Algorithmic Statistics and Prediction for Polynomial Time-Bounded Algorithms

Alexey Milovanov(✉)

National Research University Higher School of Economics,
Moscow Institute of Physics and Technology, Moscow, Russia
almas239@gmail.com

Abstract. Algorithmic statistics studies explanations of observed data that are good in the algorithmic sense: an explanation should be simple i.e. should have small Kolmogorov complexity and capture all the algorithmically discoverable regularities in the data. However this idea can not be used in practice as is because Kolmogorov complexity is not computable.

In recent years resource-bounded algorithmic statistics were created [7,8]. In this paper we prove a polynomial-time version of the following result of 'classic' algorithmic statistics.

Assume that some data were obtained as a result of some unknown experiment. What kind of data should we expect in similar situation (repeating the same experiment)? It turns out that the answer to this question can be formulated in terms of algorithmic statistics [6]. We prove a polynomial-time version of this result under a reasonable complexity theoretic assumption. The same assumption was used by Antunes and Fortnow [1].

1 Introduction

Here we give some basic notation and present results about algorithmic statistics and prediction for general (without resource restrictions) algorithms.

1.1 Algorithmic Statistics

Let x be a binary string, and let A be a finite set of binary strings containing x. Considering A as an "explanation" (statistical model) for x, we want A to be as simple and small as possible. This approach can be made formal in the framework of algorithmic information theory, where the notion of Kolmogorov complexity of a finite object is defined. The definition and basic properties of Kolmogorov complexity can be found in [5,9,11]. Informally Kolmogorov complexity $\mathrm{C}(x)$ of a string x is defined as the minimal length of a program that produces x.

We also use another basic notion of the algorithmic information theory, the *discrete a priori probability*. Consider a probabilistic machine V without input that outputs some binary string and stops. It defines a probability distribution

F. Manea et al. (Eds.): CiE 2018, LNCS 10936, pp. 287–296, 2018.
https://doi.org/10.1007/978-3-319-94418-0_29

on binary strings: $\mathrm{m}_V(x)$ is the probability to get x as the output of V. There exists a universal machine U[5,11] such that m_U is maximal up to $O(1)$-factor among all m_V. We fix some U with this property and call $\mathrm{m}_U(x)$ the *discrete a priori probability of* x, denoted as $\mathrm{m}(x)$. The function m is closely related to Kolmogorov complexity: the value $-\log_2 \mathrm{m}(x)$ is equal to $\mathrm{C}(x)$ with $O(\log \mathrm{C}(x))$-precision.

Now we can define two parameters that measure the quality of a finite set A as a model for its element x: the complexity $\mathrm{C}(A)$ of A and the binary logarithm $\log |A|$ of its size. The first parameter measures how simple is our explanation; the second one measures how specific it is. We use binary logarithms to get both parameters in the same scale: to specify an element of a set of size N we need $\log N$ bits of information.

There is a trade-off between two parameters. The singleton $A = \{x\}$ is a very specific description, but its complexity may be high. On the other hand, for a n-bit string x the set $A = \{0,1\}^n$ of all n-bit strings is simple, but it is large. To analyze this trade-off, following [3,4], let us note that every set A containing x leads to a *two-part description of* x: first we specify A using $\mathrm{C}(A)$ bits, and then we specify x by its ordinal number in A, using $\log |A|$ bits. In total we need $\mathrm{C}(A) + \log |A|$ bits to specify x (plus logarithmic number of bits to separate two parts of the description). This gives the inequality

$$\mathrm{C}(x) \le \mathrm{C}(A) + \log |A| + O(\log \mathrm{C}(A)).$$

$$\text{The difference } \delta(x, A) = \mathrm{C}(A) + \log |A| - \mathrm{C}(x)$$

is called *optimality deficiency* of A (as a model for x). As usual in algorithmic statistic, all our statements are made with logarithmic precision (with error tolerance $O(\log n)$ for n-bit strings), so we ignore the logarithmic terms and say that $\delta(x, A)$ is positive and measures the overhead caused by using two-part description based on A instead of the optimal description for x.

One could wonder why we consider only sets as explanations and not general probability distributions (in other terms, why we restrict ourselves to uniform probability distributions). The reason is that this extension is not essential: for every string x and for every distribution μ there exists a set $A \ni x$ explaining x that is almost as good as μ, as the following observation shows:

Proposition 1 ([14]). *For every string x and for every distribution μ there exists a set $A \ni x$ such that $\mathrm{C}(A|\mu) \le O(\log |x|)$ and $\frac{1}{|A|} \ge \frac{1}{2}\mu(x)$.*

There exists another approaches to algorithmic statistics (see [10,13,15]) however they are essentially equivalent.

1.2 Prediction Hierarchy

Assume that we have some experimental data represented as a binary string x. We look for a good statistical model for x and find some set A that has small optimality deficiency $\delta(x, A)$. The problem, however, is that many different models with small optimality deficiency may exist for a given x. If we want to cover

all the possibilities, we need to consider the union of all these sets, so we get the following definition.

Definition 1. *Let $x \in \{0,1\}^n$ be a binary string and let d be some integer. The union of all finite sets of strings $A \subset \{0,1\}^n$ such that $x \in A$ and $\delta(x, A) \leq d$ is called* algorithmic prediction d-neighborhood of x.

Obviously d-neighborhood increases as d increases.

There is another natural approach to prediction. Since we treat the experiment as a black box (the only thing we know is its outcome x), we assume that the possible models $A \subset \{0,1\}^n$ are distributed according to their a priori probabilities, and consider the following two-stage process. First, a finite set is selected randomly: a non-empty set A is chosen with probability $\mathrm{m}(A)$. Second, a random element x of A is chosen uniformly. In this process every string x is chosen with probability

$$\sum_{A \ni x} \mathrm{m}(A)/|A|.$$

For a given pair of strings x and y consider the conditional probability

$$P_x(y) := \Pr[y \in A \mid \text{the output of the two-stage process is } x].$$

Having some string x and some threshold d, we now can consider all strings y such that $P_x(y) \geq 2^{-d}$ (we use the logarithmic scale to facilitate the comparison with algorithmic prediction). These strings could be considered as plausible ones to appear when repeating the experiment of unknown nature that once gave x.

Definition 2. *Let x be a binary string and let d be an integer. The set of all strings y such that $p_x(y) \geq 2^{-d}$ is called* probabilistic prediction d-neighborhood of x.

It turns out that this approach is essentially equivalent to algorithmic prediction neighborhood.

Theorem 1 ([6]). (a) *For every n-bit string x and for every d the algorithmic prediction d-neighborhood is contained in probabilistic prediction $d + O(\log n)$-neighborhood.*

(b) *For every n-bit string x and for every d the probabilistic prediction d-neighborhood of x is contained in algorithmic prediction $d + O(\log n)$-neighborhood.*

Our main result is a version of this theorem for time-bounded algorithms.

2 Algorithmic Statistics for Polynomial Time

Here we present our approach to polynomial time-bounded algorithmic statistics. As explanations for strings we consider probability distributions over the set of binary strings. We can not limit ourself by sets (uniform distributions) since an analogue of Proposition 1 for polynomial time-bounded algorithms is unknown.

Let a probability distribution μ be an explanation for a string x. There is a natural parameter measuring how good is μ as an explanation for x, namely $\mu(x)$. Also we need to measure simplicity of μ. A probability distribution is called simple, if it can be sampled by a short probabilistic program with no input in polynomial time. A formal definition can be done by using the notion of *universal machines*—see [8]. There are other ways to measure acceptability of a distribution as explanation to stings [8]. However the way discussed above is the most usable for our investigation.

To measure "simplicity" we will use the notion of time-bounded *prefix-free* Kolmogorov complexity $\mathrm{K}^t(x)$. Informally it is defined as the minimal length of a prefix-free program that produces x in at most t steps (see [5] for more details). In fact the difference between prefix free and plain time-bounded complexities is not essential (the plain complexity bounded by time t of a string x is denoted by $\mathrm{C}^t(x)$).

Proposition 2 ([5]). *For every string x and for every t there exists c such that:*

(a) $\mathrm{C}^t(x) \leq \mathrm{K}^t(x) + c$.
(b) $\mathrm{K}^{ct\log^2 t}(x) \leq \mathrm{C}^t(x) + c\log|x|$.

Models of Restricted Type

So far we considered arbitrary distributions as models (statistical hypotheses). However, in practice we usually have some a priori information about the data. We know that the data was obtained by sampling with respect to an unknown probability distribution from a known family of distributions $\mathcal{M}$.

For example, we can consider the family of uniform distribution on Hamming balls as $\mathcal{M}$. (That means we know a priori that our string was obtained by flipping certain number of bits in an unknown string.) Restricting the class of allowed hypotheses was initiated in [15].

In our paper we will consider families with the following properties:

- Every element from $\mathcal{M}$ is a distribution on the strings of the same length. The family of distribution on $\{0,1\}^n$ that belong to $\mathcal{M}$ is denoted by $\mathcal{M}_n$.
- There exists a polynomial q such that $|\mathcal{M}_n| = 2^{q(n)}$ for every n.
- There exists a polynomial t such that for every $\mu_i \in \mathcal{M}_n$ there exists a program p_i that samples μ_i in time $t(n)$. (This means that for every x of length n the probability of the event "p_i outputs x" equals $\mu_i(x)$ and the running time of p_i is at most $t(n)$ for all outcomes of coin tossing.) Moreover there exists a deterministic program $p_{\mathcal{M}}$ that for $i \in \{0,1\}^{q(n)}$ outputs the program p_i in time $t(n)$.
- For every string x there exists $\mu \in \mathcal{M}$ such that $\mu(x) = 1$. Moreover the program that samples this distribution can be obtained as $p_{\mathcal{M}}(x0^{q(n)-n})$ where n is the length of x.

Any family of distributions that satisfies these four conditions is called *acceptable*. For example, the family of uniform distribution on Hamming balls is acceptable.

If a probability distribution $\mu \in \mathcal{M}$ is sampled by a program $p_i = p(i)$ then it is natural to compare $\mathrm{K}^{\mathrm{poly}}(i) - \log \mu(x)$ with $\mathrm{K}^{\mathrm{poly}}(x)$ (the difference between these values is an analogue of optimality deficiency in 'classic' algorithmic statistics). If $\mathrm{K}^{\mathrm{poly}}(i) - \log \mu(x) - \mathrm{K}^{\mathrm{poly}}(x) \approx 0$ then μ is called *optimal distribution* for x. Here is a formal definition.

Definition 3. *A distribution μ in an acceptable family $\mathcal{M}$ is called $\mathcal{M}, d, t_1, t_2$-optimal for a string x if the distribution μ can be sampled by a probabilistic program $p_{\mathcal{M}}(i) \in \mathcal{M}$ in time t_1 such that*

$$\mathrm{K}^{t_1}(i) - \log \mu(x) - \mathrm{K}^{t_2}(x) \leq d.$$

3 Prediction Hierarchy in Polynomial Time

Here for a given acceptable family $\mathcal{M}$ we introduce notions of algorithmic and probabilistic prediction neighborhoods. For simplicity first we will consider only families of **uniform** distributions.

Definition 4. *Let $x \in \{0,1\}^n$, let d, t_1, t_2 be some integers and let $\mathcal{M}$ be an acceptable family of uniform distributions. The set of all strings y such that there exists $\mu \in \mathcal{M}$ such that*

- *$\mu(y) > 0$,*
- *μ is d, t_1, t_2-optimal for x*

is called $\mathcal{M}$-algorithmic prediction d, t_1, t_2-neighborhood of x.

Such d, t_1, t_2-neighborhood increases as d and t_1 increases and t_2 decreases.

To define probabilistic prediction neighborhood we need first to recall the time-bounded version of discrete a priori probability. The t-bounded discrete a priori probability of string x is defined as

$$\mathrm{m}^t(x) = 2^{-\mathrm{K}^t(x)}.$$

Now we present results that show that this definition is consistent with the unbounded definition.

Definition 5. *A probability distribution σ over $\{0,1\}^*$ is called P-samplable, if there is randomized machine M so that $\Pr[M \text{ output } x] = \sigma(x)$ and M runs a polynomial time of the length of the output.*

Theorem 2 ([2]). *For every polynomial p, there are a P-samplable distribution σ and a constant c such that for every string x*

$$\sigma(x) \geq \frac{1}{|x|^c} \mathrm{m}^p(x).$$

The inequality in the opposite direction holds under the following assumption.

Assumption 1. *There is a set which is decidable by deterministic Turing machines in time* $2^{O(n)}$ *but is not decidable by deterministic Turing machines in space* $2^{o(n)}$ *for almost all* n.

Theorem 3 (Lemma 3.2 in [1]). *Under Assumption 1 for every* P*-samplable probability distribution* σ *there is number* d *such that for all* x *of length* n,

$$\mathrm{m}^{n^d}(x) \geq \frac{\sigma(x)}{n^d}.$$

Now we are ready to define $\mathcal{M}$-probabilistic prediction neighborhood. Recall that by the 2nd and the 4th properties of acceptability of $\mathcal{M}$ there exists a polynomial q such that every string in $\{0,1\}^{q(n)}$ defines a distribution in $\mathcal{M}$.

Consider the following two-stage process for given polynomial t. First a string $s \in \{0,1\}^{q(n)}$ is selected randomly with probability $\mathrm{m}^{t(n)}(s)$. This string s defines a distribution $\mu_s \in \mathcal{M}$. Then a string $x \in \{0,1\}^n$ is randomly chosen according the distribution μ_s. In this process every string x is chosen with probability

$$\sum_s \mathrm{m}^{t(n)}(s)\mu_s(x).$$

Consider the following probability.

$$P^t_{x,\mathcal{M}}(y) = \Pr[\mu_s(y) > 0 \mid \text{the output of the two-stage process is } x]. \quad (1)$$

Note that μ_s is a uniform distribution (now we consider only such families $\mathcal{M}$).

Definition 6. *Let* x *be a binary string, let* d *be an integer,* t *be a polynomial and* $\mathcal{M}$ *be an acceptable family. The set of all strings* y *such that* $P^t_{x,\mathcal{M}}(y) \geq 2^{-d}$ *is called* $\mathcal{M}$-probabilistic prediction d,t-neighborhood of x.

Our main result is the following

Theorem 4. (a) *Under Assumption 1 the following holds. For every polynomial* t *there exists polynomial* r *such that for every* n*-bit string* x *and for every* d *the* $\mathcal{M}-$*algorithmic prediction* $d, t(n), r(n)$*-neighborhood of* x *is contained in* $\mathcal{M}-$*probabilistic prediction* $d + O(\log n), t$*-neighborhood of* x.

(b) *Under Assumption 1 the following holds. For every polynomial* t *there exists a polynomial* r *such that for every* n*-bit string* x *and for every* d *the* $\mathcal{M}$*-probabilistic prediction* d, t*-neighborhood of* x *is contained in* $\mathcal{M}$*-algorithmic prediction* $d + O(\log n), r(n), t(n)$*-neighborhood.*

Non-uniform Distribution

Here we extend the notions of algorithmic and probabilistic prediction neighborhoods to arbitrary acceptable family of distribution M. Before we define algorithmic neighborhood note that now the condition $\mu(y) > 0$ is very weak (it is possible that for every y the value $\mu(y)$ is very small but positive). By this reason we have to add a new parameter.

Definition 7. *Let $x \in \{0,1\}^n$, let d, k, t_1, t_2 be some integers and let $\mathcal{M}$ be an acceptable family of distributions. The set of all strings y such that there exists $\mu \in \mathcal{M}$ such that*

- $\mu(y) > 2^{-k}$,
- *μ is d, t_1, t_2-optimal for x*

is called $\mathcal{M}$-algorithmic prediction d, k, t_1, t_2-neighborhood.

Such d, k, t_1, t_2-neighborhood increases as d, k and t_1 increases and t_2 decreases. To define the probability neighborhood we consider the same 2-stage process. However now we consider another the conditional probability for given x and y.

$$p^r_{x,h,\mathcal{M}}(y) = \Pr[\mu_s(y) > 2^{-h} \mid \text{the output of the two-stage process is } x]. \quad (2)$$

Definition 8. *Let x be a binary string, let λ, h be integers, r be a polynomial and $\mathcal{M}$ be an acceptable family. The set of all strings y such that $p^r_{x,h,\mathcal{M}}(y) \geq 2^{-\lambda}$ is called* $\mathcal{M}$-probabilistic prediction λ, h, r-neighborhood of x.

The generalization of Theorem 4 is the following.

Theorem 5. (a) *Under Assumption 1 the following holds. For every polynomials t there exists polynomial r such that for every n-bit string x and for every d and k the $\mathcal{M}-$algorithmic prediction $d, k, t(n), r(n)$-neighborhood of x is contained in $\mathcal{M}-$probabilistic prediction λ, h, t-neighborhood of x if $\lambda \geq d - \min(0, h-k) + O(\log n)$.*

(b) *Under Assumption 1 the following holds. For every polynomial t there exists a polynomial r such that for every n-bit string x and for every d the $\mathcal{M}$-probabilistic prediction λ, h, t-neighborhood of x is contained in $\mathcal{M}$-algorithmic prediction $d, k, t(n), r(n)$-neighborhood of x if $\lambda \geq d + \min(0, h-k) + O(\log n)$.*

4 Proof of Theorem 4

Proof (of Theorem 4(a)). This direction is simple. Assume that y belongs to $\mathcal{M}-$algorithmic prediction $d, t(n), r(n)$-neighborhood of x. Here r is a polynomial that we will define later. By definition this means that there exists $\mu \in \mathcal{M}$ such that $\mu(y) > 0$ and μ is $d, t(n), r(n)$-optimal. The later means that for some i the following inequality holds:

$$\mathrm{K}^{t(n)}(i) - \log \mu_i(x) - \mathrm{K}^{r(n)}(x) \leq d. \quad (3)$$

We need to show that y belongs to $\mathcal{M}$-probabilistic prediction $d + O(\log n), t$-neighborhood of x, i.e. $P^t_{x,\mathcal{M}}(y) \geq 2^{-d-O(\log n)}$ (see (1)). By definition (1) can be rewritten as

$$P^t_{x,\mathcal{M}}(y) = \frac{\sum_{s:\mathrm{m}_s(y)>0} \mathrm{m}^{t(n)}(s)\mu_s(x)}{\sum_s \mathrm{m}^{t(n)}(s)\mu_s(x)}. \quad (4)$$

Now we choose polynomial r such that the denominator of (4) is not greater than $\mathrm{m}^{r(n)}(x)2^{O(\log n)}$. Under Assumption 1 such polynomial r exists. Indeed,

the denominator of (4) defines a P-sample distribution that can be dominated by a polynomial-time bounded discrete a priori probability by Theorem 3.

The sum at the numerator of (4) is not less than one term that obtained by taking $s = i$. So, from (3) it follows that the numerator is not less than $\mathrm{m}^{r(n)}(x)2^{-d}$. Hence, $P^t_{x,\mathcal{M}}(y) \geq 2^{-d-O(\log n)}$. Therefore y belongs to $\mathcal{M}$-probabilistic prediction $d + O(\log n), t$-neighborhood of x.

We will derive Theorem 4(b) from the following lemma.

Lemma 1. *For every polynomial t under Assumption 1 there exists polynomial r such the following hold. Let x and y be strings of length n and let $\mathcal{M}$ be an acceptable family of distributions. Then there exists string i s. t. $\mu_i(y) > 0$ and*

$$\sum_{s:\mu_s(y)>0} \mathrm{m}^{t(n)}(s)\mu_s(x) \leq \mathrm{m}^{r(n)}(i)\mu_i(x)2^{O(\log n)}.$$

Proof (of Theorem 4(b) from Lemma 1). Let string y belongs to $\mathcal{M}$-probabilistic prediction d, t-neighborhood of x, i.e. $P^t_{x,\mathcal{M}}(y) \geq 2^{-d}$. Let us estimate $P^t_{x,\mathcal{M}}(y)$. First note that the denominator of (4) is less than $\mathrm{m}^{t(n)}(x)$. Indeed, by the last property of acceptability there exists $\mu_s \in \mathcal{M}$ such that $\mu_s(x) = 1$ and $\mathrm{m}^{O(t(n))}(s) = \mathrm{m}^{t(n)}(x) + O(1)$. The numerator of (4) can be estimated by Lemma 1 as $\mathrm{m}^{r(n)}(i)\mu_i(x)2^{O(\log n)}$ for some string i and polynomial r. So, $\log P^t_{x,\mathcal{M}}(y)$ is less than

$$\mathrm{K}^{r(n)}(i) - \log \mu_i(x) - \mathrm{K}^{t(n)}(x) + O(\log n).$$

This value is not smaller than d. Hence, y belongs to $\mathcal{M}$-algorithmic prediction $d + O(\log n), r(n), t(n)$-neighborhood of x.

Lemma 2. *Let $\mathcal{H}$ be a set of functions from $\{0,1\}^l$ to $\{0,1\}^m$ with the following properties.*

- *For every l at least $\frac{3}{4}$ of all functions from $\{0,1\}^l$ to $\{0,1\}^n$ are in $\mathcal{H}$.*
- *For some k there is a Σ^p_k machine with oracle access to a function H on input 1^l will accept exactly when H is in $\mathcal{H}$.*

Then under Assumption 1 there is a polynomial-time computable function $H'(x,r)$ with $x \in \{0,1\}^l$ and $|r| = O(\log l)$ such that for at least $\frac{2}{3}$ of the possible r, $H_r(x) = H'(x,r)$ is in $\mathcal{H}$.

Proof (of Lemma 1 from Lemma 2).

The sum $\sum_{s:\mathrm{m}_s(y)>0} \mathrm{m}^{t(n)}(s)\mu_s(x)$ is equal to the sum over all k and j of sums

$$\sum_{\substack{s:\mu_s(y)>0 \\ \mathrm{m}^{t(n)}(s)=2^{-k} \\ \mu_s(x)=2^{-j}}} \mathrm{m}^{t(n)}(s)\mu_s(x). \tag{5}$$

In fact only poly(n) of such sums are positive. Indeed, from acceptability of $\mathcal{M}$ it is follows that $\mathrm{K}^{\mathrm{poly}(n)}(\mu_s)$ is bounded by poly(n). Also, if $\mu_s(x) < 2^{-\mathrm{poly}(n)}$ then $\mu_s(x) = 0$ since μ_s is sampled by a polynomial time-bounded program. Hence it is enough to show that for every j and k there exists i and polynomial r such that $\mathrm{m}^{r(n)}(i)\mu_i(x)2^{O(\log n)}$ is greater than (5) and $\mu_i(y) > 0$.

Denote by $U^{t(n)}$ a function that works as a universal Turing machine U but if U does not outputs anything in $t(n)$ steps then it outputs empty string. Denote by w the logarithm of the number of terms in the sum (5). Denote by $\mathcal{H}$ the set of all functions h from $\{0,1\}^{k-w+8\log n}$ to $\{0,1\}^k$ with the following property:

If for a pair of strings (x', y') of length n there exist at least 2^w strings s such that $\mu_s(y') > 0$, $\mathrm{m}^{t(n)}(s) = -k$ and $\mu_s(x') = 2^{-j}$ then one of this string is in the image of $U(h)$.

Lemma 3. *At least* $\frac{3}{4}$ *of all possible functions from* $\{0,1\}^{k-w+8\log n}$ *to* $\{0,1\}^k$ *belongs to* $\mathcal{H}$.

Using Lemma 3 we prove the existence of i such that $\mathrm{m}(i)\mu_i(x)2^{O(\log n)}$ is greater than (5). (Note that here m is not polynomial-time bounded, so this is not really what we want.) By Lemma 3 *there exists* a function that belongs to $\mathcal{H}$. The lexicographically first such function has small complexity, because it can be computed given j, k, n and w. Since $h \in \mathcal{H}$ there exists $i' \in \{0,1\}^{k-w+8\log n}$ such that $\mu_i(x) = 2^{-j}$ and $\mu_i(y) > 0$ where $i = U(h(i'))$. Since $i' \in \{0,1\}^{k-w+8\log n}$ the complexity of i is not greater than $\mathrm{K}(i) \leq k - w + O(\log n)$. A simple calculation shows that $\mathrm{m}(i)\mu_i(x)2^{O\log n}$ is greater than (5).

To prove the existence of such string i which *polynomial-time bounded* complexity is less than $k - w + O(\log n)$ we need a simple and polynomial-time computable function in $\mathcal{H}$. To find it we use Lemma 4 for $l = k - w + 8\log n$ and $m = k$. We claim that family $\mathcal{H}$ satisfies properties of Lemma 4. For the first property it is true by Lemma 3. For the second property note that the property $h \in \mathcal{H}$ can written as

$$\forall(x', y')\exists 2^w s : ((\mu_s(x') = 2^{-j}) \wedge (\mu_s(y') > 0)) \Rightarrow \exists i : ((\mu_{U(h(i))}(x') = 2^{-j}) \wedge (\mu_{U(h(i))}(y') > 0)).$$

This property belongs to $\mathrm{P}^{\Sigma_p^l}$ for some l since the approximation of the number of certificates belongs to Σ_p^2 [12]. So, there exists polynomial-time computable $H'(x, r)$ such that for some fixed r function $H_r(x) = H'(x, r)$ is in $\mathcal{H}$. Since $|r| = O(\log n)$ we conclude that there exists simple and polynomial time computable function in $\mathcal{H}$ that complete the proof.

Acknowledgments. This work is supported in parts by the RFBR grant 16-01-00362, by the Young Russian Mathematics award, MK-5379.2018.1 and the RaCAF ANR-15-CE40-0016-01 grant. The study has also been funded by the Russian Academic Excellence Project '5-100'.

References

1. Antunes, L., Fortnow, L.: Worst-case running times for average-case algorithms. In: Proceedings of the 24th IEEE Conference on Computational Complexity, pp. 298–303 (2009)
2. Antunes, L., Fortnow, L., Vinodchandran, N.V.: Using depth to capture average-case complexity. In: Lingas, A., Nilsson, B.J. (eds.) FCT 2003. LNCS, vol. 2751, pp. 303–310. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45077-1_28
3. Koppel, M.: Complexity, depth and sophistication. Complex Syst. **1**, 1087–1091 (1987)
4. Kolmogorov, A.N.: Talk at the Information Theory Symposium in Tallinn, Estonia (then USSR) (1974)
5. Li, M., Vitányi, P.: An Introduction to Kolmogorov Complexity and Its Applications. TCS, vol. 3. Springer, New York (2008). https://doi.org/10.1007/978-0-387-49820-1
6. Milovanov, A.: Algorithmic statistic, prediction and machine learning. In: Proceedings of 33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016). Leibnitz International Proceedings in Informatics (LIPIcs), vol. 47, pp. 54:1–54:13 (2016)
7. Milovanov, A.: On algorithmic statistics for space-bounded algorithms. In: Weil, P. (ed.) CSR 2017. LNCS, vol. 10304, pp. 232–244. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58747-9_21
8. Milovanov A., Vereshchagin N.: Stochasticity in algorithmic statistics for polynomial time. In: 32nd Computational Complexity Conference (CCC 2017). Leibniz International Proceedings in Informatics (LIPIcs), vol. 79, pp. 17:1–17:18 (2017)
9. Shen, A.: Around kolmogorov complexity: basic notions and results. In: Vovk, V., Papadopoulos, H., Gammerman, A. (eds.) Measures of Complexity: Festschrift for Alexey Chervonenkis, pp. 75–115. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21852-6_7. ISBN: 978-3-319-21851-9
10. Shen, A.: The concept of (α, β)-stochasticity in the Kolmogorov sense, and its properties. Sov. Math. Dokl. **271**(1), 295–299 (1983)
11. Shen, A., Uspensky, V., Vereshchagin, N.: Kolmogorov Complexity and Algorithmic Randomness. ACM, New York (2017)
12. Stockmeyer, L.: On approximation algorithms for #P. SIAM J. Comput. **14**(4), 849–861 (1985)
13. Vereshchagin, N., Shen, A.: Algorithmic statistics: forty years later. In: Day, A., Fellows, M., Greenberg, N., Khoussainov, B., Melnikov, A., Rosamond, F. (eds.) Computability and Complexity. LNCS, vol. 10010, pp. 669–737. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-50062-1_41
14. Vereshchagin, N., Vitányi, P.M.B.: Kolmogorov's structure functions with an application to the foundations of model selection. IEEE Trans. Inf. Theory **50**(12), 3265–3290 (2004). Preliminary Version: Proceedings of 47th IEEE Symposium on the Foundations of Computer Science, pp. 751–760 (2002)
15. Vereshchagin, N., Vitányi, P.M.B.: Rate distortion and denoising of individual data using Kolmogorov complexity. IEEE Trans. Inf. Theory **56**(7), 3438–3454 (2010)