Adnotacja co jest gdzie.
W folderze artykuły - artykuły używane (papery na których się wzorowałem oraz proces kaskadowy)
RSan - implementacja algorytmu R/S z githuba
HurstClassifier.ipynb - notebook z projektem
pliki txt - wygnerowane wcześniej dane

Celem projektu jest odtworznie publikacji dotyczącej obliczania współczynnika Hurst-a, wielkości charakteryzującej głebokość zależności w szeregach czasowych. Wartość współczynnika zawiera się w przedziale (0,1), jest on związany w wymiarm fraktalnym. Jeśli jego wartość dla danego szeregu wynosi (0,0.5) to szereg wykazuje antykorelacje wartości, dla wartości 0.5 jest to proces losowy a dla wartości (0.5,1) wartości wykazują pamięć. Przy pomocy tej charakterystyki na postawie pływów Nilu oszacowano np odpowiednią wysokość tamy na Nilu.
Istnieją różne algorytmy obliczające ten współczynnik, jednymi z popularniejszych są Detrended Fluctuation Analysis (DFA) I Rescaled Range Analysis (R/S).

Postawiony problem to podjęcie próby obliczania ww. Wielkości za pomocą metod ML.
Jak się okazuje drzewa decyzyjne osiągają w tej dziedzinie pewne zadowalające wyniki, w szczególności ich połączena czyli modele typu ensemble jak randomforest . W tym projekcie wykonałem próbę wytrenowania klasyfikatora RandomForestClassifier I RandomForestRegreesor.

Dokładnej wartości współczynnika hursta dla szeregu zazwyczaj nie da się policzyć. Żeby porównać czy wynik jest dobry zadać pewną toleracje, która określa jak daleko oszacowanie może się różnić od teoretycznej wartości aby było uznane za poprawne. W ten sposób możemy zadając toleracje podzielić przedział (0.5,1) na klasy.
Podzielono przedzial na 11 klas wartości współczynników Hursta zaczanając od 0.51 kończąc 0.99 z krokiem co 0.05.

Dane
Potrzebny jest duży zbiór treningowy.
Aby go uzyskać możemy sztucznie wygenerować takie dane za pomocą procesu kaskadowego opisanego w jednym z odnośników (plik cascady.pdf). W dużym skrócie: zaczynamy od jednego punktu x, losujemy wagę w z rozkładu symetrycznego beta z jednym parametrem a i dostajemy dwa punty xw , x(1-w). Powtarzamy na każdym z punktów (za każdym razem losujemy inną wagę). Po n iteracjach dostajemy ostatenicznie szereg czasowy o długości 2^n. Taki szereg czasowy nie posiada wartości ujemnych. Dla zadanego parametru a rozkładu można teoretycznie obliczyć odpowiadający mu współczynnik Hursta.
Program służący do generowania danych znajduje się w pliku gen.py

Trening
Randomforest clasyfikuje dane na podstawie cech obiektu (features). Są dwa sposoby , jedną jest podawanie charaktrystyk szeregu: wartym zauważenia jest fakt że jedną z podawanych charakterystyk jest współczynnik Hursta (prawdopodobnie obliczony przez algorytm) więc klasyfikator jest wzmocnieniem algortymu. Drugim sposobem jest podanie wartości szeregu jako zestaw cech. Trenowałem model drugim sposobem. Dla clasyfikatora RandomForestClassifier jako predykcje podajemy numer przedziału. Dla RandomForestRegressor podajemy po prostu teoretyczną wartość.

Do znalezienia najlepszych parametrów został przeprowadzony grid search oraz ręczne testowanie. Jako test dokładności przerowadzano walidacje krzyżową.

Analiza
obiety typu drzew decyzyjnych posiadają pewne dwie ważne cechy pomagące w jego analizie i ręcznym dopasowywaniu parametrów:

Pierwsze: pojedyńcze drzewo można zwizualizować, na tej podstawie można zobaczyć jak uczy się model. Łatwo zobaczyć, że RandomForestClassfier bardzo zwraca uwagę na to, gdzie w szeregu stoją zera.
Drugie: Model udostępnia statystyki, które opisują jak ważne są konkretne wagi. Po zwizualizowaniu rozkładu dla klasyfikatora RandomForestClassifier widać , że większość wag jest aktywna. Natomiast dla RandomForestRegressor wykres jest wyraznie inny widać pewną dominację pewnych pojedyńczych wag.
Jak było wspomniane szereg czasowy ma tutaj wartości dodatnie. Jak wynika z testów jeśli szereg czasowy wyśrodkować (wtedy potrzeba znowu normalizacji) i na nim trenować model to dokładność spada (być może się da ale potrzebny jest inny zestaw paramatrów).

Ponieżej przedstawiono ostatecznie wyniki dokładności modeli trenowanych na różnych datasetach. Odchylenie standardowe dokładności nie przekracza 0.1 dla wystkich pomiarów (wyciągamy tą statystkę na podstawie crossvalidacji)

Wyniki
Trenowano model dla danych o długości 512 i 4096, po 500 przykładów dla każdej z 11 klas.

Długość danych \Model	RandomForestClassifier	RandomForestRegressor
512	0.82	0.84
4096	0.83	0.92

Drugi model ma pewną przewagę nad pierwszym prawdopodobnie dlatego, że korzysta z informacji że klasy są uszeregowane. Jednak trenuje się znacznie dłużej. W praktyce wydaje się, że podobne zestawy hiperparametór są dobre dla obu modeli, więc pierwszy szybszy nadaje się dobrze do ich szukania i trenowania natomiast drugi prawdopodobnie wytrenuje się lepiej.

Pozostaje pytanie jak model działa na prawdziwych danych , a nie generowanych. Przeprowadzono test dla dwóch szeregów czasowych. Pierwszy pływy nilu, tutaj klasyfikatory podają 10 klasę czyli przedział (0.975,0.99), algorytm R/S zwraca 0.89.
Drugi to fragment datasetu danych pomiarowych jakości powietrza “Non Metanic HydroCarbons concentration”. Wartość policzona przez algorytm wynosi 0.70, wartości oszacowane przez klasyfikatory to odpowiednio

Sugeruje to, że podane dane niekoniecznie dobrze symulują to co chcemy, albo mają pewne własne ukryte właściwości.