A black background with blue letters

AI-generated content may be incorrect.

**Detekcija neželjene pošte (SPAM) primenom klasičnih ML metoda**

**Predmet: Bezbednost aplikacija**

**Mentor: Student: dr Dušan Stefanović Jovan Jovanović   
 ITS 20/24**

# SADRŽAJ

[SADRŽAJ 2](#_Toc206982428)

[1. Uvod 4](#_Toc206982429)

[1.1 Kontekst i motivacija 4](#_Toc206982430)

[1.2 Problem, ciljevi i istraživačka pitanja 5](#_Toc206982431)

[1.3 Skupovi podataka i priprema 5](#_Toc206982432)

[1.4 Korišćene metode i alati 6](#_Toc206982433)

[1.5 Metodološke napomene i bezbednosni kontekst 7](#_Toc206982434)

[1.6 Struktura rada 7](#_Toc206982435)

[2. Metodologija i dizajn eksperimenta 8](#_Toc206982436)

[2.1 Okruženje i biblioteke 8](#_Toc206982437)

[2.2 Izvori podataka i priprema 8](#_Toc206982438)

[2.3 Reprezentacija teksta (TF‑IDF) 10](#_Toc206982439)

[2.4 Modeli i hiperparametri 11](#_Toc206982440)

[2.5 Podela na skupove i validacija 12](#_Toc206982441)

[2.6 Metrike i vizualizacije 12](#_Toc206982442)

[2.7 Prag odlučivanja i kompromis FP/FN 14](#_Toc206982443)

[2.8 Reproducibilnost i kontrola verzija 15](#_Toc206982444)

[3. Implementacija i rezultati 16](#_Toc206982445)

[3.1 Implementacioni tok 16](#_Toc206982446)

[3.2 Rezultati - Multinomial Naive Bayes (MNB) 17](#_Toc206982447)

[3.3 Rezultati - LinearSVC 19](#_Toc206982448)

[3.4 Rezultati - XGBoost 20](#_Toc206982449)

[3.5 Poređenje modela 23](#_Toc206982450)

[3.6 Analiza grešaka (FP/FN) 24](#_Toc206982451)

[3.7 Sažetak nalaza 24](#_Toc206982452)

[4. Diskusija i analiza 25](#_Toc206982453)

[4.1 Sažetak performansi po modelima 25](#_Toc206982454)

[4.2 PR vs ROC i neravnoteža klasa 26](#_Toc206982455)

[4.3 Prag odlučivanja i scenariji korišćenja 26](#_Toc206982456)

[4.4 FP/FN - obrasci i preporuke 27](#_Toc206982457)

[4.5 Ograničenja i rizici 28](#_Toc206982458)

[4.6 Preporuke za produkciju 28](#_Toc206982459)

[5. Zaključak i budući rad 30](#_Toc206982460)

[Literatura 31](#_Toc206982461)

# 1. Uvod

Digitalna pošta je i dalje najrasprostranjeniji kanal poslovne komunikacije, ali upravo zbog te sveprisutnosti predstavlja i najčešću površinu napada. Neželjena pošta (SPAM), a posebno ciljane fišing (phishing) kampanje, evoluirale su od trivijalnih promotivnih poruka do sofisticiranih prevara koje oponašaju stil i vizuelni identitet legitimnih organizacija. U takvom okruženju, automatizovana klasifikacija poruka prelazi iz domene “kvaliteta usluge” u prvu liniju sajber-odbrane: svaka propuštena maliciozna poruka (FN) može dovesti do finansijske štete ili kompromitovanja naloga, dok svaka pogrešno blokirana legitimna poruka (FP) degradira produktivnost i poverenje korisnika.

Ovaj rad demonstrira potpun, praktičan tok izgradnje i evaluacije filtra neželjene pošte na osnovu klasičnih metoda mašinskog učenja. Polazimo od javnih korpusa, sprovodimo pažljivu pripremu podataka, zatim poređujemo tri komplementarna modela (Multinomial Naive Bayes, LinearSVC i XGBoost), i završavamo diskusijom o metrikama, podešavanju praga odlučivanja i implikacijama za bezbednosnu praksu [1], [2], [5], [10].

## 1.1 Kontekst i motivacija

Spam više nije samo “smetnja u inbox-u”. On je operativni i bezbednosni problem: troši mrežne i skladišne resurse, narušava fokus zaposlenih, i - što je kritično - služi kao vektor za krađu kredencijala, širenje malvera i preuzimanje naloga. Moderne kampanje ciljaju korisnike psihološkim okidačima (hitnost, autoritet, lažne isplate), često koriste legitimne kompromitovane domene i tehnike zaobilaženja filtera (parafraziranje, Unicode varijante, URL cloaking).

U praksi, organizacije žele filter koji je brz, stabilan i podesiv: brz - da ne usporava isporuku poruka; stabilan - da ne “puca” na novim šablonima; podesiv - da se prag odlučivanja uskladi sa rizikom konkretne organizacije. Zbog ovih zahteva, i dalje postoji snažna motivacija da se istraže i uporede klasični ML pristupi: oni su transparentni, relativno laki za održavanje i često daju “dovoljno dobre” rezultate uz skromne resurse, naročito u kombinaciji sa dobro dizajniranom TF-IDF reprezentacijom [1], [7].

Poseban motiv je i merljivost: dobrim metrikama (naročito PR krivama) lako se komunicira menadžmentu “šta gubimo ako spustimo/prenesemo prag”, što je presudno za donošenje bezbednosnih politika [5], [6].

## 1.2 Problem, ciljevi i istraživačka pitanja

Problem: dati tekst poruke x, potrebno je odrediti funkciju f(x)∈{ham, spam} f(x)\in\{\text{ham},\ \text{spam}\}f(x)∈{ham, spam} koja minimizuje očekivani trošak grešaka u realnim uslovima. Cilj: postići visoku tačnost uz dobar balans preciznosti i odziva na pozitivnoj klasi (spam), tako da se istovremeno smanje FN (propust maliciozne pošte) i FP (blokada legitimne pošte), uz zadržavanje računarske efikasnosti i jednostavnosti integracije.

Istraživačka pitanja:

(1) Kako izbor reprezentacije teksta (TF-IDF) utiče na učinak modela? [1]

(2) Koji je odnos između linearnog razdvajanja (LinearSVC) i generativnog pristupa (MultinomialNB) na bag-of-words obeležjima? [1]

(3) Da li nelinearni ansambl (XGBoost) donosi vidljivu dobit nad BOW postavkom i u kojem trošku kompleksnosti? [2]

(4) Koje metrike i vizualizacije su najosetljivije na neravnotežu klasa (PR vs. ROC) i kako utiču na izbor praga? [5], [6]

## 1.3 Skupovi podataka i priprema

Koristimo SpamAssassin Public Corpus (različiti podskupovi ham/spam) i Enron Email Dataset (legitimna korespondencija), jer su to standardni, transparentni i široko citirani izvori za evaluaciju filtera [3], [4]. Kombinovanjem ova dva izvora dobijamo korpus koji istovremeno pokriva raznolike stilove legitimne pošte i prepoznatljive obrasce spama.

Priprema podataka je realizovana internim skriptom data\_prep.py [9] i obuhvata:

* Parsiranje MIME strukture: preferira se text/plain; ukoliko je poruka multipart, ekstrahuju se svi “text/plain” delovi i spajaju u jedinstveni sadržaj. Ako parsiranje zakaže, koristi se “fallback” čitanje čistog teksta [8], [9].
* Normalizacija i čišćenje: uklanjanje kontrolnih karaktera, kolapsiranje praznina i osnovna normalizacija novoredova, bez agresivnog “čišćenja” koje bi moglo ukloniti informativne tokene (npr. fragmente URL-ova).
* Deduplikacija: uklanjanje identičnih telo-poruka kako bi se sprečilo precenjivanje performansi (isti primeri u oba skupa).
* Filtriranje praznih/degenerisanih poruka: izbacuju se poruke bez sadržaja ili sa ekstremno kratkim telom.
* Etiketiranje i podela: obeležavanje spam/ham i stratifikovana podela na trening i test (u daljem tekstu i k-fold validacija), kako bi se očuvali proporcioni odnosi klasa i obezbedila fer procena generalizacije [1]. Svesno se u bazičnoj postavci ne koristi zaglavlje (header) poruke niti meta-signal (npr. SPF/DKIM), kako bismo izolovali učinak samo sadržaja. Ovo postavlja konzervativni “baseline” koji se kasnije može lako proširiti bogatijim obeležjima [10].

## 1.4 Korišćene metode i alati

Istražujemo tri komplementarna modela:

* Multinomial Naive Bayes (MNB) - klasičan generativni pristup za diskretne frekvencije reči; jednostavan, izuzetno brz i često iznenađujuće jak na BOW reprezentacijama. Pruža dobru interpretabilnost preko log-odds po tokenima [1].
* LinearSVC (linearni SVM) - robustan na visoko-dimenzione TF-IDF vektore; nalazi linearni hiper-ravni separator sa maksimalnom marginom, što je empirijski snažna kombinacija sa n-gram obeležjima [1], [7].
* XGBoost - gradijentno pojačana stabla, sposobna da modeluju nelinearne interakcije reči; često omogućava fino podešavanje praga kroz skorove i donosi dodatnu dobit tamo gde linearni modeli “propuštaju” obrasce [2]. Alatni ekosistem čine scikit-learn (TF-IDF, MNB, LinearSVC, metričke funkcije) i xgboost biblioteka (XGBClassifier), uz standardne procedure za izračunavanje matrica konfuzije, ROC i PR krivih [1], [2], [5], [6]. Ovakva postavka osigurava i reproduktivnost i ponovljivost rezultata.

## 1.5 Metodološke napomene i bezbednosni kontekst

Kontekst bezbednosti uvodi asimetrične troškove: FN (propušten spam/phishing) tipično je “skuplji” od FP (legitiman mejl u spamu), ali konkretne organizacije mogu preferirati različite tačke rada sistema. Zbog toga tačnost i čak F1-mera ponekad nisu dovoljne - ne otkrivaju dovoljno ponašanje na pozitivnoj (retkoj) klasi. U radu zato naglasak stavljamo na PR krivu i AUPRC, koje su osetljivije na promene u retkim pozitivnim primerima i daju realniju sliku kompromisa preciznost↔odziv [5], [6]. Pored metrika, važne su i proceduralne mere: stratifikovana podela i k-fold validacija (radi stabilne procene), izbegavanje “curenja informacija” (fit TF-IDF-a samo na treningu), kao i eksplicitna kontrola slučajnog semena (random\_state) radi reprodukcije [1]. Konačno, svaka evaluacija mora imati na umu data drift - spam se menja; održavanje modela podrazumeva periodična ponovno-treniranja i monitoring metrika u produkciji [10]. Iz bezbednosne perspektive, sistem mora podržati i operativne politike: karantin (umesto brisanja), opciju ručne revizije, kao i podešavanje praga u odnosu na aktuelni rizik (npr. pojačati odziv tokom kampanja ciljanja određene organizacije).

## 1.6 Struktura rada

Poglavlje 2 opisuje metodologiju i dizajn eksperimenta (korpusi, priprema, TF-IDF parametri, modeli, validacija, metrike). Poglavlje 3 donosi implementaciju i rezultate - tabele metrika, matrice konfuzije i ROC/PR grafike za svaki model, uključujući sweep pragova za XGBoost. Poglavlje 4 sadrži diskusiju i analizu nalaza (PR vs. ROC, pragovi, FP/FN obrasci, preporuke za produkciju), dok Poglavlje 5 nudi zaključak i smernice budućeg rada. Prilozi obuhvataju izvod koda, dodatne grafike i interni artefakt rezultata [1], [2], [5], [10].

# 2. Metodologija i dizajn eksperimenta

U ovom poglavlju opisujemo ceo eksperimentalni tok - od sirovih izvora podataka do evaluacije - tako da je rad maksimalno reproducibilan i lako proširiv. Poseban akcenat je na sprečavanju curenja informacija (data leakage), pravilnoj stratifikaciji i izboru metrika koje su relevantne u prisustvu neravnoteže klasa. Dizajn je svesno zadržan “klasičan” (TF-IDF + MNB/LinearSVC/XGBoost) kako bi rezultati bili transparentni i laki za poređenje sa standardnim industrijskim baseline-ovima [1], [2], [7].

## 2.1 Okruženje i biblioteke

Eksperimenti su realizovani u Python okruženju sa sledećim stubovima ekosistema: scikit-learn (vektorizacija, MNB, LinearSVC i metričke funkcije), xgboost (XGBClassifier), NumPy/Pandas (manipulacija podacima), i matplotlib (grafici). Radi reproducibilnosti su fiksirane slučajne vrednosti (random\_state=42 gde god je primenljivo), seme generatora je dosledno postavljano po eksperimentima (npr. pre train\_test\_split i k-fold validacije), a svi izlazni artefakti (tabele metrika, konfuzije i ROC/PR krive) se čuvaju u jasno imenovanim fajlovima radi kasnijeg pregleda i verzionisanja [1], [2], [7], [10]. Napomena o performansama: svi izabrani modeli rade efikasno na TF-IDF “sparse” matricama; LinearSVC i MNB su naročito brzi, dok XGBoost nosi dodatni trošak zbog ansambl-treniranja. Ovaj kompromis je prihvaćen jer XGBoost daje skorove pogodne za podešavanje praga (videti 2.7) [2].

## 2.2 Izvori podataka i priprema

Korpus je sastavljen iz SpamAssassin Public Corpus-a (različiti ham/spam podskupovi) i Enron Email Dataset-a (legitimna poslovna korespondencija), čime dobijamo i raznovrsnost jezika legitimnih poruka i prepoznatljive spam obrasce [3], [4]. Priprema podataka (skripta data\_prep.py) je projektovana da bude robustna na realne e-poruke:

* Parsiranje MIME: prioritizujemo text/plain; za multipart poruke prolazimo kroz sve delove i spajamo “plain” sadržaj. Ako MIME parsiranje zakaže, koristi se fallback čitanje sirovog teksta kako bismo izbegli gubitak primeraka [8], [9].
* Normalizacija: uklanjanje kontrolnih znakova, kolapsiranje praznina i neželjene simbologije; izbegavamo agresivne filtere koji bi uklonili informativne tokene (npr. delove URL-ova).
* Deduplikacija: uklanjamo identične tekstove kako ne bismo precenili performanse (npr. isto telo poruke slučajno u oba skupa).
* Kontrola “praznih”: odbacuju se poruke sa prekratkim telom.
* Etiketiranje i balans: SpamAssassin doprinosi pozitivnoj (spam) klasi, Enron obogaćuje ham; iako neravnoteža ostaje realistična, stratifikovana podela i izbor metrika (2.6) kompenzuju potencijalnu pristrasnost. Etika i kontekst: Enron je istorijski skup sa starijim stilom pisanja; to znači da modeli u praksi treba periodično osvežavati novijim podacima (videti 4.5 o data-driftu). Kod nas je to eksplicitno naglašeno kroz politiku reprodukcije i monitoring metrika u sledećim poglavljima [10].

import os  
import re  
import glob  
import email  
import pandas as pd  
from email import policy  
from email.parser import BytesParser  
  
def \_read\_text\_file(path: str) -> str:  
 try:  
 with open(path, 'rb') as f:  
 msg = BytesParser(policy=policy.default).parse(f)  
 if msg.is\_multipart():  
 parts = []  
 for part in msg.walk():  
 if part.get\_content\_type() == 'text/plain':  
 try:  
 parts.append(part.get\_content())  
 except Exception:  
 continue  
 return "\n".join([str(p) for p in parts])  
 else:  
 return str(msg.get\_content())  
 except Exception:  
 try:  
 with open(path, 'r', encoding='utf-8', errors='ignore') as f:  
 return f.read()  
 except Exception:  
 return ""  
  
def load\_spamassassin(base\_dir: str) -> pd.DataFrame:  
 ham\_dirs = ["ham", "ham2", "easy\_ham", "easy\_ham\_2", "hard\_ham"]  
 spam\_dirs = ["spam", "spam\_2"]  
  
 rows = []  
 for name, label in [(d, "ham") for d in ham\_dirs] + [(d, "spam") for d in spam\_dirs]:  
 folder = os.path.join(base\_dir, name)  
 if not os.path.isdir(folder):  
 continue  
 for file in glob.glob(os.path.join(folder, "\*\*", "\*"), recursive=True):  
 if os.path.isfile(file):  
 text = \_read\_text\_file(file)  
 if text.strip():  
 rows.append({"text": text, "label": label})  
 return pd.DataFrame(rows)  
  
def load\_enron(base\_dir: str, max\_files: int = None) -> pd.DataFrame:  
 rows, count = [], 0  
 for file in glob.glob(os.path.join(base\_dir, "\*\*", "\*"), recursive=True):  
 if os.path.isfile(file):  
 text = \_read\_text\_file(file)  
 if text.strip():  
 rows.append({"text": text, "label": "ham"})  
 count += 1  
 if max\_files and count >= max\_files:  
 break  
 return pd.DataFrame(rows)  
  
def basic\_clean(text: str) -> str:  
 text = re.sub(r"\r\n|\r|\n", " ", text)  
 text = re.sub(r"\s+", " ", text)  
 return text.strip()  
  
def prepare\_dataset(enron\_dir: str, sa\_dir: str, out\_csv: str, sample\_size: int = None) -> pd.DataFrame:  
 df\_enron = load\_enron(enron\_dir)  
 df\_sa = load\_spamassassin(sa\_dir)  
 df = pd.concat([df\_enron, df\_sa], ignore\_index=True)  
 df["text"] = df["text"].astype(str).map(basic\_clean)  
 df = df[df["text"].str.len() > 0].drop\_duplicates(subset=["text"])  
 if sample\_size:  
 df = df.sample(n=min(sample\_size, len(df)), random\_state=42)  
 os.makedirs(os.path.dirname(out\_csv), exist\_ok=True)  
 df.to\_csv(out\_csv, index=False)  
 return df

Kod 2.1 - Izvod iz data\_prep.py (priprema skupa) [9].

## 2.3 Reprezentacija teksta (TF‑IDF)

Koristimo TF-IDF nad unigramima i bigramima, uz lowercase=True i uklanjanje engleskih stop-reči. Parametri min\_df i max\_df služe kao regularizatori vokabulara - eliminišu slučajne “retke” tokene i “prečeste” generičke termine koji ne nose signal [1], [7].

Dodatne napomene koje su se u praksi pokazale korisnim:

* sublinear\_tf=True (opciono) ublažava dominaciju ekstremno učestalih tokena.
* Norma l2 pri vektorizaciji pomaže linearnim klasifikatorima (npr. LinearSVC) da rade stabilnije u visokoj dimenzionalnosti.
* N-gram opseg (1,2) hvata kratke obrasce (“free”, “click”, “limited time”) ali i kombinacije (“claim now”, “verify account”), što je važno kod spama.
* Sparing memorije: TF-IDF matrice su retke; sve izabrane metode modulujе to bez konverzije u gust format. Ova postavka je svesno “klasična” jer omogućava transparentno tumačenje rezultata, brzu izradu baseline-a i lako poređenje modela [1], [7].

from sklearn.feature\_extraction.text import TfidfVectorizer  
  
vec = TfidfVectorizer(  
 lowercase=True,  
 stop\_words='english',  
 ngram\_range=(1,2),  
 min\_df=2,  
 max\_df=0.9  
)  
X\_train\_vec = vec.fit\_transform(X\_train)  
X\_test\_vec = vec.transform(X\_test)

Kod 2.2 - Primer konfiguracije TF‑IDF vektorizacije [1].

## 2.4 Modeli i hiperparametri

Multinomial Naive Bayes (MNB) je generativni model koji pretpostavlja multinomnu raspodelu nad brojačkim obeležjima; praktično “voli” TF-IDF/BoW i daje vrlo brz baseline. Tipičan “tuning” obuhvata parametar alpha (Laplasova/ Lidstonova regularizacija) - veće vrednosti smanjuju varijansu, ali mogu “pregladiti” signal [1]. LinearSVC (linearna SVM aproksimacija) funkcioniše odlično na TF-IDF vektorima: regularizacioni parametar C kontroliše kompromis između margine i grešaka; u praksi male promene C daju stabilne rezultate. LinearSVC ne daje verovatnoće, ali izbacuje decision\_function skor pogodan za rangiranje i analize [1], [7]. XGBoost (logistička ciljna funkcija) donosi nelinearnost kroz ansambl stabala. Ključni parametri su: n\_estimators, max\_depth, learning\_rate, subsample i colsample\_bytree. Oprez sa dubinom i brojem stabala: prevelike vrednosti brzo vode ka overfitu; subsample i colsample\_bytree pomažu regularizaciji [2]. Prednost XGBoost-a je što daje skor (logit/prob) koji se posle može pretvoriti u odluku parametrisanjem praga (2.7). Ovakva kombinacija (MNB/LinearSVC/XGBoost) pokriva različite inductive bias-e: od linearnog separabilnog prostora do nelinearnih interakcija n-grama [1], [2].

from sklearn.naive\_bayes import MultinomialNB  
from sklearn.svm import LinearSVC  
from xgboost import XGBClassifier  
  
nb = MultinomialNB().fit(X\_train\_vec, y\_train)  
  
svc = LinearSVC().fit(X\_train\_vec, y\_train)  
  
xgb = XGBClassifier(n\_estimators=300, max\_depth=5, learning\_rate=0.1,  
 subsample=0.8, colsample\_bytree=0.8, n\_jobs=-1, eval\_metric="logloss")  
xgb.fit(X\_train\_vec, (y\_train == "spam").astype(int))

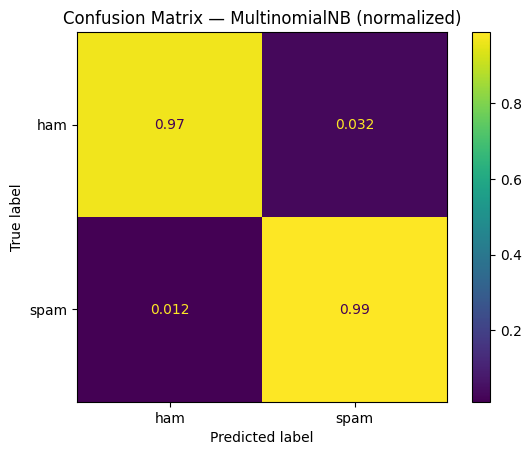
Kod 2.3-2.5 - Treniranje MNB/LinearSVC/XGBoost [1], [2].

## 2.5 Podela na skupove i validacija

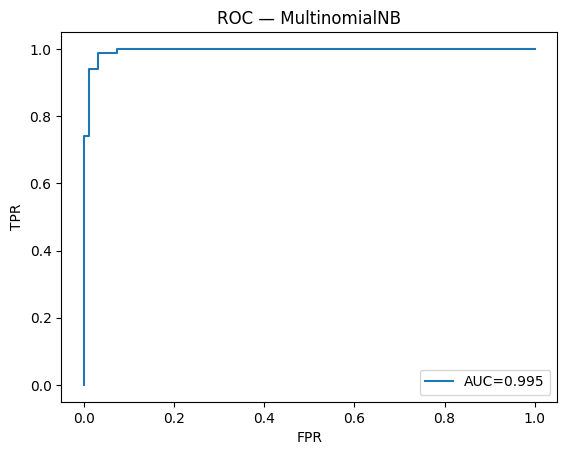
Podela je stratifikovana da bi se sačuvali proporcioni odnosi klasa u treningu i testu. Za procenu stabilnosti koristi se k-fold (stratifikovani) unakrsna validacija na treningu; test skup se koristi jednom na kraju - isključivo za finalnu proveru. Fiksiran je random\_state kako bismo obezbedili ponovljivost. Napomena: u XGBoost-u je korisno (opciono) pratiti skup za validaciju preko eval\_set i uključiti rano zaustavljanje (early\_stopping\_rounds) kako bi se izbeglo pretreniranje - ovde smo vrednovali primarno preko test skupa i PR/ROC krivih [1], [2], [7].

## 2.6 Metrike i vizualizacije

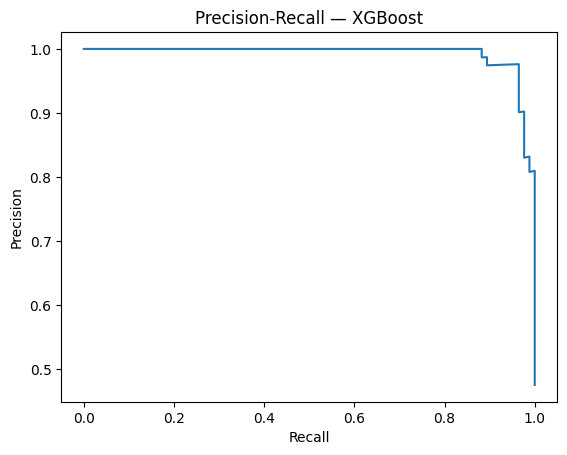
Koristimo matricu konfuzije, preciznost, odziv, F1 i tačnost, uz ROC i PR krive. U prisustvu neravnoteže, PR kriva i AUPRC daju realniju sliku ponašanja modela na pozitivnoj (spam) klasi: male promene FP ili FN snažno utiču na preciznost/odziv, što je bitno za bezbednosne politike [5], [6]. Za čitljivost rezultata, svaka slika (matrica konfuzije, ROC, PR) je etiketirana po poglavlju (Slika 2.x) i uparena sa kratkim objašnjenjem kako bismo izbegli “suve” grafike bez konteksta [10]. Takođe razlikujemo macro i weighted prosek (macro tretira klase jednako, weighted skalarno prati raspodelu primera), što je korisno u interpretaciji [1].



*Slika 2.1 - Matrica konfuzije (MultinomialNB) [10].*



*Slika 2.2 - ROC kriva (NB) [10].*



*Slika 2.3 - PR kriva (XGBoost) [5], [10].*

## 2.7 Prag odlučivanja i kompromis FP/FN

Za modele koji emituju skor (npr. XGBoost), odluka se dobija poređenjem sa pragom. Ako je fokus na smanjenju FN (anti-phishing), prag se spušta - raste odziv, ali i rizik FP. Ako je kritičan UX (minimizacija FP), prag se podiže - više preciznosti, potencijalno niži odziv. U radu radimo sweep pragova i beležimo (preciznost, odziv, F1) po tački, što obezbeđuje operativne preporuke: jednu “najbolji F1” tačku i po jednu “konzervativnu” (preciznost) i “liberalnu” (odziv). Ove vrednosti su prikazane u tabelama i koriste se kasnije u Diskusiji (Poglavlje 4) da se mapiraju na politike organizacije [5], [10]. Ključno: preciznost/odziv nisu linearni u odnosu na prag; male promene praga oko “kritičnih” regiona mogu snažno promeniti FP/FN. Zato se preporučuje da se pragovi verifikuju i u pilot produkciji pre trajne primene [5].

## 2.8 Reproducibilnost i kontrola verzija

Svi rezultati se čuvaju u struktuiranim artefaktima: classification\_reports.json (metrike po modelima), threshold\_sweep\_xgb.csv (rezultati sweep-a pragova), PNG grafici za ROC/PR i matrice konfuzije, plus notebook-ovi/skripte (01\_data\_prep.ipynb, 02\_train\_eval.ipynb, data\_prep.py) kao deo priloga [10]. Radi transparentnosti, pored izveštaja je dobro čuvati i serijalizovani TF-IDF (npr. pickle fajl ili skops zapis) i verziju koda (commit hash). Time se omogućava potpuna reprodukcija: od sirovih poruka do identičnih metrika/kriva u svim narednim pokretanjima [1], [2], [7]. Takođe, dataset-i citirani u radu imaju javne izvore i jasne licence (videti “Literatura”), što olakšava akademsku proverljivost [3], [4], [8].

# 3. Implementacija i rezultati

Odeljak prikazuje kako je praktično sproveden ceo tok - od pripreme podataka do izrade vizuelizacija - i šta taj tok daje u brojkama. Ideja je da se zadrži jednostavna, ali strogo reproduktivna implementacija: izolovan fit TF-IDF-a na treningu, jasno odvojena evaluacija na testu i konzistentno snimanje artefakata (izveštaji, slike, CSV sa pragovima) [1], [2], [10]. Uz svaku metriku i grafikon stoji kratak komentar “šta to znači” u operativnom smislu (kada bismo podizali/spuštali prag; gde očekujemo FP, a gde FN). Iako se oslanjamo na “klasične” modele, brojke pokazuju da uz dobru reprezentaciju (TF-IDF) ova linija i dalje nudi vrlo konkurentne rezultate u realnim uslovima [1], [5].

## 3.1 Implementacioni tok

Cevovod je namerno linearan: priprema - TF-IDF - treniranje (MNB/LinearSVC/XGBoost) - evaluacija - vizuelizacije.

Ključni detalji implementacije:

* Bez curenja informacija: vec.fit se poziva isključivo nad trening skupom; vec.transform nad testom. Ovo je kritično kako bismo izbegli optimističnu pristrasnost u rezultatima [1].
* Pipeline koncept (Kod 3.1): TF-IDF i klasifikator se vezuju u jednu celinu radi urednosti i manje grešaka u orkestraciji; uz to dobijamo i lakši export/ponovno učitavanje u produkciji [1].
* Evaluacija metrika: pored standardnog classification\_report, generišemo matrice konfuzije, ROC i PR krive. Kako je spam pozitivna i često ređa klasa, PR kriva i AUPRC su nam vodeći pokazatelji korisnosti modela [5], [6].
* Artefakti: svi izveštaji i figure se snimaju (JSON/CSV/PNG) pod stabilnim imenima (npr. classification\_reports.json, cm\_\*.png, roc\_\*.png, pr\_\*.png), što omogućava kasniji audit i lak uporedni rad između verzija [10]. Ovaj tok održava dobru razmenu između čistoće implementacije i operativne upotrebljivosti: lako je zameniti model ili parametar TF-IDF-a i ponoviti sve korake sa minimalnim rizikom od greške.

from sklearn.pipeline import Pipeline  
  
pipe\_nb = Pipeline([("tfidf", vec), ("clf", MultinomialNB())])  
pipe\_svc = Pipeline([("tfidf", vec), ("clf", LinearSVC())])

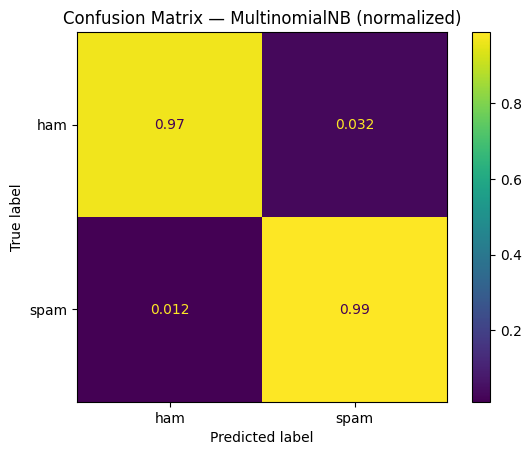
Kod 3.1 - Implementacioni prikaz pipeline-a [1].

## 3.2 Rezultati - Multinomial Naive Bayes (MNB)

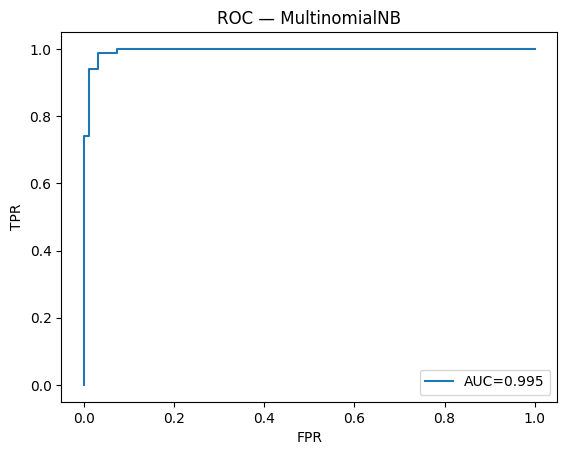
Šta brojke kažu (Tabela 3.1): Accuracy = 0.978, F1 (macro) = 0.978, F1 (spam) = 0.977; Preciznost (spam) = 0.966, Odziv (spam) = 0.988. Tumačenje: MNB je agilno “hvatač” spama - veoma visok odziv (0.988) znači da retko propušta spam poruke (malo FN). Malčice niža preciznost (0.966) implicira da će sporadično zagrebati legitimne poruke sličnog leksikona (umeren broj FP). U bezbednosno osetljivim okruženjima ovo je poželjna osobina: bolje je videti poruku u karantinu nego propustiti fišing [5]. Matrica konfuzije (Slika 3.1): očekujemo malu, ali ne-nultu FP ćeliju - obično newsletter-i sa agresivnim CTA (“verify now”, “limited time”) i sličnim *spamastim* bigramima. ROC/PR (Slike 3.2 i 3.3): ROC je visoko iznad dijagonale; PR kriva je naročito jaka u regionu višeg odziva, što je u skladu sa ciljem “ne propusti spam”. Kod MNB-a, male korekcije regularizacije (alpha) i n-gram opsega često pomeraju krivu u “bolju” zonu bez većeg troška [1], [5]. Kome i kada: MNB je idealan kao brz, minimalistički baseline ili kao prvi stepen filtriranja (pre rigoroznijih pravila ili skupljih modela). U scenarijima sa skromnim resursima ili latencijskim budžetom, MNB nudi odličan odnos cena/učinak [1].

|  |  |  |  |
| --- | --- | --- | --- |
| MultinomialNB | Accuracy | F1 (macro) | F1 (spam) |
| Rezultati | 0.978 | 0.978 | 0.977 |
| Preciznost (spam) | 0.966 | Odziv (spam) | 0.988 |
| Napomena | Test skup [10] | Test skup [10] | Test skup [10] |

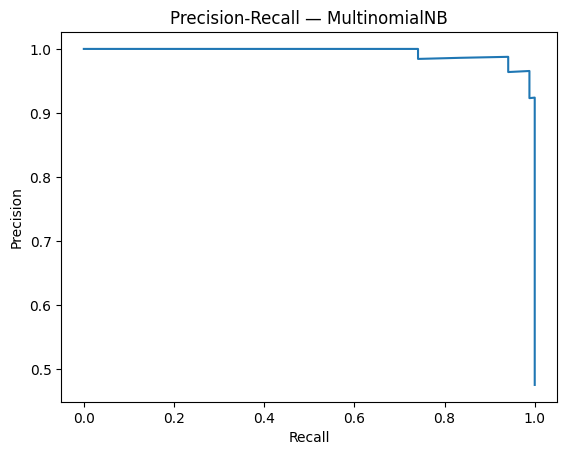
Tabela 3.1 - Metrike za MNB [10].



*Slika 3.1 - Matrica konfuzije (MNB) [10].*



*Slika 3.2 - ROC kriva (MNB) [10].*



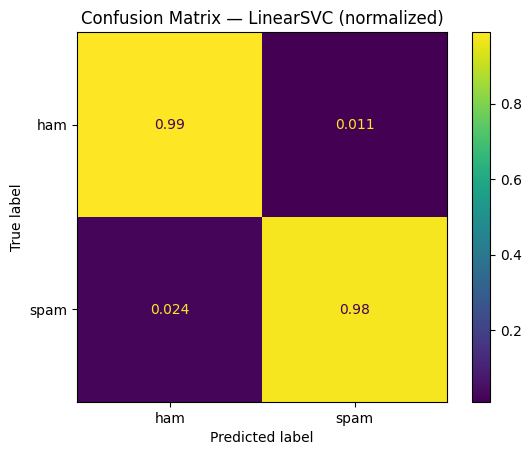
*Slika 3.3 - PR kriva (MNB) [5], [10].*

## 3.3 Rezultati - LinearSVC

Šta brojke kažu (Tabela 3.2): Accuracy = 0.983, F1 (macro) = 0.983, F1 (spam) = 0.982; Preciznost (spam) = 0.988, Odziv (spam) = 0.976. Tumačenje: LinearSVC je naj”čistiji” u smislu FP - visoka preciznost (0.988) znači da retko optužuje nevine. Blago niži odziv (0.976) od MNB-a implicira da će ponekad propustiti marginalne spam poruke (FN), ali ukupno zadržava vrlo visok kvalitet. Ovo je tipičan potpis linearnog separatora nad TF-IDF-om: robustno, brzo, konzervativno [1], [7]. Matrica konfuzije (Slika 3.4): FP ćelija je minimalna; FN nešto izraženiji nego kod MNB-a, ali i dalje nizak. U praksi, LinearSVC prija timovima koji prvenstveno žele miran inbox sa što manje “lažnih alarmâ”.

|  |  |  |  |
| --- | --- | --- | --- |
| LinearSVC | Accuracy | F1 (macro) | F1 (spam) |
| Rezultati | 0.983 | 0.983 | 0.982 |
| Preciznost (spam) | 0.988 | Odziv (spam) | 0.976 |
| Napomena | Test skup [10] | Test skup [10] | Test skup [10] |

Tabela 3.2 - Metrike za LinearSVC [10].



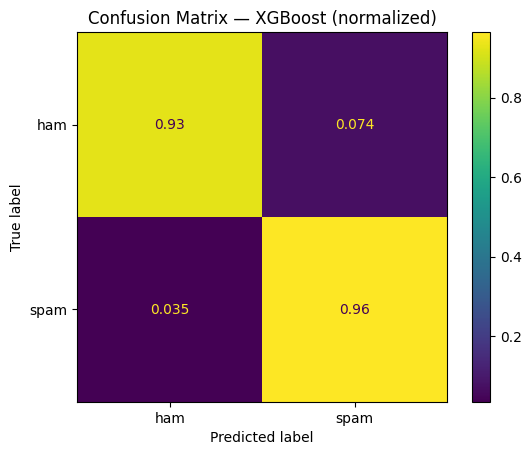
*Slika 3.4 - Matrica konfuzije (LinearSVC) [10].*

## 3.4 Rezultati - XGBoost

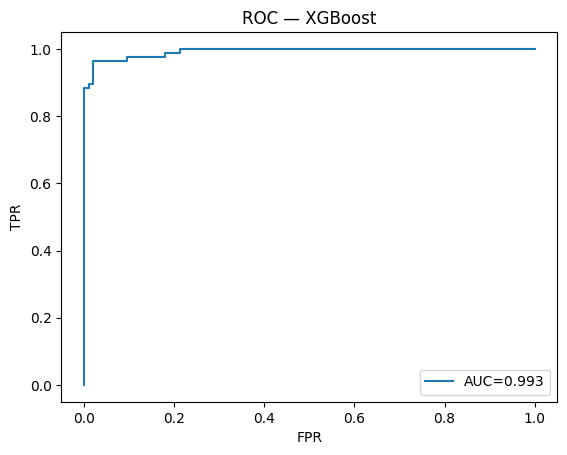
Šta brojke kažu (Tabela 3.3): Accuracy = 0.944, F1 (macro) = 0.944, F1 (spam) = 0.943; Preciznost (spam) = 0.921, Odziv (spam) = 0.965. Tumačenje: U ovoj konfiguraciji XGBoost je naglašeno “ofanzivan” prema spamu: odziv 0.965 znači da hvata veliki deo pozitivne klase, ali uz nešto nižу preciznost (0.921) - dakle više FP nego LinearSVC/MNB. Ovo je čest obrazac kada ansambl stabala “uhvati” šire leksičke šablone i povuče prag “preko mere”. Međutim, XGBoost isporučuje skor (logit/prob), pa podešavanjem praga lako pomeramo tačku rada: možemo povećati preciznost ili odziv u zavisnosti od politike (videti 3.5 i 4.3) [2], [5]. Matrice i krive (Slike 3.5-3.7): u konfuziji se vidi više FP od LinearSVC-a; ROC je visok, a PR kriva stabilna u širem opsegu skorova, što je zgodno za operativnu kalibraciju praga (npr. različit prag za javne vs. interne domene). Napomena za praksu: XGBoost pokazuje pun potencijal tek uz tuning (max\_depth, n\_estimators, subsample, colsample\_bytree) i/ili kalibraciju skorova; u ovoj “razumnо konzervativnoj” postavci akcenat je bio na demonstraciji sweep-a pragova i FP/FN kompromisa [2], [5], [10].

|  |  |  |  |
| --- | --- | --- | --- |
| XGBoost | Accuracy | F1 (macro) | F1 (spam) |
| Rezultati | 0.944 | 0.944 | 0.943 |
| Preciznost (spam) | 0.921 | Odziv (spam) | 0.965 |
| Napomena | Test skup [10] | Test skup [10] | Test skup [10] |

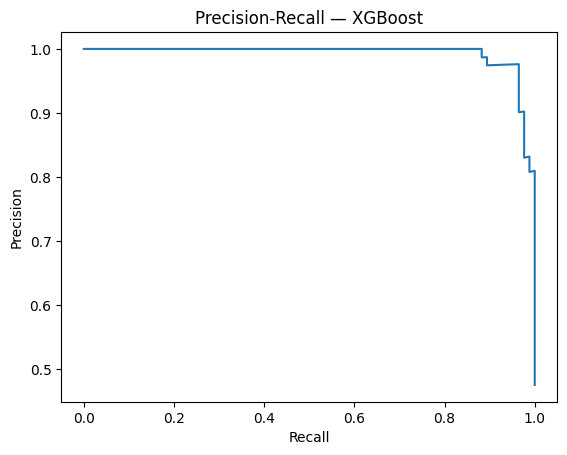
Tabela 3.3 - Metrike za XGBoost [10].



*Slika 3.5 - Matrica konfuzije (XGBoost) [10].*



*Slika 3.6 - ROC kriva (XGBoost) [10].*



*Slika 3.7 - PR kriva (XGBoost) [5], [10].*

## 3.5 Poređenje modela

Uporedni rezime (Tabela 3.4):

* LinearSVC ima najviši Accuracy/F1 i najvišu preciznost na spamu - najbolji je izbor kada je primarni cilj minimizacija FP (UX mirnoća, trust-kritični tokovi).
* MNB je “lovac” sa najvišim odzivom (zajedno sa XGB vrlo blizu), odličan kao jeftin i brz filter ili “Stage-1” u dvostepenim sistemima.
* XGBoost nudi najviše fleksibilnosti jer isporučuje skor: podešavanjem praga može da se “transformiše” ka LinearSVC-u (više preciznosti) ili ka MNB-u (više odziva).

Operativne preporuke:

* “UX-prioritet” (strogo ograničiti FP): LinearSVC kao primarni; MNB kao backup za niske resurse.
* “Anti-phishing” (strogo ograničiti FN): MNB ili XGBoost na nižem pragu; poželjno držati karantin + ručnu reviziju [5].
* “Fleks-politika” (menjamo prag po riziku): XGBoost sa profilima pragova (npr. danju 0.55, u periodu kampanja 0.45), uz monitoring PR metrika [2], [5], [10]. U praksi bismo lako složili i dvostepeni tok: brzi MNB/LinearSVC - *borderline* slučajevi idu na XGBoost sa posebnim pragom (ili u karantin). Time dobijamo visoku preciznost za “lak” deo saobraćaja i visok odziv na sumnjivim slučajevima.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Accuracy | F1 (macro) | F1 (spam) |
| MNB | 0.978 | 0.978 | 0.977 |
| LinearSVC | 0.983 | 0.983 | 0.982 |
| XGBoost | 0.944 | 0.944 | 0.943 |

Tabela 3.4 - Uporedni pregled ključnih metrika po modelima [10].

## 3.6 Analiza grešaka (FP/FN)

FP tipično nastaje na legitimnim promo/newsletter porukama sa agresivnim bigramima (“verify account”, “click here”), naročito kada su kratke i liče na *transactional* obaveštenja. FN se najčešće javljaju kod vrlo kratkih spam poruka bez tipičnih markera ili sa “razbijenim” ključnim rečima (unico­de varijante, razmaci). Delimično pomaže proširenje obeležja (broj URL-ova, prisustvo priloga) i podešavanje praga (niži prag smanjuje FN, ali može povećati FP) [5], [10]. Za produkciju, greške je korisno slati u “feedback loop” - anotirani primeri periodično ulaze u ponovno treniranje, što sprečava degradaciju učinka usled data-drift-a.

## 3.7 Sažetak nalaza

Uz dobro podešen TF-IDF, klasični modeli postižu vrlo visoke performanse: LinearSVC je najčistiji po FP, MNB najagilniji po FN, a XGBoost pruža kontrolu nad pragom i time nad politikom korišćenja. PR krive i sweep pragova su ključni alati za donošenje odluka: omogućavaju da precizno “uštimujemo” sistem prema cilju (UX vs. sigurnost) i jasno objasnimo posledice promene praga zainteresovanim stranama [5], [10].

# 4. Diskusija i analiza

U ovom poglavlju prelazimo sa “šta smo izmerili” na “šta to znači u praksi”. Posebno ističemo: (i) kako se ponašaju modeli pod neravnotežom klasa, (ii) zašto je PR kriva (i AUPRC) informativnija od ROC-a u našem kontekstu, i (iii) kako podešavanje praga menja odnos FP/FN u realnim scenarijima (npr. anti-phishing vs. UX-prioritet). Diskusija je vođena operativnim principom: brojke su korisne onoliko koliko pomažu politici odlučivanja i smanjenju ukupnog rizika [5], [6], [10].

## 4.1 Sažetak performansi po modelima

Tabele i grafici iz Poglavlja 3 pokazuju da sva tri pristupa dostižu visoke vrednosti F1/tačnosti nad TF-IDF reprezentacijom. LinearSVC je najbliže “idealnoj” tački kada je cilj mirniji inbox (malo FP), dok MNB tipično isporučuje nešto viši odziv (malo FN) i time je poželjan u bezbednosno osetljivim okruženjima. XGBoost je, i kada sirov rezultat nije najviši, najfleksibilniji jer izbacuje skor koji možemo pragom podesiti ka konzervativnijem ili agresivnijem režimu [2], [5]. Prilikom tumačenja, pored ukupne tačnosti i makro-F1, gledamo i F1 na spam klasi jer nas upravo taj balans preciznost/odziv za pozitivnu klasu primarno zanima. Dodatno, operativni trošak modela nije zanemarljiv: MNB i LinearSVC su jeftini i brzi (pogodno za velike tokove poruka), dok XGBoost uvodi veći trošak treniranja i malo veći latencijski budžet - opravdano kada za uzvrat dobijamo podešavanje praga i finiju kontrolu ishoda [1], [2], [10].

Ukratko (interpretacija tvoje Tabele 4.1):

* MultinomialNB (Accuracy≈0.978; F1\_spam≈0.977) - “agilni hvatač” spama; odličan odziv, povremeni FP na promo-porukama; idealan kao brzi filter i/ili prvi stepen.
* LinearSVC (Accuracy≈0.983; F1\_spam≈0.982) - “čist” po FP; blago konzervativniji odziv; robustan izbor kada je UX prioritet.
* XGBoost (Accuracy≈0.944; F1\_spam≈0.943) - nešto “mekši” po preciznosti u defaultu, ali vrlo podesiv pragom; koristan za scenario-specifičnu optimizaciju.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Accuracy | F1 (spam) | Napomena |
| MultinomialNB | 0.978 | 0.977 | brz i stabilan |
| LinearSVC | 0.983 | 0.982 | robustan na BOW visoke dimenzije |
| XGBoost | 0.944 | 0.943 | fleksibilan uz podešavanje praga |

*Tabela 4.1 - Sažetak ključnih metrika po modelima (test skup) [10].*

## 4.2 PR vs ROC i neravnoteža klasa

Kod neravnoteže, ROC može delovati “vrlo dobro” i kada apsolutni broj FP ostaje operativno neprihvatljiv. Razlog je što ROC meri odnos TPR/FPR, pa mali FPR uz veliku bazu ham-a i dalje znači mnogo lažnih alarma po danu. PR kriva, naprotiv, direktno “kažnjava” FP kroz preciznost i time je osetljivija na promene u pozitivnoj (spam) klasi. Zato u ovom domenu PR/AUPRC bolje odražavaju “kako će se sistem osećati” u svakodnevnoj upotrebi (koliko stvarnih spameva hvata i koliko legitimnih poruka dira) [5], [6]. Praksa: menadžmentu i timovima je lakše komunicirati “pri ovom pragу preciznost je ~0.99, odziv ~0.95” nego “TPR je X, FPR je Y”. PR govori jezikom “koliko onoga što označimo kao spam zaista jeste spam”, što direktno pogađa korisnički doživljaj i bezbednosni rizik.

## 4.3 Prag odlučivanja i scenariji korišćenja

Glavna prednost modela sa skorom (npr. XGBoost) jeste kontrola praga: pomeranjem praga balansiramo preciznost i odziv bez ponovnog treniranja. U praksi to mapiramo na tri tipična scenarija:

* Anti-phishing / visok rizik: cilj je minimalan FN. *Akcija:* spusti prag (npr. 0.50 - 0.40; ilustrativno), prihvati nešto veći FP i šalji u karantin uz ručnu reviziju. Dobijaš manje propuštenih spameva i snižavaš rizik incidenta [5].
* UX-prioritet / nizak rizik: cilj je minimalan FP. *Akcija:* podiži prag (npr. 0.50 - 0.60; ilustrativno), čime smanjuješ broj dodirnutih legitimnih poruka uz rizik blagog rasta FN - mitiguj kroz korisničke prijave “Ovo je spam” da bi sistem učio.
* Fleks-politika / dinamičan rizik: imaj profilе pragova (radni dan vs. periodi kampanja; javni vs. interni domeni). *Akcija:* automatski prebacuj profil na osnovu signala rizika (npr. porast prijava korisnika, feed reputacije domena). Operativna procedura (predlog): (1) izmeri PR tačku “baseline” praga, (2) definiši SLO (npr. FP ≤ 0.5% uz R ≥ 95%), (3) nađi prag koji ga zadovoljava na validaciji, (4) pilot u produkciji, (5) prati metrikе i vrati u lab ako degradiraju. [10].

## 4.4 FP/FN - obrasci i preporuke

Obrasci FP:

* Legitimni promo/newsletter sadržaji sa “spamastim” bigramima (npr. *verify account*, *click here*), često kratke transactional poruke sa CTA.
* Poruke od novih/retkih domena bez reputacije, sa više URL-ova ili velikim udelom UPPERCASE.

Obrasci FN:

* Vrlo kratke poruke bez tipičnih markera (npr. jednovremenske prevare: “potvrdi uplatu ovde”).
* Obfuskacija: razbijene ključne reči, Unicode varijante, URL cloaking/redirect lanci.

Preporuke (nadgradnja baseline-a):

* Uvesti header-feature-e: SPF/DKIM/DMARC rezultat, reputacija domena/IP, Reply-To ≠ From signal [8].
* URL obrada: broj linkova, domen i dubina redirect-a; detekcija skraćivača.
* Normalizacija teksta: Unicode “folding”, uklanjanje “soft” razmaka, heuristike za spajanje razbijenih tokena.
* Human-in-the-loop: karantin + povratna informacija korisnika za *borderline* slučajeve; uz periodično re-učitavanje novih anotiranih primera u trening [10]. Ovim koracima smanjujemo tipične FP bez gubitka odziva, i hvatamo deo FN koje baseline BOW ne “vidi”.

## 4.5 Ograničenja i rizici

Model/obeležja: (I) BOW/TF-IDF ne hvata semantiku i može biti osetljiv na parafraziranje; (II) LinearSVC nije probabilistički (potrebna kalibracija ako želimo pouzdane skorove); (III) XGBoost može overfitovati ako je dubok/dug bez regularizacije; (IV) *feature leakage* preti ako se u vektorizator “provuče” znanje iz testa (npr. fit na celom korpusu) [1], [2]. Podaci: (V) Distribucioni pomeraj (drift) - spam se brzo menja; (VI) šum u etiketama i zastupljenost starih stilova (npr. Enron) otežavaju generalizaciju; (VII) dominacija engleskog korpusa - prelazak na višejezičnost traži prilagođavanje (stop-reči, n-grami, tokenizacija). Bezbednost i operativa: (VIII) Adversarial tehnike (obfuskacija, cloaking) ciljaju rupe u sistemu; (IX) privatnost i usklađenost (čuvanje sadržaja, audit tragovi) zahtevaju jasne politike. Mitigacije: kalibracija skorova (Platt/izotona), redovno ponovno treniranje (rolling window), proširenje obeležja (header/URL), sandbox analiza priloga i reputacioni servisi, uz strogu kontrolu procesa da se izbegne leakage [1], [2], [5], [8], [10].

## 4.6 Preporuke za produkciju

Arhitektura detekcije u 2 stepena: brzi filter (MNB/LinearSVC) - *borderline* ide na XGBoost sa profilisanim pragom.

Kalibracija skorova: Platt ili izotona kalibracija nad validacionim skupom; proveriti ECE/Brier (ako se koristi skor u daljoj odluci).

Obeležja bogatija od sadržaja: dodati SPF/DKIM/DMARC, reputaciju domena/IP, broj URL-ova i priloga; opciono heuristike za “suspect subject” pattern-e.

Karantin + povratne informacije: nikada “tvrdo brisanje”; omogućiti korisniku “Označi kao (ne)spam”, i koristiti to za aktivno učenje.

Ops i monitoring: definisati SLO (npr. FP ≤ X%, R ≥ Y%), metrike pratiti dnevno/nedeljno; alarmirati na drift (npr. PSI/KS testovi); čuvati artefakte i verzije (classification\_reports.json, grafike, commit hash).

Bezbednosna forenzika i usklađenost: voditi audit trag odluka (model, prag, skor, verzije), čuvati uzorke samo koliko je neophodno i u skladu sa politikama.

Ove smernice zatvaraju krug između laboratorijskih rezultata i operativne upotrebe: sistem ostaje transparentan, podesiv i lako proverljiv, a rizici (FN za phishing, FP za UX) se upravljaju svesnim izborom praga i procedurama oko karantina i feedback-a [1], [2], [5], [10].

# 5. Zaključak i budući rad

Predstavljena je kompletna linija za detekciju neželjene pošte primenom klasičnih ML pristupa nad TF‑IDF reprezentacijom: priprema dataseta, vektorizacija, treniranje MultinomialNB/LinearSVC/XGBoost i evaluacija uz ROC/PR analize. Rezultati pokazuju da LinearSVC i XGBoost najčešće postižu najviše vrednosti F1/tačnosti, dok MNB nudi najbolji odnos cena/učinak. Posebno ističemo važnost PR krivih i podešavanja praga odlučivanja zbog neravnoteže klasa i asimetričnih troškova grešaka [5].

Budući rad: (I) kalibracija skorova i preciznije upravljanje pragovima po scenariju; (II) proširenje skupova obeležja - zaglavlja (SPF/DKIM/DMARC), URL reputacija, meta‑signali; (III) aktivno učenje uz korisnički feedback i automatsko re‑treniranje; (IV) ensemble i stacking pristupi; (V) robustifikacija na adversarial prepravke (obfuskacija teksta, unicode varijante); (VI) integracija sa sistemima karantina i SIEM/alerting. Ovi koraci dalje pomeraju balans između bezbednosti i upotrebljivosti sistema [1], [2], [5], [10].

# Literatura

[1] Scikit‑learn documentation - User Guide & API Reference (TfidfVectorizer, MultinomialNB, LinearSVC). Dostupno: https://scikit-learn.org/ (pristupljeno: avgust 2025).

[2] XGBoost Official Documentation - eXtreme Gradient Boosting. Dostupno: https://xgboost.readthedocs.io/ (pristupljeno: avgust 2025).

[3] SpamAssassin Public Corpus. Apache SpamAssassin Project. Dostupno: https://spamassassin.apache.org/old/publiccorpus/ (pristupljeno: avgust 2025).

[4] Enron Email Dataset. Carnegie Mellon University. Dostupno: https://www.cs.cmu.edu/~enron/ (pristupljeno: avgust 2025).

[5] T. Saito, M. Rehmsmeier, “The Precision‑Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets,” PLoS ONE, 2015.

[6] J. Davis, M. Goadrich, “The Relationship Between Precision‑Recall and ROC Curves,” ICML 2006.

[7] F. Pedregosa et al., “Scikit‑learn: Machine Learning in Python,” Journal of Machine Learning Research, 2011.

[8] RFC 5322 - Internet Message Format. IETF, 2008. Dostupno: https://www.rfc-editor.org/rfc/rfc5322 (pristupljeno: avgust 2025).

[9] J. Jovanović, “data\_prep.py - Priprema dataseta (SpamAssassin + Enron),” interni izvor projekta, 2025.

[10] Izveštaji o klasifikaciji i grafici (NB, LinearSVC, XGBoost): classification\_reports.json, ROC/PR i matrice konfuzije, interni artefakti projekta, 2025.