Детекција на аномалии во SCADA систем за водоснабдување

Благој Христов
Факултет за електротехника и информациски технологии,
Универзитет "Св. Кирил и Методиј" во Скопје
email: blhris@gmail.com
Предмет: Складишта и обработка на податоци

***Апстракт*—Во сите системи каде се присутни компјутерите за управување и набљудување се изложени на опасност од сајбер напади. Детекцијата на овие напади е од големо значење за операторите на системите, со цел да може да се спречат пред да настане оштетување на компонентите или да дојде до преголема загуба на финансиски ресурси. Во овој труд се разгледуваат повеќе можни пристапи за откривање кога се случуваат овие напади со користење на методи од областа на машинското учење. Користени се седум различни алгоритми, од кои два припаѓаат во групата на надгледувано учење, а останатите пет во групата на ненадгледувано учење. Резултатите покажаа дека поуспешна детекција се добива кога се користи ненадгледано учење, поради големата нерамнотежа во бројот на податоци со напад и без во искористеното податочно множество. Најдобри резултати се добија со користење на *Autoencoder* невронска мрежа.**

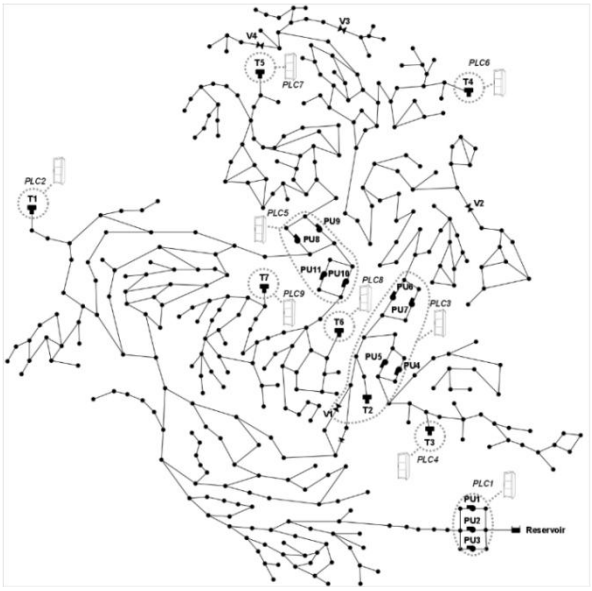
Клучни зборови – детекција на аномалии ; SCADA систем; машинско учење

I. Вовед

Модерните системи за водоснабдување зависат од компјутери, сензори и актуатори за набљудување и управување на операциите. Оваа комбинација од физички процеси и вградени системи значително го подобрува нивото на сервис на водоснабдувачкиот систем, но го изложува на закана од можни сајбер напади. Во последната декада, повеќе системи од овој вид се нападните, што доведе до создавање на агенции за сајбер безбедност и интернационални партнерства со цел да се заштитат водоводните мрежи. Но и покрај овие подвизи, малку е познато за потенцијалните влијанија на овие напади, како и за дизајнот и имплементацијата на алгоритми за детекција на напади, чија цел е да ги идентифицираат аномалиите во отчитувањата од сензорите, пумпите и другите компоненти на водоснабдувачкиот систем.

II. Опис на проблемот

Главната тема на овој труд произлегува од натпревар наменет за пронаоѓање на решение на проблемот со детекција на аномалии одржан во 2016/17 година [1]. *C-Town Public Utility (CPU)* претставува главниот водоснабдувачки систем оператор на *C-Town* (слика 1). Подолг временски период, од повеќе години, оперирањето на дистрибутивниот систем на *CPU* е статичко и без големи промени. Во последната година, се воведува нова *(smart)* технологија која овозможува автоматско собирање на податоци од сензорите во областа и далечинско управување на актуаторите. Кратко по инсталацијата на новата технологија, се набљудува аномално ниско ниво на вода во резервоарот T5 и високо ниво во резервоарот T1. Еден месец подоцна доаѓа до прелевање резервоарот T1. Иако персоналот на *CPU* ги забележаа аномалните отчитувања при првите два настани, прелевањето на T1 е неочекувано бидејќи вредностите на сензорите за нивото на водата се под алармните граници, додека пумпните операции се одвиваат нормално. Поради постоењето на можност за надворешни сајбер напади врз системот, потребно е да се создаде алгоритам кој ќе ги детектира овие напади со што е можно поголема точност и за што е можно пократко време.



Слика 1. Графичка претстава на C-Town водоводната мрежа.

III. Податоци

C-Town е базирана на реална водоводна мрежа со средна големина. Потрошувачката на вода е релативно регуларна низ целата година, без сезонски варијации. Складирањето и дистрибуирањето на водата до побарувачките јазли е овозможено од седум резервоари, чие ниво на вода ја управува операцијата на еден вентил и единаесет пумпи, поделени во пет станици за пумпање *(S1–S5).* Пумпите, вентилите и сензорите за ниво на вода во резервоарите се поврзани со девет *PLC*-а *(Programmable Logic Controller),* кои се поставени во близина на хидрауличните компоненти и се користат за набљудување и управување на системот. *C-Town* има *SCADA (Supervisory Control and Data Acquisition)* систем кој ги собира отчитувањата од сите *PLC*-а и ги координира операциите низ целата мрежа. Во табела 1 е прикажано кои сензори и актуатори се поврзани со секое *PLC*. Поголем дел од *PLC*-ата не се директно поврзани со сензорот за ниво на вода во резервоарите, но ја добиваат потребната информација од останатите *PLC*. Секое *PLC* кое управува одреден актуатор го отчитува и неговиот статус (ON/OFF или OPEN/CLOSED), протокот на вода низ него и притисокот на влезот и излезот од пумпата.

PLC	Сензор	Актуатор
PLC1	-	PU1(T1), PU2(T1)
PLC2	T1	-
PLC3	T2	V2(T2), PU4(T3), PU5(T3), PU6(T4), PU7(T4)
PLC4	T3	-
PLC5	-	PU8(T5), PU9(-), PU10(T7), PU11(T7)
PLC6	T4	-
PLC7	T5	-
PLC8	T6	-
PLC9	T7	-

Табела 1. Сензори и актуатори поврзани со секој PLC

Податочните множества кои се обезбедени се следните:

- ***Dataset03***: Податоци кои датираат една година пред инсталацијата на новата технологија. Овие податоци загарантирано се без напади и можат да се користат за анализа на нормалните операции на системот ***(множество за тренинг).***
- ***Dataset04:*** Овие податоци се земени неколку месеци по воведувањето на *smart* уредите. Содржат обележани напади кои предизвикале аномално ниски нивоа на вода во резервоар T5 и високи нивоа во резервоар T1. Во последниот месец од множеството се наоѓаат отчитувањата на сензорите и актуаторите кога дошло до прелевање на T1. Можно е да содржи и напади кои не се обележани во множеството ***(множество за валидација).***
- ***Test dataset*:** Во ова множество се содржат нови податоци во кои се обележани нападите и тоа треба да се користи за квантитативно оценување на предвидувањето на алгоритмите ***(множество за тестирање).***

Сите податоци се претставени во табеларна форма каде првата колона претставува временска марка кога е направено отчитувањето (датум и час), додека останатите колони претставуваат отчитувања од секој од сензорите. Достапните отчитувања од SCADA системот се следните:

- ниво на вода во секој резервоар;
- статус (0 за OFF/CLOSED, 1 за ON/OPEN) за секоја пумпа и вентил во системот;
- протокот низ секоја пумпа и вентил;
- влезниот и излезниот притисок на секој вентил и станица за пумпање.

Вкупно обезбедени се 43 колони (карактеристики - *features*), а секоја редица од податочните множества претставува еден час и тие содржат:

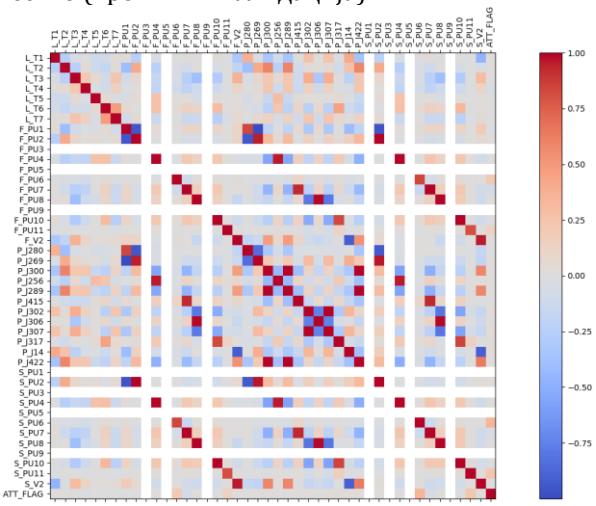
- ***dataset03*** – 8760 часови без напади;
- ***dataset04*** – 4176 часови (88% нормални, 12% напади);
- ***test_dataset*** – 2088 часови (80% нормални, 20% напади).

Бидејќи податоците претставуваат временска низа, може да се искористи *Python* библиотеката *tsfresh* за добивање на голем број дополнителни карактеристики (средна вредност, стандардна девијација, апсолутна енергија, апсолутна промена...). За секој податок, извлечени се нови карактеристики кои ја опишуваат работата на системот во претходните пет часови. Од добиените повеќе од 20,000 карактеристики, избрани се најзначајните 150 и тие се користат при тренирањето на само еден од седумте алгоритми, додека при останатите се користат оригиналните.

Во податочното множество има категорични (бинарни) карактеристики и континуални. Поради ова, потребно е да се изврши нормализација врз континуалните податоците. Нормализацијата означува делење на секој податок со сумата на сите податоци во дадена колона, со цел тие да се ограничат на вредности помеѓу 0 и 1. Исто така, голем дел од алгоритмите за машинско учење многу подобро учат кога податоците кои ги добиваат се стандардизирани. Под стандардизација се подразбира средната вредност на секоја колона да се нагоди да биде 0, додека стандардната девијација да биде 1. Бидејќи стандардизацијата исто ги ограничува податоците во граници околу нулата, за оваа цел е искористена класата *StandardScaler* од делот за препроцесирање на податоци на библиотеката *sklearn*.

IV. FEATURE SELECTION

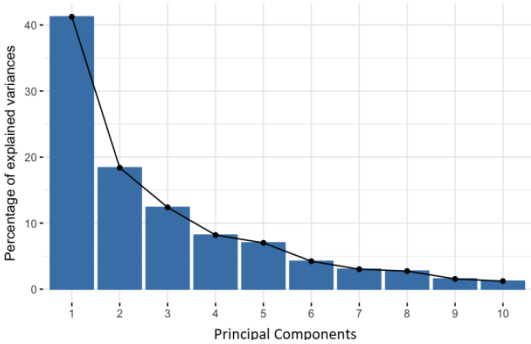
На слика 2 е прикажана корелацијата помеѓу секоја од карактеристиките од вкупното податочно множество (тренинг + валидација):



Слика 2. Корелација на карактеристиките на податочното множество

Со цел да се автоматизира процесот на избирање на оптималниот број на карактеристики за тренирање на алгоритмите, се користи дополнителна метода која се нарекува ***Principal component analysis (PCA)***. Таа претставува постапка за редукција на димензионалноста на големо податочното множество (со многу карактеристики) во помало, без притоа да се загуби голем дел од информацијата. Со губење на дел од информацијата логично би дошло до намалување на точноста на алгоритамот. Но, бидејќи малите податочни множества се многу полесни за анализирање, а најголемиот дел од информацијата се уште се содржи во множеството, во најголем дел од случаевите доаѓа до зголемување на прецизноста на алгоритамот со што се добиваат подобри резултати. Начинот на кој што работи *PCA* е следниот:

1) Стандардизација на податоците со цел сите подеднакво да влијаат врз анализата.
2) Пресметување на матрица на коваријанса. Оваа матрица претставува *nxn* симетрична матрица каде *n* е бројот на карактеристики на податочното множество. Таа ја претставува корелацијата помеѓу секоја од карактеристиките така што +1 означува целосна позитивна корелација (како се зголемува едната вредност за исто толку се зголемува и другата), -1 означува целосна негативна корелација (како се зголемува едната вредност за исто толку се намалува и другата), додека 0 означува дека податоците воопшто не се корелирани.
3) Пресметување на сопствените вектори и сопствените вредности на матрицата на коваријанса за да се одредат т.н. главни компоненти *(principal components)* на податочното множество. Тие претставуваат нови карактеристики кои се создаваат со линеарна комбинација или мешавина од иницијалните. Комбинациите се прават на тој начин што се цели новодобиените карактерситики да бидат некорелирани и најголемиот дел од информацијата од оригиналните карактеристики се втиснува во првите неколку главни компоненти. Односно, доколку имаме иницијално 10-димензионални податоци, *PCA* ќе определи 10 главни компоненти, но во првата компонента ќе смести најголем дел од информацијата што е можно, потоа во втората максимум од останатата информација итн. (слика 3)



Слика 3. Процент на варијансата за секоја главна компонента

Со овој начин на организирање на информацијата во главни компоненти се овозможува редукција на димензионалноста на податочното множество без да се загуби голем дел од информацијата така што се отстрануваат компонентите со мала варијанса, додека оние што ќе останат се земаат за новите карактеристики. Од геометриска гледна точка, главните компоненти ја претставуваат насоката на податоците што обезбедува максимална варијанса, односно, тие претставуваат прави кои опфаќаат најголем дел од податоците. Всушност, овие главни компоненти не претставуваат ништо друго туку сопствените вектори на матрицата на коваријанса, кои ја претставуваат насоката на оските по кои се наоѓа најголема варијанса (најмногу информација), додека сопствените вредности кои се поврзани за секој вектор носат информација за количеството на варијанса која ја носи секоја главна компонента.

Овој метод на избор на карактеристики е искористен во сите алгоритми освен во *Autoencoder* невронската мрежа. Одредувањето на бројот на карактеристики е извршено со *grid search* со тестирање врз множеството на валидација (при ненадгледувано учење) или при дел од целосното множество (при надгледувано учење) со користење на *Stratified K-Fold cross validation (K = 10)*.

V. Алгоритми

Како што беше наведено погоре, во овој труд се користат седум различни алгоритми за класификација за решавање на овој проблем. Од групата на надгледувано учење се користат *Support Vector Classifier* и *K-Nearest Neighbor (kNN)*, додека од ненадгледувано учење се користат *One-Class Support Vector Machine, K-Means Clustering, Multivariate Gaussian Distribution, Isolation Forest* и *Autoencoder Neural Network.*

За одредување на оптималниот број на карактеристики и вредности за параметрите на алгоритмите се користи *Stratified K-Fold cross validation.* Стандардниот *K-Fold* го поделува множеството на податоци на *K* еднакви делови од кои *K-1* се користат за тренирање а останатиот дел за тестирање. Процесот се повторува *K* пати, се додека секој од деловите не е искористен за тестирање барем еднаш. Надополнување на овој метод е со користење на *stratified* метода. Идејата на поделбата е иста, но сега дополнително на алгоритамот се испраќаат и класните лабели, со цел секој од деловите да содржи ист процент од класите. Поради големата нерамнотежа на класите во дадените податочни множества, неопходно е да се користи овој метод за да се обезбеди поголема точност од учењето.

V-1. Supervised

1) K-Nearest Neighbor (kNN)

Како *baseline* се користи наједноставен kNN класификатор. Тој работи на принцип што одредува во која класа припаѓаат најблиските *K* соседи на новиот податок и според мнозинско гласање ја одредува неговата класа. Со користење на *Stratified K-Fold* и *grid search* врз податочното множество сочинето од *dataset03* и *dataset04* се одредува оптималниот број на карактеристики кои се користат при тренирање на алгоритамот, како и бројот на соседи *K*. Најдобри резултати се добија со користење на 28 од 43те карактеристики и *K = 3* соседи.

2) Support Vector Classifier (SVC)

Втората надгледувана метода која се користи е *C-Support Vector Machine* или *SVM*. Таа работи слично на најобична логистичка регресија, но правата или хипер-рамнината која ги дели и одредува на која класа припаѓаат податоците се обидува да ја нагоди да биде што е можно поточно поставена (подеднакво оддалечена од двете класи). Во случај кога поделбата не може да се одреди линеарно, може да се користат и други кернели како сигмоидален или радијална основна функција *(RBF)*. За решавање на конкретниот проблем, најдобри резултати даде *RBF* кернелот, со користење на 31 од 43те карактеристики.

V-2. Unsupervised

1) K-Means Clustering

Еден од најосновните ненадгледувани методи е *K-Means* кластерирање на податоците. Неговото учење се почнува на тој начин што се иницијализираат *K* случајни центроиди, каде *K* е бројот на можни класи. Се пресметува Евклидово растојание од секој податок во множеството до центроидите и се одредува кој каде припаѓа според минимум од двете растојанија. Откако ќе се поделат податоците, се пресметува средна вредност на сите податоци во секој од добиените кластери. Секој центроид се преместува во соодветната новопресметана средна положба и постапката се повторува. Бидејќи пресметките кои се потребни за извршување на алгоритамот не се временски скапи, бројот на итерации до завршување на алгоритамот може да биде поголем, а исто така може и повеќекратно да се изврши со друга случајна иницијализација на центроидите. Тренирањето на алгоритамот се изврши врз оригиналните карактеристики, а најдобри резултати се добија со 32 *features*, 500 итерации и 20 случајни иницијализации.

2) One-Class SVM

Како што насочува и името, овој метод претставува модификација на надгледуваниот *SVM* алгоритам, но разликата е во тоа што се тренира врз податоци кои припаѓаат на една класа, со цел потоа да може да детектира кои податоци од тест множеството припаѓаат во таа класа, а кои не - *outliers* или аномалии. Бидејќи класната лабела на сите податоци при тренингот е иста и таа не се користи, овој метод припаѓа во групата на ненадгледувано учење. Хиперпараметри кај овој алгоритам се γ и ν (гама и ну). Ну ја одредува горната граница на делот од маргиналната грешка и долната граница на делот на "помошни" (support) вектори релативно на бројот на тренинг податоци. При детекција на аномалии, со ну може да се постави колкав процент од податоците претставуваат потенцијални аномалии. За најдоброто решение, ну е поставено на 0.2, гама е автоматски одредена од алгоритамот, избраниот број на карактеристики за тренирање е 6, а кернелот кој се користи е повторно *RBF*. Тренирањето на алгоритамот се одвива само на тренинг множеството, додека множеството за валидација (повторно со *K-Fold*) се користи за одредување на бројот на *features.*

3) Multivariate Gaussian Distribution

Овој метод за детекција на аномалии е базиран врз градење на Гаусова распределба од оригиналните тренинг податоци без напади така што се добива нормалната распределба на податоците при нормален режим на работа. Бидејќи Гаусовата распределба е дефинирана само од два параметри, потребно е само

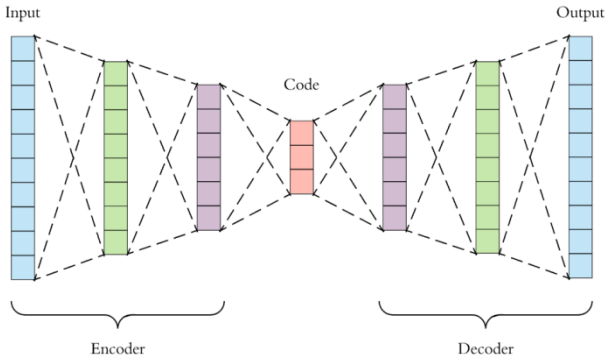
да се пресметаат средната вредност - μ и варијансата на податочното множество. Поради тоа што множеството е мултидимензионално, наместо варијансата се пресметува матрицата на коваријанса - Σ. Потоа, се пресметува рангот на оваа матрица и се отстрануваат сите линеарно зависни колони, со цел таа да не биде сингуларна. Со користење на класата *multivariate_normal* од библиотеката *scipy* и со повик на функцијата *pdf( )* врз множеството за валидација, со пресметаните μ и Σ, се пресметува густината на веројатност на податоците за валидација. Идејата на овој алгоритам е со пресметување на оваа густина да се определи дали конкретниот податок би припаѓал во одредена ε-околина во која се наоѓаат останатите податоци при нормалниот режим на работа, или е надвор од таа околина и е аномалија. Ширината на ε-околината се одредува со пресметување на густината на веројатност на множеството на валидација така што да се постигне што е можно поголема точност. Конечниот резултат се добива со пресметување на густина на веројатност на тест множеството со претходно одреденото ε, а најдобар резултат се доби со користење на 19 од 43те карактеристики.

4) Isolation Forest

Ensemble алгоритмите користат повеќе алгоритми за учење заедно како една целина со цел да им се подобри предвидувачката способност, што не би можело да се постигне доколку се користат поединечно. Еден таков ансамбл алгоритам е *Isolation Forest* или *IS* и тој користи повеќе *Isolation Trees* (дрва на одлучување). Најчесто овој метод се користи за детекција на аномалии така што аномалните податоци би имале пократки просечни должини на патеки по дрвата на одлучување отколку нормалните податоци. Изолационите дрва кои се користат се многу слични на обични пребарувачки бинарни дрва кои имаат по две гранки поврзани за секој јазел. Една клучна особина на овој тип на дрва е тоа што при случајно генерирани податоци, патеката помеѓу коренот и аномалните податоци најчесто ќе биде пократка, и таа особина се користи кај изолационите дрва. Изолационата шума се добива со поврзување на повеќе изолациони дрва , со што значително се зголемува точноста на алгоритамот. Бидејќи целта на методот е да ги пронајде аномалните податоци, тренирањето и тестирањето се вршат на истото податочно множество. Параметрите се одредуваат со користење на множествата за тренинг и валидација, додека конечната точност се базира на множеството за тест. За добивање на најдобрите резултати искористени се 25 карактеристики, 51 дрва и претпоставено е дека постојат 16% аномалии во податоците.

5) Autoencoder Neural Network

Autoencoder претставува вид на невронска мрежа која се користи за учење на клучни карактеристики на податоците на ненадгледан начин. Идејата зад неговото работење е тоа што тој создава репрезентација или *encoding* на множество од податоци со користење на димензиона редукција и со игнорирање на шумот присутен во податоците. Покрај редукциониот дел, се учи и дел за реконструкција каде невронската мрежа се обидува редуцираните податоци да ги реконструира што е можно поблиску до оригиналните. Изгледот на една типична *Autoencoder* мрежа е даден на слика 4.



Слика 4. Autoencoder Neural Network

Наједноставниот модел на *Autoencoder* е *feedforward* нерекурентна мрежа, слична на повеќеслоен перцептрон и содржи влезен слој, излезен слој и еден или повеќе скриени слоеви. Влезниот и излезниот слој имаат ист број на неврони, додека скриените слоеви имаат скалеста форма на тој начин што колку подлабоко се навлегува во мрежата толку е помал бројот на неврони. Со енкодирање на информацијата од оригиналните податоци се добива пресликување ***z*** кое се нарекува код или латентна репрезентација и тој се "наоѓа" во средината на мрежата. Декодирањето на овој код се врши со слоевите десно од него со што на излезот се добива реконструираната информација. Најчесто овие мрежи се симетрични во однос на кодот, односно секој слој од енкодерот има ист број на неврони со соодветниот слој од декодер.

Градбата на добиената оптимална мрежа се состои од влезен слој со 179 неврони (број на карактеристики од податочното множество добиено со *tsfresh*), два слоеви во енкодерот со по 80% и 60% од невроните од влезниот слој соодветно, два слоеви во декодерот со по 60% и 80% од невроните во влезниот слој соодветно и излезен слој со ист број на неврони како влезниот. За активациска функција на невроните се искористени хиперболична *tanh* функција за надворешните скриени слоеви и *ReLU* за внатрешните скриени слоеви. Тренирањето се одвива во 300 епохи со големина на еден *batch* од 256. *Batch* претставува колку податоци се процесираат пред моделот да се

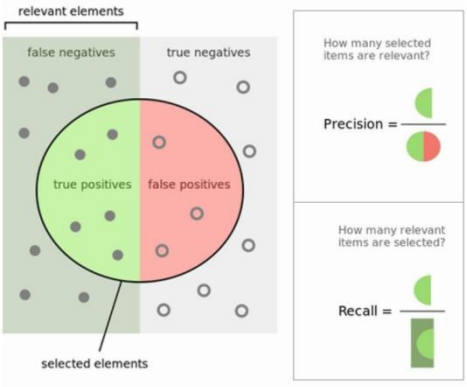
ажурира. За оптимизација на мрежата се користи *Adam* оптимизациски алгоритам, а за метрика на евалуација се користи средна квадратна грешка (MSE):

$$MSE = \frac{1}{n}\sum_{1}^{n}\left(Y_{(n)\ estimated} - Y_{(n)\ true}\right)^2 \quad (1)$$

Тренирањето на мрежата се извршува врз податочното множество без напади со цел да научи како треба да се однесуваат податоците при нормален режим на работа, за полесно да може да се детектираат аномалии. Прагот кој одредува во која класа припаѓаат податоците се одредува со проба, со тестирање на мрежата врз множеството на валидација. За најдобри резултати се добива дека прагот треба да изнесува 0.84. Добиеното решение со овој метод даде најдобри резултати од сите останати алгоритми.

VI. Метрика на оценување

Определување на точноста на алгоритмите според добро познатите метрики како средна квадратна грешка, средна апсолутна грешка и сл. дава само резултати во однос на бројот на припадност на податоците за тест во секоја од класите во споредба со вистинскиот број. Ова може да доведе до добивање на поголема точност на алгоритамот од вистинската поради постоењето на лажно позитивни и лажно негативни примероци. За таа цел во овој труд е искористена метриката F1 (слика 5).



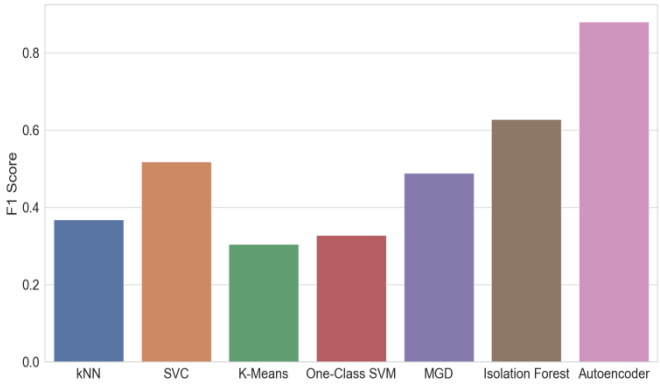
Слика 5. Графички приказ на F1 score

Принципот на оваа метрика е одредувањето на бројот на примероци кои се вистински позитивни, вистински негативни, лажно позитивни и лажно негативни и потоа одредување на *precision* и *recall*. *Precision* претставува колку од примероците класифицирани како позитивни се релевантни, односно, точно предвидени. *Recall* претставува колку од релевантните примероци, односно, вистински позитивните примероци се класифицирани како позитивни. F1 метриката е значително подобра во однос на други метрики за прецизност и таа претставува хармониска средина на *precision* и *recall*:

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

VII. Резултати

Конечните F1 резултати од сите алгоритми се дадени на графикот претставен на слика 6:



Слика 6. Споредба на F1 score за секој алгоритам

Од графикот може да се забележи дека најдобри резултати се даде *Autoencoder* невронската мрежа, со точност од 88.09%. Исто така се забележува дека *One-Class SVM* даде полоши резултати во однос на *SVC* иако е наменет за решавање на проблеми од овој тип. Тоа се должи на фактот што не е извршен *grid search* за одредување на оптимални вредности за сите параметри, туку само за бројот на карактеристики.

VIII. Заклучок и идна работа

Во овој труд се разгледаа седум можни пристапи за решавање на проблемот на детекција на аномалии во нерамнотежно распределено множество од податоци. Најдобриот модел беше *Autoencoder* невронската мрежа која успеа да ги детектира речиси сите напади, со многу мал дел на лажно позитивни примероци. Поголемиот дел од алгоритмите не дадоа добри резултати поради небалансираното множество или поради недоволна оптиматизација на нивните параметри. За во иднина, би можело сите да се подобрат со дополнителен *grid search* за определување на параметрите што се одредени автоматски од самиот алгоритам и со користење на разни ансамбл методи како stacking и boosting за подобрување на целокупната точност.

Референци

[1] Taormina, R., Galelli, S., Tippenhauer, N. O., Salomons, E., Ostfeld, A., Eliades, D. G., и др. (2018). The Battle Of The Attack Detection Algorithms: Disclosing Cyber Attacks On Water Distribution Networks. *Journal of Water Resources Planning and Management , 144*, 04018048.