Python projektfeladat

Varsányi Barnabás
DRUPS4

2025

Bevezetés

A haladó programozás laborgyakorlatára készített, az operációs rendszert monitorozó alkalmazás elsősorban felügyeleti célokra készült, egyszerű gépi tanuláshoz köthető modellt alkalmazva.

Az IsolationForest hozzájárul, hogy a program futtatása közben, a számítógép használati adataiból a modellt feltanítva, egy érték kerül kiértékelésre, amely egyfajta indikátorként szolgál a kihasználtság tekintetében. A beállítható időköz túllépése esetén a modell újratanulást alkalmaz, így elkerülhető a folyamatos riasztás, állandósult terhelés esetén.

A szoftver futtatásához szükség van a Python környezetére, illetve a requirements szöveges állományban található kiegészítő könyvtárak telepítésére, amelyet a python -m pip install -r requirements.txt utasítással egyszerűen elvégezhető.

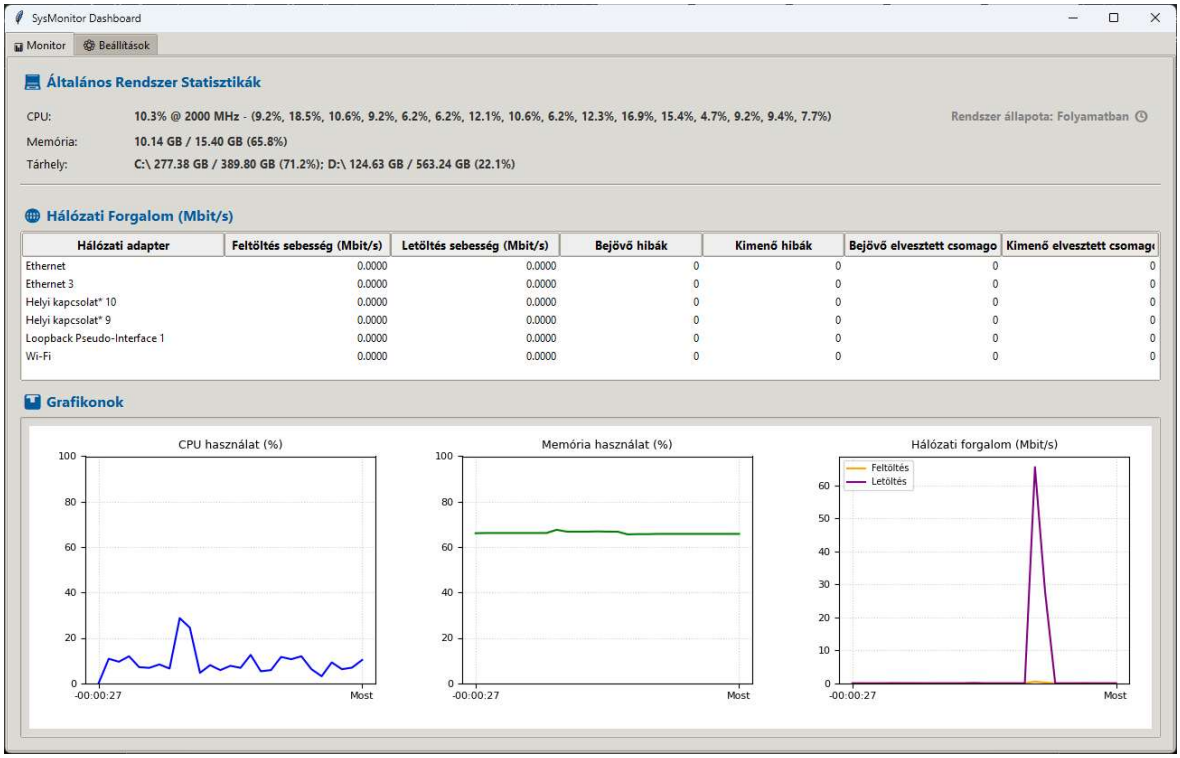
A telepítés végeztével a main.py állományt kell futtatni, amely a futtatókörnyezet beállításától függően, de általában a python main.py utasítással lehet elindítani.

A projekt verziókövetése elérhető nyilvánosan a Github felületén, az alábbi hivatkozáson keresztül: https://github.com/varsanyib/halaprog-la-sysmonitor

A Tkinter által kialakított grafikus felület implementálásában a Perplexity AI segített.

1. Felhasználói dokumentáció

A program elindítása a bevezetőben leírtak alapján történik. Az utasítás kiadása után megtörténik a grafikai felület összeállítása, létrejönnek háttérben a rendszerfolyamatok.

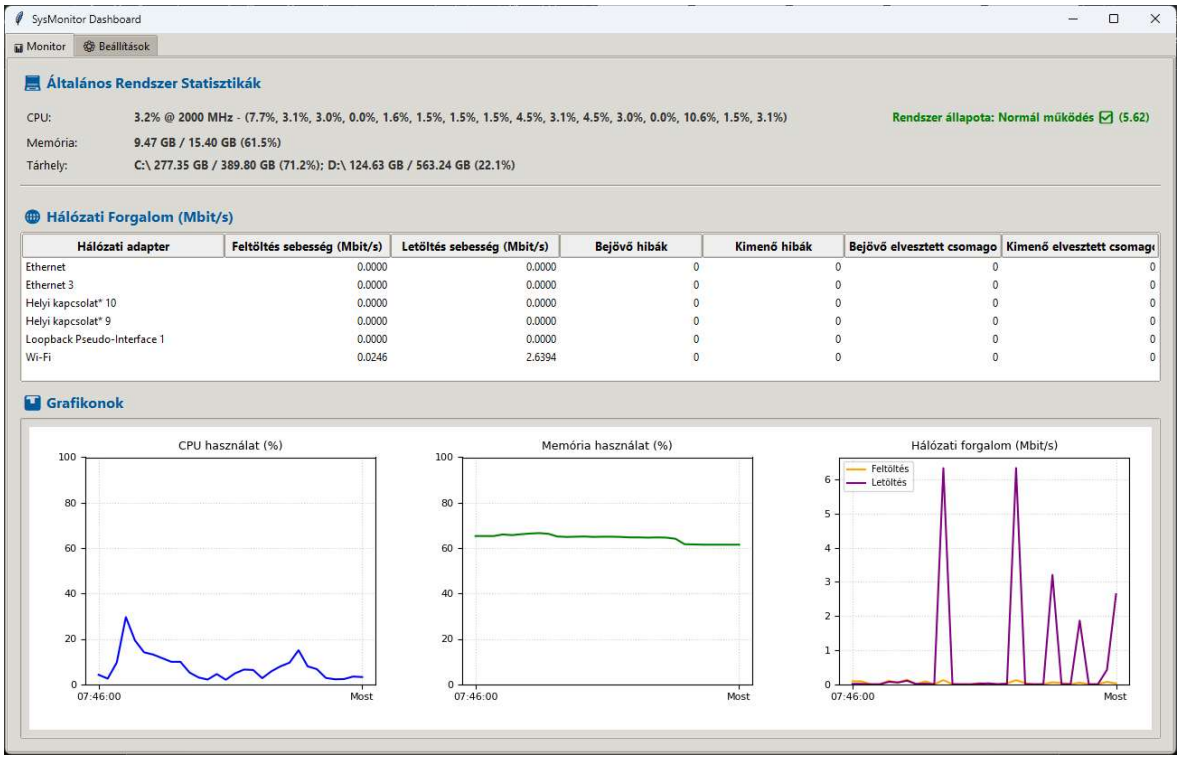


1. ábra: Indítás utáni állapot

A felületet 3 + 1 szekcióra lehet felosztani. Az első tartalmazza az általános adatokat, többek között a CPU kihasználtságát, aktuális órajelét, illetve az egyes szálak terheltségét, százalékos formátumban. A memóriánál mindig az aktuálisan kihasználtság kerül megjelenítésre, a gyorsítótár, swap értékek nem számítanak bele a mutatóba. A tárhely adatok Windows operációs rendszeren az aktuális partíciók alapján jelennek meg, Linux/Mac OS esetében az aktuális mountpoint-ok jelennek meg.

A második részben található a hálózati forgalmat részletező táblázat. Itt a fizikai és szoftveres adapterek egyaránt megjelenítésre kerülnek, amelyek egyenként mutatják meg az aktuális letöltési és feltöltési sebességet megabit / sec mértékegységgel.

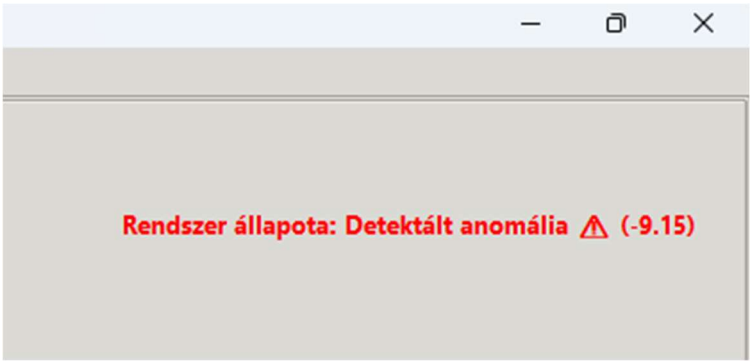
Az aktuális használatot mutató grafikonok a harmadik részben kaptak helyet, ahol a beállításokban beállítható, hogy egyszerre mekkora intervallumon jelenítse meg az adatokat. A hálózati forgalom grafikonja az adapterek kumulált értékeiből kerülnek kiszámításra.



2. ábra: Hosszabb ideig történő futtatás után

A jobb felső sarokban található rendszer állapota jelzi az IsolationForest által detektált anomália értékét. A kihasználtsági adatokból a modell feltanításra kerül és egy érték kerül kiszámításra. Ha az erőforrások kihasználtsága megváltozik (például egy nagyobb rendszerigényű szoftver indul el), akkor megváltozik a fent megjelenített érték. Ha az érték eléri a beállított küszöböt, a rendszer állapota detektált anomáliára vált át.

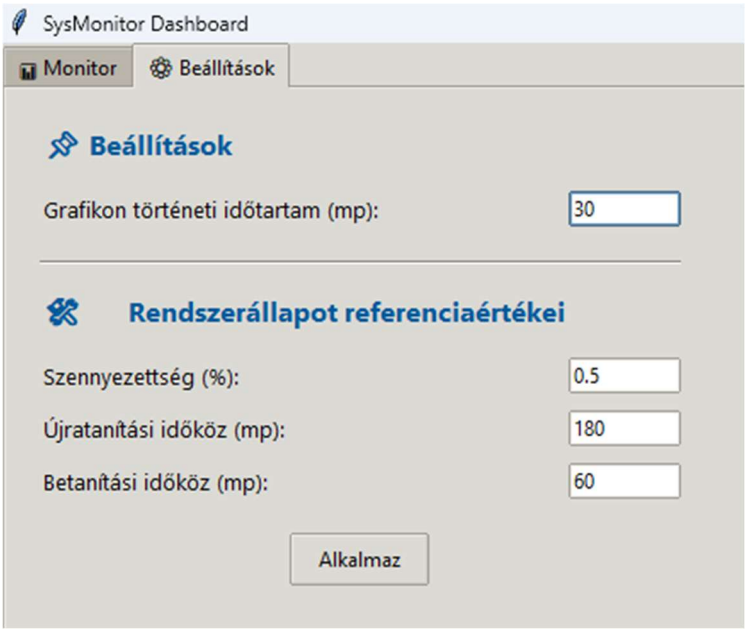
Mivel a felhasználói szokások megváltozhatnak a rendszerben (például huzamosabb ideig futtatásra kerül egy nagyobb rendszerigényű szoftver), akkor állandó anomália lenne detektálható, viszont a modell folyamatos finomhangolása ezt nem teszi lehetővé. Tehát ez esetben a szoftver a hirtelen kiugró tüskéket veszi figyelembe, nem feltétlenül a magas terhelést.



3. ábra: Detektált anomália hibajelzése

A beállítások menüpontban lehetőség nyílik a grafikon intervallumát beállítani másodperces pontossággal. Alapértelmezetten az az érték 30 másodpercre van beállítva a program indításakor. Ezen kívül az anomália detektálását végző modell értékein lehet módosítani.

A szennyezettség a modell esetében az eltérő adatok vizsgálatában hajt végre változást. Minél magasabb az érték, annál szigorúbb a modell az anomáliák detektálásában, mivel a legkisebb változást is szigorúan azonosítja. Alapértelmezetten az érték 0,5%, de egészen 50%-ig növelhető. Az újratanítási időközzel konfigurálható, hogy milyen időszakonként kerüljön az aktuális értékek segítségével a modell finomításra. A betanítási időköz a szoftver indítása utáni értékek gyűjtését végzi el. A beállított időközig nincs detektálás, addig kizárólag csak referenciaadat gyűjtése történik meg.



4. ábra: Beállítások menüpont

2. Technikai dokumentáció

2.1. Futtatandó állomány

A main.py az anomáliadetektáló alkalmazás belépési pontja, amely összekapcsolja a háttérben futó rendszerfigyelő komponenst a grafikus felülettel. Ebben a fájlban jön létre a SysMonitor példány, amely adott időközönként rendszererőforrásokat mér, és ezeket az adatokat az anomáliák észleléséhez használja fel.

A monitor egy külön, daemon módban futó szálban működik, így a folyamatos adatgyűjtés és kiértékelés nem blokkolja a tkinter alapú GUI válaszkészségét. A main.py felelős a fő ablak létrehozásáért és a SysMonitorGUI inicializálásáért is, amely a monitor által szolgáltatott adatokat jeleníti meg, illetve vizuálisan jelzi a detektált rendellenességeket.

A fájlban definiált bezárási logika gondoskodik róla, hogy az alkalmazás leállításakor a monitor szabályosan megálljon, a háttérszál befejeződjön, majd csak ezután záródjon be a grafikus felület. Emellett a billentyűzetről érkező megszakításokat is úgy kezeli, hogy váratlan leállítás esetén se maradjanak futó szálak vagy félbemaradt műveletek. Így a main.py a teljes alkalmazás életciklusának központi irányítója: elindítja a monitorozást és a GUI-t, biztosítja azok együttműködését, majd felügyeli a rendezett leállítást.

2.2. Adatgyűjtő komponsens

A SysMonitor osztály a rendszerfigyelés és az anomáliadetektálás adatgyűjtő komponense. Feladata, hogy meghatározott időközönként részletes erőforrás-statisztikákat olvasson ki a rendszerről, és ezeket időbélyeggel ellátva eltárolja a későbbi kiértékeléshez és megjelenítéshez. Az inicializáláskor egy belső adattárolót hoz létre a minták számára, beállítja a futási intervallumot, valamint egy logikai jelzővel követi, hogy a monitor fut-e még. Ez az osztály tipikusan egy külön szálban működik, és folyamatosan szolgáltatja az anomáliadetektálás alapjául szolgáló nyers adatokat.

A collect_data metódus egyetlen hívás során átfogó képet készít a rendszer aktuális állapotáról. A psutil könyvtár segítségével lekérdezi az összesített CPU-kihasználtságot és a szálankénti terhelést, valamint az aktuális CPU frekvenciát, ami lehetővé teszi a processzor terhelési mintázatainak elemzését.

A memóriahasználat esetén a teljes és a felhasznált kapacitás, illetve a kihasználtsági százalék kerül rögzítésre gigabájtban normalizált formában, ami segíti a memóriaintenzív anomáliák azonosítását. A háttértáraknál egyrészt egy alapértelmezett partíció összes, használt és százalékos kihasználtsága, másrészt az összes elérhető, fájlrendszerrel

rendelkező partíció adatai kerülnek összegyűjtésre, beleértve a csatolási pontot, a fájlrendszer típusát és a részletes használati statisztikákat. Ez a részletes disk_usages struktúra lehetővé teszi, hogy konkrét kötetekhez, mountpontokhoz köthető rendellenes lemezhasználatot is észlelni lehessen.

A hálózati statisztikák gyűjtése két időpont között mért különbségekből történik. A komponens először rögzíti az egyes hálózati interfészek forgalmi számlálóit, majd egy másodperc várakozás után újra leolvassa ezeket. A két mérés különbségéből számítja ki az adott idő alatt elküldött és fogadott bájtok mennyiségét, amelyet megabájt alapon, majd bit/szerű értékké, azaz megabit per másodpercben kifejezett feltöltési és letöltési sebességgé alakít. Emellett interfészenként rögzíti a bejövő és kimenő hibák, valamint az eldobott csomagok számának változását is. Ezek az értékek különösen fontosak a hálózati anomáliák, például hirtelen forgalomtüskék, túlterhelések vagy instabil kapcsolatok azonosításánál. A collect_data minden adatgyűjtési ciklus végén egy strukturált dictionary-t ad vissza, amely a CPU, memória, lemez és hálózati metrikákat egységes formában tartalmazza, hibák esetén pedig egy hibaüzenetet adó szerkezetet szolgáltat.

A run metódus egy folyamatosan futó ciklus, amely mindaddig végzi az adatgyűjtést, amíg a running jelző igaz. Minden iterációban új mintát vesz, az eredményt egy időbélyeggel együtt eltárolja a belső adatlistában, így időben rendezett idősor jön létre a rendszer állapotáról. A ciklusban alkalmazott várakozás úgy van beállítva, hogy figyelembe vegye a collect_data által már felhasznált egy másodperces hálózati mérési szünetet, így a teljes mintavételi periódus közelíti a megadott futási intervallumot. Ez lehetővé teszi, hogy a monitor egyenletes ütemben, szabályos időközökkel szolgáltasson adatpontokat, amelyek később grafikonokon, riasztási logikákban vagy anomáliakereső algoritmusokban használhatók fel. A stop metódus egyszerűen leállítja a ciklust azzal, hogy a running jelzőt hamisra állítja, ezáltal biztosítva a monitorozás szabályos, kívülről vezérelhető befejezését.

2.3. Gépi tanulás

Az anomaly.py fájl az anomáliadetektáló rendszer gépi tanulási rétegét valósítja meg. Az AnomalyDetector osztály az IsolationForest algoritmust használja, amely felügyelet nélküli módon tanulja meg, milyen értéktartomány tekinthető normálisnak a rendszer terhelése szempontjából, majd ehhez képest méri az aktuális állapot rendhagyóságát. Az inicializálás során létrejön az IsolationForest modell, amelynek érzékenységét a szennyezettségi paraméter határozza meg, valamint beállításra kerül egy minimális mintaszám, amely alatt a modell nem kezd tanulásba, hogy elkerülje a túl kevés adatból

adódó megbízhatatlan eredményeket. Az osztály belső állapottal jelzi, hogy megtörtént-e már a tanítás, így a rendszer biztonságosan ellenőrizni tudja, mikor használhatóak az anomália pontszámok.

A train metódus a monitor által gyűjtött idősoros adatok listáját várja bemenetként. Ezen adatlista elemeiből egy táblázatos adatszerkezet készül, majd ebből két, az anomáliadetektálás szempontjából kulcsfontosságú jellemző kerül kiválasztásra: a CPU kihasználtság és a memóriahasználat százalékos értéke. Ezek a mutatók jól tükrözik a rendszer terhelését, így a modell megtanulhatja, milyen kombinációk fordulnak elő tipikusan normál működés közben. Ha a mintaszám eléri a minimális küszöböt, az IsolationForest ezen jellemzők alapján illesztésre kerül, és az osztály jelzi, hogy a modell készen áll a használatra. Sikertelen, túl kevés adattal próbált tanítás esetén a metódus jelzi, hogy a modell továbbra sem tekinthető betanítottnak.

A predict_anomaly_score metódus egy aktuális mérési pillanat adataihoz számít anomália pontszámot. Amennyiben a modell még nem lett betanítva, védelmi mechanizmusként semleges, nulla értéket ad vissza, elkerülve a téves riasztásokat. Betanított állapotban a metódus az aktuális CPU- és memória százalékos értékből jellemzővektort képez, majd az IsolationForest döntési függvényét használja az anomália pontszám meghatározására. Ez az érték folytonos skálán fejezi ki, mennyire szokatlan az adott állapot a korábban tanult normál mintákhoz képest, ahol az alacsonyabb pontszámok nagyobb valószínűségű rendellenességre utalnak.

2.4. Grafikus felület

A gui.py fájl az anomáliadetektáló rendszer felhasználói felületét valósítja meg, amely egy modern, tabos struktúrájú tkinter alkalmazásként jelenik meg. A SysMonitorGUI osztály inicializálásakor megkapja a fő ablakot és a monitorozó komponenst, majd beállítja a történelmi adatok tárolását, például a CPU, memória és hálózati forgalom idősorait egy listában. Az anomáliadetektálás integrációja is itt történik: létrehoz egy AnomalyDetector példányt finomhangolt paraméterekkel, mint a szennyezettségi küszöb, az újratanítási ciklus és a minimális betanítási mintaszám, amelyek a rendszer normál viselkedésének tanulását szabályozzák. A fő ablak konfigurálása során egy 1280x720-as méretezett felületet állít be, míg a stílusok definiálása egységes, professzionális megjelenést biztosít címkéknek, fejlécnek és táblázatoknak, kék és szürke tónusokkal hangsúlyozva a fontos információkat.

A felület két fülre oszlik: a Monitor fül a valós idejű adatokat mutatja, míg a Beállítások fül a paraméterek módosítását teszi lehetővé.

A dashboard létrehozása során egy fő keretbe szerveződnek a statisztikák és a grafikonok szakaszai, amelyek rácsszerű elrendezésben kapnak helyet, biztosítva a skálázhatóságot. A statisztikák részében címkék jelenítik meg a CPU terhelését magonkénti bontásban, a memória használatát gigabájtban és százalékban, valamint a tárhelypartíciók részletes kihasználtságát. Az anomália állapotot egy dinamikus címke jelzi, amely a detektált rendellenesség esetén piros színnel és figyelmeztető ikont, normál állapotban pedig zölddel jelzi a pontszámot. A hálózati forgalom egy rendezett táblázatban jelenik meg, ahol az adapterek neve, a feltöltési és letöltési sebességek megabit per másodpercben, valamint a hibák és eldobott csomagok száma sorolódik fel, ABC sorrendben rendezve a könnyebb áttekinthetőség érdekében.

A grafikonok szakasza matplotlib integrációval működik: egy széles alakzatot hoz létre három alrajzzal, amelyek a CPU, memória és hálózati forgalom idősorait ábrázolják. A CPU és memória grafikonok százalékos skálán, 0-100 közötti tengellyel mutatják a legutóbbi 30 mintát (alapértelmezett beállítás alapján), míg a hálózati ábra a feltöltést és letöltést külön vonalakkal jelzi, alsó korláttal a jobb láthatóság érdekében. Az időtengely egyszerűsítve csak a kezdőpontot és a mostani időt jelöli, relatív időbélyegekkel, rácsos háttérrel a trendek követéséhez. Az update_data metódus ezredmásodpercenként frissíti az összes elemet: ellenőrzi a betanítási feltételeket, kiszámítja az aktuális anomália pontszámot, feldolgozza a történelmi adatokat, frissíti a statisztikákat, a hálózati táblázatot, majd újrarajzolja a grafikonokat, így biztosítva a folyamatos, valós idejű visszajelzést.

A beállítások fülön a felhasználó módosíthatja a grafikonok történelmi hosszát, a szennyezettségi arányt százalékban, az újratanítási intervallumot és a minimális betanítási mintaszámot. Az alkalmaz gomb validálja a bemeneteket, például pozitív egész számokat vár a hosszra és intervallumokra, valamint 0-50% közötti értéket a szennyeződésre, majd sikeres esetben újrainicializálja a detektort és üzenettel erősíti meg a változást. Hibás értékeknél részletes hibaüzenet jelenik meg, megakadályozva a rendszer instabilitását. Az anomália címke frissítése dinamikusan változtatja a szöveget és színt a pontszám alapján, ahol negatív értékeknél anomáliaként jelöli meg az állapotot.