Oh! Look what I've found!
Let's check it on the large screen.


We may use the stats to estimate the risk. Case closed!
Hang on a sec... I wonder how good this model will be.
1h

Hello model!

When browsing through examples of predictive modelling, one may get the wrong impression that the life cycle of the model begins with the data from the internet and ends with validation on an independent dataset. However, this is an oversimplification.

As you will see in a minute, we can create a model without raw data.

The life cycle of a predictive model begins with a **well-defined problem**. In this example, we are looking for a model that assesses the risk of death after being diagnosed with Covid. We don't want to guess who will survive and who won't. Instead, we want to construct a score that allows us to sort patients by their individual risk. Why do we need such a model? For example, those at higher risk of death could be given higher protection, such as providing them with pulse oximeters or preferential vaccination. For this reason, in the following sections, we introduce and use model performance measures that evaluate rankings of scores, such as Area Under Curve (AUC). Pick a model evaluation measure suitable for the problem posed.

Having defined the problem we want to solve, we can move to the next step, which is to **collect all the available information**. Often the solution to the problem can be found in the literature, whether in the form of a ready-made feature prediction function, a discussion of what features are important, or sample data.

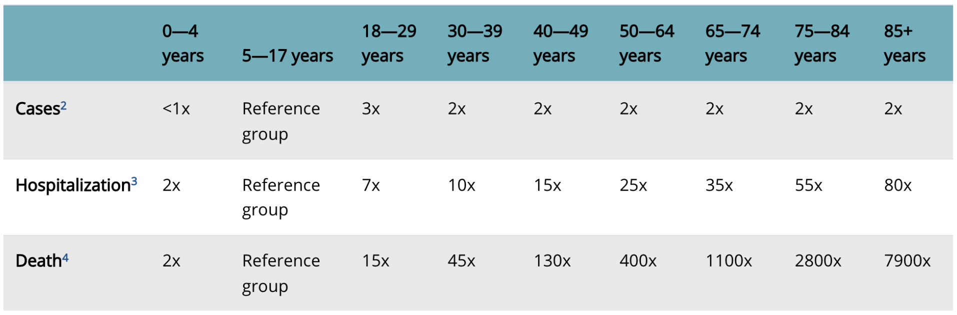
If there are no ready-to-use solutions and we have to collect the data ourselves, it is always worth considering where and what data to collect in order to build the model on a **representative sample**[8]. The problem of data representativeness is a topic for a separate book. Incorrectly collected data will create biases that are hard to discover and even harder to fix.

[8] In our study, we used data on all patients reached by the sanitary inspectorate between March and August 2020. It would seem that data collected in this way would be free of bias, but we were able to detect some. In April, the pandemic spread faster among coal mine workers, who were more likely to be young men, which may have influenced the fluctuations in mortality.

We think of a predictive model as a function that computes certain predictions for specific input data. Usually, such a function is built automatically based on the data. But technically, the model can be any function defined in any way.

Our first model will be based on the statistics collected by the Centers for Disease Control and Prevention (CDC)[9]. That's right; sometimes, we don't need raw data to build a predictive model. We'll start by turning a table with mortality statistics into a model.

[9] https://www.cdc.gov/



	0—4 years	5—17 years	18—29 years	30—39 years	40—49 years	50—64 years	65—74 years	75—84 years	85+ years
Cases[2]	<1x	Reference group	3x	2x	2x	2x	2x	2x	2x
Hospitalization[3]	2x	Reference group	7x	10x	15x	25x	35x	55x	80x
Death[4]	2x	Reference group	15x	45x	130x	400x	1100x	2800x	7900x

Figure 3: Mortality statistics as presented on the CDC website `https://tinyurl.com/CDCmortality` accessed on May 2021. This table shows rate ratios compared to the group 5- to 17-year-olds (selected as the reference group because it has accounted for the largest cumulative number of COVID-19 cases compared to other age groups).

R snippets

A predictive model is a function that transforms the $n \times p$ data frame with p variables for n observations into a vector of n predictions. For further examples, below, we define a function that calculates odds of Covid related death based on statistics from the CDC website for different age groups[10].

[10] There was no risk for the reference group in Table 3. It is not relevant if we are only interested in the ranking of relative risks. But to make the predictions easier to interpret we use here the relative risk determined on Polish data, which is 0.003% for the reference group.

```
cdc_risk <- function(x, base_risk = 0.00003) {
  rratio <- rep(7900, nrow(x))
  rratio[which(x$Age < 84.5)] <- 2800
  rratio[which(x$Age < 74.5)] <- 1100
  rratio[which(x$Age < 64.5)] <- 400
  rratio[which(x$Age < 49.5)] <- 130
  rratio[which(x$Age < 39.5)] <- 45
  rratio[which(x$Age < 29.5)] <- 15
  rratio[which(x$Age < 17.5)] <- 1
  rratio[which(x$Age < 4.5)]  <- 2
  rratio * base_risk
}
steve <- data.frame(Age = 25, Diabetes = "Yes")
cdc_risk(steve)
## [1] 0.00045
```

Predictive models may have different structures. To work responsibly with a large number of models, a uniform standardized interface is needed. In this book, we use the abstraction implemented in the `DALEX` package[11].

The `explain` function from this package creates an *explainer*[12], i.e. a wrapper for the model that will allow you to work uniformly with objects of very different structures. The first argument is a model. It can be an object of any class. The second argument is a function that calculates the vector of predictions. `DALEX` package can often guess which function is needed for a specific model, but in this book, we show it explicitly in order to emphasize how the wrapper works. The `type` argument specifies the model type and the `label` specifies a unique name that appear in the plots.

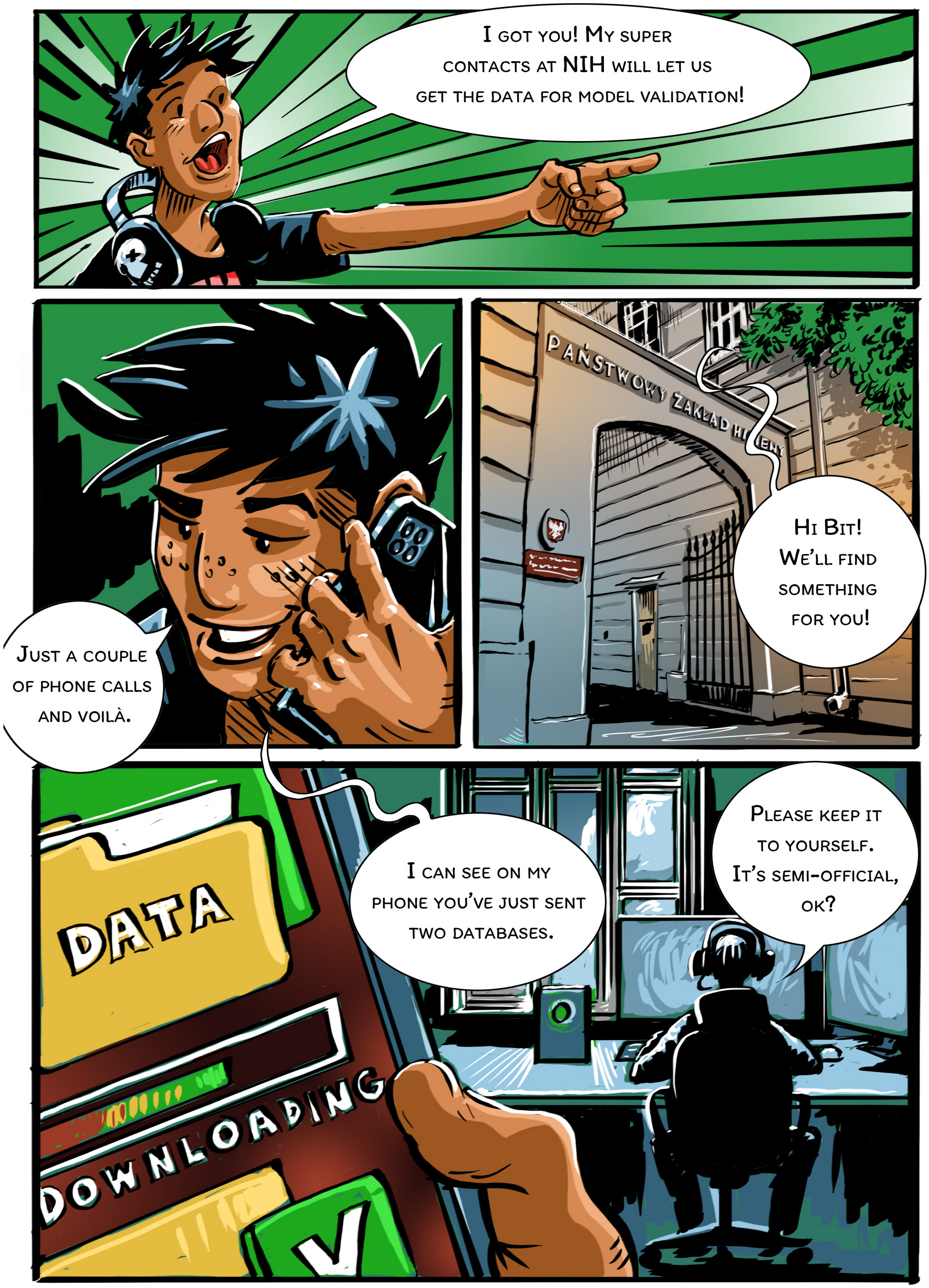
[11] Przemyslaw Biecek. DALEX: Explainers for Complex Predictive Models in R. *Journal of Machine Learning Research*, 19(84):1–5, 2018. URL `https://jmlr.org/papers/v19/18-416.html`

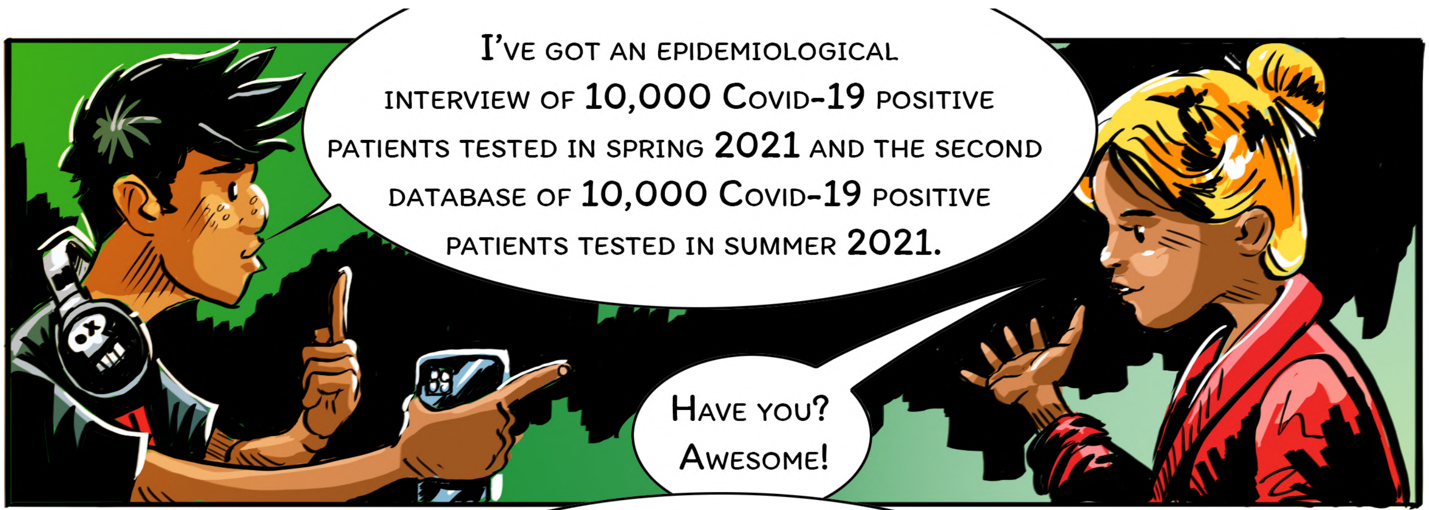
[12] Explainer is an object/adapter that wraps a model and creates a uniform structure and interface for operations.

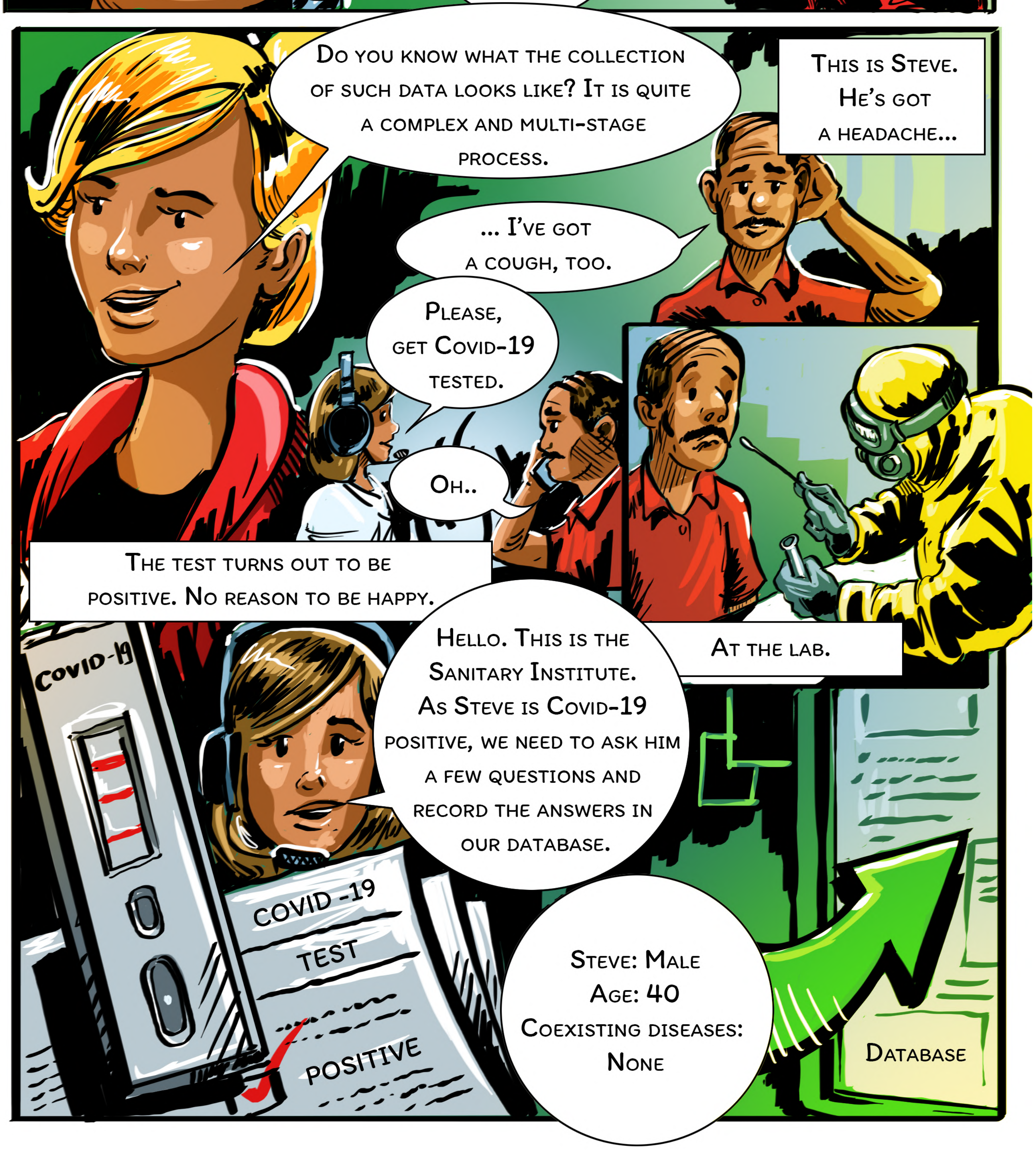
```
library("DALEX")
model_cdc <- DALEX::explain(cdc_risk,
               predict_function = function(m, x) m(x),
               type  = "classification",
               label = "CDC")
predict(model_cdc, steve)
## [1] 0.00045
```

Using the `explain` function may seem like an unnecessary complication at the moment, but on the following pages, we show how it simplifies the work.

The biggest advantage of such a constructed object (explainer) is its standardized structure, to some degree independent of the internal structure of the model.


I got you! My super contacts at NIH will let us get the data for model validation!
Just a couple of phone calls and voilà.
PAŃSTWOWY ZAKŁAD HI ENY
Hi Bit! We'll find something for you!
DATA
DOWNLOADING
I can see on my phone you've just sent two databases.
Please keep it to yourself. It's semi-official, ok?


I've got an epidemiological interview of 10,000 Covid-19 positive patients tested in spring 2021 and the second database of 10,000 Covid-19 positive patients tested in summer 2021.
Have you? Awesome!


Do you know what the collection of such data looks like? It is quite a complex and multi-stage process.
This is Steve. He's got a headache...
... I've got a cough, too.
Please, get Covid-19 tested.
Oh..
The test turns out to be positive. No reason to be happy.
At the lab.
Hello. This is the Sanitary Institute. As Steve is Covid-19 positive, we need to ask him a few questions and record the answers in our database.
Covid-19
COVID -19
TEST
POSITIVE
Steve: Male
Age: 40
Coexisting diseases: None
Database

Exploratory Data Analysis (EDA)

To build a model, we need good data. In Machine Learning, the word *good* means a large amount of representative data. Unfortunately, collecting representative data is neither easy nor cheap and often requires designing and conducting a specific experiment.

The best possible scenario is that one can design and run a study to collect the necessary data. In less comfortable situations, we look for "natural experiments," i.e., data that have been collected for another purpose but that can be used to build a model. Here we use the data[13] collected through epidemiological interviews. The number of interviewed patients is large, so we treat this data as representative, although unfortunately, this data only involves symptomatic patients who are tested positive for SARS-COV-2. Asymptomatic cases are more likely to be young adults.

[13] Please note that the attached data are not the real data collected for epidemiological purposes, but artificially generated data preserving the structure and relationships in the actual data.

The data is divided into two sets: `covid_spring` and `covid_summer`. The first set was acquired in spring 2020 and will be used as training data, while the second dataset was acquired in the summer and will be used for validation. In machine learning, model validation is performed on a separate data set called validation data. This controls the risk of overfitting an elastic model to the training data. If we do not have a separate set, then it is generated using cross-validation, out-of-sample, out-of-time or similar data splitting techniques.

R snippets

The R software offers hundreds of specialized solutions for exploratory data analysis. Certainly, many valuable solutions can be found in the book „R for Data Science”[14], but there are much more. Below we show just three examples. Let's start with loading the data.

[14] Hadley Wickham and Garrett Grolemund. *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly Media, Inc., 2017

```
covid_spring <- read.table("covid_spring.csv", sep =";",
                           header = TRUE)
covid_summer <- read.table("covid_summer.csv", sep =";",
                           header = TRUE)
```

We use the package `ggplot2` to draw a simple histogram for Age, and `ggmosaic` to draw a mosaic plot for `Diabetes`. Note that the plots in the margins are graphically edited, so they look slightly different from the plots generated by these short instructions.

```
# See Figure 4
library("ggplot2")
ggplot(covid_spring) +
     geom_histogram(aes(Age, fill = Death))
# See Figure 5
library("ggmosaic")
ggplot(data = covid_spring) +
     geom_mosaic(aes(x=product(Diabetes), fill = Death))
```

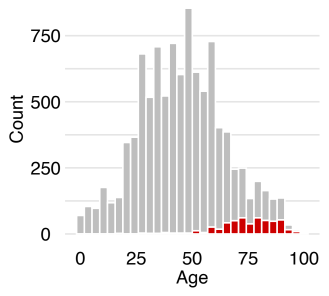


Figure 4: Histogram for the Age variable by survivor status.

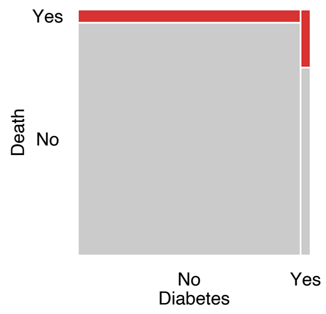


Figure 5: The mosaic plot shows that there are significantly fewer people with diabetes, but among them the mortality is higher.

A handy way to summarise tabular data in groups is the so-called *„Table 1"*. This is a summary of the main characteristics of each variable broken down into groups defined by the variable of interest (here, binary information about Covid death). The name stems from the fact that this summary of the data is usually the first table to be shown in many medical and related scientific journals.

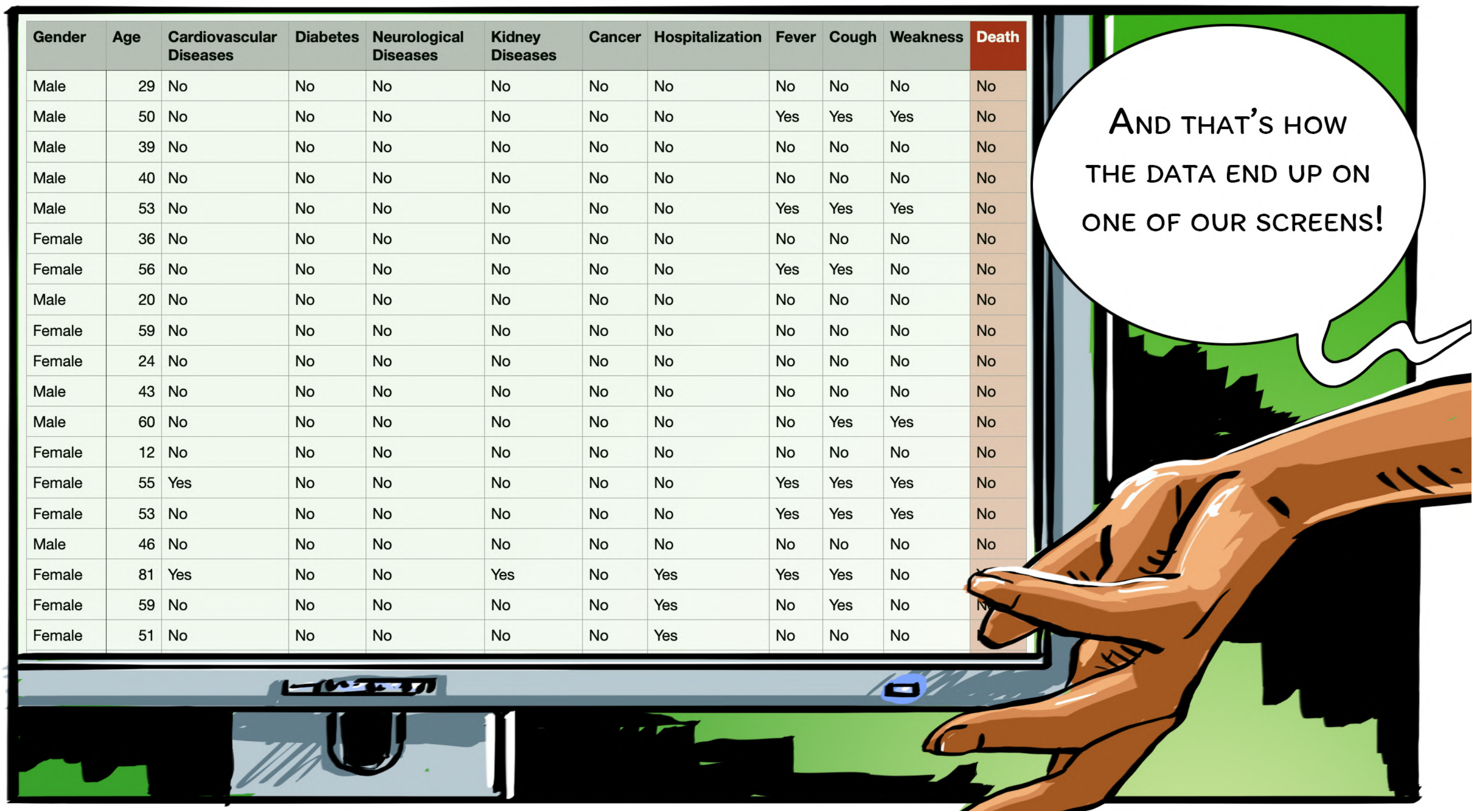
```
library("tableone")
CreateTableOne(vars = colnames(covid_spring)[1:10],
                         data = covid_spring,
                         strata = "Death")
#                                   Stratified by Death
#                                    No            Yes
#  n                                  9487          513
#  Gender = Male (%)                  4554 (48.0)    271 (52.8) 0.037
#  Age (mean (SD))                   44.19 (18.32) 74.44 (13.2) <0.001
#  CardiovascularDiseases = Yes (%)   839 ( 8.8)    273 (53.2) <0.001
#  Diabetes = Yes (%)                  260 ( 2.7)     78 (15.2) <0.001
#  Neurological.Diseases = Yes (%)    127 ( 1.3)     57 (11.1) <0.001
#  Kidney.Diseases = Yes (%)           111 ( 1.2)     62 (12.1) <0.001
#  Cancer = Yes (%)                    158 ( 1.7)     68 (13.3) <0.001
#  Hospitalization = Yes (%)          2344 (24.7)    481 (93.8) <0.001
#  Fever = Yes (%)                    3314 (34.9)    335 (65.3) <0.001
#  Cough = Yes (%)                    3062 (32.3)    253 (49.3) <0.001
```

One of the most important rules to remember when building a predictive model is: **Do not condition on future!**. I.e. do not use variables that are not defined at the time the prediction needs to be made. Note that in the discussed case variables `Hospitalization`, `Fever` or `Cough` are not good predictors because they are not known in advance before infection. So they are not useful in our case.

In the following lines, we remove invalid variables from both data sets.

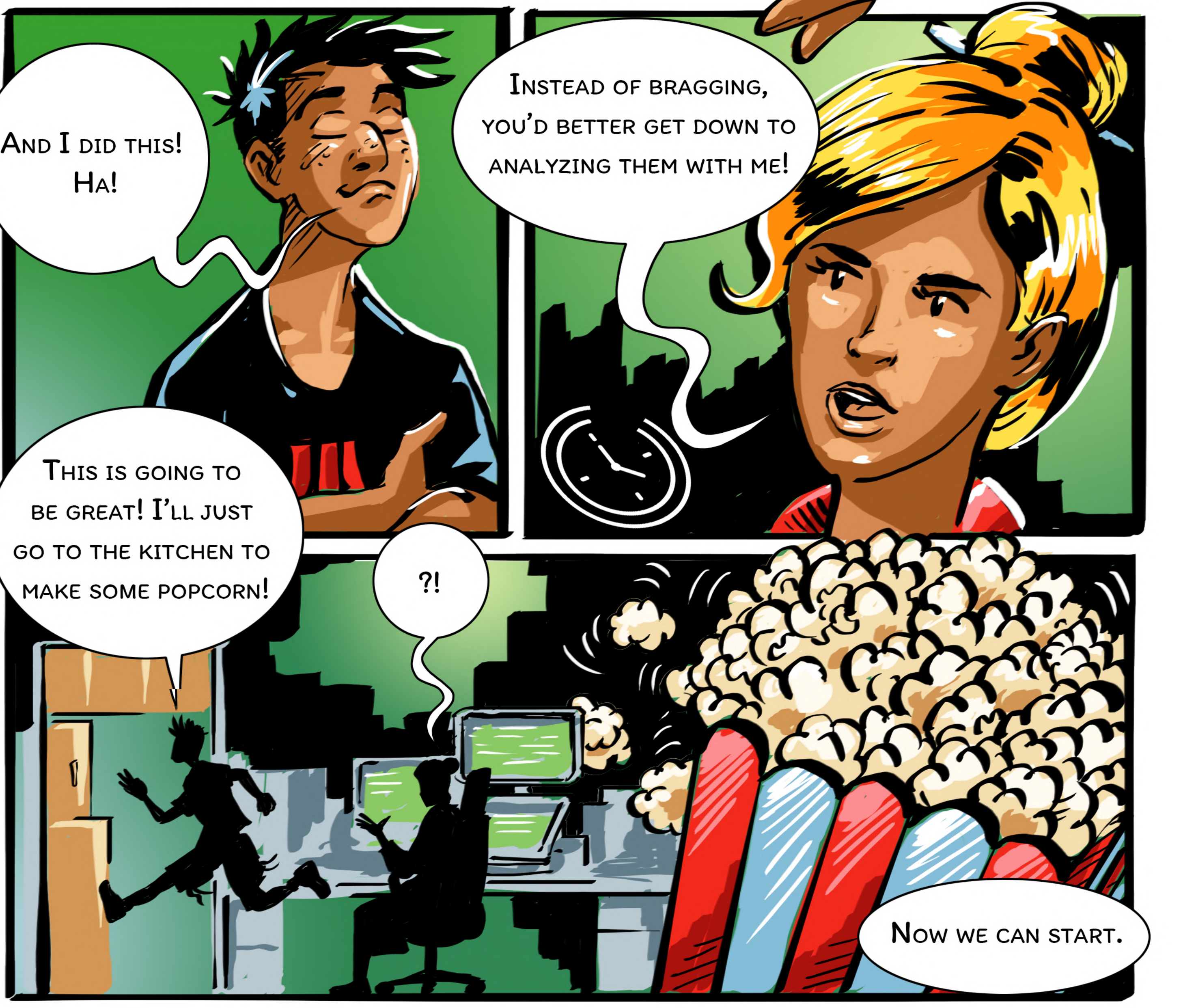
```
selected_vars <- c("Gender", "Age", "Cardiovascular.Diseases",
    "Diabetes", "Neurological.Diseases", "Kidney.Diseases",
    "Cancer", "Death")
# use only selected variables
covid_spring <- covid_spring[,selected_vars]
covid_summer <- covid_summer[,selected_vars]
```

Data exploration and cleansing often consume most of the time spent on data analysis. Here we have only touched on exploration, but even this initial analysis helped to determine that `Age` is an important characteristic (we will confirm this later). From the list of variables we removed those that are unknown before the disease develops (like hospitalization status). We will build further models only on variables from the `selected_vars` vector.



Gender	Age	Cardiovascular Diseases	Diabetes	Neurological Diseases	Kidney Diseases	Cancer	Hospitalization	Fever	Cough	Weakness	Death
Male	29	No	No	No	No	No	No	No	No	No	No
Male	50	No	No	No	No	No	No	Yes	Yes	Yes	No
Male	39	No	No	No	No	No	No	No	No	No	No
Male	40	No	No	No	No	No	No	No	No	No	No
Male	53	No	No	No	No	No	No	Yes	Yes	Yes	No
Female	36	No	No	No	No	No	No	No	No	No	No
Female	56	No	No	No	No	No	No	Yes	Yes	No	No
Male	20	No	No	No	No	No	No	No	No	No	No
Female	59	No	No	No	No	No	No	No	No	No	No
Female	24	No	No	No	No	No	No	No	No	No	No
Male	43	No	No	No	No	No	No	No	No	No	No
Male	60	No	No	No	No	No	No	No	Yes	Yes	No
Female	12	No	No	No	No	No	No	No	No	No	No
Female	55	Yes	No	No	No	No	No	Yes	Yes	Yes	No
Female	53	No	No	No	No	No	No	Yes	Yes	Yes	No
Male	46	No	No	No	No	No	No	No	No	No	No
Female	81	Yes	No	No	Yes	No	Yes	Yes	Yes	No	[illegible]
Female	59	No	No	No	No	No	Yes	No	Yes	No	No
Female	51	No	No	No	No	No	Yes	No	No	No	[illegible]

And that's how the data end up on one of our screens!


And I did this! Ha!
Instead of bragging, you'd better get down to analyzing them with me!
This is going to be great! I'll just go to the kitchen to make some popcorn!
?!
Now we can start.

Sooo much data, sooo many interactions...
COVID
19
Coffee
Tea
Yes
Yes
No
No
No
No
No

Model Performance

Depending on the type of predictive problem and what we assume about the distribution of the outcome, various measures of model performance can be used. Here is a short summary; find a more detailed description in the EMA book.

For regression problems, when we predict a quantitative variable, especially when we assume Gaussian noise, commonly used performance measures are Mean Squared Error[15] and Rooted Mean Squared Error[16].

[15] If $f : \mathcal{R}^p \to \mathcal{R}$ is a function that predicts the value of y_i using observation x_i, then $MSE = \frac{1}{n}\sum_i^n (f(x_i) - y_i)^2$

[16] $RMSE = \sqrt{MSE}$

For binary classification problems, the outcome is commonly summarized with a 2×2 contingency table with possible results coded as True Positive, True Negative, False Positive, and False Negative. Positive means that the test suggests a pregnancy, while Negative means no pregnancy. True and False describe whether the test result is correct or not. Below is an example of such a table for a simple „morning sickness" test for the pregnancy.

Table 1: Is morning sickness a good pregnancy test? This table is based on GetTheDiagnosis data `http://getthediagnosis.org/diagnosis/Pregnancy.htm`. For example, `FN = 61` means that out of 100 pregnant women for 61 the test suggested otherwise. Exemplary measures of performance are shown in the last row and column.

Morning sickness / pregnancy	Pregnant	Not pregnant	
Has sickness	**TP** = 39	**FP** = 150	PPV = Prec = 20.6%
Has not	**FN** = 61	**TN** = 850	NPV = 93.3%
	Sensitivity = Recall = 39%	Specificity = 85%	F1 = 33.8%

Based on such a contingency table, the most commonly used measures of performance are Accuracy[17], Sensitivity[18], Specificity[19], Precision[20], Recall[21], F1 score[22], Positive Predicted Value[23] and Negative Predicted Value[24].

[17] $Acc = (TP + TN)/n$

[18] $Sens = TP/(TP + FN)$

[19] $Spec = TN/(TN + FP)$

[20] $Prec = TP/(TP + FP)$

[21] $Recall = TP/(TP + FN)$

[22] $F1 = 2\frac{Prec*Recall}{Prec+Recall}$

[23] $PPV = TP/(TP + FP)$

[24] $NPV = TN/(TN + FN)$

Note that in the covid-mortality-risk-assessment problem, we are not interested in the binary prediction survived/dead, but rather in the validity of the ranking of risk scores. For such types of problems, instead of a contingency table, one looks at Receiver Operating Characteristic (ROC) curve, and the commonly used measure of performance is the Area Under the ROC Curve (AUC). Figure 6 shows how this measure is constructed.

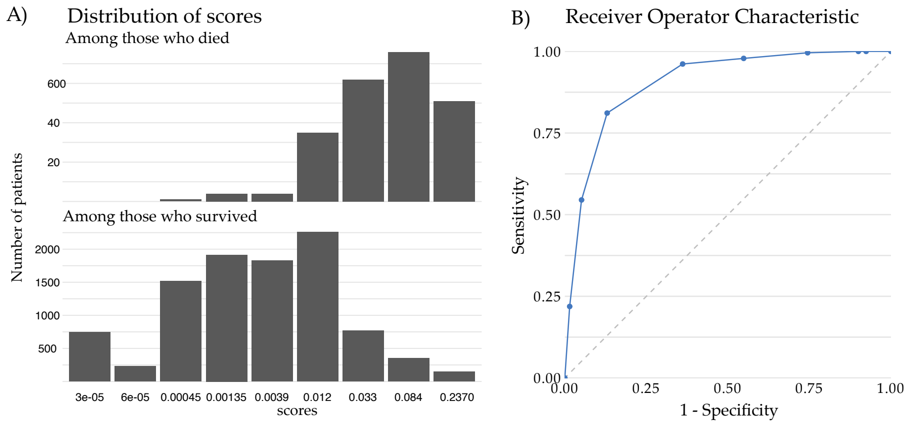


Figure 6: Panel A shows the distribution of scores obtained from the CDC model for the test data divided by the survival status. By taking different cutoffs, one can turn such numerical scores into binary decisions. For each such a split, the Sensitivity and 1-Specificity can be calculated and drawn on a plot. The CDC model returns only nine different values, making it reasonable to consider ten different cutoffs.

Panel B shows these 10 points corresponding to different cutoffs. The ROC curve is the piecewise line connecting these points and the AUC is the area under this curve. The AUC takes values from 0 to 1, where 1 is the perfect ranking and a purely random ranking leads to the AUC of 0.5.

R snippets

There are many measures for evaluating predictive models, and they are implemented in various R packages (i.e. ROCR, measures, mlr3measures). For simplicity, in this example we show only model performance measures implemented in the DALEX package.

First, we need an explainer with specified validation data (here covid_summer) and the corresponding response variable.

```
model_cdc <-  DALEX::explain(cdc_risk,
                   predict_function = function(m, x) m(x),
                   data  = covid_summer,
                   y     = covid_summer$Death == "Yes",
                   type  = "classification",
                   label = "CDC")
```

Model exploration starts with an assessment of how good is the model. The DALEX::model_performance function calculates a set of measures for a specified type of task, here classification.

```
mp_cdc <- model_performance(model_cdc, cutoff = 0.1)
mp_cdc

# Measures for:  classification
# recall     : 0.2188841
# precision  : 0.2602041
# f1         : 0.2377622
# accuracy   : 0.9673
# auc        : 0.906654
#
# Residuals:
#       0%      10%      20%       30%      40%      50%
# -0.23700 -0.03300 -0.01200  -0.01200 -0.00390 -0.00390
#      60%      70%      80%       90%     100%
# -0.00135 -0.00135 -0.00045  -0.00006  0.99955
```

Note: The model is evaluated on the data given in the explainer. Use DALEX::update_data() to specify another dataset, e.g. training data covid_spring.

```
model_cdc <-  update_data(model_cdc,
                   data  = covid_spring,
                   y     = covid_spring == "Yes")
```

Note: The explainer knows whether the model is trained for classification or regression task, so it automatically selects the right performance measures. This can be overridden if needed.

The S3 generic plot function draws a graphical summary of the model performance. With the geom argument, one can determine the type of chart.

```
# ROC curve, see Figure 6
plot(mp_cdc, geom = "roc")
# LIFT curve, see Figure 7
plot(mp_cdc, geom = "lift")
```

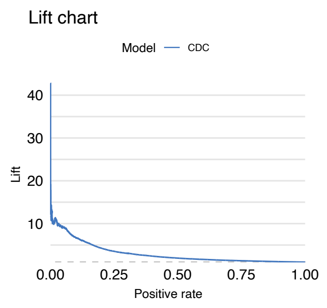
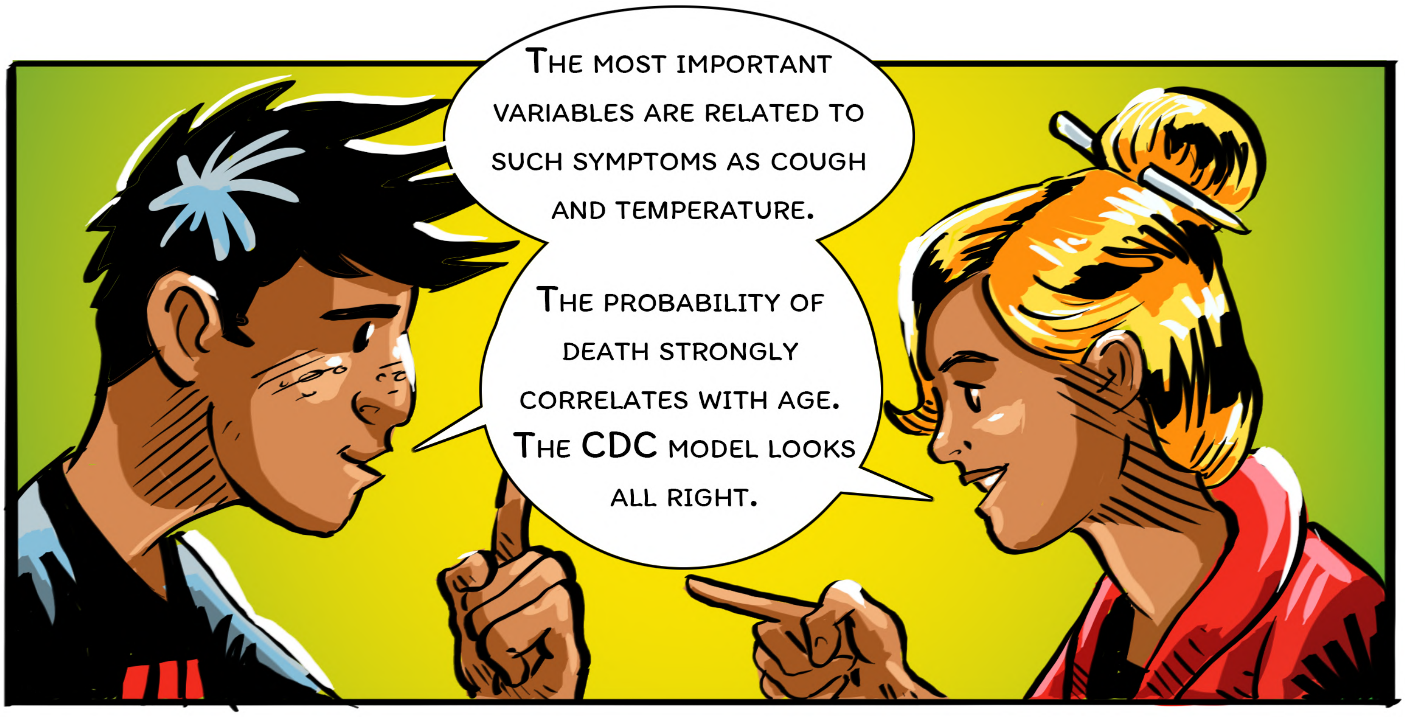
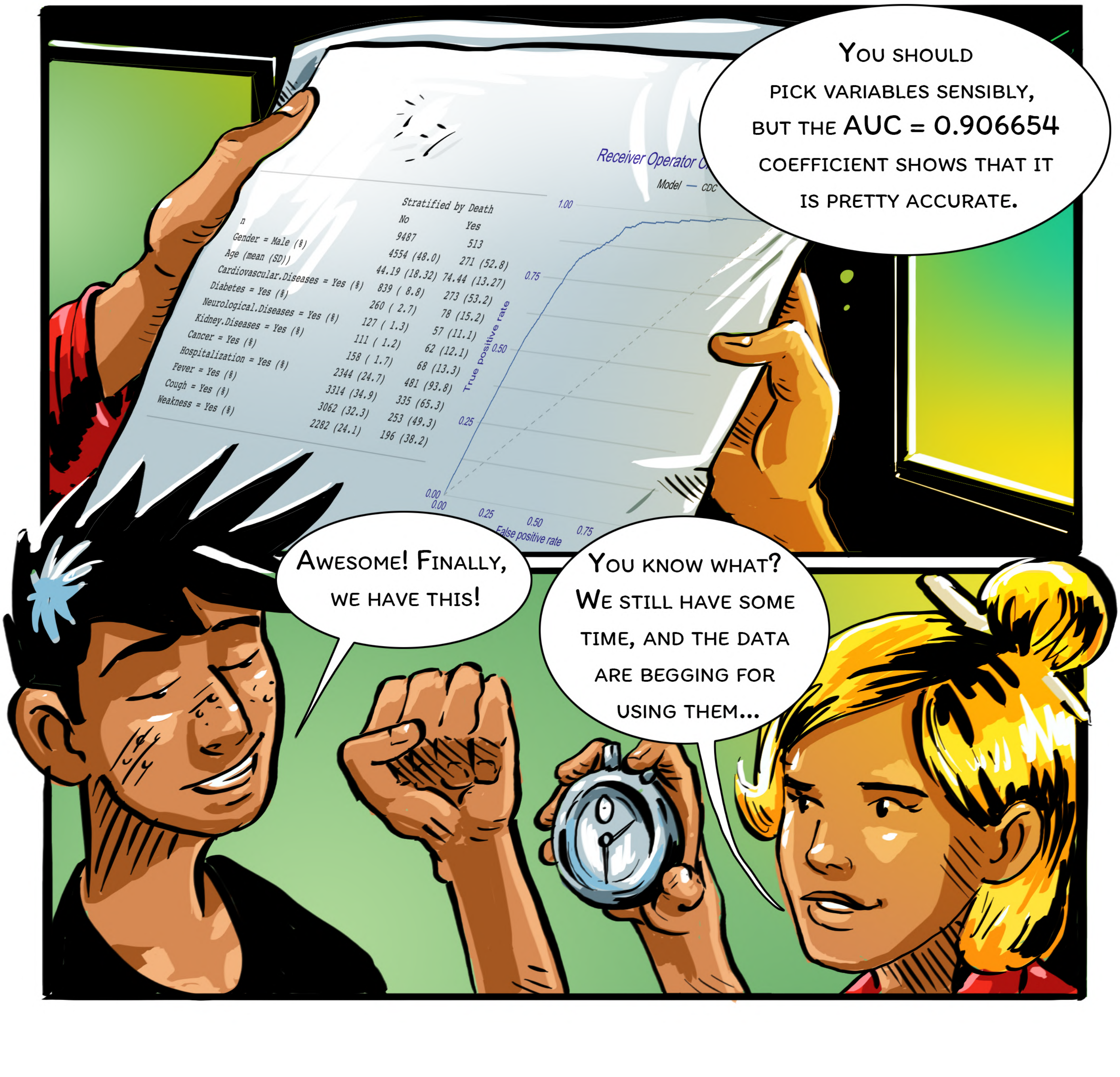


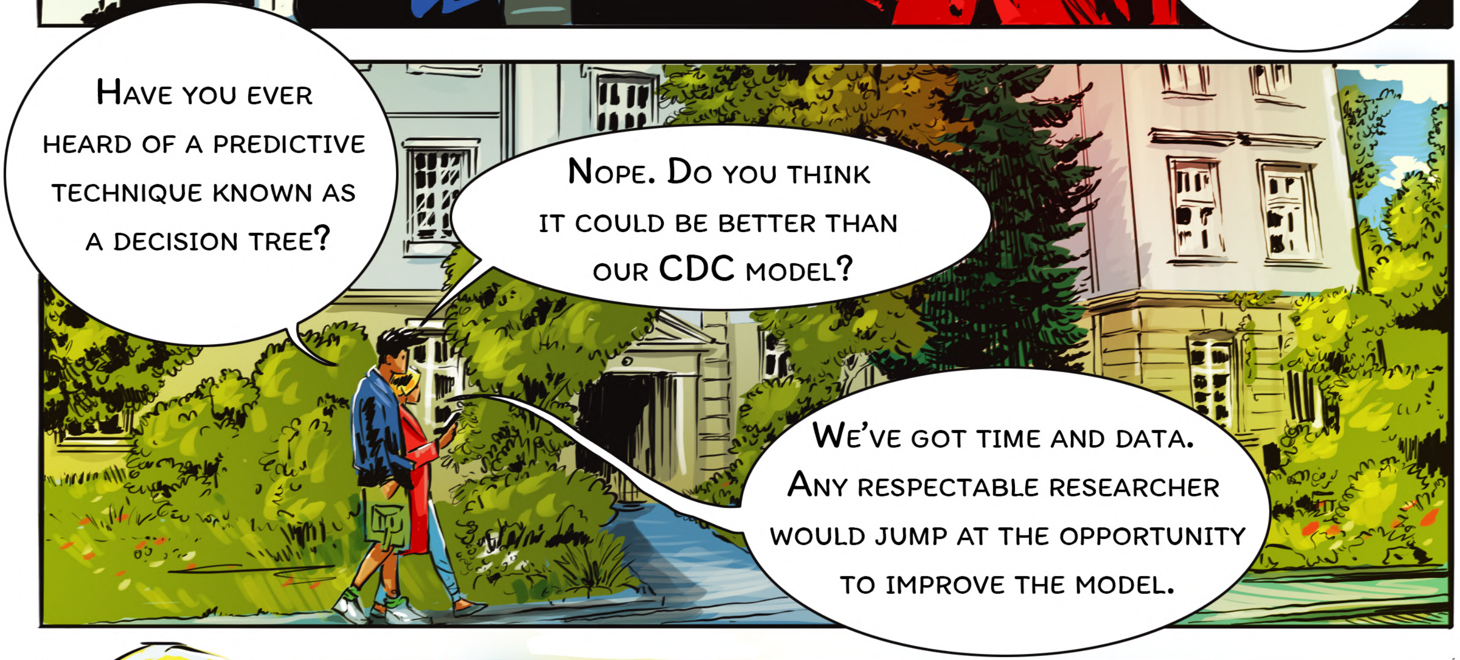
Figure 7: LIFT curve, one of the many graphical statistics used in summarizing the quality of scores, often used in credit risk scoring. The OX axis presents the fraction of assigned credits, and the OY axis presents the ratio of the Sensitivity of the tested model to the Sensitivity of the random model.

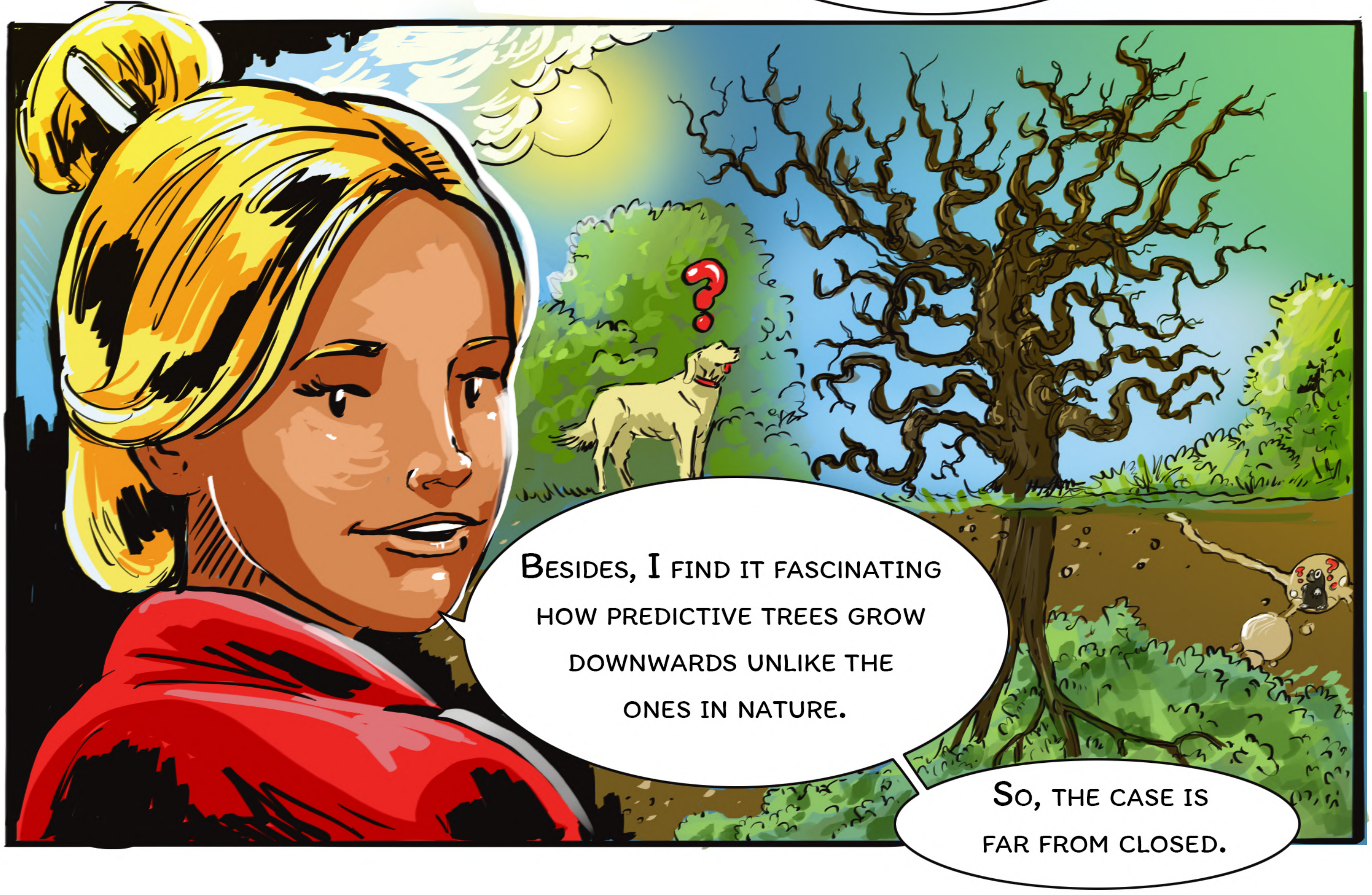

The most important variables are related to such symptoms as cough and temperature.
The probability of death strongly correlates with age. The CDC model looks all right.


Receiver Operator C
Model — CDC
1.00
0.75
0.50
0.25
0.00
True positive rate
0.00
0.25
0.50
0.75
False positive rate
You should pick variables sensibly, but the AUC = 0.906654 coefficient shows that it is pretty accurate.
Awesome! Finally, we have this!
You know what? We still have some time, and the data are begging for using them...

	Stratified by Death	
	No	Yes
n	9487	513
Gender = Male (%)	4554 (48.0)	271 (52.8)
Age (mean (SD))	44.19 (18.32)	74.44 (13.27)
Cardiovascular.Diseases = Yes (%)	839 (8.8)	273 (53.2)
Diabetes = Yes (%)	260 (2.7)	78 (15.2)
Neurological.Diseases = Yes (%)	127 (1.3)	57 (11.1)
Kidney.Diseases = Yes (%)	111 (1.2)	62 (12.1)
Cancer = Yes (%)	158 (1.7)	68 (13.3)
Hospitalization = Yes (%)	2344 (24.7)	481 (93.8)
Fever = Yes (%)	3314 (34.9)	335 (65.3)
Cough = Yes (%)	3062 (32.3)	253 (49.3)
Weakness = Yes (%)	2282 (24.1)	196 (38.2)

You're right. Let's go and get some fresh air. We can come up with something.
Well, I've already got an idea...


Have you ever heard of a predictive technique known as a decision tree?
Nope. Do you think it could be better than our CDC model?
We've got time and data. Any respectable researcher would jump at the opportunity to improve the model.


Besides, I find it fascinating how predictive trees grow downwards unlike the ones in nature.
So, the case is far from closed.