

1. MELLÉKLETEK

1.1. Végő hiperparaméterek és jelentésük

1.1.1. HIPERPARAMÉTEREK JELENTÉSE

A hiperparaméterek az egyes algoritmusokra jellemző, működésüket befolyásoló kezdeti paraméterek, amelyeket a futtatás előtt be kell állítani. Ezen értékek beállítása kulcsfontosságú egy megfelelő gépi tanulási módszer működéséhez, mert ezen paraméterek segítségével tudjuk finomhangolni az algoritmusokat a specifikus, általunk alkalmazott adatszetek legjobb megismerésének érdekében. A következőkben az egyes algoritmusokra és hálókra jellemző paraméterek jelentését fogjuk bemutatni a működésük jobb megértésének érdekében. A következő alfejezetekben bemutatott hiperparaméterek jobb megértésének érdekében javasoljuk az Hiba! A hivatkozási forrás nem található. fejezetben leírtak párhuzamos tanulmányozását.

1.1.1.1. Random forest regressor

- criterion: a fákon belüli elágazásokhoz alkalmazott minőségi kritérium
- max_depth: az egyes fák maximum mélysége
- n_estimators: az alkalmazott fák száma az „erdőben”
- min_samples_split: az elágazások létrejöttéhez szükséges minimum adatmennyiség
- max_features: A legjobb osztás keresésekor figyelembe veendő jellemzők száma

1.1.1.2. Support vector regressor

- kernel: az algoritmus „szívének”, a kernelnek (=mag) matematikai alakját megadó paraméter
- degree: a kernel egyenlet polinomiális foka
- gamma: a kernel együtthatója

- C: szabályozási paraméter, amelynek szabályozási erőssége fordítottan arányos a C paraméter értékével
- epsilon: keresési sugár, amely belül nem számol hibát az algoritmus

1.1.1.3. K nearest neighbour

- n_neighbors: a figyelembe vett szomszédok száma
- weights: a pontok súlyozási értéke közelség szempontjából
- algorithm: a számítási algoritmus
- leaf_size: a ball_tree és a kd_tree algoritmusoknak adott adatcsomagok mérete
- metric: a pontok közötti távolság számítási módja
- p: a Minkowszky paraméter teljesítmény tényezője, amely 1 vagy 2 értéket vesz fel, majd ennek függvényében az euklideszi, avagy manhattan távolságmutatót alkalmazza.

1.1.1.4. AdaBoost

- base_estimator: az alap algoritmus, amelyet fellendít az AdaBoost (jelen esetben K nearest neighbours regressor)
- n_estimators: az alkalmazott becslő egységek száma
- loss: a hibafüggvény fajtája, amely alapján a súlyozások számítása történik
- learning_rate: Az egyes regresszorokra minden egyes boosting-iterációnál alkalmazott súly, nagysága befolyásolja a tanulási gyorsaságot, ám növeli annak esélyét, hogy a keresett globális minimumot nem találja meg az algoritmus.

1.1.1.5. Multilayer perceptron

- hidden_layer_sizes: az rétegek számának és a bennük található neuronok számának meghatározása
- activation: a rejtett rétegek aktivációs függvénye
- solver: az optimalizáló technikája
- alpha: súlyozási tényező
- learning_rate: a tanulási gyorsaság beállítása, esetünkben adaptív állapotra van állítva, amely a futás közben módosítja az értéket az elért hibacsökkenés csökkenés függvényében

- `batch_size`: az adatokat mekkora csomagméretben adjuk be az algoritmusba
- `max_iter`: a maximum iterációk száma, amelyek során az algoritmus az elejétől kezdve végigfut az egész adatsoron, közben tovább tanulva.
- `early_stopping`: beépített védelmi leállás, ami megakadályozza, hogy háló a túl sok iteráció következtében „overfit” legyen, másnéven nagy pontossággal rátanuljon a tanító adatsorra, ám további adatsorok prediktálására már nem lesz alkalmas.

1.1.1.6. Long short-term memory

Az LSTM esetén a hálóstruktúra módosításával igyekeztünk javítani az előrejelzési pontosságot. A háló rétegeinek felépítése a következő volt az elsőtől az utolsó rétegig sorban:

- 1 dimenziós konvolúció, a konvolúció különböző mintázatok felismerésére és azok kinyerésére alkalmazható matematikai művelet
- LSTM: a korábban leírt LSTM, amelyből 512 algoritmus egység található
- Dense (fully connected) layer 128, ahol 128 neuron található, amelyek rendelkeznek összeköttetéssel az összes szomszédos rétegben található neuronnal.
- Dense (fully connected) layer 32
- Dense (fully connected) layer 1

1.1.2. OPTIMALIZÁLÁSI EREDMÉNYEK

A Hiba! A hivatkozási forrás nem található. c. fejezetben kifejtett optimalizációs logika alapján elvégeztük a szükséges modellezéskét, amelyek az alább prezentált algoritmusok eredményeihez vezettek. Az eredményeket a **1. táblázatban** foglaltuk össze. A táblázatban pár paraméter nem került megjelenítésre, mivel azok konstans értékkel rendelkeztek minden futtatás esetén. Ezen értékek a következők voltak:

A „`random_state`” értékek minden esetben 42-re voltak állítva, így biztosítva a modellek reprodukálhatóságát. Az MLP esetén:

- `solver`: adam
- `learning_rate`: adaptive
- `batch_size`: 32
- `max_iter`: 1000
- `early_stopping`: True

1. táblázat: A hiperparaméter optimalizáció eredményei

	Kelenföld				Csepel				Kispest			
	Tavaszi	Nyári	Ősz	Téli	Tavaszi	Nyári	Ősz	Téli	Tavaszi	Nyári	Ősz	Téli
RF	squared_error	squared_error	squared_error	squared_error	squared_error	squared_error	squared_error	squared_error	squared_error	squared_error	friedman_mse	squared_error
max_depth	40	30	20	30	20	40	20	40	30	30	30	20
max_features	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt
min_samples_split	5	5	5	5	5	4	7	4	5	4	5	7
n_estimators	1000	250	250	250	1250	500	1250	1250	250	250	500	125
C	10	10	10	10	10	1	10	10	1	10	10	10
degree	2	3	2	2	2	2	2	2	2	3	4	2
epsilon	0.1	0.001	0.001	0.001	0.01	0.1	0.1	0.001	0.1	0.1	0.001	0.1
gamma	0.01	0.01	0.1	0.1	0.1	0.1	0.1	0.1	1	1	1	1
kernel	rbf	poly	poly	rbf	rbf	rbf	rbf	rbf	poly	poly	poly	poly
learning_rate	0.1	0.01	0.001	0.001	0.01	0.001	0.001	0.1	0.001	0.1	0.001	0.1
loss	square	exponential	exponential	square	linear	exponential	square	linear	square	linear	exponential	square
n_estimators	20	50	100	100	50	100	50	100	50	100	20	20
KNN	algorithm	ball_tree	ball_tree	ball_tree	ball_tree	ball_tree	ball_tree	ball_tree	ball_tree	brute	ball_tree	ball_tree
leaf_size	7	7	7	7	7	7	7	7	5	5	5	5
metric	manhattan	manhattan	manhattan	manhattan	manhattan	manhattan	manhattan	manhattan	euclidean	cosine	manhattan	manhattan
n_neighbors	11	11	11	11	11	11	11	3	3	3	3	3
weights	uniform	uniform	uniform	distance	uniform	uniform	uniform	distance	uniform	distance	distance	uniform
MLPR	activation	relu	relu	relu	relu	relu	relu	relu	relu	relu	relu	relu
alpha	0.001	0.001	0.0001	0.0001	0.01	0.01	0.01	0.001	0.001	0.01	0.0001	0.0001
batch_size	32	32	32	32	32	32	32	32	32	32	32	32
early_stopping	true	true	true	true	true	true	true	true	true	true	true	true
hidden_layer_sizes	[300, 300, 300]	[500, 500, 500]	[300, 400, 400, 300]	[400, 400, 400]	[300, 400, 400, 300]	[400, 400, 400]	[500, 500, 500]	[300, 300, 300]	[500, 500, 500]	[500, 500, 500]	[300, 300, 300]	[300, 400, 400, 300]