LATVIJAS UNIVERSITĀTE
EKSAKTO ZINĀTŅU UN TEHNOLOĢIJU FAKULTĀTE
MATEMĀTIKAS NODAĻA

GADĪJUMA MEŽA VISPĀRINĀŠANAS KĻŪDAS AUGŠĒJO ROBEŽU ANALĪZE

REFERĀTS

Autors: Kirills Bobkovs
Studenta apliecības nr.: kb25028

RĪGA 2025

SATURA RĀDĪTĀJS

1. IEVADS

Gadījuma meža (angl. *Random Forest*) klasifikācijas algoritmi ir plaši izmantoti gan datorzinātnē, gan statistikā, pateicoties to augstajai precizitātei un spējai samazināt pārmācīšanās risku, salīdzinot ar atsevišķiem lēmumu kokiem. Tomēr, lai pilnībā izprastu šo metožu darbību un to kļūdas uzvedību, nepieciešams analizēt un formulēt teorētisko pamatu vispārināšanas kļūdas (angl. *generalization error*) augšējām robežām.

Šajā darbā galvenā uzmanība tiek pievērsta gadījuma meža vispārināšanas kļūdas un tās augšējās robežas izpētei, īpašu uzmanību pievēršot diviem kritiskiem parametriem: individuālo klasifikatoru precizitātei (angl. *strength*) un to savstarpējai korelācijai. Šie parametri tiek definēti, izmantojot jaunus jēdzienus, piemēram, *marginal function* un *raw margin function*, kas ļauj kvantitatīvi novērtēt meža klasifikatoru kvalitāti un to savstarpējo atkarību.

Darba mērķis ir iepazīties ar teorētiskajām pamatformulām un praktiski ilustrēt vispārināšanas kļūdas augšējās robežas gadījuma mežos. Mērķa sasniegšanai tika izvirzīti šādi darba uzdevumi:

1. analizēt un izpētīt svarīgākos jēdzienus vispārināšanas kļūdas un tās augšējās robežas gadījuma mežos;

2. izpētīt, vai modelis pārmācās, palielinoties koku skaitam mežā;

3. izvest vispārināšanas kļūdas augšējo robežu, izmantojot divus parametrus: individuālo klasifikatoru precizitāti un to savstarpējo korelāciju.

Darba struktūra sastāv no piecām nodaļām: Ievads, Gadījuma mežs un tā precizitāte, Praktiskā daļa un rezultāti, Secinājumi un Pielikums (programmas kods).

Šajā darbā izmantotās teorētiskās formulas un izklāsti balstās uz literatūru par gadījuma mežiem ([2]). Simulācijas rezultāti ir ģenerēti ar šī darba autora *R* kodu (skat. Pielikums A).

2. GADĪJUMA MEŽS UN TĀ PRECIZITĀTE

2.1. Definīcija. Gadījuma mežs ir klasifikators, kas sastāv no koka strukturētiem klasifikatoriem $\{h(\boldsymbol{x},\Theta_k),\ k\ =\ 1,\dots\ K\}$, kur $\{\Theta_k\}$ ir neatkarīgi un vienādi sadalīti (iid) gadījuma lielumu vektori, un katrs koks piešķir vienu balsi vispopulārākajai klasei pie dotās ievades $\boldsymbol{x}$.

Tā kā katrs koks tiek izveidots, izmantojot mācīšanās datu kopu un gadījuma vektoru Θ_k, tad kopu $\{\Theta_k\}$ var interpretēt kā vienu no veidiem, kā koka konstruēšanas procesā ieviest jeb *injicēt* (angl. *inject*) nejaušību.

Piemēram, ja tiek izmantota tā sauktā iepakošana (angl. *bagging*), tad gadījuma lieluma vektors Θ tiek ģenerēts kā skaitu sadalījums N kastēs, kas rodas, nejauši iemetot N šautriņas šajās kastēs, kur N ir piemēru skaits mācīšanās datu kopā. Citiem vārdiem sakot, tiek izmantota bootstrap metode jeb izlase ar atkārtošanu (*sampling with replacement*), kas katram kokam piešķir indeksu kopu $\{i_1, i_2, \dots, i_N\}$, kur N ir datu punktu skaits mācīšanās kopā. Šī indeksu kopa nosaka, kuri datu punkti no mācīšanās datu kopas tiek izmantoti konkrētā koka apmācībai.

Savukārt gadījumā, kad tiek izmantota nejauša sadalījumu izvēle (*random split selection*), vektors Θ sastāv no neatkarīgiem gadījuma veseliem skaitļiem intervālā no 1 līdz M, kur M ir mainīgo skaits datu punktam, kas nosaka, kuri prediktori jeb mainīgie tiek apsvērti katrā sadalījuma solī *CART* algoritmā.

Tādējādi var secināt, ka vektora Θ daba un dimensija ir tieši atkarīga no tā, kādā veidā nejaušība tiek izmantota koka konstruēšanas procesā.

Vairāk par CART algoritmu skatīt šajā avotā: [1].

2.2. Definīcija. Lai $h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \dots, h_K(\boldsymbol{x})$ būtu klasifikatoru kopa, un mācīšanas izlase būtu paņemta no gadījuma lielumu vektora Y,$\boldsymbol{X}$, tad marginālo funkciju definē šādi:

$$\mathrm{mg}(\mathbf{X}, Y) = v_Y(\mathbf{X}) - \max_{j \neq Y} v_j(\mathbf{X}),$$

kur $v_j(\mathbf{X}) = \frac{1}{K}\sum_{k=1}^{K} \mathbf{I}(h_k(\mathbf{X}) = j)$ apzīmē koku balsu proporciju klasei j pie ievades $\boldsymbol{X}$

Margināla funkcija parāda, cik lielā mērā vidējais balsu skaits, pie ievades X un klases Y, par pareizo klasi pārsniedz vidējo balsu skaitu par jebkuru citu klasi. Jo lielāka marža, jo lielāka pārliecība par klasifikāciju.

2.3. Definīcija. Par $PE^* = P_{\boldsymbol{X},Y}(mg(\boldsymbol{X},Y) < 0)$ tiek saukta vispārināšanas kļūda.

Mašīnmācīšanās kontekstā vispārināšanas kļūda mēra, cik precīzs ir modelis uz mācīšanās laikā neredzētajiem datiem.

2.1. Teorēma. *Palielinoties koku skaitam, t.i.* $K \to \infty$, PE^* *visām secībām* $\Theta_1, \Theta_2, \ldots$ *gandrīz droši (angl. almost surely) konverģē uz*

$$P_{\boldsymbol{X},Y}\left(P_\Theta\big(h(\boldsymbol{X},\Theta) = Y\big) - \max_{j \neq Y} P_\Theta\big(h(X,\Theta) = j\big) < 0 \right).$$

Pierādījums. Vispirms vēlreiz aprakstīsim vajadzīgo.

Katrs koks tiek ģenerēts ar nejaušu "recepti" Θ. Šī Θ ietver visu nejaušību. Tātad $\Theta_1, \Theta_2, \ldots$ ir secība no N neatkarīgiem koku ģenerēšanas mēģinājumiem, un $h(\Theta_n, x)$ ir n-tā koka prognoze punktam x.

Indikatoru funkcija

$$I(h(\Theta_n, x) = j)$$

ir 1, ja n-tais koks prognozē klasi j, un 0 citādi. Šie indikatori ir neatkarīgi un vienādi sadalīti (iid.), jo Θ_n ir iid.

Katrs koks Θ_n dala x telpu uz lapu mezgliem (angl. *leaf nodes*), kas ir hipertaisnstūri (angl. *hyperrectangles*). Tādēļ jebkurai klasei j, kopa $S_n := \{x : h(\Theta_n, x) = j\}$ ir hipertaisnstūra reģionu apvienojums.

Tā kā jebkuram kokam ir galīgs skaits lapu mezglu, eksistē tikai galīgs dažādu hipertaisnstūra reģionu apvienojumu skaits, pie visām iespējamām Θ. Apzīmē $S_1, S_2, \ldots, S_K$, kur $K \leq N$ un N ir koku skaits.

Tiek definēta funkcija

$$\phi(\Theta) = k, \quad \text{ja } \{x : h(\Theta, x) = j\} = S_k,$$

kas katram kokam Θ_n piešķir kādu kopu S_k klasei j.

Tālāk tiek skaitīts cik reizes katrs S_k parādās pirmajos N kokās. $N_k :=$ koku skaits no pirmajiem N, kuriem $\phi(\Theta_n) = k$, jeb

$$N_k = \sum_{n=1}^{N} I(\phi(\Theta_n) = k)$$

Tad

$$\frac{1}{N} \sum_{n=1}^{N} I(h(\Theta_n, x) = j) = \frac{1}{N} \sum_{k=1}^{K} N_k I(x \in S_k)$$

Tas ļauj izmantot stipro lielo skaitļu likumu uz N_k/N.

Kad $N \to \infty$, $\frac{N_k}{N} \to P(\phi(\Theta) = k)$ gandrīz droši.

Tad jebkuram x:

$$\frac{1}{N}\sum_{n=1}^{N} I(h(\Theta_n, x) = j) = \sum_{k=1}^{K} \frac{N_k}{N} I(x \in S_k) \xrightarrow{\text{g.d.}} \sum_{k=1}^{K} P(\phi(\Theta) = k) I(x \in S_k) = P_\Theta(h(\Theta, x) = j).$$

Tā kā, palielinoties koku skaitam $K \to \infty$, katra koka balsu proporcija

$$v_j(\mathbf{X}) = \frac{1}{K}\sum_{k=1}^{K} \mathbf{I}(h_k(\mathbf{X}) = j)$$

gandrīz droši konverģē uz atbilstošo varbūtību

$$v_j(\mathbf{X}) \xrightarrow{\text{g.d.}} P_\Theta(h(\mathbf{X},\Theta) = j),$$

tad arī marginālā funkcija gandrīz droši konverģē:

$$\text{mg}(\mathbf{X},Y) \xrightarrow{\text{g.d.}} P_\Theta(h(\mathbf{X},\Theta) = Y) - \max_{j \neq Y} P_\Theta(h(\mathbf{X},\Theta) = j),$$

un līdz ar to vispārināšanas kļūda PE^* gandrīz droši konverģē uz

$$PE^* \xrightarrow{\text{g.d.}} P_{\mathbf{X},Y}\Big(P_\Theta(h(\mathbf{X},\Theta) = Y) - \max_{j \neq Y} P_\Theta(h(\mathbf{X},\Theta) = j) < 0\Big).$$

□

Ņemot vērā, ka, palielinoties koku skaitam, atsevišķu koku balsu proporcijas gandrīz droši konverģē uz atbilstošajām varbūtībām, tika ieviesta margināla funkcija, kas raksturo gadījuma meža asimptotisko (bezgalīga koku skaita) klasifikācijas uzvedību.

2.4. Definīcija. Gadījuma meža marginālā funkcija tiek definēta kā

$$\text{mr}(\boldsymbol{X},Y) = P_\Theta\big(h(\boldsymbol{X},\Theta) = Y\big) - \max_{j \neq Y} P_\Theta\big(h(\boldsymbol{X},\Theta) = j\big).$$

2.5. Definīcija. Klasifikatoru kopas $\{h(\boldsymbol{x},\Theta)\}$ stiprums tiek definēts kā

$$s = \mathbb{E}_{\boldsymbol{X},Y}\big[\text{mr}(X,Y)\big].$$

2.1. Apgalvojums. $PE^* \leq \text{var}(mr)/s^2$

Pierādījums. Pieņemsim, ka gadījuma meža stiprums ir $s \geq 0$, kur

$$s = \mathbb{E}_{\boldsymbol{X},Y}\big[\text{mr}(\boldsymbol{X},Y)\big].$$

Atcerēsimies, ka vispārināšanas kļūda ir definēta kā

$$PE^* = P_{\boldsymbol{X},Y}\big(\text{mr}(\boldsymbol{X},Y) < 0\big).$$

Ir spēkā: $\{\mathrm{mr}(\boldsymbol{X},Y) < 0\} \subseteq \big\{\, |\mathrm{mr}(X,Y) - s| \geq s \,\big\}$.

Pārejot pie varbūtībām, seko

$$PE^* = P_{\boldsymbol{X},Y}\big(\mathrm{mr}(X,Y) < 0\big) \leq P_{\boldsymbol{X},Y}\big(|\mathrm{mr}(\boldsymbol{X},Y) - s| \geq s\big).$$

Pielietojot Čebiševa nevienādību, iegūstam

$$PE^* \leq P_{\boldsymbol{X},Y}\big(|\mathrm{mr}(\boldsymbol{X},Y) - s| \geq s\big) \leq \frac{\mathrm{Var}\big(\mathrm{mr}(\boldsymbol{X},Y)\big)}{s^2}. \tag{2.1}$$

□

Marģinālās funkcijas dispersijas izteiksmi var uzrakstīt atklātākā formā, bet vispirms jādefinē papildu jēdzieni.

Ar $\hat{\jmath}$ definē:

$$\hat{\jmath}(\boldsymbol{X},Y) := \arg\max_{j \neq Y} P_\Theta\big(h(X,\Theta) = j\big).$$

Tad

$$\begin{aligned}\mathrm{mr}(\boldsymbol{X},Y) &= P_\Theta\big(h(\boldsymbol{X},\Theta) = Y\big) - P_\Theta\big(h(\boldsymbol{X},\Theta) = \hat{\jmath}(\boldsymbol{X},Y)\big) \\ &= \mathbb{E}_\Theta\big[\mathbf{I}\big(h(\boldsymbol{X},\Theta) = Y\big) - \mathbf{I}\big(h(\boldsymbol{X},\Theta) = \hat{\jmath}(X,Y)\big)\big]\,.\end{aligned}$$

2.6. Definīcija. Neapstrādātā (angl. *raw*) marģinālā funkcija tiek definēta kā

$$\mathrm{rmg}(\Theta,\boldsymbol{X},Y) = \mathbf{I}\big(h(\boldsymbol{X},\Theta) = Y\big) - \mathbf{I}\big(h(\boldsymbol{X},\Theta) = \hat{\jmath}(\boldsymbol{X},Y)\big).$$

Tātad $\mathrm{mr}(\boldsymbol{X},Y) = \mathbb{E}_\Theta\big[\mathrm{rmg}(\Theta,X,Y)\big]$ ir neapstrādātās marginālās funkcijas $\mathrm{rmg}(\Theta,X,Y)$ matemātiskā cerība attiecībā pret Θ.

2.2. Teorēma. *Augšējā robeža vispārināšanas kļūdai ir*

$$PE^* \leq \frac{\bar{\rho}(1 - s^2)}{s^2}.$$

Pierādījums. Vispārīgai funkcijai f ir identitāte

$$\big(\mathbb{E}_\Theta f(\Theta)\big)^2 = \mathbb{E}_{\Theta,\Theta'}\big[f(\Theta)f(\Theta')\big], \tag{2.2}$$

kur Θ un Θ' ir i.i.d., jo

$$\mathbb{E}_{\Theta,\Theta'}[f(\Theta)f(\Theta')] = \iint f(\theta)f(\theta')\,dP_\Theta(\theta)\,dP_\Theta(\theta') = \Big(\int f(\theta)\,dP_\Theta(\theta)\Big)^2 = \big(\mathbb{E}_\Theta[f(\Theta)]\big)^2.$$

Tātad identitāte turas tieši no Θ un Θ' neatkarības.

Kā iepriekš jau bija pāradīts, $\mathrm{mr}(\boldsymbol{X},Y)$ var uzrakstīt kā:

$$\mathrm{mr}(\boldsymbol{X},Y) := \mathbb{E}_\Theta\big[\mathrm{rmg}(\Theta,X,Y)\big]. \tag{2.3}$$

Lai aprēķinātu marginālās funkcijas dispersiju attiecībā pret (X,Y), izmantojam dispersijas definīciju:

$$\begin{aligned}\operatorname{var}(\mathrm{mr}) &:= \operatorname{var}_{\boldsymbol{X},Y}\big(\mathrm{mr}(\boldsymbol{X},Y)\big)\\ &= \mathbb{E}_{\boldsymbol{X},Y}\big[\mathrm{mr}(\boldsymbol{X},Y)^2\big] - \Big(\mathbb{E}_{\boldsymbol{X},Y}\mathrm{mr}(\boldsymbol{X},Y)\Big)^2. \end{aligned} \tag{2.4}$$

Sākam ar pirmo locekli no 2.4. Izmantojot 2.3, iegūstam:

$$\mathrm{mr}(\boldsymbol{X},Y)^2 = \mathbb{E}_{\Theta,\Theta'}[\mathrm{rmg}(\Theta,\boldsymbol{X},Y)\,\mathrm{rmg}(\Theta',\boldsymbol{X},Y)]\,,$$

kur Θ' un Θ ir i.i.d. ((X,Y) šeit ir fiksēti).

Pielietojot Fubini teorēmu, samainām matemātisko cerību secību:

$$\mathbb{E}_{\boldsymbol{X},Y}[\mathbb{E}_{\Theta,\Theta'}[\cdot]] = \mathbb{E}_{\Theta,\Theta'}[\mathbb{E}_{\boldsymbol{X},Y}[\cdot]]\,.$$

Tādējādi

$$\mathbb{E}_{\boldsymbol{X},Y}\big[\mathrm{mr}(\boldsymbol{X},Y)^2\big] = \mathbb{E}_{\Theta,\Theta'}\big[\mathbb{E}_{\boldsymbol{X},Y}\big[\mathrm{rmg}(\Theta,\boldsymbol{X},Y)\,\mathrm{rmg}(\Theta',\boldsymbol{X},Y)\big]\big]\,. \tag{2.5}$$

Tagad aplūkojam otro locekli no (2.4).

$$\begin{aligned}\mathbb{E}_{\boldsymbol{X},Y}\mathrm{mr}(\boldsymbol{X},Y) &= \mathbb{E}_{\boldsymbol{X},Y}\mathbb{E}_{\Theta}\big[\mathrm{rmg}(\Theta,\boldsymbol{X},Y)\big]\\ &= \mathbb{E}_{\Theta}\mathbb{E}_{\boldsymbol{X},Y}\big[\mathrm{rmg}(\Theta,\boldsymbol{X},Y)\big],\end{aligned}$$

kur pēdējā vienādība atkal izriet no neatkarības un Fubini teorēmas.

Pacelot kvadrātā un ieviešot neatkarīgu kopiju Θ' (pielietojot 2.2), iegūstam

$$\Big(\mathbb{E}_{\boldsymbol{X},Y}\mathrm{mr}(\boldsymbol{X},Y)\Big)^2 = \mathbb{E}_{\Theta,\Theta'}[\mathbb{E}_{\boldsymbol{X},Y}\mathrm{rmg}(\Theta,X,Y)\;\mathbb{E}_{\boldsymbol{X},Y}\mathrm{rmg}(\Theta',X,Y)]\,. \tag{2.6}$$

Ievietojot izteiksmes (2.5) un (2.6) dispersijas formulā (2.4), iegūstam

$$\begin{aligned}\operatorname{var}(\mathrm{mr}) = \mathbb{E}_{\Theta,\Theta'}\Big[&\mathbb{E}_{X,Y}\big[\mathrm{rmg}(\Theta,\boldsymbol{X},Y)\,\mathrm{rmg}(\Theta',\boldsymbol{X},Y)\big]\\ &- \mathbb{E}_{\boldsymbol{X},Y}\mathrm{rmg}(\Theta,\boldsymbol{X},Y)\;\mathbb{E}_{\boldsymbol{X},Y}\mathrm{rmg}(\Theta',\boldsymbol{X},Y)\Big].\end{aligned}$$

Secinām, ka

$$\operatorname{var}(\mathrm{mr}) = \mathbb{E}_{\Theta,\Theta'}\big[\operatorname{cov}_{\boldsymbol{X},Y}\big(\mathrm{rmg}(\Theta,\boldsymbol{X},Y),\mathrm{rmg}(\Theta',\boldsymbol{X},Y)\big)\big]\,. \tag{2.7}$$

Izmantojot šo izteiksmi, iegūstam

$$\operatorname{var}(\mathrm{mr}) = \mathbb{E}_{\Theta,\Theta'}\Big[\operatorname{cov}_{X,Y}\big(\mathrm{rmg}(\Theta,\boldsymbol{X},Y),\mathrm{rmg}(\Theta',\boldsymbol{X},Y)\big)\Big] \tag{2.8}$$

$$= \mathbb{E}_{\Theta,\Theta'}\big[\rho(\Theta,\Theta')\,\mathrm{sd}(\Theta)\,\mathrm{sd}(\Theta')\big], \tag{2.9}$$

kur $\rho(\Theta,\Theta')$ ir neapstrādātās marģinālās funkcijas $\operatorname{rmg}(\Theta,\boldsymbol{X},Y)$ un $\operatorname{rmg}(\Theta',\boldsymbol{X},Y)$ korelācija, kad Θ,Θ' tiek turēti fiksēti, un $\operatorname{sd}(\Theta)$ ir $\operatorname{rmg}(\Theta,\boldsymbol{X},Y)$ standarta novirze, turot Θ fiksētu.

Tad iegūstam

$$\operatorname{var}(\mathrm{mr}) = \frac{\mathbb{E}_{\Theta,\Theta'}\big[\rho(\Theta,\Theta')\operatorname{sd}(\Theta)\operatorname{sd}(\Theta')\big]\,\mathbb{E}_{\Theta,\Theta'}\big[\operatorname{sd}(\Theta)\operatorname{sd}(\Theta')\big]}{\mathbb{E}_{\Theta,\Theta'}\big[\operatorname{sd}(\Theta)\operatorname{sd}(\Theta')\big]} = \bar{\rho}\big(\mathbb{E}_\Theta \operatorname{sd}(\Theta)\big)^2 \le \bar{\rho}\,\mathbb{E}_\Theta \operatorname{var}(\Theta), \tag{2.10}$$

kur $\bar{\rho}$ ir vidējā korelācija, t.i.,

$$\bar{\rho} := \frac{\mathbb{E}_{\Theta,\Theta'}\big[\rho(\Theta,\Theta')\operatorname{sd}(\Theta)\operatorname{sd}(\Theta')\big]}{\mathbb{E}_{\Theta,\Theta'}\big[\operatorname{sd}(\Theta)\operatorname{sd}(\Theta')\big]}.$$

Tā kā $\operatorname{var}(\Theta) = \operatorname{var}_{X,Y}\big(\operatorname{rmg}(\Theta,\boldsymbol{X},Y)\big) = \mathbb{E}_{\boldsymbol{X},Y}[\operatorname{rmg}(\Theta,\boldsymbol{X},Y)^2] - \big(\mathbb{E}_{\boldsymbol{X},Y}[\operatorname{rmg}(\Theta,\boldsymbol{X},Y)]\big)^2$,

un $\operatorname{rmg}(\Theta,\boldsymbol{X},Y)$ ir ierobežota, t.i., $\operatorname{rmg}(\Theta,X,Y)^2 \le 1$, tad iegūstam

$$\mathbb{E}_{\boldsymbol{X},Y}[\operatorname{rmg}(\Theta,\boldsymbol{X},Y)^2] \le 1.$$

Tādējādi $\operatorname{var}(\Theta) \le 1 - \big(\mathbb{E}_{\boldsymbol{X},Y}[\operatorname{rmg}(\Theta,\boldsymbol{X},Y)]\big)^2$.

Ņemot cerību pēc Θ:

$$\mathbb{E}_\Theta \operatorname{var}(\Theta) \le \mathbb{E}_\Theta\Big[1 - \big(\mathbb{E}_{\boldsymbol{X},Y}[\operatorname{rmg}(\Theta,\boldsymbol{X},Y)]\big)^2\Big] = 1 - \mathbb{E}_\Theta\Big(\mathbb{E}_{\boldsymbol{X},Y}[\operatorname{rmg}(\Theta,\boldsymbol{X},Y)]\Big)^2. \tag{2.11}$$

Izmantojot Jensena nevienādību (Jensen's inequality) izliektai funkcijai x^2, iegūstam

$$\mathbb{E}_\Theta\Big(\mathbb{E}_{\boldsymbol{X},Y}[\operatorname{rmg}(\Theta,\boldsymbol{X},Y)]\Big)^2 \ge \Big(\mathbb{E}_\Theta\mathbb{E}_{X,Y}[\operatorname{rmg}(\Theta,\boldsymbol{X},Y)]\Big)^2 = s^2, \tag{2.12}$$

Apvienojot 2.11 un 2.12, iegūstam galīgo nevienādību:

$$\mathbb{E}_\Theta \operatorname{var}(\Theta) \le 1 - s^2. \tag{2.13}$$

Apvienojot 2.1, 2.10 un 2.13, iegūstam augšējo robežu vispārināšanas kļūdai:

$$PE^* \le \frac{\bar{\rho}(1-s^2)}{s^2}. \tag{2.14}$$

□

2.1. Piezīme. *Nevienādība 2.10*

$$\big(\mathbb{E}_\Theta \operatorname{sd}(\Theta)\big)^2 \le \mathbb{E}_\Theta \operatorname{var}(\Theta)$$

tiek iegūta, izmantojot Koši–Švarca nevienādību:

$$\big(\mathbb{E}A\big)^2 \le \mathbb{E}\big(A^2\big),$$

kur $A = \operatorname{sd}(\Theta) = \sqrt{\operatorname{var}(\Theta)}$.

3. PRAKTISKĀ DAĻA UN REZULTĀTI

Šajā sadaļā tiek analizēta gadījuma meža vispārināšanas kļūdas augšējās robežas uzvedība, īpaši tās stabilizēšana, palielinoties koku skaitam. Iegūta robeža 2.14 palīdz izprast divus galvenos faktorus, kas ietekmē nejaušā meža kļūdu: *atsevišķo koku stiprumu* un *to savstarpējo korelāciju* neapstrādātās marģinālās funkcijas izteiksmē. Attiecība $\bar{\rho}/s^2$ (vidējā korelācija dalīta ar stipruma kvadrātu) darbojas kā indikators — jo mazāka šī attiecība, jo labāka meža vispārināšanas spēja.

Lai ilustrētu iegūto teorētisko rezultātu, tika veikta simulācija R vidē. Tika izmantota klasiskā *Iris* [3] datu kopa, kas sastāv no četrām pazīmēm un trim klašu kategorijām. Visas teorētiskajās formulās sastopamās matemātiskās cerības tika aproksimētas ar empīrisko vidējo vērtību, izmantojot galīgu datu punktu un gadījuma koku realizāciju skaitu.

Simulācijā tika ģenerēti gadījuma meži ar dažādu koku skaitu, un katram koku skaitam tika veiktas vairākas inicializācijas, izmantojot atšķirīgas `set.seed` vērtības, lai novērtētu vispārināšanas kļūdas augšējās robežas stabilitāti attiecībā pret gadījuma meža konstrukciju.

Simulācijas rezultātā iegūtie grafiki parāda, kā pārsvarā augšējā robeža stabilizējas ar palielinātu koku skaitu. Tomēr nevar teikt, ka tas izpildās vienmēr, jo daži grafiki nedod drošu iespaidu par augšējās robežas konverģenci. Svarīgi atzīmēt, ka teorētiskā konverģence uz kādu vērtību ir pierādīta tikai PE^* vērtībai, nevis tās augšējām robežām. Tāpēc mūsu brīva (angl. *loose*) augšējā robeža var nerādīt PE^* īsto konverģences dinamiku. Rezultāti ir attēloti 1, 2, 3,4.

Papildus pārbaude teorēmai (2.1) tika veikta, simulējot gadījuma meža veiktspēju sirds slimību klasifikācijas uzdevumā ([4]), izmantojot *Python*. Datu kopā ietilpst 14 klīniskie un demogrāfiskie prediktori, kas raksturo pacientus, piemēram, vecumu, dzimumu, krūšu sāpju tipu, asinsspiedienu, holesterīna līmeni, elektrokardiogrammas rādītājus un maksimālo sirdsdarbības frekvenci, kā arī citi sirds veselību raksturojoši parametri. Mērķis bija prognozēt mainīgo `num`, kas norāda sirds slimības esamību vai neesamību.

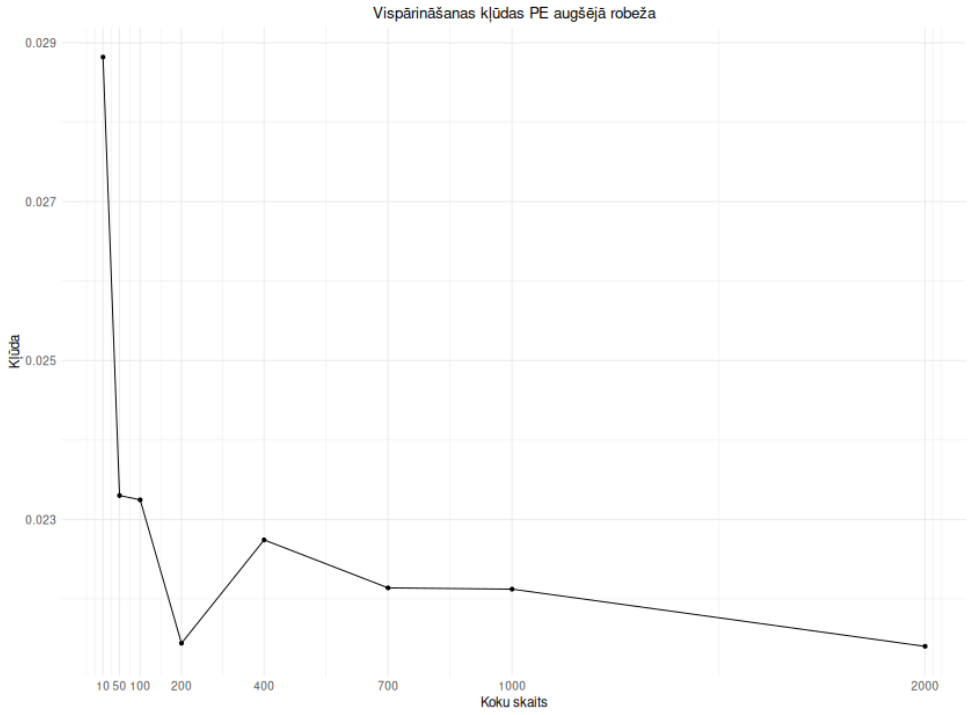
Simulācijā tika ģenerēti gadījuma meži ar dažādu koku skaitu, saglabājot citus hiperparametrus nemainīgus. Katram koku skaita parametram tika veiktas 20 inicializācijas ar dažādiem `seed` iestatījumiem, aprēķinot AUC vidējo vērtību un standartnovirzi. Rezultāti, parādīti tabulā 1, empiriski ilustrē, ka AUC vērtības pieaug, palielinot koku skaitu, līdz sasniedz noteiktu K vērtību, pēc kuras novērojams plateau efekts — modelis vairs neuzlabojas, bet aprēķina laiks pieaug.

Koku skaits	AUC vidējā vērtība	std(AUC)	Vidējais aprēķina laiks, s
1	0.749633	0.029051	0.054637
5	0.869783	0.013673	0.098653
10	0.890504	0.009983	0.139046
25	0.903147	0.004656	0.179914
50	0.907483	0.004686	0.324881
100	0.910726	0.003512	0.456434
200	0.913392	0.002666	1.007332
500	0.914867	0.001961	2.541950
1000	0.914772	0.001178	4.329879
1500	0.915053	0.000732	6.402032

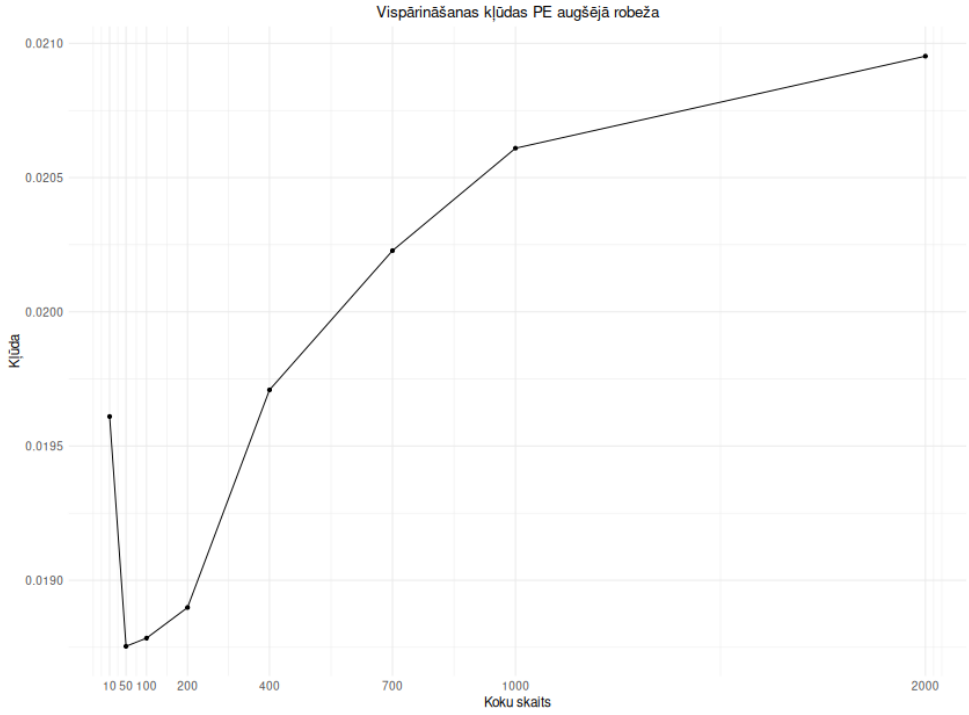
1. tabula

Gadījuma meža AUC veiktspēja un aprēķina laiks dažādam koku skaitam (atkārtojumu skaits: 20).

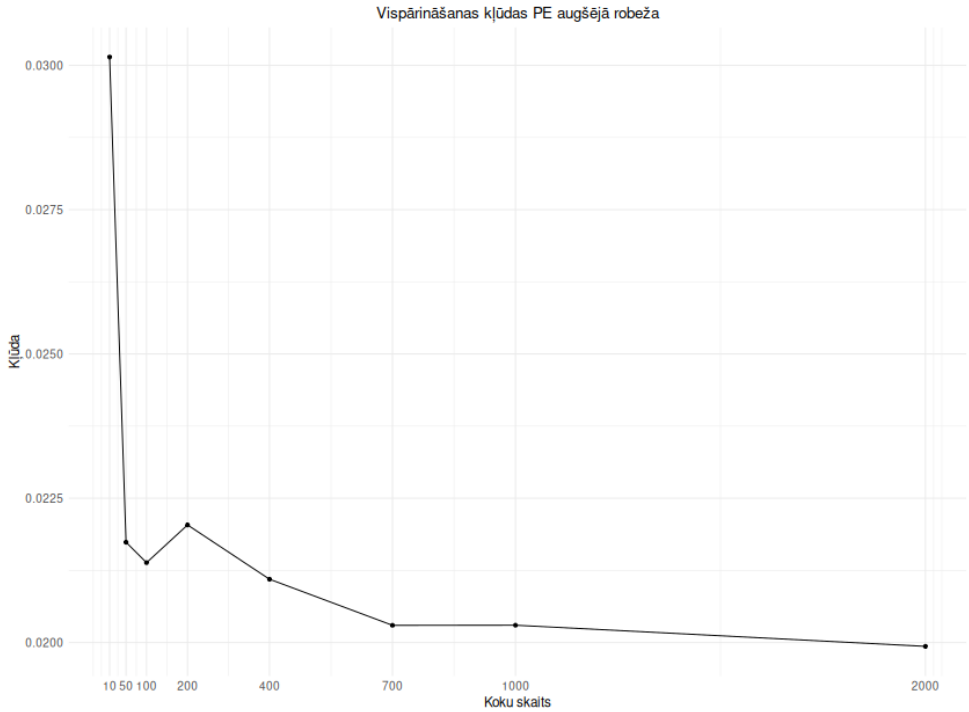
Rezultāti tiek vizualizēti divos grafikos: *AUC konverģence un standartnovirzes izmaiņas* (5) parāda, kā AUC vidējās vērtības un standartnovirze stabilizējas pie palielināta koku skaita, savukārt *Gadījuma meža veiktspēja un resursu patēriņš atkarībā no koku skaita* (6) ilustrē, ka palielinoties koku skaitam, izpildes laiks pieaug, saglabājot citus parametrus nemainīgus. Šie rezultāti empīriski apstiprina teorētiskās prognozes par vispārināšanas kļūdas konverģenci.



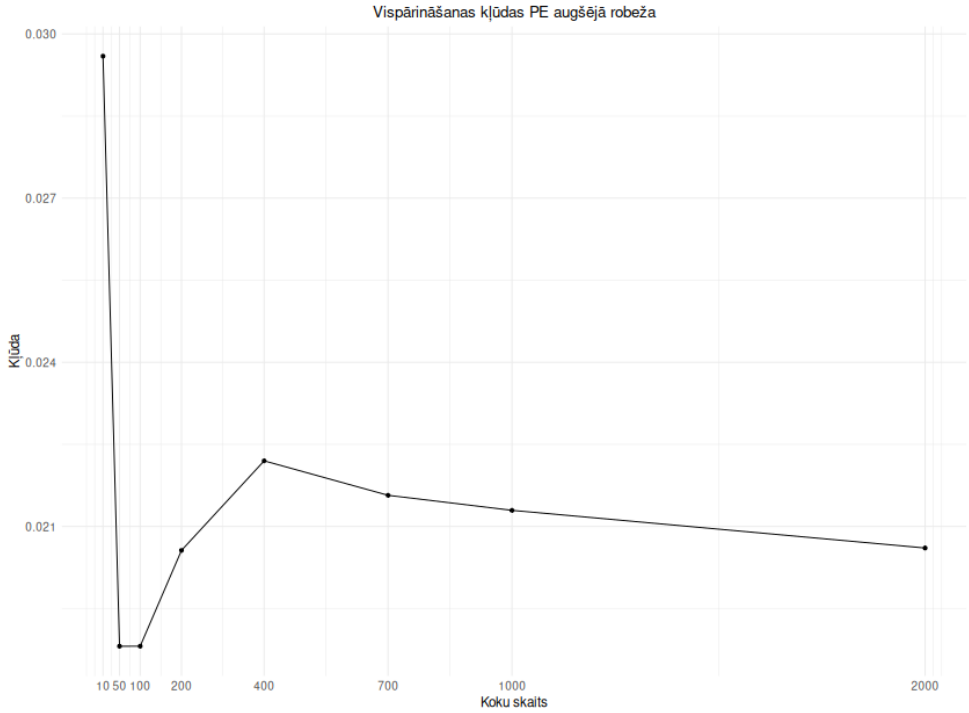
1. att. **Simulācijas rezultāts: Kļūdas PE^* augšējās robežas atkarība no koku skaita (nejaušā sēkla: 42).**



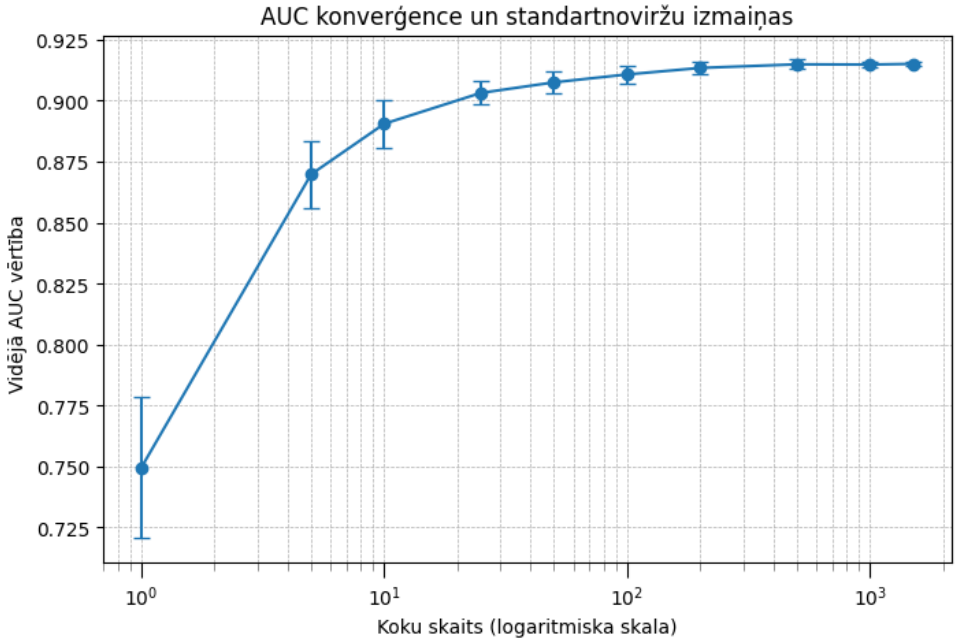
2. att. **Simulācijas rezultāts: Kļūdas PE^* augšējās robežas atkarība no koku skaita (nejaušā sēkla: 1).**



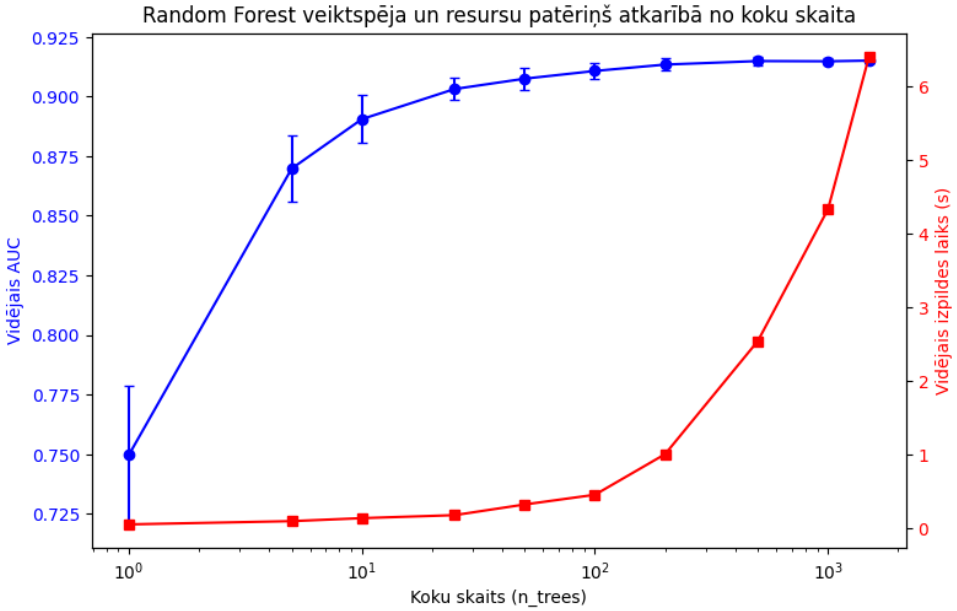
3. att. **Simulācijas rezultāts: Kļūdas PE^* augšējās robežas atkarība no koku skaita (nejaušā sēkla: 666).**



4. att. **Simulācijas rezultāts: Kļūdas PE^* augšējās robežas atkarība no koku skaita (nejaušā sēkla: 69).**



5. att. **Simulācijas rezultāts: Vidējā AUC vērtība un standartnovirze atkarībā no koku skaita (nejaušā sēkla: 42).**



6. att. **Simulācijas rezultāts: Vidējā AUC vērtība, standartnovirze un vidējais izpildes laiks (s) atkarībā no koku skaita (nejaušā sēkla: 42).**

4. SECINĀJUMI

1. Tika analizēti un izpētīti svarīgākie jēdzieni, kas saistīti ar vispārināšanas kļūdu un tās augšējo robežu;

2. Tika parādīts, ka kopējais gadījuma meža algoritms ir salīdzinoši stabils un neuzrāda pārmācīšanās pazīmes, palielinoties koku skaitam;

3. Tika izvestas formulas vispārināšanas kļūdas augšējai robežai, kurās iekļauti divi kritiski parametri: individuālo klasifikatoru precizitāti un to savstarpējo korelāciju.

Lai gan iegūtā augšējā robeža, visticamāk, ir konservatīva (*loose*), tā sniedz informatīvu funkciju gadījuma mežiem.

Praktiskajā daļā tika izveidoti simulācijas grafiki, kas ilustrē augšējās robežu uzvedību un modeļa veiktspēju, mainoties koku skaitam mežā, ļaujot vizuāli novērtēt teorētisko prognožu atbilstību empīriskajiem datiem. Darba rezultāti liecina, ka aprēķinātā augšējā robeža ne vienmēr stabilizējas, liecinot, ka robeža būtu jāprecizē un jāsaspiež tuvāk patiesajai PE^* vērtībai. Korelācijas starp klasifikatoriem un individuālais stiprums ievērojami ietekmē augšējās robežas lielumu. Savukārt rezultāti, kas iegūti sirds slimību klasifikācijas uzdevumā, parāda, ka ROC, kalpojot kā PE^* *proksi* jeb aproksimācija, konverģē uz noteikto vērtību, apstiprinot teorētiskās prognozes.

IZMANTOTĀ LITERATŪRA UN AVOTI

[1] Timothy C. Au. Random forests, decision trees, and categorical predictors: The "absent levels" problem. *Journal of Machine Learning Research*, 19:1–30, 2018. Pieejams: https://jmlr.org/papers/v19/16-474.html. Skatīts: 2026-01-05.

[2] Leo Breiman. Random forests. *Machine Learning*, 45(1), 2001. Pieejams: https://doi.org/10.1023/A:1010933404324. Skatīts: 2026-01-05.

[3] R. A. Fisher. Iris dataset. UCI Machine Learning Repository, 1936. Pieejams: https://archive.ics.uci.edu/ml/datasets/iris.Skatīts: 2026-01-05.

[4] Redwan Karim and M. S. Ony. Heart disease dataset. https://www.kaggle.com/datasets/redwankarimsony/heart-disease-data, 2021. Skatīts: 2026-01-05.

PIELIKUMI

A. PROGRAMMU KODI

```r
library(randomForest)
library(dplyr)

data(iris)

X <- iris[, 1:4]
Y <- iris$Species

compute_rmg <- function(rf, X, Y) {
  pred_trees <- predict(rf, X, predict.all = TRUE)$individual
  n_obs <- nrow(pred_trees)
  n_trees <- ncol(pred_trees)
  rmg <- matrix(0, nrow=n_obs, ncol=n_trees)
  for(i in 1:n_trees) {
    rmg[,i] <- ifelse(pred_trees[,i] == Y, 1, -1)
  }
  return(rmg)
}

compute_PE_margin <- function(rmg) {
  margins <- rowMeans(rmg)
  mean(margins < 0)
}

compute_strength_rho <- function(rmg) {
  # Strength: mean margin across all data points
  margins <- rowMeans(rmg)
  s <- mean(margins)

  # Standard deviation per tree
  sd_trees <- apply(rmg, 2, sd)

  # Weighted numerator and denominator for average correlation
  n_trees <- ncol(rmg)
  num <- 0
  den <- 0
```

```
  for(i in 1:n_trees) {
    for(j in 1:n_trees) {
      rho_ij <- cor(rmg[,i], rmg[,j])
      if(is.na(rho_ij)) rho_ij <- 0  # Treat undefined correlation as 0
      num <- num + rho_ij * sd_trees[i] * sd_trees[j]
      den <- den + sd_trees[i] * sd_trees[j]
    }
  }
  rho_bar <- num / den
  return(list(s = s, rho_bar = rho_bar))
}

compute_PE_bound <- function(rf, X, Y) {
  rmg <- compute_rmg(rf, X, Y)
  stats <- compute_strength_rho(rmg)
  s <- stats$s
  rho_bar <- stats$rho_bar
  #PE_bound <- ifelse(s>0,(rho_bar * (1 - s^2)) / (s^2),NA)
  eps <- 1e-10
  PE_bound <- (rho_bar * (1 - s^2)) / (s^2 + eps)
  return(list(s = s, rho_bar = rho_bar, PE_bound = PE_bound))
}


# Test different numbers of trees
tree_list <- c(10, 50, 100, 200, 400,700,1000,2000)
results <- data.frame(ntree=integer(), s=numeric(), rho_bar=numeric(),
                      PE_bound=numeric(), stringsAsFactors = FALSE)
seed=1
for(nt in tree_list) {
  set.seed(seed)
  rf <- randomForest(x=X, y=Y, ntree=nt)
  res <- compute_PE_bound(rf, X, Y)
  results <- rbind(results, data.frame(ntree=nt, s=res$s, rho_bar=res$rho_
      bar, PE_bound=res$PE_bound))
}

print(results)

library(ggplot2)
```

```
ggplot(results, aes(x = ntree)) +
  geom_line(aes(y = PE_bound)) +
  geom_point(aes(y = PE_bound)) +
  scale_x_continuous(breaks = tree_list) +
  labs(
    title = "āāšVisprinanas ļūkdas PE šēāaugj žrobea",
    x = "Koku skaits",
    y = "ļūKda",
    color = ""
  ) +
  theme_minimal(base_size = 6) +
  theme(
    plot.title = element_text(size = 8, hjust = 0.5),
    axis.title = element_text(size = 7),
    axis.text  = element_text(size = 6),
  )
a
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import time

from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import roc_auc_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.impute import SimpleImputer
from matplotlib.ticker import LogLocator
import warnings
warnings.filterwarnings("ignore")
```

```
# Kods ir replicējams
RANDOM_SEED = 42
np.random.seed(RANDOM_SEED)

df = pd.read_csv("heart_disease_uci.csv")

TARGET = "num"
df [TARGET] = df[TARGET].transform(lambda x: 1 if x > 0 else 0)
print(df[TARGET].unique())
X = df.drop(columns=[TARGET,"id"])
y = df[TARGET]

# Identificējam kategoriālos un skaitliskos (nepārtrauktos) mainīgus
numeric_cols = X.select_dtypes(include=["int64", "float64"]).columns
categorical_cols = X.select_dtypes(include=["object"]).columns
categorical_transformer = Pipeline(
    steps=[
        ("imputer", SimpleImputer(strategy="most_frequent")),
        ("onehot", OneHotEncoder(handle_unknown="ignore"))
    ]
)
numeric_transformer = Pipeline(
    steps= [
        ("imputer", SimpleImputer(strategy="median"))
    ]
)
preprocessor = ColumnTransformer(
    transformers=[
        ("num", numeric_transformer, numeric_cols),
        ("cat", categorical_transformer, categorical_cols)
    ]
)

X_train, X_test, y_train, y_test = train_test_split(
```

```
    X, y,
    test_size=0.3,
    random_state=RANDOM_SEED,
    stratify=y  # svarīgi, lai testa un mācīšanās datos būtu vienāda "1" un "0" pro
)

def train_rf_auc(n_trees:int, X_train, X_test, y_train, y_test, seed=42):
    """
    Trenē gadījumu mežu ar n_trees kokiem
    un atdot testa AUC.
    """
    model = RandomForestClassifier(
        n_estimators=n_trees,
        max_depth=None, # vērtība pēc noklusējuma
        min_samples_split=2, # default vērtība
        min_samples_leaf = 1, # default vērtība
        max_features="sqrt", # default vērtība
        bootstrap=True, # default
        random_state=seed,
        n_jobs=-1
    )

    pipe = Pipeline(
        steps=[
            ("preprocess", preprocessor),
            ("rfc", model)
        ]
    )

    pipe.fit(X_train, y_train)

    y_prob = pipe.predict_proba(X_test)[:, 1]   ## paņemam varbūtības
    auc = roc_auc_score(y_test, y_prob)
```

```
    return auc

tree_counts = [1, 5, 10, 25, 50, 100, 200, 500, 1000, 1500]
auc_scores = []
for k in tree_counts:
    auc = train_rf_auc(
        n_trees=k,
        X_train=X_train,
        X_test=X_test,
        y_train=y_train,
        y_test=y_test,
        seed=RANDOM_SEED
    )
    auc_scores.append(auc)

plt.figure(figsize=(8, 5))
plt.plot(tree_counts, auc_scores, marker="o")
#plt.xscale("log")
plt.xlabel("Koku skaits")
plt.ylabel("AUC")
plt.title("Gadījuma meža AUC kā funkcija no koku skaita")
plt.grid(True)
plt.show()

def repeated_auc(n_trees, n_repeats):
    """
    Funkcija, kas simulē AUC vidējo vērtību un standartnovirzi
    katram koku skaita parametram.
    """
    aucs = []
    times = []
    for i in range(n_repeats):
        start_time = time.perf_counter()
        aucs.append(
```

```
            train_rf_auc(
                n_trees,
                X_train, X_test,
                y_train, y_test,
                seed=RANDOM_SEED + i
            )
        )
        total_time = time.perf_counter()-start_time
        times.append(total_time)
    return np.mean(aucs), np.std(aucs), np.mean(times)

means, stds, mean_times = [], [],[]
n_repeats=20
for k in tree_counts:
    mean_auc, std_auc, mean_time = repeated_auc(k, n_repeats=n_repeats)
    means.append(mean_auc)
    stds.append(std_auc)
    mean_times.append(mean_time)

results= pd.DataFrame({"Koku skaits": tree_counts,
    f"AUC vidēja vērtība (atkartojumu skaits: {n_repeats})":means,
                        f"std(AUC) (atk. skaits: {n_repeats})":stds,
                        f"Vidējais aprēķina laiks, s (atk. skaits: {n_repeats})":mea
results

plt.figure(figsize=(8, 5))
plt.errorbar(tree_counts, means, yerr=stds, marker="o", capsize=4)
plt.xscale("log")
plt.gca().xaxis.set_minor_locator(LogLocator(base=10.0, subs=range(2,10)))
plt.gca().xaxis.set_major_locator(LogLocator(base=10.0))
plt.grid(True, which="both", linestyle="--", linewidth=0.5)

plt.tick_params(axis='x', which='minor', length=4, color='gray')
plt.tick_params(axis='x', which='major', length=8, color='black')
```

```
plt.xlabel("Koku skaits (logaritmiska skala)")
plt.ylabel("Vidējā AUC vērtība ")
plt.title("AUC konverģence un standartnoviržu izmaiņas")
plt.show()

fig, ax1 = plt.subplots(figsize=(8,5))
color = 'blue'
ax1.set_xlabel('Koku skaits (n_trees)')
ax1.set_ylabel('Vidējais AUC', color=color)
ax1.errorbar(tree_counts, means, yerr=stds, color=color, fmt='-o', capsize=3, label
ax1.tick_params(axis='y', labelcolor=color)
ax1.set_xscale("log")

ax2 = ax1.twinx()  # otra y ass laika vizualizācijai
color = 'red'
ax2.set_ylabel('Vidējais izpildes laiks (s)', color=color)
ax2.plot(tree_counts, mean_times, '-s', color=color, label='Vidējais laiks')
ax2.tick_params(axis='y', labelcolor=color)

fig.tight_layout()
plt.title('Gadījuma meža veiktspēja un resursu patēriņš atkarībā no koku skaita')
plt.show()
```