Estimacija proizvodnje izlazne snage solarnih elektrana metodama strojnog učenja

Mihalić, Leo

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Diplomski sveučilišni studij elektrotehnike

Diplomski rad

ESTIMACIJA PROIZVODNJE IZLAZNE SNAGE SOLARNIH ELEKTRANA METODAMA STROJNOG UČENJA

Rijeka, srpanj 2024.

Leo Mihalić

0069083367

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Diplomski sveučilišni studij elektrotehnike

Diplomski rad

ESTIMACIJA PROIZVODNJE IZLAZNE SNAGE SOLARNIH ELEKTRANA METODAMA STROJNOG UČENJA

Mentor: Prof. dr. sc. Zlatan Car

Komentor: Dr. sc. Nikola Anđelić

Rijeka, srpanj 2024.

Leo Mihalić
0069083367

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
POVJERENSTVO ZA DIPLOMSKE ISPITE

Rijeka, 14. ožujka 2023.

Zavod: **Zavod za automatiku i elektroniku**
Predmet: **Osnove robotike**
Grana: **2.03.06 automatizacija i robotika**

ZADATAK ZA DIPLOMSKI RAD

Pristupnik: **Leo Mihalić (0069083367)**
Studij: Sveučilišni diplomski studij elektrotehnike
Modul: Automatika

Zadatak: **Estimacija proizvodnje izlazne snage solarnih elektrana metodama strojnog učenja**

Opis zadatka:

Opisati probem estimacije izlazne snage u solarnoj elektrani. Napraviti pred obradu i statističku analizu javno dostupnog skupa podataka. Primjeniti različite algoritme strojnog učenja na skup podataka. Za odabir hiperparametara svakog algoritma strojnog učenja razviti i koristiti metodu nasumičnog pretraživanja hiperparametara. Ispitati da li se estimacjska točnost može poboljšati kombinacijom više algoritama strojnog učenja.

Rad mora biti napisan prema Uputama za pisanje diplomskih / završnih radova koje su objavljene na mrežnim stranicama studija.

Leo Mihalić

Zadatak uručen pristupniku: 20. ožujka 2023.

Mentor:

Prof. dr. sc. Zlatan Car

Predsjednik povjerenstva za diplomski ispit:

Prof. dr. sc. Dubravko Franković

IZJAVA

Sukladno članku 9. Pravilnika o diplomskom radu, diplomskom ispitu i završetku diplomskih sveučilišnih studija Tehničkog fakulteta sveučilišta u Rijeci izjavljujem da sam izradio ovaj diplomski rad samostalno, koristeći vlastito znanje i navedenu literaturu, u razdoblju od datuma zadavanja zadatka do datuma predaje

Rijeka, 09.07.2024.

_____

Ime Prezime

ZAHVALA

Zahvaljujem se prof. dr. sc. Zlatanu Caru na pruženoj prilici i mentorstvu pri izradi ovog diplomskog rada.

Zahvaljujem se višem asistentu dr. sc. Nikoli Anđeliću na pruženim savjetima, vodstvu i sveukupnoj pomoći tokom istraživanja i izrade diplomskog rada.

Na kraju htio bi se zahvaliti cjelokupnoj obitelji i djevojci na pruženoj potpori i savjetima tokom cijelog mog studiranja.

Sadržaj

1 UVOD

Fotonaponske elektrane su postrojenja koja pomoću solarnih ćelija spojenih u mrežu proizvode električnu energiju iz energije Sunca. Solarne ćelije proizvode istosmjernu struju i napon koji se pomoću invertera pretvaraju u izmjeničnu struju i napon te se preko dalekovoda i visokonaponskih kablova prenose do potrošača. Veliki problem fotonaponskih elektrana (kao i drugih elektrana na obnovljive izvore energije) je to što ovise o vremenskim uvjetima koji su nepredvidljivi i skloni promjenama.

U elektroenergetskom sustavu bitno je da elektrane mogu u svakom trenutku opskrbiti trošila s traženom energijom. Ako se u nekom trenutku dogodi da trošila zahtijevaju više energije nego što im elektrane mogu proizvesti, može doći do pada cijelog sustava. Iz toga razloga fotonaponske elektrane imaju problem zbog njihove nepredvidljive proizvodnje energije. Uzrok tome je nepredvidljivost radijacije Sunca. U jednom trenutku radijacija može biti visoka i elektrana može proizvoditi energiju u svom punom kapacitetu, ali u drugom trenutku mogu se pojaviti oblaci i time proizvodnja električne energije će se smanjiti.

U ovom radu će se nastojati riješiti problem estimacije proizvodnje energije pomoću fotonaponskih elektrana s predviđanjem izlazne snage elektrane pomoću algoritama strojnog učenja. Za treniranje i validaciju algoritama strojnog učenja koristit će se skup podataka izmjeren u jednoj fotonaponskoj elektrani u periodu od 34 dana. Prije odabira algoritama za treniranje i validaciju na navedenom skupu podataka provedena je pred obrada i statistička analiza skupa podataka. Evaluacijske metrike korištene za validaciju algoritama su: koeficijent determinancije (R^2), kvadratni korjen srednje kvadratne greške ($RMSE$), srednja apsolutna greška (MAE) i Kling-Gupta efficiency (KGE). Na zadanom skupu provesti će se istraživanje točnosti estimacije izlazne snage solarne elektrane pomoću algoritma strojnog učenja s nasumičnim pretraživanjem hiperparametara. Nakon toga isti ti algoritmi će biti validirani kada se koriste s unakrsnom validacijom i nasumičnim odabirom hiperparametara. Na kraju najbolji algoritmi će se koristiti u ansambl metodi kako bi se vidjelo je li moguće dobiti još bolju estimaciju izlazne snage solarne elektrane pomoću dostupnih parametara.

2 PREGLED LITERATURE

U ovom poglavlju bit će navedeni istraživački radovi koji su na sličan način pokušali rješiti navedenu problematiku u ovom radu.

Znanost traži strategije kako ublažiti globalno zatopljenje i smanjiti negativne utjecaje dugotrajnog korištenja fosilnih goriva za proizvodnju električne energije. U tom smislu implementacija i promoviranje obnovljivih izvora energije u različitim slučajevima postaje najefikasnije rješenje. Netočnost u predviđanju energije proizvedene pomoću fotonaponske elektrane je veliki problem za normalan rad električnih mreža. U radu to se pokušava rješiti pomoću algoritama strojnog učenja. Koriste se četiri modela strojnog učenja na podatke dobivene u elektrani koja se nalazi u Kolumbiji. Četiri modela koja se koriste su: *K-Nearest Neighbour (*KNN*)*, *Linearna regresija (*LR*)*, *Artificial Neural Networks (*ANN*)* i *Support Vector Machines (*SVM*)*. Modeli se validiraju pomoću srednjeg kvadratnog korijena srednje kvadratne pogreške ($RMSE$) i srednje apsolutne pogreške (MAE). Najbolji rezultati su dobiveni pomoću ANN modela. U radu kao ulazne podatke koriste se parametri: Sunčeva radijacija, temeratura zraka, vlažnost zraka i brzina vjetra, a kao izlazi parametar koristi se proizvedena snaga. Podaci su mjereni frekvencijom od 5 minuta u periodu od $01.01.2020$ do $30.06.2020.$. Podaci su podjeljeni u omjeru tako da se 70% podataka koristilo za treniranje modela, a preostalih 30% za validaciju modela. U tablici 2.1 prikazani su rezultati svih četiri modela koja su se koristila u radu [8].

Tablica 2.1. Tablica rezultata za modele iz prvog istraživačkog rada.

	KNN	LR	SVM	ANN
RMSE	92.857	94.583	93.644	86.466
MAE	8.8279	8.9632	9.6209	8.409

U jednom drugom radu istraživana je aplikacija različitih modela strojnog učenja za predviđanje izlazne snage fotonaponske elektrane koja se nalazi u Saudijskoj Arabiji. Korištena su tri modela strojnog učenja: *K- Nearest Neighbour (KNN)*, *Višestruka regresija (MR)* i *Decision Tree Regression*, svaki sa svojim setom hiperparametara i funkcija. Za estimaciju izlazne snage korišten je veliki set podataka dobiven mjerenjem radijacije Sunca i temperature zraka svakih sat vremena u periodu od $2016.$ do $2019.$. Podaci su podijeljeni u omjeru gdje se 70% koristi za treniranje modela, a preostalih 30% za validaciju modela. Modeli su se validirali pomoću: srednje apsolutne pogreške (MAE), kvadratnog korjena srednje kvadratne pogreške ($RMSE$), normaliziranog kvadratnog korjena srednje kvadratne pogreške ($nRMSE$) i koeficijentom determinancije (R^2). Od tri odabrana modela najbolji je bio KNN. Rezultat istraživanja pokazuje kako je moguće predviđati

izlaznu snagu fotonaponske elektrane pomoću algoritama strojnog učenja na području Saudijske Arabije. U tablici 2.2 prikazani su rezlutati svih tri modela koja su se koristila u radu [9].

Tablica 2.2. Tablica rezultata za modele iz drugog istraživačkog rada.

	MLR	KNN	DTR
MAE	1.14	0.08	0.13
RMSE	1.48	0.19	0.27
nRMSE	0.69	0.13	0.17
R^2	0.99	0.99	0.99

U radu koji su napisali autori *Malakout, Menhaj i Suratgar* govori se kako je bitno imati točna predviđanja proizvodnje električne energije iz solarne energije kako bi se povećala konkurentnost solarnih elektrana na tržištu energije i kako bi se smanjila ovisnost društva i ekonomije o fosilnim gorivima. To se može postići na način da postoji bolje saznanje o količini proizvedene električne energije iz solarne energije u nekom određenom periodu. U radu se koriste setovi podataka koji sadrže informaciju o vremenskoj prognozi Kalifornije u periodu od 2019. do 2021. Modeli koji se koriste u istraživanju su: *Decision tree regressor (DTR)*, *Light gradient boosing machine (LGBM)* i *Extra tree regressor (ETR)*. Za poboljšanje modela koristi se *10-fold unakrsna validacija*. Modeli su se validirali pomoću: srednja kvadratna pogreška(MSE), kvadratni korjen srednje kvadratne pogreške ($RMSE$) *MAE* i koeficijentom determinancije (R^2). Najbolji model je dobiven pomoću Extra Tree Regressora. U tablici 2.3 prikazani su rezlutati svih triju modela korištenih u radu [10].

Tablica 2.3. Tablica rezultata za modele iz trećeg istraživačkog rada.

	Decision Tree (DTR)	Light Gradient Boosting (LGBM)	Extra Tree (ETR)
MSE	25.013	20.08	8.164
RMSE	5.0002	4.48	2.851
MAE	1.661	2.0007	0.849
R^2	0.998	0.9984	0.999

U istraživačkom radu naziva *Output Power Prediction of Solar Photovoltaic Panel Using Machine Learning Approach* smatra se da je pretvorba solarne energije pomoću fotonaponskih panela jedna od najboljih održivih izvora za proizvodnju energije. Iz toga razloga estimacija izlazne snage fotonaponske elektrane postaje ophodna za njezino efikasno korištenje. Glavni cilj ovog rada je predviđanje izlazne snage fotonaponske elektrane pomoću algoritama strojnog učenja koristeći različite ulazne parametre kao što su: temperatura okoline, sunčeva radijacija, temperatura panela,

itd. Koristit će se tri različita modela strojnog učenja, a to su: *višestruka regresija (MLR)*, *Support Vector Machine Regression (SVM)* i *Gaussian regression (GR)*. Modeli su se validirali pomoću: srednje apsolutne greške (MAE), srednje kvadratne greške (MSE), koeficijenta determinancije (R^2) i preciznosti (ACC). Podaci za treniranje modela su mjereni na području Indijske države Telangana tokom mjeseca veljače $2022.$ godine. Frekvencija mjerenja podataka iznosi 30 minuta. Najbolji model je dobiven pomoću *višestruke linearne regresije*. U tablici 2.4 prikazani su rezlutati svih triju modela korištenih u radu [11].

Tablica 2.4. Tablica rezultata za modele iz četvrtog istraživačkog rada.

	Višestruka regresija (MLR)	Gaussian Regression (GR)	Support Vector Machine (SVM)
MAE	0.04505	0.29665	0.74343
MSE	0.00431	0.15262	1.20482
R^2	0.99818	0.93586	0.45170
ACC	0.99997	0.99530	0.93000

U istraživačkom radu naziva *Machine learning based solar photovoltaic power forecasting: A review and comparison* spominje se da je sve veći interes za energiju dobivenu iz obnovljivih izvora i pad cijena solarnih panela postavljaju fotonaponske elektrane u povoljan položaj za implementaciju unutar elektroenergetskog sustava. Međutim nepredvidljivost sunčeve energije predstavlja izazov i uvodi nestabilnost u sutav. U ovom radu se to nastoji riješiti pomoću algoritama strojnog učenja. Skup podataka koji se koristi u radu prikupljen je na području Nizozemske u periodu od 4 godine, frekvencija uzorkovanja je jedan sat. Ukupni broj podataka iznosi 35065. Modeli su se validirali pomoću: srednje apsolutne grške(MAE), srednje kvadratne greške (MSE), kvadratnog korjena srednje kvadratne greške ($RMSE$), normaliziranog kvadratnog korjena srednje kvadratne greške ($nRMSE$), koeficijenta determinancije (R^2) i Skill score-a (SS). Neki modeli su trenirani pomoću svojih zadanih parametara dok drugim su parametri optimizirani pomoću *grid search* metode. Podaci su podijeljeni u omjeru gdje se 70% podataka koristi za treniranje modela, a ostalih 30% za validaciju modela. U radu je korišteno ukupno 16 modela, a u tablici 2.5 moguće je vidjeti rezultate za 5 najboljih modela modela [12].

Tablica 2.5. Tablica rezultata za modele iz petog istraživačkog rada.

	Random Forest (Optimiziran)	G Boost	XG Boost (Optimiziran)	Lasso	Lasso (Optimiziran)
MAE	143.81	132.76	136.05	152.49	157.67
MSE	63558.40	63705.26	66875.34	69012.69	70375.20
RMSE	252.11	252.40	258.60	262.70	265.28
nRMSE	0.1007	0.1008	0.1033	0.1050	0.1060
R^2	0.72	0.719	0.705	0.696	0.690
SS(%)	37.33	37.26	35.71	34.69	43.05

Optimalno upravljanje sa solarnim elektranama zahtjeva kvalitetne podatke za izgradnju točnih modela za predviđanje sunčeve radijacije. U ovom radu predložena je metodologija bazirana na algoritmima strojnog učenja koji pružaju precizno predviđanje prognoze sunčevog zračenja pomoću modela baziranih na dubokom učenju. Podaci za treniranje modela prikupljeni su s područja Kolumbije. Baza podataka za treniranje modela sadrži podatke skupljene u periodu od $1998.$ do $2019.$ godine. Napravljena je od dvije baze podataka. Jedna baza podataka sadrži mjerenja Sunčeve radijacije s površine Zemlje, dok druga baza podataka sadrži mjerenja Sunčeve radijacije pomoću satelita. Podaci baze podijeljeni su u omjeru gdje se 70% podataka koristi za treniranje modela, 15% podataka za validaciju modela i ostalih 15% za testiranje modela. Modeli korišteni u radu su: *Artificial neural networks (ANN)*, *linearna regresija*, *AdaBoost regressor (ABR)* i *Random Forest regressor (RFR)*. Modeli su se validirali pomoću: koeficijenta determinancije (R^2), kvadratnog korjena srednje kvadratne greške ($RMSE$) i srednje apsolutne greške (MAE). U tablici 2.6 prikazani su rezultati za modele iz rada [13].

Tablica 2.6. Tablica rezultata za modele iz šestog istraživačkog rada.

	ANN	Linearna regresija	AdaBoost	Random Forest regressor
R^2	0.75	0.7	0.57	0.96
RMSE	167	182	219	66
MAE	85	113	172	33

Ovaj rad predstavlja jedan pristup predviđanja proizvodnje solarne energije koji se bazira na algoritmima strojnog učenja. Skup podataka za treniranje modela se sastoji od mjerenja u periodu od $2016.$ do $2018.$ godine na području Maroka. Podaci su podijeljeni tako da 80% podataka je korišteno za treniranje modela, dok je preostalih 20% podataka korišteno za validaciju i testiranje modela. Za odabir prigodnih ulaznih podataka iz skupa korišten je Pearsonov korelacijski koeficijent. Modeli koji su se koristili u radu su: *Random forest regressor (RFR)*, *Linearna regresija (LR)*, *MLP* i *SVM*. Modeli su se validirali pomoću: srednje apsolutne greške (MAE), srednje kvadratne

greške (MSE), kvadratnog korjena srednje kvadratne greške ($RMSE$), maksimalne pogreške ($MAXe$) i koeficijenta determinancije (R^2). U tablici 2.7 prikazani su rezultati za modele iz rada [14].

Tablica 2.7. Tablica rezultata za modele iz sedmog istraživačkog rada.

	Linearna regresija	Random forrest regresor	SVR	MLP
MAE	0.0013	$2.64 \cdot 10^{-5}$	0.04	0.03
MSE	$4.36 \cdot 10^{-6}$	$9.93 \cdot 10^{-7}$	0.005	0.006
RMSE	0.002	0.0009	0.07	0.08
MAXe	0.005	0.09	0.0393	0.0652
R^2	0.9999	0.9999	0.9999	0.9999

Glavni izazov za osiguravanje opsežne i besprijekorne itegracije fotonaponskih elektrana u elektroenergetsku mrežu je poboljšanje točnosti predviđanja dnevnog prihoda snage fotonaponske elektrane. Rad naziva *Day-ahead photovoltaic power production forecasting methodology based on machine learning and statistical post-processing* navedeni problem pokušat će rješiti pomoću metoda strojnog učenja potaknutih podacima i statističkoj postobradi. Podaci korišteni za treniranje i validaciju modela su mjerenu unutar jedne godine i podijeljeni su u omjeru $70\% : 15\% : 15\%$. Prvih 70% podataka korišteni su za treniranje algoritama, 15% podataka su korišteni za validaciju algoritama i ostalih 15% podataka su koristšteni za testiranje. Modeli su se validirali pomoću: normaliziranog kvadratnog korjena srednje kvadratne pogreške ($nRMSE$), skill score-a (SS) i srednje apsolutne postotne pogreške ($MAPE$). Algoritmi koji su se koristili su: Optimalni ansambl model pomoću umjetnih neuronskih mreža s ispravkom postprocesiranja ($MSIP$), referentni model bez ispravka postprocesiranja ($MBIP$) i referentni model (RFM) koji je zapravo normalna umjetna neuronska mreža. U tablici 2.8 prikazani su rezultati za modele iz rada [15].

Tablica 2.8. Tablica rezultata za modele iz osmog istraživačkog rada.

	Model s ispravkom postprocesiranja	Model bez ispravka postprocesiranja	Referentni model
nRMSE	0.0611	0.0705	0.1275
SS	69.33	67.21	-
MAPE	0.047	0.094	0.1932

Ovaj rad predlaže metodologiju za kratkoročno (1 sat unaprijed) predvidanje Sunčevog zračenja. Preložena metodologija je bazirana na meteorološkim podacima i služi za optimiziranje proizvodnje električne energije pomoću fotonaponskih elektrana. Metodologija je zapravo kombinacija algoritama strojnog učenja. Algoritmi strojnog učenja koji se koriste su *k-nearest neighbor (KNN)* i

artificial neural network (ANN). Algoritam *KNN* koristi se kao tehnika predobrade podataka koji služe kao ulazni podaci za algoritam *ANN*. Ovaj algoritam se u radu uspoređuje s bazičnim algoritmom *KNN*. Za validaciju modela koriste se: kvadratni korjen srednje kvadratne pogreške ($RMSE$) i srednja apsolutna pogreška (MAE). Baza podataka koja se koristi za treniranje modela sastoji se od podataka mjerenih na području Tajvana. Rezultati simulacije pokazuju kako *k-NN-ANN* model može predviđati Sunčevu radijaciju 1 sat unaprijed s zadovoljavajućom točnošću. U tablici 2.9 prikazani su rezultati za modele iz rada [16].

Tablica 2.9. Tablica rezultata za modele iz devetog istraživačkog rada.

	KNN	k-NN-ANN
MAE	44	42
RMSE	251	242

Točno predviđanje proizvodnje energije kod fotonaponske elektrane je nužno za optimalnu integraciju navedene tehnologije u postojeći elektroenergetski sustav. Cilj ovog rada je ocijeniti točnost predviđanja izlazne snage fotonaponske elektrane različitih algoritama strojnog učenja. Modeli koji će se ocjenjivati u radu su: *Artificial neural networks*, *support vector regression (SVM)* i *regression trees (RT)*. Modeli će se trenirati s nasumičnim odabirom hiperparametara. Bazu podataka za treniranej modela čine podaci mjereni unutar godine dana. Podaci su podijeljeni u omjeru gdje je 70% podataka korišteno za treniranje modela, 15% podataka za validaciju modela i preostalih 15% za testiranje modela. Modeli su se validirali pomoću: srednje apsolutne postotne pogreške ($MAPE$), kvadratnog korjena srednje kvadratne pogreške ($RMSE$), normaliziranog kvadratnog korjena srednje kvadratne pogreške ($nRMSE$) i skill score-a (SS). U tablici 2.10 prikazani su rezultati za modele iz rada [17].

Tablica 2.10. Tablica rezultata za modele iz desetog istraživačkog rada.

	ANN	SVM	RT
MAPE	0.61	0.75	0.98
RMSE	10.37	15.42	18.15
nRMSE	0.76	1.13	1.33
SS	92.22	88.34	86.33

2.1 Tablica rezultata iz svih citiranih istraživačkih radova

Tablica 2.11. Tablica s rezultatima iz prvih 5 *istraživačkih radova.*

Literatura	Model	R^2	MAE	RMSE	MAPE	nRMSE	MSE	SS	MABE	Accuracy	Max Error
[1]	KNN	-	8.8279	92.857	-	-	-	-	-	-	-
	LR	-	8.9632	94.583	-	-	-	-	-	-	-
	SVM	-	9.6209	93.644	-	-	-	-	-	-	-
	ANN	-	8.409	86.466	-	-	-	-	-	-	-
[2]	MLR	0.99	1.14	1.48	-	0.69	-	-	-	-	-
	KNN	0.99	0.08	0.19	-	0.13	-	-	-	-	-
	DTR	0.99	0.13	0.27	-	0.17	-	-	-	-	-
[3]	DTR	0.998	1.661	5.0002	-	-	25.013	-	-	-	-
	LGBM	0.9984	2.0007	4.48	-	-	20.08	-	-	-	-
	ETR	0.999	0.849	2.851	-	-	8.164	-	-	-	-
[4]	MLR	0.99818	0.04505	-	-	-	0.00431	-	-	0.99997	-
	GR	0.93586	0.29665	-	-	-	0.15262	-	-	0.99530	-
	SVM	0.74343	0.74343	-	-	-	1.20482	-	-	0.93	-
[5]	RF(O)	0.72	143.81	252.11	-	0.1007	63558.40	37.33	-	-	-
	G Boost	0.719	132.76	252.40	-	0.1008	63705.26	37.26	-	-	-
	XG Boost(O)	0.705	136.05	258.60	-	0.1033	66875.34	35.71	-	-	-
	Lasso	0.696	152.49	262.70	-	0.1050	69012.69	34.69	-	-	-
	Lasso(O)	0.69	157.67	265.28	-	0.1060	70375.20	43.05	-	-	-

Tablica 2.12. Tablica s rezultatima iz zadnjih 5 *istraživačkih radova.*

Literatura	Model	R^2	MAE	RMSE	MAPE	nRMSE	MSE	SS	Accuracy	Max Error
[6]	NN	0.75	85	167	-	-	-	-	-	-
	LR	0.7	113	182	-	-	-	-	-	-
	AdaBoost	0.57	172	219	-	-	-	-	-	-
	RF	0.96	33	66	-	-	-	-	-	-
[7]	LR	0.9999	0.0013	0.002	-	-	$4.36 \cdot 10^{-6}$	-	-	0.005
	RF	0.9999	$2.64 \cdot 10^{-5}$	0.0009	-	-	$9.93 \cdot 10^{-7}$	-	-	0.09
	SVR	0.9999	0.04	0.07	-	-	0.05	-	-	0.0393
	MLP	0.9999	0.03	0.08	-	-	0.006	-	-	0.0652
[8]	MSIP	-	-	-	0.047	0.0611	-	69.33	-	-
	MBIP	-	-	-	0.094	0.0705	-	67.21	-	-
	RFM	-	-	-	0.1932	0.1275	-	-	-	-
[9]	k-NN	-	44	251	-	-	-	-	-	-
	k-NN-ANN	-	42	242	-	-	-	-	-	-
[10]	ANN	-	-	10.37	0.61	0.76	-	92.22	-	-
	SVR	-	-	15.42	0.75	1.13	-	88.34	-	-
	RT	-	-	18.15	0.98	1.33	-	86.33	-	-

3 OPIS DATASETA

Dataset koji se koristi u ovom radu za treniranje i validaciju modela strojnog učenja za estimaciju izlazne snage fotonaponske elektrane preuzet je sa stranice *kaggle.com*. U ovom poglavlju biti će detaljnije opisani podaci koji se nalaze unutar navedenog dataseta. Na podacima će se provesti statistička i korelacijska analiza i rezultati tih analiza biti će grafički prikazani. Za obradu i analizu podataka korišten je programski jezik Python s njegovom knjižnicom funkcija naziva *pandas*.

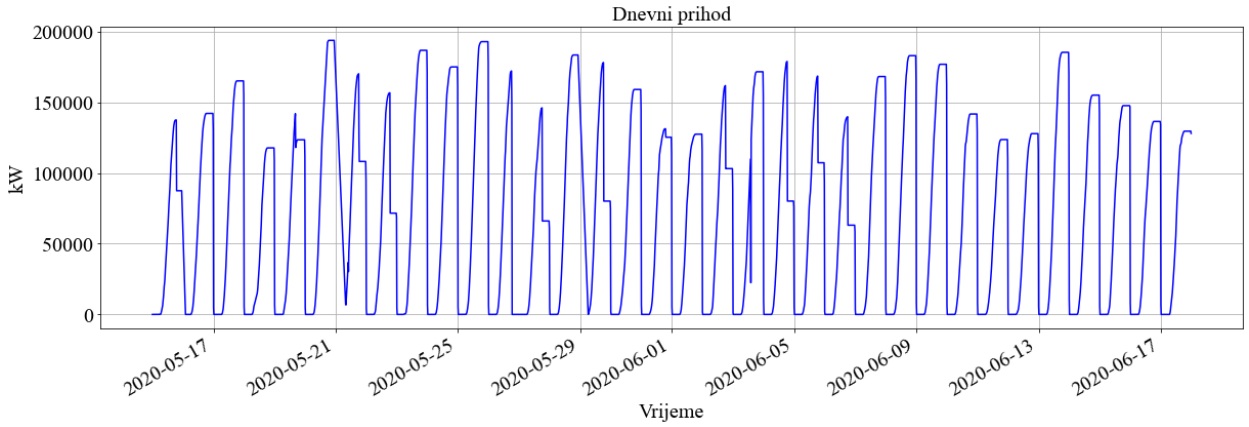
3.1 Općeniti opis dataseta

Podaci koji se nalaze unutar dataseta prikupljeni su u dvije fotonaponske elektrane koje se nalaze u Indiji u periodu od 34 dana. Podaci su grupirani unutar dvije datoteke gdje se u jednoj datoteci nalaze podaci koji su povezani s generiranjem energije dok u drugoj datoteci se nalaze podaci iz senzora. Podaci o generiranju energije prikupljeni su na razini invertera, pri čemu je svaki inverter povezan s više linija solarnih panela. Podaci iz senzora prikupljani su na nivou elektrane tako da su senzori postavljeni na optimalna mjesta unutar same eletrane. Unotač tome što dataset ima podatke prikupljene iz dvije elektrane za treniranje i validaciju modela u ovom radu koristit će se podaci samo iz jedne elektrane iz razloga što rezultati povedene statističke analize pokazuju da podaci iz druge elektrane pogoršavaju točnost modela za estimaciju izlazne snage fotonaponske elektrane.

3.2 Analiza podataka za treniranje modela strojnog učenja

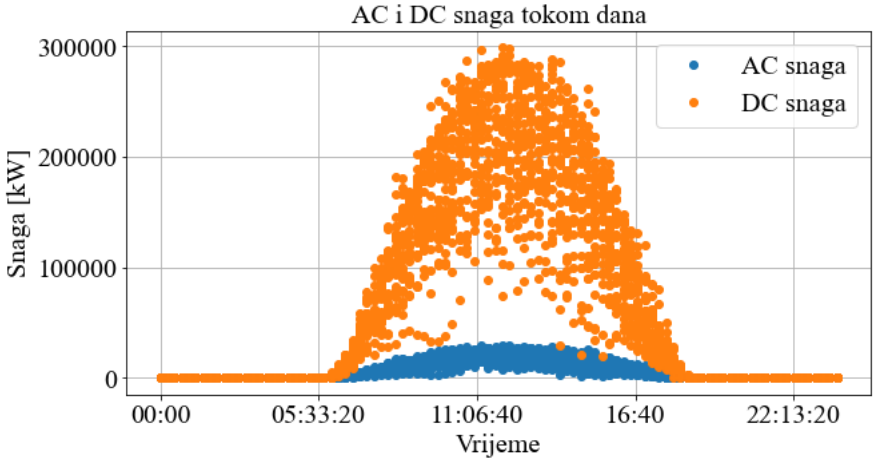
Period mjerenja podataka iznosi 15 minuta. Unutar elektrane se nalazi ukupno 22 invertera i podaci su se pojedinačno prikupljali za svaki inverter posebno. Ukupan broj podataka koji su izmjereni iznosi 68774. Na raspolaganju unutar dataseta ima ukupno 7 varijabli koje su: DC snaga, AC snaga, dnevni prihod, ukupni prihod, temperatura okoline, temperatura modula i iradijacija. Kao izlazna varijabla odabrana je AC snaga, a ostale varijable se koriste kao ulazne varijable modela strojnog učenja izuzev varijable DC snage.

Na slici 3.1 grafički je prikazana varijabla dnevnog prihoda unutar perioda od $15.05.2020.$ do $17.06.2020.$. Iz grafa se može vidjeti zašto je potrebno napraviti precizan model emstimacije izlazne snage solarne eletrane. Graf prikazuje kako dnevni prihod solarne elektrane ima velike oscilacije tokom rada elektrane.

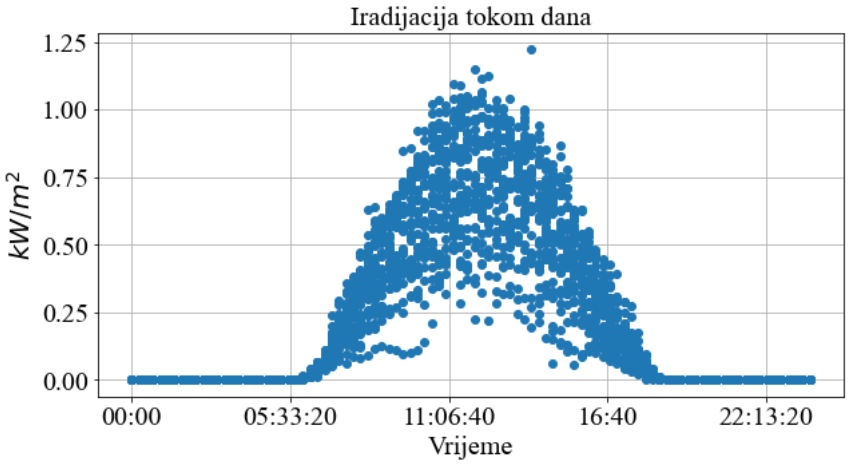


Slika 3.1. Grafički prikaz dnevnog prihoda za period od 15.05.2020. *do* 17.06.2020.

Na slici 3.2 prikazan je graf prosječnih AC i DC snaga svih 22 invertera tokom dana, a na slici 3.3 prikazan je graf iradijacije tokom dana. Iz grafa na slici 3.2 može se vidjeti kako izlazna snaga invertera ovisi o dobu dana. Inverteri daju maksimalnu izlaznu snagu u vremenskom periodu oko 12 sati. Graf na slici 3.2 prikazuje kako je efikasnost invertera izrazito niska zbog velikog omjera između proizvedene DC snage i dobivene AC snage nakon pretvorbe pomoću invertera. Na oba grafa se može vidjeti da oko 6 sati ujutro počinje proizvodnja električne energije, a završava oko 7 sati navečer.

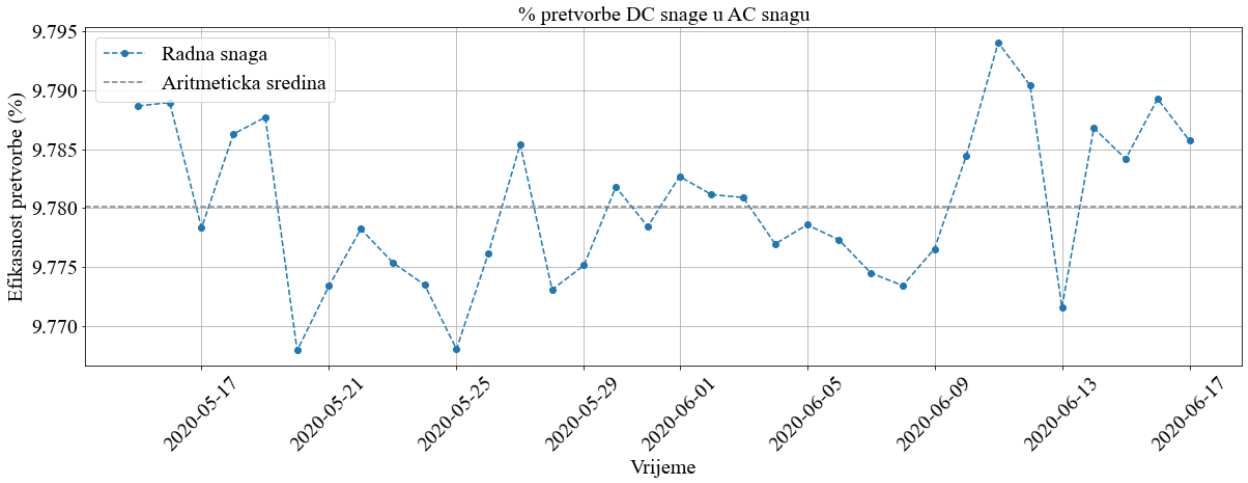


Slika 3.2. Grafički prikaz prosječne DC i AC snage tokom dana.



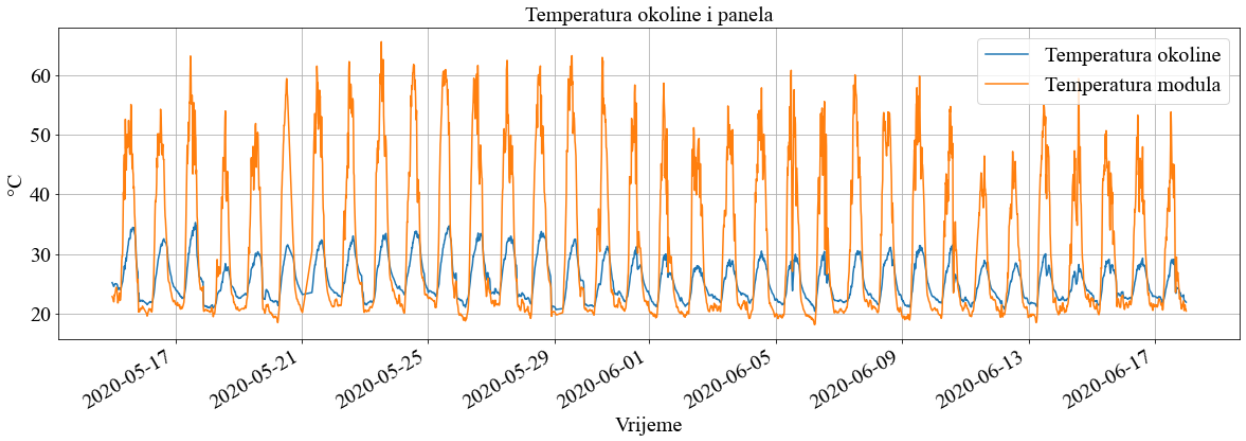
Slika 3.3. Grafički prikaz iradijacije tokom dana.

Na slici 3.4 prikazan je graf efikasnosti elektrane. Prosječna efikasnost elektrane iznosti 9.78% što je izrazito nisko i potvrđuje opažanja koja su zaključena iz grafa na slici 3.2. Za solarnu elektranu efikasnost od 9.78% je izrazito nisko i naznačuje kako je potrebno napraviti servis ili zamjenu invertera kako bi se efikasnost elektrane povećala. Za modele strojnog učenja koji služe za estimaciju izlazne snage niska efikasnost nema velikog učinka na točnost tih modela.



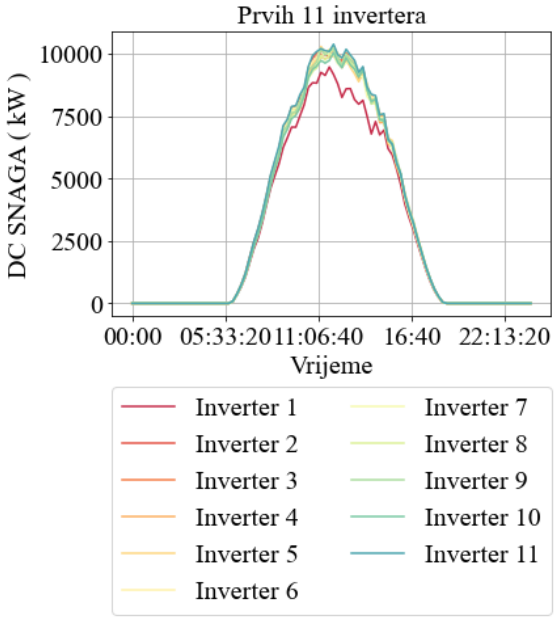
Slika 3.4. Grafički prikaz efikasnosti pretvorbe DC snage u AC snagu.

Na slici 3.5 prikazan je graf temperature okoline i temperature modula za prvu elektranu. Iz grafa se može vidjeti kako je temperatura modula znatno viša nego temperatura okoline. Maksimalna temperatura solarnih panela iznosi $60°\text{C}$ te na grafu se može vidjeti kako u nekim trenucima temperatura modula prelazi tu granicu i zbog toga dolazi do smanjivanja efikasnosti modula [18].

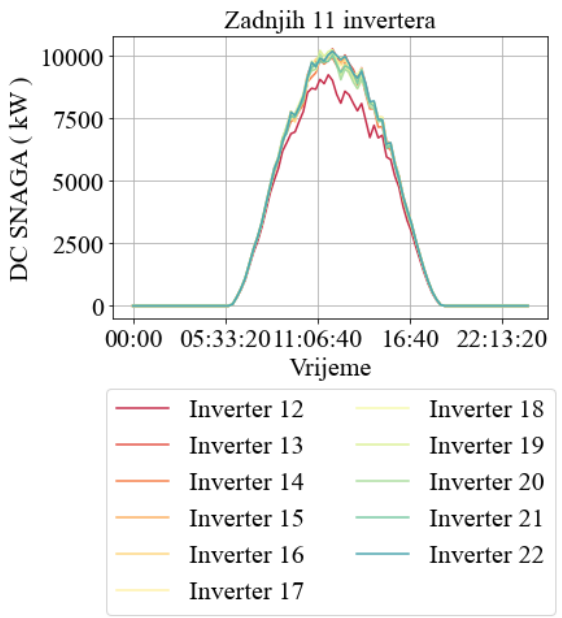


Slika 3.5. Grafički prikaz temperature modula i okoline.

Na slici 3.6 prikazani su grafovi DC snage koja dolazi iz skupova panela na pojedine invertere. Iz grafova može se vidjeti kako na većinu invertera dolazi jednaka DC snaga u svakom trenutku, ali na dva invertera dolazi u prosjeku nešto manja DC snaga. U slučaju ta dva invertera trebalo bi pregledati i ispitati performanse solarnih ćelija koje su spojene na te invertere i vidjeti koji je razlog lošijoj performansi.



(a) Grafički prikaz DC snage koja dolazi na prvih 11 invertera.

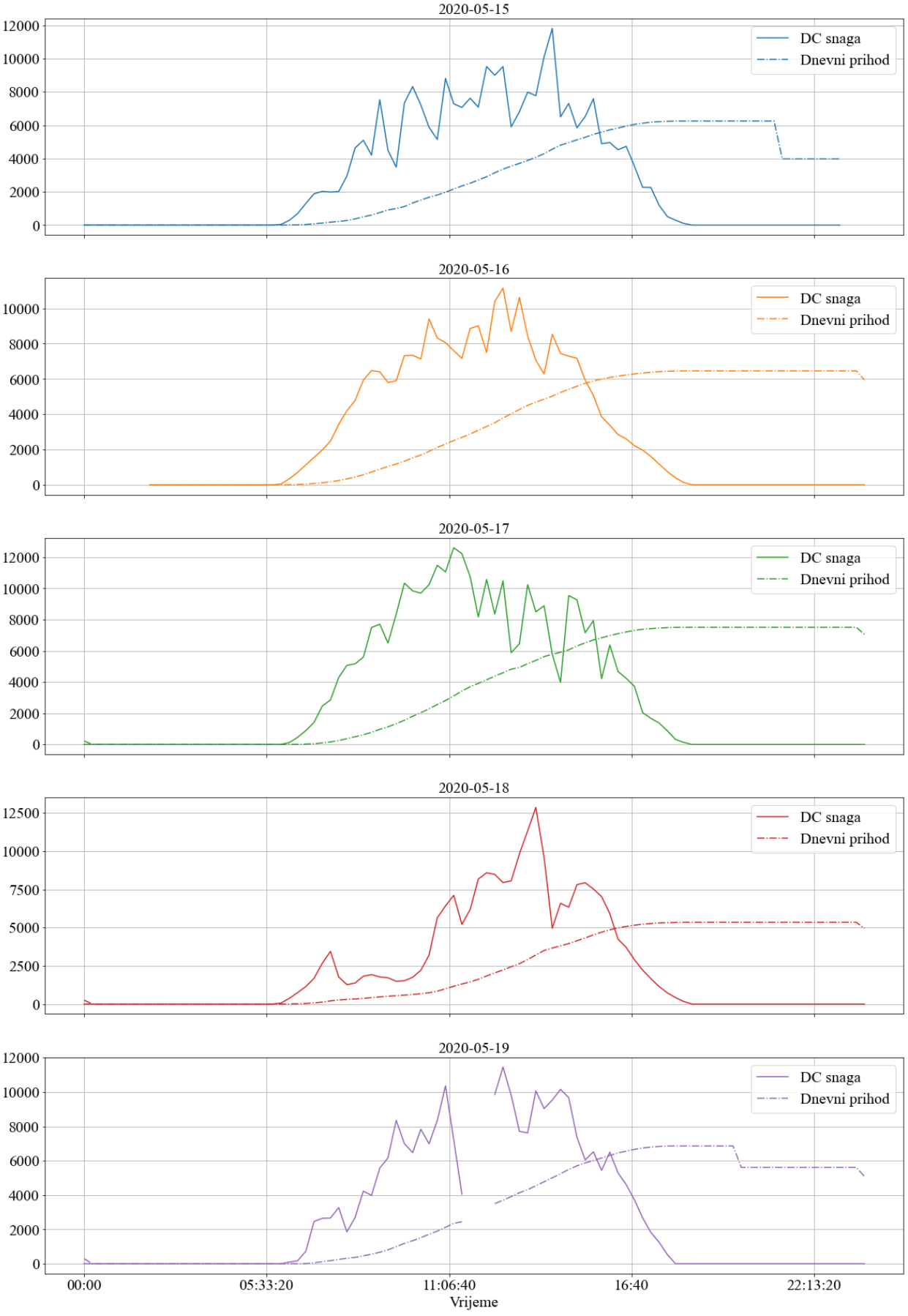


(b) Grafički prikaz DC snage koja dolazi na zadnjih 11 invertera.

Slika 3.6. Grafički prikaz DC snage koja dolazi na invertere.

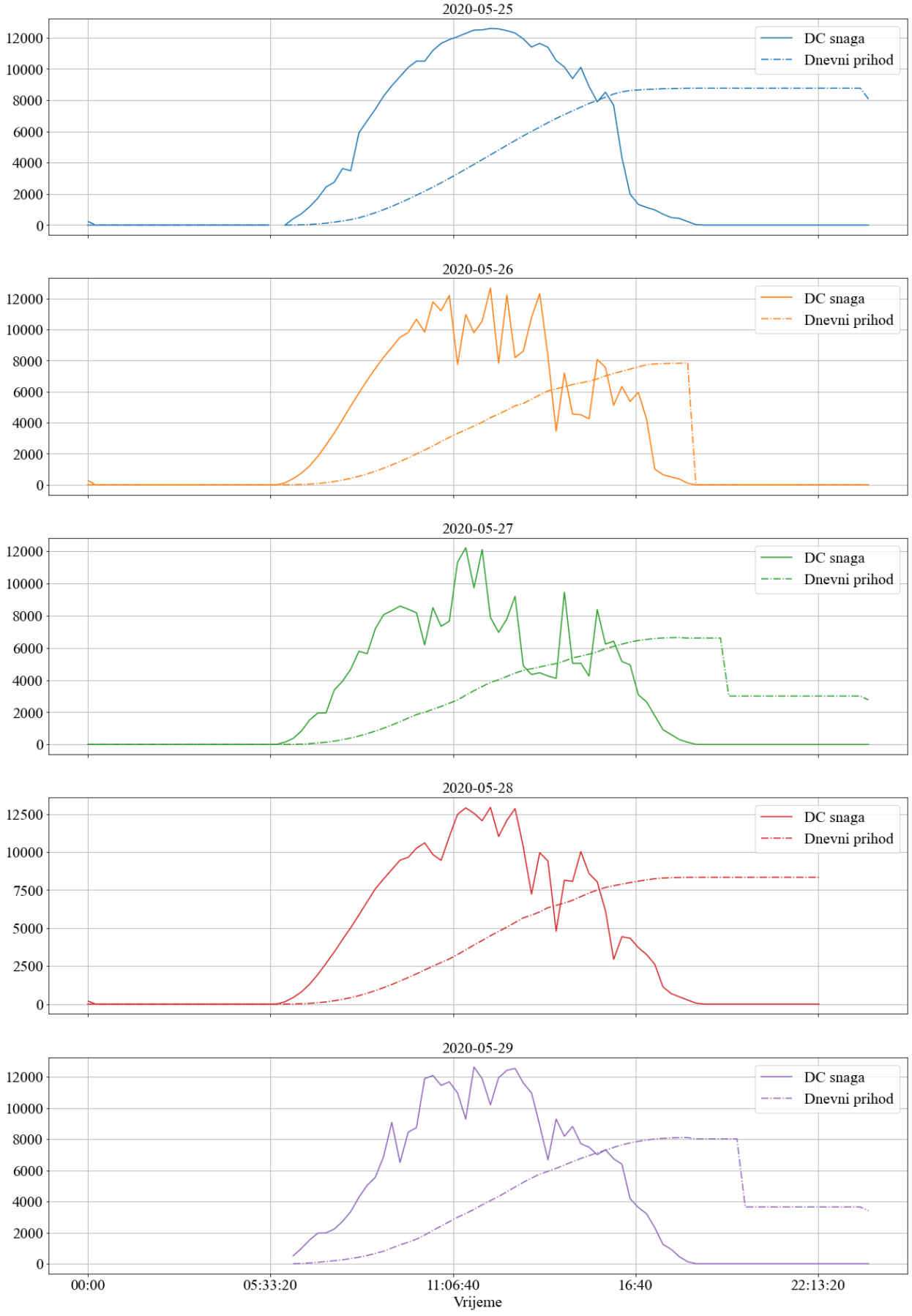
Na slikama od 3.7 do 3.13 prikazani su grafovi DC snage i dnevnog prihoda za prvu elektranu. Na grafovima se može vidjeti kako od 19.05. do 21.05. postoje mjesta na grafu gdje je prekinuta linija, to može biti indikacija da je elektrana bila van pogona. Na slikama od 3.14 do 3.20 prikazani su

grafovi temperature okoline i temperature modula za prvu elektranu. Iz grafova sa slike može se zaključiti da je elektrana od 19.05. do 21.05. bila van pogona jer nisu dostupni podaci temperatura za to vrijeme.

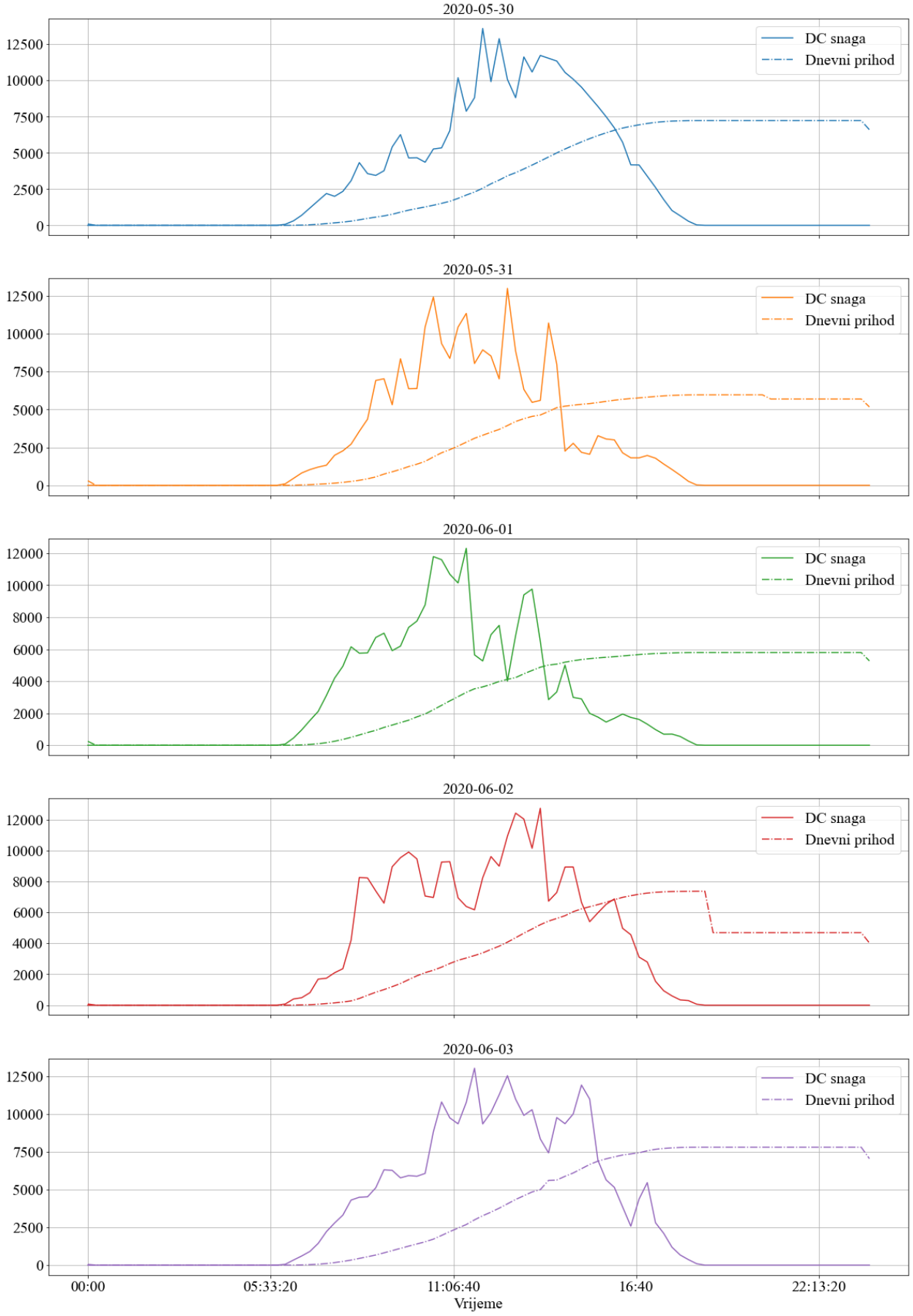


Slika 3.7. Prikaz krivulje DC snage i dnevnog prihoda za dane od 15.05.2020. *do* 19.05.2020.

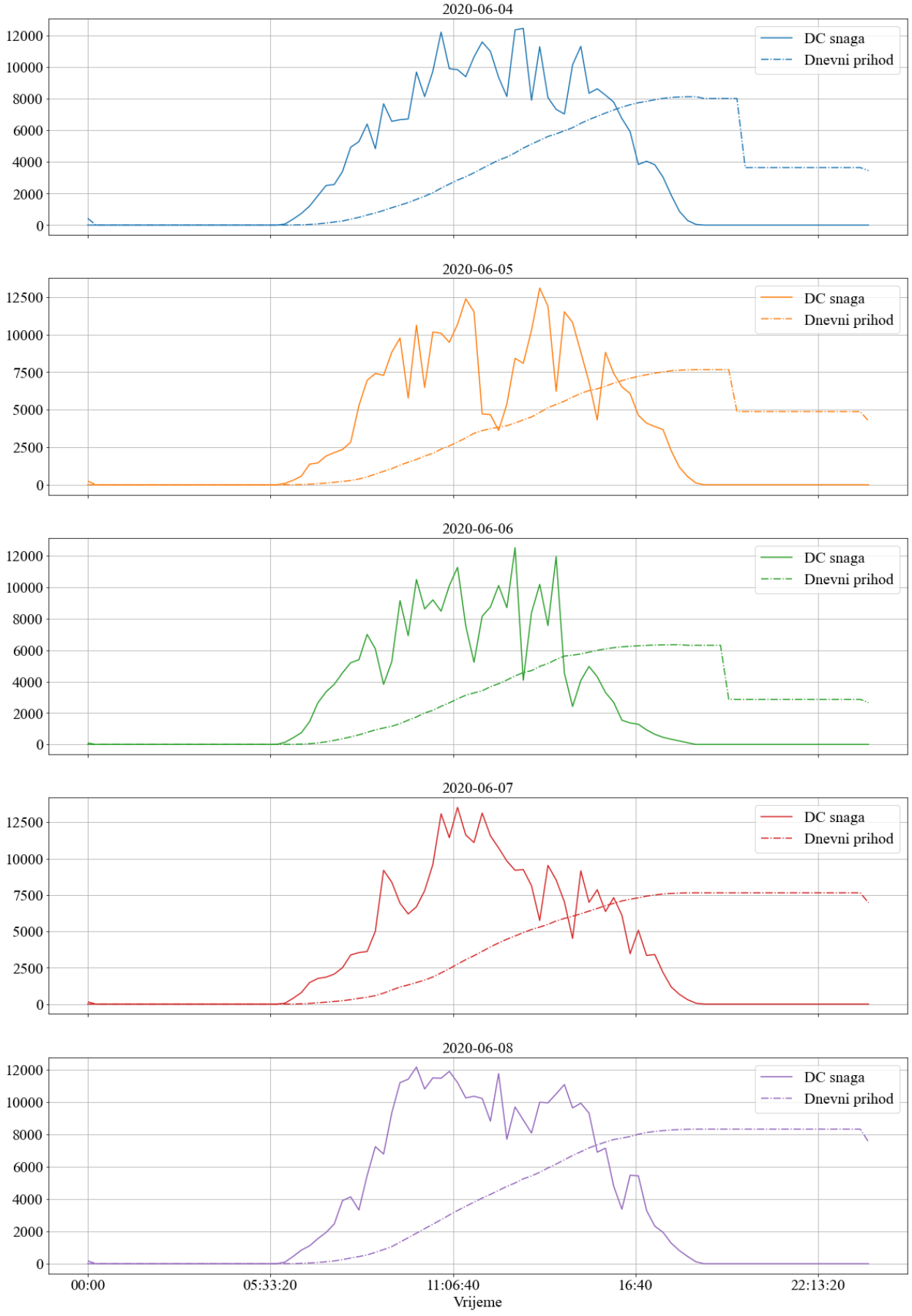
Slika 3.8. Prikaz krivulje DC snage i dnevnog prihoda za dane od 20.05.2020. *do* 24.05.2020.



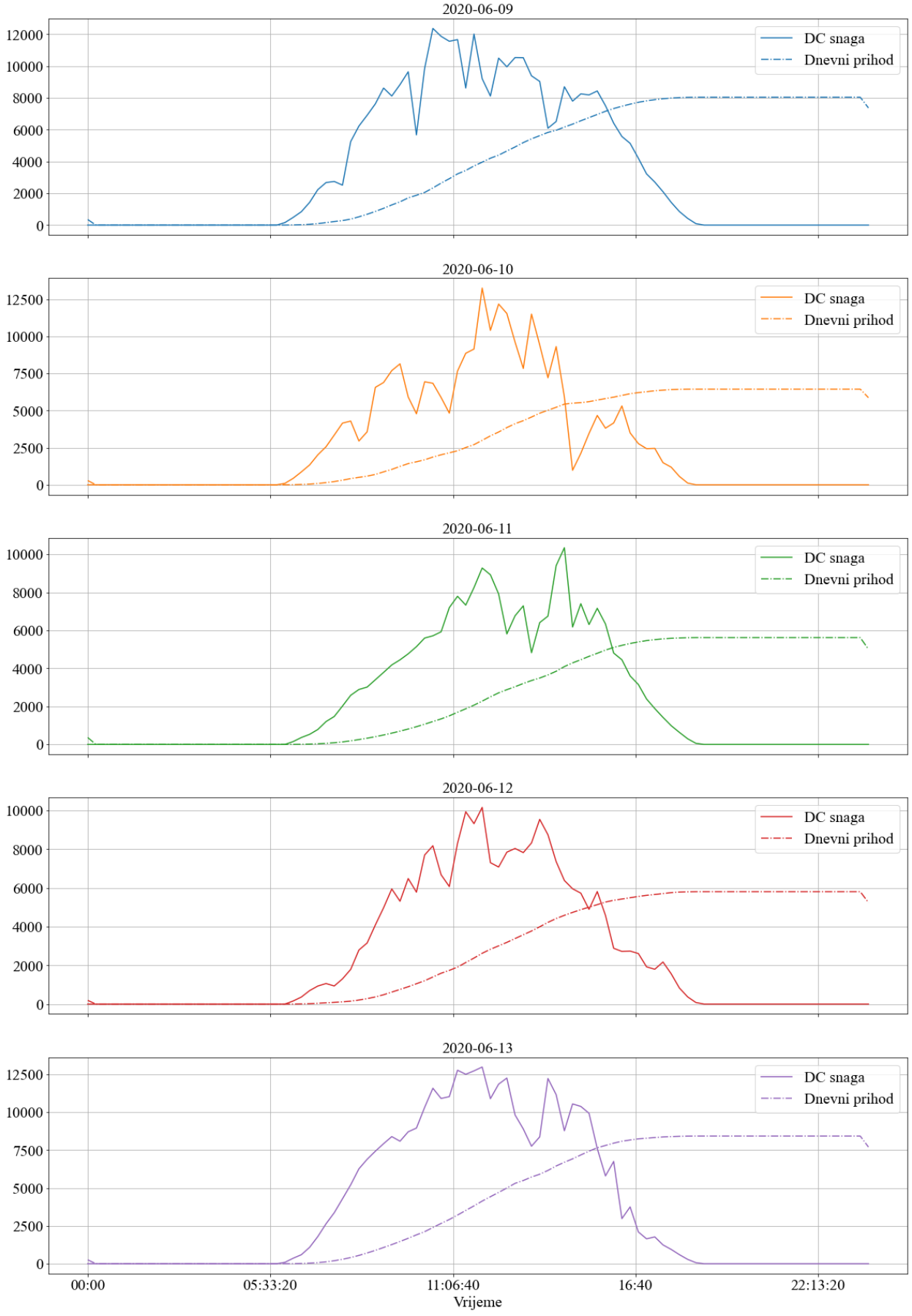
Slika 3.9. Prikaz krivulje DC snage i dnevnog prihoda za dane od 25.05.2020. *do* 29.05.2020.



Slika 3.10. Prikaz krivulje DC snage i dnevnog prihoda za dane od 30.05.2020. *do* 03.06.2020.

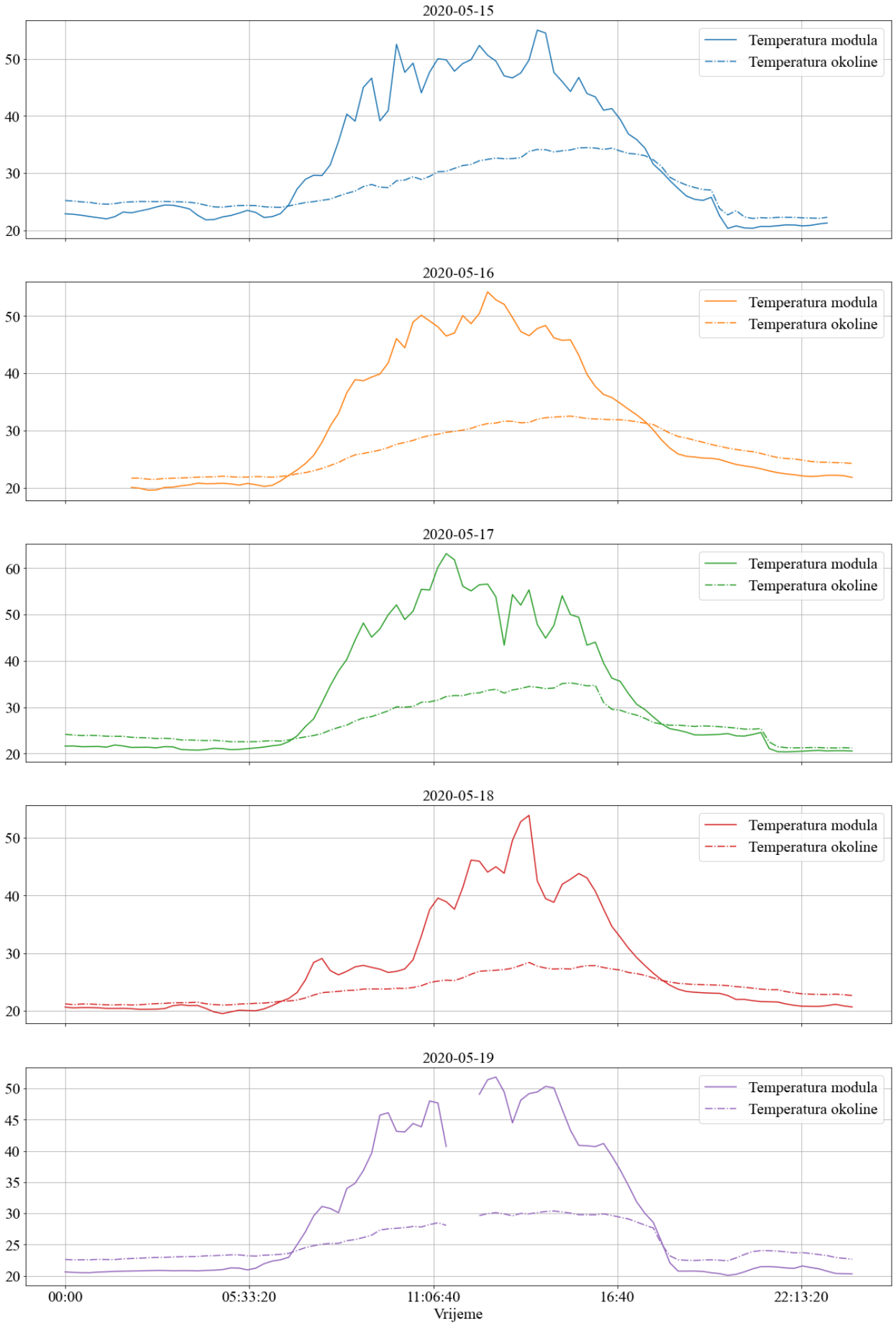


Slika 3.11. Prikaz krivulje DC snage i dnevnog prihoda za dane od 04.06.2020. *do* 08.06.2020.

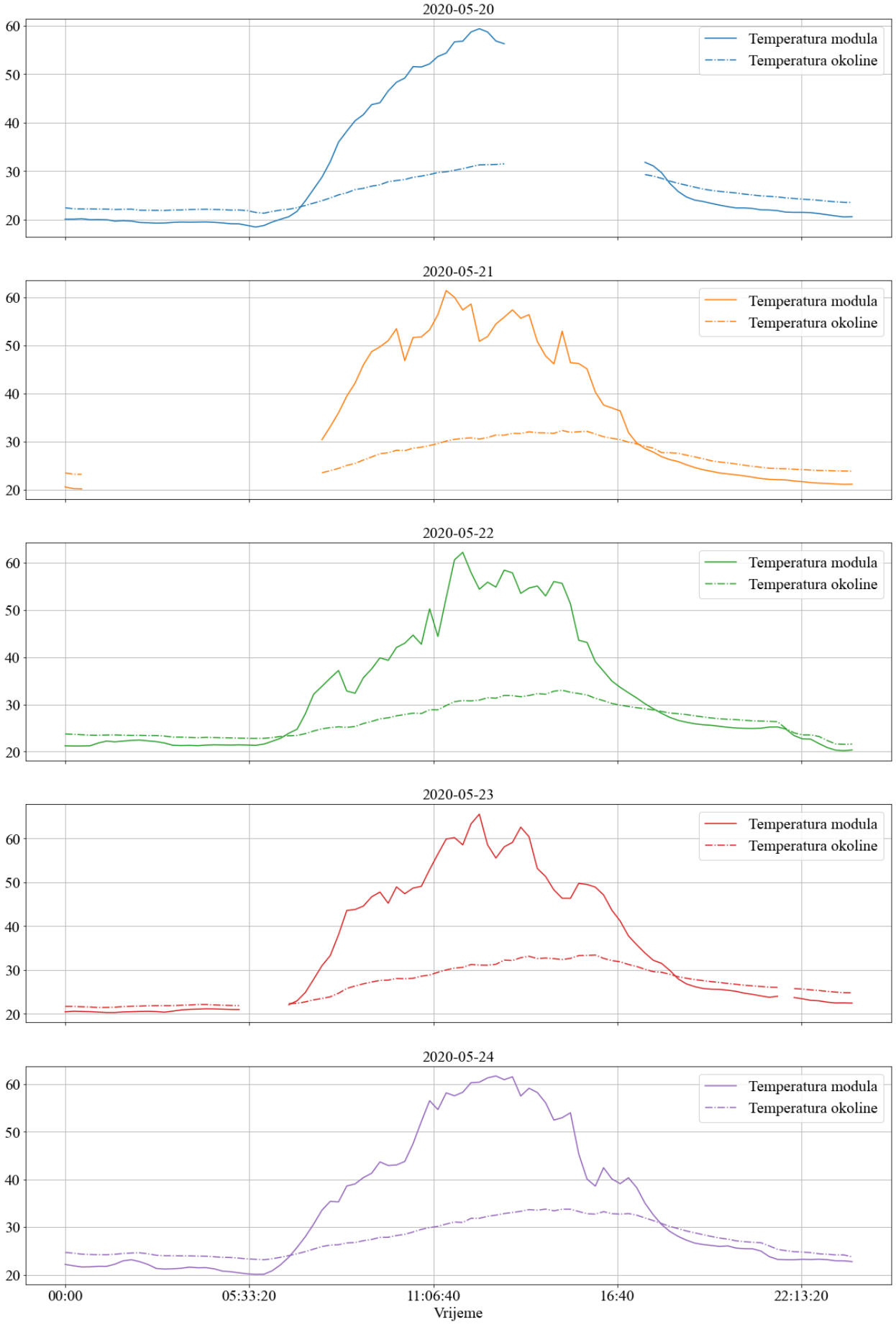


Slika 3.12. Prikaz krivulje DC snage i dnevnog prihoda za dane od 09.06.2020. *do* 13.06.2020.

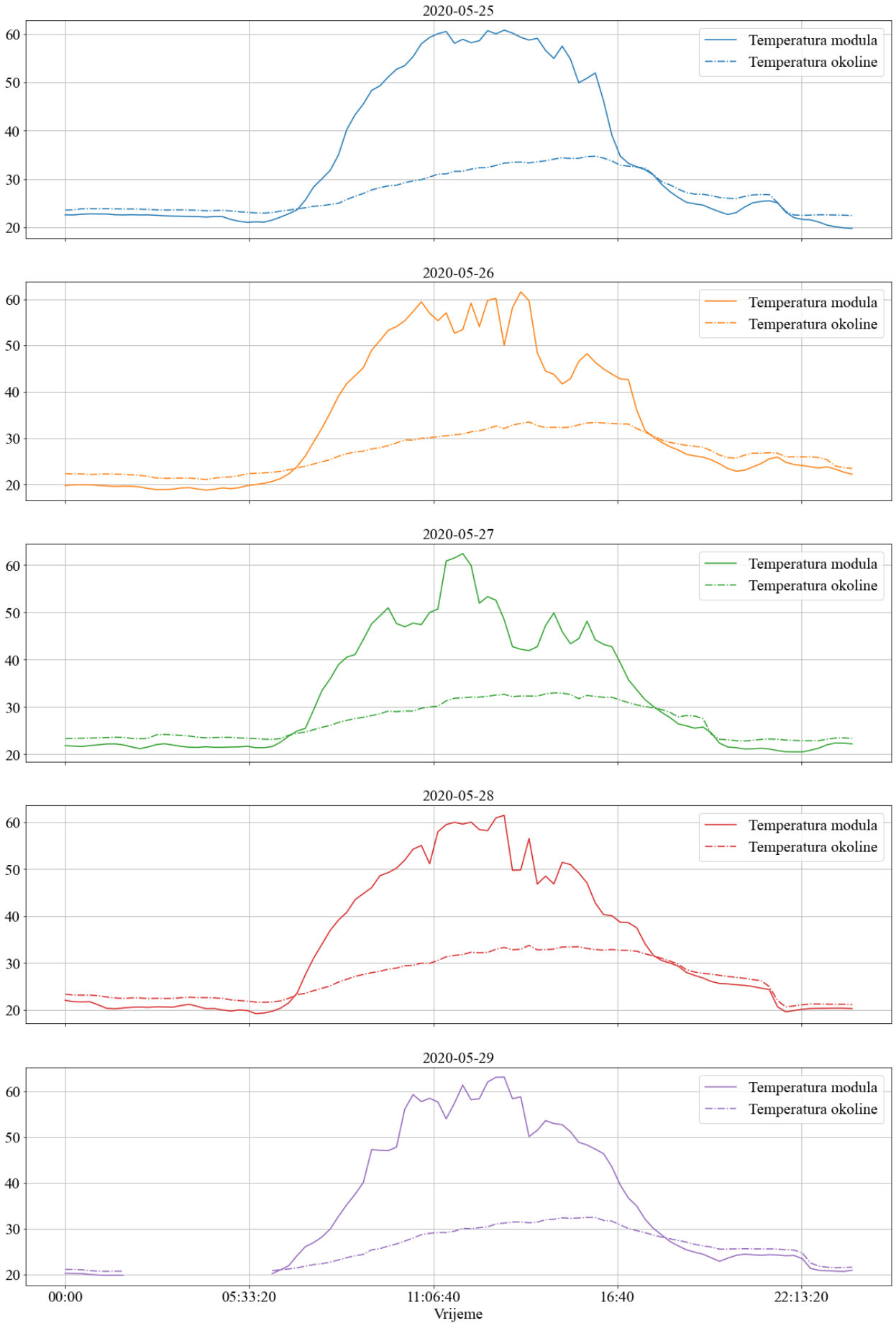
Slika 3.13. Prikaz krivulje DC snage i dnevnog prihoda za dane od 14.06.2020. *do* 17.06.2020.



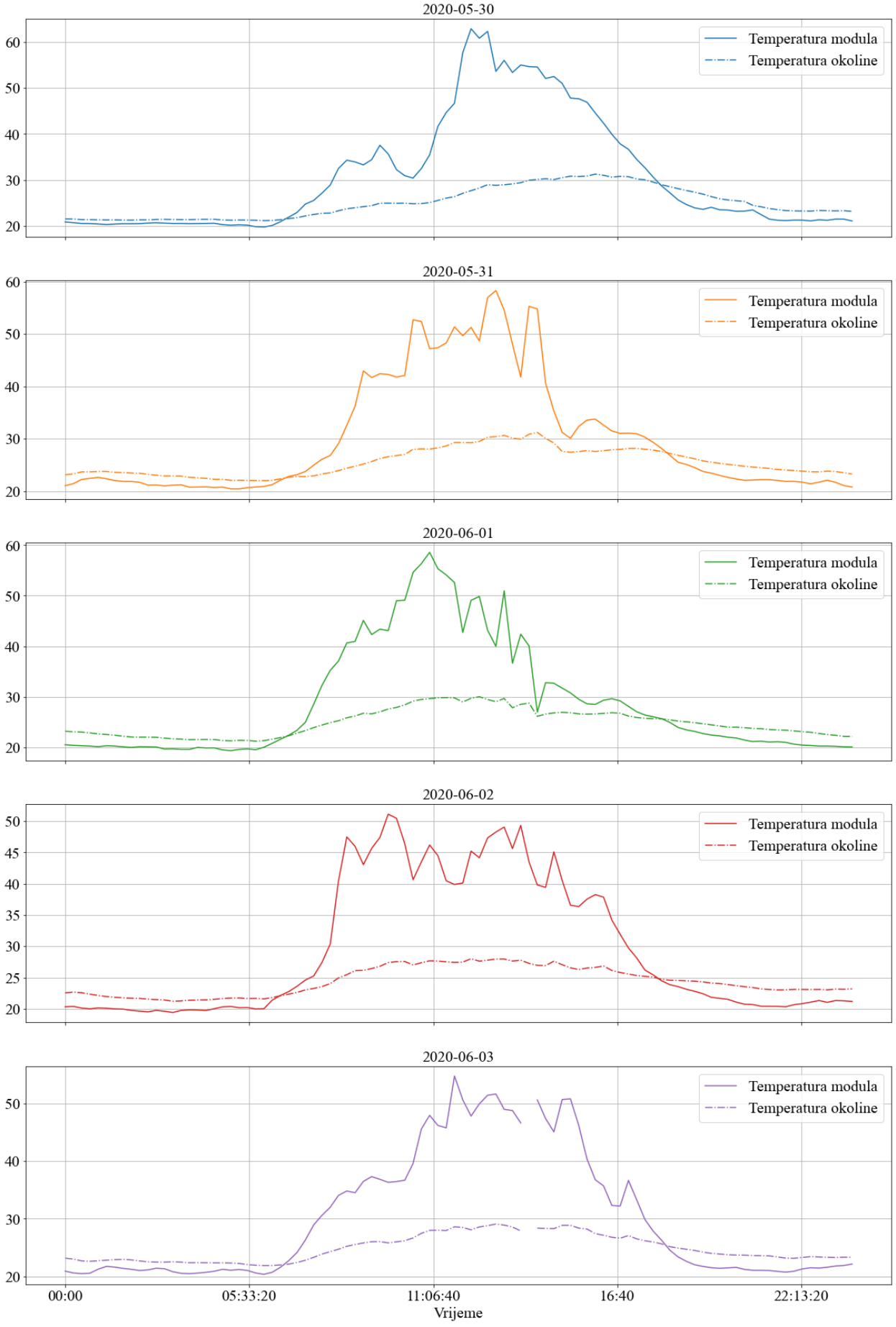
Slika 3.14. Prikaz krivulja temperatura modula i okoline za dane od 15.05.2020. *do* 19.05.2020.



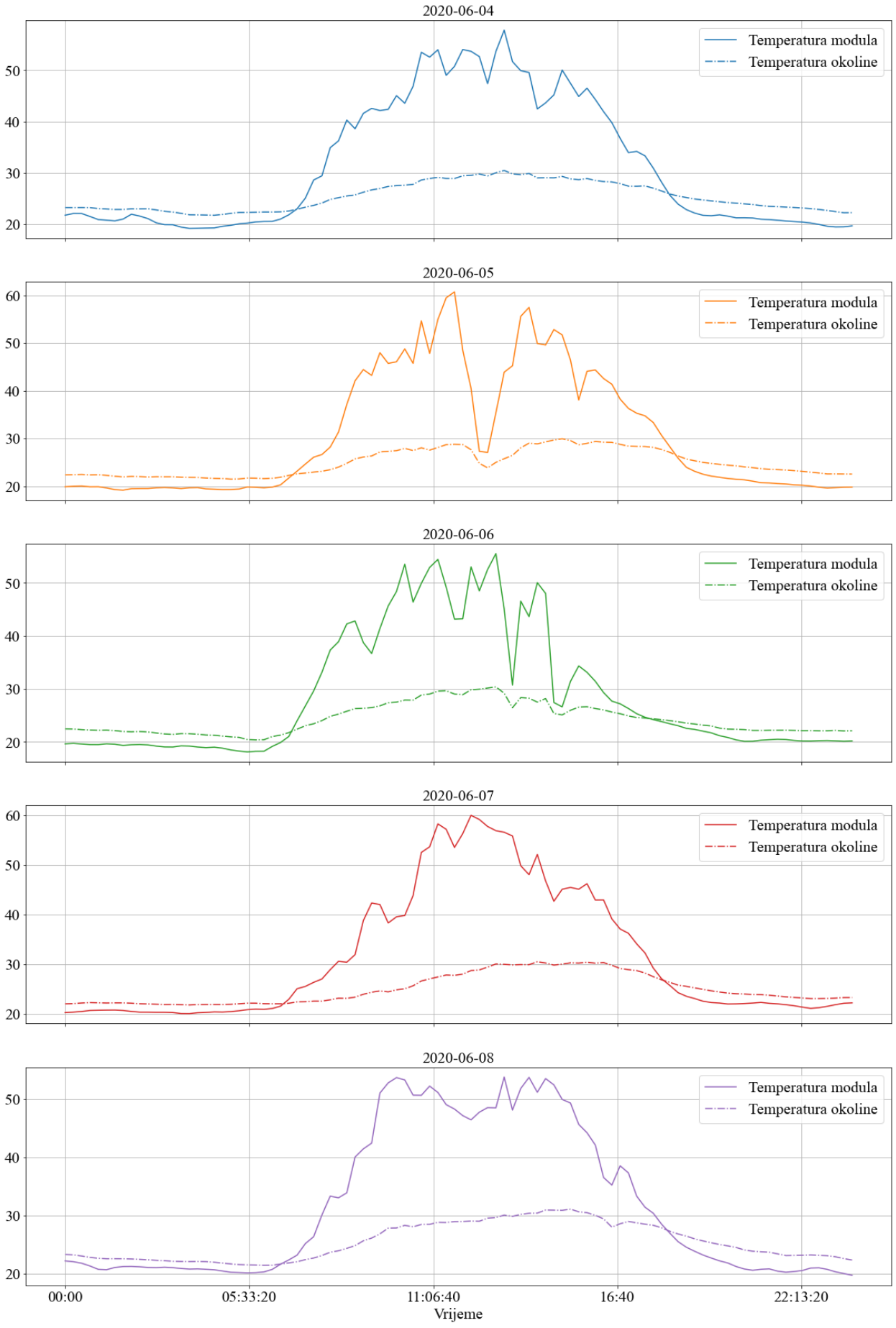
Slika 3.15. Prikaz krivulja temperatura modula i okoline za dane od 20.05.2020. *do* 24.05.2020.



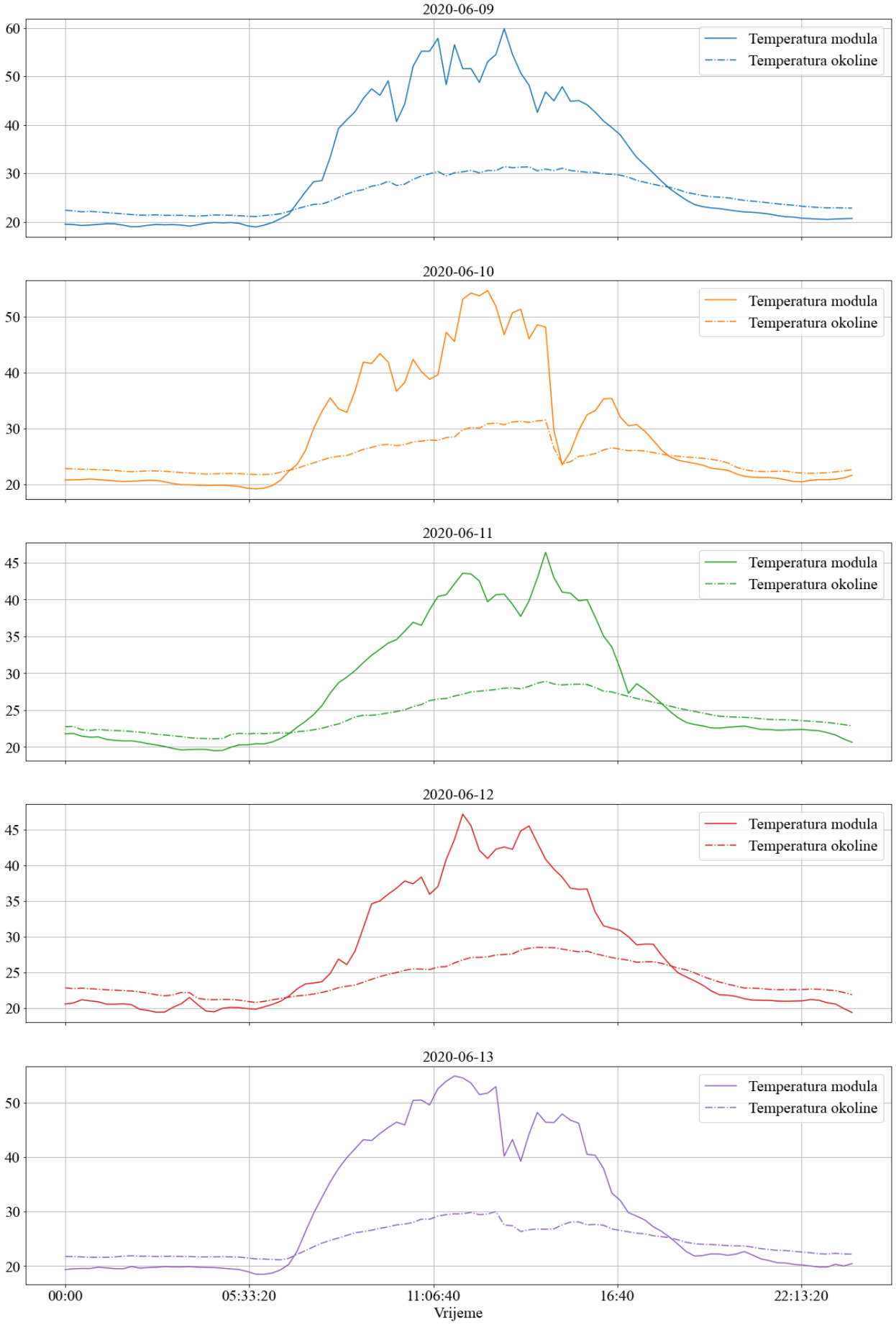
Slika 3.16. Prikaz krivulja temperatura modula i okoline za dane od 25.05.2020. *do* 29.05.2020.



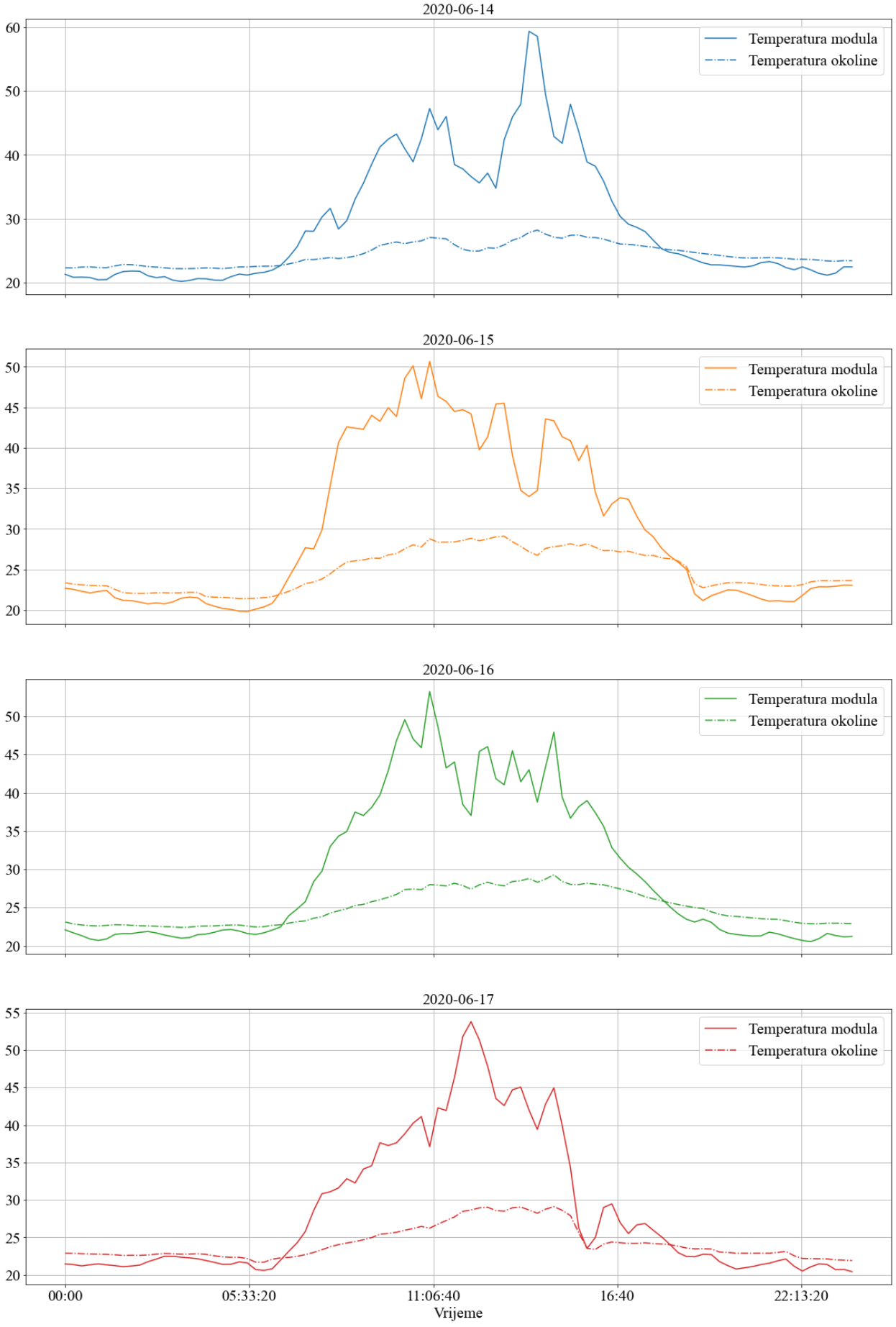
Slika 3.17. Prikaz krivulja temperatura modula i okoline za dane od 30.05.2020. *do* 03.06.2020.



Slika 3.18. Prikaz krivulja temperatura modula i okoline za dane od 04.06.2020. *do* 08.06.2020.



Slika 3.19. Prikaz krivulja temperatura modula i okoline za dane od 09.06.2020. *do* 13.06.2020.



Slika 3.20. Prikaz krivulja temperatura modula i okoline za dane od 14.06.2020. *do* 17.06.2020.

3.3 Deskriptivna statistička analiza i korelacijska analiza podataka

U tablicama 3.1 prikazani su podaci o srednjoj vrijednosti, standardnog devijaciji, maksimalnoj i minimalnoj vrijednosti podataka koji se koriste za treniranje i validaciju modela strojnog učenja. Postoji velika razlika u srednjim vrijednostima DC i AC snage zbog niske efikasnosti pretvorbe. Srednja vrijednost DC snage iznosi 3147.18 kW, a srednja vrijednost AC snage 307.78 kW. Standardna devijacija je također velika za te dvije varijable zbog toga jer postoje periodi preko noći kada nema proizvodnje električne energije. Srednja vrijednost temperature modula je veća od srednje vrijednosti temperature okoline, također je standardna devijacija temperatura modula veća od temperature okoline. Iradijacija ima srednju vrijednost $0.23kW/m^2$ te joj standardna devijacija iznosi $0.3kW/m^2$.

Tablica 3.1. Deskriptivna statistička analiza podataka.

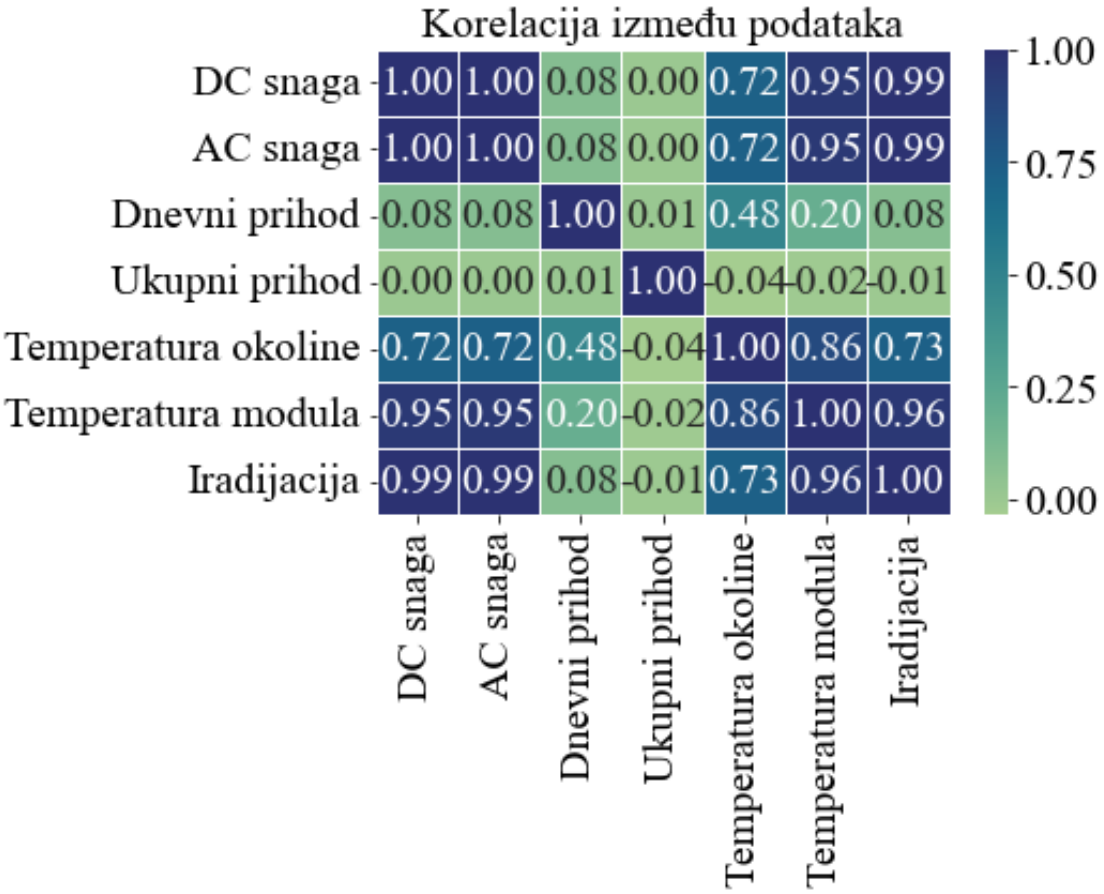
	DC snaga (kW)	**AC snaga (kW)**	**Dnevni prihod (kWh)**	**Ukupni prihod (kWh)**
Srednja vrijednost	3147.18	307.78	68774.0	$6.87 \cdot 10^4$
Standardna devijacija	4036.44	394.39	3145.22	$4.16 \cdot 10^6$
Min	0	0	0	$4.16 \cdot 10^5$
Max	14471.12	410.0	9163.0	$7.85 \cdot 10^6$

(a) Deskriptivna statistička analiza za snage i prihode.

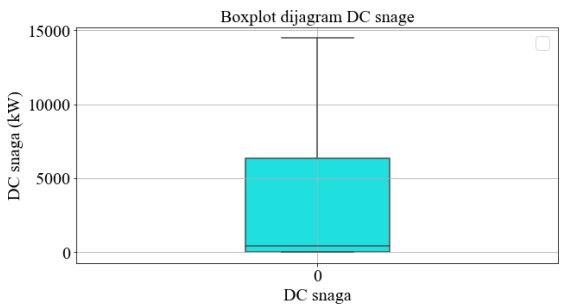
	Temperatura okoline (°C)	**Temperatura modula (°C)**	**Iradijacija (kW/m^2)**
Srednja vrijednost	25.56	31.24	0.23
Standardna devijacija	3.36	12.31	0.3
Min	20.39	18.14	0
Max	35.25	65.55	1.22

(b) Deskriptivna statistička analiza za temperature i iradijaciju.

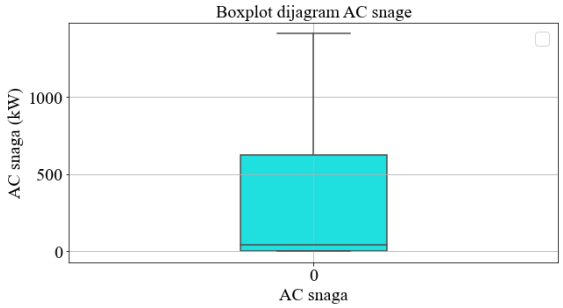
Na slici 3.21 prikazana je korelacijska mapa koja prikazuje korelaciju između podataka. Iz korelacijske mape može se vidjeti kako AC snaga ima visoku korelaciju s DC snagom, Iradijacijom, temperaturom okoline i modula, a malu korelaciju s ukupnim i dnevnim prihodom. Najveću korelaciju AC snaga ima s iradijacijom i iznosi 0.99, zatim slijedi korelacija s temperaturom modula koja iznosi 0.95. Najmanju korelaciju AC snaga ima s ukupnim prihodom i iznosi 0 što znači da ako se ta varijabla ukloni kao ulazni podatak neće biti utjecaja na točnost modela za predviđanje izlazne snage fotonaponske elektrane. Na slici 3.22 prikazani su boxplot grafovi podataka.



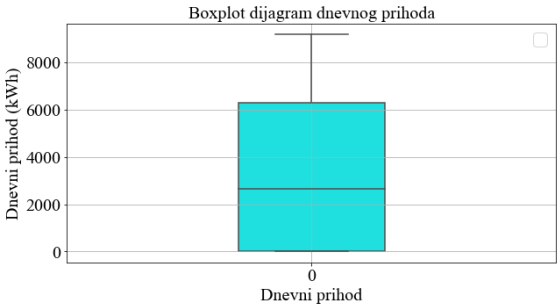
Slika 3.21. Grafički prikaz korelacije između podataka pomoću korelacijke mape.



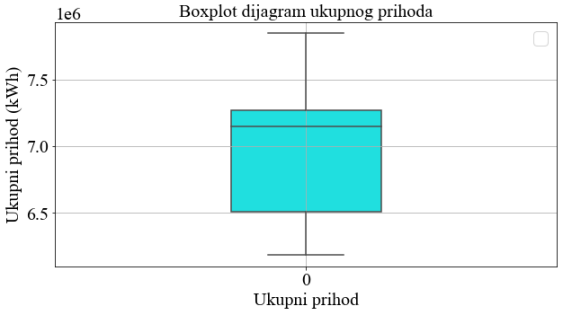
(a) Grafički prikaz Box plot dijagrama za DC snagu.



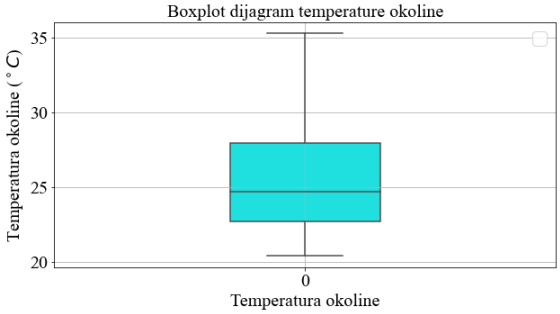
(b) Grafički prikaz Box plot dijagrama za AC snagu.



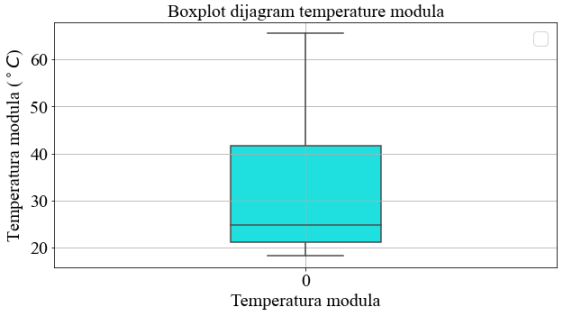
(c) Grafički prikaz Box plot dijagrama za dnevni prihod.



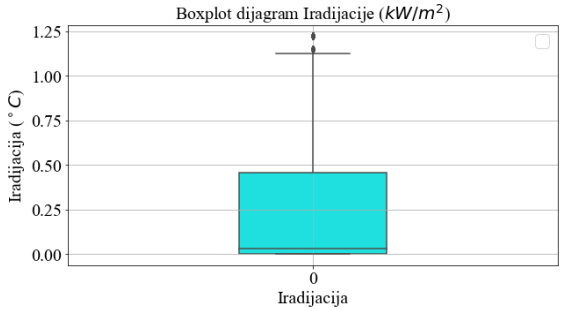
(d) Grafički prikaz Box plot dijagrama za ukupni prihod.



(e) Grafički prikaz Box plot dijagrama za temperaturu okoline.



(f) Grafički prikaz Box plot dijagrama za temperaturu modula.



(g) Grafički prikaz Box plot dijagrama za iradijaciju Sunca.

Slika 3.22. Grafički prikaz DC snage koja dolazi na invertere.

4 METODE STROJNOG UČENJA

U ovom poglavlju bit će detaljnije objašnjeni algoritmi strojnog učenja koji su se koristili za estimaciju izlazne snage solarne elektrane. Uz objašnjenje svakog algoritma biti će prikazana tablica s hiperparametrima koji su se koritili kod nasumičnog pretraživanja. Algoritmi koji su se koristili su: Autoregresivni diferencijalni regresor (ARDR), Multi-Layer Perceptron (MLP), BayesianRidge regresor, Linearna regresija (LR), Huberova regresija (HR), ElasticNet, Lasso i Stacking.

4.1 Autoregresivni Diferencijalni regresor (ARDR)

Autoregresivni modeli su modeli strojnog učenja koji automatski predviđaju sljedeću izlaznu vrijednost u nizu uzimajući izlazne vrijednosti iz prethodnih ulaza u nizu. Autoregresija je statistička tehnika koja se koristi u analizi vremenskih nizova i pretpostavlja da je trenutna vrijednost vremenskog niza funkcija njezinih prošlih vrijednosti. Autoregresivni modeli koriste slične matematičke tehnike za određivanje korelacije između elemenata u nizu. Zatim koriste stečena znanja za predviđanje sljedećeg elementa u nepoznatom nizu. ARDR koristi se za:

- **Sintezu slika**: Autoregresija dopušta *Deep learning* modelima da generiraju slike analizirajući ograničeni broj informacija. Modeli neuronskih mreža za procesiranje slika kao što su *PixelCNN* i *PixelRNN* koriste autoregresivno modeliranje za predviđanje vizualnih podataka pregledavajući postojeće informacije o pixelima.
- **Predviđanje vremenskih nizova:** Autoregresivni modeli su korisni u predviđanju vjerojatnosti događaja u vremenskim nizovima.
- **Data augmentacija:** U nekim slučajevima postoji nedostatak podataka za adekvatno treniranje modela. U tom slučaju koriste se autoregresivni modeli za generiranje novih realističnih podataka za treniranje modela.

Autoregresivni modeli primjenjuju linearnu regresiju s vremenskim kašnjenjem varijabli izlaza preuzetih iz prethodnih koraka. Za razliku od linearne regresije autoregresivni model ne koristi druge nezavisne varijable osim prethodno predviđenih rezultata [19]. Autoregresivno modeliranje može se izraziti sljedećom jednadžbom [20]:

$$X_t = c + \rho_1 X_{t-1} + \rho_2 X_{t-2} + \cdots + \rho_p X_{t-p} + \epsilon_t, \tag{4.1}$$

gdje je:

- X_t vrijednost vremenskog niza u vremenu t,

- c konstanta,
- $\rho_1, \rho_2, \ldots, \rho_p$ parametri modela,
- ϵ_t bijeli šum.

U tablici 4.1 prikazane su granice unutar kojih su se nasumično oabirali hiperparametri za treniranje ARD regresora. Težinske vrijednosti su određene prema Gausovoj distribuciji, parametrom lambda određuje se preciznost distribucije težinskih vrijednosti, a parametrom aplha određuje se preciznost distribucije šuma [21].

Tablica 4.1. Tablica granice hiperparametara korištenih za treniranje AutoRegresivnog diferencijalnog regresora.

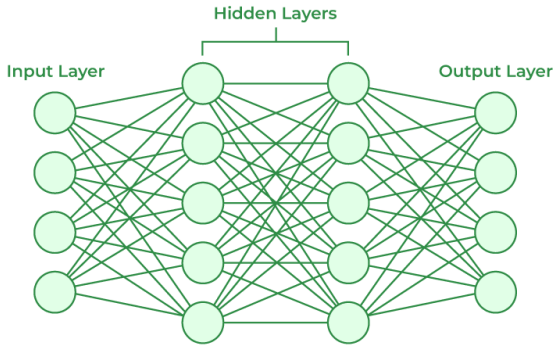
	Donja granica	Gornja granica	Odabir
Number of iteration	100	1000	-
Tolerance	$1 \cdot 10^{-28}$	$1 \cdot 10^{-26}$	-
Alpha 1	$1 \cdot 10^{-5}$	$1 \cdot 10^{-1}$	-
Alpha 2	$1 \cdot 10^{-5}$	$1 \cdot 10^{-1}$	-
Lambda 1	$1 \cdot 10^{-5}$	$1 \cdot 10^{-1}$	-
Lambda 2	$1 \cdot 10^{-5}$	$1 \cdot 10^{-1}$	-
Compute Score	-	-	true, false
Treshold Lambda	1000	100000	-

4.2 Multi-Layer Perceptron (MLP)

Multi-Layer Perceptron je jedna od prvih i temeljnih neuronskih mreža. Klasificira se kao mreža s izravnim prijenosom podataka što indicira da se podaci kreću kroz mrežu samo u jednom smjeru dok se kod rekurzivnih neuronskih mreža (RNN) podaci mogu obrađivati sekvencijalno. MLP algoritam je popularan za rješavanje zadataka kao klasifikacije slika, procesiranje govora, itd. Navedeni zadaci pripadaju pod kategoriju klasifikacije gdje model zapravo kategorizira podatke u različite klase, ali također MLP se može koristiti za rješavanje regresijskih problema gdje model pomaže u predviđanju kontinuiranih ishoda. MLP algoritam je jako dobar u učenju kompleksnih poveznica koje postoje u danom datasetu. Zbog toga može se koristiti u rješavanju nelinearnih regresijskih problema.

Kako je poznato MLP se sastoji od više slojeva sačinjenih od neurona. Svaki neuron u sloju prima podatke na ulaz iz prethodnog sloja, na primljeni podatak primjenjuje aktivacijsku funkciju

i dobivenu vrijednost šalje dalje na sljedeći sloj. MLP neuronska sastoji se od jednog ulaznog sloja, jednog ili više skrivenih slojeva i jednog izlaznog sloja [22]. Na slici 4.1 prikazan je izgled neuronske mreže s 1 ulaznim slojem, 2 skrivena sloja i 1 izlaznim slojem.



Slika 4.1. Prikaz izgleda neuronske mreže (Izvor: [23]).

Prednosti korištenja MLP regresora naspram drugih regresijskih algoritama je njegova sposobnost obrade dataseta s velikim brojem ulaza. Dok se drugi regresijski algoritmi muče s bazama podataka koje imaju veliki broj ulaznih podataka, MLP regresor može se brzo istrenirati na dani uzorak i može dati veliku točnost kod predviđanja. Nedostatak ovog algoritma je njegova interpretabilnost što znači da je kompliciranо objasniti zašto je algoritam napravio određena predviđanja. Još jedan nedostatak algoritma je njegova sklonost pretreniranju [24].

U tablici 4.2 prikazane postavljene granice za nasumučno pretraživanje hiperparametara kod treniranja algoritma strojnog učenja MLP regresor.

Tablica 4.2. Tablica granice hiperparametara korištenih za treniranje MLP-a.

	Donja granica	Gornja granica	Odabir
Broj skrivenih slojeva	2	5	-
Veličina skrivenog sloja	10	200	-
Aktivacijska funkcija	-	-	identity, logistic, tanh, relu
Solver	-	-	adam, lbfgs
Alpha	$1 \cdot 10^{-6}$	$1 \cdot 10^{-2}$	-
Batch size	200	300	-
Learn Rate	-	-	constant, invscaling, adaptive
Max Iteration	200	2000	-
Tolerance	$10 \cdot 10^{-10}$	$1 \cdot 10^{-4}$	-
Number of iteration	10	1000	-

4.3 BayesianRidge regresor

Bayesian linearnoj regresija omogućava prirodni mehanizam preživljavanja unotač manjku podataka ili lošoj distribuciji podataka formulirajući linearnu regresiju koristeći distribucije vjerojatnosti umjesto točkastih procjena. Pretpostavlja se da je izlazna vrijednost izvučena iz distribucije vjerojatnosti umjesto da se procjenjuje kao pojedinačna vrijednost. Jedna od najkorisnijih Bayesian regresija je Bayesian Ridge regresija koja estimira model vjerojatnosti regresijskog modela [25]. Prednosti Bayesian regresije su:

- Velika efikasnost kada dataset nema veliku količinu podataka.
- Može se koristiti bez potrebe za pohranom podataka.
- Može se koristiti na datasetu bez prijašnjeg znanja od tom datasetu.

Nedostaci algoritma su:

- Vrijeme treniranja modela
- Vije vrijedno trenirati model na bazi podataka koja ima veliku količinu podataka. U tom slučaju efikasniji je matematički pristup [26].

U tablici 4.3 prikazane su granice unutar kojih su se nasumično oabirali hiperparametri za treniranje Bayesian Ridge regresora.

Tablica 4.3. Tablica granice hiperparametara korištenih za treniranje Bayesian Ridge regresora.

	Donja granica	Gornja granica	Odabir
Number of iteration	500	1000	-
Tolearance	$1 \cdot 10^{-4}$	$1 \cdot 10^{-3}$	-
Alpha 1	$1 \cdot 10^{-5}$	$1 \cdot 10^{-1}$	-
Alpha 2	$1 \cdot 10^{-5}$	$1 \cdot 10^{-1}$	-
Lambda 1	$1 \cdot 10^{-5}$	$1 \cdot 10^{-1}$	-
Lambda 2	$1 \cdot 10^{-5}$	$1 \cdot 10^{-1}$	-
Compute score	-	-	true, false
Fit intercept	-	-	true, false

4.4 Linearna regresija

Linearna regresija je algoritam strojnog učenja s nazorom koji računa linearnu vezu između ovisne varijable i jedne ili više nezavisnih varijabli tako što prilagođava linearnu jednadžbu promatranim podacima. Postoje dvije vrste linearne regresije: jednostavna linearna regresija i višestruka linearna regresija.

Jednostavna linearna regresija je najjednostavniji oblik linearne regresije i uključuje smo jednu zavisnu i nezavisnu varijablu. Jednadžba jednostavne linearne regresije glasi:

$$y = \beta_0 + \beta_1 X, \tag{4.2}$$

gdje je:

- y zavisna varijabla,
- X nezavisna varijabla,
- β_0 odsječak na y osi,
- β_1 nagib pravca.

Višetruka linearna regresija uključuje više od jedne nezavisne varijable i jednu zavisnu varijablu. Jednadžba višestruke linearne regresije glasi:

$$y = \beta_0 + \beta_1 X + \beta_2 X + \cdots + \beta_n X, \tag{4.3}$$

- y zavisna varijabla,
- X nezavisna varijabla,
- β_0 odsječak na y osi,
- $\beta_1, \beta_2, \cdots, \beta_n$ nagibi.

Cilj algoritma je naći najbolju jednadžbu krivulje pomoću koje će se predviđati zavisne vrijednosti ovisno o nezavisnim vrijednostima. U tablici 4.4 prikazane su postavljene granice za nasumično pretraživanje hiperparametara kod treniranja algoritma linearne regresije [27].

Tablica 4.4. Tablica granice hiperparametara korištenih za treniranje Linearne regresije.

	Donja granica	Gornja granica	Odabir
Fit intercept	-	-	true, false

4.5 Huber regresor

Huber regresija je zapravo L2-regulirani model linearne regresije koji je robusan na outliere. Huber regresor optimizira kvadratnu pogrešku za uzorke gdje je

$$|\frac{y - Xw - c}{\sigma}| < \epsilon, \tag{4.4}$$

i apsolutnu pogrešku za uzorke gdje je

$$|\frac{y - Xw - c}{\sigma}| > \epsilon, \tag{4.5}$$

pri čemu su koeficijenti modela w, sjecište c i skala σ parametri koji se optimiziraju. Parametar σ osigurava da, ako se y skalira prema gore ili dolje za određeni faktor, nije potrebno skalirati ϵ kako bi se postigla ista robusnost. Mora se uzeti u obzir činjenica da različite značajke X mogu biti različitih skala. Huberova funkcija gubitka ima prednost što nije podložna utjecaju outlayer-a, dok istovremeno ne ne zanemaruje potpuno njihov učinak [21]. U tablici 4.5 prikazane su granice parametara za nasumičan odabir hiperparametara za treniranje modela Huber regresora.

Tablica 4.5. Tablica granice hiperparametara korištenih za treniranje Huber regresora.

	Donja granica	Gornja granica	Odabir
Epsilon	1.1	10	-
Max iteration	10000	100000	-
Alpha	$1 \cdot 10^{-10}$	$1 \cdot 10^{-3}$	-
Warm start	-	-	false
Fit intercept	-	-	true
Tolerance	$1 \cdot 10^{-30}$	$1 \cdot 10^{-20}$	-

4.6 ElasticNET

ElasticNET linearna regresija je kombinacija dva popularna algoritma linearne regresije Ridge i Lasso. Koristi kazne od algoritama za treniranje regresijskih modela. Kombinira metode Lasso i Ridge regresije učeći iz njihovih nedostataka kako bi se poboljšala regularizacija statističkih modela. Ridge koristi L2 kaznu dok Lasso koristi L1 kaznu. Kombinacijom tih dviju kazni dobije se funkcija gubitka ElasticNET algoritma:

$$\begin{aligned} ElasticNetMSE &= MSE(y,y_{pred}) + \alpha_1 \sum_{i=1}^{m} |\theta_i| + \alpha_2 \sum_{i=1}^{m} |\theta_i| \\ &= MSE(y, y_{pred}) + \alpha_1 ||\theta||_1 + \alpha_2 ||\theta||_2^2 \end{aligned} \tag{4.6}$$

Umjesto samo jednog regulacijskog parametra sada postoje dva parametra za svaku od kazni. α_1 regulira kaznu L1, a α_2 kontrolira kaznu L2. ElasticNET prelazi u Ridge regresiju ako vrijednost parametra α_1 iznosi 0, a ako vrijednost parametara α_2 iznosi 0 onda ElasticNET prelazi u Lasso regresiju. Alternativno umjesto dva parametra može se koristiti samo jedan parametar α s omjerom kazni $L1Ratio$. U tom slučaju jednadžba glasi:

$$ElasticNetMSE = MSE(y,y_{pred}) + \alpha \cdot (1 - L1Ratio) \sum_{i=1}^{m} |\theta_i| + \alpha \cdot L1Ratio \sum_{i=1}^{m} |\theta_i| \quad (4.7)$$

U ovom projektu EasticNET algoritam koji je treniran koristi jednadžbu 4.7. U tablici 4.6 prikazane su granice parametara za nasumičan odabir hiperparametara za treniranje modela ElasticNET regresora [28].

Tablica 4.6. Tablica granice hiperparametara korištenih za treniranje ElasticNET regresora.

	Donja granica	Gornja granica	Odabir
Alpha	0	1	-
L1 ratio	0	1	-
Fit intercept	-	-	true, false
Precompute	-	-	false
Max Iteration	10000	100000	-
Tolerance	$1 \cdot 10^{-30}$	$1 \cdot 10^{-5}$	-
Warm Start	-	-	False
Random state	0	50	-
Selection	-	-	cyclic, random

4.7 Lasso

Lasso regresija poznata kao L1 regularizacija je popularna tehnika koja se koristi u statističkom modeliranju i strojnom učenju za estimaciju relacija između varijabli i predviđanje. Primarni cilj Lasso regresije je naći balans između jednostavnosti i točnosti modela. To postiže dodavanjem kaznenog parametra modelu linearne regresije, što potiče rijetka rješenja gdje su neki koeficijenti prisiljeni biti točno 0. Zbog te sposobnosti Lasso je koristan za odabir značajki jer može automatski indetificirati i ukoniti varijable koje su nebitne ili se ponavljaju.

Objašnjenje načina rada Lasso algoritma:

1. **Model linearne regresije:** Lasso regresija počinje sa stantardnim modelom linearne regresije koji pretpostavlja linearnu vezu između neovisnih varijabli i ovisne varijable.

2. **L1 regularizacija:** Lasso regresija uvodi dodatni kazneni parametar temeljen na apsolutnim vrijednostima koeficijenata. L1 regularizacija je suma apsolutnih vrijednosti koeficijenata pomnoženih s parametrom podešavanje λ. Jednadžba L1 regularizacije glasi:

$$L_1 = \lambda(|\beta_1| + |\beta_2| + \cdots + |\beta_p|), \tag{4.8}$$

   gdje je:

   - λ parametar podešavanja koji kontrolira količinu primjenjene regulacije,
   - $\beta_1, \beta_2, \ldots, \beta_p$ su koeficijenti modela.

3. **Funkcija cilja:** cilj Lasso regresije je naći vrijednost koeficijenata koji minimiziraju sumu kvadratnih progrešaka između predviđenih vrijednosti i stvarnih vrijednosti, uz minimizaciju parametra L1

4. **Smanjivanje vrijednosti koeficijenata:** dodavanjem parametra L1 Lasso regresija može smanjivati vrijednost koeficijenata prema 0. Kada je parametar λ dovoljno velik, vrijednost nekih koeficijenata iznosi točno 0. Ovo svojstvo je korisno za odabir značajki jer varijable s koeficijentima koji su jednaki 0 su efikasno uklonjene iz modela.

5. **Parametar podešavanja λ:** Izbor vrijednosti parametra λ je kritičan za Lasso regresiju. Velika vrijednost parametra povećava regularizaciju i time se većina vrijednosti koeficijenata smanjuje prema 0, dok mala vrijednost parametra λ dopušta više varijabli da imaju vrijednost veću od 0.

6. **Prilagođavanje modela:** za estimaciju koeficijenata algoritma Lasso regresije koristi se optimizacijski algoritam za minimiziranje funkcije cilja.

U tablici 4.7 prikazane su granice parametara za nasumičan odabir hiperparametara za treniranje modela Lasso regresora [29].

Tablica 4.7. Tablica granice hiperparametara korištenih za treniranje Lasso regresora.

	Donja granica	Gornja granica	Odabir
Alpha	0.1	10	-
Fit intercept	-	-	true, false
Max iteration	1000	10000	-
Tolerance	$1 \cdot 10^{-30}$	$1 \cdot 10^{-5}$	-
Warm start	-	-	false
Random state	0	50	none
Selection	-	-	cyclic, random

4.8 Stacking ansambl

Ansambl algoritmi su metode koje koriste više algoritama strojnog učenja kako bi stvorili optimalni model za predviđanje. Model koji je stvoren na taj način ima bolje performanse od samostalnih baznih modela. Ostale aplikacije ansambl modela je odabir bitnih značajki unutar podataka, povezivanje podataka, itd. Ansambl algoritmi primarno se klasificiraju u tri skupine, a to su: *Bagging*, *Boosting* i *Stacking* [30].

Primarna ideja Stacking algoritma je da se predikcije osnovnih modela proslijede u model višeg nivoa, poznat kao finalni estimator, koji ih zatim kombinira kako bi se dobila konačna predikcija zadane vrijednosti. Detaljni način rada Stacking algoritma je:

1. **Priprema podataka:** prvi korak je priprema podataka za treniranje algoritma. U to spada izdvajanje bitnih značajki iz podataka, čišćenje podataka i dijeljenje baze podataka na podatke za treniranje i podatke za validaciju modela.

2. **Odabir modela:** U ovom koraku odabiru se bazni modeli koji će se koristiti u ansambl modelu. Odabire se širok izbor modela kako bi se osiguralo da proizvode različite vrste pogrešaka i međusobno se nadopunjuju.

3. **Treniranje baznih modela:** nakon odabira modela, oni se treniraju na podacima za treniranje. Kako bi se osigurala raznolikost, svaki model se trenira koristeći različiti algoritam ili skup hiperparametara.

4. **Predviđanja i validacijski set podataka:** kada se bazni modeli istreniraju, koriste se za predviđanje na validacijskom setu podataka.

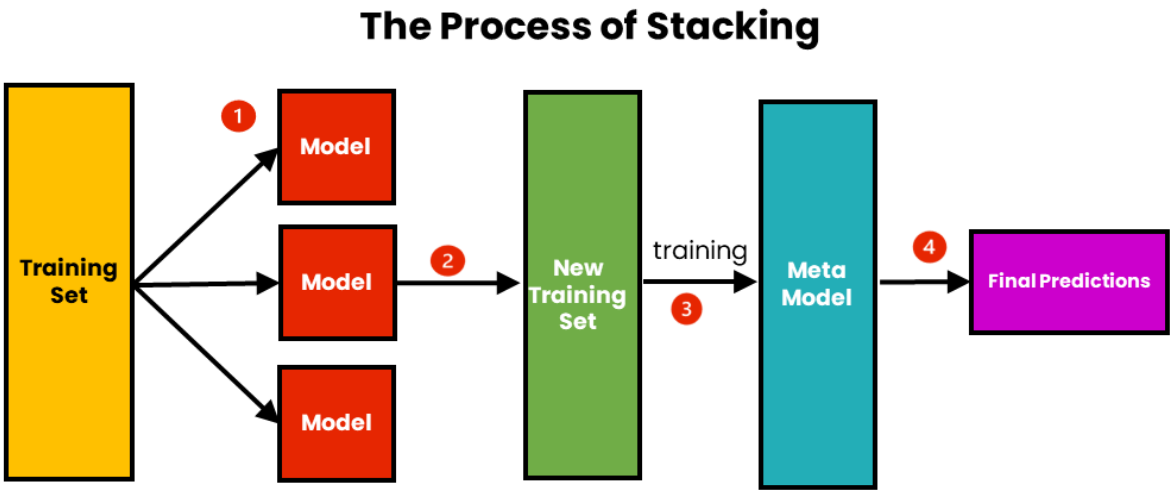
5. **Razvoj finalnog modela:** sljedeći korak je razvoj finalnog modela, koji uzima predviđanja od baznih modela i prema njima predviđa krajnju izlaznu vrijednost.

6. **Treniranje finalnog modela:** finalni model je treniran pomoću predviđanja baznih modela na podacima iz validacijskog seta. Predviđanja baznih modela služe kao značajke finalnom modelu.

7. **Predikcije na podacima za testiranje:** na kraju finalni model se koristi za predikcije na podacima za testiranje. Predikcije baznih modela prosljeđuju se finalnom modelu koji daje konačnu predikciju.

8. **Evaluacija modela**: zadnji korak je procjeniti točnost Stacking ansambla usporedbom predikcija ansambla sa stvarnim vrijednostima validacijskog seta podataka.

Cilj stacking ansambla je da kombinira prednosti raznih baznih modela prosljeđujući njihove izlazne vrijednosti u finalni estimator koji generira finalnu predikciju izlazne vrijednosti. Prednosti stackin ansamba su: poboljšana točnost predviđanja, raznolikost modela, fleksibilnost i interpretabilnost. Na slici 4.2 prikazan je izgled Stacking ansambl modela [6].



Slika 4.2. Prikaz izgleda Stacking ansambl modela.

5 EVALUACIJA MODELA I METODE TRENIRANJA

U ovom poglavlju bit će objašnjene evaluacijske metrike i metode koje su se koristile za treniranje modela stojnog učenja za estimaciju izlazne snage fotonaponske elektrane.

5.1 Evaluacijske metrike

Evaluacije metrike koje su korištene u radu su: koeficijent determinancije R^2, srednja apsolutna greška (MAE), kvadratni korijen srednje kvadratne pogreške (RMSE) i Kling-Gupta efficiency (KGE).

5.1.1 Koeficijent determinacije (R^2)

U statistici koeficijent determinancije je statistička mjera koja se koristi za procjenu koliko dobro regresijski model objašnjava varijabilnost zavisne varijable. Vrijednost koeficijenta determinancije može varirati između 0 i 1, gdje veće vrijednosti ukazuju na bolju prilagodbu modela podacima. To je statistička mjera koja se koristi u kontekstu statističkih modela čija je glavna svrha ili predviđanje budućih ishoda ili testiranje hipoteza, na temelju drugih povezanih informacija.

Postoje slučajevi kada R^2 može poprimiti negativne vrijednosti. To se može dogoditi kada predviđanja koja se uspoređuju s odgovarajućim ishodima nisu izvedena iz postupka prilagođavanja modela pomoću tih podataka. U slučajevima gdje nastaju negativne vrijednosti, srednja vrijednost podataka pruža bolje prilagođavanje ishodima od vrijednosti prilagođene funkcije, prema određenom kriteriju [1].

Koeficijent determinancije može intuitivno pružiti više informacija od: MAE, MAPE i RMSE-a u regresijskoj analizi. Jednadžbe koje se koriste za izračun koeficijenta determinancije su:

$$\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i, \tag{5.1}$$

gdje je:

- $\overline{y}$ srednja vrijednost promatranih podataka,
- n ukupni broj podataka,
- y_i pojedini podataka unutar skupa.

$$SS_{res} = \sum_{i} \left(y_i - f_i\right)^2, \tag{5.2}$$

gdje je:

- SS_{res} suma kvadrata ostataka,
- y_i stvarna vrijednost,
- f_i predviđena vrijednost.

$$SS_{tot} = \sum_i (y_i - \overline{y})^2 , \tag{5.3}$$

gdje je:

- SS_{tot} ukupna suma kvadrata,
- y_i pojedini podataka unutar skupa podataka,
- $\overline{y}$ srednja vrijednost promatranih podataka.

Nakon proračuna svih parametara pomoću navedenih jednadžbi može se izračunati koeficijent determinancije pomoću jednažbe:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}, \tag{5.4}$$

gdje je:

- SS_{res} suma kvadrata ostataka,
- SS_tot ukupna suma kvadrata.

5.1.2 Srednja apsolutna greška (MAE)

U statistici srednja apsolutna greška je mjera greške između uparenih vrijednosti koje izražavaju isti fenomen. Srednja apsolutna greška računa se kao zbroj apsolutnih pogrešaka podijeljene s veličinom uzorka:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}, \tag{5.5}$$

gdje je:

- n broj uzoraka,
- y_i prva uparena vrijednost,
- x_i druga uparena vrijednost.

Alternativne formule mogu sadržavati relativne frekvencije kao težinske faktore. Srednja apsolutna greška koristi mjernu jedinicu kao i podaci za koje se računa. Poznata je kao mjera točnosti koja je ovisna o skali i iz toga razloga ne može se koristit za usporedbu između predviđenih vrijednosti koje koriste različite skale. Srednja apsolutna greška često se koristi za mjerenje greške kod vremenske analize podataka [2].

5.1.3 Kvadratni korijen srednje kvadratne pogreške (RMSE)

Kvadratni korijen kvadratne greške (RMSE) je jedan od dva glavna pokazatelja točnosti regresijskog modela. Mjeri prosječnu razliku između predviđene vrijednosti i stvarne vrijednsti. Pruža estimaciju o tome kolika je točnost modela koji predviđa zadanu vrijednost.

Što je niža vrijednost RMSE-a to je model točniji. Savršenom modelu (model koji ima točnost predviđanja 100%) RMSE greška iznosti 0. RMSE ima prednost što grešku prikazuje i istoj mjernoj jedinici kao i podaci koji se predviđaju, zbog toga je laka za interpretaciju [3].

Jednadžba prema kojoj se računa kvadratni korijen srednje kvadratne greške je:

$$RMSE = \sqrt{\frac{SSE_w}{W}} = \sqrt{\frac{1}{W}\sum_{i=1}^{N} w_i u_i^2}, \tag{5.6}$$

gdje je:

- SSE_w ponderirana suma kvadata,
- W predstavlja ukupnu težinu populacije,
- w_i predstavlja težinu i-tog promatranja,
- u_i predstavlja grešku povezanu s i-tim promatranjem.

5.1.4 Kling-Gupta efficiency (KGE)

Kling-Gupta efficiency (KGE) je pokazatelj kvalitete prilagodbe koji se široko koristi u hidrološkim znanostima za usporedbu simulacija s opažanjima. Kreirali su je znanstvenici *Harald Kling* i *Hoshin Vijai Gupta*. Prvobitna ideja je bila da se s njom poboljšaju široko korištene metrike poput koeficijenta determinancije i Nash-Sutcliffe koeficijenta učinkovitosti modela [4]. Za izračun KGE koeficijenta potrebna su tri parametra:

- Pearsonov koeficijent korelacije, čija idealna vrijednost iznosi 1

- Parametar β koji je zapravo omjer srednjih simuliranih vrijednosti i srednjih promatranih vrijednosti. Idealna vrijednost β je 1
- Koeficijent varijance koji se može izračunati koristeći standardnu devijaciju ili koeficijent varijance od simuliranih i promatranih podataka.

Formula prema kojoj se računa Kling-Gupta efficiency glasi:

$$KGE = 1 - \sqrt{(r-1)^2 + (\alpha - 1)^2 + (\beta - 1)^2}, \tag{5.7}$$

gdje je:

- r Pearsonov koeficijent korelacije,
- α parametar koji predstavlja koeficijent varijance,
- β koeficijent koji predstavlja omjer između srednjih vrijednosti.

Formule prema kojima se računaju α i β su:

$$\alpha = \frac{\sigma_{sim}}{\sigma_{obs}}, \beta = \frac{\overline{Y_{sim}}}{\overline{Y_{obs}}}, \tag{5.8}$$

gdje je:

- σ_{sim} standardna devijacija simuliramih vrijednosti,
- σ_{obs} standardna devijacija promatranih vrijednosti,
- Y_{sim} srednja vrijednost simuliranih vrijednosti,
- Y_{obs} srednja vrijednost promatranih vrijednosti.

Ovisno o iznosu KGE parametra može se zaključiti koliko dobro model replicira stvarne uvjete:

- $0.7 < KGE < 1.00$ model ja izrazito dobar,
- $0.6 < KGE < 0.7$ model je dobar,
- $0.5 < KGE < 0.6$ model je zadovoljavajuć,
- $0.4 < KGE < 0.5$ model je prihvatljiv,
- $KGE \leq 0.4$ model je neprihvatljiv.

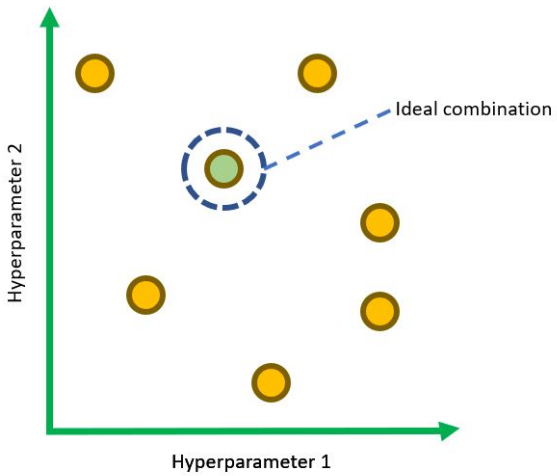
5.2 Metode treniranja

U ovome dijelu biti će objašnjene tehnike treniranja koje su se koristile u radu.

5.2.1 Nasumično pretraživanje hiperparametara

Optimizacija ili podešavanje hiperparametara je važna stavka u treniranju modela strojnog učenja. Hiperparametri modela se ne mogu odrediti iz date baze podataka tokom procesa učenja, ali oni su jako bitni za kontrolu samog procesa učenja. Ovi hiperparametri proizlaze iz matematičke formulacije modela strojnog učenja. Na primjer težinske vrijednosti naučene kroz treniranje modela linearne regresije su parametri, ali stopa učenja u gradijentnom spuštanju je hiperparametar. Točnost i brzina modela na nekoj bazi podataka uvelike ovisi o dobrom podešavanju hiperparametara. Postoje različite tehnike za optimizaciju hiperparametara kao što su: pretraživanje mreže, nasumično pretraživanje, Bayesian optimizacija, itd. U ovom radu za optimiziranje hiperparametara korištena je tehnika nasumičnog pretraživanja.

U pretrživanju mreže iscrpno se pretražuje svaka kombinacija vrijednosti hiperparametara koje su specificirane. Nasuprot pretraživanju mreže, nasumično pretraživanje ne isprobava sve dane vrijednosti hiperparametara nego uzorkuje fiksni broj postavki iz zadanih distribucija. Uzorkovanje bez ponavljanja provodi se ako su svi parametri predstavljeni kao lista, a uzorkovanje s ponavljanjem koristi se ako je barem jedan parametar dan kao distribucija. Glavna prednost ove tehnike je smanjeno vrijeme treniranja modela. Na slici 5.1 prikazan je grafički prikaz nasumičnog pretraživanja hiperparametara [5].



Slika 5.1. Prikaz nasumičnog pretraživanja hiperparametara.

5.2.2 Ansambl metoda

U statistici i strojnom učenju ansambl metode koriste više algoritama učenja kako bi dobili bolju točnost predviđanja od jednog bazičnog algoritma učenja. Za razliku od statističkog ansambla u statističkoj mehanici, koji je obično beskonačan, ansambl strojnog učenja sastoji se samo od konkretno konačnog skupa alternativnih modela, ali obično dopušta puno fleksibilniju strukturu među tim alternativama.

Ansambl učenje trenira dva ili više algoritama strojnog učenja za specifičan zadatak klasifikacije ili regresije. Algoritmi unutar anambl modela obično se nazivaju baznim modelima u literaturi. Bazni modeli mogu biti izrađeni koristeći jedan ili više različitih algoritama. Ideja je treniranje različitih baznih modela sa slabijim performansama za rješavanje istog problema. Kao rezultat toga bazni modeli imaju slabu prediktivnu sposobnost i među rezultatima svih izlaznih vrijednosti baznih modela pogreške pokazuju visoku varijancu. U osnovi ansambl model trenira više modela slabijih performansi kako bi se kombinirali u jači model s boljim performansama [6].

5.2.3 Unakrsna validacija

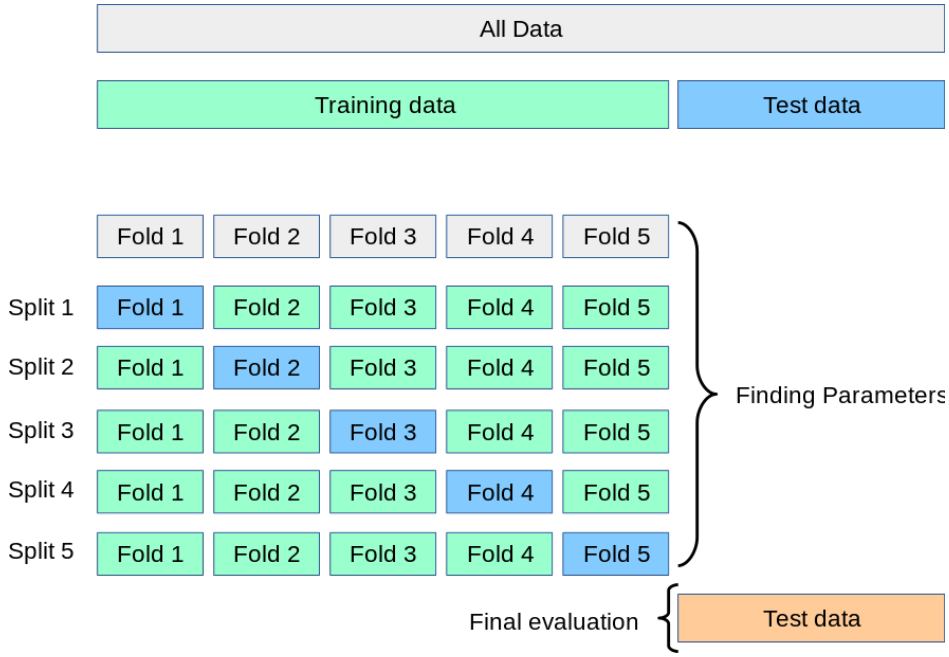
Unakrsna validacija je tehnika koja se koristi u strojnom učenju za evaluaciju performansi modela na nepoznatim podacima. Uključuje podjelu dostupnih podataka u više dijelova ili podskupina. Jedan od podskupova koristi se za validacijski set podataka, a ostali podskupovi koriste se za treniranje modela. Ovaj proces se ponavlja više puta i svaki put se drugi podskup koristi kao validacijski set podataka. Na kraju se računa srednja vrijednost pomoću rezultata od svakog validacijskog koraka. Tako se dobije robusnija procjena performanse modela strojnog učenja. Unakrsna validacija je važan korak kod strojnog učenja i pomaže osigurati da je odabrani model za implementaciju robustan i dobro generaliziran za nove podatke.

Glavna uloga Unakrsne validacije je spječavanje pretreniranja modela koji se pojavljuje kada model ima odlične performanse na treniranim podacima, a loše performanse na novim neviđenim podacima. Validacijom modela na više validacijskih setova podataka unakrsna validacija pruža realističniju estimaciju performansi modela strojnog učenja.

Postoje različiti tipovi unakrsne validacije: k-fold križna unakrsna validacija, leave-one-out unakrsna validacija, holdout unakrsna validacija i stratifield unakrsna validacija. Odabir tipa unakrsne validacije ovisi o veličini i tipovima podataka kao i o specifičnim zahtjevima problema modeliranja. U ovom radu za treniranje i validiranje modela koristila se k-fold križna unakrsna validacija s podacima podijeljenim u 5 podskupova.

Kod k-fold unakrsne validacije dataset se dijeli u k broj podskupova. Zatim se provodi treniranje

na svim podskupovima osim jednog $(k-1)$ podskupa koji se koristi za evaliaciju modela. Postupak se ponavlja k puta, pri čemu se svaki put drugi podskup koristi za validaciju podataka. Na slici 5.2 grafiči je prikazan princip rada k-fold unakrsne validacije [7].



Slika 5.2. Prikaz k-fold unakrsna validacija [21].

6 REZULTATI

U ovom poglavlju bit će prikazani dobiveni rezultati. Prvo će se opisati rezultati treniranja modela s nasumičnim pretraživanjem hiperparametara, zatim će slijediti rezultati treniranja modela s nasumičnim pretraživanjem hiperparametara i unakrsnom validacijom. Na kraju će se prikazati rezultati treniranja modela u ansamblu. Podaci koji su korišteni za treniranje i validaciju modela podijeljeni su u omjeru $70:30$ gdje se 70% podataka koristi za treniranje modela, a 30% za validaciju modela. Za treniranje i validaciju modela koristio se programski jezik Python s knjižnicama funkcija *sklearn* i *matplotlib.pyplot*.

6.1 Rezultati modela trenirani s nasumičnim pretraživanjem hiperparametara

Inicijalno, modeli su se trenirali samo pomoću nasumičnog pretraživanja hiperparametara. Ukupno je trenirano 7 modela. U tablicama od 6.1 do 6.7 prikazani su hiperparametri za pojedine modele pomoću kojih su se dobili najbolji rezultati.

Tablica 6.1. Tablica hiperparametara za ARDR model s nasumičnim pretraživanjem hiperparametara.

Number of iteration	109
Tolerance	$2.019 \cdot 10^{-27}$
Alpha 1	0.08810264845389955
Alpha 2	0.08372918844269946
Lambda 1	0.056429603178131345
Lambda 2	0.04716973040224769
Compute score	True
Treshold Lambda	2170

Tablica 6.2. Tablica hiperparametara za MLP model s nasumičnim pretraživanjem hiperparametara.

Hidden layer size	47, 92
Activation	relu
Solver	adam
Alpha	0.005514580017323831
Batch size	220
Learn rate	adaptive
Max iteration	1412
Tolerance	$9.27218 \cdot 10^{-5}$
Number of iteration	672

Tablica 6.3. Tablica hiperparametara za BayesianRidge regresor model s nasumičnim pretraživanjem hiperparametara.

Number of iteration	533
Tolerance	0.00056717
Alpha 1	0.04052867857405743
Alpha 2	0.0762618984549502
Lambda 1	0.0557072289744245
Lambda 2	0.03628641535998139
Lambda init	None
Computer score	True
Fit intercept	True

Tablica 6.4. Tablica hiperparametara za model Linearne regresije s nasumičnim pretraživanjem hiperparametara.

Fit intercept	True

Tablica 6.5. Tablica hiperparametara za Huber regresor model s nasumičnim pretraživanjem hiperparametara.

Epsilon	9.413190560643562
Max iteration	76350
Alpha	0.0007721814177468588
Warm start	False
Fit intercept	True
Tolerance	$5.2 \cdot 10^{-21}$

Tablica 6.6. Tablica hiperparametara za ElasticNET model s nasumičnim pretraživanjem hiperparametara.

Alpha	0.6032629557829275
L1 ratio	0.73545637
Fit intercept	False
Precompute	False
Max iteration	39257
Tolerance	$6.416 \cdot 10^{-6}$
Warm start	False
Random state	18
Selection	cyclic

Tablica 6.7. Tablica hiperparametara za Lasso model s nasumičnim pretraživanjem hiperparametara.

Alpha	4.279169495493141
Fit intercept	False
Max iteration	2302
Tolerance	$8.464 \cdot 10^{-11}$
Warm start	False
Random state	8
Selection	cyclic

U tablici 6.8 prikazani su rezultati svih treniranih modela validiranih pomoću: koeficijenta determinancije, srednje apsolutne pogreške, kvadratnog korjena srednje kvadratne pogreške i Kling-Gupta efficiency parametra. U tablici 6.9 prikazane su srednje vrijednosti i standardne devijacije za navedene metrike. U stupcima: R^2 train, MAE train, RMSE train prikazane su metrike validacije modela na podacima na kojima su modeli trenirani, a u stupcima: R^2 test, MAE test, RMSE test prikazane su metrike validacije modela na podacima na kojima modeli nisu trenirani nego služe za validaciju točnosti modela.

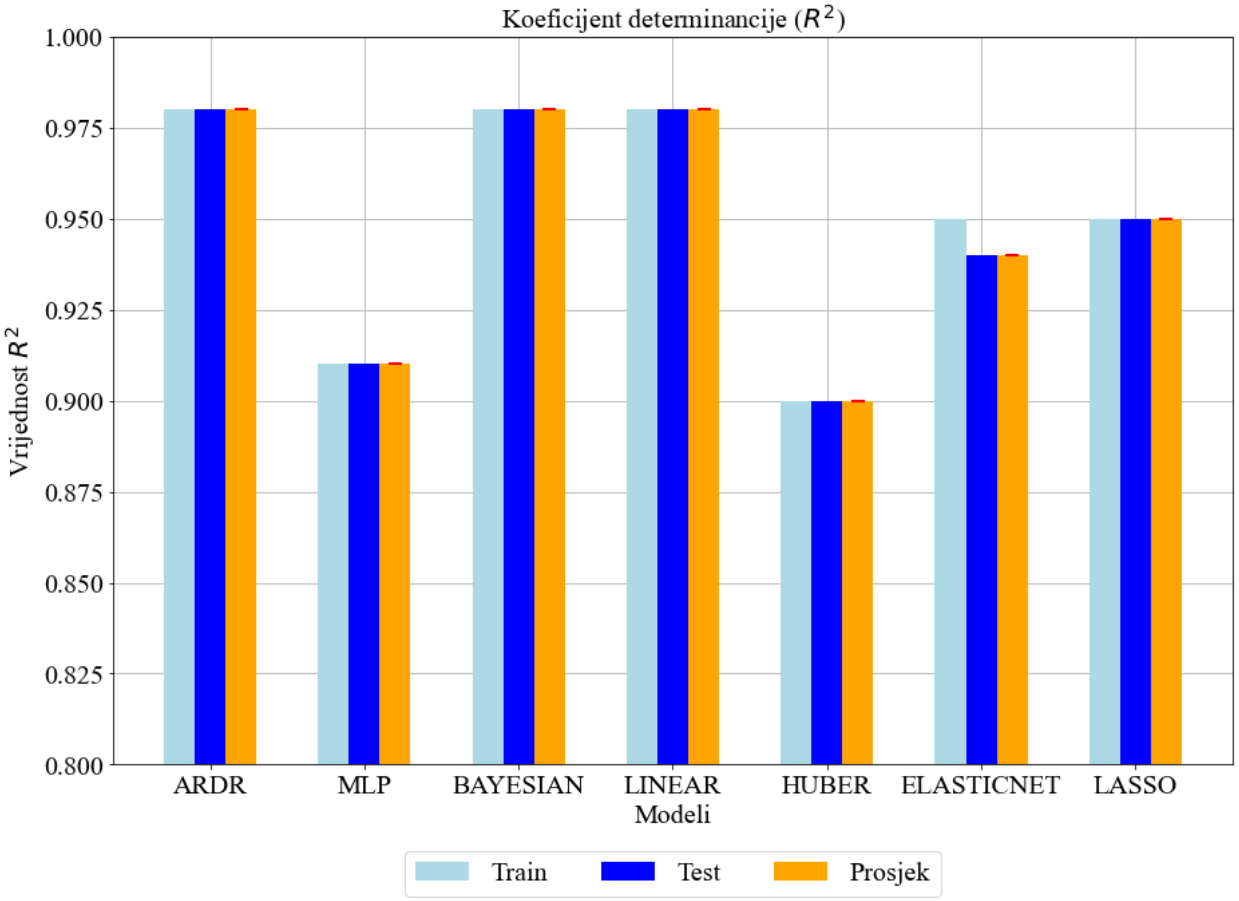
Tablica 6.8. Tablica rezultata treniranja modela s nasumičnim odabirom hiperparametara.

Model	R^2 train	R^2 test	MAE train	MAE test	RMSE train	RMSE test	KGE
ARDR	0.98	0.98	26.23	26.12	56.67	57.95	0.984
MLP	0.91	0.91	85.34	85.29	121.14	120.9	0.318
Bayesian	0.98	0.98	26.16	26.15	56.28	58.83	0.9839
Linearna regresija	0.98	0.98	26.13	26.36	56.67	57.94	0.984
Huber regresor	0.9	0.9	89.71	90.32	125.33	125.56	-0.53
ElasticNET	0.95	0.94	55.03	55.36	92.15	93.34	0.955
Lasso	0.95	0.95	51.27	51.04	87.82	85.22	0.958

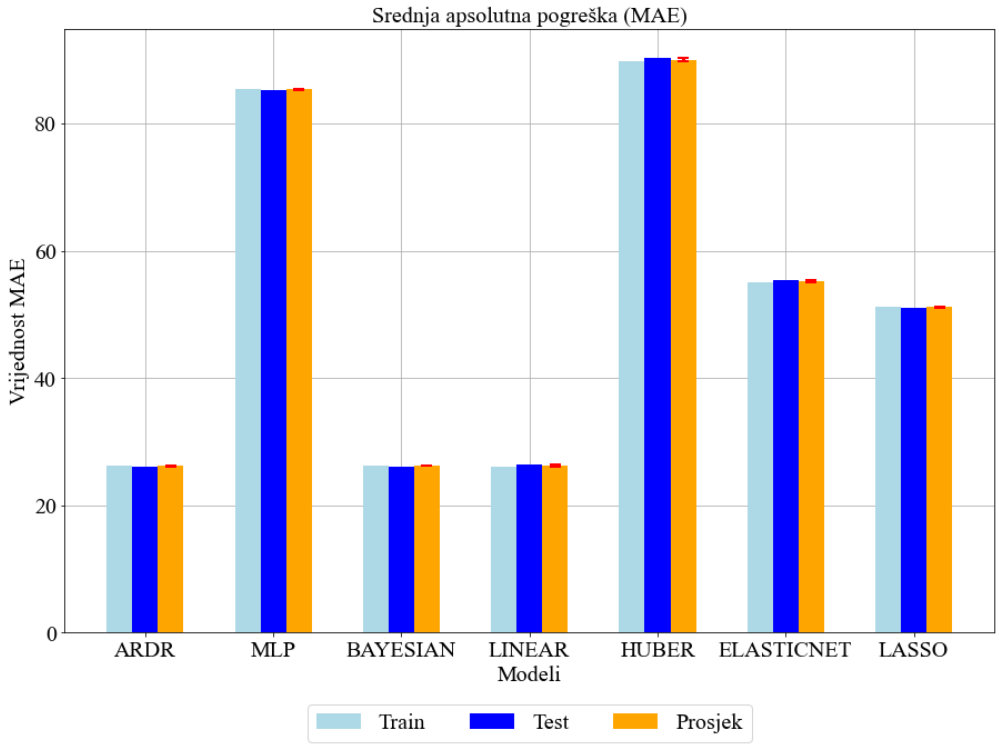
Tablica 6.9. Tablica rezultata treniranja modela s nasumičnim odabirom hiperparametara.

Model	$\overline{R^2}$	$\sigma(R^2)$	$\overline{MAE}$	$\sigma(MAE)$	$\overline{RMSE}$	$\sigma(RMSE)$
ARDR	0.98	0	26.18	0.05	57.31	0.64
MLP	0.91	0	85.31	0.03	121.02	0.12
Bayesian	0.98	0	26.16	0	57.76	1.28
Linearna regresija	0.98	0	26.25	0.12	57.31	0.64
Huber regresor	0.9	0	90.01	0.31	125.45	0.12
ElasticNET	0.94	0	55.2	0.16	92.75	0.59
Lasso	0.95	0	51.16	0.11	86.52	1.3

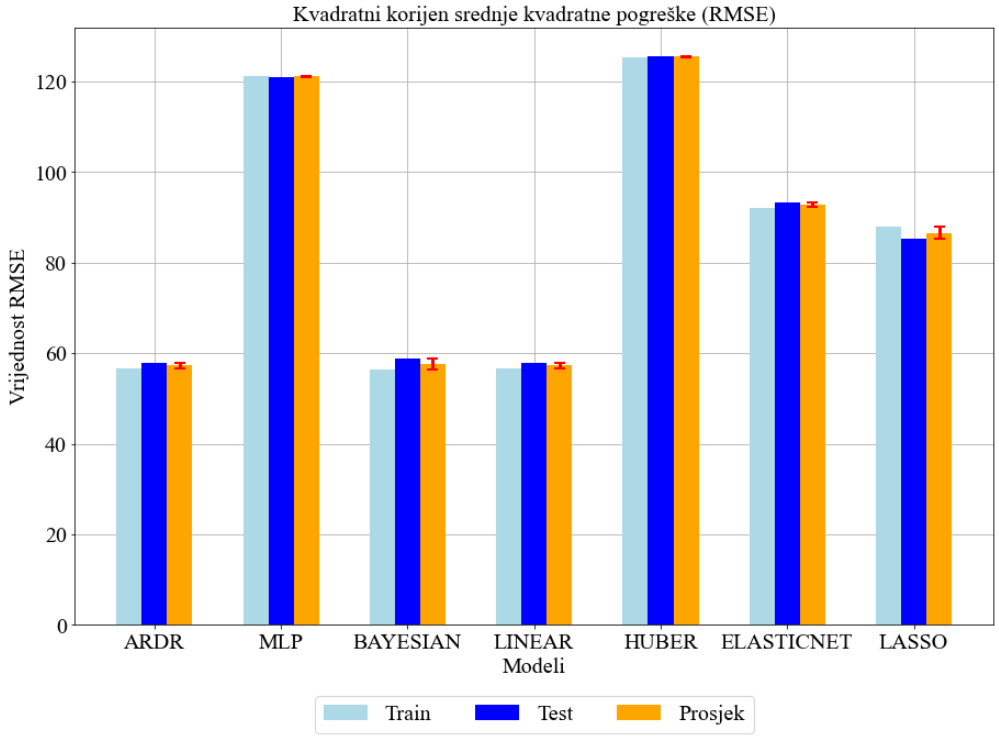
Na slikama 6.1, 6.2, 6.3 i 6.4 grafički su prikazani podaci iz prethodno navedenih tablica. Iz grafova i tablica može se vidjeti kako kod svih modela nema velike razlike u točnosti kod validacije na *train* i *test* podacima. Iz tog razloga standardna devijacija je približno 0 kod svih modela za train i test podatke. Svi modeli imaju koeficijent determinancije veći ili jednak 0.9, a prema grafu 6.1 može se vidjeti kako ARDR, Bayesian i model linearne regresije imaju najveću točnost dok Huber regresor i MLP imaju lošiju točnost. Graf na slici 6.4 potvrđuje da su MLP i Huber regressor imali najgoru točnost na datim podacima uz to što za Huber regresor KGE parametar je negativan. Grafovi srednje apsolutne pogreške i kvadratnog korjena srednje kvadratne pogreške na slikama 6.2 i 6.3 također potvrđuju prethodno navedena zapažanja.



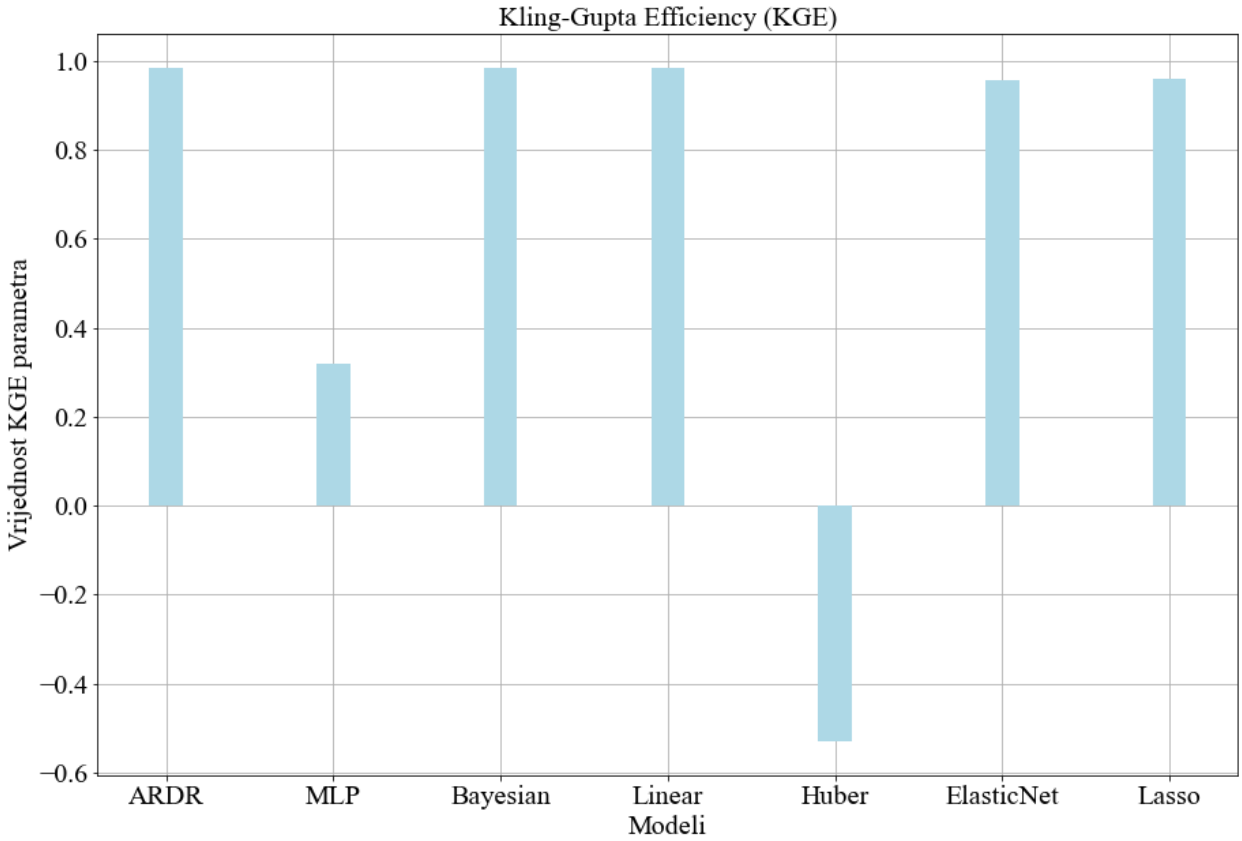
Slika 6.1. Grafički prikaz koeficijenta determinancije (R^2) za modele trenirane pomoću nasumičnog pretraživanja hiperparametara.



Slika 6.2. Grafički prikaz srednje apsolutne pogreške (MAE) za modele trenirane pomoću nasumičnog pretraživanja hiperparametara.



Slika 6.3. Grafički prikaz korjena srednje kvadratne pogreške (RMSE) za modele trenirane pomoću nasumičnog pretraživanja hiperparametara.



Slika 6.4. Grafički prikaz Kling-Gupta efficiency (KGE) parametra za modele trenirane pomoću nasumičnog pretraživanja hiperparametara.

6.2 Rezultati modela trenirani nasumičnim odabirom hiperparametara i unakrsnom validacijom

Nakon inicijalnog treniranja modela pomoću nasumičnog odabira hiperparametara u sljedećem koraku modeli su trenirani pomoću unakrsne validacije i nasumičnog pretraživanja hiperparametara. Korištena je 5-fold unakrsna validacija na svim modelima. U slučaju MLP modela korištena je i 10-fold unakrsna validacija kako bi se vidjelo hoće li se poboljšati točnost modela. U tablicama od 6.10 do 6.17 prikazani su hiperparametri modela s najboljom točnošću.

Tablica 6.10. Tablica hiperparametara za ARDR model s nasumičnim pretraživanjem hiperparametara i unakrsnom validacijom.

Number of iteration	214
Tolerance	$2.491 \cdot 10^{-27}$
Alpha 1	0.06837474921691818
Alpha 2	0.09776042328401258
Lambda 1	0.09561638401539464
Lambda 2	0.0717216373207169
Compute score	True
Treshold Lambda	5949

Tablica 6.11. Tablica hiperparametara za MLP model s nasumičnim pretraživanjem hiperparametara i 5-fold unakrsnom validacijom.

Hidden layer size	194, 163, 80
Activation	relu
Solver	lgbfgs
Alpha	0.004068865477850763
Batch size	248
Learn rate	constant
Max iteration	870
Tolerance	$8.7442735 \cdot 10^{-5}$
Number of iteration	481

Tablica 6.12. Tablica hiperparametara za MLP model s nasumičnim pretraživanjem hiperparametara i 10*-fold unakrsnom validacijom.*

Hidden layer size	33, 142
Activation	indentity
Solver	adam
Alpha	0.008508071983703827
Batch size	266
Learn rate	constant
Max iteration	1059
Tolerance	$1.9914620 \cdot 10^{-6}$
Number of iteration	793

Tablica 6.13. Tablica hiperparametara za BayesianRidge regresor model s nasumičnim pretraživanjem hiperparametara i unakrsnom validacijom.

Number of iteration	670
Tolerance	0.0004508938008
Alpha 1	0.012405879542588
Alpha 2	0.0262374491674
Lambda 1	0.03864148447738
Lambda 2	0.086342795626
Lambda init	6.98466573385
Computer score	True
Fit intercept	True

Tablica 6.14. Tablica hiperparametara za model Linearne regresije s nasumičnim pretraživanjem hiperparametara i unakrsnom validacijom.

Fit intercept	True

Tablica 6.15. Tablica hiperparametara za Huber regresor model s nasumičnim pretraživanjem hiperparametara i unakrsnom validacijom.

Epsilon	9.50865357703521
Max iteration	67397
Alpha	0.000792710032
Warm start	False
Fit intercept	True
Tolerance	$4.298 \cdot 10^{-21}$

Tablica 6.16. Tablica hiperparametara za ElasticNET model s nasumičnim pretraživanjem hiperparametara i unakrsnom validacijom.

Alpha	0.084512672
L1 ratio	0.999774892862
Fit intercept	True
Precompute	False
Max iteration	20637
Tolerance	$3.1939 \cdot 10^{-7}$
Warm start	False
Random state	41
Selection	random

Tablica 6.17. Tablica hiperparametara za Lasso model s nasumičnim pretraživanjem hiperparametara i unakrsnom validacijom.

Alpha	1.0548237821696
Fit intercept	True
Max iteration	2026
Tolerance	$2.173759 \cdot 10^{-11}$
Warm start	False
Random state	None
Selection	cyclic

U tablicama 6.18, 6.19, 6.20 i 6.21 prikazani su rezultati svih modela validranih pomoću: koeficijenta determinancije, srednje apsolutne pogreške, kvadratnog korjena srednje kvadratne pogreške i Kling-Gupta efficiency parametra. Prikazane su srednje vrijednosti i standardne devijacije navedenih metrika za train podatke, test podatke i sveukupni dataset. Na slikama 6.5, 6.6, 6.7 i 6.8

grafički su prikazani parametri navedeni u prethodnim tablicama. Iz grafičkih prikaza može se vidjeti kako je Huber regresor model s najmanjom točnošću i iz tog razloga se neće koristiti dalje za razvoj ansambl modela. MLP model s 5-fold unakrsnom validacijom ima također lošu točnost predviđanja izlazne snage solarne elektrane, ali treniranjem modela s 10-fold unakrsnom validacijom pokazuje povećanje točnosti i time je MLP još uvijek prikladan za daljnje korištenje za razvoj ansambl modela. Kod MLP (CV10) modela potrebno je naznačiti vrijednost standardne deviacije kod rezultata foldova na train i test podacima. Ostalih 5 modela imaju visoku točnost predviđanja s unakrsnom validacijom i iz toga razloga su isto prikladni za razvoj ansambl modela. Najbolji model s najvećom točnošću je ARDR model.

Tablica 6.18. Tablica koeficijenta determinancije za modele trenirane pomoću unakrsne validacije i nasumičnim odabirom hiperparametara.

Model	$\overline{R^2}$ train	$\sigma(R^2)$ train	$\overline{R^2}$ test	$\sigma(R^2)$ test	$\overline{R^2}$ all	$\sigma(R^2)$ all
ARDR	0.98	0	0.98	0.001	0.98	$3.74 \cdot 10^{-6}$
MLP (CV5)	0	0	0	0	0	$7.98 \cdot 10^{-5}$
MLP (CV10)	0.76	0.14	0.76	0.14	0.76	0.14
Bayesian	0.978	0	0.979	0	0.979	$5.63 \cdot 10^{-6}$
Linearna regresija	0.98	0	0.98	0	0.98	$1.36 \cdot 10^{-5}$
Huber regresor	-0.412	0	-0.412	0.01	-0.412	$6.18 \cdot 10^{-5}$
ElasticNET	0.978	0	0.978	0	0.978	$6.11 \cdot 10^{-6}$
Lasso	0.977	0	0.977	0	0.977	$1.36 \cdot 10^{-5}$

Tablica 6.19. Tablica srednja apsolutne pogreške za modele trenirane pomoću unakrsne validacije i nasumičnim odabirom hiperparametara.

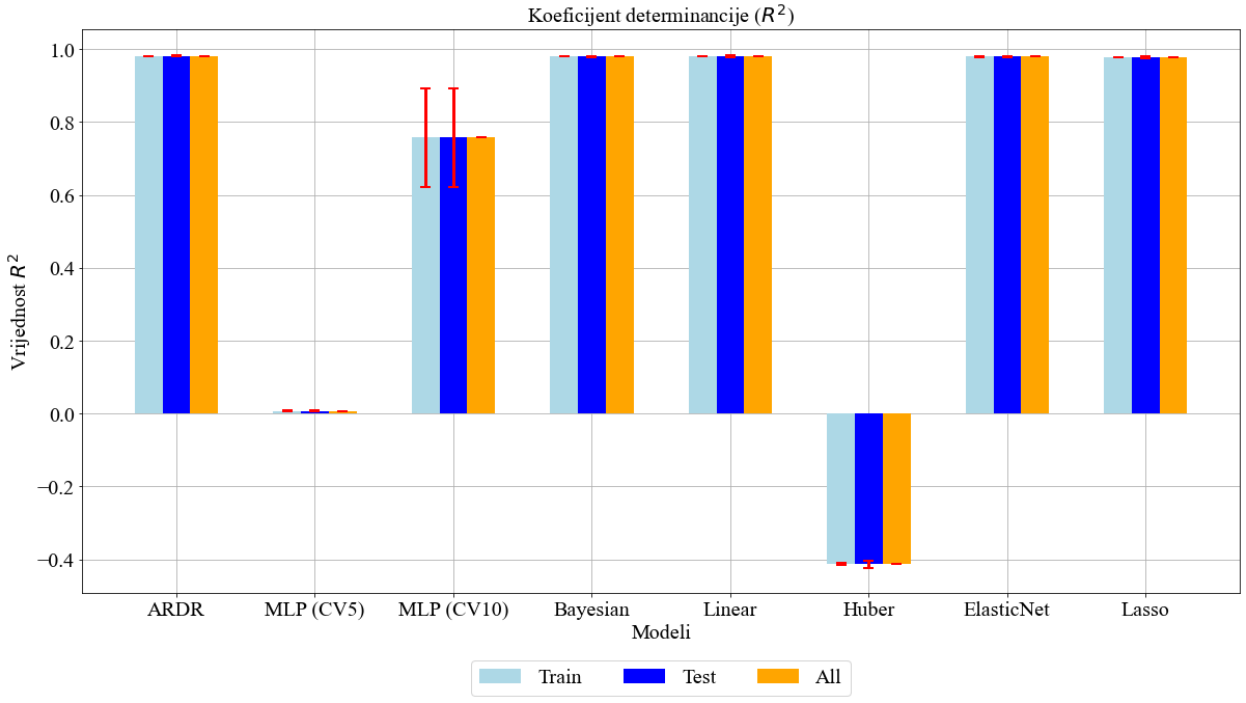
Model	$\overline{MAE}$ train	$\sigma(MAE)$ train	$\overline{MAE}$ test	$\sigma(MAE)$ test	$\overline{MAE}$ all	$\sigma(MAE)$ all
ARDR	25.82	0.18	25.82	0.61	25.82	0.001
MLP (CV5)	341.55	0.88	341.56	1.88	341.56	0.006
MLP (CV10)	150.96	38.1	150.8	37.5	150.9	0.01
Bayesian	26.04	0.11	26.05	0.29	26.04	0.004
Linearna regresija	25.73	0.29	25.73	0.56	25.73	0.005
Huber regresor	303.51	0.88	303.52	3.5	303.51	0.007
ElasticNET	26.28	0.2	26.29	0.47	26.29	0.002
Lasso	28.95	0.203	28.97	0.66	28.96	0.006

Tablica 6.20. Tablica kvadratnog korjena srednje kvadratne pogreške za modele trenirane pomoću unakrsne validacije i nasumičnim odabirom hiperparametara.

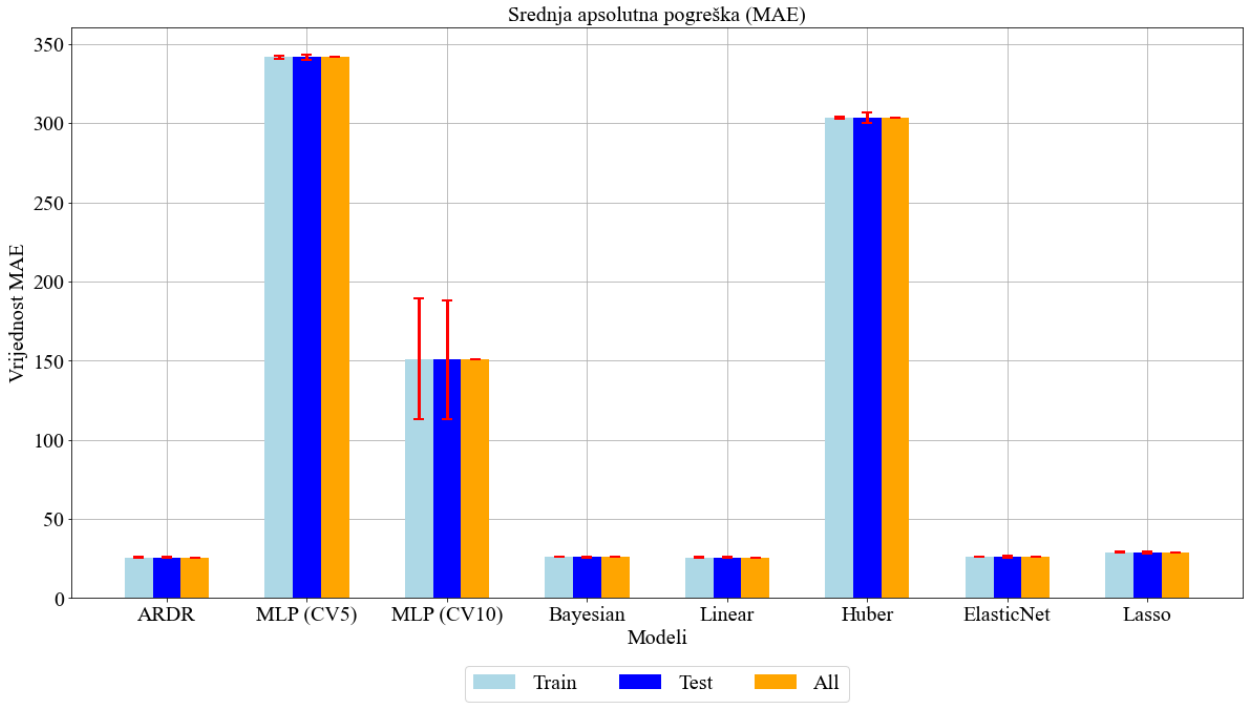
Model	$\overline{RMSE}$ train	$\sigma(RMSE)$ train	$\overline{RMSE}$ test	$\sigma(RMSE)$ test	$\overline{RMSE}$ all	$\sigma(RMSE)$ all
ARDR	54.97	0.49	54.94	1.96	54.95	0.01
MLP (CV5)	393.44	0.81	393.45	2.44	393.44	0.002
MLP (CV10)	188.56	47.65	188.3	46.8	188.43	0.13
Bayesian	56.41	0.55	56.39	2.17	56.4	0.009
Linearna regresija	55.19	1.39	54.99	5.36	55.09	0.101
Huber regresor	467.54	1.12	467.52	4.057	467.53	0.005
ElasticNET	57.21	0.72	57.16	2.87	57.19	0.03
Lasso	59.63	1.09	59.52	4.16	59.58	0.06

Tablica 6.21. Tablica KGE parametra za modele trenirane pomoću unakrsne validacije i nasumičnim odabirom hiperparametara.

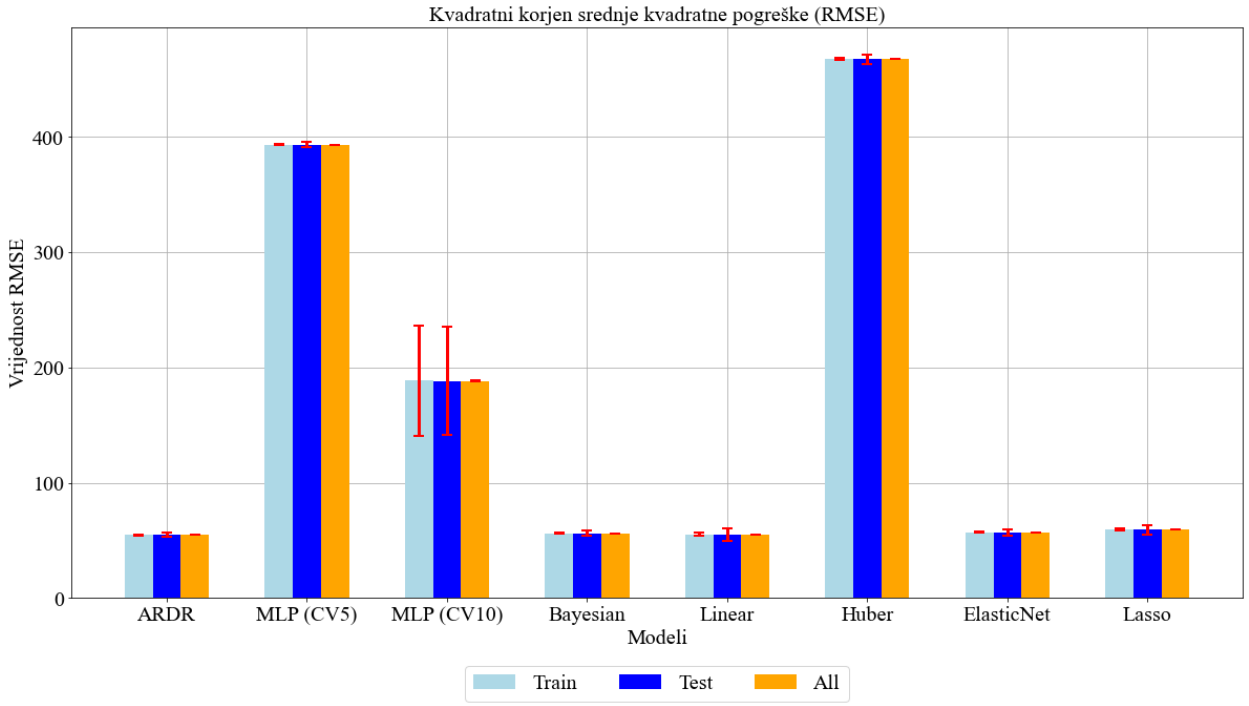
ARDR	MLP (CV5)	MLP (CV10)	Bayesian	Linearna regresija	Huber regresor	ElasticNET	Lasso
0.985	-0.295	0.63	0.99	0.985	-0.525	0.985	0.979



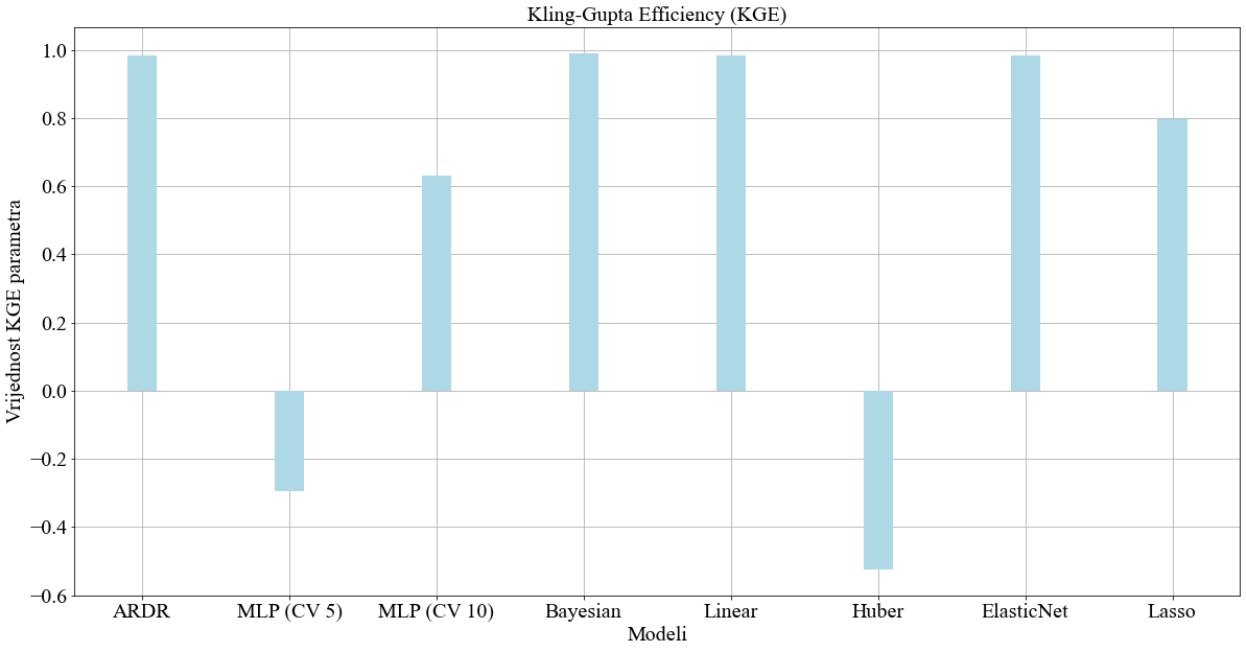
Slika 6.5. Grafički prikaz koeficijenta determinancije (R^2) za modele trenirane pomoću nasumičnog pretraživanja hiperparametara i unakrsne validacije.



Slika 6.6. Grafički prikaz srednje apsolutne pogreške (MAE) za modele trenirane pomoću nasumičnog pretraživanja hiperparametara i unakrsne validacije.



Slika 6.7. Grafički prikaz kvadratnog korjena srednje kvadratne pogreške (RMSE) za modele trenirane pomoću nasumičnog pretraživanja hiperparametara i unakrsne validacije.



Slika 6.8. Grafički prikaz Kling-Gupta efficiency (KGE) parametra za modele trenirane pomoću nasumičnog pretraživanja hiperparametara i unakrsne validacije.

6.3 Rezultati stacking ansambla

U ovom dijelu najbolji prethodno trenirani modeli koristit će se unutar stacking ansambla kako bi se dobila još točnija i robusnija estimacija izlazne snage fotonaponske elektrane. Kako bi se dobio natočniji ansambl model trenirati će se 5 stacking modela svaki s drugim finalnim estimatorom. U prethodnim analizama Huber regresor je imao malu točnost i zato se ne koristi unutar stacking ansambla. Finalni estimatori koji će se koristiti su: AdaBoost, Banging, Extra Trees, Random forest i Hist gradient boosting estimator. Modeli su trenirani pomoću 5-fold unakrsne validacije i nasumičnog odabira hiperparametara za svaki model koji se nalazi unutar stacking modela. U tablicama 6.22, 6.23, 6.24, 6.25, 6.26 i 6.27 prikazani su hiperparametri za modele koji su se koristili u stacking ansamblu.

Tablica 6.22. Tablica hiperparametara za ARDR model unutar ansambla.

Model	AdaBoost	Banging	Extra trees	Random forest	Hist gradient boosting
Number of iteration	257	556	480	941	561
Tolerance	$4.39 \cdot 10^{-27}$	$6.33 \cdot 10^{-27}$	$6.72 \cdot 10^{-27}$	$9.1 \cdot 10^{-27}$	$1.25 \cdot 10^{-27}$
Alpha 1	0.07578156667604662	0.02774454755498524	0.08225347081725076	0.04713973529025759	0.032791731673231836
Alpha 2	0.02863118646744016	0.05688836569237448	0.00372704140718783	0.08899892163332393	0.00816904411151411
Lambda 1	0.08895233056584367	0.09328368260246679	0.058653843302653164	0.02272384651604063	0.024730974747874404
Lambda 2	0.09513300171766942	0.0723634930934649	0.09331558595609028	0.09025683786185128	0.02800197012020612
Compute score	False	True	True	False	True
Treshold Lambda	81391	65714	88665	31007	66283

Tablica 6.23. Tablica hiperparametara za MLP model unutar ansambla.

Model	AdaBoost	Banging	Extra trees	Random forest	Hist gradient boosting
Hidden layer size	152, 164, 195, 146, 45	135, 165	30, 167, 153, 41, 155	75, 169, 90, 165, 75	121, 99, 187, 196, 100
Activation	tanh	relu	indentity	tanh	relu
Solver	adam	adam	adam	adam	adam
Alpha	0.0035951588714092085	0.005950262637175208	0.005081475537408525	0.008978211614367673	0.00022815032721041638
Batch size	233	288	280	251	206
Learn rate	adaptive	invscaling	constant	invscaling	invscaling
Max iteration	781	219	750	1432	1711
Tolerance	$3.87 \cdot 10^{-5}$	$4.19 \cdot 10^{-5}$	$4.18 \cdot 10^{-5}$	$9.64 \cdot 10^{-5}$	$7.77 \cdot 10^{-5}$
Number of iteration	268	105	639	777	263

Tablica 6.24. Tablica hiperparametara za Bayesian ridge model unutar ansambla.

Model	AdaBoost	Banging	Extra trees	Random forest	Hist gradient boosting
Number of iteration	566	930	832	729	903
Tolerance	0.0009097445022410391	0.000655361351582836	0.000161740149018795	0.0005696169714154588	0.000929273880238025
Alpha 1	0.042895592371643074	0.031118742111376316	0.055039970055939746	0.05832677035642352	0.02613782918500053
Alpha 2	0.09679070758669095	0.08788766745056521	0.08985630807451493	0.038756018752699906	0.07969035507687704
Lambda 1	0.003518123177745364	0.03270379021378298	0.04509762910098107	0.018654762780960375	0.06666806334308589
Lambda 2	0.07177324694631368	0.09536966654578581	0.009158204387538543	0.0985803842191315	0.01462302270722804
Lambda init	None	2.584396354894184	7.556984303346983	None	6.338484228993025
Computer score	False	False	True	True	True
Fit intercept	True	True	False	False	False

Tablica 6.25. Tablica hiperparametara za model linearne regresije unutar ansambla.

Model	AdaBoost	Banging	Extra trees	Random forest	Hist gradient boosting
Fit intercept	True	False	True	False	True

Tablica 6.26. Tablica hiperparametara za ElasticNET model unutar ansambla.

Model	AdaBoost	Banging	Extra trees	Random forest	Hist gradient boosting
Alpha	0.7337093378875563	0.9057450973518797	0.5381606530695325	0.12120272561602552	0.8856868627219607
L1 ratio	0.6251096829711442	0.03721512128022331	0.17065928471337655	0.8868597369575415	0.27338357779354283
Fit intercept	True	True	False	False	True
Precompute	False	False	False	False	False
Max iteration	97188	46842	42993	16025	83576
Tolerance	$3.62 \cdot 10^{-6}$	$3.02 \cdot 10^{-6}$	$9.9 \cdot 10^{-7}$	$3.23 \cdot 10^{-6}$	$4.84 \cdot 10^{-6}$
Warm start	False	False	False	False	False
Random state	32	45	28	28	7
Selection	cyclic	cyclic	random	cyclic	random

Tablica 6.27. Tablica hiperparametara za Lasso model unutar ansambla.

Model	AdaBoost	Banging	Extra trees	Random forest	Hist gradient boosting
Alpha	4.628226776206003	5.2941742188907845	2.287723213817942	2.9967067708185615	2.624791973485909
Fit intercept	True	True	True	False	False
Max iteration	5889	1472	5258	1963	9263
Tolerance	$7.82 \cdot 10^{-11}$	$1.87 \cdot 10^{-11}$	$1.31 \cdot 10^{-11}$	$2.53 \cdot 10^{-11}$	$3.86 \cdot 10^{-11}$
Warm start	False	False	False	False	False
Random state	44	34	None	None	5
Selection	cyclic	random	random	random	random

U tablicama 6.28, 6.29, 6.30 i 6.31 prikazani su rezultati svih ansambl modela validiranih pomoću: koeficijenta determinancije, srednje apsolutne pogreške, kvadratnog korjena srednje kvadratne pogreške i Kling-Gupta efficiency parametra. Prikazane su srednje vrijednosti i standardne devijacije prethodno navedenih metrika za train podatke, test podatke i za sveukupni skup podataka. Na slikama 6.9, 6.10, 6.11 i 6.12 grafički su prikazani parametri iz prethodnih tablica. Iz prikazanih rezultata može se vidjeti kako jedino manju točnost ima stacking ansambl s AdaBoost finalnim estimatorom kojem koeficijent determinancije iznosi približno 0.9 dok drugim modelima koeficijent determinancije iznosi približno 0.98. Razlika u koeficijentima determinancije između train i test podataka je mala dok za MAE i RMSE pogreške postoji veća razlika za train i test podatke. Najbolji model s najvećim koeficijentom determinancije i najmanjim MAE i RMSE pogreškama je stacking ansambl s random forest finalnim estimatorom. Rezultati Kling-Gupta parametra također potvrđuju kako je stacking ansambl s AdaBoost finalnim estimatorom najlošiji model.

Tablica 6.28. Tablica koeficijenta determinancije za modele Stacking ansambla.

Model	$\overline{R^2}$ train	$\sigma(R^2)$ train	$\overline{R^2}$ test	$\sigma(R^2)$ test	$\overline{R^2}$ all	$\sigma(R^2)$ all
AdaBoost	0.9439	0.019	0.9428	0.021	0.9433	0
Banging	0.99	$3.94 \cdot 10^{-4}$	0.9841	$1.94 \cdot 10^{-3}$	0.9872	$3.09 \cdot 10^{-3}$
Extra Trees	0.9901	$9.62 \cdot 10^{-4}$	0.984	$9.98 \cdot 10^{-4}$	0.987	$3.03 \cdot 10^{-3}$
Random forrest	0.9917	$1.58 \cdot 10^{-3}$	0.9824	$2 \cdot 10^{-3}$	0.987	$4.66 \cdot 10^{-3}$
Hist gradient boosting	0.9848	$1.76 \cdot 10^{-4}$	0.9836	$1.49 \cdot 10^{-3}$	0.9843	$6.09 \cdot 10^{-4}$

Tablica 6.29. Tablica srednje apsolutne pogreške za modele Stacking ansambla.

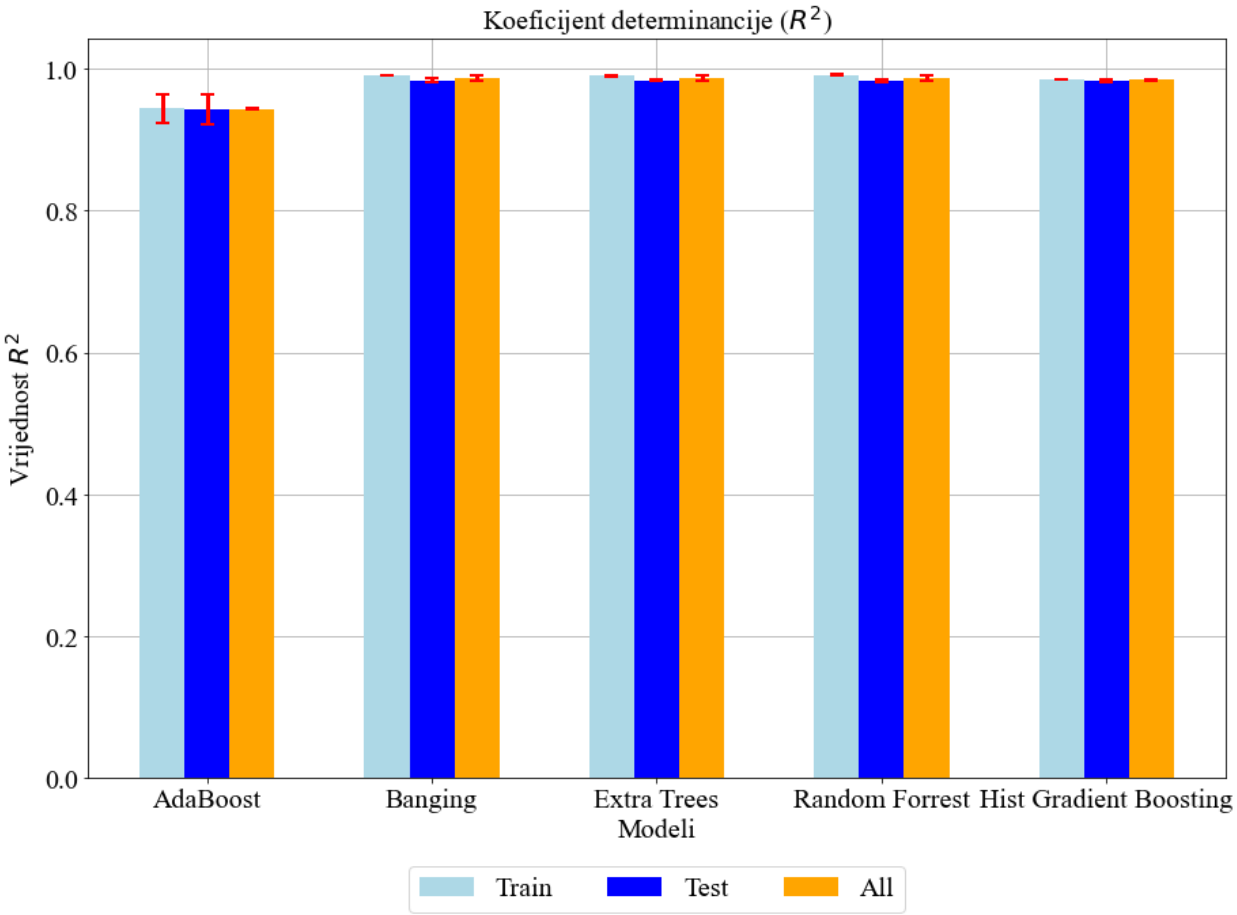
Model	$\overline{MAE}$ train	$\sigma(MAE)$ train	$\overline{MAE}$ test	$\sigma(MAE)$ test	$\overline{MAE}$ all	$\sigma(MAE)$ all
AdaBoost	44.66	7.84	44.99	8.69	44.82	0.16
Banging	13.43	0.23	18.08	0.44	15.76	2.33
Extra Trees	13.59	0.54	18.12	0.22	15.86	2.27
Random forrest	12.2	1.05	18.79	0.56	15.49	3.29
Hist gradient boosting	18.74	0.107	19.36	0.54	19.05	0.31

Tablica 6.30. Tablica kvadratnog korjena srednje kvadratne pogreške za modele Stacking ansambla.

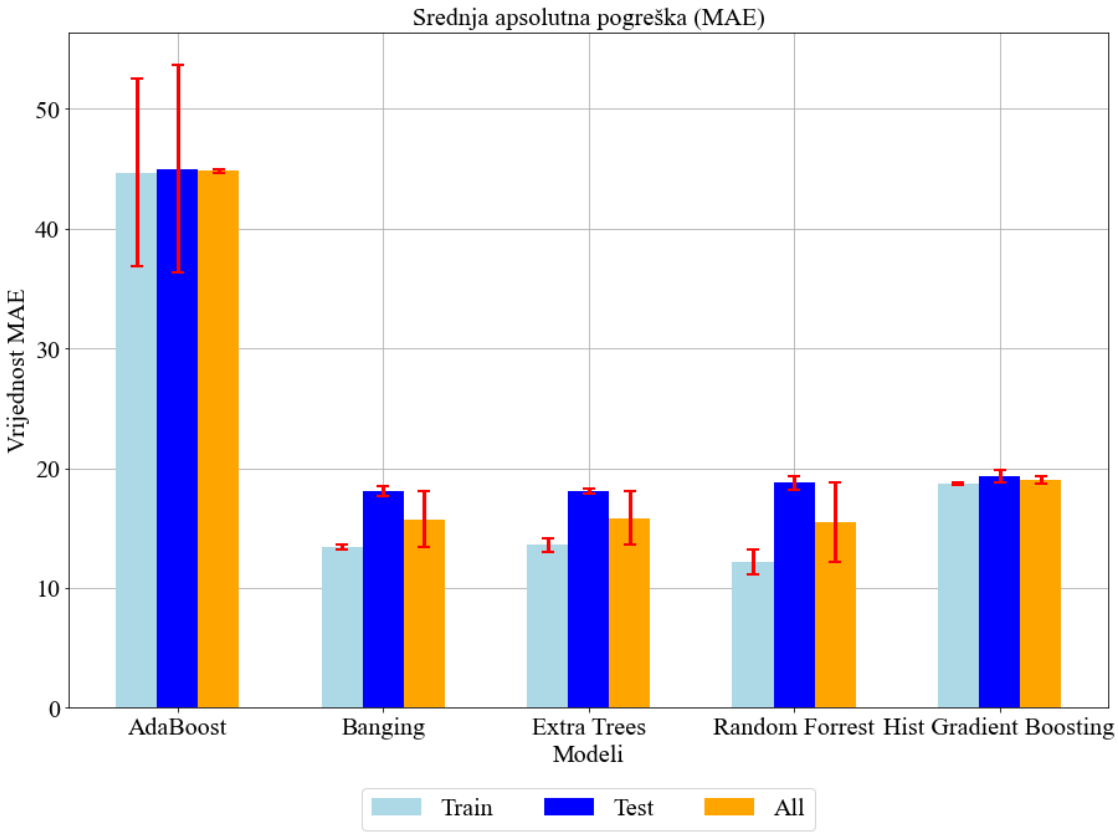
Model	$\overline{RMSE}$ train	$\sigma(RMSE)$ train	$\overline{RMSE}$ test	$\sigma(RMSE)$ test	$\overline{RMSE}$ all	$\sigma(RMSE)$ all
AdaBoost	91.92	16.31	92.77	18.2	92.34	0.43
Banging	38.85	0.77	49.65	3.02	44.25	5.4
Extra Trees	39.22	1.95	49.87	1.52	44.54	5.33
Random forrest	35.86	3.56	52.43	2.95	44.14	8.29
Hist gradient boosting	48.42	0.303	50.27	2.34	49.35	0.93

Tablica 6.31. Tablica KGE parametra za modele Stacking ansambla.

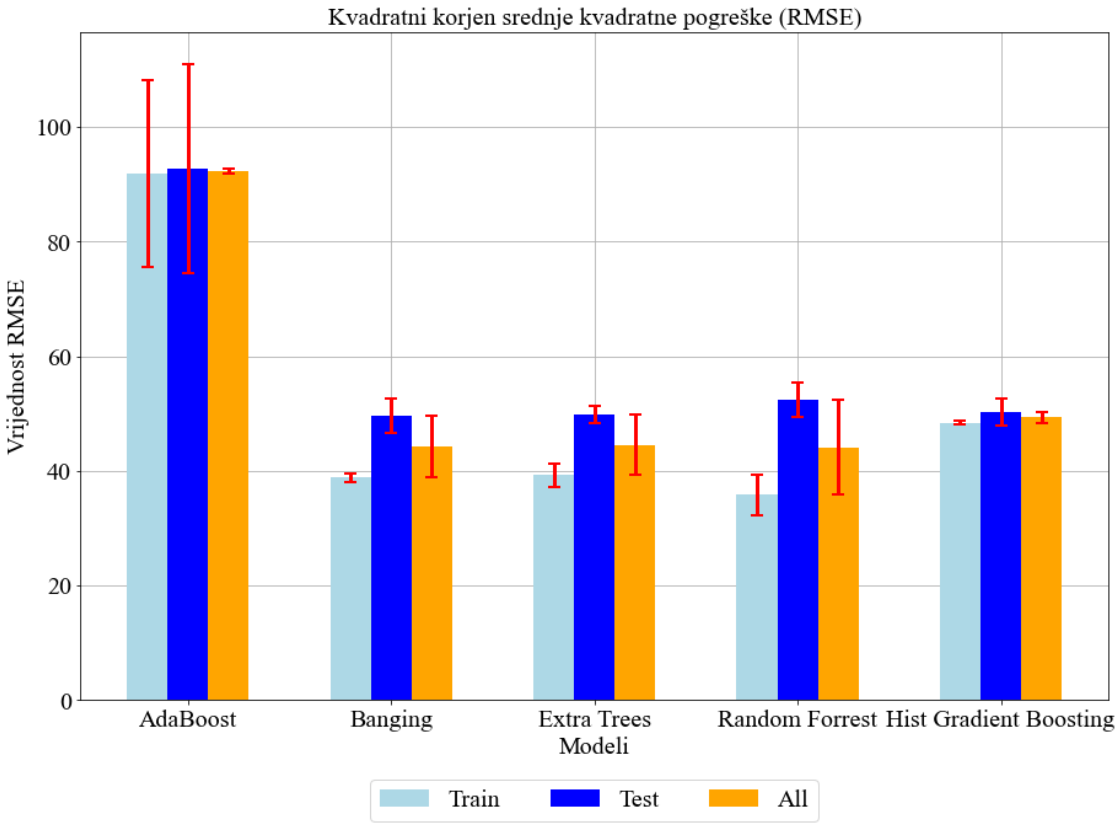
AdaBoost	Banging	Extra trees	Random forest	Hist gradient boosting
0.9	0.99	0.99	0.99	0.989



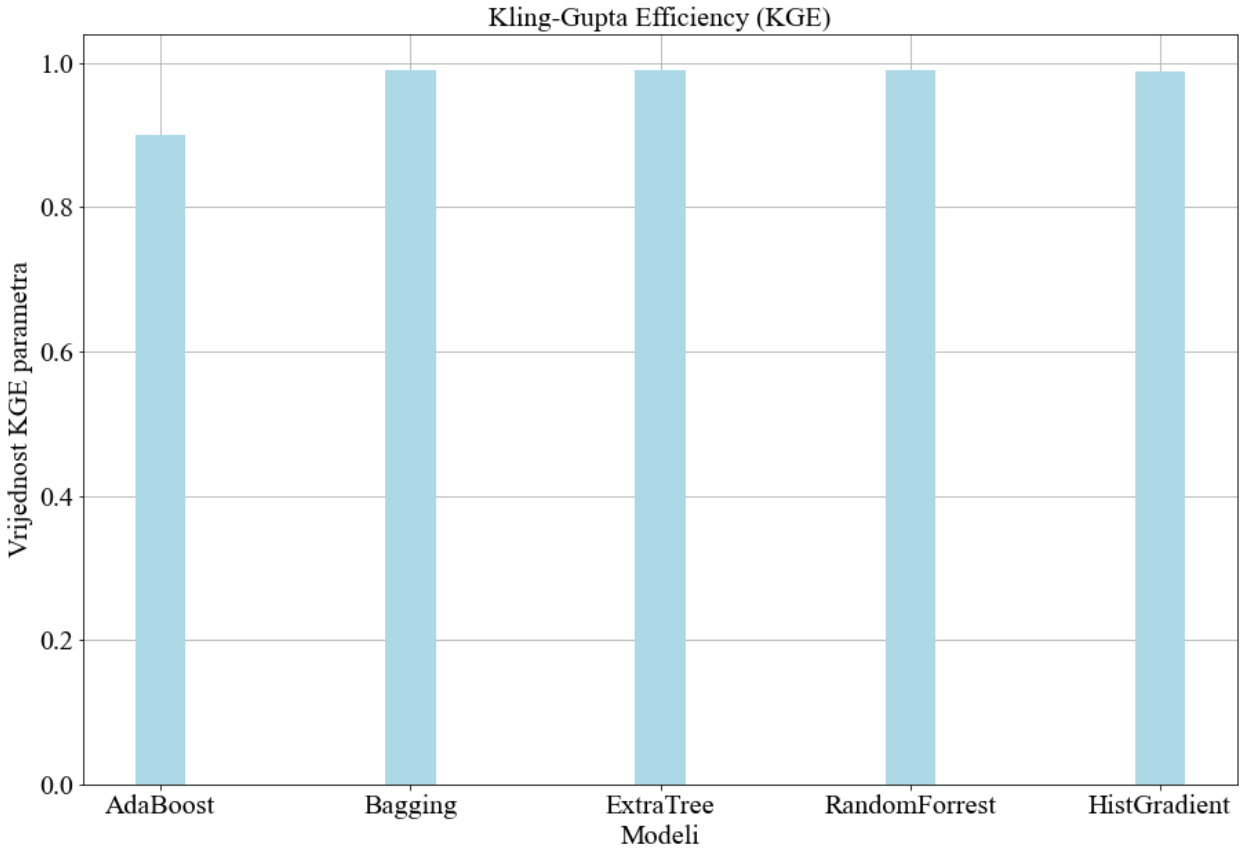
Slika 6.9. Prikaz koeficijenta determinancije za ansambl modele



Slika 6.10. Prikaz srednje apsolutne pogreške za ansambl modele



Slika 6.11. Prikaz kvadratnog korjena srednje kvadratne pogreške za ansambl modele



Slika 6.12. Prikaz Kling-Gupta efficiency parametra za ansambl modele

6.4 Rezultati krajnjeg ansambl modela

Krajnji anambl modeli napravljeni su kombinacijom prethodno treniranih stacking modela. Trenirana su 2 krajnja stacking ansambl modela. Prvi stacking ansambl model je napravljen od svih 5 prethodno treniranih stacking modela, dok u drugom stacking ansambl modelu je izbačen stacking model s AdaBoost finalnim estimatorom. Krajnji stacking ansambl model radi na principu gdje se ulazni parametri šalju na sve stacking modele unutar finalnog modela. Izlazna vrijednost krajnjeg ansambl modela dobije se tako da se izračuna aritmetička sredina od izlaznih vrijednosti stacking modela. U tablicama 6.32, 6.33 i 6.34 prikazani su rezultati kranjih stacking anasambl modela pomoću parametara: koeficijenta determinancije, srednje apsolutne pogreške i kvadratnog korjena srednje kvadratne pogreške. Prikazane su srednje vrijednosti i standardne devijacije prethodno navedenih metrika za train podatke, test podatke i za sveukupni skup podataka.

Tablica 6.32. Tablica koeficijenta determinancije za modele Stacking ansambla.

Model	$\overline{R^2}$ train	$\sigma(R^2)$ train	$\overline{R^2}$ test	$\sigma(R^2)$ test	$\overline{R^2}$ all	$\sigma(R^2)$ all
Finalni 1	0.98	0.018	0.98	0.02	0.98	0.017
Finalni 2	0.99	0	0.98	0	0.99	0

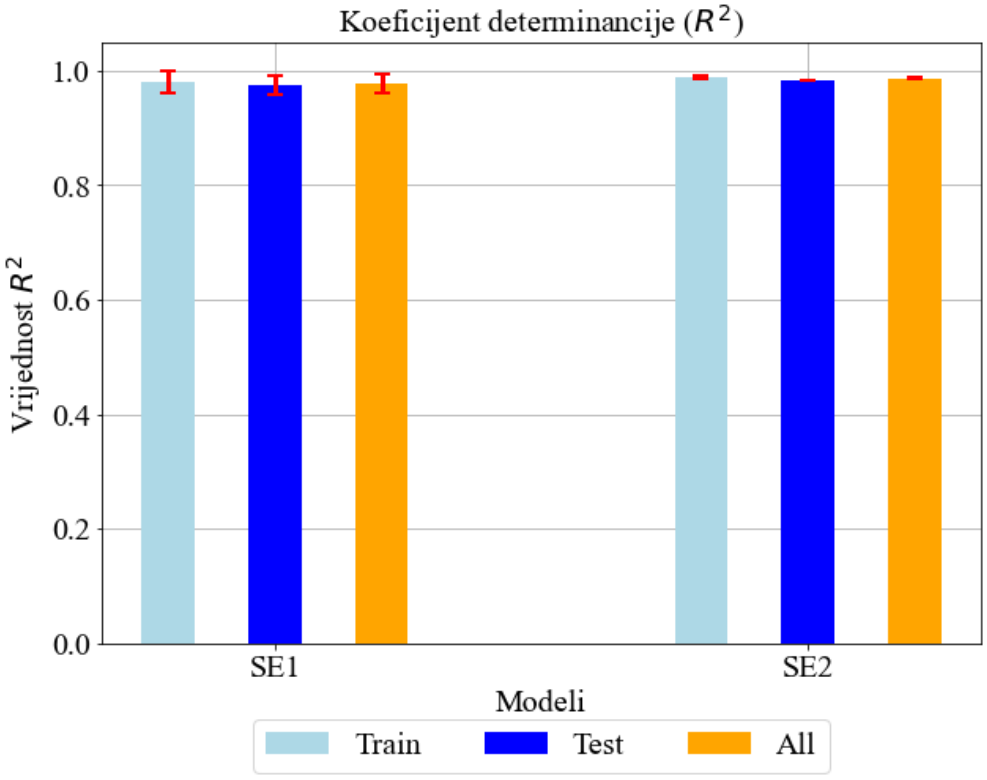
Tablica 6.33. Tablica srednje apsolutne pogreške za modele Stacking ansambla.

Model	$\overline{MAE}$ train	$\sigma(MAE)$ train	$\overline{MAE}$ test	$\sigma(MAE)$ test	$\overline{MAE}$ all	$\sigma(MAE)$ all
Finalni 1	20.52	0.018	23.87	10.57	22.2	11.4
Finalni 2	14.49	0	18.59	0.53	16.54	1.46

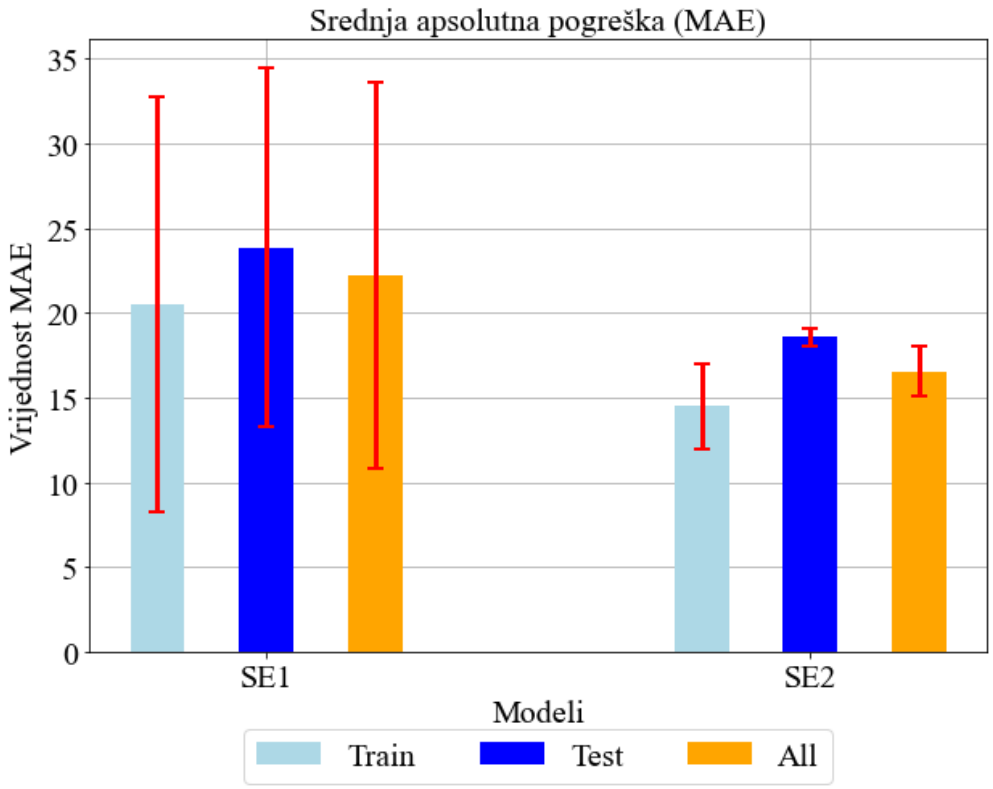
Tablica 6.34. Tablica kvadratnog korjena srednje kvadratne pogreške za modele Stacking ansambla.

Model	$\overline{RMSE}$ train	$\sigma(RMSE)$ train	$\overline{RMSE}$ test	$\sigma(RMSE)$ test	$\overline{RMSE}$ all	$\sigma(RMSE)$ all
Finalni 1	50.85	20.96	59	16.92	54.93	18.81
Finalni 2	40.59	4.71	50.56	1.104	45.57	2.19

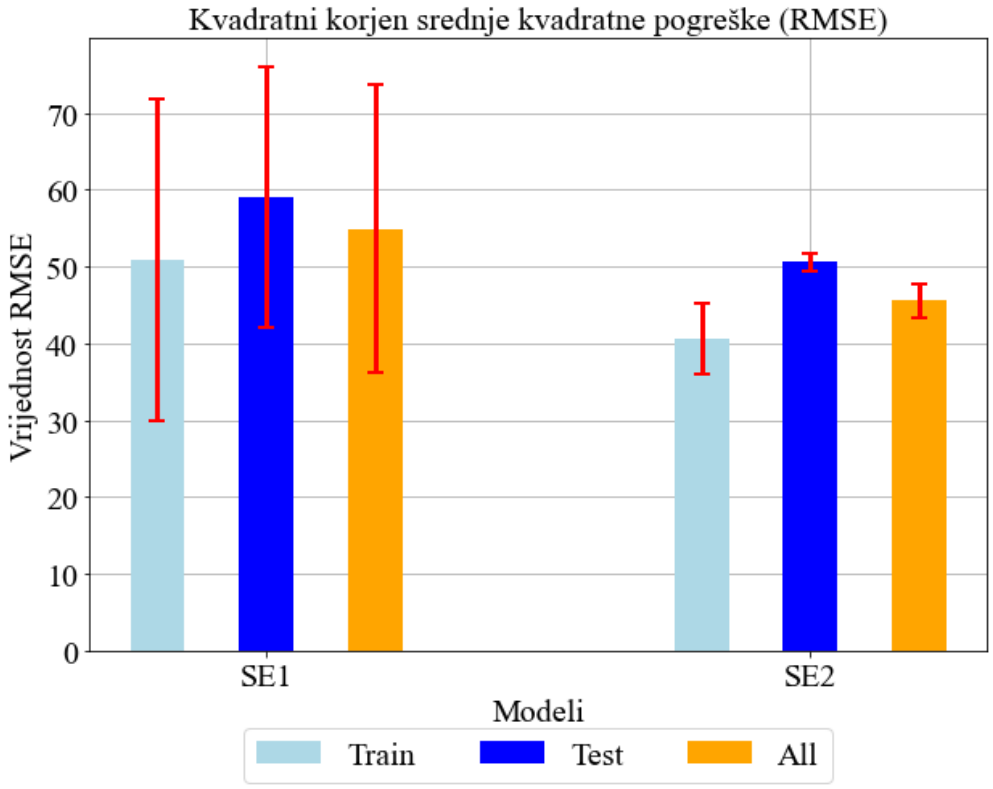
Na slikama 6.13, 6.14 i 6.15 grafički su prikazani rezultati iz prethodno navedenih tablica za dva krajnja stacking ansambl modela. Stacking ansambl 1 (SE1) model je ansambl sačinjen od prethodno treniranih stacking modela koji uključuje stacking model s AdaBoost finalnim estimatorom, dok stacking ansambl 2 (SE2) model je ansambl sačinjen od stacking modela koji ne uključuje stacking model s AdaBoost finalnim estimatorom. Iz tablica i grafičkih prikaza R^2, MAE i RMSE parametara može se vidjeti kako model stacking ansambl 2 ima veću točnost od modela stacking ansambl 1. Standardna devijacija parametara je također veća za stacking ansambl 1 model nego za stacking ansambl 2 model.



Slika 6.13. Grafički prikaz koeficijenta determinancije za finalne ansambl modele.



Slika 6.14. Grafički prikaz srednje apsolutne pogreške za finalne ansambl modele.



Slika 6.15. Grafički prikaz kvadratnog korjena srednje kvadratne pogreške za finalne ansambl modele.

7 DISKUSIJA

U ovom poglavlju bit će detaljnije objašnjen tijek izrade ovog rada i rezultati predstavljeni u prethodnom poglavlju. Nakon pronalaska već postojeće literature na temu rada obrađena je detaljna statistička analiza javno dostupnog dataseta pomoću kojeg su se trenirali odabrani modeli. Korelacijskom analizom utvrđeno je kako postoji korelacija između ulaznih i izlaznih podataka. Najveća korelacija je bila između AC snage (izlazne veličine) i 3 ulaznih veličina, a to su: temperatura okoline, temperatura modula i iradijacija Sunčeve svijetlosti. Korelacijski koeficijenti između tih veličina iznosili su redom: 0.72, 0.95 i 0.99. Dobiveni koeficijenti ukazuju kako postoji velika šansa za razvoj točnog modela za estimaciju izlazne snage fotonaponske elektrane pomoću odabranog seta podataka.

Nakon obrađene statističke analize slijedila je raspodjela podataka na podatke za treniranje i validaciju točnosti odabranih modela. Podaci su raspodijeljeni u omjeru $70:30$ gdje je 70% podataka korišteno za treniranje modela, a 30% podataka za validaciju modela. Validacijke metrike pomoću kojih su se modeli validirali su: koeficijent determinancije (R^2), srednja apsolutna pogreška (MAE), kvadratni korjen srednje kvadratne podgreške (RSME) i Kling-Gupta efficiency parametar (KGE).

Odabrani modeli su se prvobitno trenirali samo s nasumičnim odabirom hiperparametara. Ukupno je odabrano 7 modela za treniranje i validaciju. Odabrani modeli su: Autoregresivni diferencijalni regresor (ARDR), Multi-Layer perceptron (MLP), BayesianRidge regresor, linearna regresija, Huber regresor, ElasticNET i Lasso. U radu su prikazani parametri R^2, MAE, RMSE za podatke na kojima su modeli trenirani i za podatke koji služe za validaciju. Može se vidjeti kako nema razlike između navedenih parametara za train i test podatke. Također standardna devijacija je mala za sve parametre. Modeli s najboljim rezultatima za inicijalno istraživanje bili su: ARDR, BayesianRidge i linearna regresija. Sva tri modela imaju isti koeficijent determinancije koji iznosi 0.98, kod srednje apsolutne pogreške najmanju ima model BayesianRigde i iznosi 26.16, dok kvadratni korjen srednje kvadratne pogreške je najmanji kod linearne regresije i ARDR-a i iznosi 57.31. Modeli ElasticNET i Lasso imali su lošije rezultate od prethodno navedenih modela, a najlošiji rezultati dobiveni su pomoću modela MLP i Huber regresora. Njihov koeficijent determinancije je iznosio 0.91 za MLP i 0.9 za Huber regresor. Također pogreške MAE i RMSE bile su najveće za dva navedena modela. KGE parametar potvrđuje kako su MLP i Huber regresor najlošiji modeli za predviđanje izlazne veličine na treniranom setu podataka. Huber regresor ima negativan KGE parametar koji iznosi -0.53 dok MLP ima bolju vrijednost KGE parametara koja iznosi 0.318.

U sljedećem koraku modeli su se trenirali pomoću unakrsne validacije i nasumičnog pretraživanje hiperparametara kako bi se vidjelo kako će unakrsna validacija utjecati na točnost i robusnost

modela. Na svih 7 modela korištena je 5-fold unakrsna validacija jedino u slučaju MLP modela korištena je i 10-fold unakrsna validacija kako bi se vidjelo hoće li se popraviti točnost modela na taj način. Parametri R^2, MAE i RMSE su također kao i u prethodnom slučaju prikazani za train i test podatke. Ne postoji značajna razlika kod parametra za train i test podatke, također standardna devijacija je izrazito niska. Modeli ARDR i linearna regresija imaju istu vrijednost koeficijenta determinancije kao i u prethodnom slučaju koji iznosi 0.98. BayesianRidge ima malo manju vrijednost R^2 parametra koja iznosi 0.979. Model linearne regresije u ovom slučaju ima najmanju MAE pogrešku koja iznosi 25.73 dok ARDR ima najmanju RMSE pogrešku koja iznosi 54.95. Modelima Lasso i ElasticNET implementacijom unakrsne validacije povećana je točnost i njihovi koeficijenti determinancije sada iznose 0.977 i 0.978. Također smanjila im se MAE i RMSE pogreška dok KGE parametar im je ostao sličan kao i u prethodnom slučaju. Modeli MLP i Huber regresor ponovno su najlošiji modeli s time da im se točnost pogoršala uspoređujući trenutne rezultate s rezultatima iz prethodnog slučaja. Huber regresoru koeficijent determinancije se smanjio na vrijednost od -0.412, a MLP model ima koeficijent determinancije iznosa 0. Također njihove MAE i RMSE pogreške su se povećale. U slučaju MLP modela u radu ispitano je koliko će utjecati 10-fold unakrsna validacija na točnost predviđanja modela. Rezultati pokazuju kako se točnost modela povećala koristeći 10-fold unakrsnu validaciju. Kod MLP(CV10) modela koeficijent determinancije se povećao s 0 na 0.76, a MAE i RMSE se se smanjile s 341.56 i 393.44 na 150.9 i 188.43. Istodobno KGE parametar je veći za MLP koji koristi 10-fold unakrsnu validaciju naprema MLP modela koji koristi 5-fold unakrsnu validaciju.

Nakon prethodnih dva koraka slijedi spajanje najboljih modela u ansambl kako bi se dobio još točniji i robusniji model. U ovom radu uspoređivali su se stacking modeli s različitim finalnim estimatorima. Korišteni finalni estimatori su: AdaBoost, Banging, Extra trees, Random forest, Hist gradient boosting. Modeli koji su odabrani za ansambl su: ARDR, MLP, BayesianRidge, linearna regresija, ElasticNET i Lasso. Huber regresor je izbačen iz ansabmla zbog loših performansi u prethodnim koracima. MLP model je odabran unotač lošijoj performansi u prethodnim koracima zbog poboljšanih rezultata kod 10-fold unakrsne validacije. Prikazani rezultati stacking modela pokazuju kako su svi modeli imali veliku točnost predviđanja izlazne veličine osim stacking modela s Adaboost finalnim estimatorom koji je imao lošije performanse. Stacking modeli s Banging, Extra trees i Random forest finalnim estimatorom su imali koeficijent determinancije koji iznosi 0.99. Stacking model s Hist gradient boosting finalnim estimatorom je imao R^2 parametar 0.98 dok je stacking model s AdaBoost finalnim estimatorom imao najlošiji R^2 koji je iznosio 0.94. Najmanju RMSE i MAE pogešku imao je stacking model s Random forest finalnim estimatorom i pogreške su iznosile 44.14 i 15.49. KGE parametar također je najmanji za stacking models s AdaBoost finalnim estimatorom, a najveći za model s AdaBoost finalnim estimatorom.

U finalnom koraku uspoređivala su se dva krajnja stacking ansambl modela sačinjenih od više prethodno treniranih stacking ansambl modela. Prvi stacking ansambl model sačinjen je od svih prethodno treniranih stacking modela, dok drugi stacking ansambl model je sačinjen od svih prethodno treniranih stacking modela isključujući stacking model s AdaBoost finalnim estimatorom. Rezultati koeficijenta determinancije, MAE i RMSE pogrešaka pokazuju kako je stacking ansambl 2 model točniji od stacking ansambl 1 modela. Koeficijent determinancije, MAE i RMSE pogreške za stacking ansambl 1 model iznose: 0.98, 22.2 i 54.93 dok za stacking ansambl 2 model iznose: 0.99, 16.54, 45.57. Također iz rezultata se može zaključiti kako stacking ansambl 2 model ima najveći koeficijent determinancije i najmanje MAE i RMSE pogreške od svih prethodnih modela. Iz toga razloga je najprikladniji za estimaciju izlazne snage fotonaponske elektrane.

8 ZAKLJUČAK

U ovom radu istražena je mogućnost dobivanja visoke točnosti estimacije izlazne snage fotonaponske elektrane pomoću algoritama strojnog učenja. Prije početka treniranja i validiranja različitih modela strojnog učenja u radu je navedeno nekoliko drugih istraživanja na zadanu temu. Navedena istraživanja potkrijepljuju mogućnost točne estimacije izlazne snage fotonaponske elektrane.

Nakon pregleda već postojećih istraživanja odabran je javno dostupan set podataka koji će se koristiti za treniranje i validaciju modela unutar rada. Set podataka je sačinjen od mjerenja u razdoblju od 34 dana. Podaci su mjereni unutar 2 fotonaponske elektrane koje se nalaze u Indiji. Na odabranom setu podataka napravljena je detaljna statistička analiza i utvrđeno je kako podaci iz jedne elektrane nisu prikladni za treniranje modela i zato su za istraživanje korišteni podaci samo iz jedne elektrane. Nakon statističke obrade podataka odabrane su veličine koje će služiti kao ulazni podaci u modele strojnog učenja i koji će služiti za predviđanje AC snage fotonaponske elektrane. Uz odabrane ulazne i izlazne veličine modela set podataka je podijeljen na podatke za treniranje i na podatke za validaciju modela. Za validaciju modela odabrane metrike su: koeficijent determinancije (R^2), srednja apsolutna pogreška, kvadratni korjen srednje kvadratne pogreške (RMSE) i Kling-Gupta efficiency (KGE).

Za inicijalno istraživanje odabrani modeli su trenirani samo pomoću nasumičnog odabira hiperparametara. Modeli odabrani za inicijalno treniranje su: Autoregresivni diferencijalni regresor (ARDR), Multi-Layer perceptron (MLP), BayesianRidge regresor, linearna regresija, Huber regresor, ElasticNET i Lasso. Od navedenih modela, modeli s najvećom točnošću bili su: ARDR, BayesianRidge i linearna regresija s koeficijentom deterninancije 0.98. Najlošiji modeli bili su MLP i Huber regresor uz to što je Huber regresor imao KGE parametar koji je iznosio -0.53.

Unotač lošijem rezultatom nekih modela kod inicijalnog istraživanja modeli koji su imali visoku točnost ukazuju kako bi mogla postojati mogućnost za razvoj točnog i robusnog modela za estimaciju izlazne snage fotonaponske elektrane. Nakon inicijalnog istraživanja samo s nasumičnim odabirom hiperparametara sada su se modeli trenirali pomoću 5-fold unakrsne validacije i nasumičnog odabira hiperparametara. U ovom slučaju najbolji modeli su bili: ARDR i linearna regresija s jednakim iznosom koeficijenta determinancije 0.98, dok je linearna regresija imala manju MAE pogrešku 25.73, a ARDR manju RMSE pogrešku 54.95. Najlošiji modeli bili su opet MLP i Huber regresor.

U sljedećem koraku odabrani su najbolji modeli iz prethodna dva koraka koji su se koristili za stacking ansambl model. Za ansambl model odabrani su svi modeli osim Huber regresora. Unotač lošijoj performansi MLP modela odabran je za stacking ansambl model zbog poboljšane točnosti

kod treniranja s 10-fold unakrsne validacije. Trenirano je 5 stacking modela, svaki sa svojim finalnim estimatorom. Finalni estimatori koji su se koristili su: AdaBoost, Banging, ExtraTrees, Random forest, Hist gradient boosting. Od svih stacking modela najlošiji je bio s Adaboost finalnim estimatorom. Ostali modeli su imali slične performanse kod treniranja i validacije.

Krajnji ansambl modeli izrađeni su od stacking modela s različitim finalnim estimatorima. Kako bi se odabrao model s najboljom točnošću trenirana i validirana su dva krajnja modela. Prvi krajnji model je napravljen od svih stacking modela iz prethodnog koraka, dok drugi krajnji model ne uključuje stacking model s AdaBoost finalnim estimatorom. Usporedbom koeficijenta determinancije, MAE i RMSE pogrešaka utvrđeno je kako stacking ansambl 2 model bez stacking modela s AdaBoost finalnim estimatorom ima bolje performanse. Za bolji krajnji model koeficijent determinancije iznosi 0.99, MAE pogreška 16.54, a RMSE pogreška 45.57.

Rezultatima stacking ansambl 2 modela utvrđeno je kako je moguće razviti točan i robusan model za predviđanje izlazne snage fotonaponske elektrane pomoću algoritama strojnog učenja.

Prednosti ovakvog modela za predviđanje izlazne snage fotonaponske elektrane su to što se uklanja element nepredvidljivosti poizvodnje energije elektrane, što uvelike pomaže fotonaponskim elektranama unutar energetskog tržišta. Nedostatak ovakvog modela je potreba za dobrim podacima, jer ako podaci koji se koriste za treniranje modela nisu dobro mjereni onda i točnost modela neće biti zadovoljavajuća. Također model koji dobro predviđa izlaznu snagu elektrane na jednom području ne mora automatski biti dobar za predviđanje snage fotonaponske elektrane na drugom području.

Bibliografija

[1] P. (24.05.2024.), "https://en.wikipedia.org," *Coefficient of determination*.

[2] P. (24.05.2024.), "https://en.wikipedia.org," *Mean absolute error*.

[3] P. (24.05.2024.), "https://help.sap.com/docs/," *Root Mean Squared Error (RMSE)*.

[4] P. (25.05.2024.), "https://en.wikipedia.org," *Kling–Gupta efficiency*.

[5] P. (25.05.2024.), "https://www.analyticsvidhya.com/blog/2022/11/hyperparameter-tuning-using-randomized-search/,"

[6] P. (24.05.2024.), "https://medium.com," *Stacking to Improve Model Performance: A Comprehensive Guide on Ensemble Learning in Python*.

[7] P. (27.05.2024.), "https://www.geeksforgeeks.org/cross-validation-machine-learning/,"

[8] L. Gutiérrez, J. Patiño, and E. Duque-Grisales, "A comparison of the performance of supervised learning algorithms for solar power prediction," *Energies*, vol. 14, no. 15, p. 4424, 2021.

[9] A. A. Mas' ud, "Comparison of three machine learning models for the prediction of hourly pv output power in saudi arabia," *Ain Shams Engineering Journal*, vol. 13, no. 4, p. 101648, 2022.

[10] S. M. Malakouti, M. B. Menhaj, and A. A. Suratgar, "The usage of 10-fold cross-validation and grid search to enhance ml methods performance in solar farm power generation prediction," *Cleaner Engineering and Technology*, vol. 15, p. 100664, 2023.

[11] A. K. Tripathi, N. K. Sharma, J. Pavan, and S. Bojjagania, "Output power prediction of solar photovoltaic panel using machine learning approach," *International Journal of Electrical and Electronics Research*, vol. 10, no. 4, pp. 779–783, 2022.

[12] J. Gaboitaolelwe, A. M. Zungeru, A. Yahya, C. K. Lebekwe, D. N. Vinod, and A. O. Salau, "Machine learning based solar photovoltaic power forecasting: A review and comparison," *IEEE Access*, vol. 11, pp. 40820–40845, 2023.

[13] G. Narvaez, L. F. Giraldo, M. Bressan, and A. Pantoja, "Machine learning for site-adaptation and solar radiation forecasting," *Renewable Energy*, vol. 167, pp. 333–342, 2021.

[14] I. Jebli, F.-Z. Belouadha, M. I. Kabbaj, and A. Tilioua, "Prediction of solar energy guided by pearson correlation using machine learning," *Energy*, vol. 224, p. 120109, 2021.

[15] S. Theocharides, G. Makrides, A. Livera, M. Theristis, P. Kaimakis, and G. E. Georghiou, "Day-ahead photovoltaic power production forecasting methodology based on machine learning and statistical post-processing," *Applied Energy*, vol. 268, p. 115023, 2020.

[16] C.-R. Chen and U. Three Kartini, "K-nearest neighbor neural network models for very short-term global solar irradiance forecasting based on meteorological data," *Energies*, vol. 10, no. 2, p. 186, 2017.

[17] S. Theocharides, G. Makrides, G. E. Georghiou, and A. Kyprianou, "Machine learning algorithms for photovoltaic system power output prediction," in *2018 IEEE International Energy Conference (ENERGYCON)*, pp. 1–6, IEEE, 2018.

[18] P. (28.08.2023.), "https://news.energysage.com/solar-panel-temperature-overheating/,"

[19] P. (22.05.2024.), "https://aws.amazon.com," *What are autoregressive models?*

[20] P. (22.05.2024.), "https://deepai.org/machine-learning-glossary-and-terms/autoregressive-model,"

[21] P. (28.08.2023.), "https://scikit-learn.org/stable/index.html,"

[22] P. (21.05.2024.), "https://vitalflux.com/," *MLPRegressor*.

[23] P. (22.05.2024.), "https://www.geeksforgeeks.org/," *artificial-neural-networks-and-its-applications*.

[24] P. (21.05.2024.), "https://www.pycodemates.com," *Regression with Multilayer Perceptron(MLP) Using Python*.

[25] P. (22.05.2024.), "https://www.tutorialspoint.com," *Scikit Learn - Bayesian Ridge Regression*.

[26] P. (22.05.2024.), "https://www.simplilearn.com/tutorials/data-science-tutorial/bayesian-linear-regression,"

[27] P. (22.05.2024.), "https://www.geeksforgeeks.org/ml-linear-regression/,"

[28] P. (22.05.2024.), "https://machinelearningcompass.com/," *Elastic Net Regression Explained, Step by Step*.

[29] P. (22.05.2024.), "https://www.mygreatlearning.com/blog/," *A Complete understanding of LASSO Regression*.

[30] P. (24.05.2024.), "https://www.analyticsvidhya.com/blog/2021/08/ensemble-stacking-for-machine-learning-and-deep-learning/,"

9 SAŽETAK I KLJUČNE RIJEČI

U ovom radu je istraženo je li moguće dobiti točnu estimaciju izlazne snage fotonaponske elektrane pomoću algoritama strojnog učenja. Na odabranom setu podataka mjerenih u periodu od 34 dana u fotonaponskoj elekrani u Indiji napravljena je detaljna statistička analiza, odabrane su ulazne i izlazne veličine i podaci su raspodijeljeni na podatke za treniranje i na podatke za validaciju. Metrike koje su se koristile za validaciju su: koeficijent determinancije (R^2), srednja apsolutna pogreška (MAE), kvadratni korjen srednje kvadratne pogreške (RMSE) i Kling-Gupta efficiency (KGE). Inicijalno istraživanje je rađeno na 7 modela strojnog učenja pomoću nasumičnog odabira hiperparametara. Trenirani modeli su: Autoregresivni diferencijalno regresor (ARDR), Multi-Layer perceptron (MLP), BayesianRidge regresor, linearna regresija, Huber regresor, ElasticNET i Lasso. Nakon inicijalnog istraživanja sljedeći korak je bio treniranje modela pomoću nasumičnog odabira hiperparametara i unakrsnom validacijom. Rezultati iz dva prethodna koraka su pokazali kako Huber regresor ima loše performanse i neće se koristiti za daljnje istraživanje. U sljedećem koraku trenirani su stacking ansambl modeli s različitim finalnim estimatorima. Stacking modeli su sačinjeni od prethodno treniranih modela ne uključujući Huber regresor. Na kraju trenirani stacking ansambl modeli su se koristili za treniranje krajnjeg ansambl modela koji je imao zadovoljavajuću točnost estimacije izlazne snage fotonaponske elektrane. Stacking ansambl 2 model je imao koeficijent determinancije iznosa 0.99, srednju apsolutnu pogrešku iznosa 16.54 i kvadratni korjen srednje kvadratne pogreške 45.57.

Ključne riječi: *fotonaponska elektrana, umjetna inteligencija, Python, unakrsna validacija, strojno učenje, estimacija snage, energetika, iradijacija, ansambl.*

10 SUMMARY AND KEYWORDS

This paper investigates whether it is possible to accurately estimate the output power of a photovoltaic power plant using machine learning algorithms. A detailed statistical analysis was performed on a selected dataset measured over a period of 34 days at a photovoltaic power plant in India. Input and output variables were selected, and the data was divided into training and validation sets. The metrics used for validation were: coefficient of determination (R^2), mean absolute error (MAE), root mean square error (RMSE), and Kling-Gupta efficiency (KGE). The initial research was conducted on 7 machine learning models using random selection of hyperparameters. The trained models were: Autoregressive Differential Regressor (ARDR), Multi-Layer Perceptron (MLP), Bayesian Ridge Regressor, Linear Regression, Huber Regressor, ElasticNET, and Lasso. After the initial research, the next step was training the models using random selection of hyperparameters and cross-validation. The results from the previous two steps showed that the Huber Regressor had poor performance and would not be used for further research. In the next step, stacking ensemble models with different final estimators were trained. The stacking models consisted of the previously trained models, excluding the Huber Regressor. Finally, the trained stacking ensemble models were used to train the final ensemble model, which achieved satisfactory accuracy in estimating the output power of the photovoltaic power plant. The final model had a coefficient of determination of 0.99, a mean absolute error of 16.54, and a root mean square error of 45.57.

Keywords: *photovoltaic power plant, artificial intelligence, Python, cross-validation, machine learning, power estimation, energy, irradiation, ensemble.*

11 PRILOZI

11.1 PRILOG A- Popis kratica i oznaka

- R^2- koeficijent determinancije
- RMSE- kvadratni korjen srednje kvadratne pogreške
- MAE- srednja apsolutna pogreška
- KGE- Kling-Gupta efficiency
- nRMSE- normalizirani kvadratni korjen srednje kvadratne pogreške
- MSE- srednja kvadratna pogreška
- ACC- accuracy
- SS- skill score
- MAPE- srednja apsolutna postotna pogreška
- KNN- k-nearest neighbour
- LR- linearna regresija
- ANN- artificial neural networks
- SVM- support vector machines
- MLR- multiple regression
- DTR- decision tree regressor
- ETR- extra tree regressor
- LGBM- light gradient boosting machine
- GR- Gaussian regression
- ABR- AdaBoost regressor
- RFR- random forest regresor
- RT- regression trees

11.2 DODATAK B- Treniranje modela pomoću nasumičnog pretraživanja hiperparametara

```
1
2  import random
3  import os
4  import numpy as np
5  import pandas as pd
6  import matplotlib.pyplot as plt
7  import datetime as dt
8  from sklearn.model_selection import train_test_split
9  from sklearn.linear_model import ARDRegression
10 from sklearn.metrics import (r2_score, mean_absolute_error, mean_squared_error,
       mean_absolute_percentage_error)
11
12 # Odabir random hiperparametara za ARDRegresor
13 def ARDParSearch():
14   parameters= []
15
16   NumIter= random.randint(100, 1000)
17   tolerance= round(np.random.uniform(1e-28, 1e-26), 30)
18   alpha1 = round(np.random.uniform(1e-5,1e-1),20)
19   alpha2 = round(np.random.uniform(1e-5,1e-1),20)
20   lambda1 = round(np.random.uniform(1e-5,1e-1),20)
21   lambda2 = round(np.random.uniform(1e-5,1e-1),20)
22   computeScore = random.choice([True,False])
23   thresholdLambda = random.randint(1000,100000)
24   verbose = True
25
26   #Spremanje random odabaranih podataka u listu
27   parameters.append(NumIter)
28   parameters.append(tolerance)
29   parameters.append(alpha1)
30   parameters.append(alpha2)
31   parameters.append(lambda1)
32   parameters.append(lambda2)
33   parameters.append(computeScore)
34   parameters.append(thresholdLambda)
35   parameters.append(verbose)
36
37   file0.flush()
38   return parameters
39
40
41 # Ucitavanje i priprema dataseta
42 GenDataPlant2= pd.read_csv("C:/Users/leomi/Desktop/Diplomski/Data/NewData/Plant_2/
       Plant_2_Generation_Data.csv")
43 GenDataPlant2.drop("PLANT_ID", 1, inplace= True)
44
45 WeatherPlant2= pd.read_csv("C:/Users/leomi/Desktop/Diplomski/Data/NewData/Plant_2/
       Plant_2_Weather_Sensor_Data.csv")
46 WeatherPlant2.drop("PLANT_ID", 1, inplace= True)
47
48 GenDataPlant2["DATE_TIME"]= pd.to_datetime(GenDataPlant2["DATE_TIME"], format= "%Y-%m
       -%d %H:%M")
49 WeatherPlant2["DATE_TIME"]= pd.to_datetime(WeatherPlant2["DATE_TIME"], format= "%Y-%m
       -%d %H:%M")
50
51 MergeDataPlant2= pd.merge(GenDataPlant2, WeatherPlant2, on= "DATE_TIME")
52 MergeDataPlant2.drop(["DATE_TIME", "SOURCE_KEY_x","SOURCE_KEY_y",], 1, inplace= True)
53
54
55
56 # Raspodjela podataka
57 X = MergeDataPlant2[["DAILY_YIELD", "TOTAL_YIELD", "AMBIENT_TEMPERATURE", "
       MODULE_TEMPERATURE", "IRRADIATION"]]
```

```
y= MergeDataPlant2[["AC_POWER"]]

X_train, X_test, y_train, y_test= train_test_split(X,y, test_size= 0.3)

# Model
def trainTestFun(X_train, X_test, y_train, y_test):

  ARDParameters= ARDParSearch()
  Model= ARDRegression(n_iter= ARDParameters[0],
  tol= ARDParameters[1],
  alpha_1= ARDParameters[2],
  alpha_2= ARDParameters[3],
  lambda_1= ARDParameters[4],
  lambda_2= ARDParameters[5],
  compute_score= ARDParameters[6],
  threshold_lambda= ARDParameters[7],
  verbose= ARDParameters[8])

  Model.fit(X_train, y_train)

  # Rezultati
  R2_train = r2_score(y_train, Model.predict(X_train))
  R2_test = r2_score(y_test, Model.predict(X_test))
  MAE_train = mean_absolute_error(y_train, Model.predict(X_train))
  MAE_test= mean_absolute_error(y_test, Model.predict(X_test))
  RMSE_train = np.sqrt(mean_squared_error(y_train, Model.predict(X_train)))
  RMSE_test = np.sqrt(mean_squared_error(y_test, Model.predict(X_test)))

  R2_MEAN = np.mean([R2_train, R2_test])
  MAE_MEAN = np.mean([MAE_train, MAE_test])
  RMSE_MEAN = np.mean([RMSE_train, RMSE_test])
  R2_STD = np.std([R2_train, R2_test])
  MAE_STD = np.std([MAE_train, MAE_test])
  RMSE_STD = np.std([RMSE_train, RMSE_test])

  print("R2_train = {}".format(R2_train.round(2)))
  print("R2_test = {}".format(R2_test.round(2)))
  print("-----------------------------------")
  print("MAE_train = {}".format(MAE_train.round(2)))
  print("MAE_test = {}".format(MAE_test.round(2)))
  print("-----------------------------------")
  print("RMSE_train = {}".format(RMSE_train.round(2)))
  print("RMSE_test = {}".format(RMSE_test.round(2)))
  print("-----------------------------------")

  print("\n#################################\n")

  print("R2_MEAN = {}".format(R2_MEAN.round(2)))
  print("R2_STD = {}".format(R2_STD.round(2)))
  print("-----------------------------------")
  print("MAE_MEAN = {}".format(MAE_MEAN.round(2)))
  print("MAE_STD = {}".format(MAE_STD.round(2)))
  print("-----------------------------------")
  print("RMSE_MEAN = {}".format(RMSE_MEAN.round(2)))
  print("RMSE_STD = {}".format(RMSE_STD.round(2)))
  print("-----------------------------------")


  if R2_MEAN > 0.3:
    #Zapis random parametara u file
    file0.write("ARDRegression_PARAM= NumIter= {},\t tolerance= {},\t alpha1={},\t
    alpha2= {},\t lambda1= {},\t lambda2= {},\t computeScore= {},\t thresholdLambda=
    {},\t verbose= {}\n ".format(ARDParameters[0],
      ARDParameters[1],
      ARDParameters[2],
      ARDParameters[3],
      ARDParameters[4],
      ARDParameters[5],
```

```
      ARDParameters[6],
      ARDParameters[7],
      ARDParameters[8]))
    file0.write("
    ##############################################################################\
    n" +\
            "R2_train = {}\n".format(R2_train.round(2))+\
            "R2_test = {}\n".format(R2_test.round(2))+\
            "----------------------------------\n"+\
            "MAE_train = {}\n".format(MAE_train.round(2))+\
            "MAE_test = {}\n".format(MAE_test.round(2))+\
            "----------------------------------\n"+\
            "RMSE_train = {}\n".format(RMSE_train.round(2))+\
            "RMSE_test = {}\n".format(RMSE_test.round(2))+\
            "----------------------------------\n"+\
            "----------------------------------\n"+\
            "----------------------------------\n"+\
            "R2_MEAN = {}\n".format(R2_MEAN.round(2))+\
            "R2_STD = {}\n".format(R2_STD.round(2))+\
            "----------------------------------\n"+\
            "MAE_MEAN = {}\n".format(MAE_MEAN.round(2))+\
            "MAE_STD = {}\n".format(MAE_STD.round(2))+\
            "----------------------------------\n"+\
            "RMSE_MEAN = {}\n".format(RMSE_MEAN.round(2))+\
            "RMSE_STD = {}\n".format(RMSE_STD.round(2))+\
            "----------------------------------\n"+\
            "##################################################################\n")

  if R2_MEAN > 0.99:
    print("#################################################################")
    print(" Final Evaluation")
    print("#################################################################")
    file0.write("#################################################################\n")
    file0.write(" Final Evaluation\n")
    file0.write("#################################################################\n")

    Model.fit(X_train, y_train)

    R2_test = r2_score(y_test, Model.predict(X_test))
    MAE_test= mean_absolute_error(y_test, Model.predict(X_test))
    RMSE_test = np.sqrt(mean_squared_error(y_test, Model.predict(X_test)))
    print("R2_test = {}".format(R2_test.round(2)))
    print("MAE_test = {}".format(MAE_test.round(2)))
    print("RMSE_test = {}".format(RMSE_test.round(2)))
    file0.write("Final R^2 Test = {}\n".format(R2_test))
    file0.write("Final MAE Test = {}\n".format(MAE_test))
    file0.write("Final RMSE Test = {}\n".format(RMSE_test))

    file0.write("###############################################################\n")
    file0.flush()

    return R2_test

  else:
    return R2_MEAN

name= "ARDPlant2"
file0= open("{}_score.dat".format(name),"w")

while True:
  res= trainTestFun(X_train, X_test, y_train, y_test)
  if res>0.99:
    print("Solution is Found!")
    break
  else:
    continue

file0.close()
```

11.3 DODATAK C- Treniranje modela pomoću nasumičnog pretraživanja hiperparametara i unakrsnom validacijom

```
import random
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import datetime as dt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import ARDRegression
from sklearn.model_selection import cross_validate
from sklearn.metrics import (r2_score, mean_absolute_error, mean_squared_error,
    mean_absolute_percentage_error)

# Odabir random hiperparametara za ARDRegresor
def ARDParSearch():
  parameters= []

  NumIter= random.randint(100, 1000)
  tolerance= round(np.random.uniform(1e-28, 1e-26), 30)
  alpha1 = round(np.random.uniform(1e-5,1e-1),20)
  alpha2 = round(np.random.uniform(1e-5,1e-1),20)
  lambda1 = round(np.random.uniform(1e-5,1e-1),20)
  lambda2 = round(np.random.uniform(1e-5,1e-1),20)
  computeScore = random.choice([True,False])
  thresholdLambda = random.randint(1000,100000)
  verbose = True

  #Spremanje random odabaranih podataka u listu
  parameters.append(NumIter)
  parameters.append(tolerance)
  parameters.append(alpha1)
  parameters.append(alpha2)
  parameters.append(lambda1)
  parameters.append(lambda2)
  parameters.append(computeScore)
  parameters.append(thresholdLambda)
  parameters.append(verbose)

  file0.flush()
  return parameters


# Ucitavanje i priprema dataseta
GenDataPlant2= pd.read_csv("C:/Users/leomi/Desktop/Diplomski/Data/NewData/Plant_2/
    Plant_2_Generation_Data.csv")
GenDataPlant2.drop("PLANT_ID", 1, inplace= True)

WeatherPlant2= pd.read_csv("C:/Users/leomi/Desktop/Diplomski/Data/NewData/Plant_2/
    Plant_2_Weather_Sensor_Data.csv")
WeatherPlant2.drop("PLANT_ID", 1, inplace= True)

GenDataPlant2["DATE_TIME"]= pd.to_datetime(GenDataPlant2["DATE_TIME"], format= "%Y-%m
    -%d %H:%M")
WeatherPlant2["DATE_TIME"]= pd.to_datetime(WeatherPlant2["DATE_TIME"], format= "%Y-%m
    -%d %H:%M")

MergeDataPlant2= pd.merge(GenDataPlant2, WeatherPlant2, on= "DATE_TIME")

MergeDataPlant2.drop(["DATE_TIME", "SOURCE_KEY_x","SOURCE_KEY_y",], 1, inplace= True)

# Raspodjela podataka

X = MergeDataPlant1[["DAILY_YIELD", "TOTAL_YIELD", "AMBIENT_TEMPERATURE", "
    MODULE_TEMPERATURE", "IRRADIATION"]]
```

```
y= MergeDataPlant1[["AC_POWER"]]

X_train, X_test, y_train, y_test= train_test_split(X,y, test_size= 0.3)

# Model
def trainTestFun(X_train, X_test, y_train, y_test):

  ARDParameters= ARDParSearch()
  Model= ARDRegression(n_iter= ARDParameters[0],
  tol= ARDParameters[1],
  alpha_1= ARDParameters[2],
  alpha_2= ARDParameters[3],
  lambda_1= ARDParameters[4],
  lambda_2= ARDParameters[5],
  compute_score= ARDParameters[6],
  threshold_lambda= ARDParameters[7],
  verbose= ARDParameters[8])

  cvmodel= cross_validate(Model, X_train, y_train, cv=10,
  scoring= ("r2", "neg_mean_absolute_error", "neg_root_mean_squared_error", "
    neg_mean_absolute_percentage_error"), return_train_score= True)

  print("#########################################################################")
  print("# Results from CV 5 Cross Validation Using Multiple Metric")
  print("#########################################################################")
  print("All Scores From CV5 = {}".format(cvmodel))

  print("#########################################################################")
  print("# Calculate Mean and Standard Deviation of Metric values ")
  print("#########################################################################")
  AvrR2ScoreTrain = np.mean(cvmodel['train_r2'])
  StdR2ScoreTrain = np.std(cvmodel['train_r2'])
  AvrR2ScoreTest = np.mean(cvmodel['test_r2'])
  StdR2ScoreTest = np.std(cvmodel['test_r2'])
  AvrAllR2Score = np.mean([AvrR2ScoreTrain,AvrR2ScoreTest])
  StdAllR2Score = np.std([AvrR2ScoreTrain, AvrR2ScoreTest])

  AvrMAEScoreTrain = np.mean(abs(cvmodel['train_neg_mean_absolute_error']))
  StdMAEScoreTrain = np.std(abs(cvmodel['train_neg_mean_absolute_error']))
  AvrMAEScoreTest = np.mean(abs(cvmodel['test_neg_mean_absolute_error']))
  StdMAEScoreTest = np.std(abs(cvmodel['test_neg_mean_absolute_error']))
  AvrAllMAEScore = np.mean([AvrMAEScoreTrain, AvrMAEScoreTest])
  StdAllMAEScore = np.std([AvrMAEScoreTrain, AvrMAEScoreTest])

  AvrRMSEScoreTrain = np.mean(abs(cvmodel['train_neg_root_mean_squared_error']))
  StdRMSEScoreTrain = np.std(abs(cvmodel['train_neg_root_mean_squared_error']))
  AvrRMSEScoreTest = np.mean(abs(cvmodel['test_neg_root_mean_squared_error']))
  StdRMSEScoreTest = np.std(abs(cvmodel['test_neg_root_mean_squared_error']))
  AvrAllRMSEScore = np.mean([AvrRMSEScoreTrain, AvrRMSEScoreTest])
  StdAllRMSEScore = np.std([AvrRMSEScoreTrain, AvrRMSEScoreTest])

  print("CV-R^2 Score = {}".format(AvrAllR2Score))
  print("CV-STD R^2 Score = {}".format(StdAllR2Score))
  print("CV-MAE Score = {}".format(AvrAllMAEScore))
  print("CV-STD MAE Score = {}".format(StdAllMAEScore))
  print("CV-RMSE Score ={}".format(AvrAllRMSEScore))
  print("CV-STD RMSE Score = {}".format(StdAllRMSEScore))
  print("CV-MAPE Score = {}".format(AvrAllMAPEScore))
  print("CV-STD MAPE Score = {}".format(StdAllMAPEScore))


  if AvrAllR2Score >= 0.98:
    file0.write("-------------------------------------------------------------------\
    n")
    file0.write("-------------------------------------------------------------------\
    n")
    file0.write("ARDRegression_PARAM= NumIter= {},\t tolerance= {},\t alpha1={},\t
    alpha2= {},\t lambda1= {},\t lambda2= {},\t computeScore= {},\t thresholdLambda=
```

```
    {},\t verbose= {}\n ".format(ARDParameters[0],
      ARDParameters[1],
      ARDParameters[2],
      ARDParameters[3],
      ARDParameters[4],
      ARDParameters[5],
      ARDParameters[6],
      ARDParameters[7],
      ARDParameters[8]))

    file0.write("#####################################################################\n
    ")
    file0.write(" Results from CV 5 Cross Validation Using Multiple Metric\n")
    file0.write("#####################################################################\n
    ")

    file0.write("R2 Train Scores = {}\n".format(cvmodel['train_r2']))
    file0.write("R2 Test Scores = {}\n".format(cvmodel['test_r2']))
    file0.write("MAE Train Scores = {}\n".format(abs(cvmodel['
    train_neg_mean_absolute_error'])))
    file0.write("MAE Test Scores = {}\n".format(abs(cvmodel['
    test_neg_mean_absolute_error'])))
    file0.write("RMSE Train Scores = {}\n".format(abs(cvmodel['
    train_neg_root_mean_squared_error'])))
    file0.write("RMSE Test Scores = {}\n".format(abs(cvmodel['
    test_neg_root_mean_squared_error'])))

    file0.write("#####################################################################\n"+\
    "AvrR2Score Train = {}\n".format(AvrR2ScoreTrain)+\
    "StdR2Score Train = {}\n".format(StdR2ScoreTrain)+\
    "AvrR2Score Test = {}\n".format(AvrR2ScoreTest)+\
    "StdR2Score Test = {}\n".format(StdR2ScoreTest)+\
    "AvrAllR2Score = {}\n".format(AvrAllR2Score)+\
    "StdAllR2Score = {}\n".format(StdAllR2Score)+\
    "AvrMAEScore Train = {}\n".format(AvrMAEScoreTrain)+\
    "StdMAEScore Train = {}\n".format(StdMAEScoreTrain)+\
    "AvrMAEScore Test = {}\n".format(AvrMAEScoreTest)+\
    "StdMAEScore Test ={}\n".format(StdMAEScoreTest)+\
    "AvrAllMAEScore = {}\n".format(AvrAllMAEScore)+\
    "StdAllMAEScore = {}\n".format(StdAllMAEScore)+\
    "AvrRMSEScore Train = {}\n".format(AvrRMSEScoreTrain)+\
    "StdRMSEScore Train = {}\n".format(StdRMSEScoreTrain)+\
    "AvrRMSEScore Test = {}\n".format(AvrRMSEScoreTest)+\
    "StdRMSEScore Test = {}\n".format(StdRMSEScoreTest)+\
    "AvrAllRMSEScore = {}\n".format(AvrAllRMSEScore)+\
    "StdAllRMSEScore = {}\n".format(StdAllRMSEScore)+\
    "#####################################################################\n")

  if AvrAllR2Score > 0.99:
    print("#####################################################################")
    print(" Final Evaluation")
    print("#####################################################################")
    file0.write("#####################################################################\n")
    file0.write(" Final Evaluation\n")
    file0.write("#####################################################################\n")
    Model.fit(X_train,y_train)
    R2Test = Model.score(X_test,y_test)
    MAETest = mean_absolute_error(y_test, Model.predict(X_test))
    RMSETest = np.sqrt(mean_squared_error(y_test, Model.predict(X_test)))
    print("R^2 Test = {}".format(R2Test))
    print("MAE Test = {}".format(MAETest))
    print("RMSE Test = {}".format(RMSETest))
    file0.write("#####################################################\n")
    file0.write("model R^2 Test = {}\n".format(R2Test))
    file0.write("MAE Test = {}\n".format(MAETest))
    file0.write("RMSE Test = {}\n".format(RMSETest))
    file0.write("#####################################################\n")
    file0.flush()
```

```
    return R2Test
  else:
    return AvrAllR2Score
name = "ARD_CV_score_Plant1(10)"
file0 = open("{}_parameters.data".format(name), "w")

k = 0
while True:
  print("Current Iteration = {}".format(k))
  test = trainTestFun(X_train, X_test, y_train, y_test)
  k+=1
  if test > 0.99:
    print("Solution is Found!!")
    file0.write("Solution is Found!!!")
    file0.flush()
    break
  else:
  continue

```