Primjena modela strojnog učenja za predviđanje očekivanih prinosa dionica u RH

Application of Machine Learning Models for Predicting Expected Stock Returns in Croatia

Mislav Šagovac Luka Šikić Petra Palić

2026-06-18

Ovaj rad istražuje primjenu modela strojnog učenja za predviđanje očekivanih prinosa dionica na Zagrebačkoj burzi (ZSE) u razdoblju od 2000. do 2024. godine. Koristi se velik skup prediktora izvedenih iz dnevnih podataka o trgovanju te ugniježđena metoda križne validacije s pomičnim prozorom (NRWCV). Uspoređuju se četiri modela — Random Forest, XGBoost, GLMNET i neuronska mreža (NNET) — kao i njihovi jednostavni ansambli. Rezultati pokazuju da modeli stabala odlučivanja (Random Forest) i neuronske mreže ostvaruju najbolji odnos rizika i prinosa, a ansambli predikcija dodatno poboljšavaju uspješnost na relativno nelikvidnom i plitkom hrvatskom tržištu kapitala.

Sadržaj


```
library(data.table)
library(fs)
```

```
library(lubridate)
library(flextable)
library(ggplot2)
library(PerformanceAnalytics)
library(mlr3batchmark)
library(matrixStats)
library(mlr3misc)

# setup
set_flextable_defaults(
  decimal.mark = ",",
  big.mark = ".",
  digits = 2,
  width = 1,
  layout = "autofit"
)
gg_theme = theme(axis.title.x = element_blank())
PATH = "F:/zse/results_week"
TYPE = "predictions" # can be predictions and models
TARGETS = "target"

# Inline code variables
n = format(nrow(prices), big.mark = ".")
first_date = strftime(prices[, min(date)], format = "%d.%m.%Y.")
last_date = strftime(prices[, max(date)], format = "%d.%m.%Y.")
```

1 Uvod

Široka primjena modela strojnog učenja nije zaobišla područje investiranja i trgovanja dionicama. Sposobnost modela strojnog učenja da obrade velike skupove podataka i otkriju složene, nelinearne obrasce ponašanja učinila ih je korisnim alatima za investitore i istraživače. Ovaj rad istražuje primjenu modela strojnog učenja za predviđanje očekivanih prinosa dionica u Hrvatskoj, koristeći trgovinske podatke s Zagrebačke burze (ZSE) od 2000. do 2024. Većina istraživanja ovog tipa primjenjena su na razvijena tržišta kapitala. Budući daje hrvatsko tržište kapitala relativno nelikvidno i plitko, moguće je da modeli strojnog učenja pokazuju veću učinkovitost u predviđanju budućih povrata u odnosu na iste modele primjenjene na razvijenim tržištima kapitala. Drugim riječima, zbog većog stupnja nesavršenosti na hrvatskom tržištu kapitala, modeli strojnog učenja mogu potencijalno prepoznavati obrasce koji postoje duže nego na razvijenim

tržištima.

Osim primjene metodolgije temeljene na strojnom učenju na malo, plitko tržište, ovo istraživanje uvodi i nove prediktore u funkciju predviđanja. Koristi se veliki skup prediktora izvedenih iz dnevnih podataka o trgovanju, za sve dionice koje su kotirale na Zagrebačkoj burzi. Skup podataka podvrgava se rigoroznoj predobradi kako bi se osigurala njegova prikladnost za modele strojnog učenja. Takva pomna priprema podataka ključna je za pouzdanost predikcija strojnog učenja Pavlidis i ostali (2016). Novost je i primjena drugačijih metoda filtrianja prediktora, u odnosu na metode koje se koriste u drugim radovima. Koriste se dvije metode filtrianja. Prva metoda, JMI, je bazirana na filtriranju zajedničkih informacija. Druga metoda, relief, je bazirana na informacijskoj entropiji (Kononenko (1994)). Ove metode filtriranja prediktora su odabrane jer su pokazale visoku učinkovitost u smanjenju dimenzionalnosti podataka, a time i povećanju učinkovitosti modela strojnog učenja (Pavlidis i ostali (2016)).

Primjena strojnog učenja je novo, ali brzorastućo područje investicijske analize. Lopez de Prado (Lopez de Prado 2018) naglašava ključnu ulogu strojnog učenja u modernim financijama. U svojoj široko poznatoj knjizi, pruža broje savjete za praktičnu primjenu financijskog strojnog učenja, od pripreme i filtriranja podataka, opzimizacije modela do backtestinga. A seminal work by Kelly i Xiu (2023) provides an exhaustive review of the integration of machine learning within financial studies. Their work encapsulates the breadth of machine learning's applicability, ranging from asset pricing and portfolio management to risk assessment and regulatory compliance. This review article serves as a cornerstone for scholars and practitioners alike, offering insights into both the theoretical underpinnings and practical implementations of machine learning techniques in finance. Gu, Kelly, i Xiu (2020) u svom radu "Empirical Asset Pricing via Machine Learning" razmatraju primjenu metoda strojnog učenja na empirijsko određivanje cijena imovine, gdje istražuju kako tehnike strojnog učenja mogu identificirati i iskoristiti složene obrasce u podacima za predviđanje prinosa. Njihova analiza pokazuje da strojno učenje može značajno unaprijediti preciznost modela cijena imovine, doprinoseći boljem razumijevanju dinamike tržišta i otkrivanju novih faktora rizika i prinosa. Aloud, Tsang i Olsen (2021) istraživali su upotrebu genetskih algoritama za optimizaciju strategija zaštite od rizika u upravljanju portfeljem, demonstrirajući kako strojno učenje može poboljšati tradicionalne financijske prakse kroz adaptivne i dinamične modele. S druge strane, Feng, Giglio i Xiu (2020) bavili su se izazovom 'zoo faktora' u određivanju cijena imovine, primjenjujući strojno učenje za evaluaciju i selekciju ekonomski značajnih faktora, što dovodi do razvoja učinkovitijih modela za predviđanje prinosa. Nuij i suradnici (2019) koristili su obradu prirodnog jezika za analizu novinskih članaka s ciljem procjene tržišnog sentimenta, što je primjer kako dubinska analiza nestrukturiranih podataka može unaprijediti predikcije tržišnih kretanja. Rad Hestona i Sinhe (Heston i Sinha Year) istražuje prediktivnu moć sentimenta vijesti o povratima dionica, otkrivajući vremensku dinamiku utjecaja vijesti, što se izravno usklađuje s fokusom na razumijevanje utjecaja vanjskih informacija na financijska tržišta. Osim toga, nalazi Shaikha i sur. (Shaikh i Others Year)

otkrivaju potencijal korištenja strojnog učenja za predviđanje kretanja tržišta dionica, naglašavajući rastući interes za korištenje naprednih računalnih tehnika za financijsko prognoziranje. Dodatno, rad Zhao, Bose, i Maher (2021) istražuju specifičnu primjenu s trojnog učenja u predviđanju trendova cijena dionica. Kroz detaljnu analizu povijesnih podataka o cijenama dionica i primjenu različitih algoritama strojnog učenja, autori demonstriraju sposobnost ovih tehnika da efikasno predviđaju kretanja na tržištu, naglašavajući važnost modeliranja vremenskih serija i analize sentimenta za povećanje točnosti predviđanja. Ovi radovi zajedno naglašavaju ključnu ulogu strojnog učenja u predviđanju viška prinosa, koristeći napredne tehnike za bolje razumijevanje i iskorištavanje složenih obrazaca na financijskim tržištima.

Rezultati istraživanja pokazuju da modeli strojnog učenja imaju visoku učinkovitost u predviđanju očekivanih prinosa dionica na Zagrebačkoj burzi. Pri tome postoji velika razlika u uspješnosti modela. Modeli stabla odlučivanja i slučajnih šuma pokazuju veću učinkovitost od penalizirajučih regresijskih modela i neuronskih mreža. Ansambli modela (medijan i prosjek predikcija) pokazuju bolje rezultate od pojedinačnih modela. Osim toga, rezultati pokazuju da su novi prediktori, koji su konstruirani na temelju dnevnih podataka o trgovanju, ključni za postizanje visoke učinkovitosti modela strojnog učenja.

Rezultati ovog istraživanja mogu biti korisni za investitore, ali i za regulatorna tijela koja nadziru tržište kapitala. Investitori bi mogli koristiti rezultate ovog istraživanja za donošenje investicijskih odluka, dok bi regulatorna tijela mogla koristiti rezultate za nadzor tržišta kapitala. Osim toga, rezultati ovog istraživanja mogli bi biti korisni i za istraživače koji se bave primjenom modela strojnog učenja u financijama. Ovo istraživanje može poslužiti kao polazna točka za daljnja istraživanja koja se bave primjenom modela strojnog učenja na malim, plitkim tržištima kapitala.

2 Podaci

2.1 Opis podataka i prediktora

U analizi se koriste trgovinski podaci za sve dionice koje su kotirale na Zagrebačkoj burzi od njezinog osnutka. Podaci su preuzeti sa službenih web stranica Zagrebačke burze. Budući da je u početku tržište kapitala bilo izrazito plitko, te je na burzi kotiralo svega nekoliko kompanija, početno razdoblje uzorka je pomaknuto na 2000. godinu. Podaci dakle uključuju trgovinske podatke od 03.01.2000. godine do 29.03.2024..

Na podacima su poduzeti uobičajeni postupci čiščenja podataka:

- eliminirane su sve duple vrijednosti. Drugim riječima, simbol (ISIN) i datum moraju biti unikatni kroz cijeli uzorak.
- eliminirane su sve opservacije gdje je cijena potencijalno manja od 1e-008.

Table 1. *Prikaz prvih pet redova OHLCV podataka*

```
sample_data = prices[, .SD, .SDcols = -c("week", "last_week_day")]
setcolorder(sample_data,
            c("isin", "date", "open", "high", "low", "close", "volume"))
sample_data[, isin := gsub("[0-9]+", "", isin)]
sample_data = head(sample_data)
ft = qflextable(sample_data) |>
    colformat_double()
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

- u uzorku su ostavljeni samo simboli koji imaju najmanje 2 godine podataka. Ovaj uvjet je nužan jer se kod računanja prediktora koriste dugački prozori.
- nedostajuće vrijednosti za zadnje cijene su zamjenjene prosječnim vrijednostima za isti bar.
- budući da se pojedinim simbolima malo trguje, iz uzorka su izbačeni simboli za koje je udio broja trgovinskih dana u ukupnom broju mogućih trgovinskih dana najmanje 70%.

Nakon svih potonjih prilagodbi, ukupni uzorak sadrži ~1.300.000 opservacija. U tablici Table 1 je prikazano prvih pet redova OHLCV podataka.

Na temelju dnevnih OHLCV podataka, konstruirani su prediktori koji se koriste u treniranju modela strojnog učenja. Prediktori su konstruirani na temelju dnevnih podataka, a zatim agregirani na tjednu razinu. U modeliranju se na kraju koristi tjedna frekvencija s tjednim horizontom. Budući da se radi o velikom broju prediktora, u tablici su prikazani opisi grupe prediktora, kao i R i python paketi pomoću kojih su konstruirani. Prvu grupu čine testovi eksplozivnosti univarijantnih vremenskih nizova. Više o ovoj grupi možete pronaći u Pavlidis et al. (Pavlidis i ostali 2016), Phillips et al. (Phillips, Shi, i Yu 2015) i Vasilopoulos et al. (Vasilopoulos, Pavlidis, i Martínez-García 2022). Druga grupa uključuje kreiranje predviđanja na temelju tri univarijantna modela: ARIMA, ETS i NNETAR. Više o ovim modelima moežete pronaći u Hyndman et al. (R. J. Hyndman i Khandakar 2008). Sljedeća grupa prediktora uključuje testove strukturnog loma. Više o ovim testovima možete pronaći u (Otto i Breitung 2023). Preostale grupe uključuju prediktore izračunate pomoću programskih paketa koji su razvijeni za generiranje prediktora za analizu vremenskih serija. To uključuje sljedeće grupe: kanoničke karakteristike vremenskih serija ((Lubba i ostali 2019)), korištenje paketa za automatsku ekstrakciju prediktora ((O'Hara-Wild i ostali 2023), (R. Hyndman i ostali 2023), (Barandas i ostali 2020)). Sljedeća grupa sadrži predviđanja generirana iz Wavelet transformacija ((Paul, Samanta, i Yeasin 2022), (AMINGHAFARI i POGGI 2007)). Na kraju, autor je napravio svoju funkciju unutar finfeatures R paketa,

Table 2. *Opis prediktora*

```
prediktori = data.frame(
  `Grupa prediktora` = c("Testovi eksplozivnosti",
                         "Univarijantni predikcijski modeli",
                         "Testovi strutkurnog loma",
                         "Kanoničke karakteristike vremenskih serija",
                         "Automatska ekstrakcija prediktora",
                         "Wavelet ARIMA",
                         "Tehnički indikatori i statistike tržišta"),
  Paket = c("exuber (R)",
            "forecasts (R)",
            "backCUSUM (R)",
            "catch22 (R)",
            "feasts (R), tsfeatures (R), tsfel (py)",
            "WaveletArima (R)",
            "finfeatures (R)"),
    Opis = c("Namijenjen je ekonometrijskoj analizi eksplozivnih vremenskih nizova, posebice za testiran
           "Pruža metode i alate za prikazivanje i analizu prognoza jednovarijatnih vremenskih nizova, u
           "Nudi funkcionalnosti za testiranje i praćenje strukturnih promjena u podacima.",
           "Pruža 22 kanoničke karakteristike vremenskih serija dizajnirane za brzu i efikasnu analizu v
           "Ekstrakcija značajki, dekompozicije, statističke sažetke i vizualizacije.",
           "Integrira valne transformacije s ARIMA modelom kako bi poboljšao točnost prognoziranja vreme
           "Prediktori su organizirani u grupama kao što su povrati i volatilnost, tehnički indikatori (
)
ft = qflextable(prediktori)
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

koja računa brojne prediktore uključujući tehničke indikatore, statistike prinosa na pomičnim prozorima i volatilnosti (Sag 2023). Kod za izračun prediktora dostupan je na GitHub stranici[1]

Nakon generiranja svih prediktora, ponovno su poduzete mjere čišćenja podataka. Uzorak je ponovno filtriran tako da su eliminirane sve duple vrijednosti po ID varijablama (simbol i datum), eliminirane su kolone koje imaju više od 50% nedostajućih vrijednosti, maknute su kolone koje imaju više od 2% Inf vrijednosti, te potom i opservacije koje imaju Inf vrijednosti. Također su eliminirane konstantne kolone.

```
# Load registry
reg = loadRegistry(PATH, work.dir=PATH)


# Done ids
```

[1]Poveznica na kod: https://github.com/MislavSag/finfeatures

```
ids = findDone(reg=reg)
ids_notdone = findNotDone(reg=reg)

# Get metadata for done jobs
tabs = getJobTable(ids, reg = reg)
tabs = tabs[, .SD, .SDcols = c("job.id", "job.name", "repl", "prob.pars", "algo.pars")]
predictions_meta = cbind.data.frame(
  id = tabs[, job.id],
  task = vapply(tabs$prob.pars, `[[`, character(1L), "task_id"),
  learner = gsub(".*regr.|.tuned", "", vapply(tabs$algo.pars, `[[`, character(1L), "learner_id")),
  cv = gsub("custom_|_.*", "", vapply(tabs$prob.pars, `[[`, character(1L), "resampling_id")),
  fold = gsub("custom_\\d+_", "", vapply(tabs$prob.pars, `[[`, character(1L), "resampling_id"))
)

predictions_l = lapply(ids[[1]], function(id_) {
  # id_ = 1
  x = tryCatch({readRDS(fs::path(PATH, "results", id_, ext = "rds"))},
               error = function(e) NULL)
  if (is.null(x)) {
    print(id_)
    return(NULL)
  }
  x = x$prediction
  x["id"] = id_
  x
})

predictions = list()
for (i in seq_along(predictions_l)) {
  # i = 4185
  # print(i)
  x = predictions_l[[i]]
  if (length(x$test$row_ids) == 0) {
    print(i)
    next
  }
  predictions[[i]] = cbind.data.frame(
    id = x$id,
```

```
    row_ids = x$test$row_ids,
    truth = x$test$truth,
    response = x$test$response
  )
}
# predictions = lapply(predictions_l, function(x) {
#   # x = predictions_l[[1]]
#   cbind.data.frame(
#     id = x$id,
#     row_ids = x$test$row_ids,
#     truth = x$test$truth,
#     response = x$test$response
#   )
# })
predictions = rbindlist(predictions)
predictions = merge(predictions_meta, predictions, by = "id")
predictions = as.data.table(predictions)

# import tasks
tasks_files = dir_ls(fs::path(PATH, "problems"))
tasks = lapply(tasks_files, readRDS)
names(tasks) = lapply(tasks, function(t) t$data$id)

# add backend to predictions
backend_l = lapply(tasks, function(tsk_) {
  # tsk_ = tasks[[1]]
  # x = tsk_$data$backend$data(1:tsk_$data$nrow, c("symbol", "week", "..row_id", TARGETS))
  x = tsk_$data$backend$data(1:tsk_$data$nrow, c("symbol", "date", "..row_id", TARGETS))
  setnames(x, "..row_id", "row_ids")
  x
})
backends = rbindlist(backend_l, fill = TRUE)

# merge predictions and backends
predictions = backends[predictions, on = c("row_ids")]

# change month to date from Posixct
# predictions[, week := as.Date(week)]
```

```
predictions[, date := as.Date(date)]


# clean predictions
# preds = unique(predictions, by = c("row_ids", "week", "task", "learner", "cv"))
preds = unique(predictions, by = c("row_ids", "date", "task", "learner", "cv"))
preds = na.omit(preds)
preds[, week := ceiling_date(date, unit = "week") - 1]

tsk_ = tasks[[1]]
n_after = tsk_$data$nrow
n_predictors = length(tsk_$data$feature_names)
```

Nakon dodavanja i čiščenja prediktora i povećanja frekvencije s dnevne na tjednu frekvenciju, konačan uzorak sadrži ~150.000 opservacija i preko 500 prediktora.

2.2 Sažeta statistika podataka i prediktora

Svako investiranje na financijskom tržištu započinje konstruiranjem univerzuma. Univerzum se bira iz liste kotiranih dionica, a odabir se vrši na temelju različitih kriterija. Slika Figure 1 prikazuje broj dionica koje su kotirale na Zagrebačkoj burzi od 2000. godine. Vidljiv je jasan trend rasta broj dionica do Velike Recesije 2008 i jasan trend pada nakon toga. Nakon 2019. godine broj dionica se stabilizirao na oko 30 dionica. Valja podsjetiti da su iz ukupne populacije dionica u procesu čišćenja podataka neke dionice izbačene iz uzorka jer nisu zadovoljavale kriterije kvalitete podataka.

```
# number of companies through time
n_firms = prices[, .N, by = date]
setorder(n_firms, date)
n_firms[, N_SMA := TTR::SMA(N, 22)]
n_firms = na.omit(n_firms)
ggplot(n_firms, aes(x = date, y = N_SMA)) +
  geom_line() + geom_point() +
  theme_bw() +
  gg_theme +
  labs(y = "Broj kompanija")
```

Slika se generira iz izvornih podataka (`F:/zse/prices.csv`) *pokretanjem gornjeg koda s* `eval: true`. *U ovoj inačici dokumenta podaci nisu uključeni pa je prikazan samo kôd.*

Figure 1. *Broj dionica na ZSE*

Ključna zavisna vraijble u analizi je tjedni povrat, koji je izračunat iz dnevnih povrata. Tablica

Table 3. *Statističke mjere tjednih povrata*

```
# summary statistics for stocks returns
prices_month = prices[last_week_day == TRUE, .(isin, date, close)]
prices_month[, returns := close / shift(close, 1) - 1]
summary_by_symbol = prices_month[, .(

  mean = mean(returns, na.rm = TRUE),
  median = median(returns, na.rm = TRUE),
  sd = sd(returns, na.rm = TRUE),
  skew = skewness(returns, na.rm = TRUE),
  kurt = kurtosis(returns, na.rm = TRUE),
  min = min(returns, na.rm = TRUE),
  max = max(returns, na.rm = TRUE)
), by = isin]
summary_returns = summary_by_symbol[, lapply(.SD, mean),
                                    .SDcols = c("mean", "median", "sd", "skew",
                                                "kurt", "min", "max")]
ft = qflextable(summary_returns) |>
    colformat_double()
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

Table 3 prikazuje statistike tjednih povrata za cijeli uzorak.

3 Metodologija

3.1 Modeli strojnog učenja

3.1.1 Random Forest

Random Forest (RF) je napredni algoritam strojnog učenja koji koristi tehniku ansambla za poboljšanje točnosti i otpornosti na prekomjerno učenje (Breiman 2001). Osnova RF algoritma leži u kombiniranju velikog broja stabala odlučivanja, gdje svako stablo koristi slučajno izabran skup podataka dobiven metodom bagginga. Bagging podrazumijeva stvaranje jedinstvenih podskupova originalnog skupa podataka putem nasumičnog izabiranja opservacija, čime se osigurava da svako stablo u šumi dobiva malo drugačiji skup podataka za učenje. Osim toga, RF unosi slučajnost i u odabir značajki (kolona) za svako stablo, dodatno smanjujući korelaciju među stablima i pomažući u stabilizaciji predikcijske pogreške (Liaw i Wiener 2002). Jedna od ključnih prednosti RF-a u odnosu na pojedinačna stabla odlučivanja jest njezina sposobnost da smanji varijancu bez značajnog povećanja pristranosti, čime se postiže visoka točnost predviđanja (Cu-

tler, Cutler, i Stevens 2012).

Kod predviđanja, RF koristi prosječnu vrijednost predikcija svih stabala u šumi:

$$\hat{y} = \frac{1}{B} \sum_{b=1}^{B} f_b(x)$$

gdje B označava broj stabala u šumi, $f_b(x)$ je predikcija pojedinog stabla za ulaz (x), a $\hat{y}$ je konačna predikcija algoritma RF. Ova metoda omogućava da se individualne greške stabala kompenziraju, dovodeći do preciznijeg i pouzdanijeg rezultata.

Formulom varijance RF-a može se objasniti kako se smanjuje ukupna varijanca modela s povećanjem broja stabala:

$$\text{Variance(RF)} = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

Ovdje, ρ predstavlja prosječnu korelaciju između predikcija stabala, a σ^2 varijancu pojedinačnog stabla. S povećanjem (B), doprinos nekorelirane varijance se smanjuje, čime se ukupna varijanca RF-a efektivno smanjuje, dok istovremeno održava pristranost na prihvatljivoj razini (Biau i Scornet 2016).

3.1.2 XGBOOST

XGBoost (Extreme Gradient Boosting) predstavlja napredak u području strojnog učenja, posebno u ansambliranju stabala odlučivanja i njihovom sekvencijalnom (aditivnom) treniranju. Algoritam efikasno kombinira prednosti stabala odlučivanja s gradientnim pojačanjem (Chen i Guestrin 2016).

Osnovna ideja XGBoost-a leži u optimizaciji složene funkcije cilja koja ne samo da mjeri koliko je model dobar u predviđanjima, već uzima u obzir i kompleksnost modela kako bi se spriječilo prekomjerno učenje. To se postiže kroz pažljivo balansiranje između točnosti predviđanja modela i njegove generalizacijske sposobnosti.

XGBoost implementira ovo kroz dodavanje novih stabala jedno po jedno, gdje svako novo stablo korigira greške napravljene od strane prethodno dodanih stabala. Umjesto da se sva stabla treniraju odjednom, pristup sekvencijalnog dodavanja omogućava XGBoostu da preciznije prilagodi model na temelju prethodnih grešaka. Ovo sekvencijalno treniranje, gdje se svako novo stablo fokusira na prethodne greške, ključno je za smanjenje grešaka i poboljšanje točnosti modela.

Centralna formula koja se koristi za predviđanje u XGBoost modelu je:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)$$

gdje K predstavlja ukupan broj stabala, a $f_k(x_i)$ funkciju k-tog stabla. Ova formula omogućava da se predviđanja izračunaju kao suma predviđanja svih pojedinačnih stabala, čime se postiže veća točnost.

Jedna od ključnih prednosti XGBoost-a je njegova sposobnost da se prilagodi različitim funkcijama gubitka, što ga čini fleksibilnim za širok spektar problema u strojnom učenju. Osim toga, XGBoost uključuje napredne tehnike regularizacije, poput L1 i L2 penalizacije, koje pomažu u kontroli kompleksnosti modela i sprječavaju prekomjerno učenje, čime se dodatno poboljšava performansa modela (Friedman 2001, 2002).

3.1.3 GLMNET

Generalizirani linearni modeli (GLM) su fleksibilna klasa modela koja se koristi za modeliranje različitih tipova zavisnih varijabli, uključujući kontinuirane, binarne i višeklasne varijable. GLM-ovi su posebno korisni u situacijama kada je potrebno modelirati zavisnost između više prediktora i zavisne varijable, te kada je potrebno kontrolirati utjecaj svakog prediktora na zavisnu varijablu.

Postoji više podvrsta GLM modela, a jedna od najpopularnijih je LASSO (Least Absolute Shrinkage and Selection Operator) i Elastic Net modeli. Ovi modeli koriste regularizaciju kako bi se smanjila varijanca modela i spriječilo prekomjerno učenje. Regularizacija se postiže dodavanjem L1 (LASSO) ili L2 (Ridge) penala na funkciju gubitka, čime se smanjuje utjecaj manje važnih prediktora i poboljšava generalizacijska sposobnost modela.

Ciljna funkcija za Lasso model je:

$$\min_{\beta} \left\{ \frac{1}{2N} \sum_{i=1}^{N} (y_i - x_i^T \beta)^2 + \lambda \|\beta\|_1 \right\}$$

gdje je N broj opservacija, y_i zavisna varijabla, x_i prediktor, β koeficijenti, λ parametar regularizacije, a $\|\beta\|_1$ L1 norma koeficijenata predstavlja regularizacijski član koji pomaže u smanjenju varijance modela i sprječava prekomjerno učenje.

Elastic Net model kombinira L1 i L2 penale kako bi se iskoristile prednosti oba pristupa. Ciljna funkcija za Elastic Net model je:

$$\min_{\beta} \left\{ \frac{1}{2N} \sum_{i=1}^{N} (y_i - x_i^T \beta)^2 + \lambda_1 \|\beta\|_1 + \frac{\lambda_2}{2} \|\beta\|_2^2 \right\}$$

gdje su λ_1 i λ_2 parametri regularizacije za L1 i L2 penale, respektivno. Ova funkcija omogućava da se iskoriste prednosti oba pristupa, čime se postiže veća točnost i generalizacijska sposobnost modela.

3.1.4 NNET

Feed-forward neuronske mreže (FFNs) predstavljaju osnovnu arhitekturu u području neuronskih mreža i strojnog učenja. Karakterizirane svojom sekvencijalnom strukturom slojeva, gdje informacije teku u jednom smjeru od ulaza do izlaza, FFNs su svestrane u rješavanju širokog spektra problema, od zadataka regresije do klasifikacije. Odsutnost ciklusa unutar ovih mreža razlikuje ih od rekurentnih neuronskih mreža, čineći FFNs jednostavnijim, ali moćnim alatima za modeliranje linearnih i nelinearnih odnosa. Njihova arhitektura tipično uključuje tri vrste slojeva: ulazni sloj koji prima podatke, jedan ili više skrivenih slojeva koji izračunavaju transformacije i izlazni sloj koji daje konačnu predikciju.

Operacija unutar feed-forward mreže, posebno s jednim skrivenim slojem za regresiju, matematički se može predstaviti sljedećim formulama:

a) izlaz h_j svakog neurona j u skrivenom sloju izračunava se kao:

$$h_j = \sigma\left(\sum_{i=1}^{n} w_{ij}x_i + b_j\right)$$

b) Konačni izlaz y iz mreže, za zadatke regresije s jednim izlaznim neuronima, je:

$$y = \sum_{j=1}^{m} w_j h_j + b$$

gdje x_i označava ulazne značajke, w_{ij} su težine koje povezuju ulaznu značajku i sa skrivenim neuronima j, b_j je pristranost za skriveni neuron j, σ predstavlja funkciju aktivacije (npr. sigmoidna ili ReLU), w_j su težine od skrivenog neurona j do izlaznog neurona, b je pristranost za izlazni neuron, a m je broj neurona u skrivenom sloju.

Jednadžbe sažimaju bit feed-forward obrade, ilustrirajući kako se ulazi transformiraju u izlaze kroz seriju linearnih kombinacija nakon kojih slijedi nelinearna aktivacija. Izbor funkcija aktivacije, broj neurona u skrivenom sloju i metoda treninga (kao što je povratna propagacija za prilagodbu težina) ključni su za sposobnost mreže da modelira složene funkcije i postigne visoke performanse na zadacima regresije.

Table 4. *Duljine prozora ugniježđene metode križne validacije s pomičnim prozorom*

```
cv_params = data.frame(
  Trening = c(4*48, 4*72),
  Razmak = c(1, 1),
  Validacija = c(3*4, 6*4),
  Razmak = c(1, 1),
  Test = c(1, 1)
)
ft = qflextable(cv_params)
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

3.2 Ugniježđena metoda križne validacije s pomičnim prozorom

Ugniježđena metoda križne validacije s pomičnim prozorom (NRWCV) sofisticirana je metodologija dizajnirana za evaluaciju prediktivnih modela s vremenskim serijama. Ova tehnika posebno je korisna u scenarijima koji uključuju sekvencijalno prikupljanje podataka, poput podataka o cijenama dionica. Metodologija se sastoji od vanjske petlje i unutarnje petlje, pri čemu svaka koristi pristup pomičnih prozora. Vanjska petlja dijeli skup podataka na više skupova za trening i testiranje na pomični način, trenirajući model na trenutnom skupu za trening, a zatim ga testirajući na sljedećem skupu za testiranje. To simulira stvarni scenarij gdje se model primjenjuje na neviđene buduće podatke. Ugnežđena unutar vanjske petlje, unutarnja petlja usmjerena je na odabir modela i podešavanje hiperparametara dodatnim dijeljenjem skupa za trening na skupove za trening i validaciju, koristeći pristup pomičnim prozorima. To omogućava procjenu i odabir najučinkovitijih konfiguracija modela. Uvode se i razmaci između skupa za trening i skupa za validaciju, kao i između skupa za validaciju i skupa za testiranje, kako bi se spriječilo prelijevanje podataka iz budućnosti u sadašnjost. Na taj način se onemogućuje prelijevanje horizonta ciljane varijable u skup za validaciju ili test skup.

Kako bi olakšali razumijevanje ugniježđene metode križne validacije s pomičnim prozorom korištene u ovom radu, na slici Figure 2 prikazujemo uzorak skupa za treniranje, validaciju i testiranje. Slika sadrži dva grafikona jer se u radu koriste 2 različita skupa parametra za duljine skupova kako je prikazano u tablici Table 4. Trening skup u prvom NRWCV sadrži 4 godine podataka, tjedan razmaka između skupa za treniranje i validaciju, 3 mjeseca za validaciju, tjedan razmaka između validacije i testiranja, te jedan tjedan za testiranje. U drugom NRWCV skupu, duljine su povećane na 6 godina za trening i 6 mjeseci za validaciju, dok su ostali parametri identični. Lijevi graf na slici Figure 2 prikazuje skupove za treniranje, validaciju i testiranje za prvi NRWCV, dok desni graf prikazuje skupove za drugi NRWCV. Brojevi uz CV (npr. CV-1194)

označuje ukupan broj komponenti.

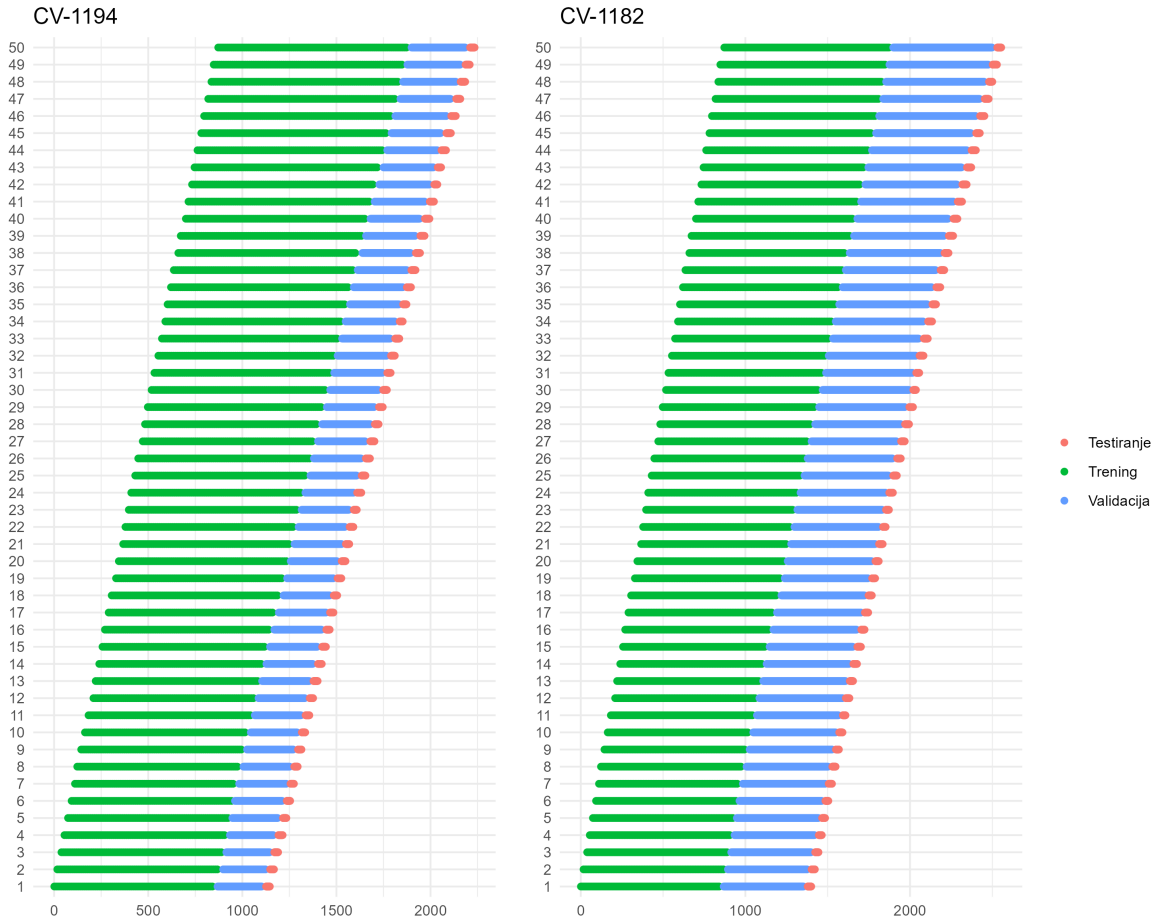

Figure 2. *Skupovi treniranja, validacije i testiranja unutar ugniježđene metode križne validacije s pomičnim prozorom*

3.3 Uglađivanje parametra strojnog učenja

U prethodnom potpoglavlju je opisana metoda ugnježđene metode križne validacije s pomičnim prozorom koja se koristi za evaluaciju prediktivnih modela s vremenskim serijama. U ovom potpoglavlju opisat ćemo kako se koristi metoda ugnježđene metode križne validacije s pomičnim prozorom za ugađanje parametara modela strojnog učenja. U ovom radu koristit ćemo 4 različita modela strojnog učenja: Random Forest, XGBoost, GLMNET i NNET. Svaki model ima svoje hiperparametre koji se moraju ugađati kako bi se postigla najbolja performansa pojedinog modela. U tablici Table 5 prikazani su parametri koji će se koristiti za svaki model.

Osim uglađivanja parametara pojedinih modela, u ovom radu koristit ćemo i različite tehnike predobrade podataka kako bismo poboljšali kvalitetu i performanse modela. U tablici Table 5 (redci označeni s *Predobrada*) prikazane su i tehnike predobrade podataka koje će se koristiti prije svake procjene parametara modela. Prije svakog modela, koristit će se sljedeće preprocesne radnje: 1) Brisanje kolona koje imaju više od 5% nedostajućih vrijednosti. 2) Brisanje opservaci-

Table 5. *Parametri za ugađanje modela*

```
tuning_parameters = data.table(
  Model = c(rep("Predobrada", 4), rep("Random Forest", 4), rep("XGBoost", 5), rep("Neuronska Mreža", 3),
  Parametar = c("dropcorr.cutoff", "winsorizesimple.probs_high", "winsorizesimple.probs_low", "filter_br
                "max.depth", "replace", "mtry.ratio", "num.trees",
                "alpha", "max_depth", "eta", "nrounds", "subsample",
                "size", "decay", "maxit",
                "s", "alpha"),
  Opis = c("Prag za odbacivanje visoko koreliranih značajki", "Gornji prag vjerojatnosti za winsorizacij
           "Maksimalna dubina stabala", "Uzorkovanje podataka s zamjenom", "Omjer varijabli dostupnih za
           "L1 regularizacijski termin na težinama", "Maksimalna dubina stabala", "Korak smanjenja velič
           "Broj jedinica u skrivenim slojevima", "Parametar smanjenja težine", "Maksimalan broj iteraci
           "Elasticnet miješajući parametar", "L1 regularizacijski put na težinama"),
  ProstorPretraživanja = c("0.80, 0.90, 0.95, 0.99", "0.999, 0.99, 0.98, 0.97, 0.90, 0.8", "0.001, 0.01,
                           "1-15", "TRUE/FALSE", "0.1-1", "10-2000",
                           "0.001-100 (logaritamska skala)", "1-20", "0.0001-1 (logaritamska skala)", "1-
                           "2-15", "0.0001-0.1", "50-500",
                           "5-30", "1e-4-1 (logaritamska skala)")
)

# Flextable
ft = qflextable(tuning_parameters)
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

ja koje imaju nedostajuće vrijednosti. 3) Brisanje konstantnih kolona. 4) *Winsorizacija* podataka. Winsorizacija ograničava ekstremne vrijednosti na određene percentilne vrijednosti, čime se smanjuje njihov utjecaj na model. Ovaj korak je dio prilagođavanja podataka kako je opsiano u tablici Table 5. 5) Odbacivanje visoko koreliranih značajki, koja smanjuje redundanciju u podacima postavljanjem praga korelacije iznad kojeg značajke smatramo previše sličnima i stoga jednu od njih odbacujemo. Ovaj korak je također dio prilagođavanja podataka kako je opsiano u tablici Table 5. 6) Uniformiyacija podataka. 7) Filtriranje prediktora pomoću dva filtera JMI i Relief, koje nam omogućavaju da se fokusiram samo na one prediktore koje pružaju najviše informacija za predviđanje.

Nakon predobrade, usredotočujemo se na optimizaciju hiperparametara modela. Za modele poput Random Foresta, XGBoost, neuronskih mreža i GLMNeta, pažljivo biramo raspon hiperparametara za istraživanje. Za Random Forest, na primjer, podešavamo maksimalnu dubinu stabala, omjer varijabli dostupnih za razdvajanje na svakom čvoru, broj stabala, i da li uzorkovanje treba biti s zamjenom ili bez. U slučaju XGBoost, optimiziramo parametre poput stope učenja, maksimalne dubine, broja iteracija pojačavanja, i omjera uzorkovanja. Za neuronske mreže, fokusiramo se na broj jedinica u skrivenim slojevima, parametar smanjenja težine i maksimalan broj iteracija. Za GLMNet, optimizacija uključuje podešavanje ElasticNet miješajućeg parametra i L1 regularizacije.

Optimizacija hiperparametara provodi se kroz postupak ugniježđene unakrsne validacije, gdje unutarnji sloj služi za podešavanje hiperparametara, dok vanjski sloj procjenjuje performanse modela s optimalnim hiperparametrima. Ovaj pristup osigurava da naša evaluacija modela bude nepristrana i da dobro generalizira na neviđene podatke.

Kroz ovaj temeljit proces, cilj je razviti modele koji nisu samo prilagođeni trenutnim podacima, već su i sposobni generalizirati i performirati dobro na širokom rasponu scenarija, čime se osigurava njihova praktična primjenjivost i robustnost.

4 Rezultati

4.1 Uspješnost modela

U nastavku prikazujemo uspješnost pojedinih modela u predviđanju tjednih povrata dionica koje su korirale na ZSE od 2000. Uspješnost će se mjeriti pomoću različitih mjera koje se koriste za analiziranje uspješnosti klasifikacijskih i regresijdkih modela. Također će se prikazati i jednostavni backtest, koji pokazuju rezultate trgovinske strategije, koja koristi predviđanja modela za konstruiranje portfolia dugih pozicija s tjednim rebalansiranjem.

Za početak analiziramo uspješnost modela s obzirom na različite mjere.Sve korištene mjere su objašnenje u tablici Table 6. Važno je napomenuti da je model strojnog učenja koji je treniran

Table 6. *Mjere uspješnosti modela*

```
# Define the measures and their interpretations
measures <- c("Točnost (acc)",
              "F-beta Ocjena (fbeta)",
              "Stopa Pravih Pozitivnih (TPR) / Odziv",
              "Preciznost",
              "Stopa Pravih Negativnih (TNR)",
              "Vrijednost Negativne Predikcije (NPV)",
              "Srednja Kvadratna Greška (MSE)",
              "Srednja Apsolutna Greška (MAE)")

interpretations <- c(
  "Ukazuje na opću točnost modela. Može biti varljiva u nebalansiranim skupovima podataka.",
  "Balansira preciznost i odziv. □ > 1 daje veću važnost odzivu, dok □ < 1 daje veću važnost preciznosti
  "Proporcija stvarno pozitivnih ispravno identificiranih. Ne uzima u obzir lažne pozitivne.",
  "Proporcija ispravno identificiranih pozitivnih. Važno kada su lažni pozitivni skupi.",
  "Proporcija stvarno negativnih ispravno identificiranih. Važno za ispravno predviđanje negativnih.",
  "Proporcija ispravno identificiranih negativnih rezultata. Korisno u medicinskom testiranju.",
  "Mjeri kvalitetu predikatora. Niže vrijednosti ukazuju na bolje pristajanje. Osjetljivo na izvanredne
  "Mjeri prosječnu veličinu grešaka u skupu predikcija, bez razmatranja njihovog smjera. Manje osjetljiv

# Combine into a data.frame
performance_measures_df = data.frame(Mjera = measures, Interpretacija = interpretations)

# Flextable
ft = qflextable(performance_measures_df)
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

regresijski model. Drugim riječima, zavisna varijabla je kontinuirana, a ne diskretna. Međutim, kako bi dobili bolji uvid u uspješnost modela, koristit ćemo i mjere koje se koriste za klasifikacijske modele. Pri tome su klase konstruirane na način da se dionice koje imaju pozitivne procjenjene tjedne povrate smatraju klasom 1, dok se dionice koje imaju negativne tjedne povrate smatraju klasom 0.

Osim pojedinačnih modela strojnog učenja, koji su korišteni u analizi, u nastavku ću procjenjivati i uspješnost ansambla modela. Ansambl modeli su modeli koji kombiniraju više modela kako bi se poboljšala uspješnost predviđanja. U ovom radu korišteni su tri jednostavna ansambla: srednja vrijednost, medijan i suma. Srednja vrijednost ansamblira predviđanja svih modela tako da se uzima prosječna vrijednost. Medijan ansamblira predviđanja tako da se uzima medijan. Suma ansamblira predviđanja tako da se uzima suma svih predviđanja.

```
# prediction to wide format
# predsw = copy(preds)
# predsw[, week := data.table::week(date)]
predsw = dcast(
  preds,
  task + cv + fold + week + date + symbol + truth  ~ learner,
  value.var = "response"
)

# ensambles
cols = colnames(predsw)
cols = cols[(which(cols == "truth")+1):ncol(predsw)]
p = predsw[, ..cols]
pm = as.matrix(p)
predsw = cbind(predsw, mean_resp = rowMeans(p, na.rm = TRUE))
predsw = cbind(predsw, median_resp = rowMedians(pm, na.rm = TRUE))
predsw = cbind(predsw, sum_resp = rowSums2(pm, na.rm = TRUE))
predsw = cbind(predsw, iqrs_resp = rowIQRs(pm, na.rm = TRUE))
predsw = cbind(predsw, sd_resp = rowMads(pm, na.rm = TRUE))
predsw = cbind(predsw, q9_resp = rowQuantiles(pm, probs = 0.9, na.rm = TRUE))
predsw = cbind(predsw, max_resp = rowMaxs(pm, na.rm = TRUE))
predsw = cbind(predsw, min_resp = rowMins(pm, na.rm = TRUE))
predsw = cbind(predsw, all_buy = rowAlls(pm >= 0, na.rm = TRUE))
predsw = cbind(predsw, all_sell = rowAlls(pm < 0, na.rm = TRUE))
predsw = cbind(predsw, sum_buy = rowSums2(pm >= 0, na.rm = TRUE))
predsw = cbind(predsw, sum_sell = rowSums2(pm < 0, na.rm = TRUE))

# Calculate measures help function
```

Table 7. *Uspješnost modela*

```
by_ = c("cv", "variable")
dt_ = na.omit(preds_perf)[, calculate_msrs(truth, value), by = by_]
# fwrite(dt_, "data/preds_perf.csv")
ft = qflextable(dt_) |>
    colformat_double()
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

```
sign01 = function(x) as.factor(ifelse(x > 0, 1, 0))
calculate_msrs = function(t, res) {
  t_sign   = sign01(t)
  res_sign = sign01(res)
  list(acc       = mlr3measures::acc(t_sign, res_sign),
       fbeta     = mlr3measures::fbeta(t_sign, res_sign, positive = "1"),
       tpr       = mlr3measures::tpr(t_sign, res_sign, positive = "1"),
       precision = mlr3measures::precision(t_sign, res_sign, positive = "1"),
       tnr       = mlr3measures::tnr(t_sign, res_sign, positive = "1"),
       npv       = mlr3measures::npv(t_sign, res_sign, positive = "1"),
       mse       = mlr3measures::mse(t, res),
       mae       = mlr3measures::mae(t, res)
       )
}
```

```
cols = colnames(predsw)
cols = cols[(which(cols == "truth")+1):which(cols == "median_resp")]
preds_perf = melt(predsw,
                  id.vars = c("task", "cv", "week", "date", "truth", "fold", "symbol"),
                  measure.vars = cols)

# Merge rolling sd from prices to the preds_perf
preds_perf = prices[, .(symbol = isin, date, date_prices = date, roll_sd)][
  preds_perf, on = c("symbol", "date"), roll = +Inf]
```

Analizom rezultata u tablici Table 7 uočavamo da postoji značajna varijacija u performansama između različitih modela strojnog učenja - glmnet, nnet, ranger, xgboost i ansambl modela. Svaki model pokazuje svoje specifične snage i slabosti u odnosu na različite evaluacijske metrike.

Najveću točnost pokazuje model nnet (neorinske mreže). Međutim, razlika između modela nije

Table 8. *Udio pozitivnih predviđanja*

```
# Calculate the proportion of positive predictions
positive_preds = preds_perf[, .N, by = .(variable, prediction_positive = value > 0)]
positive_preds = positive_preds[, pr := round(N / sum(N) * 100, 2), by = variable]
positive_preds = dcast(positive_preds, variable ~ prediction_positive, value.var = "pr")
setnames(positive_preds, c("Model",
                           "Udio negativnih vrijednosti",
                           "Udio pozitivnih vrijednosti"))
ft = qflextable(positive_preds) |>
    colformat_double()
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

jako velika. Najlošije rezultate po pokazatelju točnosti pokazuje glmnet. Općenito možemo zaključiti da rezultati svih modela pokazuju relativnost niske performanse. Ovo potvrđuje niski omjer signala i šuma u financijskim podacima, što čini predviđanje kretanja cijena dionica izazovnim zadatkom. Fbeta je pokazatelj koji se vrlo često koristi kod binarnih klasifikacija. Kod ove mjere, najlošrije rezultate pokazuje nnet, koji je pokazivao najbolje rezultate po točnosti. Nijedan model se ne ističe posebno prema mjeri fbeta. Sljedeće dvije mjere otkrivaju mnogo predviđanjima koje generiraju modeli. Mjera tpr je visoka kod svih modela, dok je mjera precision niska. Ovo ukazuje na to da modeli često predviđaju pozitivne klase, ali su te predikcije često pogrešne. To potvrđuje i tablica Table 8 koja pokazuje udio pozitivnih predviđanja u ukupnom broju predviđanja. Vidljivo je da svi modeli imaju mnogo više pozitivnih nego negativnih predikcija. Kada se analiziraju rezultati prema regresijskim mjerama, najbolje rezulate pokazuju modeli ranger, glmnet i medain_resp. Ovi modeli imaju najmanje srednje kvadratne greške i srednje apsolutne greške.

U zaključku, detaljna analiza i usporedba modela pokazuje da nema univerzalno najboljeg modela; umjesto toga, izbor modela trebao bi biti vođen specifičnim zahtjevima i ograničenjima investicijske strategije. Razumijevanje prednosti i slabosti svakog modela ključno je za optimizaciju performansi strojnog učenja u različitim primjenama.

4.2 Backtest

Prikazane mjere uspješnosti modela daju korisne uvide u performanse pojedinih modela, no ne daju uvid u performanse modela u kontekstu investicijske strategije. Kako bi se dobio bolji uvid u performanse modela, u ovom dijelu ćemo provesti jednostavan backtest invesicijskih stategija. Backtest je simulacija performansi investicijske strategije na povijesnim podacima. U ovom radu, investicijska strategija je konstruiranje portfolia dugih pozicija s tjednim rebalansiranjem.

Portfolio se konstruira na temelju predviđanja modela, a svaki tjedan se rebalansira na temelju novih predviđanja.

```
############## BUG ################
# Create portfolio returns
# portfolios = preds_perf[value > 0] # BUG!
# portfolios[, fold := as.integer(fold)]
# # portfolios = portfolios[cv == 1057]
# portfolios = portfolios[
#   , .(dt_ = .(dcast(.SD, fold + date ~ symbol, value.var = "truth"))),
#   by = .(cv, variable)]
# portfolios[, dt_ := lapply(dt_, setnafill, fill = 0)]
# portfolios[, dt_ := lapply(dt_, function(x) as.xts.data.table(x[, .SD, .SDcols = -"fold"]))]
#
# Return.portfolio(portfolios[2, dt_][[1]])
# portfolios[, portfolio_returns := map(dt_, function(x) Return.portfolio(x))]
############## BUG ################
# preds_perf[, weigts_equal := , by = .(date)]

# Define weights for equal, prediction and prediction + volatility portfolios
portfolios = preds_perf[, .(
 weights_equal      = ifelse(value > 0, 1 / nrow(.SD[value > 0]), 0),
 weights_prediction = ifelse(value > 0, value / .SD[value > 0, sum(value)], 0),
 weights_vol        = ifelse(value > 0, (roll_sd / .SD[value > 0, sum(roll_sd, na.rm = TRUE)]), 0),
 truth
), by = .(cv, variable, week)]

# Calculate portfolio returns
portfolio_returns = portfolios[, .(
  returns_equal = sum(weights_equal * truth),
  returns_preds = sum(weights_prediction * truth),
  returns_vol   = sum(weights_vol * truth, na.rm = TRUE)
),by = .(cv, variable, week)]
portfolio_returns = melt(portfolio_returns,
                         id.vars = c("cv", "variable", "week"),
                         variable.name = "weights",
                         value.name =  "returns")
setorder(portfolio_returns, cv, variable, weights, week)
```

Table 9. *Statističke mjere portfolia*

```
# Portfolio returns
calculatePortfolioStats <- function(portfolio_ret) {
  # portfolio_ret = portfolio_returns[cv == 1057 & variable == "median_resp" & weights == "returns_vol",
  # .(week, returns)]
  portfolio_ret = as.xts.data.table(portfolio_ret)
  if (!is.xts(portfolio_ret)) {
    stop("portfolioReturns must be an xts object.")
  }

  # Calculate statistics
  annualizedReturn = Return.annualized(portfolio_ret)[[1]]
  annualizedSD = sqrt(52) * sd(portfolio_ret)
  sharpeRatio = SharpeRatio.annualized(portfolio_ret)[[1]]
  maxDrawdown = maxDrawdown(portfolio_ret)
  sortinoRatio = SortinoRatio(portfolio_ret)[[1]]

  # Create data.table from statistics
  portfolio_perf = data.table(
    "Godišnji povrati" = annualizedReturn,
    `Godišnja SD` = annualizedSD,
    `Sharpov omjer` = sharpeRatio,
    `Maksimalni gubitak` = maxDrawdown,
    `Sortinov omjer` = sortinoRatio
  )

  return(portfolio_perf)
}
portfolio_stats = portfolio_returns[, calculatePortfolioStats(.SD[, .(week, returns)]),
                                    by = .(cv, variable, weights)]

# Flextable
ft = qflextable(portfolio_stats) |>
  colformat_double()
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

Table 10. *Sharpov omjer za svaki model*

```
# Portfolio SR
portfolio_stats_sr = portfolio_stats[, .("Sharpov omjer" = mean(`Sharpov omjer`)),
                                     by = variable]
setorder(portfolio_stats_sr, "Sharpov omjer")

# Flextable
ft = qflextable(portfolio_stats_sr) |>
  colformat_double()
set_table_properties(
  ft,
  width = 1,
  layout = "autofit"
)
```

Tablica Table 9 prikazuje statističke mjere portfolia za svaki model i svaku ugniježđenu metodu križne validacije. Usporedbom statističkih mjera portfolia možemo dobiti uvid u performanse investicijske strategije koja trguje na temelju korištenih modela. Mjere uključuju godišnje povrate, godišnju standardnu devijaciju, Sharpeov omjer, maksimalni gubitak i Sortinov omjer.

Budući da se kao mjera uspješnosti portfolia u literaturi najčešće koristi Sharpeov omjer, kreirana je i posebna talbia koja pokazuje samo Sharpeov omjer za svaki model, pri čemu je uzet prosjek rezultata po križnim validacijama. Rezultati su dani u tablici Table 10.

Modeli glmnet pokazuju najniže godišnje povrate (0% i 8%) s prilično niskom godišnjom standardnom devijacijom (0,08 i 0,09), što rezultira najmanjim Sharpeovim omjerom (0,01 i 0,43) i Sortinovim omjerom (0,04 i 0,01). Ako kao uspješnost koristimo Sharpov omjer, nakon glmneta, po uspješnosti slijede xgboost i mean?resp Modeli xgboost pokazuju poboljšanje u godišnjim povratima (10% i 15%) uz sličnu ili nešto nižu standardnu devijaciju u usporedbi s modelima glmnet. Sharpeov i Sortinov omjer su bolji (0,86 i 1,08 za Sharpeov omjer te 0,07 i 0,11 za Sortinov omjer). Modeli mean_resp pokazuju još veće godišnje povrate (19% i 24%) s većom godišnjom standardnom devijacijom (0,07 i 0,10), ali i dalje zadržavaju prilično visoke Sharpeove i Sortinove omjere (1,08 i 1,21 za Sharpeov omjer te 0,11 i 0,14 za Sortinov omjer). Ansambl median_resp pokazuje još bolje rezultate. To sugerira da ansambl modeli dobro upravljaju rizikom, pružajući pritom solidne povrate. Najbolji model po godišnjem povratu je ranger (random forest) sa 27% prinosa i godišnjom standardnom devijacijom od 0,09. Ipak najbolje rezultate po Sharpeovom omjeru pokazuje neuronska mreža (nnet) sa 22% godišnjim povratima i godišnjom standardnom devijacijom od 0,07. Ovaj model ima Sharpeov omjer 1,53 i Sortinov omjer 0,13. Modeli ranger i nnet pokazuju i znatno bolje rezultate po pitanju rizika. Maksimlani gubitak im je znatno manji od ostalih modela (oko 45%).

Ukratko, tablica pruža uvid u to kako različiti modeli balansiraju između rizika i povrata. Modeli s visokim Sharpeovim i Sortinovim omjerima, kao što su ranger i nnet, izgledaju kao najpriv-

lačnije opcije za investitore koji traže optimalnu ravnotežu između rizika i mogućih povrata.

Radi laše predodžbe o performansama portfolia, prikazat ćemo i grafikon koji prikazuje povrat portfolia kroz vrijeme. Radi bolje usporedbe, prikazat ćemo povrat portfolia za svaki model. Svaki mjesec je konstruiran *lon-only* portfolio koji se sastoji od dionica koje su predviđene kao pozitivne. Portfolio se rebalansira svaki tjedan. Portfolio daje iste težine svakoj dionici.

```
# Merge all returns to DT
# equity_curves = Reduce(function(x, y) cbind(x, y), portfolios[, portfolio_returns])
# colnames(equity_curves) = portfolios[, paste0(variable, "_", cv)]
# equity_curves = equity_curves[, seq(2, ncol(equity_curves), 2)]
# chart.CumReturns(equity_curves, plot.engine = "ggplot2") +
#   theme_minimal() +
#   gg_theme +
#   labs(title = "Krivulja kapitala portfolia")

equity_curves = portfolio_returns[cv == 1057 & weights == "returns_vol",
                                  .(week, variable, returns)]
equity_curves = dcast(equity_curves, week ~ variable, value.var = "returns")
setorder(equity_curves, week)
chart.CumReturns(as.xts.data.table(equity_curves)[, 1], plot.engine = "ggplot2")
```

Krivulja kapitala generira se iz rezultata modela (registry pod `F:/zse/`*) pokretanjem gornjeg koda s* `eval: true`*. U ovoj inačici dokumenta rezultati nisu uključeni pa je prikazan samo kôd.*

Figure 3. *Krivulja kapitala portfolia*

Slika Figure 3 prikazuje krivulje kapitala portfolia za svaki model. Prikazani su rezultati za samo jednu verziju križne validacije (s 1057 foldova). Uočavamo da su krivulje kapitala portfolia za modele mean_resp i median_resp najviše, što upučuje na drugačije rezultateu u odnosu na one prikazane u tablici Table 10. Ovo sugerira da su median_resp i mean_resp modeli najbolji izbor za investitore koji traže optimalnu ravnotežu između rizika i mogućih povrata. Na slici se vidi da nedostaju neke opservacije za GLMNET model jer model u nekim razdobljima predviđa isključivo pad cijena dionica, pa sukladno tome ne drži nikakve pozicije.

5 Zaključak

U radu su primjeni modeli strojnog učenja unutar domene predviđanja prinosa dionica, s posebnim fokusom na hrvatsko tržište dionica. Kroz iscrpno ispitivanje podataka o trgovanju s Zagrebačke burze u razdoblju od 2000. do 2024. godine, istraživanje osvjetljava značajan po-

tencijal koji modeli strojnog učenja imaju u predviđanju očekivanih prinosa dionica, posebno unutar relativno nelikvidnih i plitkih tržišta poput hrvatskog. Inovativna uključivanost širokog spektra prediktora izvedenih iz dnevnih podataka o trgovanju, zajedno s rigoroznom predobradom podataka, naglašava ključnu važnost metodičke pripreme u poboljšanju prediktivne učinkovitosti modela strojnog učenja.

Nalazi otkrivaju nijansirani krajolik performansi modela, pri čemu *radnom forest* i neuronska mreža pokazuju bolje rezultate od ostalih modela. Evidentna visoka učinkovitost modela strojnog učenja u predviđanju prinosa dionica na ZSE pruža uvjerljiv argument za njihovo usvajanje u financijskom predviđanju.

Primjena naprednih metodologija poput Ugniježđene metode križne validacije s pomičnim prozorom (NRWCV) dodatno je doprinijela robustnom okviru evaluacije, naglašavajući sofisticiranost i preciznost potrebnu u prediktivnom modeliranju vremenskih serija. Naše istraživanje ne samo da je demonstriralo održivost modela strojnog učenja u financijskom predviđanju, već je također istaknulo dinamičku interakciju između odabira modela, pripreme podataka i prediktivne izvedbe.

Zaključno, dok put do postizanja visoke točnosti u predviđanjima prinosa dionica ostaje zahtjevan, naša studija nudi vrijedne uvide i čvrstu osnovu za buduće istraživačke pothvate u ovom području. Za investitore, regulatorna tijela i znanstvenike, implikacije našeg istraživanja nadilaze teoretski interes, predstavljajući praktične primjene u odlučivanju o investicijama i nadzoru tržišta. Ovo istraživanje moguće je proširiti na različite načine. Moguće je doati nove modele, eksperimentirati sa duljinom prozora za trenirnje ili testiranje, kreirati različite načine rebalansiranj portfolia i kreiranja portfolia. Moguće je doadavti nove prediktore, te kreirati nove oblike filtriranja prediktora. Očekujem da će nadolazeća istraživanja dodatno razotkriti sposobnosti i ograničenja ovih modela, otvarajući put za njihovu poboljšanu primjenu u stalno evoluirajućem području financijske analize.

6 Literatura

AMINGHAFARI, MINA, i JEAN-MICHEL POGGI. 2007. „FORECASTING TIME SERIES USING WAVELETS“. *International Journal of Wavelets, Multiresolution and Information Processing* 05 (05): 709–24. https://doi.org/10.1142/s0219691307002002.

Barandas, Marília, Duarte Folgado, Letícia Fernandes, Sara Santos, Mariana Abreu, Patrícia Bota, Hui Liu, Tanja Schultz, i Hugo Gamboa. 2020. „TSFEL: Time Series Feature Extraction Library“. *SoftwareX* 11: 100456.

Biau, Gérard, i Erwan Scornet. 2016. „A Random Forest Guided Tour“. *Test* 25 (2): 197–227.

Breiman, Leo. 2001. „Random Forests“. *Machine Learning* 45 (1): 5–32.

Chen, Tianqi, i Carlos Guestrin. 2016. „XGBoost: A Scalable Tree Boosting System“. *Proceedings*

of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–94.

Cutler, Adele, D. Richard Cutler, i John R. Stevens. 2012. „Random Forests". U *Ensemble Machine Learning*, 157–75. Springer.

Friedman, Jerome H. 2001. „Greedy function approximation: A gradient boosting machine". U *Annals of statistics*, 1189–1232. JSTOR.

———. 2002. „Stochastic gradient boosting". *Computational Statistics & Data Analysis* 38 (4): 367–78.

Heston, Steven L., i Nitish R. Sinha. Year. „The Impact of News Sentiment on Stock Returns". *Journal Name* Volume Number (Issue Number): Page Range.

Hyndman, Rob J, i Yeasmin Khandakar. 2008. *Automatic Time Series Forecasting: The forecast Package for R*. https://cran.r-project.org/web/packages/forecast/vignettes/JSS2008.pdf.

Hyndman, Rob, Yanfei Kang, Pablo Montero-Manso, Mitchell O'Hara-Wild, Thiyanga Talagala, Earo Wang, Yangzhuoran Yang, i ostali. 2023. *tsfeatures: Time Series Feature Extraction*. https://github.com/robjhyndman/tsfeatures/issues: CRAN. https://pkg.robjhyndman.com/tsfeatures/.

Kelly, B. T., i D. Xiu. 2023. „Financial Machine Learning". Working Paper NBER Working Paper No. TBD. National Bureau of Economic Research.

Kononenko, Igor. 1994. „Estimating Attributes: Analysis and Extensions of RELIEF". U *European Conference on Machine Learning*, 171–82.

Liaw, Andy, i Matthew Wiener. 2002. „Classification and Regression by randomForest". *R News* 2 (3): 18–22.

Lopez de Prado, Marcos. 2018. *Advances in Financial Machine Learning*. Publisher Address: Publisher Name.

Lubba, Carl H., Sarab S. Sethi, Philip Knaute, Simon R. Schultz, Ben D. Fulcher, i Nick S. Jones. 2019. „catch22: CAnonical Time-series CHaracteristics". *Data Mining and Knowledge Discovery* 33: 1821–52. https://doi.org/10.1007/s10618-019-00647-x.

O'Hara-Wild, Mitchell, Rob Hyndman, Earo Wang, Di Cook, Thiyanga Talagala, i Leanne Chhay. 2023. *feasts: Feature Extraction and Statistics for Time Series*. https://github.com/tidyverts/feasts/: CRAN. http://feasts.tidyverts.org/.

Otto, Sven, i Jörg Breitung. 2023. „Backward CUSUM for testing and monitoring structural change with an application to COVID-19 pandemic data". *Econometric Theory* 39 (4): 659–92. https://doi.org/10.1017/S0266466622000159.

Paul, Ranjit Kumar, Sandipan Samanta, i Md Yeasin. 2022. *WaveletArima: Wavelet-ARIMA Model for Time Series Forecasting*. https://CRAN.R-project.org/package=WaveletArima.

Pavlidis, Efthymios, Alisa Yusupova, Ivan Paya, David Peel, Enrique Martínez-García, Adrienne Mack, i Valerie Grossman. 2016. „Episodes of Exuberance in Housing Markets: In Search of the Smoking Gun". *The Journal of Real Estate Finance and Economics* 53: 419–49. https://doi.

org/10.1007/s11146-015-9531-2.

Phillips, Peter C. B., Shuping Shi, i Jun Yu. 2015. „Testing for Multiple Bubbles: Historical Episodes of Exuberance and Collapse in the S&P 500“. *International Economic Review* 56 (4): 1043–78. https://doi.org/10.1111/iere.12132.

Sag, Mislav. 2023. „finfeatures: A collection of financial feature engineering functions“. https://github.com/MislavSag/finfeatures.

Shaikh, I., i Others. Year. „Using Machine Learning to Predict Stock Market Movements“. *Journal Name* Volume Number (Issue Number): Page Range.

Vasilopoulos, Kostas, Efthymios Pavlidis, i Enrique Martínez-García. 2022. „exuber: Recursive Right-Tailed Unit Root Testing with R“. *Journal of Statistical Software* 103 (10): 1–26. https://doi.org/10.18637/jss.v103.i10.