České vysoké učení technické v Praze
Fakulta jaderná a fyzikálně inženýrská

Katedra softwarového inženýrství v ekonomii
Obor: Inženýrská informatika
Zaměření: Softwarové inženýrství v ekonomii

Modely zátěže výpočetních serverů

Computational requirements modelling

BAKALÁŘSKÁ PRÁCE

Vypracoval: Dmitriy Burdin
Vedoucí práce: Ing. Jan Doubek
Rok: Červenec 2015

Před svázáním místo téhle stránky vložíte zadání práce s podpisem děkana a do pdf verze oskenované zadání.

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne ....

Dmitriy Burdin

Poděkování

Děkuji Ing. Janu Doubkovi za vedení mé bakalářské práce a za podnětné návrhy, které ji obohatily.

Dmitriy Burdin

Název práce:
Modely zátěže výpočetních serverů

Autor: Dmitriy Burdin

Obor: Inženýrská informatika

Druh práce: BAKALÁŘSKÁ PRÁCE

Vedoucí práce: Ing. Jan Doubek
CISCO Systems s.r.o.

Abstrakt: Cílem této práce je efektivně modelovat časovou, výpočetní a paměťovou náročnost distribuovaných úloh. Implementace modelů na základě časových řad výsledků minulých úloh. Výsledkem práce bude studie použitelnosti ekonometrických metod v prostředí velkých data center. Dále návrh algoritmů pro implementaci studovaných metod. Výstupní modely budou použity pro lepší plánování rozvržení výpočetních zdrojů.

Klíčová slova: Časová řada, ekonometrie, predikce, analýza, server

Title:
Computational requirements modelling

Author: Dmitriy Burdin

Abstract: The goal of this thesis is to effectively make a model of temporal, computational and memory requirements of distributed tasks. Model implementation based on time series results of past tasks. The result of the thesis will be a study of applicability of econometric methods in large scale data centres. In addition, design of algorithms for implementation of the studied methods. Output models will be used for better planning of computational sources arrangement.

Key words: Times series, econometrics, forecasting, analysis, server

Obsah

Úvod

Obrovské a rostoucí množství informací zaplavují dnešní podniky. To se stává hlavně kvůli rostoucímu počtu lidí, firem a zařízení připojených k internetu. Kolem třetiny světové populace má dnes přístup k internetu. Velké množství informace, jinak řečeno - velká data, představuje rychle rostoucí sféru, ve které se pro maximální efektivitu vyžaduje vhodná kombinace softwaru, hardwaru a navíc speciální úpravy s ohledem na oblast působnosti. Tyto fakty vedou k výraznému zaměření na velká data a na nové metody správy a analýzování tohoto proudu informací.

Důležitým prvkem řešení této problematiky je datacentrum. Jsou to specializované prostory pro umístění a zajištění stabilního provozu výpočetní a serverové techniky. Většinou se tato technika skládá z počítačových clusterů, což je několik spolupracujících počítačů propojených počítačovou sítí. Clustery se využívají pro výpočet komplikovaných početních úloh. Jeden z hlavních parametrů kvalitních datacenter je správné planování distribuovaných úloh pro rychlé zpracování dat. Při stále se zvětšujícím množstvím dat se tento problém komplikuje. Proto je potřeba vědět v jakém pořadí zpracovávat určité úlohy a jak dlouho to bude trvat.

Cílem této bakalářské práce bylo navrhnout a vytvořit softwarovou aplikaci pro analytické zpracování dat, se kterými pracjuí výpočetní servery. Podstatou aplikace je predikce budoucího chování určitých hodnont na základě analýzy minulých výsledků. Protože velká data obvykle obsahují dynamické systémy dat, které se mění s časem, zkoumaná data se předpokládají být ve formě časových řad. Proto se v této práci nejdřív věnuje způsobům zpracování, analýzy a predikce časových řad. Poté následjue popis algoritmu používání vybraných postupů pro vhodnou analýzu a implementace příslušného algoritmu. Na závěr je demonstrováno využití aplikace na příkladových datech.

1. Teoretická část

1.1. Časové řady

1.1.1. Úvod a charakteristika

Jak již bylo zmíněno, časové řady jsou základním zkoumaným prvkem při analýze různých dynamických systémů obsahující chronologicky uspořádaná data. Časové řady jsou vlastně soubory jednoznačně uspořádáných podle času pozorování v příslušném systému. Data ve formě časových řad vznikají v úplně různých odvětvích bud' to fyzikální věda, biologie, společenská věda, medicína atd.

Například ve společenských vědách, časové řady jsou užitečné při popsání počtu obyvatel, porodnosti, nemocnosti. V ekonomii teorie časových řad je jednou z nejdůležitějších metod při analýze ekonomických procesů. Mohou popisovat vývoj určitého ukazatele jako objem výroby, produktivita práce, nezaměstnanost nebo spotřeba surovin. V technice časové řady mohou představovat průběh signálu, spolehlivost nebo intenzitu zatížení elektrického zařízení.

Pro lepší porozumění určitého mechanismu nebo procesu, popsaného časovou řadou, existuje analýza časových řad skládající se z různých metod. Tyto metody pomáhají vytvořit vhodný model popisující chování pozorovaných hodnot. Znalost takového modelu umožňuje kontrolovat činnost a sledovat vývoj určitého systému. Jako důsledek spravné provedené analýzy stává se možným predikovat budoucí chování systému.

Časové řady se skládají ze dvou prvků:

- časový úsek, během kterého byly udělány pozorování
- hodnoty příslušných ukazatelů časové řady

Podle časového úseku časové řady se člení do dvou typů. Jedním je okamžiková časová řada, jejíž pozorování jsou naměřena v jisté časové okamžiky a sčítání hodnot příslušných pozorování nedává smysl. Příkladem může být řada udávající počet zaměstnanců ve firmě na začátku roku. Druhým typem je intervalová časová řada. Pozorování v intervalových časových řadách jsou závislé na délce časového intervalu sledování a v tomto případě už sčítání hodnot ukazatele časové řady dává smysl, nebot' hodnotu ukazatele za větší interval lze získat sčítáním hodnot za jednotlivé části příslušného intervalu. Například porodnost ve státě za rok.

Časové úseky jsou obvykle stejnoměrně rozděleny, což znamená, že mezi jednotlivými pozorování stejné časové intervaly. V opačném případě pozorování jsou rozděleny různorodě a tím se analýza časových řad komplikuje.

Obvykle rozlišujeme dva základní modely časových řad:

- Determenistický - model, ve kterém časová řada nemá náhodné prvky a je generována známou matematickou funkcí. Důsledkem je srovnatelně jednoduchá analýza časové řady.
- Stochastický - model popisuje náhodný proces a časová řada obsahuje náhodný prvek. Většina běžně se vyskytujících časových řad v praxi mají stochastické modely.

Jedna z dalších důležitých charakteristik časových řad, vyplývající ze stochastického modelu, je jejich stacionarita, případně nestacionarita. Při stacionaritě střední hodnota a rozptyl časové řady se v čase nemění, což nelze říct o nestacionárních časových řadách, ve kterých se objevují změny ve střední hodnotě či rozptylu. Říká se proto, že nestacionární časové řady mají určitý trend vývoje. I když stacionarita je běžným předpokladem většiny metod analýzy časových řad, některé modely se omezují na modely stacionárních procesů.

1.1.2. Analýza časových řad

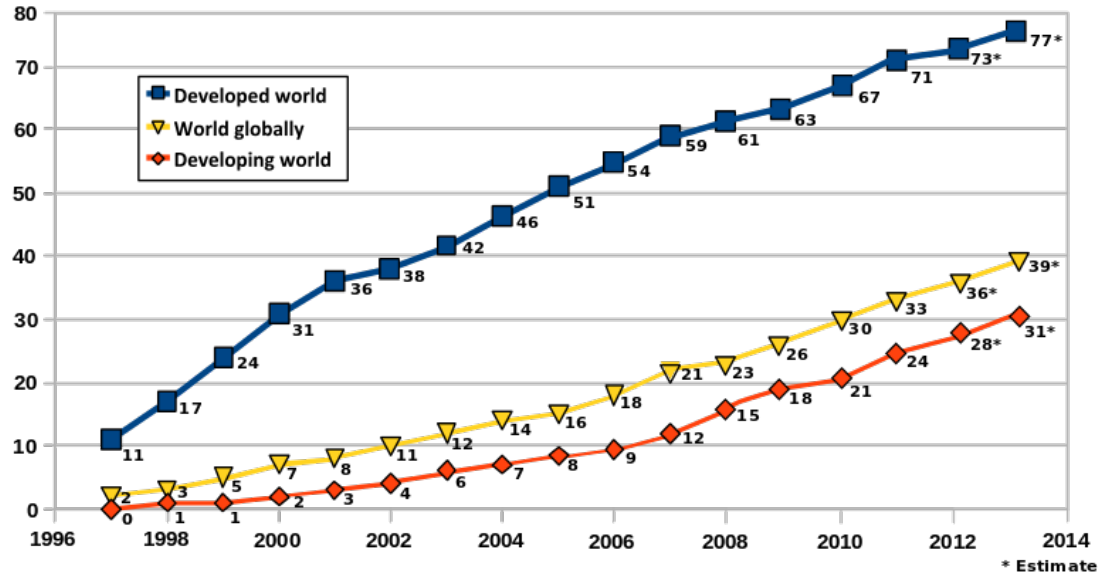
Grafická analýza časových řad

Jedním ze základních prostředků prezentace časových řad je jejich graf. Nejčastěji se graficky znázorňují původní hodnoty časové řady, nebo kumulativní časové řady, které vznikají postupným načítáním (kumulováním) jednotlivých hodnot (u okamžikových časových řad nemají smysl, neboť výše jejich hodnot nezávisí na daném časovém intervalu). Často se ale časové řady zobrazují tak, aby více vynikly jejich charakteristické vlastnosti a rysy. K tomu slouží speciální typy grafů jako jsou například spojnicové grafy jedné nebo více časových řad, krabičkové grafy, grafy sezónních hodnot.

Prvotní informace pro analýzu časových řad získáme ze **spojnicových grafů**. Jejich princip spočívá v zakreslení jednotlivých hodnot časové řady do souřadných os, na kterých jsou vyznačeny příslušné stupnice. Na osu horizontální se vynáší časová proměnná a na osu vertikální hodnoty časové řady nebo její funkce.

Do spojnicového grafu můžeme zakreslit i více časových řad. V případě, že zobrazujeme např. dvě časové řady lišící se měřítkem, je možné použít kromě levé i pravou vertikální osu.

V některých případech je užitečné provést detailnější pohled na časovou řadu. *Krabičkový graf* na rozdíl od jiných grafů obsahuje souhrnné charakteristiky zkoumané časové řady. Tento graf umožní odhalit některé důležité vlastnosti řady, které z jiných grafů nejsou zřetelné. Jeho základním prvkem je krabička, jejíž dolní a horní hrana je tvořena 25% a 75% kvartilem, uvnitř je vyznačen medián a symbolem „+“ aritmetický průměr. Na koncích svislých čar vycházejících z krabičky leží hodnoty minima a maxima. Protože délka této svislé čáry může být maximálně 1,5x delší než krabička, jsou hodnoty přesahující tyto hranice označovány jako odlehlé a jsou zakresleny jako samostatné body.



Obrázek 1.1.: Počet uživatelů Internetu na 100 obyvatel ve světě.
(Příklad spojnicového grafu se třemi časovými řadami)

Dekompozice časových řad

Abychom mohli s řadou pracovat, ve smyslu něco předpovídat, dělat odhady (ať už intervalové či bodové) apod., potřebujeme mít alespoň stacionární časovou řadu. Respektivě alespoň slabě stacionární časovou řadu.

Většina pozorovaných časových řad není řadou nezávislých stejně rozdělených náhodných veličin ani stacionární časovou řadou. Ale mnoho kdy se nám může podařit vhodným očistěním řady, získat stacionární řadu. Eliminace jednotlivých složek časové řady, která se při dekompozici provádí, může mít různé cíle:

1. Zajímavé poznatky může přinést samotné studium oddělených (eliminovaných) složek časové řady, nebot' tak lze objevit některé zákonitosti chování sledované řady, rozpoznat vnější vlivy působící na její průběh a provést účinné srovnání průběhu několika časových řad. Např. pomocí trendové složky lze posuzovat tempo rozvoje dvou příbuzných průmzslových odvětví nebo pomocí sezónní složky lze posoudit průběh poptávky po určitém druhu zboží během roku.
2. Podle bodu 1 je úkolem dekompozice časové řady proniknout hlouběji do podstaty historického průběhu řady. Neméně důležitým cílem dekompozice je ale také cíl extrapolační, kdy nás zajímá budoucí vývoj jednotlivých složek časové řady (např. jaké bude budoucí tempo růstu letecké přepravy) a nebo konstruujeme předpověď' v celé časové řadě tak, že ji složíme z předpovědí v jednotlivých složkách, které se obvykle sestrojí poměrně jednoduše a přesně.
3. Často je vzhledem k podstatě řešeného problému výhodné znát chování časové řady "očištěné"od některých jejich složek. V ekonomických časových řadách se například velice často provádí tzv. sezónní očišťování, kdy z řady odstraňjeme sezónní fluktuace.

Klasická analýza ekonomických časových řad vychází z předpokladu, že časovou řadu y_t pro t = 1, 2, ... , T je možné rozložit na čtyři složky: trendovou, cyklickou, sezónní a nesystematickou.

Trendová složka (T_t) vyjadřuje dlouhodobou tendenci vývoje zkoumaného jevu. Je výsledkem faktorů, které dlouhodobě působí stejným směrem např. technologie výroby, demografické podmínky, podmínky na trhu apod.

Cyklická složka (C_t) vyjadřuje kolísání okolo trendu, ve kterém se střídají fáze růstu a poklesu. Jednotlivé cykly (periody) se vytvářejí za období delší než jeden rok a mají nepravidelný charakter, tj. různou délku a amplitudu. Cykly jsou v ekonomických časových řadách způsobeny ekonomickými a neekonomickými faktory. V posledních letech se věnuje pozornost zejména technologickým, inovačním či demografickým cyklům.

Sezónní složka (S_t) vyjadřuje pravidelné kolísání okolo trendu v rámci kalendářního roku. Sezónní výkyvy se opakují každoročně ve stejných obdobích (délka periody je jeden rok) a vznikají v důsledku střídání ročních období nebo vlivem různých institucionalizovaných zvyků, jako jsou např. svátky, dovolené apod.

Poslední složkou časové řady je *nesystematická složka* (I_t nebo a_t). Tato složka vyjadřuje nahodilé a jiné nesystematické výkyvy, ale také chyby měření apod.

Dekompozice časové řady může být:

1. aditivní, hodnoty časové řady se dají určit jako součet hodnot jednotlivých složek, tj.
$$y_t = T_t + C_t + S_t + I_t \tag{1.1}$$

2. multiplikativní, hodnoty časové řady se dají určit jako součin hodnot jednotlivých složek
$$y_t = T_t \cdot C_t \cdot S_t \cdot I_t \tag{1.2}$$

Po aditivní dekompozici jsou jednotlivé složky časové řady ve stejných měrných jednotkách jako původní časová řada. Aditivní dekompozice se používá v případě, že variabilita hodnot časové řady je přibližně konstantní v čase.

Po multiplikativní dekompozici je trendová složka časové řady ve stejných měrných jednotkách jako původní časová řada, ale ostatní složky časové řady (cyklická, sezónní a nesystematická) jsou v relativním vyjádření. Multiplikativní dekompozice se používá v případě, že variabilita časové řady roste v čase, nebo se v čase mění. V praxi se dekompozice časových řad často používá z těchto důvodů:

a) analýzou jednotlivých složek řady lze odhalit určité zákonitosti vývoje zkoumaného jevu,

b) časové řady je možné očistit od sezónnosti, tj. z časové řady se odstraní sezónní složka, což umožňuje porovnávat trend několika časových řad současně,

c) časové řady lze očistit od trendu, tj. z řady se odstraní trendová složka, což umožňuje lépe modelovat sezónnost, protože charakter sezónnosti je výraznější,

d) často umožňuje přesněji určit předpovědi nejen jednotlivých složek časové řady, ale v konečném důsledku také samotné časové řady, v tom smyslu, že předpovědi jednotlivých složek se sečtou anebo vynásobí podle toho, který typ dekompozice jsme použili.

1.1.3. Predikce

1.2. Regresní analýza

1.2.1. Úvod

1.2.2. Lineární regresní model

1.2.3. Nelineární regresní model

1.3. Softwarová aplikace

1.3.1. Způsob realizace

Po důkladném seznamením s problematikou práce a se související s ní teorií byl stanoven způsob dosažení cílů práce, tj. způsob její praktické realizace. Bylo navrhnuto použít již existující nebo vytvořit svou knihovnu v jazyce Java pro implementaci metod regresní analýzy. Pomocí této analýzy zjistit určité parametry v datech, které nejméně nebo nejvíc ovlivňují ostatní parametry, a jestli mezi sebou vůbec mají nějakou souvislost. Potom na základě těchto údajů, s úvahou o výsledcích regresní analýzy udělat nejlepší predikci budoucích hodnot určitého parametru. V daném případě jde o parametr *Computation time* (z angl. doba trvání výpočtu), jelikož tento parametr vyvolává zatížení, snížení kterého je vlastně úkolem této práce.

mozna jeste obrazek, ukazujici tento proces realizace. 1 - 2 - 3 -...

1.3.2. Základní předpoklady a nástroje

Softwarová aplikace, která by měla být užitečnou pomůckou pro dosažení cíle práce, může být také považována za výsledek této práce. Její podstatou je možnost nejdřív načíst datový soubor, poté ho zanalyzovat regresní metodou, určit potřebnou informaci ze souboru, použít dosažené výsledky pro predikci budoucich hodnot a vykreslit vývoj nejlépe predikovaných hodnot ve tvaru grafu časových řad.

Jako programovací nástroj jsem vybral objektově orientovaný programovací jazyk *Java*. Jedná se o populární a docela starý programovací jazyk jehož historie sahá do roku 1990, kdy ve firmě Sun Microsystems začali pracovat na jeho vytvoření. Na začátku to byl efektivní a jednoduchý jazyk určený pro spotřební elektroniku. S postupem času a s rostoucím užitím internetu Java se stal jedním z nejpoužívanějších programovacích jazyků ve světě. Hlavními důvody proč jsem zvolil pravě tento jazyk jsou:

- výkonnost a jednoduchost syntaxe
- široká použitelnost
- přenositelnost a nezávislost na architektuře nebo na operačním systému

Pro snadnější programatorskou práci se hojně využívá *vývojové prostředí* neboli *IDE* (IDE - *Integrated Development Environment*). Je to vlastně soubor nástrojů pro přehlednější přístup k zdrojovému kódu, ladění programů, hledání a opravu chyb a další užitečné věci pro programování. Skládá se vývojové prostředí většinou z textového editoru, kompilátoru a debuggeru.

Pro programovací jazyk Java se používá několik různých IDE. Liší se podle nabídnutých vlastností a možností. Pro svou práci jsem zvolil vývojové prostředí *IntelliJ IDEA*.

Pro lepší koordinaci se svým vedoucím během práce použil jsem webový nástroj *GitHub*. Je to služba neboli prostředek pro spolupraci vývojářů, kteři používají verzovací nástroj Git. Využívá se pro sdílení a společnou práci na softwarových projek-

tech. Tato služba poskytuje možnost sdílet, upravovat projekty jiných programatorů, což vede k lepším výsledkům a rychlejšímu ladění kódu.

2. Praktická část

2.1. Chybné pokusy

Při výběru programovacích nástrojů byla uvažována možnost využití cizích, svobodně distribuovaných knihoven nebo frameworků vhodných pro práci s daty a případně pro jejich analýzu. Takové pomocné nástroje se ovšem předpokladaly být použitelny spolu s programovacím jazykem Java.

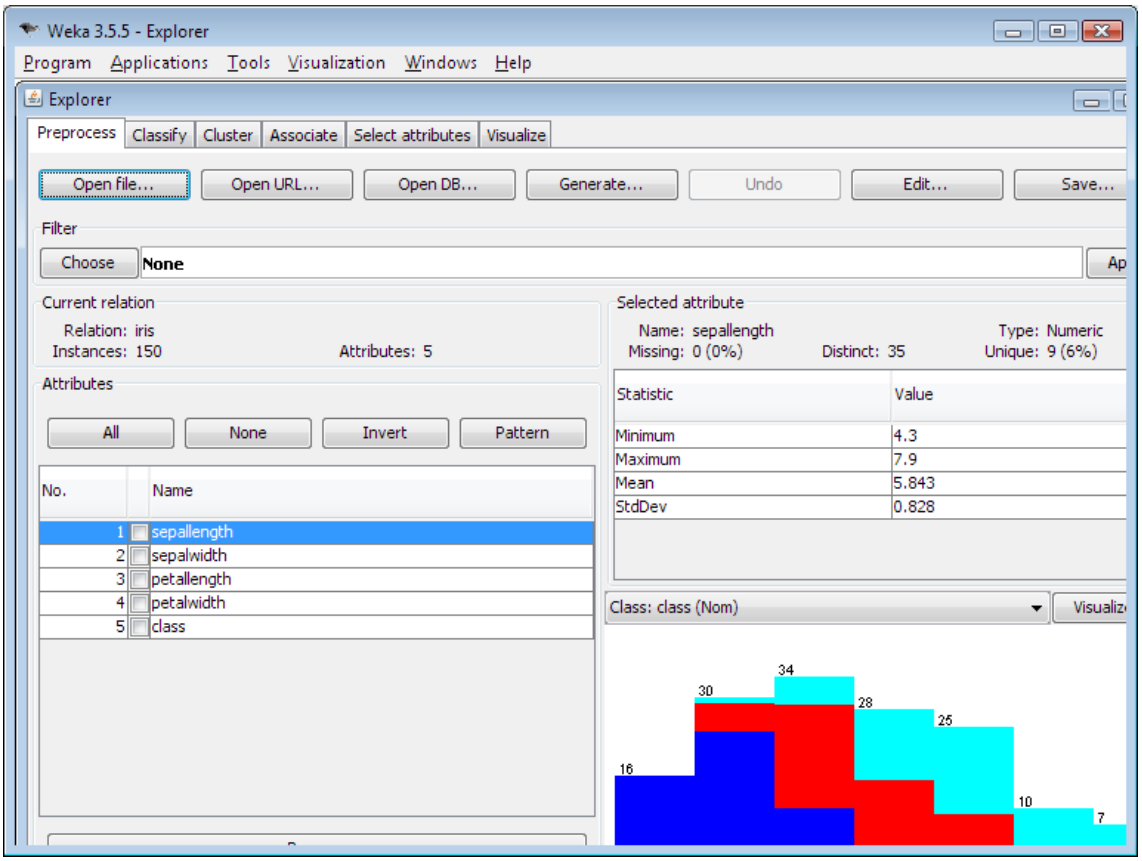
Knihovna je v programování vlastně soubor funkcí, procedur, datových typů a tříd, které slouží pro různé programy. Každá knihovna má své *API* (z angl. *application programming interface* - rozhraní pro programování aplikací). Pomocí něho se určuje jakým způsobem jsou funkce knihovny volány. Zatímco framework je taková softwarová struktura, která oproti knivoně volá samotný kód uživatele. Framework může obsahovat jak své procedury, podprogramy tak i knihovny. Protože framework má svou určitou architekturu, má to v závislosti na případě dobrou i špatnou stránku.

Během hledání pomocných nástrojů neboli vhodných knihoven našel jsem dva zdroje, které jak jsem si myslel by mohly zlepšit průběh práce. Po hlubším zkoumání však nebyly tak vhodné než se zdálo na první pohled. Jsou to velká sbírka algoritmů pro práci s daty *Weka* a framework *Encog*.

Weka

Weka je docela populární soubor neboli balík programů, který slouží pro analýzu dat, modelování predikce a využívá se v strojovém učení. Pro práci se používají textové soubory s formátem ARFF (*Attribute Relationship File Format*), který obsahuje informaci ve tvaru atributů a jim příslušná data. Weka má i grafické uživatelské rozhrání pro přehledný přístup k využití všech možností programu, což je velkou výhodou. Zaujalo mě taky to, že pomocí tohoto balíku by šlo snadno zvizualizovat výsledky analýzy. Na to Weka má vhodné nástroje a může proto vykreslit graf časové řady s veškerou potřebnou informaci o její vývoje.

Tato sbírka nástrojů pro analýzu dat je napsaná v jazyce Java a proto jsem se rozhodl ji zkusit implementovat a využít jeji možnosti ve svém programu. Narazil jsem ale na jeden problém ze samého začátku. Při konverzi CSV souboru do formátu ARFF vyskytovaly se chyby v datech. Dřív než najít řešení tohoto problému, chtěl jsem zkusit naimplementovat nějaký model pomocí Weka programu a zjistit jestli se mi to podaří. Bohužel načíst datový soubor do programu se nepodařilo. Proto hlavní důvod proč jsem odmítmul balík programů Weka, je nekompatibilita se strukturou kódu mého programu.



Obrázek 2.1.: Datový soubor otevřený v uživatelském rozhraní Weka

Encog

Encog je framework pro strojové učení (*Machine Learning*), což je oblast která studuje algoritmy umožňující strojům "učit se". Je to vlastně podoblast umělé inteligence (*Artificial Intelligence*), kterou používají pro studování inteligentního chování, adaptaci ve strojích, a taky pro řízení, rozhodování a plánování procesů v různých systémech.

Tento framework má hodně technik zahrňujících regresní modely, které jsem pravě potřeboval. Komplikace jeho využití byla v tom, že zaklád tohoto frameworku je vytvořen za použití neuronových sítí. Jak jsem zjistil později naučení těchto technologií je časově náročné. Kvůli tomu jsem se rozhodl nechat tento nápad realizace programu a hledat jiné řešení.

2.2. Implementace

2.2.1. Datová vrstva

Jako prvek dat, se kterými pracují výpočetní servery, dostal jsem od svého vedoucího *log file* (nebo taky *žurnál*). Log je to většinou textový soubor s příponou *.log*, který obsahuje záznam činností nějakého programu nebo procesu. Tento záznam je ve tvaru posloupnosti jednotlivých kroků přislušného běhu programu, tj. skládá se z informace o tom jak, kdy, čím nebo kým byla využívána konkretní aplikace, jaké zatížení bylo vyvoláno nebo kolik času trvalo aby se příslušná aplikace proběhla. Používájí se logy pro kontrolu procesů a zjištění přesné příčiny nějaké chyby, pokud k takové došlo. Z jejich záznamů je možno určit co přivedlo k chybě, jaký měla vliv tato chyba a jaké pak měl běžící proces důsledky.

Daný soubor skládá se z dostatečně velkého množství řádků, v každém ze kterých zaznamenávano několik parametrů. Tyto parametry obsahují svou hodnotu a/nebo svůj název.

Obrázek 2.2.: Log soubor (fragment)

Pro pohodlnější práci se souborem a odstranění nepotřebných parametrů (např. čas, datum) rozhodl jsem se překonvertovat log soubor do souborového formatu *CSV* (z angl. *comma-separated values*, nebo hodnoty oddělené čárkami). Je to jeden z nejednoduších formatů pro práci s tabulkovými daty. Považuji vybraný format za vhodný, protože má podobnou strukturu jako u log souboru a je velice používán pro výměnu informací mezi různými systémy. Také dost důležité je, že se sloupce CSV souboru můžou tvářit jako časové řady, které se hodně využívají v regresní analýze.

compTimeStat.csv

```
 1 11487,30671,4377,0,0,3,153,2991,416276,28,132,0,7203
 2 11488,36236,3526,0,0,3,162,5737,741679,33,212,0,2822
 3 11489,35395,1368,0,0,3,127,4351,495597,28,203,0,2203
 4 11490,32391,1928,0,0,3,152,3186,421065,28,131,0,2878
 5 11491,34507,1251,0,0,3,155,3872,412623,25,181,0,2071
 6 11492,31664,2456,0,0,3,148,3765,456284,33,150,0,1938
 7 11493,27661,2086,0,0,3,150,2375,271744,25,101,0,1577
 8 11494,31260,1805,0,0,3,140,4871,532823,33,256,0,1741
 9 11495,33617,1405,0,0,3,153,4001,503794,34,213,0,1975
10 11496,33283,1219,0,0,3,186,2828,368302,28,180,0,2533
11 11487,30671,4377,0,0,3,153,2991,416276,28,132,0,7203
12 11488,36236,3526,0,0,3,162,5737,741679,33,212,0,2822
13 11489,35395,1368,0,0,3,127,4351,495597,28,203,0,2203
14 11490,32391,1928,0,0,3,152,3186,421065,28,131,0,2878
15 11491,34507,1251,0,0,3,155,3872,412623,25,181,0,2071
16 11492,31664,2456,0,0,3,148,3765,456284,33,150,0,1938
17 11493,27661,2086,0,0,3,150,2375,271744,25,101,0,1577
18 11494,31260,1805,0,0,3,140,4871,532823,33,256,0,1741
19 11495,33617,1405,0,0,3,153,4001,503794,34,213,0,1975
20 11496,33283,1219,0,0,3,186,2828,368302,28,180,0,2533
21 11497,37169,3919,0,0,3,169,5565,641585,31,293,0,6942
22 11498,37074,1479,0,0,3,201,5013,608810,30,291,0,2932
23 11499,35905,3587,0,0,3,186,4570,535530,29,229,0,2475
24 11497,37169,3919,0,0,3,169,5565,641585,31,293,0,6942
25 11498,37074,1479,0,0,3,201,5013,608810,30,291,0,2932
```

Obrázek 2.3.: CSV soubor (fragment)

2.2.2. Aplikační vrstva

2.2.3. Prezentační vrstva

2.3. Testování aplikace

2.3.1. Testovací data

2.3.2. Zpracování a analýza dat

2.3.3. Predikce

2.3.4. Vizualizace

Závěr

Literatura

[1] Fumio Hayashi. *Econometrics*. Princeton. Princeton University Press. 2000.

[2] Tomáš Cipra. *Analýza časových řad s aplikacemi v ekonomii*. Praha. SNTL - Nakladatelství technické literatury. 1986.

[3] Petr Fiala. *Úvod do ekonometrie*. Praha. Nakladatelství ČVUT. 2008.

A. Název přílohy