A mesterséges intelligencia alkalmazása a szoftvertesztelés automatizálásában

Artificial intelligence in software testing automation

Dr. HORNYÁK Olivér

Miskolci Egyetem, Informatikai Intézet
3515 Miskolc-Egyetemváros, Egyetem út 1., oliver.hornyak@uni-miskolc.hu

Abstract

The article presents the role of artificial intelligence (AI) in software testing automation. AI technologies, such as machine learning and natural language processing, help automate test case generation and increase the efficiency of testing processes. While AI offers numerous advantages, its implementation also comes with challenges. The article aims to provide an overview of how AI could shape the future of software testing practices and improve the quality and reliability of software.

Keywords: Artificial intelligence, software testing automation, machine learning, software testing, test case generation

Kivonat

A cikk bemutatja a mesterséges intelligencia (MI) szerepét a szoftvertesztelés automatizálásában. Az MI technológiák, mint a gépi tanulás és a természetes nyelvfeldolgozás, segíthetik a tesztesetek automatikus generálását és a tesztelési folyamatok hatékonyságának növelését. Bár az MI számos előnyt kínál, bevezetése kihívásokkal is jár. A cikk célja, hogy képet adjon arról, hogyan formálhatja az MI a jövő szoftvertesztelési gyakorlatait, és hogyan növelheti a szoftverek minőségét és megbízhatóságát.

Kulcsszavak: mesterséges intelligencia, szoftvertesztelés automatizálása, gépi tanulás, szoftver tesztelés, teszteset generálás

1. BEVEZETÉS

Az elmúlt évtizedekben a szoftverfejlesztés nagy fejlődésen ment keresztül, és ennek megfelelően a szoftvertesztelés is egyre fontosabbá vált. Ahogy a szoftverek komplexitása növekszik, a hibák felderítése és kijavítása is egyre nehezebbé és időigényesebbé válik. A hagyományos tesztelési módszerek, mint például a manuális tesztelés, már nem mindig felelnek meg a mai gyors tempójú, folyamatos integrációt és fejlesztést követelő környezeteknek. Ezért a tesztelés automatizálása kulcsszereplővé vált a modern szoftverfejlesztési ciklusokban.

Az automatizálás azonban önmagában nem elég a szoftverek egyre bonyolultabb működésének teljes lefedésére. Itt jön képbe a mesterséges intelligencia alkalmazása, amely új lehetőségeket nyit meg a tesztelési folyamatok hatékonyabbá tételére. Az MI technológiák, például a gépi tanulás, a természetes nyelvfeldolgozás, a prediktív analitika, olyan eszközöket biztosítanak, amelyek képesek nagy mennyiségű adat elemzésére, hibák előrejelzésére, valamint tesztesetek automatikus generálására. Az MI-alapú szoftvertesztelés különösen vonzó lehet a vállalatok számára, mivel lehetőséget nyújt a tesztelési idő csökkentésére, a tesztlefedettség növelésére, valamint a tesztelési költségek optimalizálására. Ugyanakkor számos kihívás is felmerül az MI alkalmazásával kapcsolatban, például az adatminőség kérdése, az MI modellek tréningjének komplexitása, valamint a mesterséges intelligencia és az emberi tesztelők közötti egyensúly megtalálása.

Ez a tanulmány arra törekszik, hogy feltárja a mesterséges intelligencia alkalmazásának lehetőségeit és kihívásait a szoftvertesztelés automatizálásában, különös tekintettel arra, hogyan alakíthatja át az MI a jövő szoftverfejlesztési és tesztelési folyamatait.

2. A SZOFTVERTESZTELÉS SZEREPE

A szoftvertesztelés a szoftverfejlesztés egyik legkritikusabb fázisa, amely során a fejlesztők és tesztelők biztosítják, hogy a szoftver megfeleljen a meghatározott követelményeknek, és hibamentesen működjön. Az elmúlt évtizedekben számos kutatás és fejlesztés foglalkozott a szoftvertesztelés módszereinek és eszközeinek fejlesztésével, beleértve a manuális és automatizált tesztelési technikákat is.

A manuális tesztelés a szoftvertesztelés egyik legrégebbi formája, amely során a tesztelők manuálisan végrehajtanak különböző teszteseteket annak érdekében, hogy azonosítsák a hibákat. Ez a módszer nagyfokú emberi erőforrást igényel, és időigényes lehet, különösen nagy méretű rendszerek esetében [1]. Bár a manuális tesztelés még mindig fontos szerepet játszik egyes területeken, például a felhasználói élmény tesztelésében, az automatizálás egyre nagyobb figyelmet kap.

Az automatizált tesztelés fő iránya a szoftvertesztelési folyamatok hatékonyságának növelésére irányult, különösen olyan környezetekben, ahol gyakoriak a változások és frissítések, mint például az agilis fejlesztési módszertanokban. Az automatizált tesztelési eszközök lehetővé teszik a tesztek gyors lefuttatását, valamint a tesztelési eredmények automatizált elemzését. Az automatizálás hozzájárulhat a tesztelési idő csökkentéséhez és a hibák gyorsabb felismeréséhez [2]. Ennek ellenére az automatizált tesztelés is rendelkezik bizonyos korlátokkal, például a kezdeti bevezetés magas költségeivel és azzal, hogy csak bizonyos típusú tesztek automatizálhatók hatékonyan [3].

Az elmúlt években egyre nagyobb figyelem irányult a mesterséges intelligencia (MI) alkalmazására a szoftvertesztelés automatizálásában. A gépi tanulás, képes a tesztadatok elemzésére, hibák előrejelzésére és tesztesetek automatikus generálására [4]. Az MI-alapú rendszerek képesek a tesztelési eredmények automatikus kiértékelésére, az anomáliák azonosítására, valamint a szoftver teljesítményének folyamatos monitorozására [5]. A mesterséges intelligencia integrációja a tesztelési folyamatokba új lehetőségeket teremt a szoftverhibák hatékonyabb felismerésére és megelőzésére. Az MI alkalmazása a szoftvertesztelésben nem csupán az automatizálást könnyíti meg, hanem a tesztelés prediktív aspektusait is támogatja. A prediktív analitika lehetővé teszi a jövőbeni hibák és teljesítményproblémák előrejelzését, ezáltal minimalizálva a szoftver működési hibáinak kockázatát [6]. Ennek ellenére az MI-alapú tesztelés bevezetése komoly kihívásokkal jár, ide tartozik többek között az MI modellek tréningjéhez szükséges adatok biztosítása és az adatbiztonsági kérdések kezelése [7]. A mesterséges intelligencia és az emberi tényező közötti egyensúly megtalálása kritikus a szoftvertesztelési folyamatokban [8]. Míg az MI alapú automatizálás jelentős előnyöket kínál, az emberi tesztelők tapasztalata és intuíciója továbbra is nélkülözhetetlen a tesztelési folyamatok sikeréhez.

3. AZ MI TECHNOLÓGIÁK ALKALMAZÁSA A SZOFTVERTESZTELÉSBEN

3.1. Gépi tanulás a tesztesetek generálásához–

A gépi tanulás (Machine Learning, ML) az MI egyik legismertebb ága, amely képes nagy mennyiségű adatból mintákat felismerni, majd ezen minták alapján döntéseket hozni. A szoftvertesztelés területén az ML alkalmazható a tesztesetek automatikus generálására és optimalizálására. Hagyományosan a teszteseteket manuálisan írják meg, ami időigényes és emberi hibákat vonhat maga után. Az ML algoritmusok képesek elemezni a korábbi tesztadatokat, hibajelentéseket és a felhasználói viselkedést, majd ezek alapján új teszteseteket létrehozni [4]. Az ML segítségével a tesztek célzottabbak lehetnek, mivel a gépi tanulási modellek felismerhetik azokat a mintákat, amelyek a legnagyobb valószínűséggel vezetnek hibákhoz. Ez különösen hasznos lehet nagy és komplex rendszerek tesztelésekor, ahol a manuálisan írt tesztesetek nem képesek lefedni a teljes funkcionalitást [2].

3.2 Természetes nyelvfeldolgozás

A természetes nyelvfeldolgozás (Natural Language Processing, NLP) az MI egy olyan területe, amely lehetővé teszi, hogy a számítógépek megértsék és feldolgozzák az emberi nyelvet. A szoftvertesztelés során az NLP különösen hasznos lehet a követelményanalízis és a hibakeresés területén. A követelmények gyakran természetes nyelven vannak megfogalmazva, ami homályosságot és félreértéseket eredményezhet. Az NLP technológiák képesek automatikusan elemezni a követelményeket, azonosítani a kétértelműségeket, és javaslatokat tenni a pontosításra [8]. Az NLP alkalmazható hibajelentések és tesztelési dokumentumok feldolgozására is. A hibajelentések elemzése során az NLP algoritmusok képesek gyorsan megtalálni a

releváns információkat, és automatikusan kategorizálni a hibákat. Ez csökkentheti a hibakeresési folyamat idejét, különösen akkor, ha nagy mennyiségű hibajelentés érkezik a fejlesztőcsapathoz

3.3 Prediktív analitika a hibák előrejelzésében

A prediktív analitika az MI egy olyan ága, amely statisztikai modellek és adatelemzés segítségével előrejelzéseket készít a jövőbeli eseményekre vonatkozóan. A szoftvertesztelésben a prediktív analitika felhasználható a hibák előrejelzésére, még mielőtt azok bekövetkeznének. Azáltal, hogy az MI elemzi a korábbi hibajelentéseket, a kódbázis változásait és más releváns adatokat, képes felismerni azokat a mintákat, amelyek a hibákhoz vezethetnek. Ez különösen értékes lehet a regressziós tesztelés során, ahol a rendszer frissítései után kell vizsgálni, hogy a korábban megírt funkcionalitások továbbra is hibamentesen működnek. A prediktív analitika segíthet azonosítani azokat a területeket, amelyek a legnagyobb valószínűséggel hibásak lehetnek, így a tesztelési erőforrásokat hatékonyabban lehet felhasználni [7].

3.4 Automatikus tesztelés irányítása intelligens rendszerekkel

Az MI alkalmazása lehetővé teszi a tesztelési folyamatok intelligens irányítását. Az intelligens rendszerek képesek dinamikusan alkalmazkodni a tesztelés során felmerülő változásokhoz, és automatikusan módosítani a teszteseteket a szoftverfrissítések vagy a rendszer átalakítása alapján. Az MI-alapú rendszerek folyamatosan figyelik a szoftver viselkedését, és a kapott adatok alapján frissítik a tesztelési stratégiát. Ez jelentős időmegtakarítást eredményez, mivel nem kell manuálisan átírni a teszteseteket minden egyes változtatásnál Ezek az intelligens rendszerek képesek arra is, hogy a tesztelés során valós időben tanuljanak, és azonnal reagáljanak a hibákra. Ez azt jelenti, hogy a hibákat már a fejlesztés korai szakaszában felismerhetik, ami minimalizálja a hibás kód kiadásának esélyét.

3.5 MI-alapú eszközök és platformok a szoftvertesztelés támogatására

Számos MI-alapú eszköz és platform jelent meg az utóbbi években, amelyek kifejezetten a szoftvertesztelési folyamatok automatizálását és optimalizálását célozzák. Ezek az eszközök lehetővé teszik, hogy a tesztelők kevesebb manuális beavatkozással, gyorsabban és hatékonyabban hajtsák végre a szükséges teszteket. Ezek az eszközök nemcsak a tesztek futtatására alkalmasak, hanem képesek az eredmények kiértékelésére is, azonosítva az esetleges eltéréseket és hibákat. Az MI-alapú platformok segítségével a tesztelők jobban megérthetik a szoftver teljesítményét és viselkedését, ami lehetővé teszi a gyorsabb hibaelhárítást és fejlesztést. Az 1. táblázat néhány ilyen eszközt mutat:

MI alapú szoftvertesztelő eszközök 1. táblázat

#	*Név*	*URL*	*Megnevezés*
1	Test.ai	https://www.test.ai/	MI-t használ mobilalkalmazások UI elemeinek felismerésére és automatikus tesztek futtatására.
2	Applitools	https://applitools.com/	Az Applitools vizuális MI alapú tesztelést kínál, amely a felhasználói felületek vizuális elemeinek automatikus ellenőrzésére szakosodott. A rendszer képes összehasonlítani a szoftver különböző verzióinak vizuális megjelenését, és azonnal felismerni a különbségeket. Ez a technológia ideális olyan esetekben, amikor a vizuális megjelenés kritikus a felhasználói élmény szempontjából ellenőrzésére és összehasonlítására.
3	Eggplant AI	https://www.eggplantsoftware.com/	Az Eggplant AI egy átfogó tesztelési eszköz, amely különböző tesztelési módszereket kombinál: funkcionális tesztelést, teljesítménytesztelést és vizuális tesztelést is kínál. Az MI használata lehetővé teszi, hogy intelligensen irányítsa a tesztelési folyamatot, és automatikusan kiválassza a legkritikusabb teszteseteket. Az eszköz célja, hogy minimalizálja a manuális beavatkozást, miközben maximalizálja a tesztelés hatékonyságát

#	*Név*	*URL*	*Megnevezés*
4	Functionize	https://www.functionize.com/	A Functionize platform mesterséges intelligencia segítségével automatizálja a tesztelési folyamatokat, miközben egyszerűvé teszi a tesztek karbantartását és frissítését. Az eszköz gépi tanulási algoritmusokat használ a tesztesetek létrehozására és optimalizálására, és képes dinamikusan alkalmazkodni a szoftver változásaihoz. Ez lehetővé teszi, hogy a tesztek könnyen skálázhatók legyenek anélkül, hogy folyamatos manuális frissítést igényelnének
5	Mabl	https://www.mabl.com/	A Mabl egy intelligens tesztelési platform, amely az MI segítségével automatizálja a funkcionális és regressziós tesztelést. Az MI képes azonosítani a szoftverfejlesztési folyamat során bekövetkező változásokat, és ennek megfelelően frissíti a teszteseteket. A Mabl egyszerű felhasználói felülettel rendelkezik, amely lehetővé teszi a nem technikai felhasználók számára is, hogy részt vegyenek a tesztelési folyamatban.

4. ELŐNYÖK

4.1 Tesztelési idő és költségek csökkentése

Az MI-alapú eszközök egyik legnagyobb előnye a tesztelési idő és költségek jelentős csökkentése. A hagyományos tesztelési módszerek, különösen a manuális tesztelés, rendkívül időigényesek lehetnek, mivel minden egyes tesztesetet kézzel kell kidolgozni, futtatni és kiértékelni. Az MI azonban képes automatikusan generálni teszteseteket, futtatni azokat, és gyorsan elemezni az eredményeket. Ez a folyamat minimalizálja az emberi beavatkozást és jelentősen lerövidíti a tesztelési ciklusokat, ami csökkenti a fejlesztési időt és költségeket.

4.2 Tesztelési lefedettség növelése

Az AI eszközök képesek elemezni a szoftver teljes kódbázisát és felhasználói interakcióit, ami lehetővé teszi a tesztelési lefedettség növelését. Míg a hagyományos tesztelési módszerek gyakran csak a legfontosabb funkciókra koncentrálnak, az AI képes a teljes rendszer lefedésére, és olyan rejtett hibákat is azonosíthat, amelyeket a manuális tesztelés esetleg figyelmen kívül hagyna. Ez különösen fontos olyan komplex rendszerek esetében, ahol a manuális tesztelés nem biztosít teljes lefedettséget.

4.3 Skálázhatóság

Az MI-alapú szoftvertesztelés lehetőséget ad a tesztelési folyamatok egyszerű és gyors skálázására. A skálázhatóság ebben a kontextusban azt jelenti, hogy a szoftvertesztelési folyamatok és rendszerek könnyen bővíthetők vagy adaptálhatók a növekvő igényekhez anélkül, hogy jelentős hatékonyságvesztést tapasztalnának. Az MI-alapú szoftvertesztelés esetében a skálázhatóság főként arra utal, hogy a tesztelési rendszerek képesek nagymértékű növekedést kezelni anélkül, hogy a teljesítmény csökkenne vagy a folyamatok túlságosan összetetté válnának. Például, ha egy vállalat egyszerre több új funkciót vezet be egy szoftvertermékben, az MI-alapú tesztelés képes gyorsan létrehozni új teszteseteket és ezeket párhuzamosan lefuttatni, függetlenül a projekt méretétől. A skálázhatóság tehát a tesztelési folyamat automatizálásának kulcsfontosságú aspektusa, mivel lehetővé teszi, hogy a tesztelési rendszer megbirkózzon a nagy volumenű adatokkal, komplex rendszerekkel vagy folyamatos fejlesztési iterációkkal anélkül, hogy aránytalanul több erőforrást igényelne vagy lassulna a működés.

4.4 Integráció a DevOps folyamatokkal

Az MI-alapú tesztelési eszközök integrálhatók a modern DevOps folyamatokkal, lehetővé téve a folyamatos tesztelést és fejlesztést. Az MI képes valós időben figyelni a szoftver változásait, automatikusan

frissíteni a teszteket, és azonnal visszajelzést adni a fejlesztőknek. Ez a folyamatos integráció és visszacsatolás gyorsabb hibajavítást és jobb együttműködést tesz lehetővé a fejlesztőcsapatok között.

5. KIHÍVÁSOK ÉS KORLÁTOK

Bár a mesterséges intelligencia alkalmazása a szoftvertesztelésben számos előnyt és lehetőséget kínál, több jelentős kihívással és korláttal is szembe kell nézni. Ezek a kihívások mind technológiai, mind emberi tényezőkből fakadnak, és gyakran akadályozzák az MI-alapú tesztelési rendszerek széleskörű elterjedését. Az alábbiakban bemutatjuk a legfontosabb akadályokat, amelyekkel a fejlesztők és tesztelők szembesülhetnek az AI alkalmazása során.

5.1 Az adatok minősége és hozzáférhetősége

Az MI-alapú rendszerek működésének egyik alapfeltétele a nagy mennyiségű és kiváló minőségű adat. A gépi tanulási algoritmusok megfelelő működése érdekében szükség van olyan adatokra, amelyek pontosan tükrözik a szoftver működését és a felhasználói interakciókat. Azonban a valóságban gyakran előfordul, hogy nem áll rendelkezésre elegendő adat, vagy az elérhető adatok hiányosak, pontatlanok vagy nem reprezentatívak. Az ilyen adatok felhasználása esetén az MI rendszerek hibás következtetéseket vonhatnak le, ami pontatlan tesztelési eredményekhez vezethet.

5.2 A modellek pontossága, tréningje és frissítése

Az MI modellek, különösen a gépi tanulás alapú rendszerek, nem mindig nyújtanak tökéletes eredményeket. Az MI rendszerek pontossága nagymértékben függ a felhasznált adatok minőségétől, a modellek tréningjétől és az alkalmazott algoritmusoktól. Ha ezek közül bármelyik nem megfelelően van kialakítva vagy karbantartva, az AI által generált tesztelési eredmények pontatlanok vagy félrevezetőek lehetnek. Ennek következtében a szoftverfejlesztők és tesztelők nem feltétlenül bízhatnak teljes mértékben az AI rendszerek által nyújtott eredményekben, és szükség lehet további manuális ellenőrzésre. Az AI modellek, különösen a gépi tanulási alapú rendszerek esetében, folyamatosan frissítésre és újra tréningre szorulnak. Ahogy a szoftverek változnak, az MI modelleknek is követniük kell ezeket a változásokat annak érdekében, hogy továbbra is pontos teszteredményeket biztosítsanak. A modellek tréningje azonban időigényes és költséges folyamat, különösen akkor, ha a rendszert gyakran kell frissíteni. Emellett a tréninghez szükséges adatok és erőforrások korlátozottak lehetnek, ami akadályozhatja a modellek hatékonyságát.

5.3 Az MI rendszerek bonyolultsága, költsége és erőforrásigénye

Az MI-alapú rendszerek komplexitása gyakran kihívást jelent a szoftvertesztelő csapatok számára. Az MI rendszerek megértése és hatékony használata speciális tudást és készségeket igényel, amelyek nem feltétlenül áll rendelkezésre minden fejlesztő vagy tesztelő számára. Az MI rendszerek bevezetése tehát további képzést és szakértői támogatást igényelhet, ami növelheti a kezdeti költségeket és az implementálás idejét. Továbbá, az MI döntéseinek átláthatósága is problémát jelenthet: a gépi tanulási modellek gyakran „fekete dobozokként” működnek, amelyek eredményeit nehéz értelmezni vagy ellenőrizni. Az MI-alapú tesztelési rendszerek fejlesztése és fenntartása jelentős erőforrásokat igényel. Bár hosszú távon az MI csökkentheti a tesztelési időt és költségeket, a kezdeti beruházások magasak lehetnek. Az MI rendszerek bevezetése előtt a vállalatoknak jelentős összegeket kell befektetniük a szükséges infrastruktúrába, az adatok előkészítésébe és a modellek tréningjébe. Emellett a folyamatos karbantartás és frissítés további költségeket generálhat, ami akadályozhatja a kisebb szervezetek számára az MI-alapú megoldások bevezetését

5.5 Az emberi tényező és az MI közötti egyensúly

Bár az MI jelentős mértékben képes automatizálni a szoftvertesztelési folyamatokat, az emberi tényező továbbra is kritikus szerepet játszik. Az MI nem képes minden helyzetet és tesztelési követelményt megfelelően kezelni, különösen a kreatív, nem várt problémák észlelésében és megoldásában. Az emberi tapasztalat és intuíció elengedhetetlen a hibakeresés során, különösen az olyan területeken, mint a felhasználói élmény vagy a szoftver funkcionális követelményeinek értékelése. Az MI és az emberi tényező közötti egyensúly megtalálása tehát kulcsfontosságú a hatékony szoftvertesztelési folyamatok kialakításához.

5.6 Adatbiztonság és adatvédelem

Az MI-alapú rendszerek általában nagy mennyiségű adatot használnak fel a tanuláshoz és az elemzésekhez, ami adatvédelmi és biztonsági aggályokat vet fel. A szoftvertesztelés során összegyűjtött adatok érzékeny információkat tartalmazhatnak, például felhasználói adatokat vagy üzleti titkokat. Az adatok védelme és a vonatkozó szabályozásoknak való megfelelés, mint például a GDPR, kihívást jelenthet az MI-alapú tesztelési rendszerek alkalmazása során. Az adatok védelmére irányuló intézkedések implementálása bonyolultabbá teheti a rendszert, és további erőforrásokat igényelhet [6].

6. KONKLÚZIÓ

A mesterséges intelligencia szerepe a szoftvertesztelésben már ma is jelentős, de a technológia fejlődése azt sugallja, hogy a jövőben még nagyobb mértékben integrálódik a szoftverfejlesztési és tesztelési folyamatokba. Az MI nem csupán felgyorsítja a tesztelési folyamatokat, hanem olyan intelligens képességeket kínál, amelyek növelik a tesztek lefedettségét, csökkentik a hibák felismerési idejét, és elősegítik a prediktív hibajavítást. Bár az MI alkalmazása jelentős előnyökkel jár, fontos figyelembe venni a technológiával járó kihívásokat is, például az adatminőség kérdését, a modellek tréningjének nehézségeit és az MI-rendszerek komplexitását. Az MI rendszerek megfelelő működéséhez szükség van az emberi tényezőre is, különösen az összetett vagy kreatív hibák azonosítása és megoldása során. Az MI-alapú tesztelés jövője fényes, és a technológia fejlődésével még nagyobb hatékonyságnövekedés várható a szoftverfejlesztési folyamatokban. Azon vállalatok, amelyek képesek kihasználni az MI által kínált lehetőségeket, versenyelőnyhöz juthatnak, mivel gyorsabban és megbízhatóbban tudják piacra dobni szoftvertermékeiket.

IRODALMI HIVATKOZÁSOK

[1] Sommerville, I. *Software Engineering.* Pearson Education, New York, 2016.

[2] Myers, G. J., Badgett, C., Sandler, C. *The Art of Software Testing*. John Wiley & Sons, , New York, 2011.

[3] Hass, A. M. Guide to advanced software testing. Artech House Boston, 2014

[4] Ricca, F., Marchetto, A., Stocco, A. Ai-based test automation: A grey literature analysis. (2021 IEEE International Conference on Software Testing, Verification and Validation Workshops ICSTW), pp. 263-270).

[5] Wang, J., Huang, Y., Chen, C., Liu, Z., Wang, S., Wang, Q. *Software Testing with Large Language Models: Survey, Landscape, and Vision*. (IEEE Transactions on Software Engineering, 01), pp.1-27. (2024).

[6] Olaleye, T.O., Arogundade, O.T., Misra, S., Abayomi-Alli, A., Kose, U. Predictive Analytics and Software Defect Severity: A Systematic Review and Future Directions. Scientific Programming. 2023, pp 1.18

[7] Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., nushu, B., Zimmermann, T.). Software engineering for machine learning: A case study. (2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice, ICSE-SEIP) (pp. 291-300). IEEE. 2019

[8] Kaner, C., Bach, J., Pettichord, B. *Lessons Learned in Software Testing*. Wiley, New york, 2008.

[8] Sarro, F.Predictive analytics for software testing. (2018 IEEE/ACM 11th International Workshop on Search-Based Software Testing SBST) (pp. 1-1). IEEE.