

OTDK-dolgozat

2025

Budapesti Corvinus Egyetem



Terméselőrejelzés a gépi tanulási módszerek segítségével

Crop production forecasting using machine learning methods

Rajka László

Konzulensek: Stump Árpád, Fülöp András

Kézirat lezárásának dátuma: 2023. január 29.

Absztrakt

A dolgozat a mezőgazdasági modellezés témakörén belül a búzatermés időjárásalapú előrejelzését vizsgálja. A tanulmány arra keresi a választ, hogy építhető-e olyan hatékony előrejelzési modell, amely az időjárási információk segítségével képes megbecsülni a jövőbeni búzatermés mértékét. A dolgozat hipotézise, hogy van kapcsolat az időjárási adatok és a búzatermés mértéke között. Ennek igazolásához a dolgozat először áttekinti a főbb szakirodalmi elemeket, valamint az aktuális modellezési irányokat. Módszertani szempontból a faalapú modellek hatékonyságát vizsgálja a dolgozat, melyen belül részletes összefoglalót ad az XGBoost algoritmus háttéréről. A modell segítségével az írás megállapítja, hogy RMSE értéke az XGBoost modell esetén a hyperopt algoritmus által javasolt paraméter beállítással 0,86-ra csökkenthető a mintaátlagtól vett 3,78-as referencia értékről. Az előrejelzés kapcsán a legjobb változók a februári felhőtakaró, az augusztusi és a decemberi csapadék mennyisége. A tanulmány továbbá arra az eredményre jut, hogy összességében több hónap esetén is fontos szerepe van a csapadék mennyiségének, az esős napok számának, illetve a talajmenti fagynak az termés becslése szempontjából.

Tartalomjegyzék

1. Bevezetés	1
2. Irodalomáttekintés	2
2.1. A búzaelőrejelzés fejlődése	2
2.2. Az XGBoost metódus alkalmazásai	4
3. A módszertan és a felhasznált adatok bemutatása	5
3.1. Az XGBoost modell felépítése	6
3.2. XGBoost fák struktúrálása	8
3.3. Hyperopt algoritmus	9
4. Eredmények	10
4.1. Az adatbázis előállítása	10
4.2. Az adatok leíró jellegű elemzése	11
4.3. Modellezési eredmények	12
5. Összefoglalás	14
Hivatkozások	16
A. Függelék	18

Táblázatok jegyzéke

1. A CRU adatbázis változói. forrás: Harris, Osborn, Jones, és Lister (2020) . . .	10
2. A hyperopt által talált legjobb paraméterek. forrás: saját szerkesztés	13

Ábrák jegyzéke

1. A célváltozó eloszlása és dobozábrája forrás: saját szerkesztés	12
2. Feature importance érték a változók esetén forrás: saját szerkesztés	14

1. Bevezetés

„A májusi eső aranyat ér” - egy igen híres közmondás, mely bizonyítja, hogy a bonyolult előrejelzési rendszerek előtt is a józan paraszti ész segítségével a földművelők gyakorlatilag egy trendet találtak egy időjárási tényező és a várható termésük között. A mezőgazdaság és az időjárás mindig kéz a kézben jártak egymással. Egészen az ókortól napjainkig a termés szempontjából fontosnak tekinthető, hogy milyen idő van kilátásban. Gyakran lehet hallani a hírekben, hogy az óriási hőhullámok, a sokáig húzódó fagy mennyire lerontja a mezőgazdasági termékek volumenét. Persze a pozitív hatásokat is fontos megemlíteni, például az esős napokat vagy a jókor jó helyen érkező csapadékot. Természetesen a búzatermesre számos tényező hatással van, ilyen például a növénybetegségek, a termőföld és a mezőgazdasági szaktudás. A dolgozatomban, viszont kizárólag az időjárási tényezőket vizsgálom.

A dolgozatomban szeretnék mélyebbre ásni és az egyszerű megfigyelések helyett az elérhető időjárás és termés adatok alapján modellt javasolni a búzatermes előrejelzésére. A tanulmány kutatási kérdése, hogy építhető-e kis hiábval dolgozó előrejelzési modell, amely az időjárásadatok segítségével meghatározza az éves várható termés mértékét az Európai Unió régióban. A dolgozat hipotézise, hogy van kapcsolat az időjárás és a búzatermes mértéke között és az időjárási információkkal erős becslés adható a termés mértékére. A búzából készülő liszt számos élelmiszer alapanyaga, mely a társadalom számára könnyen elérhető, így egy alapvető élelmészeti cikk, ezért sem véletlen, hogy a történelem során számos háború célja a minőségi termőföld volt. Az olyan mezőgazdasági termékek, mint a búza fontos szerepet játszanak a világkereskedelemben, így az előrejelzési modelleknek jelentős gazdasági haszna is van.

Az eredmények tekintetében két XGBoost modell kerül összehasonlításra, melyek tíz időjárási információt tartalmazó adatsor alapján végzik a becslést. A dolgozat megállapítja, hogy a paraméter optimalizált XGBoost modell segítségével a hektáronkénti tonnában megadott termés RMSE értéke 0,93-ra csökkenthető, mely jelentős javulást jelent a 3,78-as termésátlagtól számolt értékhez képest. A dolgozat javaslatot tesz egy olyan paraméter beállításra, mellyel hatékony becslés adható. A tanulmányból megállapítja, hogy a legfontosabb változók az esős napok száma, illetve a csapadék mennyisége, melyek a januári és a nyári hónapok során jelentősek.

A mostani bevezető rész után szeretném bemutatni, hogy az eddigiekben hogyan fejlődött a terméselőrejelzés, illetve a módszertanban használatos XGBoost technika felhasználásával, milyen eredményeket értek el az előrejelzés tekintetében. A harmadik fejezetben kifejtem az említett modell módszertani vonatkozásait, illetve a felhasznált paraméter optimalizáló csomagokat. A tanulmány negyedik részében mutatom be a becslés eredményeit, végül egy összefoglalóval zárom a dolgozatot.

2. Irodalomáttekintés

Jelen fejezetben foglalom össze, hogy miként fejlődött a terméselőrejelzés egészen napjainkig. Igyekeztem a technikai fejlődéssel párhuzamosan összefoglalni a becslési módszerek alakulását is időrendi sorrendben. A fejezet második részében külön mutatom be a frissebb gépi tanulási előrejelzésen alapuló kutatásokat, melyek alkalmazzák az XGBoost módszert. A tanulmányok tekintetében is szeretném bebizonyítani, hogy a terméselőrejelzés szempontjából jelentős megbízhatóság növekedést nyújt a fejlettebb becslési módszerek alkalmazása.

2.1. A búzaelőrejelzés fejlődése

Már az ókori Egyiptom világában is fontos szerepet játszott a termés előrejelzése és meghatározása, hiszen az egyiptomi civilizáció kialakulásában jelentős szempont volt a Nílus termékeny völgye. A sivatagban aligha volt bármennyi csapadék ám a természet körforgásának köszönhetően bizonyos előre jelezhető időközönként a Nílus megáradt és ez megváltást jelentett az itt élő népesség számára. Az áradás előrejelzése igen fontossá vált, így ennek kalkulálásával tudták tervezni a vetést. Az egyiptomi uralkodók ennek érdekében csillagászokat alkalmaztak, akik ha mondhatjuk már idő számításunk előtt 3000 évvel is foglalkoztak a témával. (Borhegyi, 2020)

Természetesen a későbbiekben sokkal fejlettebb technikák is megjelentek de a XXI. századi technológiai fejlődés előtt is több cikk született azzal kapcsolatosan, hogy az időjárás segítségével adható-e megfelelő előrejelzés búzatermesre. Chipanshi, Ripley, és Lawford (1997) Saskatchewan kanadai tartomány mintáit vizsgálták 1960 és 1990 közötti információkon. Modelljükben az első három negyedév időjárás adatai segítségével próbálták meghatározni a termés mennyiségét. Az alkalmazott regressziós technika segítségével bár még a ma használatos modellekhez képest kevésbé pontos becslést végeztek, ám kimutathatóan tudták magyarázni a termést.

De Wolf és Franel (1997) a problémát elsősorban epidemiológiai oldalról fogták. A modelljükben a növekedési időszak során használták fel időjárás adatokat. Korrelációt találtak a cserfoltosságra a növény növekedési szakaszára és a következő változók között: hőmérsékletet, relatív páratartalmat, szélesebséget, szélirányt, napsugárzást. Az említett adatokat 24 órás összefoglalásban használták fel, azon hónapok esetén, amelyekre jellemző volt a búzát fertőző betegségek elterjedése. Modellént összehasonlították a neurális hálók és a regressziós eszközök hatékonyságát a problémán. Az eredményeik tekintetében a neurális háló (ANN) megbízhatóbb becslést adott a diszkriminancia analízisnél és a logisztikus regressziónál. Az említett tanulmány is alátámasztja, hogy érdemes fejlettebb statisztikai módszerekkel megközelíteni a termés előrejelzését.

Hasonló problémát fogalmazott meg egy ausztrál cikk is, ahol ugyanúgy neurális hálók segítségével próbáltak becsléseket készíteni. A tanulmány, viszont más területi mintát, Ausztrália,

Brazília és Columbia időjárás és búzatermés adatait használta fel. Chakraborty et al. (2004) is bizonyítja, hogy a gépi tanulási módszerek alkalmasak a termés és az időjárás adatok közötti összefüggések leírására.

Sajnos napjainkban egy igen erős éghajlat változásnak lehetünk szemtanúi, mivel egy folyamatosan emelkedő hőmérsékletet tapasztalhatunk. Lobell, Schlenker, és Costa-Roberts (2011) cikke is rögzíti, hogy a hőmérsékleti trendek 1980 és 2008 között meghaladták a történelmi évenkénti változékonyság átlagát és szórását. Tanulmányában kitér arra, hogy csupán a hőmérséklet emelkedésével a búzatermés átlagosan 5,5 %-al esett vissza. A felmelegedés negatív hatásaival kapcsolatban Hatfield (2016) is hasonló eredményekre jutottak, sőt Prasad, Bheemanahalli, és Jagadish (2017) a hóhullámok eseményeit is kapcsolatba tudták hozni a termés mennyiségének csökkenésével.

Kamir, Waldner, és Hochman (2020) tanulmányában azt vizsgálja, hogy jelentős eltérés figyelhető meg a hivatalos közlésekben megjelenő potenciális búzatermés és a valós hozamok között. tanulmányukban több gépi tanulási algoritmus használnak a termés előrejelzéséhez. A modellek magyarázó változóiként időjárás és műholdas felvételek adatait használták fel. A módszereik közül a legjobb előrejelzést a vektor tartógép eljárás adta a vizsgált modellek közül. Megfogalmazásuk szerint a két legerősebb változót a maximális hőmérséklet és a csapadék jelentette. Összehasonlították a hagyományos statisztikai eljárásokat is a gépi tanulási módszerekkel és arra jutottak, hogy a búzatermés előrejelzés témakörében érdemes alkalmazni a fejlettebb módszereket, mivel ezek adhatnak jelentős javulást az előrejelzés pontosságában.

A kutatásom az Európai Unió (továbbiakban: EU) országait vizsgálja, így több cikket is szeretnék bemutatni, amely a térségben vizsgálta a problémát. Számomra sokat segített a probléma felkutatásában Cantelaube és Terres (2005) cikke, akik alapvetően az időjárás előrejelzések hasznosságát vizsgálták meteorológiai szempontból a termés előrejelzések hatékonyságán. A cikk is a téli periódus kezdetétől nézve használta fel az időjárás adatokat egészen augusztusig. A kutatásomban én is ezt a logikát használtam fel a modellezéshez. Az Európai Unió egyik szervezete a JRC (Joint Research Centre) minden évre végez specifikus terméselőrejelzést. Ezen termés előrejelzéseket egészítették ki a fellelhető időjárás adatokkal és arra jutottak, hogy a modell egy valószínűségi eloszlást felhasználó technikával megbízhatóbb becslést ad az EU termésadatokra.

Az egyik legösszetettebb kutatást a témában Van der Velde et al. (2018) végezték, mely végül a Nature szakmailapban is megjelent. Az említett tanulmány is a JRC adatairól szól, viszont itt az előrejelzések megbízhatóságát értékeli. A cikk összefoglalót is ad az EU által használt modellezési folyamatról, melynek eredményeit a MARS (Monitoring Agricultural ResourceS) nevű jelentésben teszik közzé. A JRC következő három adattípust használja fel a predikciókhoz: meteorológiai indikátorok, termésmodell indikátorok, vegetációs és távolsági adatok. Az Európai

Unió rendelkezik ehhez megfelelő adatmennyiséggel. Modellezési szempontból a JRC egy komplex modellt használ, amelyben felhasználnak trendelemzést (lineáris, logaritmikus), illetve a regresszióelemzés mellett PCA klaszterezést is használnak például évek közötti hasonlóságok kereséséhez. A tanulmány megállapítja, hogy a JRC elemzései a medián években 2% alatti hibával adtak előrejelzést, míg az alacsony és magashozamú évek során erősebben alul becsülték a termelt mennyiséget (10% és 8%). A tanulmány emellett rögzíti, hogy a 2000 utáni extrém években 10,4 és 70,8 százalék közötti alul becsléseket találtak. A cikk az eredmények mellett javaslatot tesz arra, hogy az EU fejlessze a megfigyelő rendszereit a probléma elhárítására.

Úgy gondolom az irodalomáttekintés is bizonyítja, hogy a terméselőrejelzés fontos eleme a gazdasági folyamatoknak. A szakmai életben is egy aktuális témának számít, hiszen a bemutatott tanulmányok többsége megemlíti az egyre extrémebb időjárási körülményeket, illetve neves szaklapokban is téma a modellek hatékonysága.

2.2. Az XGBoost metódus alkalmazásai

Ahogy az előző részben is részleteztem a gépi tanulási módszerek komoly hangsúlyt kaptak az elmúlt évtizedek során a búzatermés előrejelzése szempontjából. Ahogy bemutattam a döntési fák, neurális hálók, klaszterezési eljárások és a trendelemzést is teret nyert a tudomány ezen területén. A dolgozatomat viszont egy manapság igen népszerű algoritmusra építettem fel, melyről több tanulmány is született a termés kérdéskörében. Az előző alfejezetben bemutattam, hogy miként fejlődött az előrejelzés és milyen irányelveket találhatunk módszertani és szakmai szempontból, viszont a mostani fejezetben szeretnék kifejezetten olyan tanulmányokat bemutatni, amelyek az XGBoost regressziós eljárást alkalmazzák. Az eljárás pontos részleteit a módszertani részben fogom kifejteni.

Egy friss tanulmányban indiai adatokon végeztek előrejelzést az időjárás, a csapadék és az öntözött terület magyarázó változók segítségével. A tanulmányban is az XGBoost regressziós eljárást alkalmazzák, mely bizonyítja, hogy több keresztvalidációs eljárás eredményeként is a legjobb eredményt az említett eljárás adja. A tanulmány azt is rögzíti, hogy a Bayes-féle eljárás segítségével jobb performancia és robosztusság adható. Az összehasonlíthatóság érdekében a modellezés során célváltozónak a hektáronkénti becslést végeztek. A tanulmány ad egy meghatározott értéket a jövő évi várható búzatermés mértékére, illetve azt is rögzíti, hogy az extrém időjárási viszonyok miatt erős csökkenő trend figyelhető meg az ő mintájukban is a termés kapcsán (Gong, Huang, és Zhang, 2022).

Srivastava et al. (2022) az őszi búza előrejelezhetőséget vizsgálta a növény genotípusa, az időjárás és a gazdálkodási formák mentén. A kutatásuk során Németország 271 megyéjében végeztek vizsgálatokat 1999 és 2019 közötti mintán. Megállapításuk szerint a CNN (Konvolúciós neurális hálók) módszer alkalmazása adja a leghatékonyabb eredményt, melyet 8 különböző gépi

tanulási módszerrel hasonlítottak össze. Fontos megállapítást tesznek azzal kapcsolatban, hogy a nem lineáris modellek alkalmazása jelentős javulást ad a becslés pontossága szempontjából. Három hatékonyan becselő modell kerül meghatározásra, melyek közül az egyik az XGBoost eljárás. A CNN modell alapján a termés szempontjából a legkritikusabb információk a szélesebb és a tizedik héten és a sugárzás mennyisége a hetedik héten az időjárási változók tekintetében.

Az XGBoost eljárások alkalmazása a búzatermés meghatározásakor nem csak az időjárási adatok kapcsolatában lehet releváns. Yang et al. (2021) A talajnövény-elemzési fejlődési mérések alapján a búzalevelek klorofiltartalmát használták fel az előrejelzéshez. A felhasznált adataik igen pontosak voltak, mivel drónokkal gyűjtötték be az információt, melyet a kínai észak-anhui nemzeti talajmegfigyelő és kísérleti állomáson végeztek el. A búzanövekedés és termés tekintetében összehasonlították a döntési fák a random forest és a k-means eljárást is. Az említett algoritmusok közül a legjobb eredményt az XGBoost eljárás adta. Megállapítják, hogy a talaj szervesanyag és nitrogén tartalmának információja pozitívan befolyásolja a becslés pontosságát.

Oikonomidis, Catal, és Kassahun (2022) cikke kielemezi, hogy jelenleg még mindig sok a pontatlan modell a termés előrejelzése kapcsán és szeretnék alternatívákat bemutatni a kérdésben. A tanulmányban összehasonlítják az XGBoost gépi tanulási algoritmust, a konvolúciós neurális hálózatok (CNN)-mély idegi hálózati típusát (DNN), a CNN és XGBoost kombinációjából épített modell, a CNN-ismétlődő neurális hálózatok (RNN) és a CNN-hosszú távú memóriára épülő hálózatok (LSTM) teljesítményét. Fontos megjegyezni, hogy a cikk szójába előrejelzést végez időjárási és talajparaméter adatok segítségével. Amiatt szeretném mégis bemutatni, mert igen fejlett módszereket hasonlít össze a termés tekintetében. A kombinált hibrid modell adja a legerősebb előrejelzést ám ettől nem marad el sokkal hatékonyságban az XGBoost regressziós eljárás sem. Nagyon alacsony RMSE értéket mutatnak be a szerzők, melynek oka a hatékony modellen túl a nagy minta (25 345 darab).

A felsorolt cikkek vizsgálatában véleményem szerint megállapítható, hogy a gépi tanulási algoritmusok jelentős hatékonyság növekedést adnak a búzatermés előrejelzése kapcsán. A lineáris modellek hatékonysága nem megfelelő a nem lineáris módszerek közül pedig a neurális hálók és a fejlettebb döntési fa alapú modellek bizonyulnak a leghatékonyabbnak. Mivel a tanulmányom során kis mintát elemzek és a modellezés szempontjából az egyszerűbb paraméterezése és a könnyebb érthetőségre törekszem a fenti cikkek alapján az XGBoost eljárás szeretném megvizsgálni európai mintán.

3. A módszertan és a felhasznált adatok bemutatása

A dolgozat következő szekciójában szeretném bemutatni a projekthez felhasznált módszertani eszközöket, illetve az elemzés során alkalmazott programcsomagokat és azok statisztikai hátterét.

Az összes számítást, átalakítást a python3 környezetében végeztem el (Van Rossum és Drake, 2009).

3.1. Az XGBoost modell felépítése

AZ XGBoost módszer alkalmazásához a python XGBoost csomagját használom, melynek fejlesztői egy részletes leírást készítettek az eljárás módszertanáról. A módszertani részben erről készítettem egy összefoglalót (xgboost developers, 2022). Az XGBoost az úgynevezett felügyelt tanulási problémák esetén használatos, ahol x_i magyarázó változókat (több funkcióval is rendelkezhetnek) egy y_i célváltozó előrejelzésére használjuk. Ahhoz, hogy bemutassam a fákkal való lehetőségek tárházát, először kezdjük a felügyelt tanulás alapvető elemeinek áttekintésével. A felügyelt tanulás során több magyarázó változót alkalmazunk egy célbecslés eléréséhez, ilyen például egy lineáris model $\hat{y}_i = \sum_j \theta_j x_{ij}$, ahol becslés végpontja y_i a bemeneti értékek súlyozott lineáris kombinációja lesz. A predikciós érték, így könnyen felírható pár paraméter megadásának segítségével, de a regresszió kis módosításokkal átalakítható osztályozási problémára is. Ám egy felügyelt tanulási módszer valószínűségi paraméterértelmezéssé is átalakítható például egy logisztikus eljárás alkalmazásával. Látható, hogy az adott bemeneti adatok adottságként kezelhetők, melyeket bár át lehet alakítani az információ gyökere ugyanaz marad. Amikor egy modellezési eljárást végzünk, amin módosíthatunk azok a paraméterek. Ahogy fentebb is részleteztem a lineáris esetben például a θ paraméterek módosításával adott becslés lehet a módszertani mozgástér. A paraméterek módosításához a modellek tanítása az eszköz, melyek végeztével meg kell határozni, hogy mennyire pontos az illeszkedés, hogy kiválaszthassuk a leghatékonyabb eljárást.

Az eljárások során meg kell határozni egy célfüggvényt, melynek optimalizálása a feladat. Az ilyen említett célfüggvények két részből állnak: egy tanító adaton mért becslési veszteségből (training loss) és egy regularizációs tényezőből:

$$\text{obj} \tag{1}$$

ahol L a becslési veszteség és Ω a regularizációs tényezőt. A becslési veszteség adja meg, hogy mennyire ad erős predikciót az általunk meghatározott modell. A regularizációs tényező pedig arra szolgál, hogy a tanulási modell, hogy viszonyul a képzési adatokhoz, ennek akkor lesz jelentősége, hogy segít elkerülni a túlillesztést, illetve szabályozza a modell összetettségét. Gyakori választás az L felírására az átlagos négyzetes hiba (RMSE), melye a következőként írható fel:

$$L(\theta) = \sum_i (y_i - \hat{y}_i)^2 \tag{2}$$

A bemutatott módszerek értékeivel meghatározhatunk különbségeket a performancia szempontjából például a döntési fák, a random forest, illetve a boosted forest eljárások kapcsán, így

formalizált módon tudunk állni a paraméterek módosításához.

Az előzőekben részletezett felügyelt tanulás mellett az XGBoost eljárás másik alapját a döntési fák együttese jelenti, melyek (akár klasszifikációs és regressziós esetben) halmazból optimalizációs eljárás segítségével modellt építhetünk (jelölése: CART). Ilyenre egy egyszerű példa lehet, hogy valakinek tetszene-e egy számítógépes játék vagy sem. A példában veszünk egy mintát egy család tagjaitól, akkor számos információ rendelkezésünkre áll: kor, nem, foglalkozás és sok egyéb más. Az adott információk mentén a család tagjait különböző levelekbe sorolhatjuk, ám a CART modellekben csak döntési értékeket határozzunk meg. Az így kapott levelek a valós pontszám értékeket tartalmazzák, amely egy erős eszköz az egységes optimalizáláshoz a modellezés során. Általában egy fa felépítése nem ad erős információt, így érdemes olyan eljárást készíteni, ahol több fa együttese kerül kiértékelésre.

Ahhoz, hogy jobban értsük a fák alkalmazását írjunk fel egy két fából álló faegyüttest. Az egyes fák előrejelzési pontszámait összeadjuk, hogy megkapjuk a végső pontszámot. Itt fontos tény, hogy a két fa igyekszik kiegészíteni egymást. Matematikai szempontból ez a következőképp írható fel:

$$\hat{y}_i = \sum_{k=1}^K f_k(x_i), f_k \in \mathcal{F} \quad (3)$$

Ahol K a fák száma, f_k egy függvény az F függvényterben, és F tartalmazza az adott CART összes lehetséges kimenetét. Ahogy az előzőekben részleteztem meg kell keresni az adott kombi-nációs halmazon a megfelelő azaz a legpontosabb kimenetet, melyhez a következő optimalizációs célfüggvényt lehet rendelni:

$$\text{obj}(\theta) = \sum_i^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \omega(f_k) \quad (4)$$

ahol $\omega(f_k)$ a fa összetettségét jelöli az f_k pedig az előzőekben definiált függvényt. Az említett fa együttesek képzése nagyon hasonló a random forest eljáráshoz, hiszen ezek is a faegyüttesek értelmezésén alapulnak. A random forest és a boosted tree-k között a különbség tanító adaton történő képzésből adódik, melyből következik, hogy egy predikciós függvény felhasználható mindkét eljárás esetében.

Természetesen adódik a kérdés, hogy milyen paraméterek mentén differenciálhatunk a faegyüttesek képzése során. Azaz milyen stratégiát veszünk figyelembe akkor, amikor az f_i függvények tanulását és kiértékelését végezzük egy algoritmussal, mely függvények tartalmazzák a fa szerkezetét és a levél pontszámait. A fák struktúrájának tanulása sokkal összetettebb, mint egy egyszerű optimalizálási probléma, ahol egyszerűen a gradiensok kerülnek figyelembevételre, így a technika során additív jellegű stratégiát alkalmazunk. Gyakorlatban megnézzük, hogy eddig mit tanultunk és az milyen irányba javítható, mely képez egy új fát. Ennek értelmében felírhatunk egy előrejelzési (predikciós) értéket t lépésenként, mint $\hat{y}_i^{(t)}$. Így a következőképp

adható meg a módosított predikciós érték:

$$\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\
\hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
&\dots \\
\hat{y}_i^{(t)} &= \sum_{k=1}^t f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned} \tag{5}$$

Fontos kérdés az additív eljárásnál, hogy melyik fát szeretnénk használni az egyes lépéseknél. Ebben az esetben mindig hozzáadjuk azt amelyik optimalizáló hatással van a célunkra:

$$\begin{aligned}
\text{obj}^{(t)} &= \sum_{i=1}^n l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^t \omega(f_i) \\
&= \sum_{i=1}^n l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \omega(f_t) + \text{constant}
\end{aligned} \tag{6}$$

re

$$\begin{aligned}
\text{obj}^{(t)} &= \sum_{i=1}^n (y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)))^2 + \sum_{i=1}^t \omega(f_i) \\
&= \sum_{i=1}^n [2(\hat{y}_i^{(t-1)} - y_i)f_t(x_i) + f_t(x_i)^2] + \omega(f_t) + \text{constant}
\end{aligned} \tag{7}$$

Ebben az esetben most csak az RMSE függvényt mutattam be, viszont több veszteségfüggvény kezelésére is képes az XGBoost eljárás.

Ahogy korábban is kifejtésre került az optimalizáló lépések mellett a felügyelt tanulás tartalmaz egy regularizációs tényezőt is azaz meg kell határozni a $\omega(f)$ fa összetettségét. Ennek érdekében elsőként mélyebben kell értelmezni a fa $f(x)$ definícióját:

$$f_t(x) = w_{q(x)}, w \in R^T, q: R^d \rightarrow \{1, 2, \dots, T\}. \tag{8}$$

ahol w a levelek pontszámainak vektora, q pedig egy olyan függvény, amely minden adatpontot hozzárendel a megfelelő levélhez, T pedig a levelek számát jelöli. Így az XGBoost esetén a komplexitást a következőképp definiálhatjuk:

$$\omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 \tag{9}$$

3.2. XGBoost fák struktúrálása

A fák átstruktúrálásának segítségével kaphatunk egy pontszámot, melyet a szaknyelv struktúrális pontszámként (Structure Score) tart számon. A struktúrális értéket rövidített formában a

következőképp kaphatjuk meg:

$$\begin{aligned} w_j^* &= -\frac{G_j}{H_j + \lambda} \\ \text{obj}^* &= -\frac{1}{2} \sum_{j=1}^T \frac{G_j^2}{H_j + \lambda} + \gamma T \end{aligned} \quad (10)$$

Az egyenletben w_j értékek függetlenek egymástól és a legjobb értéke adja a legjobb $q(x)$ struktúrát, mely gyakorlatban azt adja meg, hogy egy adott fa mennyire jól teljesít. Alapvetően egy adott fa struktúra esetében a g_i és h_i statisztikákat a megfelelő levelekre toljuk, a statisztikákat összegezzük, és a képlet segítségével kiszámítjuk, hogy mennyire jó a fa. Ez a pontszám gyakorlatilag a döntési fa szennyezettségi mérőszáma, mely figyelembe veszi a modell komplexitását is.

$$\text{Gain} = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma \quad (11)$$

A képletet értelmezhetjük úgy, hogy az első elem az új baloldali levél pontszáma, a második elem az új jobb oldali levél pontszáma, a harmadik elem az eredeti levél pontszáma, végül pedig a levélre vonatkozó regularizációt láthatjuk. Itt fontos megállapítani, hogy a növekedési érték, amennyiben kisebb mint γ , akkor nem érdemes hozzáadni a modellhez az adott ágat. Valós értékű adatok esetén ennek segítségével közelítjük az optimális felosztást, ahol gyakorlatilag balról jobbra haladva a metszési technikával kikeressük az optimalizált modellt (xgboost developers, 2022).

3.3. Hyperopt algorithms

Azon modellezési problémák esetén, melyek során összetett algoritmusokat alkalmazunk gyakran lehet találkozni a paraméterek kérdéskörével. Egy-egy modell esetén számos paramétert ismerhetünk, melyek jelentősen befolyásolják a tanulási folyamatot. Ilyen paraméter például a fák száma, a tanulás mélysége, vagy az elvégzett becslések száma. Sok gépi tanulási módszer hatékonysága jelentősen függ a paraméterek hatékony optimalizálásától. A hangolást a fejlesztők gyakran kézzel, illetve próbálgatással végzik, mellyel próbálják elérni az adott becslési hatékonyság mutató legoptimálisabb mértékét. A probléma feloldására Bergstra, Yamins, és Cox (2013) fejlesztett egy metamodellezési megközelítést mely során egy kinevezett mutatóra optimalizálják a modell paramétereit. A teljesítmény mutató optimalizálásával szeretném csökkenteni a becslési hibákat a modell építés során, mellyel véleményem szerint jelentős hatékonyság növekedést sikerült elérnem.

4. Eredmények

A következő részben szeretném részletezni az adatbázisösszeállítását a modellezés folyamatát és az elért eredményeket. Ahogy a bevezetőben is bemutattam a dolgozat célja, hogy vizsgálja az időjárásadatokkal való terméselőrejelzés lehetőségét és a gépi tanulási metódusok hatékonyságát a kérdésben. A projekt három fő részből állt. Az adatok letöltéséből és átalakításából, az információk vizsgálatából és végül az előrejelzési modellek összehasonlításából. Az említett részeket ebben a szekcióban mutatom be részletesen.

4.1. Az adatbázis előállítása

A modellezéshez használt időjárás adatokat a Harris et al. (2020) által közölt weboldalról töltöttem le. Az időjárásadatok havi bontásban kerültek publikálásra a világ összes országára 1901-től 2021-ig. A weboldalon rendelkezésre álló 10 változót az 1. táblázatban mutatom be. Az adatbázis láthatóan tartalmazza a főbb időjárás adatokat, melyek mindegyike folytonos változó. Az adatbázisban több esetben is volt hiányzó érték melyet a -999-es értékkel jelöltek. Az ilyen értékeket tartalmazó adatsorokat kizártam a modellezés során, hogy ne zavarják a megbízható becslést. Országonként és változónként külön fájlokban voltak közzé adva az adatok, így külön programkódot kellett írni a végső adatbázis kialakításához.

Jelölés	Jelentés	Mértékegység
cld	felhőtakaró	százalék
dtr	napi hőmérséklet tartomány	Celsius
frs	talajmenti fagy	nap
pet	potenciális evapotranspiráció	mm/nap
pre	csapadék	mm/hónap
tmn	minimum hőmérséklet	Celsius
tmp	átlagos hőmérséklet	Celsius
tmx	maximum hőmérséklet	Celsius
vap	légnymomás	hPA
wet	esős napok száma	nap

1. táblázat. A CRU adatbázis változói.

forrás: Harris et al. (2020)

Az előrejelzések másik lábát képezik a búzatermeszre vonatkozó adatok, melyet a Food and Agriculture Organization of United Nations (2021) növénytermesztési és állattenyésztési adatok oldaláról kértem le. Az adatbázis 1961 és 2021 közötti időszakon országos bontásban

tartalmaz a búzatermésre tonnában kifejezett mennyiségeket. Az adatok országonként így összesen 60 rekordot tartalmaznak, amely igen kevésnek tekinthető modellezési szempontból, emiatt egy közös modellben kell vizsgálni az összes termésadatot a megbízható becslés céljából. A termésadatok jelentősen függenek a bevetett területtől, így a következő módon határoztam meg a célváltozót: $Y = \text{termelés} / \text{bevetett terület}$. Az így előállt célváltozó már egységesen leírható egy közös modellben.

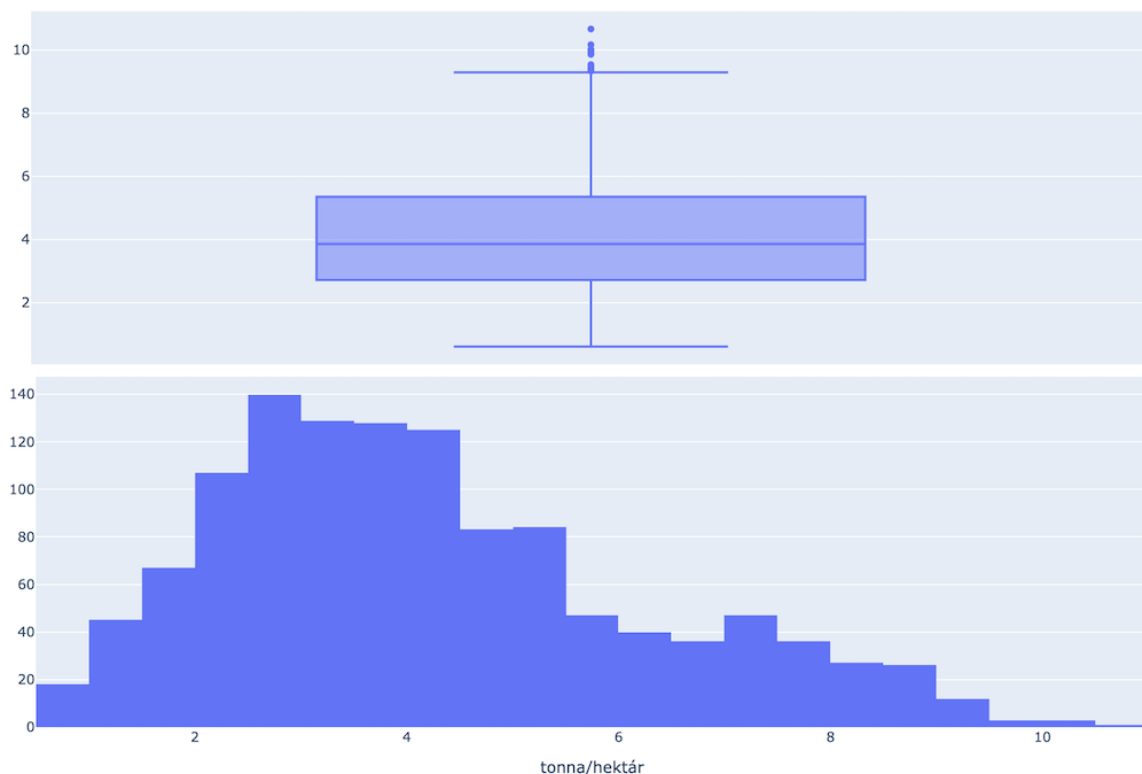
Mivel nem rendelkeztem egységes adatbázissal a búza és az időjárásadatok tekintetében, két különböző információ összekapcsolására volt szükséges. Az adatok feldolgozásához és kezeléséhez kialakításra került a python környezetben a Pandas nevű bővítmény, melynek segítségével könnyen elvégezhetők az adatátalakítások (pandas development team, 2020). Az időjárási adatok tekintetében országonkénti kimutatás állt rendelkezésre szezonális és havi átlagolt értékekkel. Mivel a búzatermésre vonatkozó adatok éves bontásban voltak elérhetőek az időjárás adatok kimutatásait át kellett alakítani idősoros kimutatásra, majd egységesíteni egy táblában országonkénti megjelöléssel. Az említett konverzió után Cantelaube és Terres (2005) által is alkalmazott módszer alapján a kiválasztottam a január és augusztus közötti időjárási adatokat és a havi átlagos értékeket külön változóként definiáltam minden évhez. A modellezés során úgy ítélt meg, hogy a decemberi információ javítja az előrejelzés hatékonyságát, illetve az ország mint kategória is ad többlet információt. Az említett változókat is előállítottam a modellezési adatbázisban. Így előállt összesen 12 változó esetén 9 időszak, mely összesen 108 magyarázó változót jelentett minden évhez. Végül az így rendelkezésre országonkénti időjárásadatokhoz hozzátudtam kapcsolni a célváltozót is, melyel előállt egy 1204 rekordból álló modellezési adatbázis. Az országok tekintetében az adatbázis az EU országait tartalmazta, viszont a becslések pontosítása érdekében Ukrajna adatait is hozzáadtam az adatbázishoz.

4.2. Az adatok leíró jellegű elemzése

Röviden szeretném bemutatni a magyarázó változók főbb statisztikai mutatóit, illetve a célváltozó felépítését, mielőtt a modellezési részre térnék. Összesen az éves aggregálás és a szűrések előtt 40656 darab időjárásadatot tartalmazó rekordom volt. Itt nem szeretnék túlságosan a részletekbe bocsátkozni, így inkább kiemelem az erős magyarázó erővel bíró változók főbb jellemzőit. A felhőtakaró változó 2,1 és 99,9 százalék között egy közel normális eloszlást vesz fel, melynek átlaga 64,66. A napi hőmérséklet tartomány minimuma 2,7 maximuma pedig 15,1. A csapadék mennyisége balra ferde jobbra elnyúló eloszlást követ, melynek minimuma 0, maximuma 365,1 mm/nap. A csapadék átlaga 62,17 mm/nap. Az átlag hőmérséklet minimuma -20,1 míg a maximuma 30,5 az átlaga pedig 9,67. A hőmérséklet tekintetében egy növekvő trend figyelhető meg. Az esős napok száma minimuma 0 a maximuma pedig 30,2 nap volt, míg az átlag 12,7. Jellemzően jól strukturált adatbázissal dolgoztam, ahol a változók esetén nem igazán jelentkeztek

kiugró értékek, illetve az eloszlás tekintetében is egészséges adatkészlet volt látható. Fontos kiemelni, hogy a potenciális evapotranspiráció esetén 1452 adatsor hiányzó értéket vett fel.

A célváltozó tekintetében a maximum érték 10,7 a minimum pedig 0,61 az átlag pedig 4,23, mely leolvasható a dobozábráról is. Láthatóan jelentős outlier érték nem figyelhető meg a célváltozó tekintetében. A célváltozó eloszlást és a dobozábrát az 1. ábrán tüntettem fel. Az eloszlás tekintetében inkább egy balraferde jobbra elnyúló alak figyelhető meg, melyről a dobozábra felső vége is árulkodik. A célváltozó interkvartilis terjedelme 2,64 tonna/hektár a szórás pedig 2,02.



1. ábra. A célváltozó eloszlása és dobozábrája

forrás: saját szerkesztés

4.3. Modellezési eredmények

Ahogy az adatbázis előállítása részben bemutattam létrehoztam egy modellezési táblát. A modellezést megelőzően két részre osztottam az adatot, egy tanító adatra, amelyen a modelleket betanítottam és egy teszt adatra, ahol a modellek performanciáját értékeltem ki. Az adatvágásnál 75%-ban definiáltam a tanító adatot, míg a tesztadat 25%-ot tett ki. A felosztáshoz a Scikit learn random kiválasztását használtam, tehát a vágás nem időrendben történt Pedregosa et al. (2011). Első körben a döntési fák hatékonyságát vizsgáltam meg. A hatékonyság mérését a módszertani részben részleteztem RMSE (root-mean-square error) segítségével hasonlítottam össze. Mivel a

kérdés egy regressziós probléma, így a jobb összehasonlíthatóság érdekében a teljes időszakon vett egy hektárra jutó búzatermés átlagával számolt RMSE értékét is vizsgáltam. Az átlagtól vett RMSE értéke 3,78 volt egységesen.

Első körben a döntési fák hatékonyságát vizsgáltam, ahol a tanító adaton elhanyagolható mértékű volt az RMSE értéke, viszont a teszt adaton 1,69-es RMSE értéket kaptam. Láthatóan a döntési modell az adatbázis segítségével jobban előre tudott már jelezni, viszont az eltérés így is nagynak mondható egy 3,8-as mediánnal és 2,6-os interkvartilis terjedelemmel rendelkező célváltozónál, így a hyperopt eljárás alkalmazása előtt indokoltnak tartottam egy fejlettebb módszer alkalmazását.

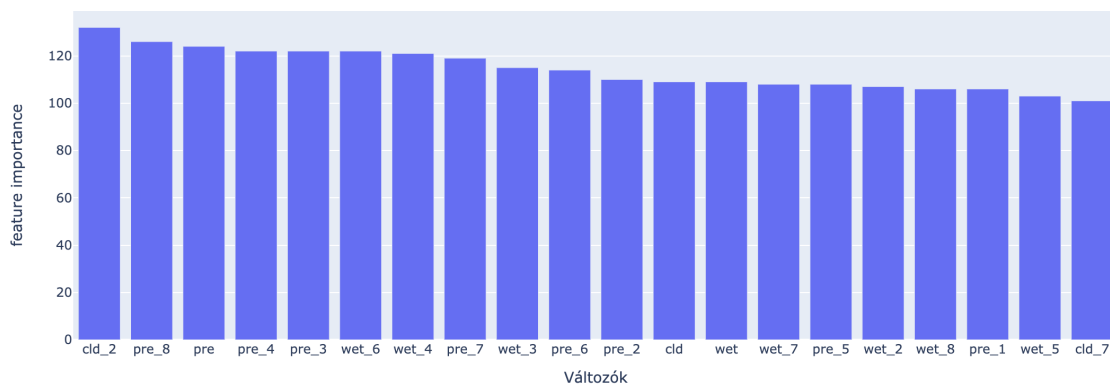
Paraméter	Érték
colsample_bylevel	0.7
colsample_bynode	0.8
colsample_bytree	0.7
eta	0.1
eval_metric	rmse
learning_rate	0.25
max_depth	9
min_child_weight	9
n_estimators	1600
random_state	42
seed	42
subsample	0.9

2. táblázat. A hyperopt által talált legjobb paraméterek.
forrás: saját szerkesztés

Következő lépésként az XGBoost algoritmust használtam a terméselőrejelzéshez. A tanító adat tekintetében ebben az esetben is elhanyagolható mértékű RMSE értéket kaptam, viszont a teszt adat esetén jelentős javulással 1,02-es értéket kaptam eredményként. Mivel a javulást jelentős áram az algoritmus bemeneti paraméterein nem változtattam érdemesnek találtam a hyperopt algoritmus alkalmazását is, mellyel tovább lehetett javítani a modell hatékonyságát. A hyperopt algoritmussal beállított paraméterek mellett az RMSE értéke 0,86-ra csökkent, melyet a végső legjobb modellként definiáltam a fa alapú modellek tekintetében. A hyperopt algoritmussal meghatározott paramétereket a 2. táblázatban mutatom be.

Az így előállt modell esetén persze fontos kérdés maradt még, hogy mely változók adták a legtöbb információt. Ehhez a legalkalmasabbnak a feature importance (jellemzők fontossága)

csomagot találtam, mivel döntési fákat vizsgáltam (Pedregosa et al., 2011). A legmagasabb 132-es értéket az februári felhőtakaró adja, ezt követi az augusztusi csapadék majd harmadik helyen pedig a decemberi csapadék áll. Láthatóan a legfontosabb jellemző szorosan összefüggnek a csapadék mennyiségével, mivel 100 fölötti értéket, kizárólag az esős napok száma, felhőtakaró és a csapadék mennyisége kapott. A fontossági jellemzők tekintetében a 80 és 100 pont közötti sávban az előbb említett két változó mellett erős magyarázó erővel bír a téli hónapok esetén a talaj menti fagy mértéke, illetve a májusi hőmérséklet tartomány. Az itt részletezett adatokat vizuálisan is ábrázoltam a 100 fölötti értékkel rendelkező változókat. Ezt a 2. ábrán mutatom be, ahol a változókhoz rendelt számok a vonatkozó hónapot jelölik (Ahol nincs szám az jelöli az előző évi decemberi hónapot).



2. ábra. Feature importance érték a változók esetén

forrás: saját szerkesztés

5. Összefoglalás

A dolgozatban az időjárási adatok alapján készítettem előrejelzést a búzatermésre. A modellezés során a faalapú modellezési technikákat vizsgáltam meg a feldolgozott szakirodalmak javaslatai alapján. Összeségében megállapítható, hogy érdemes kísérletezni a terméselőrejelzés kapcsán gépi tanulási módszerekkel, mivel jelentős javulás adható a modellek hatékonyságát tekintve. A modellezés során az 1204 rekordból álló táblában 75%-os tanító adaton készített modelleket egy 25%-os teszt mintán értékeltem ki. A teszt adatbázisban a teljes mintán számolt átlaggal az RMSE értéke 3,78 volt. Az egyszerű döntési fa modellel ezt 1,69-re lehet javítani, míg az XGBoost metódus 1,02-es értékig javult. A hyperopt algoritmussal elért paraméteroptyimalizálás segítségével az XGBoost modell hatékonysága 0,86-re javult, melyet a leghatékonyabb modellnek jelöltem meg. Az így kapott paraméterek beállításával javaslatot ad a dolgozat egy hatékony paraméter beállításra a terméselőrejelzés tekintetében. A legjobb változók az előrejel-

zés szempontjából főként a csapadék mennyiségével függenek össze. A legjobb pontszámot a jellemzők fontossága tekintetében a februári felhőtakaró kapta, viszont erős változó a decemberi, augusztusi, márciusi és áprilisi csapadék mennyisége, továbbá egységesen fontos az esős napok száma mind a nyári mind a tavaszi időszakokban.

Hivatkozások

- Bergstra, J., Yamins, D., és Cox, D. (2013). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning* (pp. 115–123).
- Borhegyi, P. (2020). *Történelem tankönyv 5*. Budapest: Oktatási Hivatal.
- Cantelaube, P., és Terres, J.-M. (2005). Seasonal weather forecasts for crop yield modelling in europe. *Tellus A: Dynamic Meteorology and Oceanography*, 57(3), 476–487.
- Chakraborty, S., Ghosh, R., Ghosh, M., Fernandes, C. D., Charchar, M. J., és Kelemu, S. (2004). Weather-based prediction of anthracnose severity using artificial neural network models. *Plant Pathology*, 53(4), 375–386.
- Chipanshi, A., Ripley, E., és Lawford, R. (1997). Early prediction of spring wheat yields in saskatchewan from current and historical weather data using the ceres-wheat model. *Agricultural and Forest Meteorology*, 84(3-4), 223–232.
- De Wolf, E., és Franel, L. (1997). Neural networks that distinguish infection periods of wheat tan spot in an outdoor environment. *Phytopathology*, 87(1), 83–87.
- Food and Agriculture Organization of United Nations. (2021). *Crops and livestock products*. Megtekintés időpontja: 2021-11-25. Hozzáférés: <https://www.fao.org/faostat/en/#data/QCL>
- Gong, Q., Huang, H., és Zhang, B. (2022). Grain yield prediction model based on the analysis of climate and irrigated area conditions in the wheat grain-filling period. *Irrigation and Drainage*.
- Harris, I., Osborn, T. J., Jones, P., és Lister, D. (2020). Version 4 of the cru ts monthly high-resolution gridded multivariate climate dataset. *Scientific data*, 7(1), 1–18.
- Hatfield, J. L. (2016). Increased temperatures have dramatic effects on growth and grain yield of three maize hybrids. *Agricultural & Environmental Letters*, 1(1), 150006.
- Kamir, E., Waldner, F., és Hochman, Z. (2020). Estimating wheat yields in australia using climate records, satellite image time series and machine learning methods. *ISPRS Journal of Photogrammetry and Remote Sensing*, 160, 124–135.
- Lobell, D. B., Schlenker, W., és Costa-Roberts, J. (2011). Climate trends and global crop production since 1980. *Science*, 333(6042), 616–620.
- Oikonomidis, A., Catal, C., és Kassahun, A. (2022). Hybrid deep learning-based models for crop yield prediction. *Applied artificial intelligence*, 1–18.
- pandas development team, T. (2020, February). *pandas-dev/pandas: Pandas*. Zenodo. Hozzáférés: <https://doi.org/10.5281/zenodo.3509134> doi: 10.5281/zenodo.3509134
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*,

12, 2825–2830.

- Prasad, P. V., Bheemanahalli, R., és Jagadish, S. K. (2017). Field crops and the fear of heat stress—opportunities, challenges and future directions. *Field Crops Research*, 200, 114–121.
- Srivastava, A. K., Safaei, N., Khaki, S., Lopez, G., Zeng, W., Ewert, F., ... Rahimi, J. (2022). Winter wheat yield prediction using convolutional neural networks from environmental and phenological data. *Scientific Reports*, 12(1), 1–14.
- Van der Velde, M., Baruth, B., Bussay, A., Ceglar, A., Garcia Condado, S., Karetsos, S., ... others (2018). In-season performance of european union wheat forecasts during extreme impacts. *Scientific reports*, 8(1), 1–10.
- Van Rossum, G., és Drake, F. L. (2009). *Python 3 reference manual*. Scotts Valley, CA: CreateSpace.
- xgboost developers. (2022). *Introduction to boosted trees*. Megtekintés időpontja: 2023-01-25. Hozzáféres: <https://xgboost.readthedocs.io/en/stable/tutorials/model.html>
- Yang, X., Yang, R., Ye, Y., Yuan, Z., Wang, D., és Hua, K. (2021). Winter wheat spad estimation from uav hyperspectral data using cluster-regression methods. *International Journal of Applied Earth Observation and Geoinformation*, 105, 102618.

A. Függelék

Az alábbi függelék tartalmazza azokat a kódokat, amelyeket az elemzések során futtattam. Az első kódrészlet bemutatja, hogy miként töltöttem le az online elérhető adatokat, amelyeket a további elemzésekhez használtam.

A második kódrészlet részletesen bemutatja, hogy az adatokat hogyan kellett átalakítanom ahhoz, hogy a modellezési eljárás végrehajtható legyen. Az adatok tisztítása és előkészítése kulcsfontosságú lépés volt, hiszen az adatok hibái, vagy az esetleges hiányzó értékek komoly hatással lehettek volna a modellezésre és az eredményekre. A kódban az átalakítások, például a formátumok módosítása, a hiányzó adatok kezelése és az egyéb szükséges transzformációk kerülnek bemutatásra.

A harmadik és egyben utolsó szekcióban az általam épített modelleket mutatom be. Itt részletesen ismertetem a modellek felépítését, a különböző algoritmusok alkalmazását és az egyes modellek teljesítményének kiértékelését. Végül a hyper-opt algoritmus alkalmazásával bemutatom, hogyan található meg az XGBoost paraméterei közül az a konfiguráció, amely a legjobb teljesítményt nyújtja a terméselőrejelzésre. Az optimalizálás folyamata kulcsfontosságú volt a legjobb eredmények eléréséhez, és segített abban, hogy a modellek a lehető leghatékonyabban végezzék az előrejelzést.

Alkalmazott példa kódok

1. Listing. Adatok letöltéséhez használt kód

```
import pandas as pd

import numpy as np
from pandas.api.types import CategoricalDtype
from plotnine import *
from plotnine.data import mpg
%matplotlib inline

df = pd.read_csv('wheat.csv')
df = df[["Area_Code_(M49)", "Area", "Year", "Value"]]
df.rename(columns={"Area_Code_(M49)": 'Area_code'}, inplace=True)
df.head()

wheat_grouped = df.groupby(['Area', 'Year'])[['Value']].sum()
#type(wheat_grouped)
wheat_grouped.head(100)
```

```

tmp = df.groupby(['Area', 'Year'], as_index=False)[['Value']].sum().
    pivot(index='Year', columns='Area', values='Value')
tmp['sum_production'] = tmp['Ukraine'] / tmp[tmp.columns].T.sum()
tmp['Ukraine_ratio'] = tmp['sum_production'] / tmp['Ukraine']
tmp.columns

def लेकरdez(es(variables, countries):

    var1 = variables[0]
    cnt1 = countries[0]
    data = pd.read_csv('https://crudata.uea.ac.uk/cru/data/hrg/cru_ts_4
        .06/crucy.2205251923.v4.06/countries/'+var1+'/crucy.v4
        .06.01.1901.2021.'+cnt1+'.'+var1+'.per',
        delimiter=r"\s+",
        skiprows= 3)
    nn = cnt1 + '_' + var1
    data = data.melt(id_vars=["YEAR"],
        var_name="month",
        value_name=nn)
    data["Date"] = data["YEAR"].astype(str) + data["month"]
    data = data[["Date", nn]]

    for varia in variables:
        for country in countries:
            if varia == "cld":
                url = 'https://crudata.uea.ac.uk/cru/data/hrg/cru_ts_4
                    .06/crucy.2205251923.v4.06/countries/'+varia+'/crucy.
                    v4.06.01.1901.2021.'+country+'.'+varia+'.per'
            else:
                url = 'https://crudata.uea.ac.uk/cru/data/hrg/cru_ts_4
                    .06/crucy.2205251923.v4.06/countries/'+varia+'/crucy.
                    v4.06.1901.2021.'+country+'.'+varia+'.per'
            print(url)
            df = pd.read_csv(url, delimiter=r"\s+", skiprows= 3)
            nn = country + '_' + varia
            df = df.melt(id_vars=["YEAR"],
                var_name="month",
                value_name=nn)
            df["Date"] = df["YEAR"].astype(str) + df['month']
            data [nn] = df[[nn]]

    return data

countries = ['Austria', 'Belgium', 'Bulgaria', 'Croatia', 'Cyprus', '
    Czech_Republic', 'Denmark',

```

```

        'Estonia', 'Finland', 'France', 'Germany', 'Greece', '
        Hungary', 'Ireland',
        'Italy', 'Latvia', 'Lithuania', 'Luxembourg', 'Malta', '
        Netherlands',
        'Poland', 'Portugal', 'Romania', 'Slovakia', 'Slovenia', '
        Spain', 'Sweden', 'Ukraine', 'Russia']
variables = ["cld", "dtr", "frs", "pet", "pre", "tmn", "tmp", "tmx", "
        vap", "wet"]

```

```

Weatherdata = lekerdezes(variables, countries)
Weatherdata.head()

```

2. Listing. Adatok transzformálásához használt kód

```

import pandas as pd
import numpy as np
from pandas.api.types import CategoricalDtype

weather = pd.read_csv('Weatherdata.csv')
countries = ['Austria', 'Belgium', 'Bulgaria', 'Croatia', 'Cyprus', '
        Czech_Republic', 'Denmark',
        'Estonia', 'Finland', 'France', 'Germany', 'Greece', '
        Hungary', 'Ireland',
        'Italy', 'Latvia', 'Lithuania', 'Luxembourg', 'Malta', '
        Netherlands',
        'Poland', 'Portugal', 'Romania', 'Slovakia', 'Slovenia', '
        Spain', 'Sweden', 'Ukraine'] # 'Russia'
variables = ["cld", "dtr", "frs", "pet", "pre", "tmn", "tmp", "tmx", "
        vap", "wet"]
weather.head()

```

```

weather = weather [~weather ["Date"].str.contains("DJF")]
weather = weather [~weather ["Date"].str.contains("MAM")]
weather = weather [~weather ["Date"].str.contains("JJA")]
weather = weather [~weather ["Date"].str.contains("SON")]
weather = weather [~weather ["Date"].str.contains("ANN")]

```

```

weather["Year"] = weather["Date"].str[0:4]
weather["Month"] = weather["Date"].str[4:7]
monthdict = {"JAN": "01", "FEB": "02",
        'MAR': "03", "APR": "04",
        "MAY": "05", "JUN": "06",
        "JUL": "07", "AUG": "08",
        "SEP": "09", "OCT": "10",

```

```

"NOV": "11", "DEC": "12"}

weather = weather.replace({"Month": monthdict})
weather["Date"] = weather["Year"] + weather["Month"]
#weather = weather.drop(columns=['Month', 'Year'])
weather["Date"] = pd.to_datetime(weather["Date"], format='%Y%m')
weather = weather.sort_values(by=['Date']).reset_index(drop=True)
weather = weather.drop('Unnamed: 0', axis=1)
weather.head()

wetvar = ["Date", 'country'] + variables
weather2 = pd.DataFrame(columns = wetvar)
for country in countries:
    weathertech = pd.DataFrame()
    weathertech["Date"] = weather["Date"]
    weathertech.loc[:, 'country'] = country
    for var in variables:
        name = country+"_"+var
        weathertech[var] = weather[name]
    #weather2 = weather2.append(weathertech, ignore_index = True)
    weather2 = pd.concat([weather2, weathertech])
weather2.head()
weather2.to_csv("visual.csv", index=False)

dfs = []
for name, group in weather2.groupby('country'):
    for step in range(1, 9):
        for var in variables:
            group[f'{var}_{step}'] = group[var].shift(step)
        dfs.append(group)
weatherfinal = pd.concat(dfs)
weatherfinal["Date"] = weatherfinal["Date"].shift(9)
#weatherfinal.head(12)

weatherfinal['month'] = weatherfinal['Date'].dt.month
weatherfinal= weatherfinal[(weatherfinal['Date'] >= '1961-08-01')]
weatherfinal= weatherfinal[(weatherfinal['Date'] < '2021-01-01')]
weatherfinal= weatherfinal.reset_index(drop=True)
weatherfinal = weatherfinal[weatherfinal['month'] == 8]

df = pd.read_csv('wheat.csv')
df = df[["Area_Code_(M49)", "Area", "Year", "Value"]]
df.rename(columns={"Area_Code_(M49)": 'Area_code'}, inplace=True)
wheat_grouped = df.groupby(['Area', 'Year'], as_index=False)[['Value']].

```

```

        sum()
wheat_grouped.loc[wheat_grouped["Area"] == "Czechia", "Area"] = '
    Czech_Republic'
wheat_grouped["key"] = wheat_grouped['Area'] + "_" +wheat_grouped['
    Year'].astype(str)
wheat_grouped

cl = wheat_grouped['Area'].unique()
df_ah = pd.read_csv('area_harvested.csv')
df_ah = df_ah[df_ah["Area"].isin(cl)]
df_ah = df_ah[["Area_Code(M49)", "Area", "Year", "Value"]]
df_ah.rename(columns={"Area_Code(M49)": 'Area_code'}, inplace=True)
df_ah = df_ah.groupby(['Area', 'Year'], as_index=False)[['Value']].sum()
df_ah.loc[df_ah["Area"] == "Czechia", "Area"] = 'Czech_Republic'
df_ah["key"] = df_ah['Area'] + "_" +df_ah['Year'].astype(str)
df_ah.rename(columns={'Value': 'area_harvested'}, inplace=True)
df_ah.head(100000)

joined_wheat = wheat_grouped.merge(df_ah[[' area_harvested', 'key']],
    how = 'left',
        left_on = 'key', right_on = 'key')
joined_wheat["target"] = joined_wheat["Value"] / joined_wheat["
    area_harvested"]
joined_wheat.head()

weatherfinal["key"] = weatherfinal["country"] + "_" +weatherfinal["Date"
    ].dt.year.astype(str)
weatherfinal

tablefinal = weatherfinal.merge(joined_wheat[[' target', 'key']], how = '
    left',
        left_on = 'key', right_on = 'key')
tablefinal

tablefinal = tablefinal.drop(['Date', 'month', 'key'], axis=1)
print(len(tablefinal))

tablefinal = tablefinal.dropna()
print(len(tablefinal))

for var in tablefinal.columns:
    tablefinal = tablefinal[tablefinal[var] != -999 ]
print(len(tablefinal))
tablefinal.to_csv("modeling_table.csv", index=False)

```

3. Listing. Modellezési eljárások kódjai

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from xgboost import XGBRegressor
import xgboost as xgb
from sklearn.metrics import mean_squared_error, r2_score
from xgboost import plot_tree
import matplotlib.pyplot as plt
import graphviz
import numpy as np
from functools import partial
from hyperopt import STATUS_OK, Trials, fmin, hp, tpe, space_eval
from sklearn.preprocessing import RobustScaler, StandardScaler
from sklearn.pipeline import Pipeline
from IPython.display import clear_output
from sklearn import preprocessing

table = pd.read_csv('modeling_table.csv')

le = preprocessing.LabelEncoder()
le.fit(table["country"])
table["country_encoded"] = le.transform(table["country"])

X = table.drop(['target', 'country'], axis=1)
y = table['target']

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0,
                                                    test_size=0.25)
len(table)

dt_model = DecisionTreeRegressor()
dt_model.fit(X_train, y_train)

m = dt_model
pred_train = m.predict(X_train)
tr_rmse = mean_squared_error(y_train, pred_train)

predictions = m.predict(X_test)
ts_rmse = mean_squared_error(y_test, predictions)

y_mean = [table["target"].mean()] * len(y_test)
me = mean_squared_error(y_test, y_mean)
```

```

print("train_rmse:", tr_rmse)
print("test_rmse:", ts_rmse)
print("mean_rmse:", me)

xgb_model = XGBRegressor()
xgb_model.fit(X_train, y_train)

m = xgb_model
pred_train = m.predict(X_train)
tr_rmse = mean_squared_error(y_train, pred_train)

predictions = m.predict(X_test)
ts_rmse = mean_squared_error(y_test, predictions)

y_mean = [table["target"].mean()] * len(y_test)
me = mean_squared_error(y_test, y_mean)

print("train_rmse:", tr_rmse)
print("test_rmse:", ts_rmse)
print("mean_rmse:", me)

def objective(space, data, optimizer_metrics):
    X_train, _, y_train, _ = data
    data_dmatrix = xgb.DMatrix(data=X_train, label=y_train)
    xgb_cv = xgb.cv(dtrain=data_dmatrix, params=space, nfold=5, metrics=
        optimizer_metrics)
    metric = 'test-mae-mean' if optimizer_metrics == 'mae' else 'test-
        rmse-mean'
    score = np.min(xgb_cv[metric]) # ez lehet, hogy nem lesz j , mert
        a metric n v nem tuti, csekkold le k l n , hogy mi lesz az xgb.
        cv kiment ben!
    return {'loss': score, 'status': STATUS_OK}

space = {
    'booster': 'gbtree',
    'max_depth': hp.choice('max_depth', np.arange(5, 12, 1, dtype=int)),
    'n_estimators': hp.choice('n_estimators', np.arange(700, 2000, 100,
        dtype=int)),
    'learning_rate': hp.quniform('learning_rate', 0.05, 0.25, 0.05),
    'colsample_bytree': hp.quniform('colsample_bytree', 0.5, 0.9, 0.1),
    'colsample_bynode': hp.quniform('colsample_bynode', 0.5, 0.9, 0.1),
    'colsample_bylevel': hp.quniform('colsample_bylevel', 0.5, 0.9, 0.1)
    ,

```

```

    'min_child_weight': hp.choice('min_child_weight', np.arange(4, 12,
        1, dtype=int)),
    'subsample': hp.quniform('subsample', 0.5, 0.9, 0.1),
    'eta': hp.quniform('eta', 0.1, 0.3, 0.1),
    'eval_metric': 'rmse',
    'seed': 42,
    'random_state': 42
}

fmin_objective = partial(objective, data=(X_train,X_test, y_train,
    y_test), optimizer_metrics='rmse')
best_hyper = fmin(fn=fmin_objective,
    space=space,
    algo=tpe.suggest,
    max_evals=60,
    trials=Trials())

best = space_eval(space, best_hyper)
clear_output()
print(best)

xgd_model_hpt = xgb.XGBRegressor(**best)
xgd_model_hpt.fit(X_train, y_train,
    eval_set=[(X_train, y_train), (X_test, y_test)],
    eval_metric='rmse', verbose= 0)

m = xgd_model_hpt
pred_train = m.predict(X_train)
tr_rmse = mean_squared_error(y_train, pred_train)

predictions = m.predict(X_test)
ts_rmse = mean_squared_error(y_test, predictions)

y_mean= [table["target"].mean()] * len(y_test)
me = mean_squared_error(y_test, y_mean)

print("train_rmse: ",tr_rmse)
print("test_rmse: ",ts_rmse)
print("mean_rmse: ",me)

import plotly.express as px
import operator
feature_importance = xgd_model_hpt.get_booster().get_score(
    importance_type='weight')

```

```

N = 200
feature_importance = dict(sorted(feature_importance.items(), key=
    operator.itemgetter(1),reverse=True))
feature_importance = dict(list(feature_importance.items())[0: N])
vars_imp = [x for x in feature_importance]
vals_imp = [feature_importance[x] for x in vars_imp]
the_dict = {'Valtozok': vars_imp, 'feature_importance': vals_imp}
fig = px.bar(the_dict, x='Valtozok', y='feature_importance')
fig.show()

```