Practical application of the Wilcoxon-Mann-Whitney test in valuation.

Selection of attributes as pricing factors based on the principle of unbiased estimates

K. A. Murashev

August 28, 2022

Appraisers often face the need to take into account differences in quantitative and qualitative characteristics of objects in their practice. In particular, one of the standard tasks is to determine the attributes that influence the cost (so-called "pricing factors") and to separate them from the attributes that do not or cannot be determined.

Subjective selection of attributes taken into account in determining the value is widespread in valuation practice. In this case, specific quantitative indicators of the impact of these attributes on the cost are often taken from the so-called "reference books". While not denying the speed and low cost of this approach, it should be recognized that only data directly observed in the open markets is a reliable basis for a value judgment. The priority of such data over other data, in particular those obtained by expert survey, is enshrined, among others, in RICS Valuation — Global Standards 2022 [19], International Valuation Standards 2022 [21], as well as in IFRS 13 "Fair Value Measurement" [15]. Therefore, we can say that mathematical methods for analyzing data from the open market are the most reliable means of interpreting market information used in market research and predicting the value of individual objects.

The aim of this work is to justify the necessity and possibility of using a rigorous mathematical Wilcoxon–Mann–Whitney test, which allows us to answer the question about the necessity of taking into account the binary attribute as a price-generating factor. Judgmental approach is most commonly used by appraisers in selecting the attributes to be considered in appraisal. But this paper proposes the idea of prioritizing the measuring approach based on the results of a mathematical test. It allows to draw a conclusion about the importance or otherwise of the binary attribute influence on the value. It should be noted that despite the fact that the statistical test under consideration belongs to frequentist statistics, it, through its connection to ROC analysis and AUC, is related to modern machine learning methods, which will be discussed later in the text of this material. The presence of this relationship and elements of Bayesian statistics seems particularly interesting and promising from the point of view of introducing machine learning and data analysis methods into the everyday practice of appraisers.

Users should have some general math background and basic Python and R programming skills to understand and practice all of the material in the text, but lack of that knowledge and skill is not a barrier to learning most of the material and implementing the test in the spreadsheet that comes with it.

The material consists of four blocks:

- a description of the Wilcoxon–Mann–Whitney test (hereafter "U-test"), its probabilistic meaning, and its relationship to other mathematical methods;
- a practical implementation of the U-test in a spreadsheet on an example of test random data;

- practical implementation of the U-test on the real data of the residential real estate market of St. Petersburg agglomeration by means of Python programming language, the purpose of the analysis was to check the significance of the difference in the unit price between the objects located in the urban and suburban parts of the agglomeration;
- practical implementation of the U-test on real data of residential real estate market of Almaty by means of R programming language, the purpose of the analysis was to check the significance of difference in unit price between the objects sold without demountable improvements and the objects sold with them.

The current version of this material, its source code, Python and R scripts, and the spreadsheet are in the repository on the GitHub portal and are available at the permanent link [22].

Contents

List of Tables

List of Figures

Listings

1 Technical details

This material, as well as the appendices to it, are available at permanent link [22]. The source code for this work was created using the language TEX [45] with a set of macro extensions LATEX 2_ε [46], distribution TeXLive [47] and Editor TeXstudio [82]. The spreadsheet calculation was done with LibreOffice Calc [30] (Version: 7.3.4. 2 / LibreOffice Community Build ID: 30(Build:2); CPU threads: 4; OS: Linux 5.11; UI render: default; VCL: kf5 (cairo+xcb) Locale: en-US (en_US.UTF-8); UI: en-US Ubuntu package version: 1:7.3.4 rc2-0ubuntu0.20.04.1 lo1; Calc: threaded). The calculation in R [48] (version 4.2.1 (2022-06-23) – "Funny-Looking Kid") was done using an IDE RStudio (RStudio 2022. 02.3+492 "Prairie Trillium" Release (1db809b8, 2022-05-20) for Ubuntu Bionic; Mozilla/5.0 (X11; Linux x86_64); AppleWebKit/537.36 (KHTML, like Gecko); QtWebEngine/5.12.8; Chrome/69.0.3497.128; Safari/537.36) [44]. The calculation in Python (Version 3.9.12) [29] was performed using the development environment Jupyter Lab (Version 3.4.2) [36] and IDE Spyder (Spyder version: 5.1.5 None* Python version: 3.9.12 64-bit * Qt version: 5.9.7 * PyQt5 version: 5.9.2 * Operating System: Linux 5.11.0-37-generic) [43]. The graphics used in the subsection 4.1 were prepared using Geogebra (Version 6.0.666.0-202109211234) [31]. The following values were used in this material as well as in most of the works in the series:

- significance level: $\alpha = 0.05$;
- confidence interval: $Pr = 0.95$;
- initial position of the pseudo-random number generator: $seed = 19190709$.

A dot is used as a decimal point. Most of the mathematical notations are written as they are used in English-speaking circles. For example, a tangent is written as tan, not tg. The results of statistical tests are considered significant when

$$p \leq \alpha. \tag{1.1}$$

This decision is based, in part, on the results of the discussion that took place on researchgate.net [35].

2 Subject of research

When working with market data, the appraiser is often faced with the task of testing the hypothesis of whether a quantitative, ordinal or nominal attribute has a significant effect on the price. Real estate market analysts, developers, realtors, employees of collateral departments of banks, leasing and insurance companies, tax inspectors and other specialists have a similar task. At the same time, it is often impossible to collect large amounts of data that would allow a wide range of machine learning methods to be applied. In some cases appraisers consciously narrow the area of data collection to the narrow market segment, resulting in only very small samples of less than thirty observations at their disposal. In this case, the price data most often has a distribution that differs from the normal one. In this case, a rational solution is to use U-test. Let us formulate the problem:

- suppose that we have two samples of unit prices for commercial premises, some of which have some attribute (e. g., having a separate entrance) and some of which do not;
- it is necessary to determine whether the presence of this feature has a significant impact on the unit value of this type of real estate or not.

At first glance, according to established practice, an appraiser can simply subjectively recognize some attributes as significant and others as not, and then accept the adjustment values for differences in these attributes from the reference books. However, as mentioned above, this approach is hardly considered best practice because it lacks any market analysis. Also, in that case, it is unlikely that such work is of any serious value at all.

Instead, it is possible to use random samples of market data and apply mathematical analysis to them, allowing scientific and evidence-based conclusions to be drawn about the significance of a particular attribute's impact on value. The data used in this paper to perform the U-test using Python and R are real market data, some of which were collected by the author through web scraping and some provided by colleagues for the analysis. The attached spreadsheet is set up so that test raw data can be generated in a pseudo-random fashion.

The subject of this paper is the nonparametric Wilcoxon-Mann-Whitney test, specifically designed for samples that have a distribution other than normal. This circumstance is important because the price data that appraisers deal with most often have this distribution, which excludes the possibility of applying the parametric t-criterion and z-criterion. In addition, the test under consideration is of great interest because it has a connection to machine learning methods through AUC, the calculation of which

through the formula provided in the test framework gives a value equal to that calculated by ROC analysis. Thus, the study of the U-test paves the way for a further dive into the world of machine learning, which is entering many areas of human activity and will significantly change the field of value estimation in the foreseeable future.

The material contains a description of the test and instructions for performing it, sufficient in the author's opinion for its demonstrable use in the estimation process.

3 Basic information about the test

3.1 Assumptions and formalization of hypotheses

First of all, it should be said that, in spite of the stated common name, it is more correct to speak of two tests:

- Wilcoxon rank-sum test developed by Frank Wilcoxon in 1945 [40];
- Mann–Whitney U-test which is a further development of the aforementioned criterion developed by Henry Mann and Donald Whitney in 1947 [38].

Looking ahead we can say that the statistics of these criteria are linearly related and their p-values are almost the same which from a practical point of view allows us to talk about variations of one test rather than two separate tests. This paper uses the common name throughout the text, as well as a shortened version of "U-test" which historically refers to the Mann-Whitney test. Some authors[13] recommend using the Wilcoxon rank-sum test when there are no assumptions about variance, and the Mann-Whitney U-test when variance of the two samples are equal. However, the experimental data indicate that the Wilcoxon rank-sum test and Mann-Whitney U-test values are essentially the same when the variance of the samples is significantly different. Adhering to the KISS principle [63] underlying the entire series of publications, the author concludes that a unified approach is possible. Also remember that the Wilcoxon signed-rank test is a separate test designed to analyze differences between two matched samples, whereas the Mann-Whitney U-test discussed in this paper is designed to work with two independent samples.

Suppose that there are two samples:

$$x^m = (x_1, x_2, \ldots, x_m), x_i \in \mathbb{R}; \quad y^n = (y_1, y_2, \ldots, y_n), y_i \in \mathbb{R} \quad : m \leq n.$$

- Both samples are simple and random (i.e., SRS [76]), the combined sample is independent.
- The samples are taken from unknown continuous distributions *F(x)* and *G(y)*, respectively.

Simple random sample (SRS) — is a subset of individuals (*a sample*) chosen from a larger set (*a population*) in which a subset of individuals are chosen randomly, all with the same probability. It is a process of selecting a sample in a random way. In **SRS**, each subset of k individuals has the same probability of being chosen for the sample as any other subset of k individuals.A simple random sample is an unbiased sampling technique. Equivalent definition: a sample

$x^m = (x_1, x_2, \ldots, x_m)$ is simple if the values $(x_1, x_2, \ldots, x_m)$ are realizations of m independent equally distributed random variables. In other words, the selection of observations is not only random but also does not imply any special selection rules (e.g., choosing every 10th observation).

The U-test — is a nonparametric criterion to test the null hypothesis that for randomly chosen from two samples of observations $x \in X$ and $y \in Y$ the probability that x is greater than y is equal to the probability that y is greater than x. In mathematical language, the null hypothesis is written as follows

$$H_0 : P\{x < y = \frac{1}{2}\}. \tag{3.1}$$

For the test's own consistency, an alternative hypothesis is required, which is that the probability that the value of a characteristic of observation from X is greater than that of observation from Y differs upward or downward from the probability that the value of a characteristic of observation from Y is greater than that of observation from X. In mathematical language, the alternative hypothesis is written as follows

$$H_1 : P\{x < y\} \neq P\{y < x\} \vee P\{x < y\} + 0.5 \cdot P\{x = y\} \neq 0.5. \tag{3.2}$$

According to the basic concept of the U-test, if the null hypothesis is true, the distribution of the two samples is continuous; if the alternative hypothesis is true, the distribution of one sample is stochastically greater than the distribution of the other. In this case, it is possible to formulate a number of null and alternative hypotheses for which this test will give a correct result. His most extensive generalization lies in the following assumptions:

- the observations in both samples are independent;
- the data type is at least ranked, i. e., with respect to any two observations you can tell which one is greater;
- the null hypothesis assumes that the distributions of the two samples are equal;
- the alternative hypothesis assumes that the distributions of the two samples are unequal.

With a stricter set of assumptions than those given above, for example the assumption that the distribution of the two samples is continuous if the null hypothesis is valid and that the distribution of the two samples has a shift in the distribution if the alternative one is valid i. e. $f_1(x) = f_2(x + \sigma)$, we can say that the U-test is a test for the hypothesis of equality of medians. In this case, the U-test can be interpreted as a test of whether Hodges–Lehman's estimate of the difference in central tendency measures differs from zero. In this situation, the Hodges–Lehman estimate is the median of all possible values of differences between the observations in the first and second samples. However, if both

the variance and the shape of the distribution of the two samples differ, the U-test cannot correctly test the medians. Examples can be shown where the medians are numerically equal and the test rejects the null hypothesis because of the small p-value. Thus, a more correct interpretation of the U-test is to use it to test the shift hypothesis [39].

Shift hypothesis — is a statistical hypothesis often considered as an alternative to the hypothesis of complete homogeneity of samples. Let us have two samples of data. Let us also give two random variables X and Y, which are distributed as elements of these samples and have distribution functions *F(x)* and *G(y)*, respectively. In these terms, the shift hypothesis can be written as follows

$$H : F(x) = G(x + \sigma) \ : \forall x, \ \sigma \neq 0. \tag{3.3}$$

In this case, the U-criterion is valid regardless of the characteristics of the samples.

Simply put, the essence of the U-test is that it allows us to answer the question of whether there is a significant difference in the value of the quantitative attribute of the two samples. With regard to valuation, we can say that the use of this test helps to answer the question of whether it is necessary to take into account one or another attribute as a price-generating factor. It follows from the above that by default we are talking about a two-sided test. In practice, this means that the test does not give a direct answer to the question, for example: "Is there a significant excess of the unit value of premises with a separate entrance to the premises that do not have it. At the same time, there are also one-sided realizations that allow us to answer the question about the sign of the difference in the value of the attribute in the two samples.

In addition to the above requirements for the samples themselves, the conditions for applying the U-test are:

- the distribution of quantitative attribute values of samples is different from normal (otherwise it is advisable to use parametric Student's t-test or z-test for independent samples).
- at least three observations in each sample, it is allowed to have two observations in one of the samples, provided that there are at least five in the other sample.

To summarize the above, there are three variants of the null hypothesis, depending on the level of rigor outlined in the table below 3.1.

3.2 Test implementation

3.2.1 Test statistic

Suppose that the elements $x_1, \ldots, x_n$ represent a simple independent sample from $X \in \mathbb{R}$, and the elements $y_1, \ldots, y_n$ represent a simple independent sample from $Y \in \mathbb{R}$ and the samples are independent of each other. Then the relevant U-statistic is defined

Table 3.1: Variants of the null hypothesis when using the U-test in valuation.

Type of hypothesis	Formulation
Scientific	The two samples are completely homogeneous, i. e. they belong to the same distribution, there is no shift and the estimate made for the first sample is unbiased for the second one.
Practical	The medians of the two samples are equal to each other.
Set forth in terms of valuation	The difference in the attribute between the two samples of object-analogues is not significant, its accounting is not required and this attribute is not a pricing factor.

as follows:

$$U = \sum_{i=1}^{m} \sum_{j=1}^{n} S(x_i, y_j),$$

$$S(x, y) = \begin{cases} 1, & x > y, \\ \frac{1}{2}, & x = y, \\ 0, & x < y. \end{cases} \tag{3.4}$$

3.2.2 Calculation methods

The test involves calculating a statistic usually called the U-statistic whose distribution is known if the null hypothesis is true. When working with very small samples, the distribution is specified tabularly; when the sample size is more than twenty observations, it is approximated quite well by the normal distribution. There are two methods of calculating U-statistics: manual calculation using the formula 3.4 or using a special algorithm. The first method, due to its labor-intensive nature, is only suitable for very small samples. The second method can be formalized as a step-by-step set of instructions and will be described below.

1. You must construct a common variation series for the two samples and then assign a rank to each observation, starting with one for the smallest of them. If there are ties, i. e. groups of repeating values (such a group can be, e. g., only two equal values), each observation from such a group is assigned a value equal to the median of the group ranks before adjustment (for example, in the case of a variation series (*3, 5, 5, 5, 5, 8*) the ranks before adjustment are (*1, 2, 3, 4, 5, 6*) after — (*1, 3.5, 3.5, 3.5, 3.5, 6*).

2. It is necessary to calculate the sums of the ranks of the observations of each sample, denoted as R_1, R_2 respectively. In this case, the total sum of ranks can be calculated by the formula

$$R = \frac{N(N+1)}{2}, \tag{3.5}$$

where N —the total number of observations in both samples.

3. Next, we calculate the U-value for the first sample:

$$U_1 = R_1 - \frac{n_1(n_1+1)}{2}, \tag{3.6}$$

where R_1 —the sum of ranks of the first sample, n_1 — the number of observations in the first sample.

4. The U-value for the second sample is calculated in the same way:

$$U_2 = R_2 - \frac{n_2(n_2+1)}{2}, \tag{3.7}$$

where R_2 —the sum of ranks of the second sample, n_2 — the number of observations in the second sample.

From the above formulas it follows that

$$U_1 + U_2 = R_1 - \frac{n_1(n_1+1)}{2} + R_2 - \frac{n_2(n_2+1)}{2}. \tag{3.8}$$

It is also known that

$$\begin{cases} R_1 + R_2 = \dfrac{N(N+1)}{2} \\ N = n_1 + n_2. \end{cases} \tag{3.9}$$

Then

$$U_1 + U_2 = n_1 n2. \tag{3.10}$$

Using this formula as a control ratio can be useful for checking the correctness of calculations in a spreadsheet processor.

5. From the two values of U_1, U_2 in all cases we choose the smaller which will be the U-statistic and used in further calculations. Let us denote it as U.

3.2.3 Interpretation of the result

For a correct interpretation of the test result it is necessary to specify:

- size of each sample;
- values of the measure of central tendency for each sample (given the nonparametric nature of the test, the median appears to be the appropriate measure of central tendency);
- the value of the U-statistic itself;
- the CLES index [53] the value of which is equivalent to the AUC and ρ-statistic;
- rank-biserial correlation coefficient (RBC) [72];

- the accepted level of significance (usually 0.05);
- the calculated p-value.

The concept of U-statistic was discussed earlier and most of the other indicators are widely known and do not require any particular consideration.

3.2.3.1 CLES = ρ-statistic = AUC

First of all, it must be said that all of these indicators are equivalent to each other. Thus

$$CLES = f = AUC_1 = \rho. \tag{3.11}$$

3.2.3.1.1 Common language effect size (CLES)

Common language effect size (CLES) — is the probability that the value of a randomly chosen observation from the first sample is greater than the value of a randomly chosen observation from the second sample. This indicator is calculated by the formula

$$CLES = \frac{U_1}{n_1 n_2}. \tag{3.12}$$

The designation *f (favorable)* is often used instead of *CLES*. This sample value is an unbiased estimate of the value for the entire population of objects belonging to the set.

It should be noted that the value and meaning of this indicator is equivalent to the value and meaning of the AUC[73]. Thus, we can say that this indicator characterizes the quality of the U-test as a binary classifier.

$$CLES = f = AUC_1 = \frac{U_1}{n_1 n_2}. \tag{3.13}$$

The relationship between the U-statistic and AUC is discussed in 3.4.

3.2.3.1.2 ρ-statistic A statistic called ρ that is linearly related to U and widely used in studies of categorization (discrimination learning involving concepts), and elsewhere, is calculated by dividing U by its maximum value for the given sample sizes, which is simply $n1 \times n2$. Thus, ρ is a non-parametric measure of the overlap between two distributions; it can take values between 0 and 1, and it is an estimate of $P(Y > X) + 0.5P(Y = X)$, where X and Y are randomly chosen observations from the two distributions. Both extreme values represent complete separation of the distributions, while a $= 0.5$ represents complete overlap. This statistic is useful in particular when despite a large p-value the medians of the two samples are essentially equal to each other.

3.2.3.2 Rank-biserial correlation (RBC)

The method of representing the measure of impact for the U-test is to use a measure of rank correlation known as rank-biserial correlation (hereafter RBC). As in the case of other measures of correlation, the value of the RBC coefficient has a range of values [-1;1], with a zero value indicating the absence of any relationship. The RBC coefficient is usually denoted as r. A simple formula based on the CLES (AUC, t, ρ) value is used to calculate it. Let us state the hypothesis that in a pair of random observations, one of which is taken from the first sample and the other from the second, the value of the first is greater.Let's write it down in mathematical language:

$$H : x_i > y_j, \quad x \ \in X, \ y \in Y. \tag{3.14}$$

Then the value of the RBC coefficient is the difference between the proportion of random pairs of observations that are favorable (f) to the hypothesis and the complementary proportion of random pairs that are unfavorable to the hypothesis. Thus, this formula is a formula for the difference between the CLES scores for each of the groups.

$$r = f - u = CLES_1 - CLES_2 = f - (1 - f) \tag{3.15}$$

There are also a number of alternative formulas that give identical results:

$$r = 2f - 1 = \frac{2U_1}{n_1 n_2} - 1 = 1 - \frac{2U_2}{n_1 n_2}. \tag{3.16}$$

3.2.4 Calculation of the p-value and the final test of the null hypothesis

If the number of observations in both samples is large enough, the U-statistic has an approximately normal distribution. Then its standardized value (z-score) [78] can be calculated by the formula

$$z = \frac{U - m_U}{\sigma_U}, \tag{3.17}$$

where m_U is mean for U and σ_U is its standard deviation. A visualization of the concept of *standardized value for a normal distribution* is shown in Figure 3.1. The mean for the U is calculated by the formula

$$m_U = \frac{n_1 n_2}{2}. \tag{3.18}$$

The formula for the standard deviation in the case of no ties is as follows:

$$\sigma_U = \sqrt{\frac{n_1 n_2 (n_1 + n_2 + 1)}{12}}. \tag{3.19}$$

In case of the presence of tied ranks, a different formula is used:

$$\sigma_{U_{ties}} = \sqrt{\frac{n_1 n_2 (n_1 + n_2 + 1)}{12} - \frac{n_1 n_2 \sum_{k=1}^{K} ({t_k}^3 - t_k)}{12n(n-1)}} = \sqrt{\frac{n_1 n_2}{12}\left((n+1) - \frac{\sum_{k=1}^{K} ({t_k}^3 - t_k)}{n(n-1)}\right)}, \tag{3.20}$$

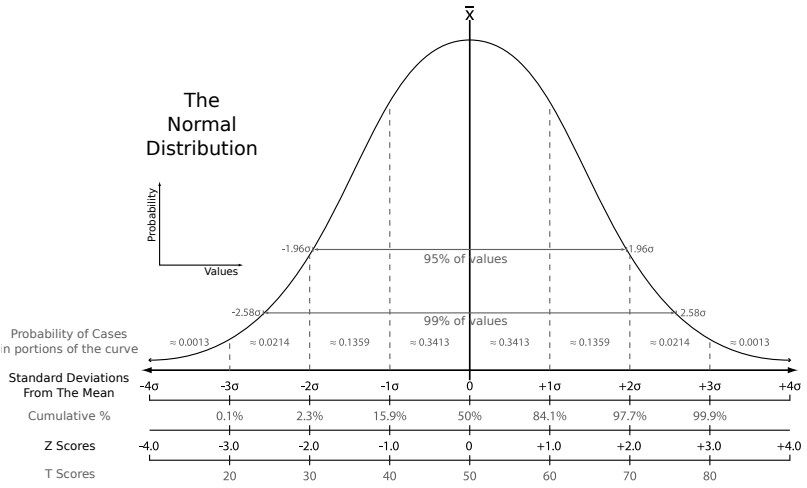

Figure 3.1: A visualization of the concept of standardized value for a normal distribution [78]

where t_k is the number of observations with rank k and K is the total number of tied ranks. Then, by obtaining a standardized value (z-score) and using an approximation of the standard normal distribution, the p-value for a given level of significance (usually 0.05) is calculated. The interpretation of the result is as follows:

$$\begin{aligned} p \leq 0.05 &\Rightarrow \text{the null hypothesis is rejected} \\ p > 0.05 &\Rightarrow \text{the null hypothesis can not be rejected.} \end{aligned} \tag{3.21}$$

However, there is also an alternative interpretation:

$$\begin{aligned} p < 0.05 &\Rightarrow \text{the null hypothesis is rejected} \\ p \geq 0.05 &\Rightarrow \text{the null hypothesis can not be rejected.} \end{aligned} \tag{3.22}$$

To date, there is no unambiguous position on how the situation when $p = \alpha$ should be interpreted. This paper uses the version described in 3.21.

3.3 Relationship to other statistical tests

3.3.1 Comparison of Wilcoxon-Mann-Whitney U-test with Student's t-test

You often hear that the U-test is the nonparametric counterpart of the Student's t-test, designed for data whose distribution differs from the normal one. From a purely practical

point of view, we can indeed say that in the case of a normal distribution it is advisable to determine whether there is a significant difference between the two samples by means of the t-test, and in the case of a distribution that differs from the normal by means of the U-test. Thus, it can be said that these tests are used for the same ultimate purpose.

However, the mathematical meaning of the U-test and the t-test are significantly different. As stated earlier, the U-test is designed to test the null hypothesis, which is that for randomly chosen from two samples of observations $x \in X$ and $y \in Y$ the probability that x is greater than y is equal to the probability that y is greater than x, the alternative hypothesis carries the claim that these probabilities are not equal. At the same time, the t-test is designed to test the null hypothesis that the means of the two samples are equal, while the alternative hypothesis is that the means of the two samples are not equal. In this regard, when comparing these tests, we should keep in mind that, in general, the U-test and the t-test check different null hypotheses, although they have partly similar practical meaning. The result of the U-test is most often very close to the result of the two-sample t-test for ranked data. Table 3.2 then provides a general comparison of the U-test with the t-test.

Table 3.2: Properties of the U-test relative to the t-test.

Property	Description
Applicability to ordinal data	When working with ordinal (rank) data, rather than quantitative data, the U-test is preferable to the t-test, remembering that the distance between neighboring values of the variation series cannot be considered constant.
Robustness	Since the U-test handles the sum of ranks rather than trait values, it is less likely than the t-test to erroneously indicate significance due to outliers. However, in general, the U-test is more prone to type I error in the case when the data simultaneously have the property of heteroscedasticity and have a distribution other than normal.
Efficiency	In the case of a normal distribution, the asymptotic efficiency of the U-test is $\frac{3}{4}\pi \approx 0.95$ of the t-test [8]. If the distribution differs significantly from the normal one and the number of observations is large enough, the efficiency of the U-test is significantly higher than the efficiency of the t-test [6]. However, this efficiency comparison should be interpreted with caution, because the U-test and the t-test examine different hypotheses and estimate different values. In the case, for example, of the need to compare means, the use of the U-test is not justified in principle.

3.3.2 Alternative tests in the case of inequality of distributions

If it is necessary to test the stochastic ordering of two samples (i.e. the alternative hypothesis: $H1: \ P(Y > X) + 0.5P(Y = X) \neq 0.5$) without assuming equality of their distributions (i.e. when the null hypothesis is $H0: \ P(Y > X) + 0.5P(Y = X) = 0.5$ but not $F(X) = G(Y)$), more appropriate tests should be used. These include the Brunner-Munzel test [16], which is a heteroskedasticity-resistant analog of the U-test, and the Fligner-Policello test [28], which is a test for equality of medians. In particular, in the case of a more general null hypothesis $H0: \ P(Y > X) + 0.5P(Y = X) = 0.5$, the U-test can often lead to a type I error even in the case of large samples (especially in the case of disparity of variance and significantly different sample sizes), so that in such cases the use of alternative tests is preferable [17]. Thus, in the absence of the assumption of equality of distributions in case the null hypothesis is valid, the use of alternative tests will be preferable.

In the case of testing the hypothesis of a shift with significantly different distributions, the U-test may give an erroneous interpretation of significance [9], so in such circumstances it is preferable to use a variant of the t-test [50] designed for cases of unequal variance [9]. In some cases, it may be justified to convert quantitative data into ranks and then perform the t-test in some variant depending on the assumption of equality of variance. When converting quantitative data to ordinal data, the original variances will not be preserved; they must be recalculated for the ranks themselves. In the case of equal variance, a suitable nonparametric substitute for the F-test [27] can be the Brown-Forsythe test [5].

3.3.3 The relationship between the U-test and the classification tasks

The U-test is a particular case of the ordered logit model [41].

3.4 The relationship between the U-test and the concepts of Receiver operating characteristics (ROC) and Area under the curve (AUC)

Based on what was said in 3.3.3, we can conclude that the U-test is not only a test for testing the shift hypothesis (or another one similar in meaning), but also represents a kind of classifier. Looking ahead, the meaning of the U-test as a classifier is as follows:

- there is a "positive" outcome of comparing two random observations, which is that the observation from X is greater than the observation from Y;
- the proportion of the sum of the ranks of the "positive" elements is calculated.
- as in general with ROC, if the value of the share of "positive" elements exceeds 0.5, this means that the classifier generally performs its function; if it is equal to 0.5, its efficiency is equal to guessing with a coin flip; if it is less than 0.5, using such classifier yields the opposite result.

At first glance, the relationship between the U-test and ROC does not seem obvious. This section will attempt to understand why these concepts are related and what is the essence of the U-test as a classifier.

ROC analysis itself is outside the scope of this paper. Therefore, let us consider only its main points.

ROC curve — is a graphical plot that allows us to evaluate the quality of binary classification. It displays the ratio between the proportion of objects from the total number of feature carriers correctly classified as carrying the feature (True Positive Rate (TPR), called the *sensitivity of the classification algorithm*) and the proportion of objects from the total number of objects not carrying the feature, incorrectly classified as carrying the feature (False Positive Rate (FPR), the **1-FPR** value is called the *specificity of the classification algorithm*), when varying the threshold of the deciding rule. It is also known as **error curve**. Analysis of classifications using ROC curves is called **ROC analysis**.

Quantitative interpretation of the ROC curve gives the Area under the curve (AUC).

Area under the curve (AUC) — is the area bounded by the ROC curve and the axis of the proportion of false positive classifications (abscissa axis).

The higher the AUC, the better the quality of the classifier, while a value of 0.5 demonstrates the unsuitability of the chosen classification method (corresponding to a random coin guessing). A value of less than 0.5 indicates that the classifier works exactly the other way around: if you call positive results negative and vice versa, the classifier will perform better [73].

Let's introduce some terms.

Condition positive (P) — the number of real positive cases in the data.

Condition negative (N)— the number of real negative cases in the data.

True positive (TP) — a test result that correctly indicates the presence of a condition or characteristic.

True negative (TN) — a test result that correctly indicates the absence of a condition or characteristic.

False positive (FP) — a test result which wrongly indicates that a particular condition or attribute is present.

False negative (FN) — a test result which wrongly indicates that a particular condition or attribute is absent.

Based on the above, we can create a contingency table of the results of applying the binary classifier. The rows contain data on the actual presence or absence of the feature, the columns on the predicted with the classifier. As can be seen from Table 3.3, the binary classifier can lead to errors of two types. Let's introduce some more definitions and

Table 3.3: Binary classifier contingency table.

Total $P + N$	Predicted Positive (PP)	Predicted negative (PN)
Positive (P)	TP	FN, type II error [79]
Negative (N)	FP, type I error [79]	TN

Table 3.4: Additional definitions and formulas for calculating the probabilities of binary classifier outcomes (part 1 of 3).

Notation	Formula	Deciphering the notation and alternative terms.
TPR (SEN)	$TPR = \frac{TP}{P} = 1 - FNR = \frac{TP}{TP + FN}$ (3.23)	**true positive rate**, **sensitivity** [74], recall [70], probability of detection, hit rate [60], power
FPR	$FPR = \frac{FP}{N} = 1 - TNR = \frac{FP}{FP + TN}$ (3.24)	**false positive rate**, probability of false alarm, fall-out [58]
FNR	$FNR = \frac{FN}{P} = 1 - TPR = \frac{FN}{FN + TP}$ (3.25)	**false negative rate** [80], miss rate
TNR (SPC)	$TNR = \frac{TN}{N} = 1 - FPR = \frac{TN}{TN + FP}$ (3.26)	**true negative rate**, **specificity**, selectivity [74]
PPV	$PPV = \frac{TP}{TP + FP} = 1 - FDR$ (3.27)	**positive predictive value** [69], precision [61]
NPV	$NPV = \frac{TN}{TN + FN} = 1 - FOR$ (3.28)	**negative predictive value** [69]
FDR	$FDR = \frac{FP}{FP + TP} = 1 - PPV$ (3.29)	**false discovery rate** [57]

Table 3.5: Additional definitions and formulas for calculating the probabilities of binary classifier outcomes (part 2 of 3).

Notation	Formula	Deciphering the notation and alternative terms.
FOR	$FOR = \frac{FN}{FN+TN} = 1 - NPV$ (3.30)	**false omission rate** [69]
LR+	$LR+ = \frac{TPR}{FPR}$ (3.31)	**positive likelihood ratio** [66]
LR-	$LR- = \frac{FNR}{TNR}$ (3.32)	**negative likelihood ratio** [66]
PT	$PT = \frac{\sqrt{TPR(-TNR+1)}+TNR-1}{TPR+TNR-1} = \frac{\sqrt{FPR}}{\sqrt{TPR}+\sqrt{FPR}}$ (3.33)	**prevalence threshold** [74]
TS (CSI)	$TS = \frac{TP}{TP+TN+FP}$ (3.34)	Jaccard index **threat score**, **critical success index** [62]
PRV	$PRV = \frac{P}{P+N}$ (3.35)	**prevalence** [71]
ACC	$ACC = \frac{TP+TN}{P+N} = \frac{TP+TN}{TP+TN+FP+FN}$ (3.36)	**accuracy** [51]

define the formulas for calculating the probabilities of its outcomes (see tables 3.4–3.6).
The TPR probability can be written as

$$P_{TPR} = \mathbb{P}(1,\ x \in C_1), \tag{3.44}$$

which means that if object x belongs to class C_1, this indicator estimates the probability that the binary classifier assigns object x to this class. The probability of FPR is written as

$$P_{FPR} = \mathbb{P}(1,\ x \in C_0), \tag{3.45}$$

which means the probability that an object belonging to class C_0 will be mistakenly assigned to class C_1.

Typically, the working principle of a binary classifier is based on comparing the measurement of x with some fixed threshold c. It follows that the previous two expressions can be rewritten and combined into a system.

$$\begin{cases} P_{TPR} = \mathbb{P}(x > c,\ x \in C_1) \\ P_{FPR} = \mathbb{P}(x > c,\ x \in C_0) \end{cases} \tag{3.46}$$

It follows that the ROC curve is a diagram

$$P_{FPR}(c),\ P_{TPR}(c), \tag{3.47}$$

Table 3.6: Additional definitions and formulas for calculating the probabilities of binary classifier outcomes (part 3 of 3).

Notation	Formula	Deciphering the notation and alternative terms.
BA	$$BA = \frac{TPR + TNR}{2} \quad (3.37)$$	**balanced accuracy**
F1 score	$$F_1 = 2 \times \frac{PPV \times TPR}{PPV + TPR} = \frac{2TP}{2TP + FP + FN} \quad (3.38)$$	F1 score is the harmonic mean of precision and sensitivity [56]
MCC (ϕ or r_ϕ)	$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (3.39)$$	**Matthews correlation coefficient**,**phi coefficient** [68]
FM	$$FM = \sqrt{\frac{TP}{TP + FP} \times \frac{TP}{TP + FN}} = \sqrt{PPV \times TPR} \quad (3.40)$$	Fowlkes–Mallows index [59]
BM	$$BM = TPR + TNR - 1 \quad (3.41)$$	**bookmaker informedness**, informedness [81]
MK (δP)	$$MK = PPV + NPV - 1 \quad (3.42)$$	**markedness**, deltaP [67]
DOR	$$\frac{LR+}{LR-} \quad (3.43)$$	**diagnostic odds ration** [54]

thus, drawing the curve means changing the value of threshold c.

Let's consider the example [25]. Let's take $f(x \in C_0) = \mathcal{N}(0, 1)$ and $f(x \in C_1) = \mathcal{N}(2, 1)$ as probability density functions C_0 and C_1, respectively. Next we build the ROC curve step by step using the Python language. At the first step, consider diagram 3.2, built using the code given in script 3.1. The area shaded blue shows the probability of FPR, i. e., false-positive significance detection, while the area shaded green shows the probability density of TPR, i. e., correct significance detection. The ROC curve shows the values of these very indicators. The vertical dashed line is the sensitivity threshold c. In this situation it is at 0 on the abscissa axis. If it is moved to 1, the area under the FPR curve (blue) will significantly decrease, i. e. the probability of false-positive detection will decrease, but the TPR area (green) will decrease as well, which means an increase in the probability of false-negative results. This situation is illustrated in Diagram 3.3.

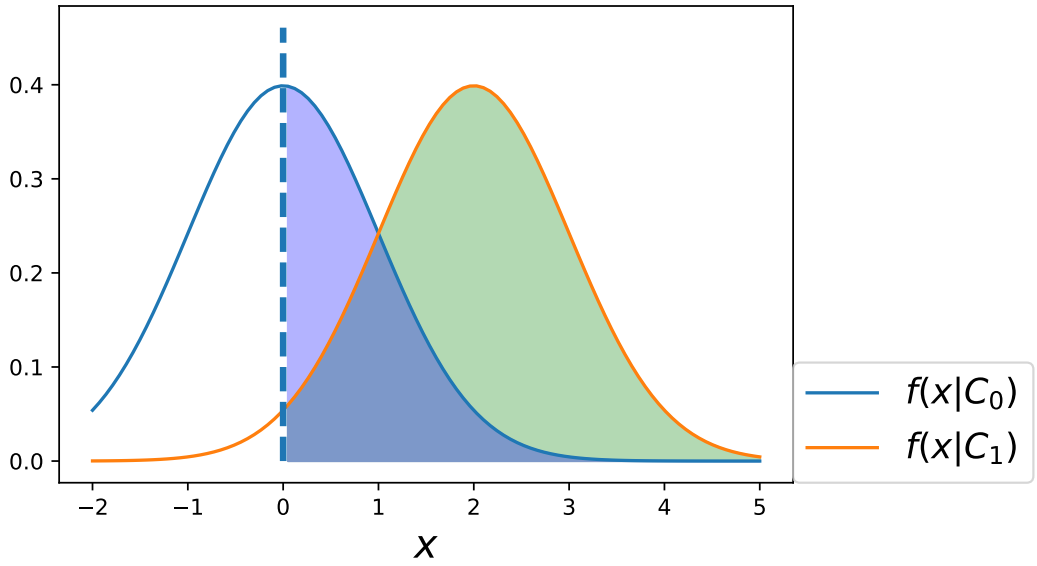

Figure 3.2: Diagram of TPR and FPR probability distribution densities at threshold 0.

As you can see from the diagrams above, increasing the threshold leads to the loss of a part of both true-positive and false-positive results, while decreasing it leads to an increase in the number of fixations of the feature presence (both true and false). In extreme cases, too low a threshold value will lead to the fact that all results will be interpreted as positive, too high — to a zero number of observations in which the feature was detected. The task of ROC analysis is to choose a rational threshold value.

Let's add the ROC curves corresponding to thresholds 0 and 1 to the already existing diagrams. And also create an interactive diagram using the code from script 3.2. The PDF format does not allow you to add such interactive elements, so let's consider cases with fixed values of 0 and 1, shown in Diagrams 3.4 and 3.5, respectively. The left side of each of them shows the already familiar probability density function graphs for the TPR and FPR distributions. The right part shows the ROC curve and the point

Listing 3.1: Plotting TPR and FPR probability density functions

```
# Import Libraries
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

# Plot
f0 = stats.norm(0, 1)
f1 = stats.norm(2, 1)
fig, ax = plt.subplots()
xi = np.linspace(-2, 5, 100)
ax.plot(xi, f0.pdf(xi), label=r'$f(x|C_0)$')
ax.plot(xi, f1.pdf(xi), label=r'$f(x|C_1)$')
ax.legend(fontsize=16, loc=(1, 0))
ax.set_xlabel(r'$x$', fontsize=18)
ax.vlines(0, 0, ax.axis()[-1] * 1.1, linestyles='--', lw=3.)
ax.fill_between(xi, f1.pdf(xi), where=xi > 0, alpha=.3, color='g')
ax.fill_between(xi, f0.pdf(xi), where=xi > 0, alpha=.3, color='b')

# Save to .pdf
plt.savefig('Plot-ROC-step-1.pdf', bbox_inches='tight')
```

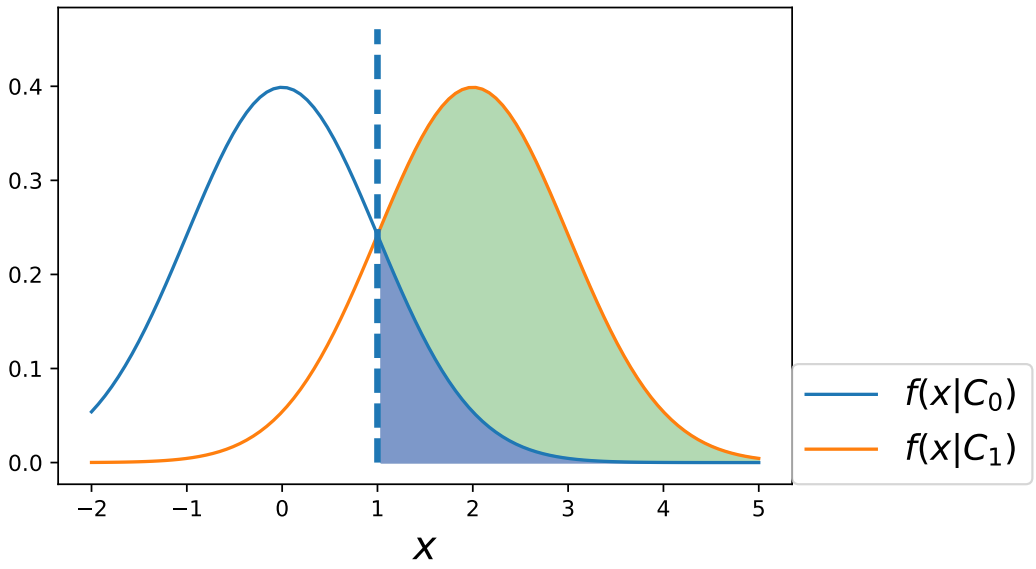

Figure 3.3: Diagram of TPR and FPR probability distribution densities at threshold 1.

corresponding to the set threshold value. It is easy to guess that the x-coordinate of the point matches the area under the FPR curve, and the y-coordinate matches the area under the TPR curve. Increasing the threshold value entails shifting the point to the left, decreasing it to the right.

The better the binary classifier itself, the closer to the upper left corner will be the ROC curve corresponding to it, because in this case a high TPR value will be combined with a low FPR value. The binary classifier, which works as well (actually badly) as the coin flip guessing algorithm (in case the coin is "fair"), gives a ROC curve, which is a straight line between (0,0) and (1,1). In this case, the left part of the diagram will show a complete overlap of TPR and FPR probability density function curves. Such a case is shown in Diagram 3.4. For self-practice, you can use Script 3.2 by running it in the Jupyter Lab environment, which allows you to use the interactive features of the browser.

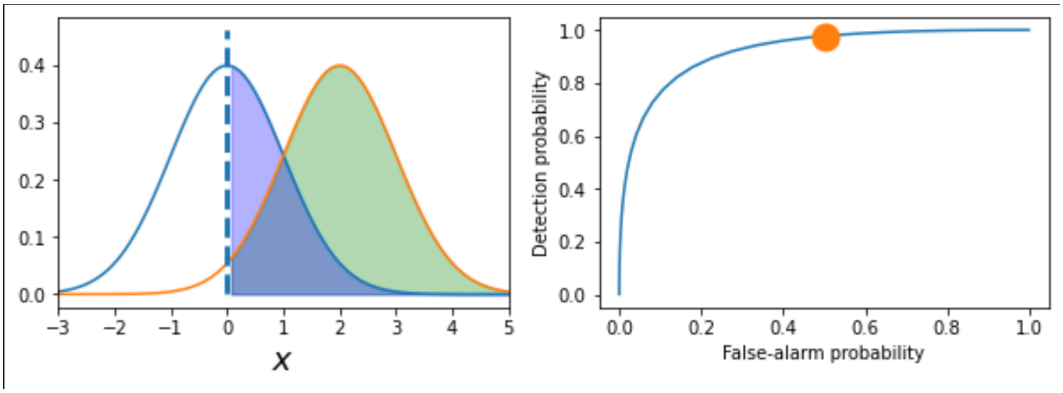

Figure 3.4: Diagram of TPR and FPR probability distribution densities at threshold 0.

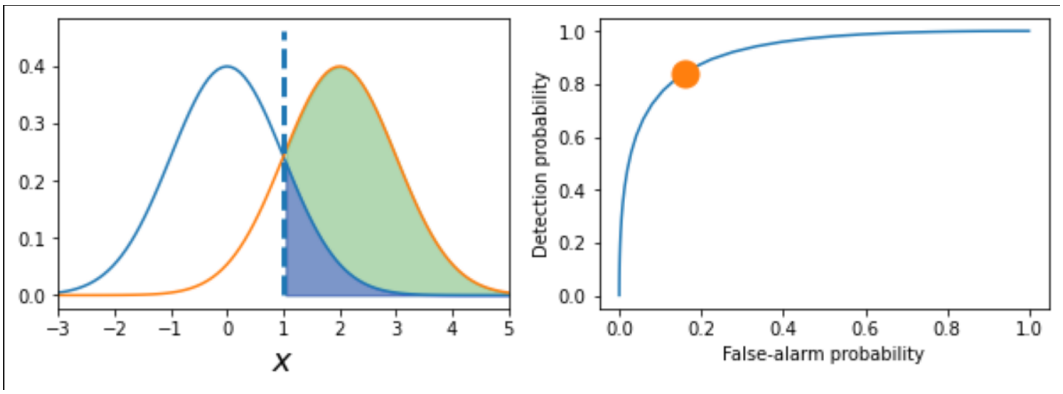

Figure 3.5: Diagram of TPR and FPR probability distribution densities at threshold 1.

Listing 3.2: Build an interactive graph of TPR and FPR distribution density and its corresponding ROC curve for a given threshold value

```
# Import Libraries
%matplotlib inline
from ipywidgets import interact
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

# Plot
f0 = stats.norm(0, 1)
f1 = stats.norm(2, 1)
fig, ax = plt.subplots()
xi = np.linspace(-2, 5, 100)
ax.plot(xi, f0.pdf(xi), label=r'$f(x|C_0)$')
ax.plot(xi, f1.pdf(xi), label=r'$f(x|C_1)$')
ax.legend(fontsize=16, loc=(1, 0))
ax.set_xlabel(r'$x$', fontsize=18)
ax.vlines(0, 0, ax.axis()[-1] * 1.1, linestyles='--', lw=3.)
ax.fill_between(xi, f1.pdf(xi), where=xi > 0, alpha=.3, color='g')
ax.fill_between(xi, f0.pdf(xi), where=xi > 0, alpha=.3, color='b')

# Plot ROC-curve and make all interactive
def plot_roc_interact(c=0):
xi = np.linspace(-3,5,100)
fig,axs = plt.subplots(1,2)
fig.set_size_inches((10,3))
ax = axs[0]
ax.plot(xi,f0.pdf(xi),label=r'$f(x|C_0)$')
ax.plot(xi,f1.pdf(xi),label=r'$f(x|C_1)$')
ax.set_xlabel(r'$x$',fontsize=18)
ax.vlines(c,0,ax.axis()[-1]*1.1,linestyles='--',lw=3.)
ax.fill_between(xi,f1.pdf(xi),where=xi>c,alpha=.3,color='g')
ax.fill_between(xi,f0.pdf(xi),where=xi>c,alpha=.3,color='b')
ax.axis(xmin=-3,xmax=5)
crange = np.linspace(-3,5,50)
ax=axs[1]
ax.plot(1-f0.cdf(crange),1-f1.cdf(crange))
ax.plot(1-f0.cdf(c),1-f1.cdf(c),'o',ms=15.)
ax.set_xlabel('False-alarm probability')
ax.set_ylabel('Detection probability')

interact(plot_roc_interact,c=(-3,5,.05))
```

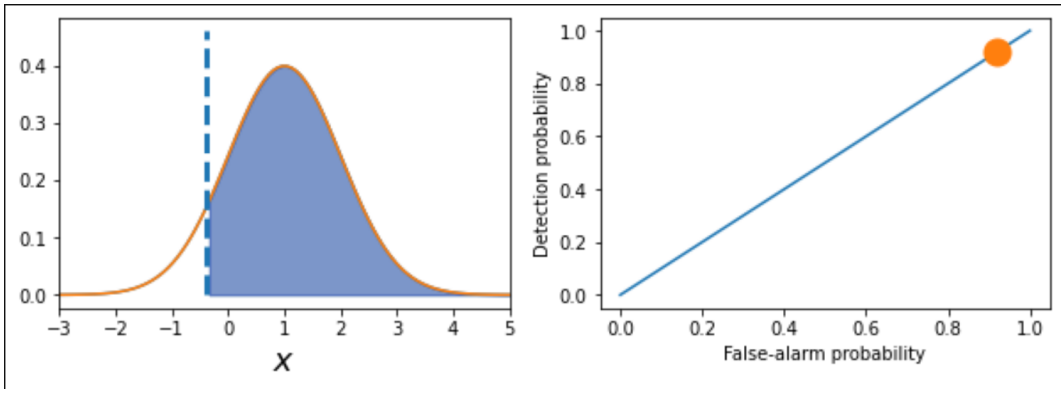

Figure 3.6: Diagram of probability densities of TPR and FPR probability distributions at equal mean.

3.4.1 The concept of AUC and its calculation

As the name implies, the AUC is the area under the ROC curve bounded by the point corresponding to a given threshold value. In the normalized space in which the ROC curve is usually plotted, the AUC value is equivalent to the probability that the classifier assigns a higher weight to a randomly chosen positive entity than to a randomly chosen negative entity. The AUC does not depend on a specific threshold value, because the ROC curve is constructed by fitting it. This means that the AUC is calculated by integrating over the thresholds. The AUC is given by the expression:

$$AUC = \int P_{TPR}(P_{FPR})dP_{FPR}. \tag{3.48}$$

The step-by-step calculation of the AUC is as follows.

$$P_{TPR}(c) = 1 - F_1(c), \tag{3.49}$$

where F_1 is the cumulative density function for C_1. Similarly calculate

$$P_{FPR}(c) = 1 - F_0(c), \tag{3.50}$$

where F_0 is the cumulative density function for C_0.

Let us take some particular value of c^* to which a certain $P_{FPR}(c^*)$ corresponds. In other words, it corresponds to the probability that a random element x_0 belonging to class C_0 is greater than the threshold value c^*, i.e.

$$P_{FPR}(c^*) = \mathbb{P}(x_0 > c^* | x_0 \in C_0). \tag{3.51}$$

Then, reasoning similarly with respect to TPR, we get

$$P_{TPR}(c^*) = \mathbb{P}(x_1 > c^* | x_1 \in C_1). \tag{3.52}$$

Next, based on the fact that the AUC is realized through an integral, we select its value so that the distribution of c^* matches the distribution of F_0. In this case, P_{TPR} is an independent random variable with a corresponding expectation in the form of

$$\mathbb{E}(P_{TPR}) = \int P_{TPR} dP_{FPR} = AUC. \tag{3.53}$$

It is now possible to formulate a definition for the AUC.

AUC — is the expected probability that element $x_1 \in C_1$ will be assigned to C_1 with higher probability than element $x_0 \in C_0$. Thus,

$$1 - F_1(t) > 1 - F_0(t) \forall t. \tag{3.54}$$

The wording "for any t" means that $1-F_1(t)$ is *stochastically* greater than $1-F_0(t)$. The latter circumstance is key in terms of the relationship of the AUC to the U-test, which will be shown later.

3.4.2 Relation between U-test and AUC

A fairly detailed description of the U-test was given earlier. This subsection contains only brief information about it, which is directly relevant to the question of its relationship to the AUC.

The U-test is a non-parametric test that allows you to test whether two samples belong to the same distribution. His basic idea is that if there is no difference between two classes, then combining them into one larger class (set) and then calculating any statistic for the new larger class will give an unbiased estimate for any of the initial classes. In other words, if there is no difference in the distribution of the two samples, combining them and assuming that the actually observed data from the two samples represent only one of the equal-valued variants of the moving observations means that there is no difference in any statistical estimate for any of the moving variants relative to the other, and relative to the combined set.

Let's suppose that we need to compare two samples using the median, the mean, or some other measure of central tendency. In terms of cumulative distribution functions for the two populations, in the case of $H0$ we have the following:

$$H_0 : F_X(t) = F_Y(t), \quad \forall t, \tag{3.55}$$

which indicates that all observations belong to the same distribution. Then an alternative hypothesis is that

$$H_1 : F_X(t) < F_Y(t), \quad \forall t, \tag{3.56}$$

which is possible, in particular, in the case of the existence of a shift of one distribution relative to the other. In this case, the samples $X_{ii=1}^{n}$, $X_{jj=1}^{m}$ represent independent groups of observations. In this case, the size of the samples may vary.

The test technique consists of combining two samples into one set and assigning ranks to each item within it. The U-statistic is the sum of the ranks for the set X. If the value

Listing 3.3: Calculation of the p-value for the test data

```
print('p-value:',stats.wilcoxon(f1.rvs(30), f0.rvs(30))[1])
```

of the statistic is small enough, it means that the distribution of set X is stochastically shifted to the left relative to the distribution of set Y, i. e. $F_X t < F_Y t$.

Since with a sufficiently large number of observations (20 or more) the distribution of U-statistics is well approximated by the normal distribution, the p-value is suitable for assessing significance. Let's calculate it using the Python language according to the script 3.3. The p-value is 1.9729484515803686e-05, which is less than the significance level (0.05), so we can reject the null hypothesis 3.55. Since the data were randomly generated, if the experiment is repeated, the particular p-value will differ from that obtained when writing this paper. However, it will always be below the threshold because of the parameters set in the algorithm.

The U-statistic can be written as follows:

$$U = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbb{1}(Y_j > X_i), \tag{3.57}$$

where $\mathbb{1}(Y_j > X_i)$ is the indicator (characteristic) function showing that the statistic (for the discrete case) estimates the probability that Y is stochastically greater than X. Thus, this correspondence means that its value is equal to the AUC. The relationship between the AUC and the U-test is in a similar sense: checking the stochastic excess value of observations belonging to one sample relative to observations belonging to another sample.

3.4.3 Practice of ROC analysis and AUC calculation.

This subsection is not required reading if the goal is only the practical implementation of the U-test itself. However, it gives an insight into machine learning methods that are not related to the so-called *frequentist statistics* to which the U-test itself belongs, and shows the relationship between these areas of data analysis. In addition, it will provide sufficient knowledge to perform a ROC analysis as such, which may be useful in other situations that an appraiser may encounter in his or her practice.

3.4.3.1 Plotting the ROC curve

ROC analysis and in particular the construction of ROC curves are widely used to find a compromise between the *sensitivity* and *specificity* of a binary classifier. Most of the classifiers used in machine learning produce a result in the form of a quantification that a given object has a "positive" feature value. Some threshold value is needed to convert such a quantitative assessment into a concrete "yes" or "no" prediction. In his case, observations with a score above this threshold will be classified as "positive", below as "negative". Different thresholds provide different levels of sensitivity and specificity.

Listing 3.4: Creating a function to calculate TPR and FPR

```
# create own function for ROC
appraiserRoc <- function(labels, scores){
labels <- labels[order(scores, decreasing=TRUE)]
data.frame(TPR=cumsum(labels)/sum(labels),
FPR=cumsum(!labels)/sum(!labels), labels)
}
```

Setting a relatively high threshold value provides a conservative approach to the issue of classifying a particular case as "positive", which reduces the likelihood of false positives. At the same time, this increases the risk of missing the observed positive values, i. e., it reduces the level of true positive classification results. A relatively low threshold value provides a more liberal approach to classifying observations as "positive". This reduces specificity (increases the number of false negatives) and increases sensitivity (increases the number of true positives). The ROC curve shows the ratio of true positives to false positives, giving an overview of the entire spectrum of such trade-offs. There are many R language libraries that plot ROC curves and calculate metrics for ROC analysis. In this case, to better understand the essence of ROC analysis, some actions will be performed by writing our own functions. The following will show an algorithm for constructing a ROC curve based on a set of real outcomes and their corresponding estimates. The calculation involves two steps:

- sort the observed outcomes in descending order by their predicted scores;
- calculation of total true positive (TPR) and true negative (TNR) scores for ordered observed outcomes.

Let's create an appropriate function (script 3.4). This function has two inputs:

- *labels* — Boolean vector containing actual classification data;
- *scores* — a vector of real numbers containing data about the scores predicted by some classifier.

Since only two classification outcomes are possible, the labels vector can only contain *TRUE* or *FALSE* values (or *1* and *0* depending on the analyst's preference). A sequence of such binary values can be interpreted as a set of instructions for a turtle graphics [49]. There is one important feature: in this case the turtle has a compass and receives instructions for absolute directions of movement: "to the north" or "to the east" instead of relative "to the right" and "to the left". The turtle starts its movement from the starting point with coordinates (0,0) and makes its way on the plane according to the sequence of instructions. When a *TRUE* command is received, it takes one step north, i. e., in the positive direction of the y-axis, and when a *FALSE* command is received, it takes one step east, i. e., in the positive direction of the x-axis. The length of the steps

is chosen in such a way that if all *TRUE (1)* commands are received consecutively, the turtle will be at a point with coordinates (0,1), all *FALSE (0)* commands at a point with coordinates (1,0). Thus, the length of the step "to the north" may be different from the length of the step "to the east". The path in the plane is determined by the order of the *TRUE (1)* and *FALSE (0)* commands and always ends at (1,1).

Advancing the turtle through the bits of the instruction string is an adjustment of the classification threshold to less and less stringent. Once the turtle has passed the bit, it means that it has decided to classify that bit as "positive". If this bit was actually "positive", it is a true positive, if it was actually "negative" it is a false positive. The y-axis shows the TPR, calculated as the ratio of the number of positive results detected to this time to the total number of actual positive results. The x-axis shows the (FPR), calculated as the ratio of the number of currently detected positive results to the total number of actual negative results. The vectorized implementation of this logic uses cumulative sums (the **cumsum** function) instead of going through the values one by one, although that is what the computer does at a lower level.

The ROC curve calculated in this way is actually a step function. With a very large number of positive and negative cases, these steps are very small, and the curve looks smooth. In this case, with a really large number of observations, the construction of each point is difficult. As a consequence, in practice, most ROC curve functions used for practical purposes contain additional steps and often use some form of approximation.

As an example, consider a situation in which an appraiser evaluates parts manufactured by an enterprise. Some of the parts are known to be of good quality and some are defective. The valuation of quality parts is carried out on the basis of cost market approaches in the usual manner. And defective parts are valued at a scrap value. In this case, it is necessary to assign each part to one or another category. There is some feature x, which can be measured by the appraiser. And there is also some feature y, which cannot be measured by the appraiser. The value of the feature y allows you to classify parts as quality or defective. It is also known that there is some finitary relation function between features x and y. Thus, knowing the value of x, we can infer the value of y with some probability. In this case, it is advisable to take a certain sample of parts. Then, together with the specialists of the customer company, measure the values of features *0* and y for each element of this sample.

We will use simulated data to consider the example. There is some input feature x that is linearly related to the implicit result y. This relationship implies the presence of some randomness. The y-value shows whether the part exceeds the tolerance requirements. If so, it should be classified as defective. The algorithm used in this paper involves the following steps:

- create the **sim_parts_data** function that generates data according to certain rules and sets the "$y > 100$" threshold value to classify parts as defective.
- create the dataframe **parts_data** with this function;
- create the **test_set_idx** rule, whereby 80 % of the data is randomly assigned to the training sample, and 20 % to the test sample;

- applying the rule **test_set_idx** to data **parts_data**;
- create training (**«training_set»**) and testing (**«test_set»**) sub-samples;
- plot the diagram showing the distribution of observations from the training sample.

To implement the above algorithm, the code from script 3.5 was used.

The result was diagram 3.7. As you can see, if the value of the parameter x is less than 15, all dots are red, which means that the parts are defective. Above 96 are green, which means that the parts are of good quality. If the value is higher than 96, they are green, which means that the parts are of good quality. Between these values is an area of uncertainty, the right side of which is dominated by green dots, and the left side by red dots.

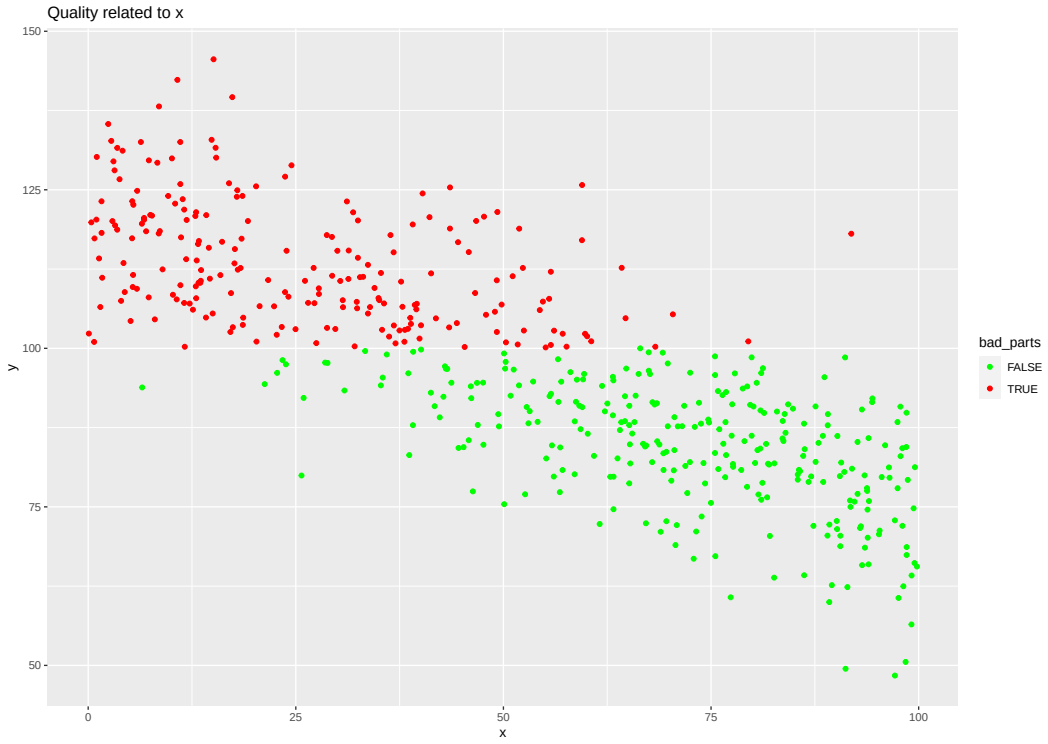

Figure 3.7: Diagram of the distribution of parts with respect to the parameter x.

The training sub-sample will be used to create a logistic regression model based on the values of the attribute x, which allows you to assign a particular part to quality or defective. This model will be used to assign scores to the observations in the training sample. In the future, these scores will be used to construct the ROC curve together with the true labels. Recall that the ROC curve is plotted for observations with known values of parameters x and y. This ROC curve is then applied to the entire set of objects for which x values are known but y values are unknown. The scores themselves as well

Listing 3.5: Creation and primary visualization of data on quality and defective parts

```
# Sample of ROC-analysis

# enable libraries
library(ggplot2)
library(dplyr)
library(pROC)

#set seed
set.seed(19190709)

# create own function for ROC
appraiserRoc <- function(labels, scores){
labels <- labels[order(scores, decreasing=TRUE)]
data.frame(TPR=cumsum(labels)/sum(labels),
FPR=cumsum(!labels)/sum(!labels), labels)
}

# create function
sim_parts_data <- function(N, noise=100){
x <- runif(N, min=0, max=100)
y <- 122 - x/2 + rnorm(N, sd=noise)
bad_parts <- factor(y > 100)
data.frame(x, y, bad_parts)
}

# create dataset
parts_data <- sim_parts_data(2000, 10)

# create rule for test subset
test_set_idx <- sample(1:nrow(parts_data),
size=floor(nrow(parts_data)/4))

# create training and test subsets
test_set <- parts_data[test_set_idx,]
training_set <- parts_data[-test_set_idx,]

# plot graph
test_set %>%
ggplot(aes(x=x, y=y, col=bad_parts)) +
scale_color_manual(values=c("green", "red")) +
geom_point() +
ggtitle("Bad parts related to x")
```

Listing 3.6: Comparing ”link” and ”response” predictions

```
fit_glm <- glm(bad_parts ~ x, training_set,
family=binomial(link="logit"))

glm_link_scores <- predict(fit_glm, test_set, type="link")

glm_response_scores <- predict(fit_glm, test_set, type="response")

score_data <- data.frame(link=glm_link_scores,
response=glm_response_scores,
bad_parts=test_set$bad_parts,
stringsAsFactors=FALSE)

score_data %>%
ggplot(aes(x=link, y=response, col=bad_parts)) +
scale_color_manual(values=c("green", "red")) +
geom_point() +
geom_rug() +
ggtitle("Both link and response scores put cases in the same order")
```

as the x and y values are not displayed on the graph and are only used for sorting labels. Two different classifiers sorting labels in the same order will give identical ROC curves regardless of the absolute values of the scores. This can be seen by constructing an ROC curve based on ”response” or ”link” predictions from a logistic regression model. The ”response” scores were mapped to a (0, 1) scale using a Sigmoid function[75], the ”link” scores were left untransformed. In this case, the points showing specific observations are ordered in the same way. To test this hypothesis, we use the code 3.6. As you can see in Figure 3.8, the order of the dots is the same for ”link” and ”response”.

Let’s go directly to the construction of the ROC curve. We use both the ready function from the ”pROC” package and the previously created **”appraiserRoc”** function (see script 3.7). The result of the first is represented as an orange curve, the second as circles of red for defective parts and black for quality parts (see Diagram 3.9). It is not difficult to guess that the red dot corresponded to the ”North” command and the black dot to the ”East” command. Since the library function and the own function perform the same actions, the two curves are identical.

Note that the ”Specificity” scale is plotted on the abscissa axis, not the FPR, so the values on the axis are inverted. Since, according to Table 3.4, ”$Specificity = 1 - FPR$” we can talk about the mutual unambiguity of these indicators. Consequently, when plotting the ROC curve any of them can be used. This version of the scale display was self-selected by the **roc** function from the ”pRoc” library. If the user does not set his settings, the function chooses to display the scale so that the AUC value is always greater

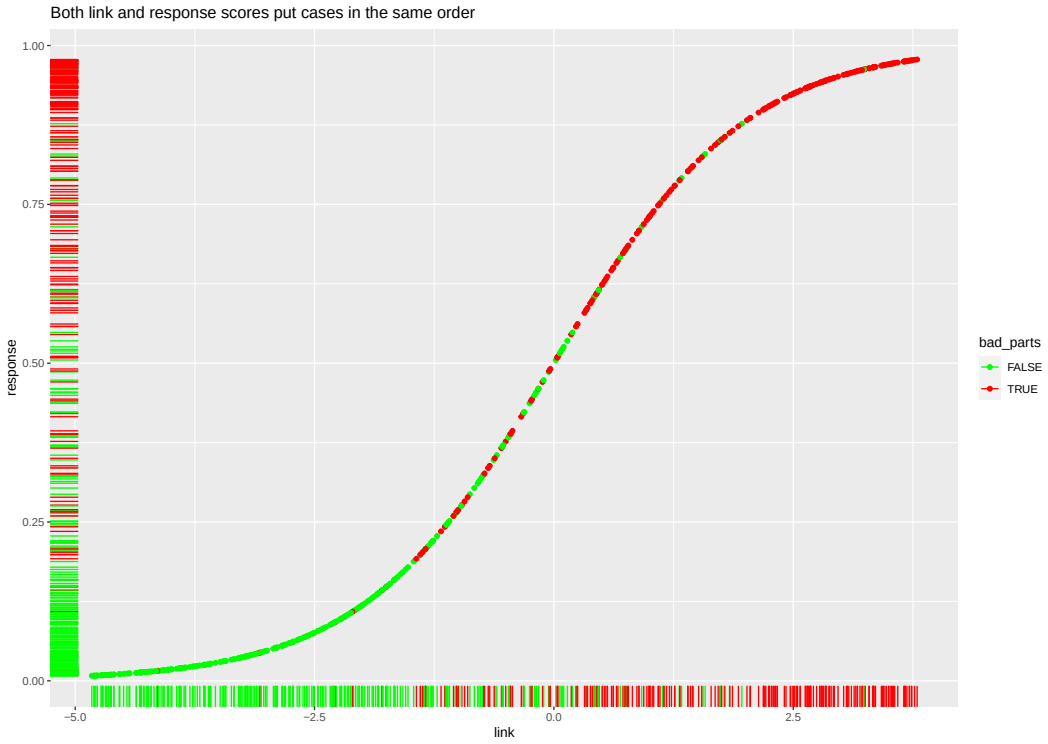

Figure 3.8: Comparison of the order of points for ”link” and ”response”.

Listing 3.7: Plotting the ROC curve using library and own functions

```
# plot ROC
plot(roc(test_set$bad_parts, glm_response_scores, direction="<"),
col="orange", lwd=3, main="The turtle finds its way", xlim = c(1, 0))
glm_simple_roc <- appraiser_roc(test_set$bad_parts=="TRUE",
glm_link_scores)
with(glm_simple_roc, points(1 - FPR, TPR, col=1 + labels))
```

than 0.5. This calculation is based on which group (quality parts, defective parts) has a higher median score. Since the **appraiserRoc** function is of course not that smart, a simple subtraction was performed during its use, making it possible to build a joint diagram.

This approach has one limitation: based on the prognostic nature of the ordering of outcomes, it does not allow correct processing of information if the sequence consists of identical estimates. "Turtle" assumes that the order of the labels matters, but there is no meaningful order in the situation of the same scores. These areas should be displayed with a diagonal line, but not the traditional steps.

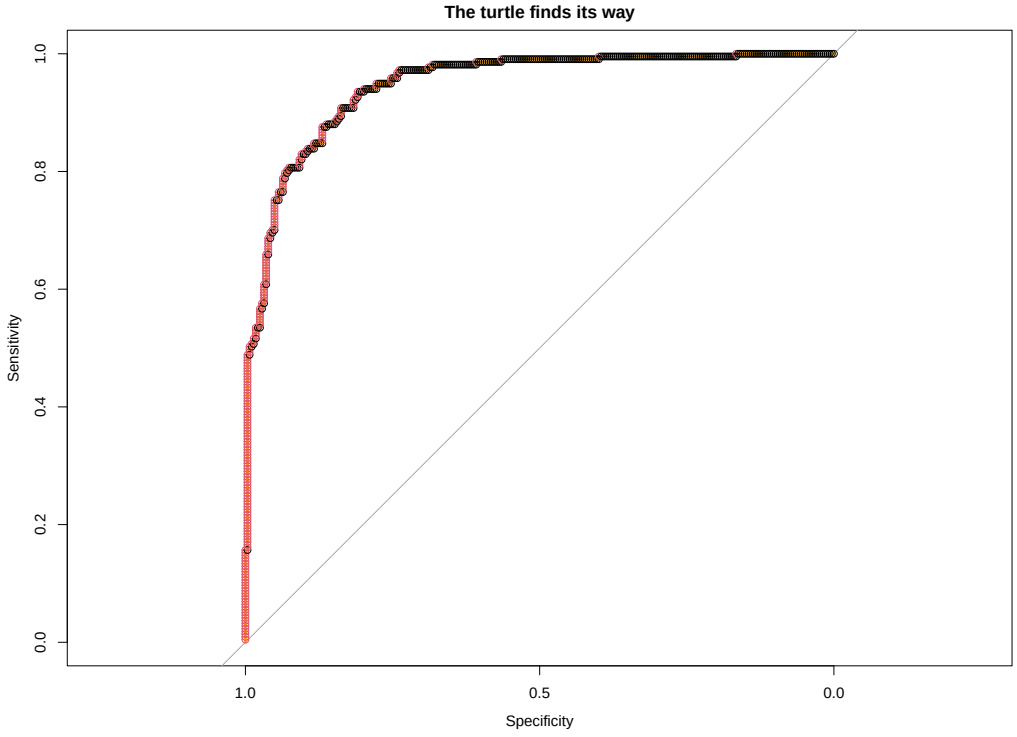

Figure 3.9: Identical ROC curves plotted with library and own functions.

Listing 3.8: Plotting the ROC curve in case of absence of order of scores

```
# plot ROC for 99% negative cases
N <- 2000
P <- 0.01
rare_success <- sample(c(TRUE, FALSE), N, replace=TRUE, prob=c(P, 1-P))
guess_not <- rep(0, N)
plot(roc(rare_success, guess_not), print.auc=TRUE)
appr_roc <- appraiserRoc(rare_success, guess_not)
with(appr_roc, lines(1 - FPR, TPR, col="blue", lty=2))
```

Consider an example where a diagonal is the only adequate way to plot a ROC curve. To do this, create an extremely unbalanced data set in which only 1 % of the observations are "positive". In this case, the result of the prediction will always be negative. Since all scores will be the same, there is no need for any ordering. The **roc** function from the "pRoc" package correctly recognizes such situations and draws a diagonal line (1,0; 0,1). In doing so, the turtle assumes that the order of scores has some significance, and moves between these points along a random trajectory, alternating between "north" and "east" directions. The code calling the construction of such a ROC curve is given in script 3.8. In Diagram 3.10, the black diagonal line was plotted by the library function, while the blue dashed line was plotted by our own previously written **appraiserRoc** function. As you can see, the library function correctly determined the case of identical estimates, while applying our own function resulted in random turtle wanderings.

The greater the value of N, the closer to the diagonal the turtle will wander. Greater unbalance requires more points in order for the path to run roughly close to the diagonal. In less extreme cases, the emergence of diagonal sections is possible, in particular, in the case of rounding of estimates, leading to equality of some of them.

To further familiarize yourself with the topic of constructing ROC curves, we can recommend studying this theoretical material [12], as well as practice on the online simulator [42].

3.4.3.2 The concept of AUC and its calculation

The ROC curve is a popular means of visualizing the trade-off between sensitivity and specificity of a binary classifier. Earlier in 3.4.3.1, the issue of constructing the ROC curve was considered in terms of the turtle steps, which takes a vector with instructions as steps "north", i. e., in the positive direction on the y-axis, and "east", i. e., in the positive direction on the x-axis. In this case, the sequence of scores, on the basis of which the vector of bitwise instructions is formed, is ordered in such a way that the cases that are most likely to be positive come first. In doing so, the turtle assumes that all cases are positive. Next, for the training sample (in which true values are known for certain), we determine whether the case was true- (TP) or false-positive (FP). The step size on the y-axis is inversely proportional to the number of positive observations, on the x-

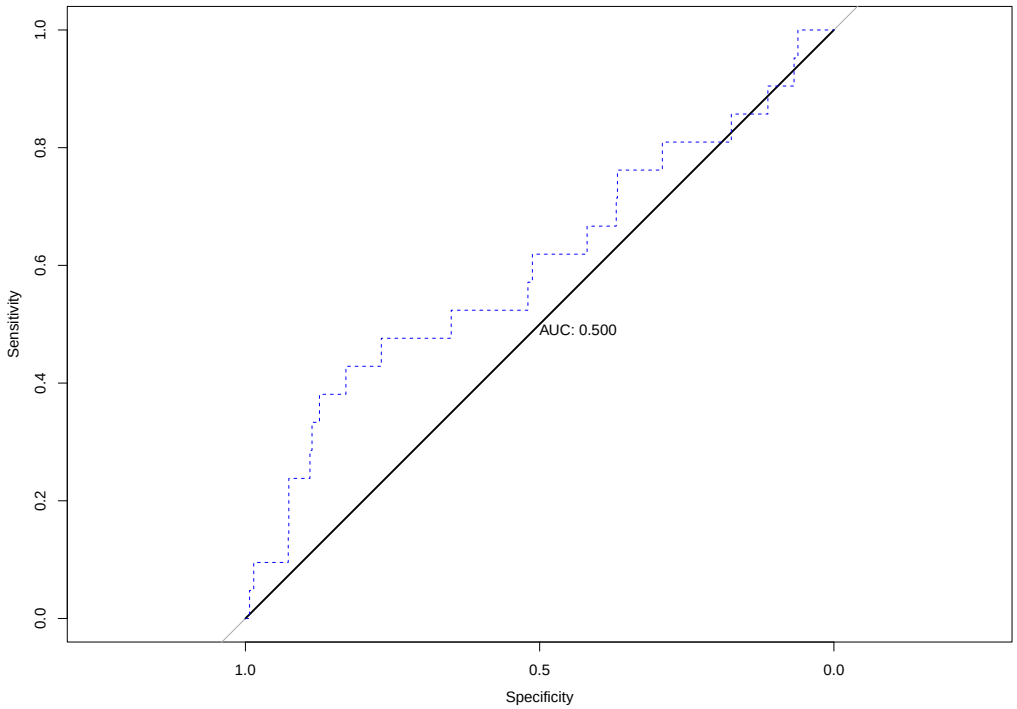

Figure 3.10: ROC curve arising when there is no value of the order of scores.

Listing 3.9: Create a test data set

```
# Geometric interpretation of AUC

# activate libraries
library(pROC)

# create dataset
category <- c(1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0,
0)
prediction <- rev(seq_along(category))
prediction[9:10] <- mean(prediction[9:10])
```

axis to the number of negative observations. Thus, the curve path always ends at (1,1). The result is a plot of the frequency of true positives (TPR or $sensitivity$) against the frequency of false positives (FPR or $1 - specificity$), which is actually the ROC curve. At the same time, the graph itself does not provide any quantitative estimates of the quality of the binary classifier.

Calculating the area under the ROC curve is one way to summarize the quantitative assessment of classifier quality. This metric is so common that in the context of data analysis, the terms *Area under the curve* or AUC refer specifically to the area under the ROC curve, unless explicitly stated otherwise.

At first glance, the simplest and most intuitive metric for classifier performance is its accuracy. Unfortunately, in some cases, such a metric simply will not work. For example, in the case of a disease that occurs in one person per million, a completely useless test that always shows a negative result would be 99.9999% accurate. In contrast to the accuracy index, ROC curves are insensitive to unbalanced classes. The aforementioned useless test will have an $AUC = 0.5$, which is equivalent to no test at all (see diagram 3.10).

In this subsubsection, we will first consider the geometric approach to the concept of AUC and develop a function that calculates its value. Next, we turn to another — probabilistic — interpretation of the concept of AUC.

3.4.3.2.1 Geometric approach to the concept of AUC First, let's create a test dataset using the code shown in 3.9. The **prediction** vector contains pseudo-estimates, which in practice are assigned by the classifier. In this study problem, they are a decreasing sequence, which generally corresponds to the "category" labels. The scores for observations with ordinal numbers 9 and 10, one representing the positive case and the other the negative case, are replaced by their mean values to create a ties effect.

To construct the ROC curve, TPR and FPR must be calculated. It was shown in 3.4.3.1 how this can be done semi-automatically by calculating cumulative sums for positive and negative labels. This section will use the library "pRoc", which performs all calculations automatically at a low level. Let's calculate the TPR, FPR, and AUC

Listing 3.10: Calculation of the AUC using the pRoc library

```
# create ROC object&dataframe and calculate AUC
roc_obj <- roc(category, prediction)
auc(roc_obj)
roc_df <- data.frame(
TPR=rev(roc_obj$sensitivities),
FPR=rev(1 - roc_obj$specificities),
labels=roc_obj$response,
scores=roc_obj$predictor)
```

values with the code 3.10. A dataframe will also be created, containing the data for each observation, and shown in Table 3.7. The **roc** function can return the values of many indicators, but at this point we only need TPR and FPR. Recall that TPR means sensitivity and FPR is equivalent to the expression $1 - specificity$. By default, the **roc** function returns values in ascending order. As a consequence, they have been inverted so that the starting point has coordinates (0,0). The AUC value returned by the function was 0.825. In the following, we will compare it with the one that will be obtained in the course of its independent semi-automatic calculation.

3.4.3.2.1.1 Plotting the graph If the ROC curve were a perfect step function, the area under it could be found by adding a set of vertical bars equal to the spaces between the points on the abscissa axis (FPR) and to the height of the step on the ordinate axis (TPR). Real ROC curves may include sections corresponding to repeated values. In this case, there are segments other than steps. Such repeats require proper consideration. In Figure 3.11, the area of ordinary steps is shown in green, the cases of repeats (ties) are indicated by blue rectangles divided in half by sloping ROC curve segments. Thus, half of the area of these steps is included in the total area under the curve.

The following steps were taken to plot diagram 3.11.

1. Define a **rectangle** function that takes as arguments:
   - the initial x and y coordinates;
   - width and height of step;
   - angle of rotation after each step;
   - hatching density.

   The algorithm in script 3.11 builds the rectangle as follows:
   - the function accepts the initial x and y coordinates from the appraiser.
   - then it gets the value of the step length "to the east" (to the right on the x-axis), calculates the new value of the x-coordinate, keeping the value of the y-coordinate.

Table 3.7: TPR, FPR, labels, scores for the training dataset.

	TPR	FPR	labels	scores
1	0.00	0.00	1.00	20.00
2	0.10	0.00	1.00	19.00
3	0.20	0.00	1.00	18.00
4	0.30	0.00	1.00	17.00
5	0.40	0.00	0.00	16.00
6	0.40	0.10	1.00	15.00
7	0.50	0.10	1.00	14.00
8	0.60	0.10	0.00	13.00
9	0.60	0.20	1.00	11.50
10	0.70	0.30	0.00	11.50
11	0.80	0.30	1.00	10.00
12	0.80	0.40	0.00	9.00
13	0.90	0.40	1.00	8.00
14	0.90	0.50	0.00	7.00
15	0.90	0.60	0.00	6.00
16	1.00	0.60	1.00	5.00
17	1.00	0.70	0.00	4.00
18	1.00	0.80	0.00	3.00
19	1.00	0.90	0.00	2.00
20	1.00	1.00	0.00	1.00

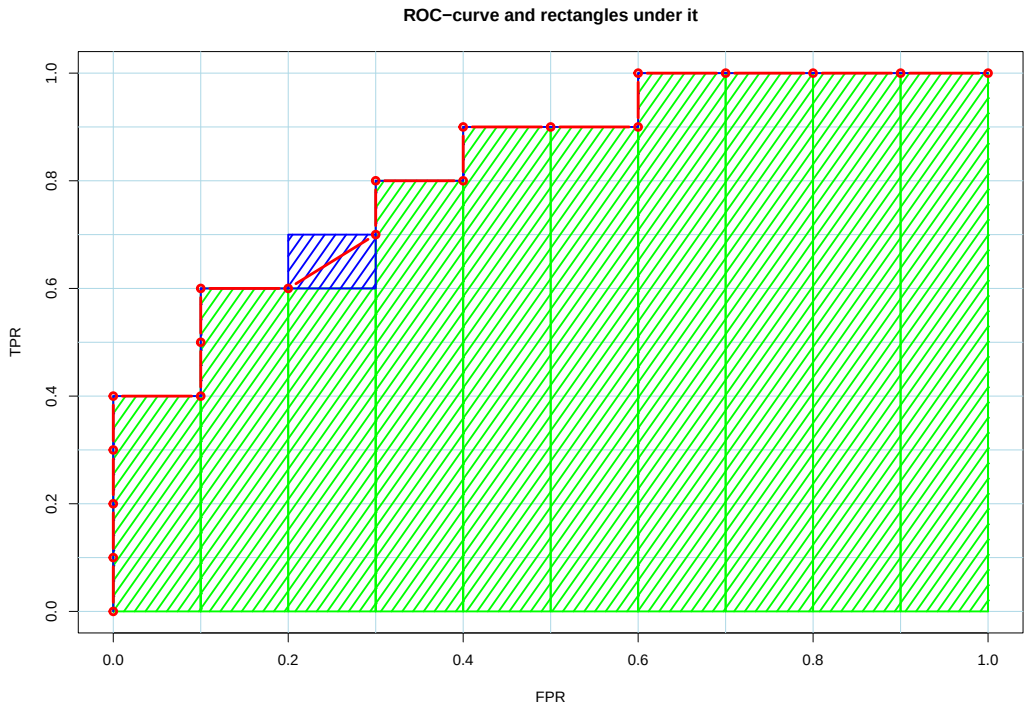

Figure 3.11: ROC curve and the area under it, taking into account the presence of ties.

Listing 3.11: Create the **rectangle** function

```
# create function for plotting rectangles
rectangle <- function(x, y, width, height, density=12, angle=45, ...)
polygon(c(x,x+width,x+width,x), c(y,y,y+height,y+height),
density=density, angle=angle, ...)
```

Listing 3.12: Adding *dFPR* and *dTPR* columns

```
# add dFPR and dTPR columns
roc_df <- transform(roc_df,
dFPR = c(diff(FPR), 0),
dTPR = c(diff(TPR), 0))
```

- after performing the step is a turn of 45°counterclockwise.
- then it gets the value of the step length "to the North" (up on the y-axis), calculates a new value of the y-coordinate, keeping the value of the x-coordinate.
- this is followed by a new turn to the left by 45°and new steps in opposite directions.
- thus, performing one step each to the "east", "north", "west" and "south", as well as three turns of 45°counterclockwise function returns to the starting point, completing the construction of the rectangle.

2. To calculate the step length "to the East" and "to the West" it is necessary to calculate the difference between neighboring values of FPR, "to the North" and "to the South" between neighboring values of TPR. To do this, we add the two columns *dFPR* and *dTPR*, respectively, using the code shown in 3.12. Since the number of pairs for which the difference is calculated is less than the number of observations by one, zero should be added at the end (since the dataframe data are sorted in descending order).

3. Next, the coordinate grid is laid out from zero to one on each axis using the code 3.13.

4. For the case of repeating values (ties), there is a special kind of step "to the North-East" in the form of a diagonal line.

5. The final step is to build the ROC curve itself and the rectangles that form the area under it, according to script 3.14. The **mapply** function allows you to apply the **rectangle** function sequentially to each row of the data frame.

Listing 3.13: Drawing an empty graph and marking axes from 0 to 1

```
# plot empty graph from 0 to 1 for each axis
plot(0:10/10, 0:10/10, type='n', xlab="FPR", ylab="TPR",
main = 'ROC-curve and rectangles under it')
abline(h=0:10/10, col="lightblue")
abline(v=0:10/10, col="lightblue")
```

Listing 3.14: Construction of ROC curve and rectangles under it

```
# plot ROC-curve and rectangles under it
with(roc_df, {
mapply(rectangle, x=FPR, y=0,
width=dFPR, height=TPR, col="green", lwd=2)
mapply(rectangle, x=FPR, y=TPR,
width=dFPR, height=dTPR, col="blue", lwd=2)
lines(FPR, TPR, type='b', lwd=3, col="red")
})
```

3.4.3.2.1.2 The summation of areas by means of an own function The area under the curve (AUC) (highlighted in red) is the sum of the areas of all the green rectangles and half the area of the blue one. To calculate the area of each rectangle it is not necessary to know the absolute coordinates of its vertices, its width and height are enough. Since one of the sides of each rectangle lies on the x-axis, the height of any of them is determined by the value of TPR, the width by dFPR. Then the total area of all green rectangles is equal to the dot (scalar) product [55] of TPR and dFPR. This vector approach calculates the area for each data point, even if its width or height is zero. However, in this case their further inclusion in the calculation is of no importance. The area of the blue rectangles (if any) is determined by the values of dFPR and dTPR and also represents their scalar product as vectors. For areas of the graph containing northward or eastward steps, one of these values (dTPR, dFPR) will be zero. As a consequence, blue rectangles are possible only if TPR and FPR are changed simultaneously. In this case, only half of such rectangle is under the curve.

Recall that the previously calculated AUC was 0.825. Now let's calculate it in semi-automatic mode according to the algorithm described in the previous paragraph. To do this, first create the function **appraiser_auc**, and then apply it to the test data (see script 3.15). The returned value will be 0.825, which indicates that the logic and algorithm are correct.

3.4.3.2.2 The rank comparison approach It is also possible to use a fundamentally different approach to calculate the AUC. To implement it, it is necessary to create a matrix containing all possible combinations of positive and negative cases. Each row

Listing 3.15: Creating a function to calculate AUC in semi-automatic mode and applying it to test data

```
# create function for AUC calculation
appraiser_auc <- function(TPR, FPR){
# inputs already sorted, best scores first
dFPR <- c(diff(FPR), 0)
dTPR <- c(diff(TPR), 0)
sum(TPR * dFPR) + sum(dTPR * dFPR)/2
}

# apply function to data
with(roc_df, appraiser_auc(TPR, FPR))
```

represents a positive case. They are ordered so that the top row contains the case with the lowest score, and the bottom row contains the case with the highest score. Similarly, the columns contain the negative cases, sorted so that the left column contains the highest scores. Then each cell represents a comparison of a particular positive case with a particular negative case. If the score or rank of a positive case is higher than that of a negative case, that cell takes TRUE. In the case of a good enough classifier, most positive cases will have higher scores (ranks) than negative cases. All exceptions will be concentrated in the upper left corner, where positive cases with low scores and negative cases with high scores are located. The 3.16 script contains code to implement this algorithm and its visualization, shown in Figure 3.12.

The plotting of the ROC curve in this case is done almost in the usual way. The only difference is that it is slightly shifted and stretched to make the coordinates coincide with the corners of the matrix cells. This way of constructing the ROC curve makes the following fact obvious: the ROC curve represents the boundary of the area where positive cases have higher scores (ranks) than negative cases. Thus, the AUC can be calculated by replacing the values in the matrix so that

- cells in which the scores (ranks) of positive cases exceed the scores (ranks) of negative cases take the value 1;
- cells in which the scores (ranks) are equal take the value 0.5;
- cells in which the scores (ranks) of negative cases exceed the scores (ranks) of positive cases take the value 0.

Since applying the **sign** function results in one of three possible values: -1, 0, 1, we use the following trick to place the values in the desired range: we add one to them and divide by two. The final calculation of the AUC is done by calculating the mean value.

3.4.3.2.3 Probabilistic approach to calculating AUC The probabilistic interpretation is that if you randomly choose a positive case and a negative case, the probability that

Listing 3.16: Create the function to build a comparison matrix and apply it to the test data

```
# create function for rank comparison
rank_comparison_auc <- function(labels, scores, plot_image=TRUE, ...){
score_order <- order(scores, decreasing=TRUE)
labels <- as.logical(labels[score_order])
scores <- scores[score_order]
pos_scores <- scores[labels]
neg_scores <- scores[!labels]
n_pos <- sum(labels)
n_neg <- sum(!labels)
M <- outer(sum(labels):1, 1:sum(!labels),
function(i, j) (1 + sign(pos_scores[i] - neg_scores[j]))/2)

AUC <- mean (M)
if (plot_image){
image(t(M[nrow(M):1,]), ...)
library(pROC)
with( roc(labels, scores),
lines((1 + 1/n_neg)*((1 - specificities) - 0.5/n_neg),
(1 + 1/n_pos)*sensitivities - 0.5/n_pos,
col="blue", lwd=2, type='b'))
text(0.5, 0.5, sprintf("AUC = %0.4f", AUC))
}

return(AUC)
}

# apply function to data
rank_comparison_auc(labels=as.logical(category), scores=prediction)
```

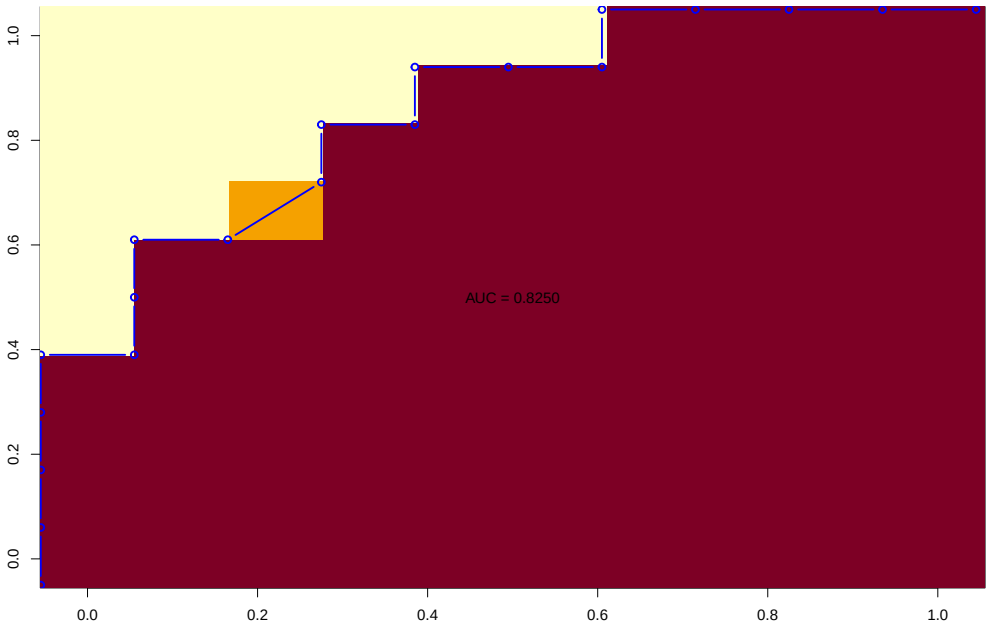

Figure 3.12: Visualization of the matrix of comparisons of scores (ranks).

Listing 3.17: Creating and applying a function to calculate AUC as a probability

```
# create function for calculation AUC as probability
auc_probability <- function(labels, scores, N=1e7){
pos <- sample(scores[labels], N, replace=TRUE)
neg <- sample(scores[!labels], N, replace=TRUE)
# sum( (1 + sign(pos - neg))/2)/N # does the same thing
(sum(pos > neg) + sum(pos == neg)/2) / N # give partial credit for ties
}

# apply function to data
auc_probability(as.logical(category), prediction)
```

the score (rank) value of the positive case will be greater than that of the negative case is determined by AUC and equal to it. This follows, in particular, from Diagram 3.12, in which the total area of the graph is normalized to one. The total area of the graph is normalized to one, the cells of the matrix contain information about all possible combinations of positive and negative cases. The area under the curve consists of cells in which the scores (ranks) of positive cases exceed those of negative cases. To approximate the AUC in the probabilistic approach, let's create and apply a function according to the script 3.17. The returned AUC value was 0.8248116, which approximately corresponds to its exact value calculated earlier.

3.4.3.3 Calculation of AUC for large data sets

In the previous sections, we worked with a very small dataset that allowed us to almost manually go through the data. In this subsubsection, we will work with a previously created dataset containing information about quality and defective parts that require different valuation methods. Recall that in this dataset, the score vector is contained in the *glm_responce_scores* variable, and the label vector is contained in the *bad_parts* variable.

The previously created dataset does not contain repeating values. Therefore, the tie effect will be added artificially to explore the issue more deeply. For this purpose, rounding of scores was performed. Then we plot the ROC curve for the original data and then for the rounded data. The code to perform the above actions is given in script 3.18. To create a tie effect, a new vector of scores was created, containing their values rounded to one decimal place. The result of plotting the two ROC curves is shown in Diagram 3.13. The black line corresponds to the ROC curve plotted from the original data. These data contain unique scores for each observation. The red line corresponds to the ROC curve plotted with rounded scores. As an input vector we used the data of variable 5, taking a value from 0 to 1. Consequently, scores rounded to one decimal place have only eleven possible values from 0.0 to 1.0. The AUC value for both cases is plotted directly on the chart. As can be seen, the values are approximately equal due

Listing 3.18: Plotting the ROC curve on the original (black line) and rounded (red line) defective parts data

```
# create function for calculation AUC as probability
auc_probability <- function(labels, scores, N=1e7){
pos <- sample(scores[labels], N, replace=TRUE)
neg <- sample(scores[!labels], N, replace=TRUE)

# sum( (1 + sign(pos - neg))/2)/N # does the same thing
(sum(pos > neg) + sum(pos == neg)/2) / N # give partial credit for ties
}

# apply function to data
auc_probability(as.logical(category), prediction)
```

to a sufficiently large number of observations.

Next, let's try to apply the standard, as well as three of our own functions to calculate the AUC. Next, we construct two ROC curves based on the rank comparison matrices for the original and rounded scores. The code for applying the four functions to calculate AUC, as well as constructing ROC curves based on comparison matrices, is given in script 3.19.

Diagram 3.14 contains the ROC-curve based on the original data. Diagram 3.15 contains the ROC curve based on rounded scores. As you can see, each section is a diagonal segment running to the "North-East". It was shown earlier that this direction corresponds to repeated values.

The final results of the AUC calculations by the four methods are shown in Table 3.8. As can be seen, all three of our own functions calculate the AUC quite well. They give close results with respect to the result of the function from the package **pROC**. Naturally, these self-made functions serve more for training purposes. In practice, you should use the standard tools, such as **pROC** or **ROCR**.

Table 3.8: Results of AUC calculation by four methods.

	Full.Resolution	Rounded.Scores
auc	0.9302279874213836841079	0.9301379061844864404307
appraiser_auc	0.9302279874213836841079	0.9301379061844863294084
rank_comparison_auc	0.9302279874213835730856	0.9301379061844864404307
auc_probability	0.930719999999999917577	0.929174999999999731770

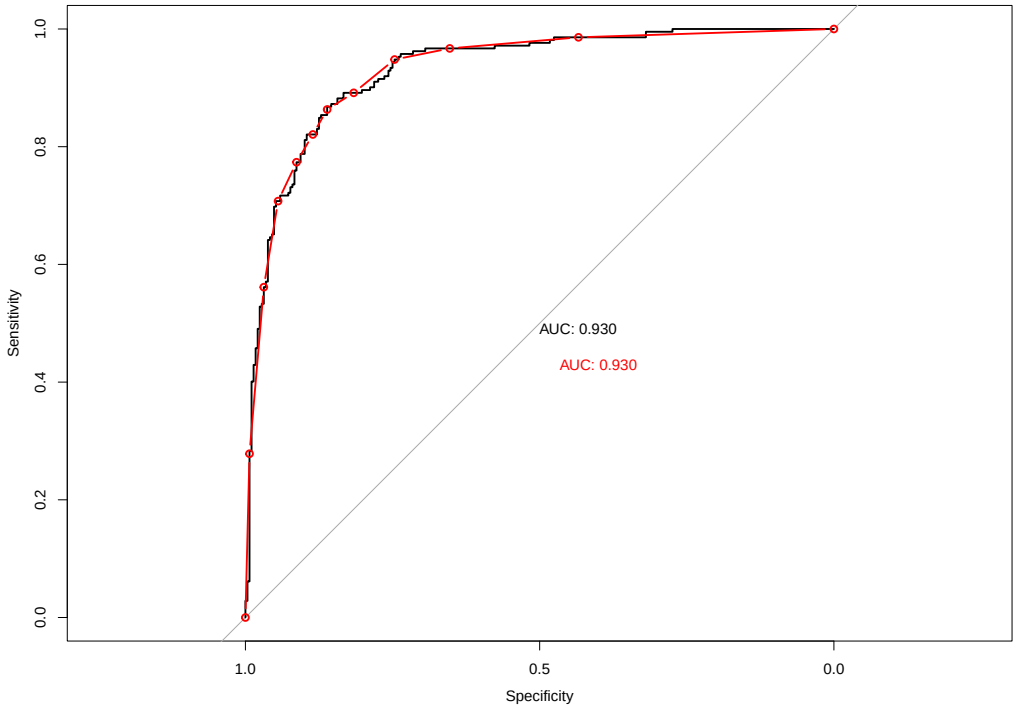

Figure 3.13: ROC curve for original (black line) and rounded (red line) data about defective parts.

Listing 3.19: Using four functions to calculate AUC and plot ROC curves based on comparison matrices for original and rounded scores

```
# apply all functions to data
results <- data.frame(
`Full Resolution` = c(
auc = as.numeric(auc(roc_full_resolution)),
appraiser_auc = appraiser_auc(rev(roc_full_resolution$sensitivities),
rev(1 - roc_full_resolution$specificities)),
rank_comparison_auc = rank_comparison_auc(test_set$bad_parts,
glm_response_scores,
main="Full-resolution scores (no ties)"),
auc_probability = auc_probability(as.logical(test_set$bad_parts),
glm_response_scores)
),
`Rounded Scores` = c(
auc = as.numeric(auc(roc_rounded)),
appraiser_auc = appraiser_auc(rev(roc_rounded$sensitivities),
rev(1 - roc_rounded$specificities)),
rank_comparison_auc = rank_comparison_auc(test_set$bad_parts,
rounded_scores,
main="Rounded scores (ties in all segments)"),
auc_probability = auc_probability(as.logical(test_set$bad_parts),
rounded_scores)
)
)
```

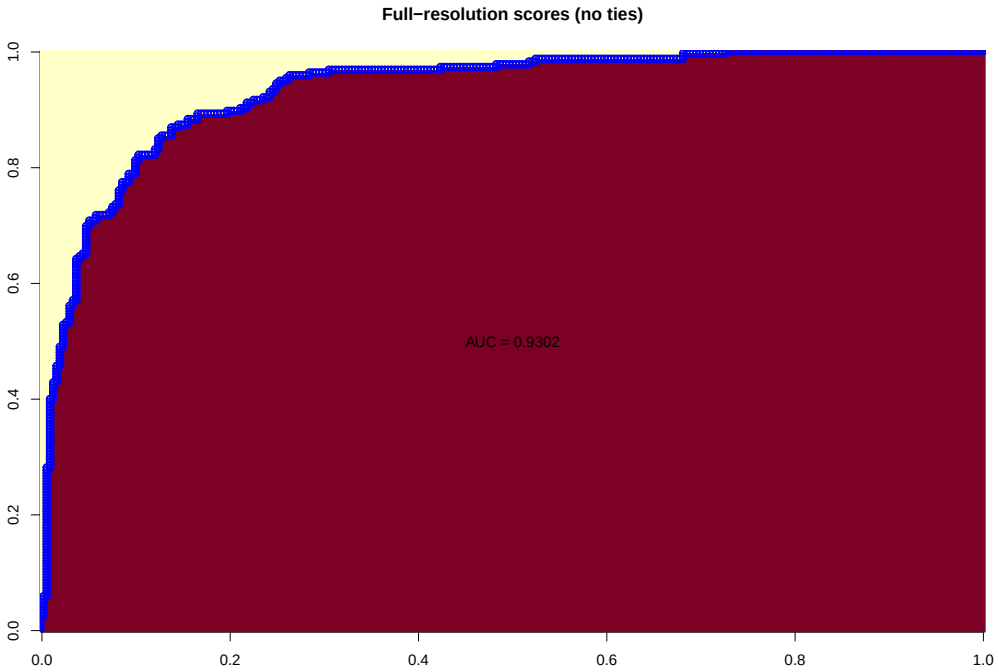

Figure 3.14: ROC curve for the original defective parts data based on the comparison matrix.

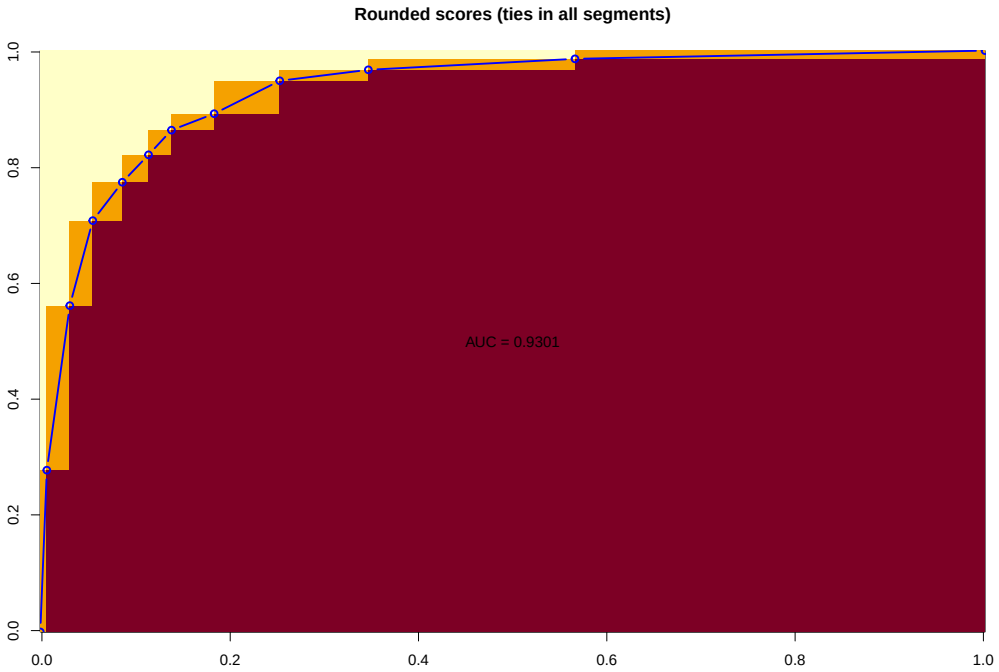

Figure 3.15: ROC curve for the rounded defective parts data based on the comparison matrix.

3.4.3.4 Summary of AUC findings

The analysis of the concept of AUC was carried out in the previous sections. The algorithms for calculating its value, including those based on the probabilistic approach, were also presented. In general, it can be said that an in-depth study of issues related to ROC and AUC is not strictly necessary for the successful application of the U-test in the practical work of the appraiser. While the geometric approach to calculating the AUC is still quite intuitive and simple, interpreting the AUC as a probability seems somewhat abstruse and requires a certain level of general mathematical background.

However, in the context of valuation activities, the concept of AUC as a probability is indeed quite useful. Appraisers often deal with relatively small samples. In this case, they need to understand the extent to which they can be confident that the value of the object that has some of the features exceeds the value of the object that does not have it (or vice versa). Thus, the use of the AUC refers us to the Bayesian approach to probability, which includes the concept of *level of certainty*. This draws a bridge between traditional frequentist statistics, within the framework of which the U-test was once developed, and modern machine learning, based for the most part on a Bayesian approach to probability. In the author's opinion, this circumstance alone is sufficient reason to delve into the topics of ROC and AUC when examining the practical issues of using the U-test by appraisers.

In addition, the objectives of appraisers are not limited only to determining the value itself. The use of classifiers can often have independent utility at an earlier stage of the valuation process, for example, at the stage of determining the properties of the objects being valued. The general behavior of the classifier over the entire range of possible threshold values is quite often of less interest than its behavior in a certain relatively narrow range. For example, when solving the above discussed problem of separating defective parts from quality parts, the performance of the classifier in the boundary zone of threshold values is crucial. In this case, the specific setting of such a threshold value may depend on the purpose of the valuation. For example, from the point of view of the lending bank or the Central Bank, which monitors the adequacy of collateral, the sensitivity of the classifier is in a sense more important than its specificity, because the assignment of defective parts to quality parts carries more risk than the reverse situation. At the same time, from the point of view of, for example, the tax inspection, classifying serviceable assets as faulty with their subsequent write-off from the balance sheet of the enterprise, leading to a decrease in the taxable base, is a more serious problem than the erroneous attribution of faulty to serviceable, i. e. a situation arises where specificity is more important than sensitivity. Thus, there may be situations where the cost of Type I and Type II errors is not the same. Such situations present a definite problem because the AUC concept assumes the same cost of these errors. However, this topic is quite specific and goes beyond the scope of this material, the topic of which is the U-test. Briefly, we can say that there are metrics other than AUC to account for the different cost of Type I and Type II errors. For further study of the topic of AUC on your own, we can recommend this material [37].

4 Practical implementation

4.1 Implementation in the LibreOffice Calc spreadsheet

At this point, it is safe to say that spreadsheets are the standard tool for appraisers' calculations. Introducing development tools such as Python and R programming languages into professional activities is quite slow. In addition, an independent step-by-step calculation allows for a better understanding of the U-test methodology. Therefore, it was decided to create a step-by-step instruction for the U-test in a spreadsheet. The software product LibreOffice Calc (Version: 7.3.5.2 / LibreOffice Community Build ID: 30(Build:2) CPU threads: 4; OS: Linux 5.11; UI render: default; VCL: kf5 (cairo+xcb) Locale: en-US (en_US.UTF-8); UI: en-US Ubuntu package version: 1:7.3.5 rc2-0ubuntu0.20.04.1 lo1 Calc: threaded) was used for this purpose. A significant part of its functionality is also available in the most common application of this kind, Microsoft Excel. There is no reason to believe that the calculations made will not work correctly in applications other than LibreOffice Calc. However, it is also impossible to guarantee this. For an unambiguously correct test it is recommended to use this application, which has versions for all major operating systems. In addition, the use of free software is generally good practice. The current version of the spreadsheet U-test.ods is in the repository along with the rest of this work.

The data considered in this section are fictitious and were generated by the LibreOffice Calc pseudorandom number generation algorithm. To re-generate, use the key combination *ctrl+shift+F9*.

Consider the training task. Cells I3:J30 contain data of values of some quantitative feature for observations of two samples taken from sets I and J, respectively. The difference between the elements of these sets is the presence of some attribute in the set I and its absence in the set J. The task is to test the hypothesis that the difference in a given attribute should be recognized as significant, and the attribute itself is a pricing factor. Let us create the null-hypothesis by formulating it in three variants corresponding to the three levels of rigor described earlier in Table 3.1. It should be noted that the U-test refers to statistics based on the so-called frequentist approach to probability. For a discussion of the differences between the *frequentist* and *Bayesian* approaches to probability as applied to value estimation, see, in particular [18]. As you know, the *frequentist* approach is based on the premise that randomness is a consequence of objective uncertainty, which can only be reduced by a series of experiments. In the frequentist approach there is a clear division into random and non-random parameters. A typical task is to estimate certain parameters of the general population, which is a set of random variables. For this purpose, deterministic sampling parameters such as mean, mode, variance, etc. are used. The latter represent specific values in which there

is no longer any randomness. Thus, we initially accept the fundamental assumption of the random nature of the variables under study. Then we apply some methods of mathematical statistics, which allow us to obtain specific values of parameter estimates. This allows us to reduce the level of uncertainty, which nevertheless always remains. It follows that the null hypothesis is most often pessimistic. It carries the assertion that the phenomenon or process under study is based on chance, so that we are unable to draw reliable conclusions. In contrast, the alternative hypothesis says that the level of uncertainty can be reduced due to the success of the experiment. Considering all of the above, let us formulate the null and alternative hypotheses in three variants according to different levels of rigor and record them in Table 4.1. Cells C2:C19 contain some

Table 4.1: The null hypothesis and the alternative hypothesis in the analysis of training data.

Type of hypothesis	The null hypothesis (H0)	The alternative hypothesis (H1)
Scientific	The distribution of the specific value indicators is the same for the analogues with attribute *"X"* (set of objects I) and without it (set of objects J). There is no bias between them, the statistical estimates made for one set of objects are unbiased for the other set.	The distribution of unit values for objects from set I differs from the distribution that occurs in set J. There is a bias, the estimate made for objects belonging to one set will be biased for objects belonging to another set.
Practical	The median value of the unit value of objects with attribute *"X"* does not differ from the median value of the unit value of objects without attribute *"X"*, their medians are equal.	The median value of the unit value of objects with attribute *"X"* differs from the median value of the unit value of objects without attribute *"X"*, their medians are not equal.
Set forth in terms of valuation	The presence or absence of an attribute *"X"* does not have any noticeable effect on the value, so the attribute *"X"* is not a pricing factor.	The presence or absence of an attribute *"X"* has a noticeable effect on the value, therefore, the attribute *"X"* is a pricing factor.

descriptive statistics. It can be useful to show the properties of samples graphically for ease of primary analysis. Figure 5 shows a Boxplot diagram that allows us to draw some conclusions based on one glance. As can be seen, the values of the means and medians of the two samples are different. The minimum values also vary. At the same time, the maximum values are the same. It should also be noted that although the mean and median of the first sample are higher than those of the second, the minimum value of the first is lower than that of the second. In such circumstances, it is even more difficult to conclude whether the difference in the attribute is significant, or whether the

difference in the unit value indicator is incidental in nature. The next preparatory step

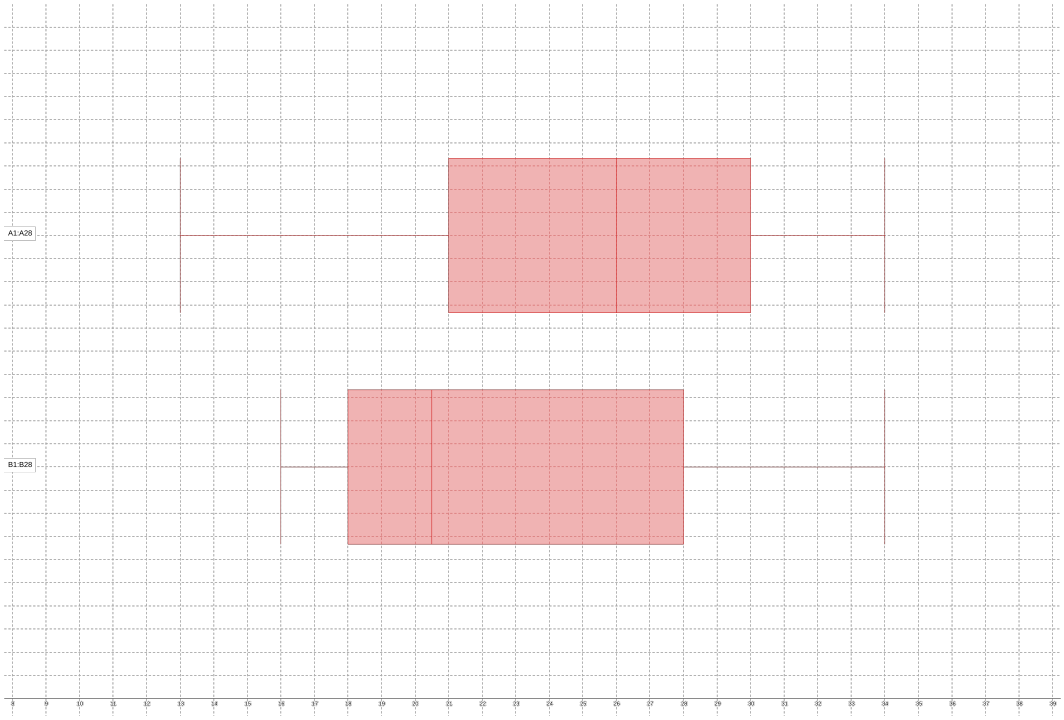

Figure 4.1: Boxplot for both samples.

is to check the normality of the distribution of values of the quantitative attribute (in this case, the notional unit cost). There are a number of rigorous tests that allow such testing by numerical methods. Appropriate ways of performing such tests will be shown in sections 4.2 and **??**. We only use the graphical method in this section. Histograms of probability density distributions for the first and second samples are shown in Figures 4.2 and 4.3, respectively. They are combined with the curves of the probability density function for the normal distribution.

The shape of the distribution of both samples differs significantly from the shape of the curve of the probability density function of the normal distribution, which can be clearly seen in the diagrams 4.2, **??**.When working with real data, quantitative tests for normality of distributions must be performed in all cases. Let us focus on the interpretation of the diagrams in this section. The shape of both histograms allows us to conclude that the distributions of both samples differ from normal. This allows us to conclude that parametric methods of statistical estimation are not applicable. It is necessary to use methods of non parametric statistics, which include the U-test, in such a situation.

There is no need to construct a common variation series for two samples when working with a spreadsheet. You can go straight to the calculation of the observation ranks through the standard LibreOffice Calc tools. Considering the possible presence of ties

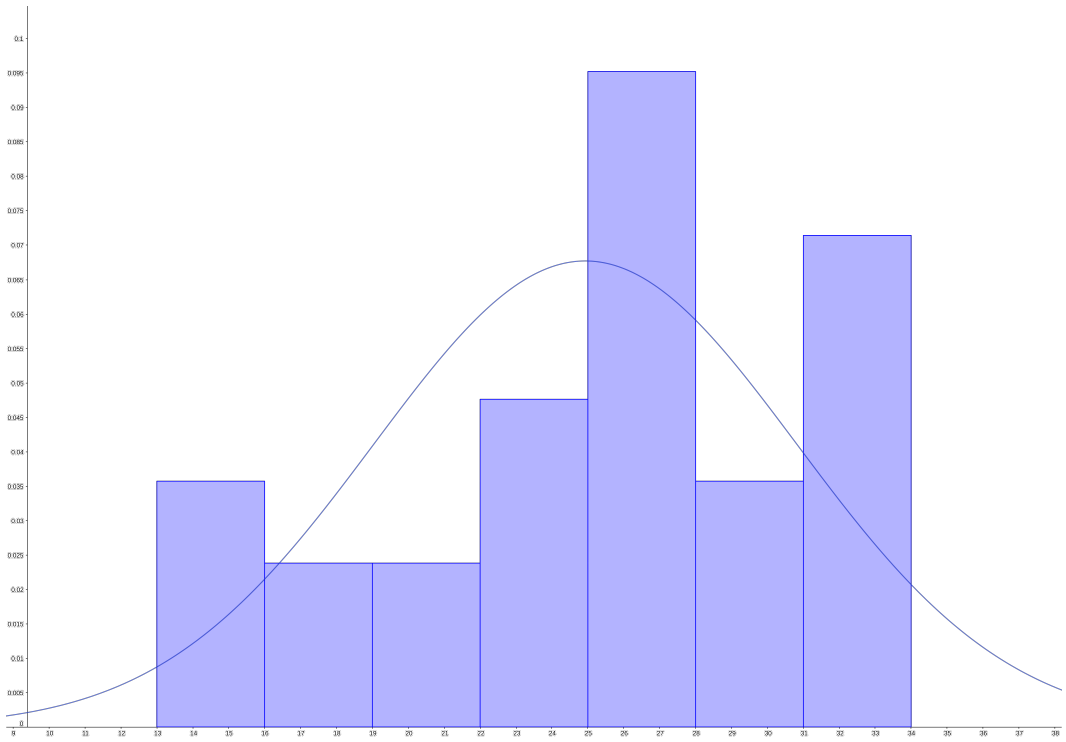

Figure 4.2: Histogram of the first sample, combined with the curve of the probability density function for the normal distribution.

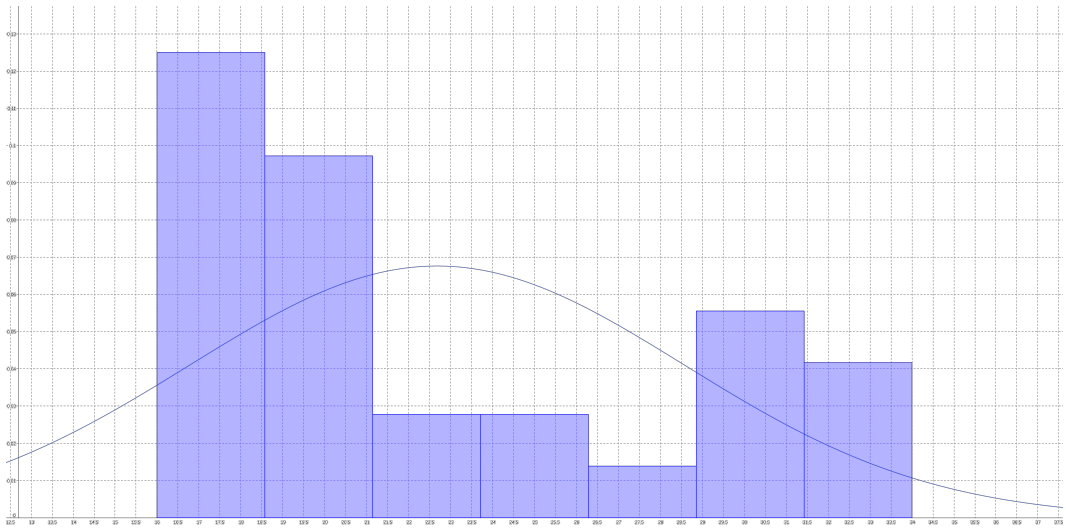

Figure 4.3: Histogram of the second sample, combined with the curve of the probability density function for the normal distribution.

(repeating values) you should use the `RANK.AVG` function. The three arguments for it to work correctly must be specified consistently:

- the observation for which the rank is calculated;
- the range of all values of the common variation series;
- sorting type: 0 — descending, 1 — ascending, in our case it is necessary to specify 1.

Columns M and O contain the ranks of the corresponding observations. Columns L and N contain duplicate values.

After that, in cells C20:C21 we calculate the sums of ranks for each of the samples. After that, calculate the total sum of the ranks of both samples in cell C22. Let's calculate the same indicator according to formula 3.5 for verification.

Next, in cells C25, C26 we calculate the values of U1 and U2 according to formulas 3.6, 3.7, respectively. Then in cell D27 we check the correctness of the control ratio 3.10. In C28 we select a smaller value, which will be used later as the U-statistic. In our case, the sample from set J has a smaller U-statistic value.

Let's calculate the CLES indicator. To do this we use formula 3.12. The result is found in C29. In this example, the value of the indicator is 0.61. This should be interpreted as follows: *the probability that the value of the unit value of a randomly chosen observation from set I is greater than that of a randomly chosen observation from set J is 0.61 (61%).* Next, we calculate the value of the rank-biserial correlation coefficient by formulas 3.15, 3.16. Put it in cell C36. In this case, the value was 0.21. So the strength of the correlation between the presence of the attribute *"X"* and the unit value of the object is 0.21.

After that, we proceed to calculate the standardized value (z-score) by formula 3.17. To do this in cell C38, calculate the mean using formula 3.18. Then we move on to the question of calculating the standard deviation. It should be noted that there are two formulas for this. The first (3.19) applies if there are no ties (cell C39). The second (3.20) applies when they are present (cell C40). In the case at hand, ties were present. They were processed in columns P and Q, as well as in cells E35:E49. In this paper, both formulas were applied. The result was two values with a difference of less than one percent. Consideration of the ties factor is necessary from the point of view of maximum scientific correctness of the result. However, some appraisers in their daily practice may find it difficult to correctly account for the bundling factor. Or they may not have enough time for additional calculations. Practical experience tells us that any significant difference in the standard deviation values obtained using these two formulas occurs in cases of a large number of ties or the presence of large groups. In other situations, a simpler formula that automatically calculates the σ value gives the correct result. Its accuracy for valuation is quite sufficient. In any case, the decision to use more scientific and rigorous or simple methods is up to the appraiser. The ties factor has been taken into account in the example under consideration. Knowing the mean and standard deviation, we calculate the z-score in cell C45. The continuity of the distribution is one of the prerequisites of the U-test. However, the empirical data are discrete. Therefore,

the implementation of the adjustment by subtracting 0.5 is necessary. Then in cell C46 we calculate the p-value by approximating the normal distribution. In this example it was 0.173. According to the rule 3.21, we conclude that it is impossible to reject the null hypothesis. Using the wording closest to the valuation activity (see Table 4.1), we can come to the following conclusion: the presence or absence of attribute "X" does not have any noticeable effect on the value, attribute "X" is not a pricing factor.

In this section, we looked at the step-by-step calculation of the criterion statistics, as well as carried out the interpretation of the result. As shown above, conducting the U-test in a spreadsheet is quite convenient. However, it is difficult to consider as a best practice. Preference should be given to professional development tools in machine learning and statistical inference. Python and R programming languages are such tools in the first place.

4.2 Python implementation.

Python language has already become the de facto standard in machine learning and especially in a number of areas such as deep learning. In addition, it is versatile and perfectly suited for the development of various expert systems. Its popularity means, among other things, that there are a huge number of tutorials on all aspects of development in the field of data analysis, designed for users of any level of training. Most of the calculations required by the appraiser can be done by calling standard functions from the data analysis plug-in libraries. These include, but are not limited to: **TensorFlow**, **NumPy**, **SciPy**, **Pandas**, **Matplotlib**, **Keras**, **SciKit-Learn**, **PyTorch**, **Scrapy**, **BeautifulSoup**. There is no need to write a large amount of code in this case. You don't need to have a deep knowledge of programming either. According to the author of this paper, the future of valuation lies precisely in the implementation of expert systems based on learning models from open market datasets. The use of Python significantly reduces the time of the U-test as will be shown below. It also allows you to create visualizations of market data without resorting to third-party tools. In addition, the use of professional functions virtually eliminates the possibility of errors in calculations. The Python language version 3.9.12 was used when writing the code. The Python language version 3.9.12 was used when writing the code. The author used the Spyder IDE (version 5.1.5) and the Jupyter Lab development environment (version 3.3.2). Python scripts containing the used code are available in the repository [23], [24].

Let us consider a real data set containing information on the unit cost of apartments in the St. Petersburg agglomeration. The data were scraped on September 28, 2021 from cian.ru and are available at the link [20]. The dataset under consideration contains 34821 observations. As you know, the St. Petersburg agglomeration includes territories that are part of the Federal City, as well as those that are formally part of the Leningrad Region (Leningrad Oblast). The division into city and region is of a purely legal nature. From a socio-economic point of view, the nearest territories of the Leningrad Region are inextricably linked to St. Petersburg and are part of one agglomeration. By the way, it is the largest in the world at this latitude. When forming the queries used in the scraping process, the southern boundary of the agglomeration was established approximately along the axis of the A-120 highway, while the northern boundary was established along the 41A-189 highway. At the same time, it included some settlements outside these boundaries, such as the cities of Kirovsk and Schlisselburg.

Let us formulate the problem. It is necessary to determine whether or not there is a statistically significant difference in the prices of objects located within the boundaries of St. Petersburg and objects formally located in the Leningrad region. Similarly to the previous case, let us formulate the null and alternative hypotheses, which this time make practical sense (see Table 4.2).

The Python language was not originally designed specifically for data analysis. Therefore, many of the functions required for calculations will be missing in its basic version. Fortunately, there are a number of libraries that contain many necessary functions. Their number is not as large as, for example, in the R language. Their functionality is also somewhat less than that of the R. However, they are exhaustive for the overwhelming

Table 4.2: The null and alternative hypotheses in the analysis of data from the St. Petersburg urban agglomeration

Type of hypothesis	The null hypothesis (H0)	The alternative hypothesis (H1)
Scientific	The distribution of unit cost indicators for apartments located within the borders of St. Petersburg and apartments located in the adjacent areas of the Leningrad Region is the same. There is no shift between the two. The statistical estimates made for the set of analogues located in one part of the agglomeration are unbiased for objects located in the other part.	The distribution of unit cost indicators for apartments located within the borders of St. Petersburg differs from the distribution of unit cost indicators for apartments located in the adjacent territories of the Leningrad Region. There is a shift. The estimate made for objects located in one part of the agglomeration will be biased for objects located in another part of it.
Practical	The median of the unit value of apartments located within the borders of St. Petersburg is equal to the median of the unit value of apartments located in the adjacent territories of the Leningrad Region.	The median of the unit value of apartments located within the borders of St. Petersburg is not equal to the median of the unit value of apartments located in the adjacent territories of the Leningrad Region.
Set forth in terms of valuation	The location of the apartment within the borders of St. Petersburg or in the adjoining areas of the Leningrad Region is not a significant difference and does not require any special accounting.	The location of the apartment within the borders of St. Petersburg or in the surrounding areas of the Leningrad Region is a significant difference and requires separate accounting.

Listing 4.1: Enabling the required libraries

```
# import libraries
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt
import scipy.stats as stats
from scipy.stats import norm
from scipy.stats import normaltest
from scipy.stats import shapiro
from scipy.stats import anderson
from scipy.stats import mannwhitneyu
```

Listing 4.2: Set the applicable level of significance

```
# set significance level
alpha = 0.05
```

number of tasks facing appraisers. We need the following libraries: **numpy**, **pandas**, **math**, **matplotlib.pyplot**, **scipy.stats** to solve the problems discussed in this paper. Use the code provided in script 4.1 to enable them.

Let us set the significance level α, which is adopted for all further work. The choice of its value is the appraiser's prerogative. However, the most common value in papers on econometrics and operations research is 0.05. Code 4.2 was used to set the significance level value.

Now everything is ready to get to work. Let's create a dataframe based on the text file containing the data set under analysis (see script 4.2). The dataframe repeats the contents of the original file exactly and contains 34821 observations and 4 variables. The following variables are present.

- Number.
- Link to ad.
- Price per 1 sq. m.
- A six-letter location code:

```
# import dataset
df = pd.read_csv('spba-flats-210928.csv')
print(df)
type(df['price_m'])
```

Listing 4.3: Creating a dataframe containing only necessary variables and freeing memory from objects not used later

```
# get only prices and counties, release RAM
df1 = df[['price_m', 'county']]
del [[df]]
```

- the first of which indicates the region (s — St. Petersburg, l — Leningrad Region);
- the second and third indicates the administrative district;
- the last three indicate the municipality or territory.

Python added its own variable containing the observation numbers. That makes five variables. Note that numbering in Python begins with zero, not one, as in most programming languages. Variables containing observation numbers and links to ads from the source file will not be used further. So we will create a new dataframe containing only the necessary variables. Let's also release the virtual memory by unloading the first dataframe from it to optimize the computer's resources. See script 4.3. Such micro-optimization does not play a role in the case at hand. However, in order to learn how to create good code, it is better to write one extra line.

The appraiser now has at his disposal a working dataframe in a convenient form. It contains data on the apartment market of the entire St. Petersburg agglomeration. Let's plot a histogram combined with the density curve for the normal distribution to form the first view about the distribution. The Heinhold-Gaede formula [3] was used to determine the rational number of intervals (histogram columns) — k.

$$k = \sqrt{n}, \tag{4.1}$$

where n — is the number of observations. 4.4. Script 4.4 was used to plot the histogram.

Consider the resulting histogram 4.4. The x-axis contains the values of prices per square meter, the y-axis the values of the probabilities of intervals. Both axes are presented in standard form. The expectation and standard deviation values for the theoretical normal distribution are also shown. As you can see, the distribution has a heavy right tail. This allows us to draw a preliminary conclusion that it is different from normal. Formalized normality tests will be performed in the future. We can limit ourselves to the primary subjective interpretation of the histogram for now. The subject of the research is the difference between the objects located in the two parts of the agglomeration. The initial dataset contains information about observations from both parts. Therefore, it will be necessary to create two separate dataframes.A small digression should be made here. Practical experience tells us that data analysis itself and model development take less than 20 % of the time, while more than 80 % is spent on data collection and preprocessing. Correct data markup is one of the important elements of these processes. In the

Listing 4.4: Plotting the histogram for the agglomeration of St. Petersburg.

```
# calculate the number of observations on data frame
spbaLenR = round(math.sqrt(len(df1.index)))

# fit a normal distribution to the data: mean and standard deviation
mu, std = norm.fit(df1["price_m"])

# plot the histogram
plt.hist(df1["price_m"], bins=spbaLenR, density=True)

# plot the PDF
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)

plt.plot(x, p, 'k', linewidth=2)
title = "Fit Values: {:.2f} and {:.2f}".format(mu, std)
plt.title(title)

# save to .pdf
plt.savefig('spba-price-histogram-py.pdf')
```

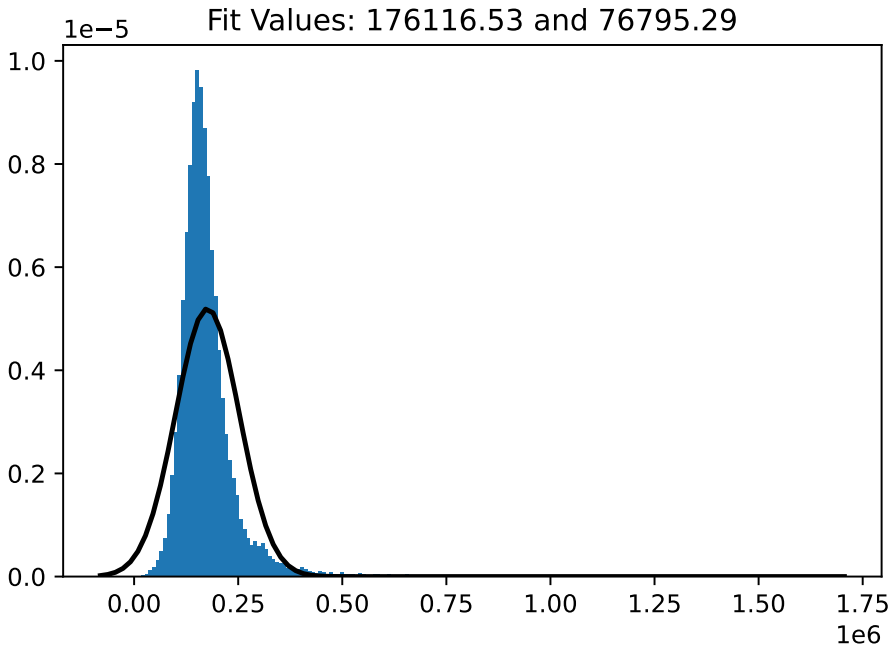

Figure 4.4: Histogram of the density distribution of apartments' prices per 1 sq. m. in agglomeration of St. Petersburg combined with the curve of the probability density function for the normal distribution.

Listing 4.5: Creating separate dataframes for St. Petersburg and the Leningrad Region.

```
# create separate dataframes for city and suburbs
dfs = df1[df1['county'].str.startswith('s')] # Saint-Petersburg
dfl = df1[df1['county'].str.startswith('l')] # Leningradskaja oblastq
```

Listing 4.6: Histogram plotting for St. Petersburg.

```
# Saint-Petersburg
# calculate the number of observations on data frame
spbLenR = round(math.sqrt(len(dfs.index)))

# fit a normal distribution to the data: mean and standard deviation
muS, stdS = norm.fit(dfs["price_m"])

# plot the histogram
plt.hist(dfs["price_m"], bins=spbLenR, density=True)

# plot the PDF
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
ps = norm.pdf(x, muS, stdS)

plt.plot(x, ps, 'k', linewidth=2)
title = "S-Pb. Fit Values: {:.2f} and {:.2f}".format(muS, stdS)
plt.title(title)

# save to .pdf
plt.savefig('spb-price-histogram-py.pdf')
```

case of the data set under consideration, their analysis in terms of individual territories up to the level of municipalities was initially provided by specifying a territory index for each observation. The first letter of the index contains an indication of which part of the agglomeration the observation is located in, as mentioned above. Two lines of code (see script 4.5) containing simple regular expressions are enough to create two separate dataframes in this case. The *dfs* dataframe contains data for observations from St. Petersburg (28643 observations). The *dfl* dataframe contains data for observations from the Leningrad Region (6178 observations). Let's plot the histograms for both parts of the agglomeration. See scripts 4.6 and 4.7.

The histogram is sometimes confused with the bar graph. Recall that a properly plotted histogram is a representation of the probabilistic properties of the data. The sum of the areas of all its rectangles must be equal to one. Therefore, the probability

Listing 4.7: Histogram plotting for the Leningrad Region

```
# LO
# calculate the number of observations on data frame
loLenR = round(math.sqrt(len(dfl.index)))

# fit a normal distribution to the data: mean and standard deviation
muL, stdL = norm.fit(dfl["price_m"])

# plot the histogram
plt.hist(dfl["price_m"], bins=loLenR, density=True)

# plot the PDF
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
pl = norm.pdf(x, muL, stdL)

plt.plot(x, pl, 'k', linewidth=2)
title = "LO. Fit Values: {:.2f} and {:.2f}".format(muL, stdL)
plt.title(title)

# save to .pdf
plt.savefig('lo-price-histogram-py.pdf')
```

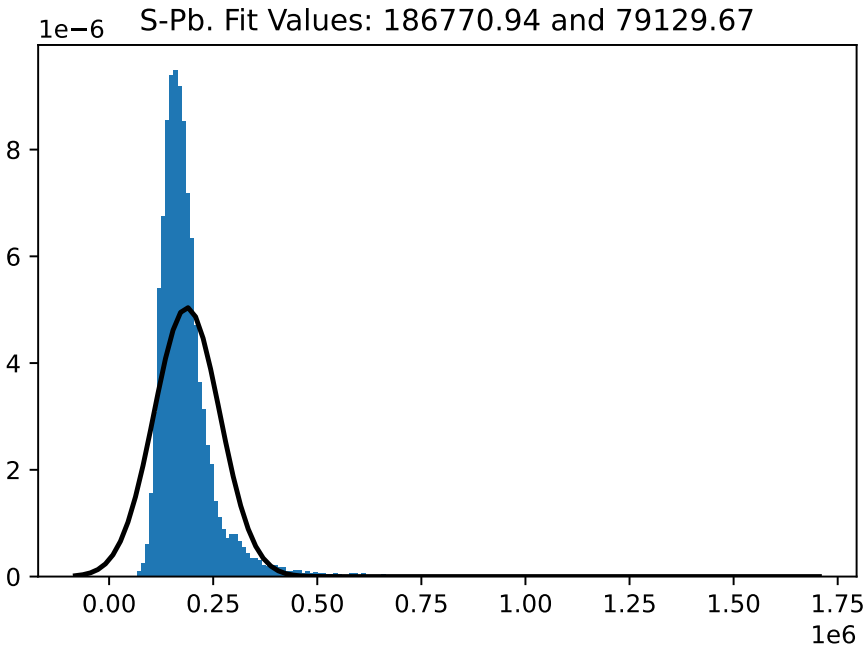

Figure 4.5: Histogram of the density distribution of apartments' prices per 1 sq. m. in St. Petersburg combined with the curve of the probability density function for the normal distribution.

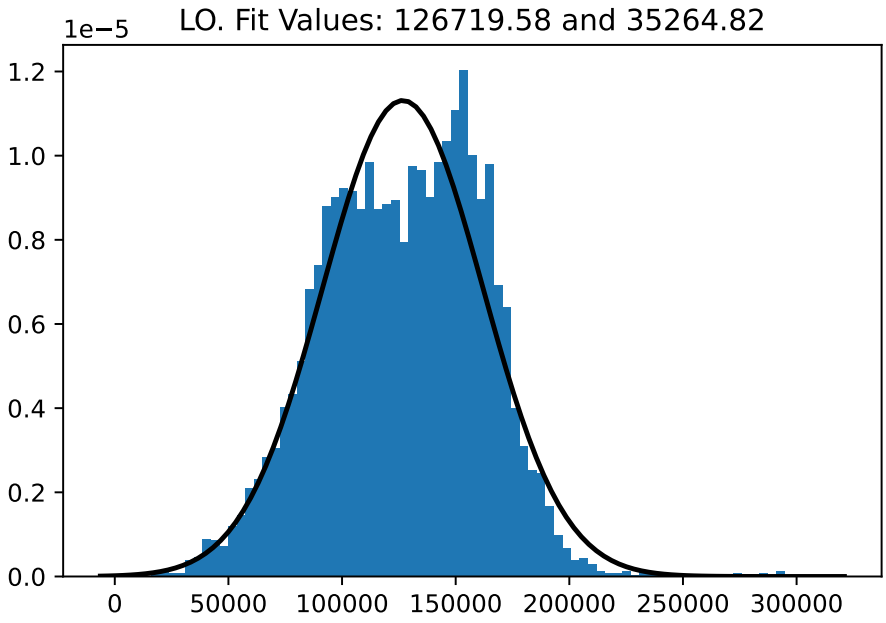

Figure 4.6: Histogram of the density distribution of apartments' prices per 1 sq. m. in the Leningrad Region located within the borders of the St. Petersburg agglomeration combined with the curve of the probability density function for the normal distribution.

Listing 4.8: Plotting the boxplot for both subsamples

```
# add labels to data
dfs["region"] = "SPb"
dfl["region"] = "LO"

# plot boxplot
prices = [dfs, dfl]
allPrices = pd.concat(prices)
plt.figure()
allPrices.boxplot(by="region")

# save to .pdf
plt.savefig('spb-lo-boxplot-py.pdf')
```

values of the ranges (histogram columns), rather than the number of observations in each range, must be plotted on the y-axis. As can be seen in histogram 4.5, the distribution of unit prices in St. Petersburg as well as in the case with the distribution of prices for the entire agglomeration has a heavy right tail. In contrast, the distribution of prices for the agglomeration objects located outside the boundaries of St. Petersburg, shown in histogram 4.6, looks relatively symmetrical.

Let's also plot the boxplot for both dataframes using script 4.8. As can be seen in Diagram 4.7, the value of median unit prices of properties located in St. Petersburg is higher than the value of the third quartile of prices of properties located in the adjacent areas of the Leningrad Region.

This circumstance allows us to make a subjective judgment that the null hypothesis should be rejected. However, graphical methods of analysis are only suitable for quick primary interpretation, as well as for presentation purposes. Performing the U-test itself will be necessary to form an objective evidentiary judgment.

The normality tests for both dataframes (*dfs*, *dfl*) should be performed to test the applicability of the U-test. There are many criteria for testing the hypothesis that the sample distribution is normal. Three tests were performed in this case:

- Shapiro– Francia test [4];
- D'Agostino's K-squared test [7];
- Anderson-Darling test [2].

The Shapiro– Francia test evaluates a sample of data and calculates how likely it is that it was drawn from a general population that has a normal distribution. This test is considered one of the most powerful normality tests [13]. At the same time, there are some premises indicating that it works well for samples of average size not exceeding five thousand observations (the minimum number is five).

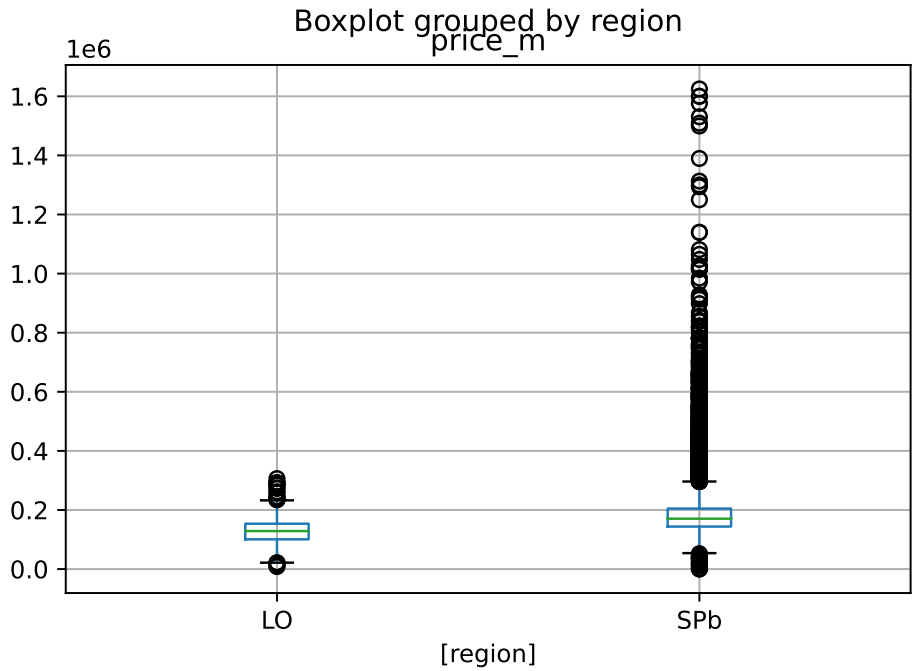

Figure 4.7: The boxplot diagram for the unit prices of apartment offers in the St. Petersburg agglomeration in the context of regional affiliation.

Listing 4.9: Performing the Shapiro-Wilk test for St. Petersburg data

```
stat, p = shapiro(dfs['price_m'])
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
if p <= alpha:
print('Sample does not look Gaussian (reject H0)')
else:
print('Sample looks Gaussian (fail to reject H0)')
```

The D'Agostino's K-squared test is based on the analysis of skewness [77] and kurtosis [65] indices representing the realizations of the third and fourth central moments [52], respectively. This test is also considered one of the most powerful and has no limit on the maximum number of observations.

The Anderson-Darling test is a modified version of the Kolmogorov-Smirnov test of goodness-of-fit [64]. It is used to test the hypothesis that the empirical distribution conforms to one of the known theoretical distributions. Its result is not the p-value, but the criterion statistics, unlike the previous two tests. This requires a more complex interpretation of the result. However, it is already automated in Python.

Let us formulate the null hypotheses:

- H0(SPb): the distribution of the values of unit prices of apartment offers in St. Petersburg does not differ from the normal distribution;
- H0(LO): the distribution of the values of apartment supply prices in the territories of the Leningrad Region that are part of the St. Petersburg agglomeration does not differ from the normal distribution.

Thus, a total of 6 tests will be performed. Their results will be summarized in Table 4.3. All three tests allowed *H0(SPb)* to be rejected for St. Petersburg data. Two of the three tests also allowed *H0(LO)* to be rejected. We can conclude that the distribution of the subsample containing data about St. Petersburg is unambiguously different from normal, based on the results of the tests. We can also conclude that the distribution of the subsample containing data about the Leningrad region differs from normal with high probability, based on the results of the tests. The use of parametric tests is therefore inappropriate. As a consequence, the U-test discussed above should be used. Now all that remains is the U-test itself. We use script 4.15 for this. Its results are presented in Table 4.4. The p-value is less than the specified level of significance. Therefore, we can make the practical conclusion that the difference in the unit value of objects located in the urban and suburban parts of the agglomeration is significant and requires appropriate accounting. Other interpretations of the result can be obtained from the *The alternative hypothesis (H1)* column of Table4.2.

Listing 4.10: Performing the Shapiro-Wilk test for Leningrad Region data

```
stat, p = shapiro(dfl['price_m'])
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
if p <= alpha:
print('Sample does not look Gaussian (reject H0)')
else:
print('Sample looks Gaussian (fail to reject H0)')
```

Listing 4.11: Performing the D'Agostino's K-squared test for St. Petersburg data

```
stat, p = normaltest(dfs['price_m'])
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
if p <= alpha:
print('Sample does not look Gaussian (reject H0)')
else:
print('Sample looks Gaussian (fail to reject H0)')
```

Listing 4.12: Performing the D'Agostino's K-squared test for Leningrad Region data

```
stat, p = normaltest(dfl['price_m'])
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
if p <= alpha:
print('Sample does not look Gaussian (reject H0)')
else:
print('Sample looks Gaussian (fail to reject H0)')
```

Listing 4.13: Performing the Anderson-Darling test for St. Petersburg data

```
result = anderson(dfs['price_m'])
print('Statistic: %.3f' % result.statistic)
p = 0
for i in range(len(result.critical_values)):
sl, cv = result.significance_level[i], result.critical_values[i]
if result.statistic < result.critical_values[i]:
print('%.3f: %.3f, data looks normal (fail to reject H0)' % (sl, cv))
else:
print('%.3f: %.3f, data does not look normal (reject H0)' % (sl, cv))
```

Listing 4.14: Performing the Anderson-Darling test for Leningrad Region data

```
result = anderson(dfl['price_m'])
print('Statistic: %.3f' % result.statistic)
p = 0
for i in range(len(result.critical_values)):
sl, cv = result.significance_level[i], result.critical_values[i]
if result.statistic < result.critical_values[i]:
print('%.3f: %.3f, data looks normal (fail to reject H0)' % (sl, cv))
else:
print('%.3f: %.3f, data does not look normal (reject H0)' % (sl, cv))
```

Table 4.3: The results of the tests of checking the data on the St. Petersburg agglomeration for normality at $\alpha = 0.05$.

The test	St. Petersburg	Leningrad Region
Shapiro-Wilk:	4.9	4.10
criterion statistic (W)	0.689	0.991
p-value	0.000	0.000
H0	rejected	rejected
K^2 D'Agostino's K-squared test:	4.11	4.12
criterion statistic (K^2)	28166.251	4.067
p-value	0.000	0.131
HO	rejected	can't be rejected
Anderson-Darling:	4.13	4.14
criterion statistic (A^2)	1688.671	15.795
H0:	rejected	rejected
Bottom line.:		
H0	rejected	rejected

Listing 4.15: Wilcoxon– Mann– Whitney test for the data of unit prices of apartment offers in the agglomeration of St. Petersburg

```
stat, p = mannwhitneyu(dfs['price_m'], dfl['price_m'])
print('stat=%.3f, p=%.3f' % (stat, p))
if p <= 0.05:
print('Probably different distributions')
else:
print('Probably the same distribution')
```

Table 4.4: The results of the U-test for the St. Petersburg agglomeration data at $\alpha = 0.05$.

The indicator	Value
Criterion statistic	142555441.000
p-value	0.000
The null hypothesis (see Table 4.2)	rejected
AUC	0.806
RBC	0.611

Listing 4.16: Calculation of AUC and RBC for St. Petersburg agglomeration data

```
# calculate AUC&RBC
n1n2 = len(dfs.index) * len(dfl.index)
auc = stat/n1n2
rbc = auc-(1-auc)
```

4.2.1 ROC analysis, AUC calculation

The theoretical issues of the relationship between the U-test and the concept of AUC were discussed earlier in 3.4. A detailed discussion of the practical issues of calculating AUC for real data is given in **??**. This subsection will show the code for calculating the AUC and RBC in Python. The code is given in script 4.16. The values obtained along with the other results are shown in Table 4.4.

Bibliography

[1] Donald R. Mann Henry B.and Whitney. "On a test of whether one of two random variables is stochastically larger than the other". In: *The annals of mathematical statistics* (1947), pp. 50–60.

[2] T. W. Anderson; D. A. Darling. "Asymptotic Theory of Certain "Goodness of Fit" Criteria Based on Stochastic Processes". English. In: *Annals of Mathematical Statistics* 23.2 (1952), pp. 193–212. DOI: 10.1214/aoms/1177729437. URL: https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-23/issue-2/Asymptotic-Theory-of-Certain-Goodness-of-Fit-Criteria-Based-on/10.1214/aoms/1177729437.full (visited on 05/29/2022).

[3] J. Heinhold and K. W. Gaede. *Ingenieur-Statistik*. 1965, p. 327.

[4] S. S. Shapiro and M. B. Wilk. "An analysis of variance test for normality (complete samples)". English. In: *Biometrika* 52 (3-4 1965-12-01), pp. 591–611.

[5] Morton B. Brown and Alan B. Forsythe. "Robust tests for the equality of variances". In: *Journal of the American Statistical Association* 346.69 (1974), pp. 364–367. DOI: 10.1080/01621459.1974.10482955. URL: https://www.tandfonline.com/doi/abs/10.1080/01621459.1974.10482955 (visited on 06/06/2022).

[6] William J. Conover. *Practical Nonparametric Statistics*. John Wiley and Sons, 1980.

[7] Ralph B. Agostino, Albert Belanger, and Jr. Ralph B. Agostino. "A suggestion for using powerful and informative tests of normality". English. In: *The American Statistician* 44.4 (1990), pp. 316–321. DOI: 10.2307/2684359. URL: https://web.archive.org/web/20120325140006/http://www.cee.mtu.edu/~vgriffis/CE%205620%20materials/CE5620%20Reading/DAgostino%20et%20al%20-%20normaility%20tests.pdf (visited on 05/29/2022).

[8] Erich L Lehamnn. "Elements of Large Sample Theory". In: *Springer* (1999), p. 176.

[9] Eiiti Kasuya. "Mann–Whitney U test when variances are unequal". In: *Animal Behaviour* 6.61 (2001), pp. 1247–1249. DOI: 10.1006/anbe.2001.1691. URL: https://www.sciencedirect.com/science/article/abs/pii/S0003347201916914 (visited on 06/06/2022).

[10] Michael C Mozer. *Optimizing classifier performance via an approximation to the Wilcoxon-Mann-Whitney statistic*. English. 2003.

[11] Corinna Cortes and Mehryar Mohri. "AUC optimization vs. error rate minimization". In: *Advances in neural information processing systems* 16.16 (2004), pp. 313–320.

[12] Tom Fawcett. "An introduction to ROC analysis". In: *Pattern Recognition Letters* 27 (2005-12-19), pp. 861–874. URL: https://web.tresorit.com/l/APSpC%5C#AfkTKO5%5C_-ijMhPuXE-qEzg (visited on 06/25/2022).

[13] . . . . 2006.

[14] Dennis D. Boos and L. A. Stefanski. *Essential Statistical Inference*. English. Springer, 2013.

[15] The IFRS Foundation. *IFRS 13 Fair Value Measurement*. UK, London: The IFRS Foundation, 2016-01-31. URL: http://eifrs.ifrs.org/eifrs/bnstandards/en/IFRS13.pdf (visited on 06/10/2020).

[16] E. Brunner and Arne C Bathke. *Rank and pseudo-rank procedures for independent observations in factorial designs: Using R and SAS*. English. Springer International Publishing, 2018. ISBN: 978-3-030-02912-8.

[17] Julian D. Karch. "Psychologists Should use Brunner-Munzel's instead of Mann-Whitney's U-test as the Default Nonparametric Procedure". In: *Advances in Methods and Practices in Psychological Science* 2.4 (2021). ISSN: 2515-2459. DOI: 10.1177/2515245921999602. URL: https://journals.sagepub.com/doi/10.1177/2515245921999602 (visited on 06/06/2022).

[18] K. A. Murashev. "Short Introduction to the differences between Frequentist and Bayesian approaches to probability in valuation". In: (2021-10-10). URL: https://github.com/Kirill-Murashev/AI_for_valuers_book/tree/main/Parts-Chapters/Frequentist-and-Bayesian-probability (visited on 05/23/2022).

[19] Royal Institution of Chartered Surveyors (RICS). *RICS Valuation — Global Standards*. English. UK, London: RICS, 2021-11-30. URL: https://www.rics.org/uk/upholding-professional-standards/sector-standards/valuation/red-book/red-book-global/ (visited on 05/11/2022).

[20] . . . *spba-flats-210928*. 2021-09-28. URL: https://github.com/Kirill-Murashev/datasets/blob/main/Saint-Petersburg/flats/spba_flats_210928.csv.

[21] International Valuation Standards Council. *International Valuation Standards*. 2022-01-31. URL: https://www.rics.org/uk/upholding-professional-standards/sector-standards/valuation/red-book/international-valuation-standards/.

[22] K. A. Murashev. "Practical application of the Mann-Whitney-Wilcoxon test (U-test) in valuation". English, Spanish, Russian, Interslavic. In: (2022-05-15). URL: https://github.com/Kirill-Murashev/AI_for_valuers_book/tree/main/Parts&Chapters/Mann-Whitney-Wilcoxon (visited on 05/15/2022).

[23] URL: https://github.com/Kirill-Murashev/AI_for_valuers_book/blob/main/Parts-Chapters/Mann-Whitney-Wilcoxon/U-test.py.

[24] URL: https://github.com/Kirill-Murashev/AI_for_valuers_book/blob/main/Parts-Chapters/Mann-Whitney-Wilcoxon/U-test.ipynb.

[25] *AUC-Derivation.ipynb*. URL: https://colab.research.google.com/github/unpingco/Python-for-Signal-Processing/blob/master/AUC_Derivation.ipynb#scrollTo=BgrH5C49LqMx (visited on 06/14/2022).

[26] Creative Commons. *cc-by-sa-4.0*. URL: https://creativecommons.org/licenses/by-sa/4.0/ (visited on 01/27/2021).

[27] *F-test*. URL: https://en.wikipedia.org/wiki/F-test (visited on 06/06/2022).

[28] *Fligner-Policello test in R*. URL: https://search.r-project.org/CRAN/refmans/RVAideMemoire/html/fp.test.html (visited on 06/06/2022).

[29] Python Software Foundation. *Python site*. . Python Software Foundation. URL: https://www.python.org/ (visited on 08/17/2021).

[30] The Document Foundation. *LibreOffice Calc*. . URL: https://www.libreoffice.org/discover/calc/ (visited on 08/20/2021).

[31] *GeoGebra official site*. URL: https://www.geogebra.org/ (visited on 08/26/2021).

[32] Bob Horton. *AUC Meets the Wilcoxon-Mann-Whitney U-Statistic*. URL: https://blog.revolutionanalytics.com/2017/03/auc-meets-u-stat.html (visited on 07/14/2022).

[33] Bob Horton. *Calculating AUC: the area under a ROC Curve*. URL: https://blog.revolutionanalytics.com/2016/11/calculating-auc.html (visited on 07/14/2022).

[34] Bob Horton. *ROC Curves in Two Lines of R Code*. URL: https://blog.revolutionanalytics.com/2016/08/roc-curves-in-two-lines-of-code.html (visited on 07/14/2022).

[35] *If p-value is exactly equal to 0.05, is that significant or insignificant?* URL: https://www.researchgate.net/post/If_p-value_is_exactly_equal_to_005_is_that_significant_or_insignificant.

[36] *Jupyter site*. URL: https://jupyter.org (visited on 05/13/2022).

[37] Nicolas Kruchten. *Machine Learning Meets Economics*. URL: http://nicolas.kruchten.com/content/2016/01/ml-meets-economics/ (visited on 07/11/2022).

[38] Machinelearning.ru. *U- - *. Russian. URL: http://www.machinelearning.ru/wiki/index.php?title=%D0%9A%D1%80%D0%B8%D1%82%D0%B5%D1%80%D0%B8%D0%B9_%D0%A3%D0%B8%D0%BB%D0%BA%D0%BE%D0%BA%D1%81%D0%BE%D0%BD%D0%B0-%D0%9C%D0%B0%D0%BD%D0%BD%D0%B0-%D0%A3%D0%B8%D1%82%D0%BD%D0%B8 (visited on 05/14/2022).

[39] Machinelearning.ru. . URL: http://www.machinelearning.ru/wiki/index.php?title=%D0%93%D0%B8%D0%BF%D0%BE%D1%82%D0%B5%D0%B7%D0%B0_%D1%81%D0%B4%D0%B2%D0%B8%D0%B3%D0%B0 (visited on 05/15/2022).

[40] Machinelearning.ru. . Russian. URL: http://www.machinelearning.ru/wiki/index.php?title=%D0%9A%D1%80%D0%B8%D1%82%D0%B5%D1%80%D0%B8%D0%B9_%D0%A3%D0%B8%D0%BB%D0%BA%D0%BE%D0%BA%D1%81%D0%BE%D0%BD%D0%B0_%D0%B4%D0%B2%D1%83%D1%85%D0%B2%D1%8B%D0%B1%D0%BE%D1%80%D0%BE%D1%87%D0%BD%D1%8B%D0%B9 (visited on 05/14/2022).

[41] *Ordered logit*. URL: https://en.wikipedia.org/wiki/Ordered_logit (visited on 06/06/2022).

[42] Kennis Research. *Receiver Operating Characteristic (ROC) Curves*. URL: https://kennis-research.shinyapps.io/ROC-Curves/ (visited on 06/25/2022).

[43] *Spyder IDE site*. URL: https://www.spyder-ide.org/.

[44] PBC Studio. *RStudio official site*. . URL: https://www.rstudio.com/ (visited on 08/19/2021).

[45] CTAN team. *TeX official site*. English. CTAN Team. URL: https://www.ctan.org/ (visited on 11/15/2020).

[46] LaTeX team. *LaTeX official site*. English. URL: https://www.latex-project.org/ (visited on 11/15/2020).

[47] *TeXLive official site*. URL: https://www.tug.org/texlive/ (visited on 11/15/2020).

[48] The R Foundation. *The R Project for Statistical Computing*. . The R Foundation. URL: https://www.r-project.org/ (visited on 08/17/2021).

[49] *Turtle graphics*. URL: https://en.wikipedia.org/wiki/Turtle_graphics (visited on 06/22/2022).

[50] *Welch-t-test*. URL: https://en.wikipedia.org/wiki/Welch's_t-test (visited on 06/06/2022).

[51] Wikipedia. *Accuracy and precision*. URL: https://en.wikipedia.org/wiki/Accuracy_and_precision (visited on 08/09/2022).

[52] Wikipedia. *Central Moment*. URL: https://en.wikipedia.org/wiki/Central_moment (visited on 05/29/2022).

[53] Wikipedia. *Commom Language Effect Size*. English. URL: https://en.wikipedia.org/wiki/Effect_size#Common_language_effect_size (visited on 05/16/2022).

[54] Wikipedia. *Diagnostic odds ratio*. URL: https://en.wikipedia.org/wiki/Diagnostic_odds_ratio (visited on 08/10/2022).

[55] Wikipedia. *Dot product*. URL: https://en.wikipedia.org/wiki/Dot_product (visited on 07/14/2022).

[56] Wikipedia. *F-score*. URL: https://en.wikipedia.org/wiki/F-score (visited on 08/09/2022).

[57] Wikipedia. *False discovery rate*. URL: https://en.wikipedia.org/wiki/False_discovery_rate (visited on 08/09/2022).

[58] Wikipedia. *False positive rate*. URL: https://en.wikipedia.org/wiki/False_positive_rate (visited on 08/08/2022).

[59] Wikipedia. *Fowlkes–Mallows index*. URL: https://en.wikipedia.org/wiki/Fowlkes%E2%80%93Mallows_index (visited on 08/10/2022).

[60] Wikipedia. *Hit rate*. URL: https://en.wikipedia.org/wiki/Hit_rate (visited on 08/08/2022).

[61] Wikipedia. *Information retrieval*. URL: https://en.wikipedia.org/wiki/Information_retrieval (visited on 08/09/2022).

[62] Wikipedia. *Jaccard index*. URL: https://en.wikipedia.org/wiki/Jaccard_index (visited on 08/09/2022).

[63] Wikipedia. *KISS principle*. URL: https://en.wikipedia.org/wiki/KISS_principle (visited on 11/06/2020).

[64] Wikipedia. *Kolmogorov–Smirnov test*. URL: https://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test (visited on 05/29/2022).

[65] Wikipedia. *Kurtosis*. URL: https://en.wikipedia.org/wiki/Kurtosis (visited on 05/29/2022).

[66] Wikipedia. *Likelihood ratios in diagnostic testing*. URL: https://en.wikipedia.org/wiki/Likelihood_ratios_in_diagnostic_testing (visited on 08/09/2022).

[67] Wikipedia. *Markedness*. URL: https://en.wikipedia.org/wiki/Markedness (visited on 08/10/2022).

[68] Wikipedia. *Phi coefficient*. URL: https://en.wikipedia.org/wiki/Phi_coefficient (visited on 08/10/2022).

[69] Wikipedia. *Positive and negative predictive values*. URL: https://en.wikipedia.org/wiki/Positive_and_negative_predictive_values (visited on 08/09/2022).

[70] Wikipedia. *Precision and recall*. URL: https://en.wikipedia.org/wiki/Precision_and_recall (visited on 08/08/2022).

[71] Wikipedia. *Prevalence*. URL: https://en.wikipedia.org/wiki/Prevalence (visited on 08/10/2022).

[72] Wikipedia. *Rank-biserial correlation*. English. URL: https://en.wikipedia.org/wiki/Effect_size#Rank-biserial_correlation (visited on 05/16/2022).

[73] Wikipedia. *Receiver operating characteristic*. URL: https://en.wikipedia.org/wiki/Receiver_operating_characteristic (visited on 05/17/2022).

[74] Wikipedia. *Sensitivity and specificity*. URL: https://en.wikipedia.org/wiki/Sensitivity_and_specificity (visited on 08/08/2022).

[75] Wikipedia. *Sigmoid function*. URL: https://en.wikipedia.org/wiki/Sigmoid_function (visited on 06/24/2022).

[76] Wikipedia. *Simple random sample*. URL: https://en.wikipedia.org/wiki/Simple_random_sample (visited on 07/26/2022).

[77] Wikipedia. *Skewness*. URL: https://en.wikipedia.org/wiki/Skewness (visited on 05/29/2022).

[78] Wikipedia. *Standard score*. URL: https://en.wikipedia.org/wiki/Standard_score (visited on 05/17/2022).

[79] Wikipedia. *Type I and type II errors*. URL: https://en.wikipedia.org/wiki/Type_I_and_type_II_errors (visited on 06/08/2022).

[80] Wikipedia. *Type I and type II errors*. URL: https://en.wikipedia.org/wiki/Type_I_and_type_II_errors (visited on 08/08/2022).

[81] Wikipedia. *Youden's J statistic*. URL: https://en.wikipedia.org/wiki/Youden's_J_statistic (visited on 08/10/2022).

[82] Benito van der Zander. *TeXstusio official site*. URL: https://www.texstudio.org/ (visited on 11/15/2020).