**Primena veštačke inteligencije u razvoju softverskih rešenja: Studija slučaja prediktivnog održavanja ESP pumpi**

**1. Uvod**

Veštačka inteligencija (VI, eng. AI) u poslednjih nekoliko godina transformisala je gotovo sve industrijske grane, pa tako i oblast softverskog inženjerstva. Sa pojavom moćnih jezičkih modela i tehnika dubokog učenja, otvorene su nove mogućnosti za unapređenje procesa razvoja softvera, automatizaciju testiranja, generisanje koda, pa čak i predikciju kvarova u složenim tehničkim sistemima. Ovaj rad istražuje način na koji se veštačka inteligencija koristi za razvoj softverskih rešenja, sa akcentom na konkretnu primenu kroz studiju slučaja prediktivnog održavanja ESP pumpi u naftnoj industriji.

U okviru hakaton izazova "DigiHack 2024", naš tim je razvijao end-to-end rešenje koje koristi metapodatke i telemetrijske informacije prikupljene iz bušotina kako bi predvideo potencijalne kvarove ESP pumpi. Pored uvida u aktuelne naučne radove, ovaj rad obuhvata detaljan prikaz metodologije, arhitekture softverskog rešenja i konkretne rezultate dobijene korišćenjem AI modela u analizi vremenskih serija.

**2. Veštačka inteligencija**

**2.1. Osnovni pojmovi i definicije**

Veštačka inteligencija predstavlja oblast računarskih nauka koja se bavi razvojem sistema sposobnih da obavljaju zadatke koji zahtevaju "ljudsku" inteligenciju. U najširem smislu, AI obuhvata:

* *Učenje (machine learning)*
* *Razumevanje jezika (natural language processing)*
* *Računarsku viziju (computer vision)*
* *Planiranje i odlučivanje*

Prema Durrani et al. (2024), integracija AI-a u softversko inženjerstvo se odvija u tri glavna pravca: asistivna inteligencija (pomaganje programerima), autonomna inteligencija (automatsko generisanje rešenja) i inteligencija sa objašnjenjem (XAI - explainable AI).

**2.2. Kategorije AI sistema**

* *Uža AI (Narrow AI)* – specijalizovani sistemi za jedan zadatak (npr. klasifikacija kvarova)
* *Opšta AI (General AI)* – još uvek teorijska, podrazumeva razumevanje i primenu znanja na više oblasti
* *Generativna AI* – modeli koji generišu nove podatke: slike, tekst, kod (GPT, DALL·E itd.)

**3. Generativna veštačka inteligencija i LLM**

Generativna veštačka inteligencija (GenAI) predstavlja klasu veštačke inteligencije koja je sposobna da na osnovu velike količine ulaznih podataka generiše nove, smisleno strukturirane izlaze – tekst, kod, slike, pa čak i vremenske serije. Ovi modeli se oslanjaju na duboko učenje, najčešće korišćenjem transformera, a njihova osnovna snaga leži u učenju statističkih obrazaca u podacima i njihovoj kreativnoj reimplementaciji u generisanom sadržaju.

U savremenoj AI praksi, jedan od najzastupljenijih oblika GenAI sistema su veliki jezički modeli (LLM – Large Language Models). Među najpoznatijima su OpenAI GPT-3 i GPT-4, Google-ov PaLM, Meta-ov LLaMA, kao i Claude kompanije Anthropic. Ovi modeli su sposobni da generišu programski kod, pišu komentare, prevode kod iz jednog jezika u drugi, objašnjavaju greške i automatizuju ceo tok razvoja softverskih komponenti.

LLM modeli trenirani su na ogromnim skupovima podataka koji uključuju tekstove sa interneta, akademsku literaturu, i repozitorijume sa otvorenim kodom poput GitHub-a. Zbog toga su razvili sposobnost da odgovaraju na zahteve korisnika na način koji podseća na ponašanje eksperta iz određene oblasti. Na primer, GPT-4 je sposoban da napiše funkcionalan Python kod za klasifikaciju vremenskih serija u okviru prediktivnog održavanja, dok Claude može analizirati logiku već postojećeg sistema i dati predloge za refaktorisanje koda.

Jedna od ključnih prednosti korišćenja GenAI modela u softverskom inženjerstvu jeste ubrzanje procesa razvoja. Arora i saradnici (2025) navode da je u internim studijama nekoliko kompanija zabeležen pad vremena potrebnog za izradu prototipa softverskog modula i do 50%. Međutim, autori takođe ističu i opasnosti: automatski generisan kod često nije objašnjen, može sadržavati sigurnosne propuste, kao i nepotrebno ponavljanje funkcionalnosti koje generišu "tehnički dug" ako se ne kontroliše kvalitet.

Nadalje, studije kao što su Kokol (2024) i Upadhyaya (2024) upozoravaju da programeri često imaju preveliko poverenje u GenAI, oslanjajući se na generisani kod bez temeljne validacije, što može dovesti do ozbiljnih problema u produkcionom okruženju. Takođe, postoje etičke dileme u vezi sa vlasništvom nad kodom koji GenAI modeli generišu, jer se ne zna da li je deo tog koda direktno izveden iz licenciranih izvora.

Sa stanovišta primene u industriji, GenAI i LLM-ovi su se pokazali naročito korisnim u oblastima kao što su:

* **Analiza vremenskih serija** – gde modeli kao što su TimeGPT mogu da nauče obrasce i predviđaju anomalije;
* **Automatizacija održavanja** – u kontekstu prediktivnih analiza i preporuke servisa;
* **Generisanje dokumentacije** – uključujući API reference, test slučajeve i korisnička uputstva;
* **Objašnjavanje postojećeg koda** – što pomaže junior developerima da bolje razumeju nasleđene sisteme.

U slučaju ESP pumpi, ovi modeli mogu pomoći u detekciji obrazaca kvarova analizom istorijskih podataka i automatskim klasifikovanjem rizika, a uz minimalno ljudsko uplitanje. Takođe, LLM-ovi se mogu koristiti za generisanje objašnjenja zašto je do otkaza došlo, što dodatno povećava poverenje korisnika u sistem.

Zbog ovih prednosti, mnoge kompanije integrišu GenAI u svoje DevOps tokove, koristeći LLM modele kao asistente u svakodnevnim zadacima. Međutim, uz sve prednosti, GenAI i dalje zahteva visok stepen nadzora, evaluacije i razumevanja od strane korisnika kako bi se obezbedila sigurnost i pouzdanost u realnim sistemima.

**4. Primena AI u razvoju softvera**

**4.1. Automatizacija kroz AI**

AI alati se koriste za:

* Generisanje programskog koda (npr. GitHub Copilot, Amazon CodeWhisperer)
* Automatsko testiranje i detekciju grešaka
* Refaktorisanje koda i otkrivanje tehničkog duga
* Predikciju performansi softvera

Kokol (2024) sistematski analizira kako se AI koristi u svakoj fazi SDLC (Software Development Life Cycle) procesa.

**4.2. LLM alati za razvoj**

LLM modeli su pokazali naročit potencijal u pisanju koda na više jezika, kreiranju testova, kao i u analizi logova i predikciji grešaka na osnovu velikih količina podataka. Upadhyaya (2024) istakao je kako prompt-inženjering i fine-tuning omogućavaju preciznije i pouzdanije rešenje specijalizovanih zadataka.

**5. Pregled literature**

Pregledom dostupne literature identifikovani su sledeći ključni doprinosi:

* Kokol et al. (2024): Širok pregled AI alata u softverskom inženjerstvu
* Upadhyaya (2024): Detaljna klasifikacija AI zadataka u razvoju softvera
* Durrani et al. (2024): Istorijska analiza integracije AI u SDLC
* Paroha et al. (2024): Uporedna analiza TimeGPT i Time-LLM modela u prediktivnom održavanju ESP pumpi
* Arora et al. (2025): Uloga XAI u razvoju pouzdanih softverskih rešenja

**6. Studija slučaja: Prediktivno održavanje ESP pumpi**

**6.1. Opis problema i industrijski kontekst**

ESP pumpe su ključne za mehaničku eksploataciju nafte u bušotinama gde pritisak nije dovoljan. Njihovi kvarovi izazivaju ozbiljne operativne i finansijske gubitke. Njihovo održavanje zahteva značajna sredstva, jer su locirane duboko ispod površine.

U ovom projektu analizirani su podaci iz 50 bušotina, sa preko 3 miliona redova telemetrijskih podataka i klasifikacijom otkaza u tri kategorije: mehaničke primese, hidrati i nehermetičnost tubinga.

**6.2. Dataset i obrada podataka**

* Preko 3M redova sa sledećim atributima: pritisak, temperatura, frekvencija, vreme, oznaka bušotine, i tip otkaza
* Korišćena biblioteka: pandas, numpy, matplotlib, seaborn, scikit-learn
* Korišćeni koraci:
  + Grupisanje i agregacija po bušotinama i vremenskim intervalima
  + Feature engineering – kreiranje indikatora rasta temperature, pritiska i oscilacija
  + Label encoding i balansiranje podataka (SMOTE metoda)

**6.3. Izbor modela i metrika**

* Isprobani modeli: Random Forest, XGBoost, LSTM
* Finalni model: XGBoost sa najvećom F1 ocenom
* Metrike evaluacije:
  + F1-score, Precision, Recall
  + ROC AUC kriva (analiza pod signalom)

**6.4. Karakteristični delovi koda**

* Korišćenje pipeline objekta za obradu podataka i treniranje
* GridSearchCV za optimizaciju hiperparametara
* classification\_report i confusion\_matrix za evaluaciju

**6.5. Vizualizacije**

* Prikaz trendova za svaku bušotinu (npr. kako temperatura raste neposredno pre otkaza)
* Matrice konfuzije za svaki model
* ROC krive za tri kategorije otkaza

**7. Diskusija i ograničenja**

Model pokazuje zadovoljavajuće rezultate u većini slučajeva, ali postoje izazovi:

* Neki tipovi otkaza imaju mnogo manje uzoraka (neravnoteža)
* Mogući noise u telemetrijskim podacima
* Nema eksplicitne verifikacije modela na stvarnim novim bušotinama

**8. Zaključak**

Kroz studiju slučaja prikazano je kako AI može biti iskorišćen za izradu skalabilnog i korisnog softverskog rešenja za prediktivno održavanje. Upotreba vremenskih serija, obrada podataka i selekcija odgovarajućeg modela omogućili su visok stepen tačnosti i pouzdanosti.

Ovaj projekat pokazuje značaj AI-a u unapređenju industrijskih operacija i pruža osnovu za buduće šire implementacije.

**9. Etika i izazovi primene AI u softverskom inženjerstvu**

**9.1. Algoritamska pristrasnost**

Jedan od najvažnijih izazova u primeni veštačke inteligencije u softverskom inženjerstvu jeste algoritamska pristrasnost. AI modeli, posebno oni koji se treniraju na velikim količinama podataka, mogu internalizovati pristrasnosti prisutne u tim podacima. U kontekstu prediktivnog održavanja, to može dovesti do preteranog označavanja određenih tipova bušotina kao rizičnih, bez stvarnog razloga.

**9.2. Transparentnost i objašnjivost (Explainable AI)**

Potrebno je da korisnici softverskih rešenja razumeju kako AI model donosi odluke, posebno u industrijskim aplikacijama gde su sigurnost i preciznost kritični. Primena XAI (Explainable AI) pristupa omogućava korisnicima i inženjerima da bolje razumeju procese odlučivanja unutar modela i time izgrade poverenje u sistem.

**9.3. Privatnost i zaštita podataka**

U sistemima koji analiziraju velike količine operativnih i tehničkih podataka iz industrije, postoji rizik od curenja poverljivih informacija. Primenom tehnika kao što su diferencijalna privatnost i enkripcija podataka, može se značajno smanjiti rizik od zloupotrebe.

**9.4. Odgovornost i zakonska regulativa**

Postavlja se pitanje odgovornosti za odluke koje donose autonomni sistemi. Ako model pogrešno klasifikuje stanje bušotine i dođe do otkaza, ko je pravno odgovoran? Neophodno je da zakonski okvir prati razvoj ovih tehnologija.

**10. Prilozi**

**A white board with writing on it

AI-generated content may be incorrect.**

Na slici je prikazana **inicijalna konceptualna arhitektura sistema**, razvijena kao deo planiranja potencijalne **produkcione implementacije** rešenja. Arhitektura je osmišljena kao **mikroservisna**, sa jasno podeljenim komponentama koje komuniciraju putem API-ja i rade u odvojenim kontejnerima:

* **Machine Learning servis** zadužen je za obradu podataka, generisanje predikcija i periodično (npr. na svakih 24h) ponovno treniranje modela na novim podacima;
* **Baza podataka** skladišti telemetrijske podatke, izlaze iz modela i istorijat otkaza;
* **Backend servis** upravlja logikom sistema, distribuiranjem predikcija i služi kao posrednik između modela i korisničkog interfejsa;
* **Frontend (UI)** prikazuje rezultate korisniku kroz interaktivnu aplikaciju.

U realnom okruženju, **telemetrijski senzori iz bušotina bi neprekidno slali podatke**, koji bi se automatski unosili u sistem. Na osnovu toga, AI model bi vršio **automatsku dnevnu analizu**, predikciju i ažuriranje statusa mašina, čime se omogućava **prava primena koncepta prediktivnog održavanja**.

Ovakva arhitektura omogućava **skalabilnost, fleksibilnost i otpornost sistema**, kao i mogućnost da se različiti delovi rešenja nezavisno razvijaju, testiraju i unapređuju — što je savremen standard u razvoju naprednih softverskih sistema zasnovanih na veštačkoj inteligenciji.

A screenshot of a computer screen

AI-generated content may be incorrect.

U ovom delu vrši se konvertovanje vremenskih podataka u datetime format i računa se broj dana do otkaza (days\_to\_failure), koji predstavlja ciljnu promenljivu u regresionom modelu.

Iz dataset-a se izdvajaju relevantne kolone za modeliranje, a zatim se vrši konverzija kategorijalnih promenljivih u numeričke pomoću get\_dummies. Podaci se zatim dele na trening i test skup i normalizuju pomoću StandardScaler kako bi model brže i stabilnije učio.

Ovde se definiše struktura **višeslojne neuronske mreže** korišćenjem Keras biblioteke. Mreža se sastoji od tri Dense sloja sa ReLU aktivacijom i Dropout slojevima za regularizaciju, a završni sloj je linearni sa jednim izlazom za regresiju. Mreža se trenira koristeći Adam optimizator i mean\_squared\_error kao funkciju greške, dok se metrike procenjuju pomoću srednje apsolutne greške (mae).

A screen shot of a graph

AI-generated content may be incorrect.

Ovaj deo koda prikazuje vizualizaciju promena vrednosti funkcije greške tokom epoha treniranja modela. Prikazane su dve linije: Training Loss označava grešku na trening skupu, dok Validation Loss prikazuje grešku na validacionom skupu. Ova analiza omogućava uvid u to da li model uči efikasno i da li postoji pojava overfittinga (ako val\_loss počne da raste dok loss i dalje opada).

A screenshot of a computer

AI-generated content may be incorrect.

**A graph with a line and a point

AI-generated content may be incorrect.**

Na osnovu binarizovanih regresionih predikcija (prag: 10 dana), prikazana je ROC kriva modela. Površina ispod krive (AUC) iznosi **0.62**, što ukazuje na **umerenu sposobnost modela da razlikuje skor odloženih otkaza**.

Iako rezultat nije idealan, model pruža korisne signale za **ranu detekciju potencijalnih problema** u radu ESP pumpi, što može doprineti pravovremenom održavanju i smanjenju troškova zastoja. Dodatnim unapređenjima modela moguće je povećati pouzdanost klasifikacije.

A screen shot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

U ovom delu izvršena je analiza **srednjih vrednosti ključnih numeričkih parametara** za različite uzroke otkaza ESP pumpi (mehaničke primese, hidrati, nehermetičnost tubinga).

Podaci su grupisani po tipu otkaza (label), a zatim je za svaku od odabranih numeričkih promenljivih (npr. napon, struja, pritisak, temperatura) izračunata prosečna vrednost neposredno **pre trenutka otkaza**.

Na grafikonima su prikazani **trendovi parametara po tipu otkaza**, što omogućava uvid u karakteristične promene koje prethode kvarovima i može pomoći u ranijoj dijagnostici problema.

A screen shot of a computer code

AI-generated content may be incorrect.

A graph with green and white dots

AI-generated content may be incorrect.

Korišćenjem **K-means algoritma** izvršeno je klasterovanje podataka na osnovu dve ključne numeričke karakteristike: **napon AB** i **električna struja faze A**. Model je podelio uzorke u četiri klastera, a zatim su ti klasteri **mapirani na moguće tipove grešaka** u radu ESP pumpi (mehanička primesa, hidratacija, nehermetičnost tubinga i ispravan rad).

Rezultat je vizualizovan na dijagramu rasipanja, gde se svaka tačka predstavlja telemetrijskim zapisom, a boja ukazuje na pripadnost određenom tipu ponašanja sistema. Ova analiza pomaže u **vizuelnoj identifikaciji obrazaca koji prethode različitim vrstama kvarova**.

A screenshot of a machine overview

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A graph on a computer screen

AI-generated content may be incorrect.

U završnoj fazi projekta razvijen je prototip korisničke aplikacije koja omogućava **pregled svih mašina**, njihovog **trenutnog statusa** i **procene vremena do otkaza** na osnovu predikcija AI modela. Aplikacija je dizajnirana tako da bude **pristupačna krajnjem korisniku**, bez potrebe za tehničkim znanjem iz oblasti veštačke inteligencije.

Korisnik može:

* Pregledati sve ESP mašine i njihove ključne parametre u jednoj tabelarnoj celini,
* Videti predikciju za svaki uređaj, uključujući **koliko dana je preostalo do mogućeg otkaza**, kao i **vrstu detektovanog rizika** (mehanička primesa, hidrati, nehermetičnost tubinga),
* Ući u **detaljan prikaz pojedinačne mašine**, gde su dostupni:  
  • istorijski telemetrijski podaci,  
  • dinamički grafikon koji prikazuje promene izabranog parametra kroz vreme,  
  • dodatne informacije za dijagnostiku i planiranje servisa.

Cilj ovakvog softverskog rešenja jeste da omogući **efikasno prediktivno održavanje**, bez potrebe da se inženjeri oslanjaju na konzolne alate, terminale ili ručne analize. Sve ključne funkcionalnosti AI modela su integrisane u **intuitivan korisnički interfejs**, što omogućava brzu reakciju i donošenje odluka u realnom vremenu.

Ovime se veštačka inteligencija ne koristi samo kao alat u pozadini, već postaje **konkretan deo softverskog proizvoda**, usmerenog na rešavanje stvarnog industrijskog problema — što je suština moderne primene AI u razvoju softvera.

**11. Reference**

* Arora, S., Singh, R., & Kumar, P. (2025). Explainable AI in Software Engineering. *arXiv preprint arXiv:2505.07058*.
* Durrani, T., Awan, A., & Ahmed, S. (2024). A Decade of Progress: AI in Software Engineering (2013–2023). *ResearchGate*.
* Kokol, P., Zorman, M., & Blažun Vošner, H. (2024). The Use of AI in Software Engineering. *MDPI Information, 15(6), 354*.
* Paroha, V., Singh, D., & Raghuwanshi, S. (2024). TimeGPT vs Time-LLM in ESP Maintenance. *ResearchGate*.
* Upadhyaya, S. (2024). The Role of Artificial Intelligence in Software Development: A Literature Review. *ResearchGate*.