Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Teorie her a jejich aplikace na e-commerce

Bc. Ondřej Komárek

Diplomová práce

2018

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2017/2018**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Ondřej Komárek**
Osobní číslo:	**I15211**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Teorie her a jejich aplikace na e-commerce**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Cílem diplomové práce je kompletní zpracování teorie her, jejich principů a jejich využití v e-commerce. V teoretické části budou statistické výpočty včetně příslušné teorie, vysvětlených použitých metod a grafů. V praktické části diplomové práce bude vyvinut software, který zpracuje strukturovaná statistická data a na základě nich doporučí produkty/kategorie/doplňky k produktům, které by bylo nejvhodnější danému zákazníkovi v danou dobu zobrazit. Součástí bude uživatelský manuál. Diplomová práce by měla využívat reálná data a na základě nich poskytnout reálné výsledky z dané oblasti e-commerce.

Rozsah grafických prací:	**10**
Rozsah pracovní zprávy:	**60**
Forma zpracování diplomové práce:	**tištěná**

Seznam odborné literatury:

MAŇAS, Miroslav. Teorie her a konflikty zájmů. Vyd. 1. V Praze: Vysoká škola ekonomická v Praze, 2002, 114 s. ISBN 80-245-0450-2.
MAŇAS, Miroslav. Teorie her a její aplikace. 1. vyd. Praha: Stát. nakl. techn. lit., 1991, 278 s. Teoretická knižnice inženýra. ISBN 80-030-0358-X.

Vedoucí diplomové práce:	**Mgr. Alena Pozdílková, Ph.D.** Katedra matematiky a fyziky
Datum zadání diplomové práce:	**30. října 2017**
Termín odevzdání diplomové práce:	**18. května 2018**

Ing. Zdeněk Němec, Ph.D.
děkan

L.S.

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2017

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 18. 5. 2018

Bc. Ondřej Komárek

Poděkování

Tímto bych rád poděkoval své vedoucí práce Mgr. Pozdílkové Aleně Ph.D. za odborné vedení, trpělivost a ochotu, kterou mi v průběhu zpracování diplomové práce věnovala.

ANOTACE

Cílem této práce je vyvinout knihovnu, kterou bude možné nasadit na stávající e-commerce řešení. Ta na základě historie statistických záznamů doporučí produkty a výši jejich slevy, které by bylo nejvhodnější danému zákazníkovi nabídnout. V teoretické části je pojednáno o teorii her obecně a následně o jejím využití v oblasti e-commerce. V následujících kapitolách jsou představeny principy funkce knihovny a její nasazení na internetový obchod.

KLÍČOVÁ SLOVA

Teorie her, PHP, e-commerce, knihovna

TITLE

Game theory and their application to e-commerce

ANNOTATION

The purpose of this paper is to develop a library which could be applicable in the existing e-commerce solutions. On the basis of available statistical records, the library will recommend the products, including possible discount levels, that would be most appropriate for offering to a particular customer. The theoretical section deals with the game theory in general and with its ensuing application in the e-commerce. The following chapters describe the principles of the library's functioning and of its utilization in an existing e-shop.

KEYWORDS

Game theory, PHP, e-commerce, library

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM ZKRATEK

PPC	Product Payoff Coefficient
PHP	PHP: Hypertext Preprocessor)
UML	Unified Modeling Language
LTS	Long Term Support
PSR	PHP Standards Recommendations
MVC	Model View Controller
MIT	Massachusetts Institute of Technology

ÚVOD

V posledním desetiletí došlo k obrovskému rozmachu e-commerce. V dnešní době skoro každý drobný i rozsáhlý nákupní řetězec používá svůj internetový obchod. Jedná se o obchodníkův sen, jelikož je možné s náklady v podstatě zanedbatelnými, oproti kamenné prodejně, oslovit zákazníky na celém světě.

Díky tomuto faktu značně poklesly náklady na vstup do této sféry podnikání. Na trhu je dostupné řešení internetového obchodu vytvořené na základě šablony a stojí pouze několik stovek až tisíc korun měsíčně. Dále je možné využít i komplexní na míru dělaný software, jehož cena se může vyšplhat až do řádu milionů korun, proto se prodejci snaží odlišit i jinými způsoby od nízkých cen, přes nabízené služby, až po individuální přístup pro každého zákazníka. Ambicí této práce je komplexní zpracování teorie her a jejích principů v této oblasti. Ty spolu s vybranými statistickými postupy budou využity k porozumění zákazníkovi a k předvídání jeho budoucího rozhodování při nákupním procesu v oblasti e-commerce.

V první kapitole teoretické části bude představena teorie her a teorie optimálního rozhodování. V teoretické části práce budou podrobně představeny matematické postupy, které budou dále využity ve zbytku práce. V praktické části práce bude vytvořena PHP knihovna, jejíž nasazení bude umožněno pomocí PHP správce balíčků Composer.

Cílem této knihovny bude vytvoření frameworku, který bude uživateli pomocí srozumitelného API nabízet implementaci postupů a nástrojů, které byly zmíněny v teoretické části práce. Bude se jednat o automatizovaný nástroj, který přijme standardizované statistické údaje, následně je agreguje a využije k nabídce nejvhodnějších produktů, které by měly být danému zákazníkovi nabídnuty a případně doporučí nasazení slev.

S pomocí teoretického základu bude zpracován ucelený pohled na existující internetový obchod. Ten se bude skládat z analýzy dosavadního chování zákazníků v různých situacích. Dále dojde k praktické implementaci knihovny na tento internetový obchod.

V neposlední řadě bude součástí práce odborný uživatelský manuál, který slouží k usnadnění implementace této knihovny do již existujících e-commerce řešení. Samozřejmostí bude i komplexní dokumentace umožňující doplňování či rozšiřování knihovny.

1 TEORIE HER

Teorie her je vědní disciplína, zabývající se studiem lidského konfliktu a kooperace v konkurenčních situacích. Koncepty teorie her se aplikují, pokud jsou akce několika entit nezávislé. Tyto entity zahrnují jednotlivce, skupiny, firmy, nebo jejich kombinace. Klíčovými průkopníky byli matematici John von Neumann, John Nash a také ekonomik Oskar Morgenstern. Koncepty teorie her nám poskytují ucelený systém, podle kterého můžeme formulovat, analyzovat a porozumět strategickým situacím.

Aplikace teorie her vyžaduje znalost identity nezávislých rozhodovatelů, jejich zálib, jaké tahy mohou provádět a jak jejich rozhodnutí ovlivňují výsledek hry. Předpokládáme také, že se každý rozhodovatel chová racionálně. Teorie her má rozsáhlé spektrum aplikací, jak v politice, armádě, ekonomii, informatice, tak i v psychologii a evoluci. [1]

Základním pojmem teorie her je rozhodovací situace. Do rozhodovací situace se dostáváme neustále, ať už hrajeme salónní hry, jako jsou šachy, ruleta, karty, nebo kdekoliv v reálném životě při situaci, kde vystupuje jeden nebo více rozhodovatelů.

Pro snadnější orientaci budeme dále rozhodovatele (účastníky) nazývat hráči. Při výskytu rozhodovací situace se dvěma a více hráči často dochází ke konfliktu zájmů tím, že rozhodnutí, které je optimální pro jednoho hráče, nemusí být vhodné pro jiného hráče nebo mu může uškodit. V tomto případě už nestačí pouze zjistit správné řešení pro každého hráče zvlášť, ale musíme zohlednit výhodnost situace, pokud možno pro všechny hráče. Každý z hráčů má svou množinu přípustných tahů (funkcí), kterou budeme nazývat prostorem strategií. Z této množiny volí hráči svoje rozhodnutí a následnou strategii.

Soubor možných akcí v dané pozici může záležet na hráči, který je na řadě, nebo také na souboru náhodných akcí jako třeba hod kostkou. Je také nutné vědět, s jakými informacemi hráči disponují. Zda ví, jaké akce byly provedeny v minulosti, či jaký důsledek bude mít každý z tahů.

Ukazatel, který zhodnotí důsledek volby strategie, budeme dále nazývat výplatní funkcí a hodnotu tohoto ukazatele nazýváme výhrou. Matematický model rozhodovací situace je uváděn jako hra v normálním tvaru. [2]

Hry jsou charakterizovány počtem hráčů nebo rozhodovatelů, kteří na sebe vzájemně působí, navzájem se ohrožují, tvoří koalice a provádí akce s nejasným důsledkem. V některých případech je možné s relativní jistotou předpovědět ideální další tah, bohužel ve

většině situací se pokusy o takovéto předpovědi stávají pouze snahou lépe pochopit již vykonané akce a jejich důsledky s nadějí na lepší předpovědi v budoucnu.

Modely teorie her jsou vysoce abstrahovanými reprezentacemi situací z reálného světa. Například teorie Nashovy rovnováhy bývá často využita ke studiu oligopolní a politické konkurence. Hranice mezi aplikovanou a čistě teoretickou teorií her bývá většinou vágní. Přestože je matematika využívána k formálnímu vyjádření myšlenek, lze je beze změny významu vyjádřit i bez použití matematiky. Matematická formulace ale napomáhá k přesné definici konceptů, k udržení jejich konzistence a k lepšímu porozumění implikace předpokladů.

V teorii her je využito tří hlavních matematických modelů: extenzivní formy, strategické formy a koaliční formy. Ty jsou v zásadě odlišovány podle toho, jak detailně je v nich hra popsána. Nejlépe je hra popsána extenzivní formou. Přestože by se extenzivní forma mohla zdát jako ideální, při velkém počtu hráčů se i abstraktnější strategická forma hry stává natolik komplexní pro jakoukoliv smysluplnou analýzu. V těchto případech je vhodné využít koaliční formu, ve které je od strategie abstrahováno a pracuje se hlavně s koalicí a s její hodnotou.

Většina her, které zahrnují větší množství účastníků, vykazuje tendenci utváření koalic. Pokud k tomu dojde, vzniká předpoklad, že každá koalice svým členům nabídne jistou část výplaty. Tato částka se stává hodnotou dané koalice. Dá se také předpokládat, že může dojít k vytvoření takzvané velké koalice, což je koalice skládající se ze všech hráčů, kteří se v dané hře nachází. [3]

1.1 Vymezení pojmů

Hráč

Hráčem rozumíme entitu, která mnohdy musí provádět rozhodnutí v nejistotě. Hráč obvykle neví o akcích ostatních hráčů, které nejsou deterministické, o jejich uvažování, nebo není dostatečně informován o událostech odehrávajících se během hry. [4]

Hra

Hra je popisem strategických interakcí a situací. Obsahuje soubor akcí, které mohou hráči provést, ale nespecifikuje akce provedené hráči. [4]

Obecná znalost

Fakt je obecnou znalostí tehdy, pokud ho všichni hráči znají a zároveň každý hráč ví, že tento fakt znají všichni ostatní hráči. Pravidla hry jsou většinou považována za obecnou znalost. [1]

Dominantní strategie

Strategie je silně dominující nad strategií protihráče právě tehdy, pokud vždy vynese hráči lepší zisk nezávisle na tom, co jeho protihráč podnikne. Pokud je tato strategie vždy alespoň tak dobrá, jako strategie protihráče, jedná se o strategii slabě dominující. [1]

Mixovaná strategie

Mixovaná strategie je takovou strategií, jež v sobě zahrnuje aktivně využívanou náhodnost s danými pravděpodobnostmi, které určí hráčovo rozhodnutí. [1]

Nashova rovnováha

Konečný počet tahů a konečný počet voleb v každém tahu (konečné hry) mají vždy rovnovážný bod, ve kterém všichni hráči vyberou takovou akci, která je pro ně ideální vzhledem k jejich protivníkům. [1]

Výhra

Výhrou rozumíme číslo odrážející výhodnost výsledku pro hráče. Předpokládaná výhra také zahrnuje přístup hráče k riziku. Výhra se nazývá přenosnou v případě, že se hráči mohou smluvně zavázat, že účastník, kterému dohoda přináší podstatně lepší důsledek, předá zbývajícímu účastníkovi jistou úhradu. Jestliže jsou dohody možné, ale sdílení důsledků nikoliv, jedná se o výhru nepřenosnou. [2]

Hra s nulovým součtem

Hra s nulovým součtem nebo také striktně kompetitivní hra, je taková hra, ve které dochází k rovnováze výher tak, že jejich výsledný součet je vždy nula. To znamená, že zisk jednoho hráče se přímo promítá na ztrátě hráče jiného. [3]

Perfektní informace

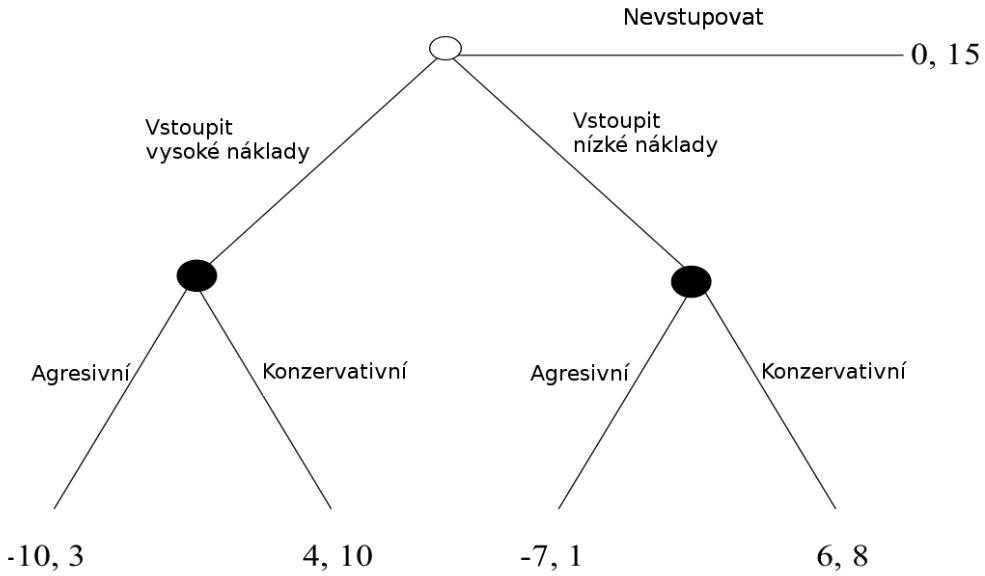
Perfektní informace ve hře nastává, pokud v jakémkoliv bodě v čase jen jediný hráč provede tah a zná všechny akce, které byly v dané hře doposud provedeny. [1]

Racionalita

Racionálním hráčem se rozumí hráč hrající v takové podobě, která maximalizuje jeho osobní výhru. Racionalita je často pokládána za obecnou znalost. [1]

Herní strom

Herní stromy se od ostatních analytických nástrojů, jako jsou optimalizační a rozhodovací stromy, podstatně liší. Na rozdíl od nich neberou akce ostatních stran jako dané a jejich analýza se nezaměřuje na optimalizaci pouze z pohledu jednoho hráče, ale berou v potaz strategické chování ostatních hráčů. Lze je také poměrně snadno reprezentovat při modelování s využitím výpočetní techniky. [5]



Obrázek 1: Herní rozhodovací strom [5]

Strategická forma

Hra ve strategické formě, nazývána též normální formou, je kompaktní reprezentace hry, v níž hráči zároveň volí své strategie. Výsledné výhry jsou prezentovány v tabulce, která má buňku pro každou kombinaci strategií. [1]

Koaliční forma

Koaliční model je od ostatních odlišen hlavně tím, že se zaměřuje na to, co mohou dosáhnout skupiny hráčů, než na to, co může dokázat individuální hráč. Abstrahuje také od vnitřního fungování daných skupin. [4]

Extenzivní forma

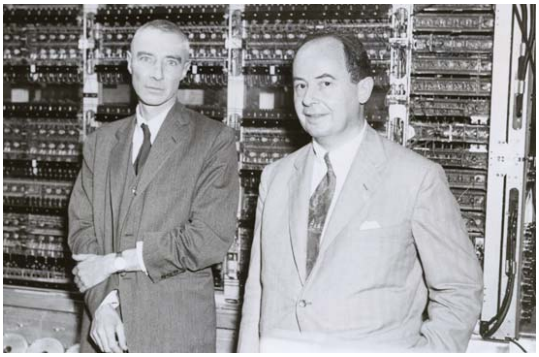
Extenzivní forma hry popisuje pomocí stromu průběh hry. Ukazuje pořadí, v jakém hráči provádí své tahy a informace, které má každý hráč v daném bodě rozhodování. Nejdůsledněji dodržuje skutečná pravidla hry. [1]

Strategie

Ve hře, která se nachází ve strategické formě, se strategií nazývá jedna z možných akcí daného hráče. V extenzivní hře je strategií kompletní plán tahů pro každý bod rozhodování. [1]

1.2 Historie

Nejstarším příkladem formální herně-teoretické analýzy je studie francouzského filozofa jménem Antoine Augustin Cournot z roku 1838, která se zabývala duopoly. V roce 1921 publikoval francouzský matematik Emile Borel sérii prací, v nichž byla poprvé definována teorie her. Ta byla dále rozvinuta maďarsko-americkým matematikem a fyzikem doktorem John Von Neumannem v roce 1928 v jeho práci Theory of parlor games.



Obrázek 2: Dr. John Von Neumann (vpravo) a Dr. J. Robert Oppenheimer [6]

Samostatnou vědeckou oblastí se teorie her stala vydáním rozsáhlé knihy s názvem Theory of Games and Economic Behavior. Napsal ji John Von Neumann spolu s ekonomem Oskarem Morgensternem. Tuto publikace vydal v roce 1944 Princeton University Press a obsahovala metody pro hledání optimálních řešení pro hry o dvou hráčích. Poskytovala také základní terminologii a vysvětlení problémů, které se používají dodnes.

V roce 1950 vědci firmy RAND vytvořili jednu z nejvlivnějších koncepcí teorie her nazvanou Vězňovo dilema. Ta pojednává o konfliktu zájmu jednotlivce na úkor obecného dobra. Studie tohoto konceptu ukázala velkou sílu ve vysvětlení chování inteligentních organismů. [7]

Roku 1950 americký matematik John Forber Nash demonstroval, že hry ve kterých má každý hráč konečný počet tahů a konečný počet voleb v každém tahu (konečné hry),

mají vždy rovnovážný bod, v němž všichni hráči vyberou takovou akci, která je pro ně ideální vzhledem k jejich protivníkům. Tento centrální koncept nekooperativní teorie her se nazývá Nashova rovnováha a od té doby se stal ohniskem dalších analýz. V roce 1994 za tuto teorii dostal spolu s Johnem Harsanyim a Reinhardem Seltenem Nobelovu cenu v ekonomii.

Mezi lety 1950 a 1970 byla teorie rozšířena a aplikována na problémy válek, politiky a ekonomie. Dále si našla uplatnění v sociologii, psychologii, výpočetní technice a také byla využita pro efektivnější rozdělování pásem elektromagnetického spektra v telekomunikačním průmyslu.[1]

1.3 Dělení rozhodovacích situací

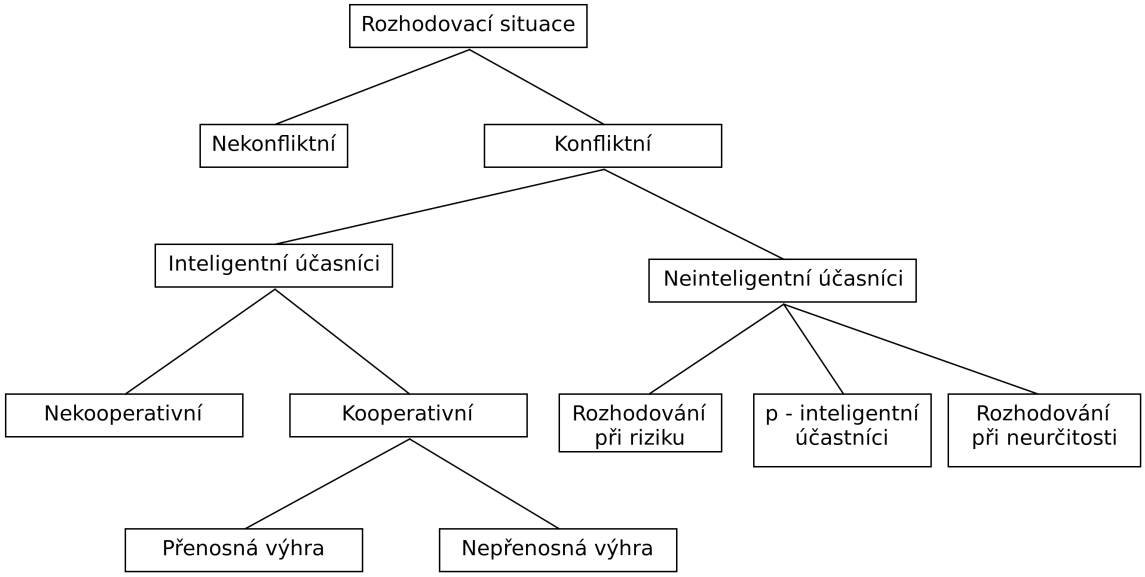
Rozhodovací situací nazveme takovou situaci, v níž vystupuje jeden nebo více účastníků. Za předpokladu, že dané rozhodnutí má pro hráče nějaký důsledek a on může určit, který důsledek je pro něho příznivější. Na základě tohoto rozhodnutí dělíme situace na nekonfliktní a konfliktní. Tento důsledek x lze charakterizovat číslem $M(x)$, a to tak, že hráč dává přednost přednost důsledku rozhodnutí x oproti x' právě v případě, když $M(x) > M(x')$.

Jednodušší rozhodovací situací je situace nekonfliktní. V této situaci není samotné rozhodnutí ovlivněno dalšími účastníky a jde jen o to, aby byl z množiny přípustných tahů vybrán takový, který maximalizuje výplatu. Příkladem by mohlo být rozhodnutí, jaký výrobní program při pevně stanoveném odbytu bude v továrně vybrán tak, aby došlo k maximalizaci zisku

Oproti nekonfliktní rozhodovací situaci je konfliktní situace mnohem složitější. Důsledek rozhodnutí již není závislý pouze na rozhodnutí jediného účastníka, ale i dalších jejích účastníků. V tomto případě předpokládáme, že první účastník je jednotlivec, nebo jednomyslně jednající organizace. Ostatní účastníci mohou být inteligentní bytosti podobné povahy jako například zákazníci. V reálném světe konfliktní rozhodovací situace ve značné míře převládají nad situacemi nekonfliktními. Příkladem konfliktní situace může být rozhodnutí, na jaké zboží uplatnit v daný čas slevu. Zdrojem nejistoty se pak stává nepředvídatelná poptávka, konkurence, nebo libovolné kombinace jiných faktorů.

Konfliktní situace mohou být dále zkomplikovány nepředvídatelnými, náhodnými mechanizmy. Ty mohou být reprezentovány počasím, ale i v některých případech jinými ra-

cionálními účastníky. Tento účastník se snaží dosáhnout pro něj nejlepšího výsledku, ale z nějaké příčiny při tom není schopen postupovat zcela racionálně a tím se jiným účastníkům jeví jako kompromis mezi racionálním účastníkem a náhodným mechanizmem. Takový účastník je nazýván jako p-inteligentní. [2]



Obrázek 3: Dělení rozhodovacích situací [2]

1.4 Vězňovo dilema

Vězňovo dilema je klasickým případem uplatnění teorie her v praxi. V tomto příkladu jsou hlavními aktéry dva vězňové ze stejného kriminálního gangu (vězeň A a vězeň B). Oba jsou zadrženi a umístěni do oddělených cel, kde k sobě nemají přístup. Jsou usvědčeni z menšího zločinu, který by jim sám o sobě vysloužil například jeden rok ve vězení. Tento trest jim bude udělen, pokud by oba u soudu mlčeli.

Policie ale ví, že provedli zločin mnohem závažnější, ale nemá pro něj žádné důkazy. Proto každému z vězňů nabídnou možnost spolupráce výměnou za zproštění obžaloby. Pokud by jeden z vězňů spolupracoval (S) a druhý se rozhodl zamlčet (Z), nebude tomu vězni, který se rozhodl spolupracovat, vyměřen žádný trest a ten, co zamlčel, bude odsouzen na čtyři roky. Pokud by se ale oba rozhodli spolupracovat, každý z nich by strávil ve vězení nejméně dva roky.

	S(B)	Z(B)
S(A)	-2 (A) / -2 (B)	0 (A) / -4 (B)
Z(A)	-4 (A) / 0 (B)	-1 (A) / -1 (B)

Obrázek 4: Vězňovo dilema

Vězňovo dilema, jako jakákoliv hra dvou hráčů ve strategické formě, může být popsána tabulkou, jako je zobrazena na obrázku 4. Zde řádky reprezentují strategie hráče A a sloupce hráče B. Kombinace těchto strategií definují výplatu, která je pro každého hráče zobrazena v buňce tabulky.

V tomto příkladu strategie spolupráce dominuje strategii mlčení. Strategie spolupracovat u hráče A dominuje strategii mlčení, protože v případě, že se hráč B rozhodne spolupracovat, tak výplata hráče a je -4, ale pouze -2, když by se rozhodl spolupracovat. Pokud by se hráč B také rozhodl mlčet, tak je výplata -1 oproti 0 v případě spolupráce. Vzhledem k symetrii této hry, to samé platí pro hráče B.

Strategie spolupráce je proto dominantní, protože nabízí lepší výplatu bez ohledu na to, co udělá druhý hráč, ale na druhou stranu je jejím výsledkem nižší výplata pro oba hráče dohromady. Každý racionální hráč se vždy rozhodne pro dominantní strategii, přestože v tomto případě by bylo pro oba vězně lepší, kdyby se rozhodli pro nedominantní strategii mlčení. [4]

1.5 Teorie opakujících se her

Model opakujících se her slouží ke zkoumání logiky dlouhodobých interakcí. Bere v potaz to, že hráč do své současné strategie zahrnuje možná chování ostatních hráčů v budoucnosti.

Opakující se hry jsou velmi dobře ilustrované na opakujícím se vězňově dilematu. Tato hra má ve svém základě Nashovu rovnováhu pro volbu spolupráce, která striktně dominuje strategii mlčení. Toto se ale v případě opakování se stejnými hráči zásadně mění. V tomto případě je vzájemně nejprospěšnější strategie mlčení a stává se stabilně vybíranou strategií pro oba hráče. To je dáno tím, že oba hráči vědí, že výběr spolupráce sice přinese krátkodobý zisk, ale zároveň s velkou pravděpodobností ukončí spolupráci, a to by v dalších hrách znamenalo ztrátu silně převažující prvotní zisk.

Primárním cílem této teorie je izolovat ty strategie, které mají za důsledek požadovaný zisk v jakémkoliv kole hry. Dává nám také náhled na to, jak individuální hráči reagují opakovaně, a tím vzniká struktura, která může být interpretována jako „sociální norma“. Výsledky pak ukazují, že tato sociální norma, která je nutná k udržení vzájemně prospěšných výsledků, v sobě zahrnuje trest, který hráč použije na každého jiného hráče, jehož chování není žádoucí. Hrozba trestu samozřejmě musí být uvěřitelná a hráči musí mít podnět trest udělovat.

Tyto takzvané lidové teorémy demonstrují, že žádoucí výsledky, které není možné udržet, pokud jsou hráči krátkozrací, jdou udržet v případě, že hráči mají dlouhodobé cíle. Na druhou stranu ale ukazují, že soubor rovnováh opakovaných výsledků je tak obrovský, že představa rovnováhy ztrácí vypovídající hodnotu. [4]

1.5.1 Nekonečně se opakující hry a konečně se opakující hry

Model opakovaných her má dvě verze. V první je konec hry viditelný a je ho možné v nějakém určitém časovém horizontu dosáhnout. Pokud je již zmíněné vězňovo dilema hrou konečně se opakující, jediný výsledek podléhající Nashově rovnováze, který mohou hráči vybrat, je strategie mlčení. Naproti tomu v případě nekonečného opakování je soubor perfektně rovnovážných členských her obrovský. Proto je před aplikací modelu opakujících se her ve specifických situacích nutné rozhodnout, zda je vhodné využít konečný, či nekonečný počet opakování. [4]

1.5.2 Nekonečně se opakující hry

Model nekonečně opakujících se her odráží situaci, ve které hráč opakovaně hraje strategickou hru G, které se také říká ustavující hra. Na G není dán žádný limit a všichni hráči

si své tahy vybírají najednou, přičemž každý hráč ví o všech předchozích tazích, které byly ve hře provedeny.

Definice takové hry je takováto:

Nechť $G = \langle N, (a_i), (\succsim_i)\rangle$ je strategická hra, kde N je množina hráčů, a_i je akcí hráče i, $\succsim_i$ je preferenční relace hráče i, Nechť nekonečně opakující se hra G je extenzivní hra s perfektní informací a současnými tahy $\langle N, H, P, (\succsim_i^*)\rangle$, kde $(\succsim_i^*)\rangle$ je profil preferenční relace nad C, kde C je množina následků. Ve které:

- $H = \{\emptyset\} \cap (\cap_{t=1}^{\infty} A^t) \cap A^{\infty}$, kde $\emptyset$ je počáteční historie a A^{∞} je nekonečný soubor série $(a^t)_{t=1}^{\infty}$ akčních profilů v G.
- $P(h) = N$ pro každou neterminální historii $h \in H$.
- $\succsim_i^*$ je preferenční relace pro A^{∞}, která rozšiřuje preferenční relaci $\succsim_i$ v takovém smyslu, aby splnila následující podmínku slabé oddělitelnosti: pokud $(a^t) \in A^{\infty}, a \in A, a' \in A, a \succsim_i a'$ pak platí $(a^{1,}..., a^{t-1}, a, a^{t+1}, ...) \succsim_i^* (a^{1,}..., a^{t-1}, a', a^{t+1}, ...)$ pro všechny hodnoty t.

Historie je terminální jen a pouze v případě, že je nekonečná. Po každé neterminální historii každý hráč zvolí akci. Tudíž strategie hráče i je funkcí přiřazující akci v A_i každé konečné sekvenci výsledků v G.

Je dokázáno, že soubor Nashových rovnováh nekonečně opakovaných her zahrnuje výsledky, které nejsou opakováním Nashovy rovnováhy ustavující hry. Aby bylo daného výsledku dosaženo, každý hráč musí být odrazen od odchýlení se pomocí trestu. Takovýto trest může nabývat mnoho forem. Jedna z možností je, že každý hráč, který využívá spouštěcí strategii (trigger strategy), v tom případě jakékoliv odchýlení znamená pro daného hráče represivní akci, která má nekonečné trvání.

Represivní akce s nekonečným trváním se dále dělí podle způsobů, jak je trest vynucen. Prvním možností je trestání vykonavatelů trestu. V tomto případě je každý pokus hráče zvýšit svůj zisk pomocí deviace na základě jakékoliv historie, a to včetně chvíle, kdy má být vykonán trest je kompenzována následným trestem od ostatních hráčů. To znamená, že dochází k postihu i toho hráče, který nevykonal předpokládaný trest, tak aby byla zpět nastolena rovnováha hry.

Další deviace by vyžadovaly delší a delší tresty, proto by strategie, při kterých se tento typ trestu používá, měly být postaveny tak, aby bylo možné vynášet neomezeně dlouhé

tresty. Přesto všechno se může stát, že nějaký trest vyústí v takové ztráty, že je nebude možné nikdy nahradit.

Další možností je odměňování hráčů, kteří se rozhodnou podílet na trestu, čímž dochází k motivaci trestat. Jako v předchozím případě je nutné vytvořit strategický profil, v němž je rovnovážná cesta reprezentována jediným a přísně vymahatelným výsledkem. [4]

1.5.3 Konečně se opakující hry

U konečně se opakujících se her je formální popis velmi podobný nekonečně se opakujícím hrám: pro každé pozitivní číslo T a T-periodu konečně se opakující hry o strategii $\langle N, (a_i), (\succsim_i)\rangle$, kde N je množina hráčů, a_i je akce hráče i, $\succsim_i$ je preferenční relace hráče i. $\langle N, (a_i), (\succsim_i)\rangle$ je extenzivní hra s perfektní informací, když splňuje podmínku:

- $H = \{\oslash\} \cup (\cup_{t=1}^{T} A^t) \cup A^T$ kde $\oslash$ je počáteční historie, A^T je konečný soubor série $(a^t)_{t=1}^{T}$ akčních profilu v G.
- $P(h) = N$ pro každou terminální historii $h \in H$.

Přičemž preferenční relace $\succsim_i^*$ pro každého hráče i v konečně se opakující hře je reprezentována funkcí $\sum_{t=1}^{T} u_i(a^t)/T$, kde u_i je výplatní funkce, která reprezentuje preferencí hráče i v ustavující hře.

Podobně jako u nekonečně se opakujících her je hráč odrazován od odchýlení od vzájemně žádoucího výsledku hrozbou trestu, pokud tak udělá. Přesto jsou nutné některé modifikace, protože výsledek posledního kola musí mít Nashovu rovnováhu stejnou jako je Nashova rovnováha ustavující hry. Problém nastává v posledním kole hry, kde se hráč může odchýlit bez nějaké hrozby trestu. [4]

1.5.4 Bayesovské hry

U her s kompletní informací se předpokládá, že hráči mají perfektní přehled o všech elementech hry. Tyto hry ale nejsou v reálných aplikacích jediné. Dalším typem jsou hry s nekompletní informací. Ty se snaží simulovat případ, kdy někteří hráči mají před začátkem hry informaci, kterou si nechají pro sebe. Počáteční soukromá informace se nazývá hráčův typ. Například může jít o konkrétní hodnotu slevy produktu ještě před jejím zveřejněním, nebo o naceněnou hodnotu předmětu v aukci. Typem může být obecně jakákoliv soukromá informace, která je relevantní vůči hráčovu rozhodování.

Bayesovská hra je charakterizována jako strategická forma hry s nekompletní informací a skládá se z:

- Souboru hráčů $N = \{1, ..., n\}$, kde každý hráč $i \in N$ a $-i \in N \wedge -i \neq i$.
- Souboru akcí A_i, kde $A = \prod_{i \in N} A_i$.
- Souboru typů Θ_i, kde $\Theta = \prod_{i \in N} \Theta_i$.
- Pravděpodobnostní funkce $p_i : \Theta_i \to \Delta(\Theta_{-i})$.
- Výplatní funkce $u_i : A \times \Theta \to \mathbb{R}$.

Zde funkce p_i shrnuje, co si hráč i myslí o typech ostatních hráčů vzhledem k jeho typu. Tudíž $p_i : (\theta_{-i}|\theta_i)$ je podmíněná pravděpodobnost přiřazená typu profilu $\theta_{-i} \in \Theta_{-i}$. Podobně $u_i(a|\theta)$ je výplatní funkcí hráče i, kde akční profil je a a typový profil je θ.

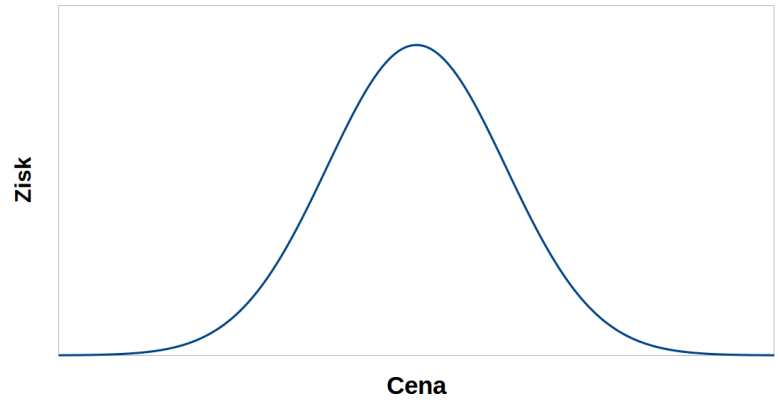
Bayesovskou hru nazýváme konečnou, pokud N, A_i a Θ_i jsou konečné pro všechny $i \in N$. Čistá strategie Bayesovské hry pro hráče i je funkce, která mapuje typ hráče i do jeho souboru akcí tak, že $a_i(\theta_i)$ je vybranou akcí typu θ_i hráče i. Mixovaná strategie hráče i je $\alpha : \Theta_i \to \Delta(A_i)$ tak, že $\alpha(a_i|\theta_i)$ je pravděpodobností přiřazenou α_i akci a_i o typu θ_i hráče i. [8]

1.6 Kooperativní a nekooperativní vyjednávací teorie

Vyjednávání bude pravděpodobně součástí jakékoliv transakce, kde objekt obchodu je v jistém smyslu pro hráče unikání, ale jeho potřebnost je omezená. Unikátnost dává hráči, který tento objekt vlastní, jistý monopol či vyjednávací sílu. Pokud by na daný objekt monopol neexistoval, hráči by jednoduše šli za jiným hráčem vlastnícím daný objekt. Monopol umožňuje hráči ovlivňovat podmínky obchodu.

1.6.1 Kooperativní vyjednávání

Jako příklad by se dala uvést firma, která vyrábí a prodává produkt, který je sice na trhu unikátní, ale pro zákazníky není nezbytně nutný. Firma má možnost stanovit cenu daného produktu a tím zvýšit svůj zisk. Pokud by ale cena vzrostla neměrně hodnotě pro spotřebitele, zisk firmy by klesal. Proto je nutné najít rovnovážný stav, díky kterému budou maximalizovány zisky.



Obrázek 5: Cena a zisk

Vyjednávání v kooperativní hře se oproti individuální hře, kde každý racionální hráč nebude souhlasit s žádnou dohodou, která by mu přinesla menší zisk než odmítnutí této dohody, liší v tom, že hráči obvykle shodnou na něčem, čeho nemohou individuálně dosáhnout, tedy Paretova optima.

John Forbes Nash Jr. dokázal, že existuje unikátní řešení problému vyjednávání, které je známo jako Nash bargaining solution (Nashovo řešení pro vyjednávání). Výsledek tohoto řešení je takový, ve kterém je maximalizován produkt hráčů z jakékoliv dohody. Tento produkt je známý jako Nashův produkt.

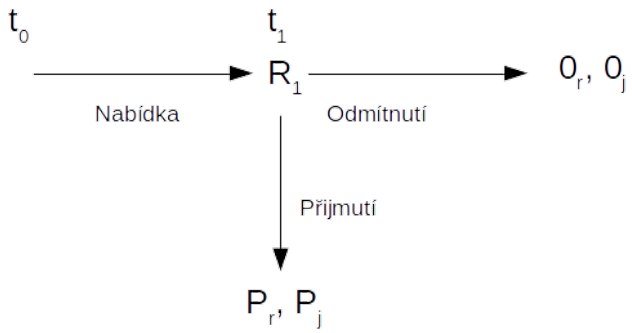
1.6.2 Nekooperativní vyjednávání s alternativními nabídkami

Ve strategické vyjednávací teorii je proces vyjednávání modelován jako dynamická hra, ve které hráči vytvářejí nabídky a proti-nabídky. Tento proces končí, když jedna strana vyjednávání přijme nabídku druhé strany. Protože se hráči pohybují postupně, je odpovídající řešení koncept, který je perfektní sub-herní Nashovou rovnováhou.

Ve vyjednávacích modelech jsou strategiemi hráčů podíly, které tito hráči nabízí. Protinávrh reprezentuje iniciativu počkat na lepší nabídku od jiného hráče. V sub-herní perfektní Nashově rovnováze musí být jakákoliv protinabídka důvěryhodná. Pokud existuje limit omezující počet nabídek a protinabídek, tak je daná hra konečná. Konečné vyjednávací hry mohou být vyřešeny pomocí zpětné indukce propracovávajíc se od poslední nabídky. Extrémním případem konečné vyjednávací hry je hra s ultimátem. V této hře jeden hráč přednese počáteční nabídku a druhý buď přijme, nebo odmítne. Pokud druhý hráč prvního odmítne, tak oba obdrží nulovou výplatu.

1.6.3 Vyjednávací hry s netrpělivostí

Při výpočtu je také nutno brát v potaz netrpělivost účastníků dohody. Tato netrpělivost je v zásadě dána ztrátami, které hráč během jednání utrží. Ty mohou být přímé, nebo nepřímé. Přímá ztráta musí být hráčem vyplacena a nemá na ni vliv výsledek vyjednávání. U nepřímých ztrát se obvykle jedná o hodnotu objektu vyjednávání, jehož cena se s uběhlým časem snižuje. Rychlost, s jakou ke snižování dochází, muže být přímo úměrná počáteční hodnotě objektu. Tento typ ztráty také ze hry z principu dělá konečnou vyjednávají hru, protože pokaždé nastane chvíle, ve kterou bude cena objektu vyjednávání nulová a tím ztratí vyjednávání smysl.



Obrázek 6: Nepřímá ztráta, příklad

V případě přímé ztráty hráči obvykle objektu vyjednávání přidávají tím větší hodnotu, čím dříve ho mají. Příkladem by mohl být stroj, který sice časem svoji hodnotu neztrácí, ale pokud ho hráč získá, tak mu okamžitě začne generovat zisk. Tím pádem je v zájmu hráče nabídku přijmout co nejdříve.

Jako příklad nepřímé ztráty by se dala uvést elektronika, která velmi rychle po svém uvedení na trh ztrácí na hodnotě. V čase $t = 0$ je výrobcem navržena cena pr_m produktu p, na kterou v čase $t = 1$ dá prodejce odpověď. V případě přijmutí získá prodejce odměnu v podobě produktu p a výrobce v podobě peněžní částky pr_m. Pokud by ale nabídku odmítnul, ani prodejce ani výrobce nezískají nic. S využitím zpětné indukce je možné

dojít k rovnováze této hry. V rozhodovacím bodě R, je v zájmu prodejce nabídku přijmout v jakémkoliv případě, kdy je prodejní cena pr_s produktu p splňuje $pr_s - pr_m > 0$. [9]

2 TEORIE HER V E-COMMERCE

Aplikace herně-teoretických postupů na oblasti obchodu a ekonomiky jsou stejně staré jako matematická disciplína samotná. Pohnutky k tomu jsou celkem jasné. Interakce se zákazníky, obchodními partnery a konkurencí ve většině případů úzce korespondují s modely v teorii her. To je ještě obohaceno rozmachem informačních technologií a e-commerce. Zde se zákazník a obchodník dostávají do formálních stavů, které mohou být modelovány jako racionální hráči určitých her.

Herní teorií se v poslední době začíná zabývat stále více velkých českých i zahraničních internetových obchodů. Není divu, internetové obchody se jeví jako ideální skupina, na kterou je možné herní teorii aplikovat. Téměř každý e-shop, který funguje delší dobu, má k dispozici rozsáhlé množství dobře utříděných dat o svých zákaznících, jejich chování a reflexi cen na prodejnost produktů v podobě minulých objednávek. Přesto byla zatím prozkoumána pouze část možností využití, hlavně v oblasti inteligentního doporučování produktů zákazníkovi individuálně podle rozsáhlých statistik.

Zde ale tyto možnosti nekončí. Aplikací aparátu teorie her je také možné v kombinaci se statistickými metodami vytvořit komplexní strategické postupy, jejichž cílem je maximalizovat zisk. Ať už pomocí přestavovaní pořadí produktů podle skupin zákazníků, individuálních okamžitých slev (flash sales) či pomocí cílené manipulace cen.

Snahou této kapitoly bude aplikovat modely teorie her a postupy teoreticky zmíněné v předchozí kapitole na sféru e-commerce. Tím bude vytvořen teoretický základ, který napomůže při praktických výpočtech a vysvětlí mnohé postupy použité při vytváření knihovny v poslední části práce.

V první podkapitole budou definovány jednotlivé prvky, se kterými se bude dále pracovat. Dále dojde k vypracování matematických postupů pro tyto jednotlivé dílčí skupiny a k rozšíření na internetový obchod jako celek. Na konci dojde ke kombinaci s vybranými statistickými postupy, které umožní podrobné analýzy ve zbytku práce. V poslední části této kapitoly budou prozkoumány možnosti aplikace umělých neuronových sítí v kombinaci s herní teorií a statistikou.

2.1 Metodologie

V této kapitole budou specifikovány jednotlivé entity, se kterými bude dále pracováno. Hra v oblasti e-commerce je nejlépe definována jako nekonečně se opakující bayesovská hra, s kooperativním vyjednáváním. Nekonečně opakující se je hlavně z hlediska prodejce, který musí myslet na dlouhodobou výhru oproti krátkodobému zisku. To je důležité hlavně kvůli stálým zákazníkům, kteří jsou velmi volatilní a jakákoliv jim nevyhovující strategie bude znamenat jejich odchod ke konkurenci. Pro zákazníka je sice z definice tato hra také nekonečná, ale povětšinou bude docházet k jejich příchodu a odchodu. Jak bylo již řečeno, bude se jednat o hru s nekompletní informací a kooperativní vyjednávání se bude soustřeďovat kolem produktů a jejich cen.

Hráč

Dále v této práci bude hráčem myšlena taková entita, která se konečně či nekonečně bude účastnit hry v nějakém jejím kole. Soubor všech současných i minulých hráčů bude označen jako N. Hráč bude označen jako n, kde $n \in N$.

Zákazník

Zákazníkem bude hráč, jehož výplatou a výsledkem kooperativního vyjednávání bude produkt prodejce. Soubor všech hráčů bude označený jako C a jednotlivý hráč jako c s tím, že $c_i \in C$, a zároveň $C \subseteq N$, kde $i \in \{1, \ldots, n\}$.

Prodejce

Prodejce bude takový typ hráče, který se bude účastnit kooperativního vyjednávání se zákazníkem a jeho výplatou bude výdělek z produktu, který zákazník koupil. Každý prodejce bude označen písmenem d a soubor prodejců písmenem D. Každý $d \in D$,$D \subseteq N$, a zároveň $d_i \neq c_i$ pro všechna $i \in N$, kde $i \in \{1, \ldots, n\}$.

Bayesovský typ

Typ zákazníka Θ_c, typ prodejce Θ_d či zjednodušeně typ hráče Θ_i bude taková informace o předchozím průběhu hry, která bude pro daného hráče soukromá. Typem budou zpravi-

dla disponovat prodejci a bude se jednat hlavně o data týkajících se minulých kol, podle kterých budou prováděna budoucí rozhodnutí.

Produkt

Produktem se dále v této práci bude rozumět položka vyjednávání. I-tý produkt bude značen jako p_i. Platí, že $p_i \in P$, kde $P = \{p_i\}_{i=1}^{n}$ je soubor všech produktů ve hře, pro $i \in \{1, \ldots, n\}$. Každý produkt p_i bude disponovat cenou pr_i. Produkt p_i bude brán jako výplata zákazníka c.

Cena produktu

Cena pr_i produktu p_i je takovou částkou, kterou stanovil prodejce d, je na produkt aplikována dlouhodobě a splňuje $pr_i \in PR$, pro $i \in \{1, \ldots, n\}$, kde PR je množina všech cen. Tato cena bude také odrážet výplatu prodejce d_i. Speciálním případem je opr_{ij}, cena produktu p_i v době nákupu o_j.

Sleva produktu

Sleva ds_i produktu p_i je takovou částkou, která je na produkt aplikována striktně krátkodobě na časové období t, může být také aplikována pouze pro určitou skupinu zákazníků g, či dokonce pro individuálního zákazníka c. Přitom pro každý produkt $i \in \{1, \ldots, n\}$ platí, že $pr_i - ds_i \in PR$. Speciálním případe je ods_{ij}, sleva produktu p_i v době nákupu o_j.

Expirace produktu

Expirace exp_i produktu p_i je nezáporné celé číslo určující odhad obchodníka, za jak dlouhý časový úsek bude zákazník chtít daný produkt obnovit. Expirace je velmi důležitá u nespotřebního zboží, které zákazník s velkou pravděpodobností nebude kupovat opakovaně.

Doplněk produktu

Doplněk $pc_{ij} \in PC_j$ je takový produkt p_i, který obchodník považuje jako vhodný komplement produktu p_j. Například stůl k židli. Produkt p_j může mít více doplňků. Doplňky produktu nahrazují produkt jako takový v případě, že zákazník navštíví obchod ještě před jeho expirací.

Historie zákazníka

Historii zákazníka označujeme h_i, kde $h_i \in H$ je soubor všech tahů, které zákazník c_i doposud ve hře provedl. Historie je známá pouze prodejci d, a zákazníkovi c_i, jemuž tato historie náleží.

Parametry zákazníka

Parametry zákazníka pa_{c_i}, je množina hodnot, které ho identifikují. Není vyloučeno, že může nastat $pa_{c_i} = pa_{c_{i_j}}$, pro $i \neq j$. Skupina zákazníků G je charakterizována parametry všech zákazníků, kteří do ní náleží.

Nákup

Nákup or_i, kde $or_i \in OR$, pro $i \in \{1, \ldots, n\}$, je reprezentován množinou produktů, jejich zakoupeného počtu, cen, slev, zákazníkem a parametry nákupu $or_i = \{op, opr, opa, ods, c, pps\}$, kde $op \in P$, $opr \in PR$ a $ods \in DS$. Zároveň $or \in H$, i když se nemusí jednat o nákup dokončený a do výpočtu strategie být zařazen.

Parametry nákupu

Parametry nákupu pps_{ij} jsou otiskem parametrů zákazníka c_i v době vytvoření nákupu o_j.

Skupina zákazníků

Skupina zákazníků g_i je takovým souborem hráčů, že $g_i \subseteq C$. Skupina zákazníků se vyznačuje hlavně tím, že sdílí nějakou společnou vlastnost. Například geografické umístění, čas, či datum nákupu. Díky tomu je možné využít historii této skupiny h_i k předvídání chování zákazníka c_j, aniž by pro něj existovala jeho vlastní historie h_j.

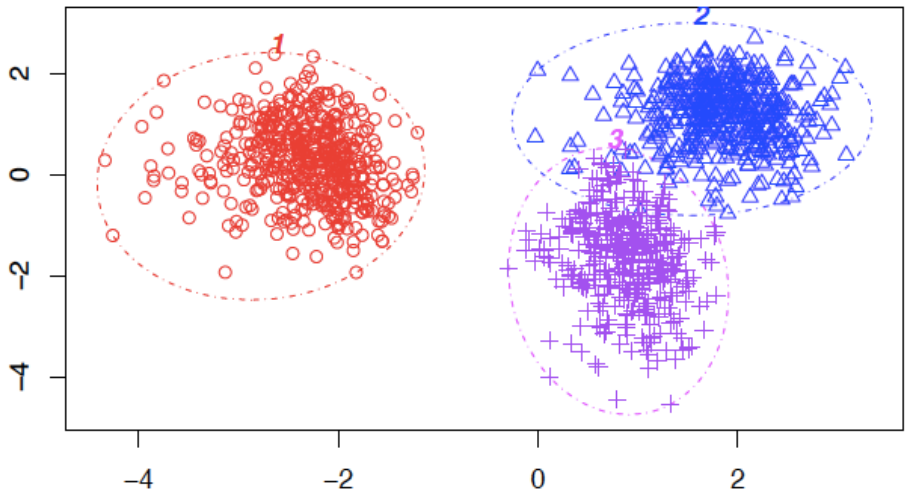
2.2 Dělení zákazníků do skupin

Jak bylo zmíněno v předchozí kapitole, v naprosté většině případů není možné vytvořit přípustnou strategii s využitím analýzy historie pouze jednoho zákazníka, či hráče. Toto je hlavně dáno nedostatečným souborem dat. V těchto případech je ideální rozdělit hráče

do větších či menších skupin na základě společných vlastností. Velikost skupiny bude dána prvořadě požadovanou hladinou významnosti.

2.2.1 Shluková analýza

Vzhledem k předem neznámému počtu parametrů se jako ideální metoda vytváření skupin jeví shluková analýza. Ta analyzuje skupiny objektů na základě informací vyskytujících se v datech popisujících dané objekty, nebo v jejich vztazích. Cílem je, že objekty ve skupině budou podobné, či příbuzné a odlišující se od objektů v jiných skupinách. Čím vyšší je homogenita ve skupině a rozdílnost mezi skupinami, tím je shlukování lepší.

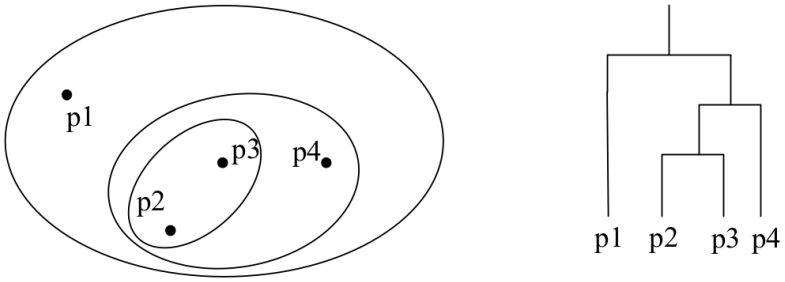


Obrázek 7: Vizualizace shlukové analýzy [10]

Většina shlukových analýz se snaží o co nejostřejší klasifikaci dat do jednotlivých nepřekrývajících se shluků. Základní typy technik shlukové analýzy se dělí na hierarchické a částečné.

Hierarchické techniky produkují sekvenci do sebe vnořených částí s jedním shlukem, který obsahuje všechny ostatní shluky a minimální shluky, které obsahují pouze jedinou hodnotu. Každá úroveň může být vnímána jako kombinace dvou shluků z nižší úrovně. Tyto shluky je možné popsat pomocí binárního stromu. Většina hierarchických algoritmů využívá právě kombinování, či naopak rozdělování shluků. [11]

Techniky využívající částečnou shlukovou analýzu produkují jednoúrovňové, nevnořené rozdělení datových bodů. Pokud je k požadovaný počet shluků, pak částečný přístup obvykle najde všechny k shluky najednou, na rozdíl od hierarchických schémat. Samozřejmě



Obrázek 8: Hierarchická shluková analýza [11]

při vícenásobném použití technik částečné shlukové analýzy je možné vytvořit víceúrovňové shluky.[11]

2.2.2 Výběr vhodných proměnných

Prvním krokem před aplikací algoritmů shlukové analýzy je výběr vhodných proměnných, podle kterých budeme zákazníky porovnávat. Shlukování s využitím adresy zákazníka je výhodné, ať už z důvodů ekonomických, či lokačních. Větším problémem v tomto případě je, že pro potřeby zařazení je nutné používat kvantitativní jednotky, které ale není tak jednoduché získat.

Další jednotkou, která se jeví jako velmi příhodná, je čas a datum, ve kterém zákazník obchod navštíví. To může o zákazníkovi prozradit ledacos. Obvykle bude velký rozdíl v tom, co lidé kupují v zimě před Vánoci, či o letních prázdninách. Podle hodiny, ve které zákazník obchod navštíví, je možné říci poměrně přesně, do jaké kategorie patří. Důchodce či matka na mateřské dovolené budou na rozdíl od pracujícího člověka pravděpodobněji nakupovat dopoledne. Školák zase ve většině případů nebude nakupovat pozdě večer. Nejvhodnější bude využít nejdříve kvantitativní jednotky měsíc příchodu 1-12, den v týdnu 1-7 a poté hodinu 0-24.

Přestože tyto hodnoty budou ve většině internetových obchodů sdílené, je možné, že u specifických implementací dojde k jejich rozšíření. Mohou například existovat obchody, ve kterých bude požadován věk klienta. V tomto případě můžeme lehce rozřadit zákazníky podle věkové kategorie a díky tomu jim přizpůsobit nabízené produkty.

U většiny internetových obchodů je také součástí nákupu registrace. Pokud se zákazník zaregistruje, bude nutné z jeho statistik vytvářet vážené průměry a ošetřit odlehlé hodnoty, protože to, že zákazník, který obvykle nakupuje dopoledne a jednou nakoupí o půlnoci,

je s největší pravděpodobností pouze izolovaný případ, který by mohl v případě malého množství dat ovlivňovat výpočty.

2.2.3 Očištění parametrů

Při opakovaných návštěvách internetového obchodu je možné daného zákazníka přesněji zařadit, ale jako u většiny statistických souborů vzniká nutnost data před zpracováním očistit. Vzhledem k povaze dat není nutné se při čištění parametrů zaobírat chybějícími hodnotami. Na druhou stranu existence extrémních hodnot je více než pravděpodobná. Zákazník nakupující kolem poledne si před Vánoci vzpomene, že zapomněl koupit dárek a objedná ho o půlnoci, nebo je na dovolené a objedná ho z jiné země.

Nákup	Měsíc	Den	Hodina	Zeměpisná délka	Zeměpisná šířka
pps_1	4	5	11	49.652456	16.259766
pps_2	6	6	12	49.652456	16.259766
pps_3	8	1	9	49.652456	16.259766
pps_4	10	2	13	35.320802	25.138551
pps_5	11	6	10	49.652456	16.259766
pps_6	12	1	23	49.652456	16.259766

Tabulka 1: Nákupy zákazníka s extrémy před očištěním

U cirkulárních hodnot, jako je hodina či měsíc v roce je nutné nejdříve provést transformaci na bod ve 2D prostoru pomocí $x_x = \sin(2 \cdot \pi \cdot x/z), x_y = \cos(2 \cdot \pi \cdot x/z)$, kde $1 \leq x \leq z$ [12], jinak by docházelo k chybám v dalších výpočtech, protože 1 hodina ráno je blíže 23 hodině večer, i když by to tak nebylo vyhodnoceno.

Nákup	Měsíc-X	Měsíc-Y	Den-X	Den-Y	Hodina-X	Hodina-Y	Zeměpisná délka	Zeměpisná šířka
pps_1	0,866	-0,5	-0,975	-0,223	0,259	-0,966	49,652456	16,259766
pps_2	0	-1	-0,782	0,623	0	-1	49,652456	16,259766
pps_3	-0,866	-0,5	0,782	0,623	0,707	-0,707	49,652456	16,259766
pps_4	-0,866	0,5	0,975	-0,223	-0,259	-0,966	35,320802	25,138551
pps_5	-0,5	0,866	-0,782	0,623	0,5	-0,866	49,652456	16,259766
pps_6	0	1	0,782	0,623	-0,259	0,966	49,652456	16,259766

Tabulka 2: Nákupy zákazníka s extrémy po očištění

Pro detekci odlehlých hodnot bude využito modifikované z-skóre (THE MODIFIED Z-SCORE). Tento přístup modifikuje standardní z-skóre, tak aby nebylo ovlivňováno vysokými extrémy. Modifikované z-skóre M_i pro hodnotu x_i bude vypočítáno následovně:

$$M_i = \frac{0,6745 \cdot (x_i - \tilde{x})}{MAD}$$

kde $\tilde{x}$ je medián datového souboru, a $MAD = median\{|x_i - \tilde{x}|\}$.

Iglewicz a Hoaglin (1993) doporučují, aby pozorování bylo označeno jako odlehlá hodnota, pokud $|M_i| > 3,5$. [13]

Problém s touto metodou nastává v případě, že více než 50% pozorování má shodnou hodnotu. V tom případě se MAD rovná 0 a rovnice selhává. IBM [14] v tomto případě doporučuje nahradit MAD (Median absolute deviation), pomocí $MeanAD$ (Mean absolute deviation), kde $MeanAD = \frac{\sum |x_i - \tilde{x}|}{n}$.

Vzhledem k povaze dat není vhodné tyto odlehlé hodnoty odříznout. Všechny hodnoty budou proto ohodnoceny váhou 1, která bude v případě odlehlých hodnot upravena o Hubertovu váhovou funkci, která je vypočítána následovně:

$$W_c(x) = min\{1, \frac{c}{|x - \hat{\mu}|}\}$$

kde c je ohybové kritérium, v tomto případě bude použito $c = 1,28$, které dává 90% asymptotickou efektivnost a je méně náchylné k datovému souboru kontaminovanému odlehlými hodnotami. $\hat{\mu}$, reprezentuje odhad lokace, který bude v tomto případě roven absolutní hodnotě mediánu $|\tilde{x}|$, pro $|\tilde{x}| \neq -x$ a $|\tilde{x}| + 1$, pro $|\tilde{x}| = -x$. [15] Z této rovnice vyplývá, že čím více se hodnota liší od mediánu, tím menší váha jí bude přiřazena.

Poté je z hodnot proveden vážený průměr a tím jsou získány finální parametry zákazníka.

Měsíc-X	Měsíc-Y	Den-X	Den-Y	Hodina-X	Hodina-Y	Zeměpisná délka	Zeměpisná šířka
-0,228	0,061	0	0,341	0,158	-0,59	49,401	16,509

Tabulka 3: Očištěné parametry zákazníka

2.2.4 Tvorba shluků

Vzhledem ke složitosti tvorby adekvátních shluků u velkých skupin dat bude nejvhodnější nevytvářet shluky dynamicky podle nových příchodů, ale vždy v nějakém časovém intervalu, ve kterém by došlo k zahrnutí všech entit, které byly do systému zařazeny v době od posledního přepočtu. V naprosté většině případů bude přepočet probíhat v čase nejmenší aktivity internetového obchodu, proto bude přechod na standardní týdenní interval proveden až po dosažení minimální velikosti datového souboru určeného při implementaci. Standardní interval bude nejdéle jednou za týden, přesto bude v případě anomálně velkého nárůstu dat možné provést kontrolu i dříve, ale nejvýše jednou za den. Tento anomální nárůst bude určen procentuální konstantou v závislosti na implementaci. Pokud bude tato konstanta zvolená jako 10%, bude to znamenat, že pokud se bude nárůst za předchozí den lišit o 10% od průměru, dojde k aktivaci předčasného přepočtu shluků. Tento průměr bude samozřejmě očištěn o 5% nejvyšších a nejnižších hodnot.

Vzhledem k tomu, že soubor dat bude vždy kvantitativního charakteru, bude jako metoda pro shlukování využit k-means++ algoritmus. K-means pracuje s maticí MxN, kde M bude dáno počtem zákazníků a N počtem parametrů. Nejdříve je nutné určit číslo K, které reprezentuje počet shluků, na které bude datový soubor rozdělen. To bude zadáno při implementaci.

C	p_1	p_2	p_2	p_4	p_5	p_6	p_7	p_8
c_1	-0,228	0,228	0,299	-0,54	0,158	-0,59	49,549	16,509
c_2	-0,083	-0,144	-0,536	0,178	0,029	-0,017	49,549	16,509
c_3	-0,144	0,417	-0,267	0,213	-0,507	-0,353	49,549	16,509
c_4	-0,122	0	-0,032	-0,308	-0,124	-0,328	49,549	16,509
c_5	0,35	0,561	0,122	0,213	-0,035	-0,096	49,549	16,509
c_6	-0,205	0,189	0,072	0,432	-0,515	0,201	49,549	16,509

Tabulka 4: Příklad matice pro k-means

- $p_{1,2}$ – Měsíc v roce,
- $p_{3,4}$ – Den v týdnu.
- $p_{5,6}$ – Hodina ve dni.
- p_7 – Zeměpisná šířka.
- p_8 – Zeměpisná délka.

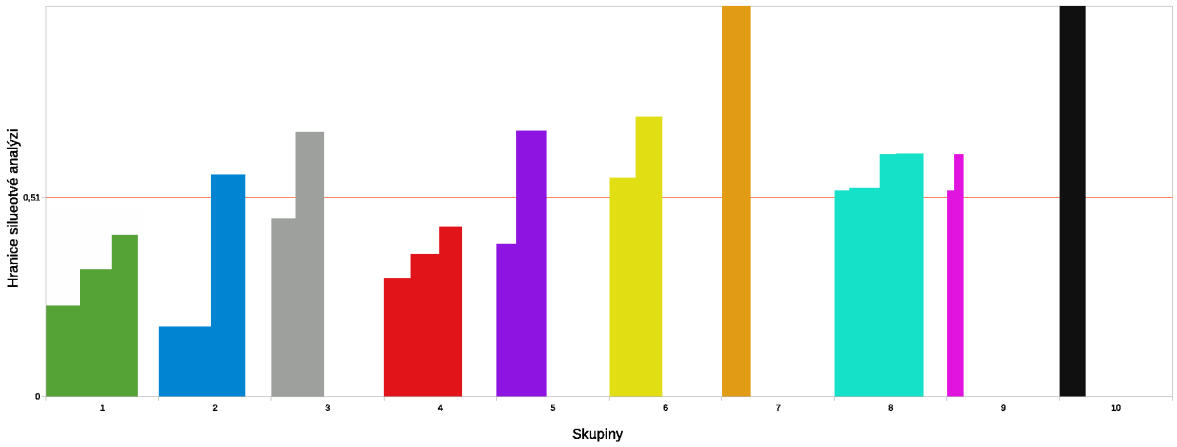
K-means++ je rozšíření algoritmu k-means, za pomoci kterého je získán algoritmus, který je $O(logk)$ kompetitivní k optimálnímu řešení k-means shlukování.[16] Definice k-means++ je následující:

Nechť je $D(x)$ nejkratší vzdáleností z bodu do nejbližšího již zvoleného centra. Algoritmus k-means++ poté pracuje následovně:

1. Vybrání počátečních center $GC = \{gc_1, gc_2, \ldots, gc_k\}$.
   a) Náhodné vybrání jednoho centra gc_1 z X.
   b) Vytvoření nového c_i centra, výběrem z X s pravděpodobností $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$.
   c) Opakování kroku b, dokuď není vytvořeno k center.
2. Pro každé $i \in \{1, ..., k\}$, je nastaven shluk G_i skládající se z takových zákazníků C, že jsou blíže k gc_i než k gc_j pro každé $j \neq i$.
3. Pro každé $i \in \{1, ..., k\}$ je nastaven gc_i centroidem všech bodů $GC_i : gc_i = \frac{1}{|GC_i|} \cdot \sum_{x \in GC_i} x$.
4. Kroky 2 a 3 jsou opakovány, dokud se GC nepřestane měnit. [16]

2.2.5 Optimalizace počtu shluků

Samozřejmě v případě dynamicky rostoucích dat není výhodné, aby bylo k statické, proto bude v tomto případě k reprezentovat pouze počáteční minimální počet shluků. Ten bude nadále upravován s využitím siluetové analýzy.



Obrázek 9: Příklad siluetové analýzy

Optimální počet shluků je v siluetové analýze určen tzv. siluetovým indexem. Nejdříve při implementaci dojde k nastavení počátečního počtu shluků K_s. Hodnota K_s bude z počátku pouze odhadem počátečního počtu shluků a ve většině případů bude naprosto do-

stačující $K_s = 1$. Pouze u již déle fungujících internetových obchodů s rozsáhlou množinou zákazníků bude vhodné určit vyšší hodnotu K_s a tím snížit výpočetní náročnost prvotní shlukové analýzy. Definice siluetového indexu je založena na siluetách od Rousseeuw [17], které jsou konstruovány jako grafická pomůcka značící jak dobře je každý objekt zařazen od shluku.[18] Siluetový index $s(m)$ pro shluk G_m je počítán následovně:

1. Jestliže c_i tvoří samostatný shluk, je položeno $s_i = 0$ a výpočet je ukončen. V ostatních případech se pokračuje kroky 2 až 5.
2. Je vypočten $a_i = \frac{1}{\|G_m\|-1} \cdot \sum_{i'=1, i' \neq i}^{N} d(c_i, c_{i'})$, pro $c_i, c_{i'} \in G_m$, kde $\|G_m\|$ značí počet prvků v daném shluku. Tedy průměrnou vzdálenost mezi zákazníkem c_i a všemi dalšími zákazníky $c_{i'}$, se kterými je c_i ve stejném shluku.
3. Pro všechna $c_j \in \{1, \ldots, n\}$ taková, že $c_j \in G_k$, pro $k \in \{1, \ldots, n\}$, kde $k \neq m$ je vypočteno $d(i, G_k) = \frac{1}{G_k} \cdot \sum_{j:G_k} d(c_i, c_j)$, tedy vzdálenost c_i ke všem c, ze shluku G_k, kde $k \neq m$.
4. $b_i = \min_{k \neq m} d(i, G_k)$.
5. $s_i = \frac{b_i - a_i}{max\{a_i, b_i\}}$.

Rozmezí s_i	Interpretace
0,71 až 1,00	c_i se silně váže k danému shluku
0,51 až 0,71	c_i je dobře zařazeno do shluku
0,26 až 0,51	c_i se slabě váže k danému shluku
-1,00 až 0,25	c_i je pravděpodobně slabě zařazeno

Tabulka 5: Interpretace hodnot siluetové funkce [19]

Dále je definována průměrná šíře rozkladu jako

$$w_K := \frac{1}{N} \cdot \sum_{i=1}^{N} s(i)$$

Siluetová analýza je poté prováděna pro $K+N$, dokud nezačne docházet ke snižování w_K. Optimální počet shluků K^* se proto rovná $K^* = \underset{\{K=2,3,\ldots,N-1\}}{agrmax} w_K$[19]

Vzhledem k povaze dat, u kterých se předpokládá pouze růst, se $K = K^*$ pro GC_i pro všechny další výpočty. Pouze v případě, že $GC_i < GC_{i+1}$ dojde k nastavení K na K_s a přepočítání shluků.

2.2.6 Zařazení zákazníka do shluku

Pokud přijde nový neznámý zákazník, dojde k očištění jeho parametrů na základě kapitoly 2.3.4 s tím, že nebude nutné počítat vážené průměry, protože je k dispozici pouze jeden záznam. Dále mu bude přiřazen shluk tak, aby druhá mocnina odchylek byla napříč všemi dimenzemi co nejmenší: Nechť je c zákazníkem a GC jsou aktuální centra shluků, pak je cílem pro každého zákazníka $c \in C$:

$$min \sum_{i=1}^{d} (c_i - gc_i)^2$$

kde $gc_i \in GC$. V případě, že bude více vhodných center c, bude zákazník zařazen podle pravidla „first best“, tedy prvního nejlepšího shluku. Po přiřazení shluk zákazníkovi zůstává až do chvíle, než obchod opustí nebo dojde ke změně proměnných.

2.3 Určení strategie

Pokud je již známá historie zákazníka, či předpokládaná historie zákazníka, je možné určit jeho pravděpodobnou strategii. Hlavním cílem strategie zákazníka bude uspokojit svou potřebu a při tom utratit co nejméně peněz. Na druhou stranu cílem strategie prodejce bude maximalizovat zisk. Je zřejmě, že se tyto dvě strategie budou navzájem v mnoha případech podstatně lišit.

V následujících kapitolách bude popsán způsob vytváření strategie zákazníka a prodejce. Poté bude hlavním cílem nalézt mezi těmito dvěma soubory strategií Nashovu rovnováhu tak, aby došlo k maximalizaci zisku a zároveň k uspokojení potřeb zákazníka.

2.3.1 Strategie zákazníka

Vzhledem k znalosti všech předchozích rozhodnutí zákazníka je na jejich základě možné určit jeho pravděpodobnou strategii. Tato strategie se samozřejmě v průběhu zákazníkova pobytu v internetovém obchodu bude měnit. Pokud například zákazník vloží do košíku produkt p_3, bude mnohem pravděpodobnější, že jeho strategie bude korespondovat s některým z předchozích nákupů, ve kterých došlo k výběru tohoto produktu.

Seřazení produktů v dané strategii bude realizováno pomocí vah, které se budou měnit na základě situací zmíněných výše. Při implementaci dojde k určení koeficientu, kterým se

budou násobit hodnoty tak, aby docházelo k upřednostňování vlastní historie zákazníka, pokud nějaká existuje, před historií jeho skupiny. Zároveň ale bude v případě, že zákazník nebude mít žádnou historii produktu, ale jeho shluk ano, brána pouze historie shluku.

Pokud by například internetový obchod prodávající produkty $P = \{p_1; p_2; p_3; p_4\}$ měl násobící koeficient $x = 2$, nákupy $OR = \{or_1, \ldots, or_{11}\}$ a historii zákazníka:

	Počet p_1	Počet p_2	Počet p_3	Počet p_4
or_1	0	2	1	0
or_2	3	0	1	0
or_3	3	1	2	0
or_4	0	1	0	0

Tabulka 6: Ukázková historie nákupů zákazníka

kde produkty by byly oceněny podle hodnot v tabulce 7:

	Cena p_1	Cena p_2	Cena p_3	Cena p_4
or_1	600	300	900	1500
or_2	400	400	900	1500
or_3	600	300	900	1500
or_4	600	250	900	1100

Tabulka 7: Ceny produktů u historie nákupů zákazníka

Tabulka 8 ukazuje historii objednávek shluku daného zákazníka bez jeho vlastních objednávek:

	Počet p_1	Počet p_2	Počet p_2	Počet p_4
Počet or_5	2	3	2	0
or_6	0	1	1	1
or_7	2	0	0	0
or_8	0	3	0	2
or_9	3	3	0	1
or_{10}	0	3	0	2
or_{11}	0	1	0	0

Tabulka 8: Ukázková historie nákupů zákazníkova shluku

kde hodnoty produktů při daných nákupech jsou zobrazené v následující tabulce:

	Cena p_1	Cena p_2	Cena p_2	Cena p_4
or_5	600	300	750	1500
or_6	600	300	750	1500
or_7	400	400	900	1500
or_8	600	300	900	1500
or_9	600	250	900	1100
or_{10}	600	250	900	1500
or_{11}	600	250	900	1100

Tabulka 9: Ceny produktů u historie nákupů zákazníkova shluku

Každý produkt má také definovaný čas expirace. Ten má za úkol při výpočtu strategie zákazníka zohlednit fakt, že ne každé zboží je spotřebního charakteru. Vyřazení neexpirovaného produktu ze strategie však není vhodné, proto bude s neexpirovaným produktem zacházeno jako s produktem koupeným zákazníkovou skupinou. To znamená, že produkt, jenž byl koupen dříve, než vyprší expirace tohoto produktu, nebude při výpočtu v tomto daném nákupu násoben koeficientem x. Zároveň ale budou zvýhodněny doplňky tohoto produktu. Zvýhodnění proběhne tak, že k danému nákupu budou přičteny hodnoty vypočítané pro $p_j \in PC_i$: $opa_{jnew} = opa_j + opa_i \cdot max(1, \frac{x}{2})$, kde $i, j \in \{1, \ldots, n\}$, $i \neq j$.

Pokud by byl produkt p_2 v objednávce or_3 neexpirovaný a jeho doplňkem by byl produkt p_4, pak by tabulka nákupů zákazníka po vynásobení koeficientem x a zvýhodněním doplňků vypadala následovně:

	Počet p_1	Počet p_2	Počet p_3	Počet p_4
or_1	0	4	2	0
or_2	6	0	2	0
or_3	6	1	4	1
or_4	0	2	0	0

Tabulka 10: Ukázková historie nákupů zákazníka po vynásobení koeficientem x a zařazením doplňků

Ceny produktů v době nákupu or_j poslouží jako váhy při výpočtu průměru. Váhy W_j je nutné nastavit tak, aby došlo k preferování těch hodnot nákupů produktů, kde se cena blíží co nejvíce ceně aktuální pr_i. Protože s nižší cenou je pravděpodobnější, že zákazník

daný produkt koupí, musí také dojít k upřednostňování těch nákupů, kde byla cena vyšší, než je cena aktuální. Aby w_i, kde $w_i \in W_j$ měla takovouto vypovídající hodnotu, je nutné provést přepočet ceny *opr* a její normalizaci na základě následujícího vzorce:

$w_i = \frac{1}{1+(\tilde{pr}_i - \tilde{opr}_{ij})}$, kde $\tilde{pr}_i = \frac{pr_i}{||y||}$, $\tilde{opr}_{ij} = \frac{opr_{ij}}{||y||}$ a $||y|| = \sqrt{\sum_{j=1}^{n} opr_{ij}^2 + pr_i^2}$, pro takové produkty p_i v objednávce or_j, pro které platí $opa_{ij} > 0$, pro $i, j \in \{1, \ldots, n\}$ [20]

Jako poslední krok dojde k vypočtení váženého průměru hodnot nákupů, kde jako váhy poslouží právě ceny produktů, tak aby pro každý produkt, který je momentálně v prodeji, připadala právě jedna hodnota. Díky vahám, bude do výpočtu zařazena i cena produktu, kterou měl v době nákupu.

$$P(pr_i) = \sum_{j=1}^{n} w_i \cdot opa_{ij}$$

Pokud by by aktuální ceny produktů odpovídaly hodnotám v následující tabulce:

	Cena p_1	Cena p_2	Cena p_2	Cena p_4
pr	600	250	900	1100

Tabulka 11: Aktuální ceny produktů

Výsledná tabulka hodnot bude vypadat následovně:

p_1	p_2	p_2	p_4
0,319	0,386	0,156	0,139

Tabulka 12: Výsledné nákupy zákazníka a jeho shluku

Tyto hodnoty budou aktualizovány v průběhu nákupu zákazníka. Pokud by například vložil do košíku produkt jednou p_1, došlo by k jeho odebrání z výpočtu a zároveň by došlo k vynásobení všech předchozích nákupů koeficientem $oc = max(1; x \cdot ps)$, kde ps je počet produktů, které v předchozím nákupu odpovídají produktům, které má zákazník momentálně v košíku. To znamená, že pokud je momentálně v košíku jednou vybrán produkt p_1 a historický nákup také jednou p_1 obsahuje a $x = 2$ bude $oc = max(1; 2 \cdot 1)$. Pokud by měl zákazník vybraný jednou produkt p_1 a dvakrát produkt p_3 a historický nákup by také obsahoval jednou p_1 a dvakrát p_3, pak by $oc = max(1; 2 \cdot 3)$. Historie nákupů zákazníka jako takového by nadále byla ještě k tomu násobena koeficientem x. Při tomto příkladu a vybrání jednoho produktu p_1 do košíku by bylo nadále počítáno s předchozími tabulkami následovně:

	Počet p_2	Počet p_3	Počet p_4
or_1	4	2	0
or_2	0	4	0
or_3	2	8	2
or_4	2	0	0

Tabulka 13: Ukázková historie nákupů zákazníka po vynásobení koeficientem x a oc

	Počet p_2	Počet p_2	Počet p_4
or_5	6	4	0
or_6	1	1	1
or_7	0	0	0
or_8	3	0	2
or_9	6	0	2
or_{10}	3	0	2
or_{11}	1	0	0

Tabulka 14: Ukázková historie nákupů zákazníkova shluku po vynásobení koeficientem oc

2.3.2 Strategie prodejce

Strategie prodejce bude na rozdíl od strategie zákazníka podstatně přímočařejší. Hlavním ukazatelem bude, jakou výplatu prodej daného produktu nabízí. Tímto ukazatelem bude koeficient výplaty produktu (Product Payoff Coefficient), PPC. Bude se jednat o nezáporné celé číslo, jehož výše bude odpovídat výplatě při prodeji daného produktu. Výpočet PPC bude velmi různorodý na základě implementace. V některých případech do něj bude nutné zahrnout expiraci produktu, množství skladových zásob a možné změny na trhu. Jindy bude tímto ukazatelem pouhá výše výdělku. Expirace se ale nemusí vztahovat pouze na produkty, které budou po jejím datu nepoužitelné jako například potraviny. Může se také jednat o produkt, u něhož je očekáváno uvedení nové řady, která zapříčiní značné znehodnocení daného zboží.

V mnoha případech bude možné učinit produkt pro zákazníka atraktivnějším, aniž by významně klesla jeho hodnota PPC. Hlavním cílem strategie prodejce bude vytvořit Nashovu rovnováhu takovou, aby bylo PPC co nejvyšší.

Řekněme, že produkty na internetovém obchodě jsou reprezentovány následující tabulkou:

	p_1	p_2	p_2	p_4
Výsledná cena	600	250	900	1100
Výdělek	420	180	150	400
Počet na skladě	120	15	40	10
Skladová hladina	30	60	10	5

Tabulka 15: Produkty internetového obchodu

PPC bude vypočítáno následujícím způsobem. Jako základ bude vybrán prostý výdělek na produktu a pokud je na skladě méně produktů, než je dáno skladovou hladinou, dojde ke snížení PPC o 30%. Tím bude zaručeno, že nedojde k nechtěnému nabízení produktů, u nichž by v dohledné době mohlo dojít k vyprodání. Tomu je velmi důležité předejít právě u nekonečných her. Přestože by z krátkodobého hlediska mohl prodej daného produktu přinést vyšší výplatu, po jeho vyprodání by mohlo dojít k odlivu zákazníků ke konkurenci, a tím z dlouhodobějšího pohledu přinést menší výplatu. Proto bude tímto krokem produktu uměle snížena hodnota produktu a nedojde k jeho zbytečnému zviditelňování, dokud nebude stav skladových zásob obnoven.

Na základě předchozích údajů bude PPC pro tento případ vypadat následovně:

	p_1	p_2	p_2	p_4
PPC	420	126	150	400

Tabulka 16: Výsledné PPC

Následně je vypočten poměr v jakém je dané PPC zastoupeno tak, aby strategie prodejce s předchozími daty vypadala následovně:

p_1	p_2	p_3	p_4
0.383	0,115	0,137	0,365

Tabulka 17: Strategie prodejce

Na rozdíl od strategie zákazníka se strategie prodejce v průběhu nákupu nemění. Dochází pouze k odebírání produktů, které má zákazník již v košíku.

2.3.3 Rovnovážná strategie

Pro výpočet Nashovy rovnováhy je nejdříve nutné dosadit strategie zákazníka a prodejce do následující matice:

Prodejce	Zákazník			
	0,319	0,386	0,156	0,139
0,383	0,383;0,319	0,383;0,475	0,383;0,156	0,383;0,139
0,115	0,115;0,219	0,115;0,386	0,115;0,156	0,115;0,139
0,137	0,137;0,219	0,137;0,386	0,137;0,156	0,137;0,139
0,365	0,365;0,219	0,365;0,386	0,365;0,156	0,365;0,139

Tabulka 18: Strategie zákazníka a prodejce

Je nutné brát v úvahu, že ať je strategie prodejce jakákoliv, vždy výsledek volí zákazník, proto je možné pracovat pouze s hodnotami na diagonále tabulky. V tomto případě by strategií zákazníka byl nákup produktu p_2 následovaný produktem p_1.

Prodejce	Zákazník			
	0,319	0,386	0,156	0,139
0,383	0,383;0,319			
0,115		0,115;0,386		
0,137			0,137;0,156	
0,365				0,365;0,139

Tabulka 19: Strategie zákazníka a prodejce, vyznačena dominantní strategie

Je ale zřejmé, že by takový výsledek byl pro prodejce nevýhodný z hlediska jeho vlastní strategie, proto budou vynásobením hodnot strategie zákazníka a strategie prodejce v tabulce získány nové hodnoty. Ty budou sloužit jako metrika, která s respektem ke strategii prodejce určí pořadí produktů, v jakém je nejvýhodnější je zákazníkovi prodat, tak aby byla zároveň co nejlépe zachována předpovězená strategie daného zákazníka. V tomto případě, by tato tabulka vypadala následovně:

Pořadí	1	2	3	4
Produkt	p_1	p_2	p_4	p_3
Váha	0,122	0,044	0,021	0,051

Tabulka 20: Strategie zákazníka a prodejce

Interpretace této tabulky pak záleží na implementaci. Ve většině případů ji bude možné využít jako vodítko, s jakou agresivitou daný produkt zákazníkovi nabízet. Může se jednat o zvýhodňování při vyhledávání či přednostní pozici na úvodní stránce.

Samozřejmě není nutné využívat pouze pasivní metody. Přestože prodejce nemůže na základě své strategie přímo ovlivňovat strategii zákazníka, může změnit samotná pravidla hry. Hlavním nástrojem pro tyto změny je cena produktu. V případě, že zákazník nevyžaduje přímo jeden daný produkt při jeho nákupu, je možné aplikováním správných slev změnit jeho strategii, bez nutnosti markantně měnit strategii prodejce.

Produkty, na které bude možné tyto změny aplikovat, budou hlavně ty, u kterých se podstatně liší strategie prodejce, ale nijak zásadně strategie zákazníka. Dobrým kandidátem z předchozích příkladů je produkt p_1. Pro zákazníka je na druhém místě jeho strategie za produktem p_2 a pro prodejce by bylo značně výhodnější, ho vzhledem k jeho vlastní strategii prodat, oproti produktu p_2.

K tomu je nejdříve nutné určit kandidátní produkt či produkty, a to takové produkty, u kterých mají pokusy aplikací slev změnit strategii zákazníka smysl. V tomto případě, je vhodným produktem právě produkt p_1, naopak produkty p_3 a p_4 jsou značně nevhodné, protože jsou ve strategii zákazníka položeny extrémně nízko.

Pokud by se zachovaly hodnoty představené v předchozích kapitolách, musel by prodejce nabídnout zákazníkovi produkt p_1 s 41% slevou, aby došlo v zákazníkově strategii k upřednostnění produktu p_1 oproti produktu p_2. Po změně původních hodnot PPC v závislosti na navrhované slevě by jejich tabulka spolu s tabulkou strategie zákazníka vypadala následovně:

p_1	p_2	p_2	p_4
0,383	0,365	0,137	0,115

Tabulka 21: Strategie prodejce po aplikování konzervativní slevy

p_1	p_2	p_2	p_4
0,363	0,362	0,146	0,130

Tabulka 22: Strategie zákazníka po aplikování konzervativní slevy

Z tabulky je patrné, že i při tak masivní slevě bude pro prodejce podstatně výhodnější prodat produkt p_1. Samozřejmě v závislosti na rozhodnutí prodejce je možné slevy aplikovat i jiným než takto konzervativním způsobem. Přestože byla aplikována značná sleva, je možné vzhledem ke strategii prodejce tuto slevu dále navyšovat.

Z tohoto důvodu bude k dispozici agresivní politika vynucování strategie prodejce. Její využití bude u kandidátních produktů, aby se při zachování pořadí ve strategii prodejce co nejmarkantněji změnil poměr ve strategii zákazníka.

V tomto případě je možné na produkt p_1 nasadit 48% slevu, aniž by došlo ke snížení jeho strategické hodnoty pod hodnotu produktu p_2.

p_1	p_2	p_2	p_4
0,163	0,156	0,186	0,495

Tabulka 23: Strategie prodejce po aplikování agresivní slevy

p_4	p_1	p_3	p_2
0,471	0,205	0,176	0,148

Tabulka 24: Strategie zákazníka po aplikování agresivní slevy

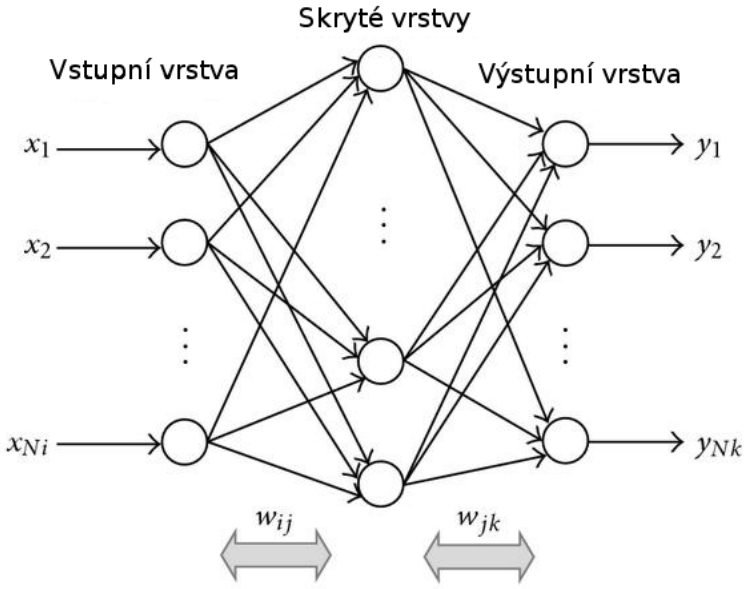
Samozřejmě při aplikování takto vysoké slevy markantně klesá výhodnost daného produktu pro prodejce, proto není možné takovouto strategii ve většině případů doporučit. Přesto se mohou najít případy, ve kterých bude vhodné agresivní slevu nasadit a proto bude tato strategie volitelně dostupná.

V případě konzervativní resp. agresivní slevy, bude dále pracováno pouze se strategií zákazníka. Pokud by byl použit stejný postup jako u výpočtu strategií bez aplikování slev, docházelo by k znehodnocování výsledků, protože produkty po slevě ztrácí ve strategii prodejce markantně hodnotu a do popředí by se tak dostávaly produkty, které jsou pro zákazníka nezajímavé.

Zároveň je strategie prodejce zhodnocena již při výpočtu slev, tudíž není nutné ji dále využívat. Výsledkem bude tedy pouze pořadí produktů založené na předpovězené strategii zákazníka s danými slevami.

2.4 Využití umělých neuronových sítí

Přestože jsou umělé neuronové sítě téma dalece mimo rozsah této práce, je vhodné se o jejich obrovském potenciálu alespoň částečně zmínit. Zároveň se jedná i o poměrně logický další krok v tomto tématu. Nejzajímavější jsou neuronové sítě s učením pomocí zpětné propagace. Ty jsou momentálně nejvyužívanějším typem umělých neuronových sítí. Skládají se z plně vzájemně propojených vrstev pracovních jednotek, kde mimo dopředných spojení každá jednotka v každé skryté vrstvě dostává „chybovou vazbu“ ze spoje jednotek ve vrstvě nad ní. Učení sítě probíhá v cyklech, kde každý cyklus sestává z „probublávání“ vstupů sítě z nejnižší do nejvyšší skryté vrstvy a poté následného „prosakování“ z nejvyšší do nejnižší skryté vrstvy. [22]



Obrázek 10: Schématický diagram obecné umělé neuronové sítě se zpětnou propagací [21]

Díky tomuto principu je možné nastavit očekávané výstupy a hodnoty neuronové sítě budou měněny tak, aby se reálný výsledek co nejvíce přiblížil danému cíli.

Jako příklad se nabízí autonomní správa cen produktů v internetovém obchodu. Při implementaci by bylo zadáno pouze rozmezí, ve kterém se cena produktu může pohybovat

a pomocí umělých neuronových sítí by docházelo ke zlevňování či zdražování produktu s cílem co nejvyššího zisku. Je také zřejmé, že by bylo možné využít podobného principu při vytváření PPC či při jiných postupech probraných v této práci.

Snahou proto bude ne přímo umělé neuronové sítě implementovat, ale přesto při vytváření samotné praktické knihovny připravit základ, na který by bylo možné v budoucnu sítě nasadit.

3 IMPLEMENTACE

V této kapitole bude popsán proces zpracování teoretického základu z předchozích kapitol do funkční knihovny. K tomu bude využit především modelovací jazyk UML, zároveň budou podrobněji popsány sekce, které byly natolik stěžejní, že bylo vhodné jejich rozvedení nad rámec obsahu uživatelského manuálu v příloze.

Cílem této knihovny je co nejsnazší koncové nasazení a kompatibilita. Vzhledem k těmto požadavkům bude využito jazyka PHP, protože naprostá většina internetových obchodů, ať už přímo nebo ve formě nějakého frameworku, je v tomto jazyce napsána. Hlavní verzí tohoto jazyka při vývoji bude 7.1. Nejnižší vyžadovanou verzí, u které bude zaručena plná kompatibilita, bude ale již verze 5.6. Ta je totiž jako LTS podporována až do prosince 2018 a je stále velmi rozšířená.

Knihovna bude implementována ve formě Composer balíku a s dodržováním PHP autoloading standardu PSR-4, který composer implementuje [23]. Zdrojový kód bude publikován s využitím hostovací služby GitHub a verzovazího serveru Git [25]. Kompletní knihovna bude také k dispozici z Composer repozitáře Packagist, který umožňuje velmi jednoduchou distribuci PHP software.

```
{
  "name": "okomarek/ecgm",
  "license": "MIT",
  "type": "library",
  "description": "Application of game theory to e-commerce solutions",
  "autoload": {
    "psr-4": {
      "ECGM\\": "src/ECGM",
      "ECGM\\tests\\": "tests"
    }
  },
  "require": {
    "php": ">=5.6"
  },
  "require-dev": {
    "phpunit/phpunit": "5.*"
  }
}
```

Obrázek 11: composer.json knihovny

Jak je zřejmé z předchozích kapitol, bude nutné v knihovně používat poměrně složité matematické postupy. Ty ale mají výhodu v relativně dobře předvídatelných výstupech, proto bude její součástí testovací prostředí založené na frameworku PHPUnit ve verzi 5. Tuto starší verzi bylo nutné použít z důvodu již zmíněné zpětné kompatibility s PHP 5.6.

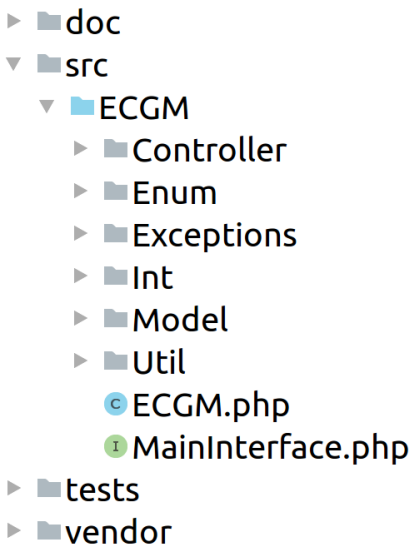
Jako základní architektura projektu bude využito MVC. Ta je pravděpodobně nejrozšířenější architekturou vůbec. MVC také poskytuje mnoho výhod, z nich nejdůležitější je

přehlednost a jasně daná struktura. Tyto vlastnosti jsou velmi důležité u aplikací, které mají za úkol sloužit jako knihovny a je předpoklad, že je budou uživatelé chtít rozšiřovat. [24] Zároveň je na architektuře MVC založen framework Symfony, na kterém je postaven internetový obchod, který sloužil jako testovací platforma této práce.

Kompletní aplikační část společně s veškerou dokumentací bude publikována pod MIT licencí, která je bezesporu nejvyužívanější a nejjednodušší z rodiny OpenSource licencí. Umožňuje naprosto volné šíření, modifikování a distribuování software. [26] Tato licence je použita hlavně z důvodu, že téma, kterým se tato práce zaobírá, je tak rozsáhlé, že budoucí rozšiřování a modifikace je velmi žádanou vlastností. Zároveň je díky této licenci umožněno tuto aplikaci bez jakýchkoliv zábran komerčně využívat.

3.1 Struktura

V této kapitole bude povrchově popsána struktura složek knihovny. Funkce jednotlivých souborů bude podrobněji popsána níže. Knihovna je kvůli přehlednosti rozdělena na několik složek a podsložek. Jejich hierarchie vychází hlavně z již existujících knihoven a neoficiálních standardů pro composer balíky. Přestože nejsou nikde publikovány, většina významných projektů se tímto rozdělením řídí.



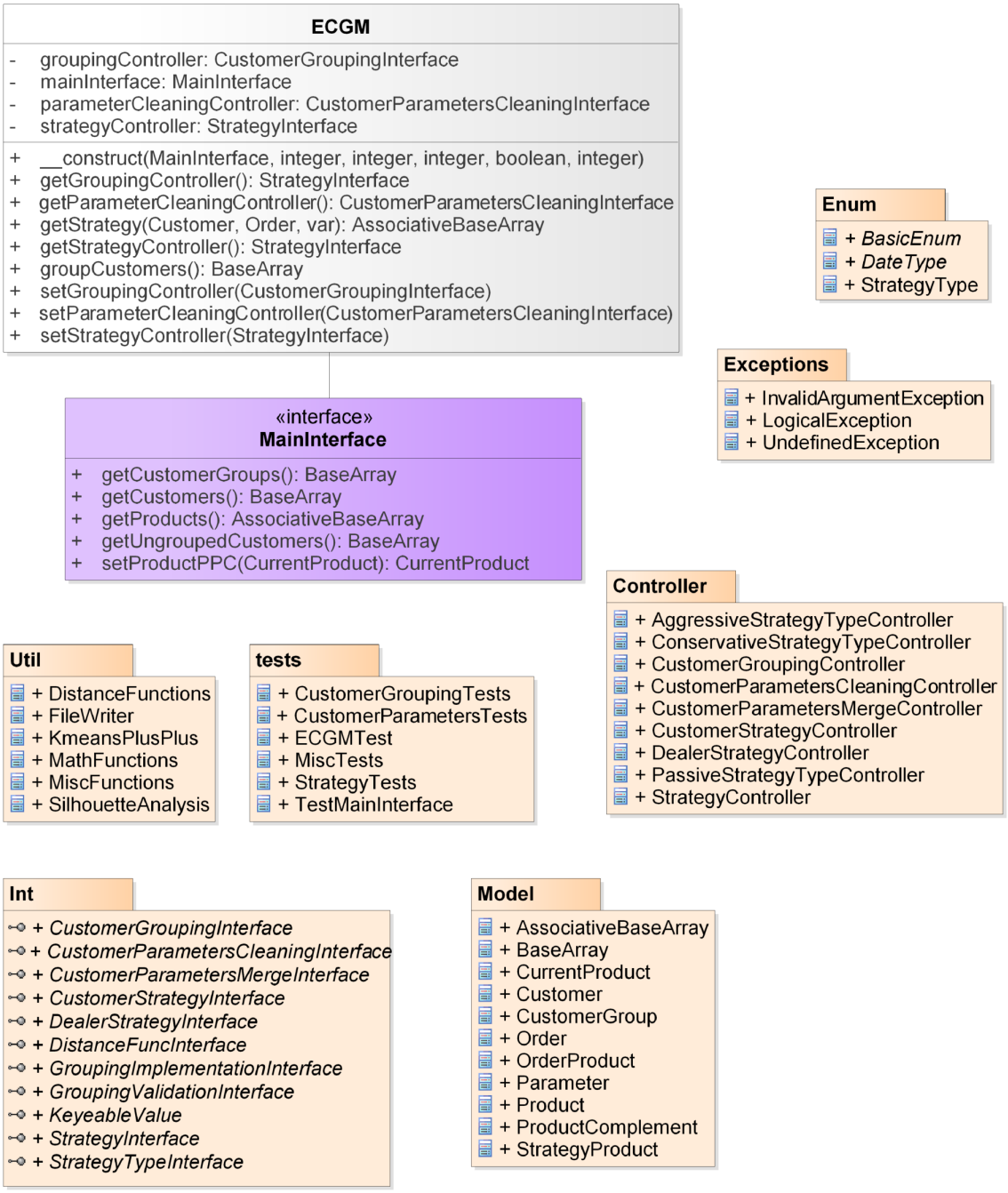
Obrázek 12: Struktura složek knihovny

- doc – Složka obsahující potřebnou dokumentaci projektu.
- src/ECMG – Složka obsahující všechny soubory knihovny.

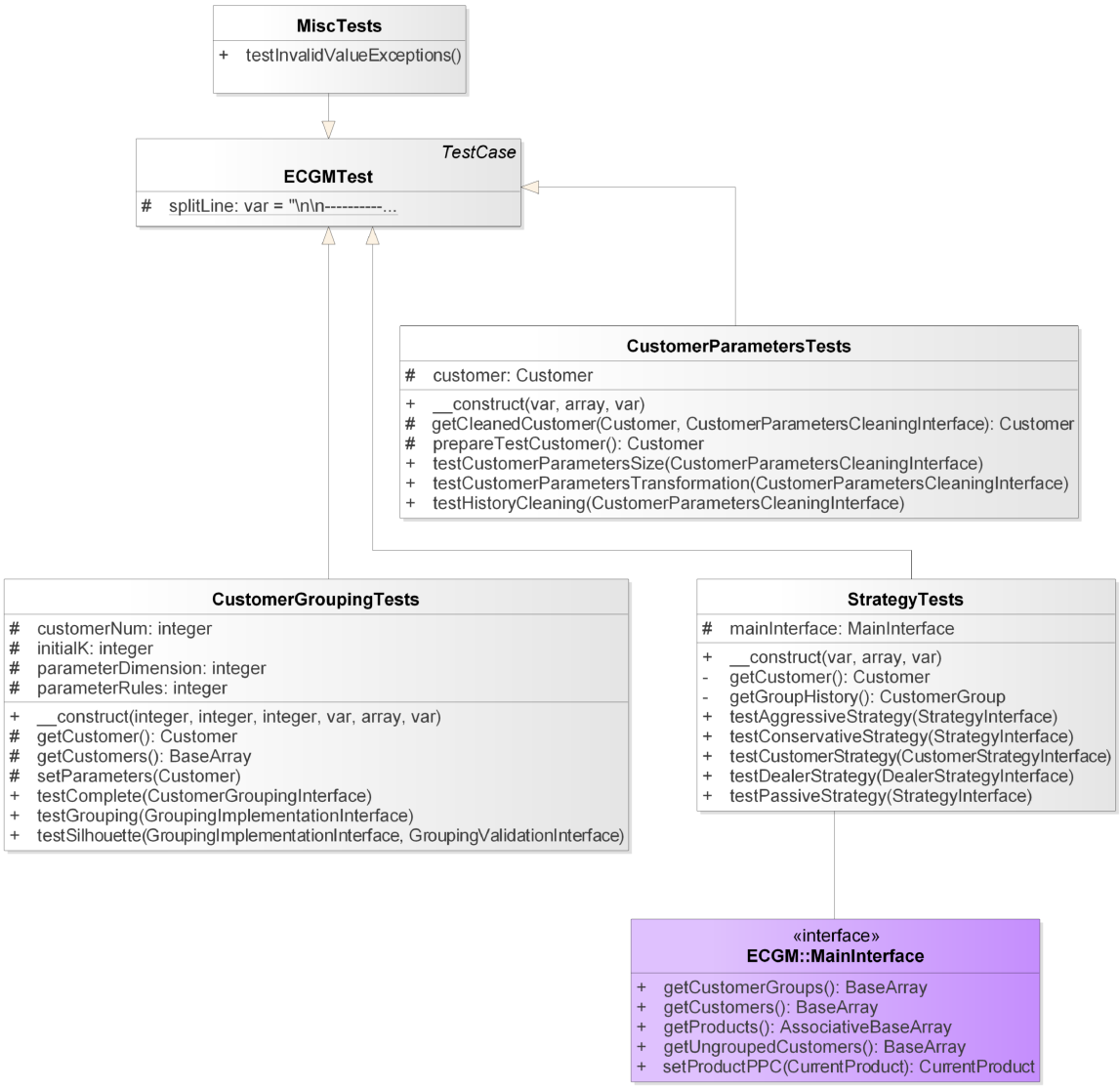
- src/ECMG/Controller – Složka obsahující většinu logických funkcí určeních přímo pro tuto knihovnu.
- src/ECMG/Enum – Složka obsahující výčtové seznamy.
- src/ECMG/Exceptions – Složka obsahující chybové třídy.
- src/ECMG/Int – Složka obsahující rozhraní, které jsou implementovány všemi controllery a ostatnímy třídami, u kterých je umožněno depencency injection.
- src/ECMG/Exceptions – Složka obsahující třídy, které slouží jako model.
- src/ECMG/Exceptions – Složka obsahující třídy a funkce, které nejsou specifické pro tuto knihovnu. Příkladem může být implementace k-means++ algoritmu.
- src/ECMG/ECGM.php – Třída, která slouží jako vstupní rozhraní do knihovny.
- src/ECMG/MainInterface.php – Rozhraní, jehož implementaci je nutné vytvořit v závislosti na projektu.
- tests – Složka obsahující potřebné testy.
- vendor – Složka vytvořená programem composer, která obsahuje všechny závislosti této knihovny.

3.2 Vývojový diagram

Vzhledem k rozsáhlosti knihovny bude nejdříve zobrazen celkový pohled na strukturu knihovny a poté dojde k detailnějšímu představení tříd a rozhraní v jednotlivých složkách.

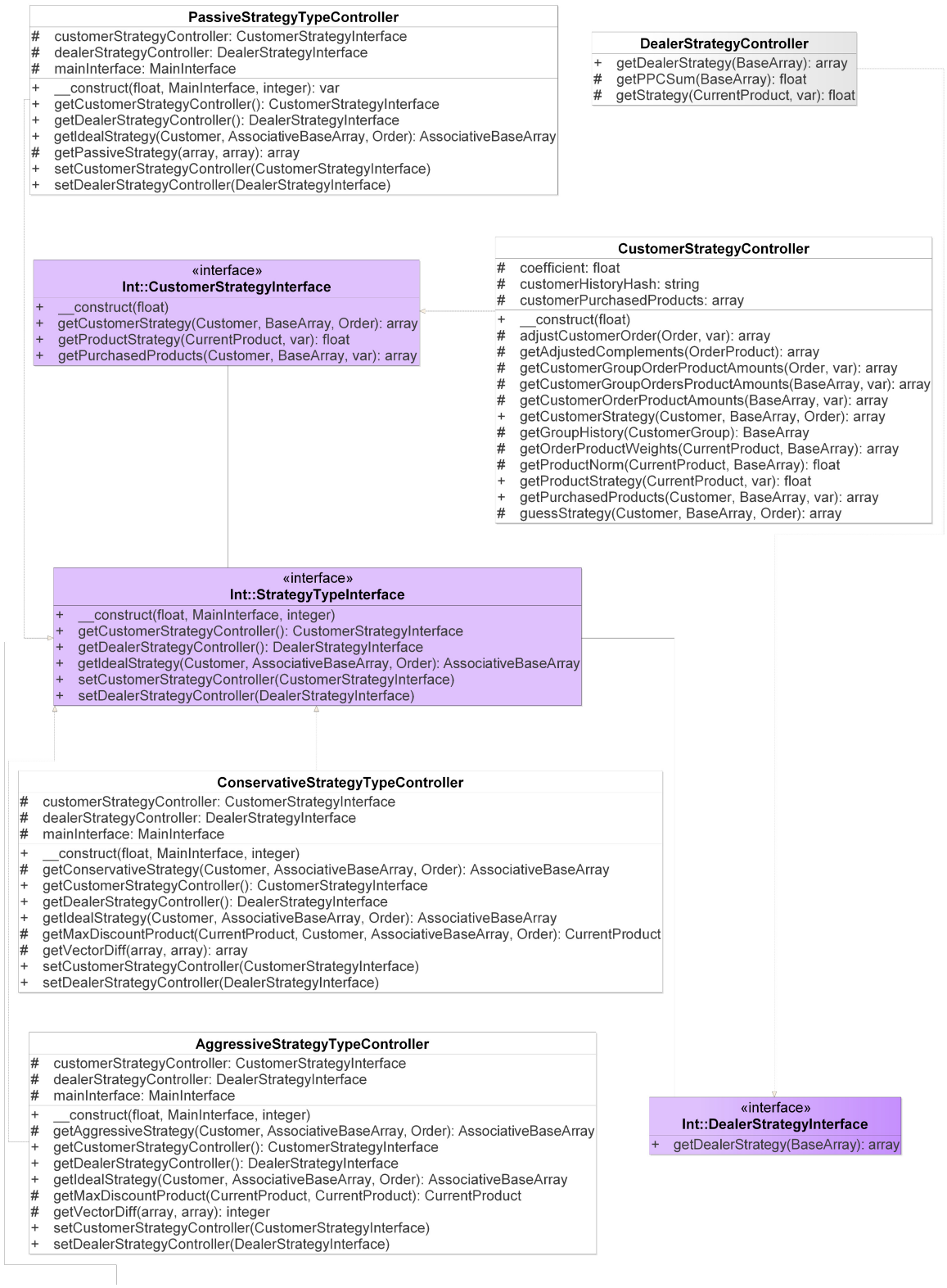


Obrázek 13: UML – celkový pohled



Obrázek 14: UML – testy

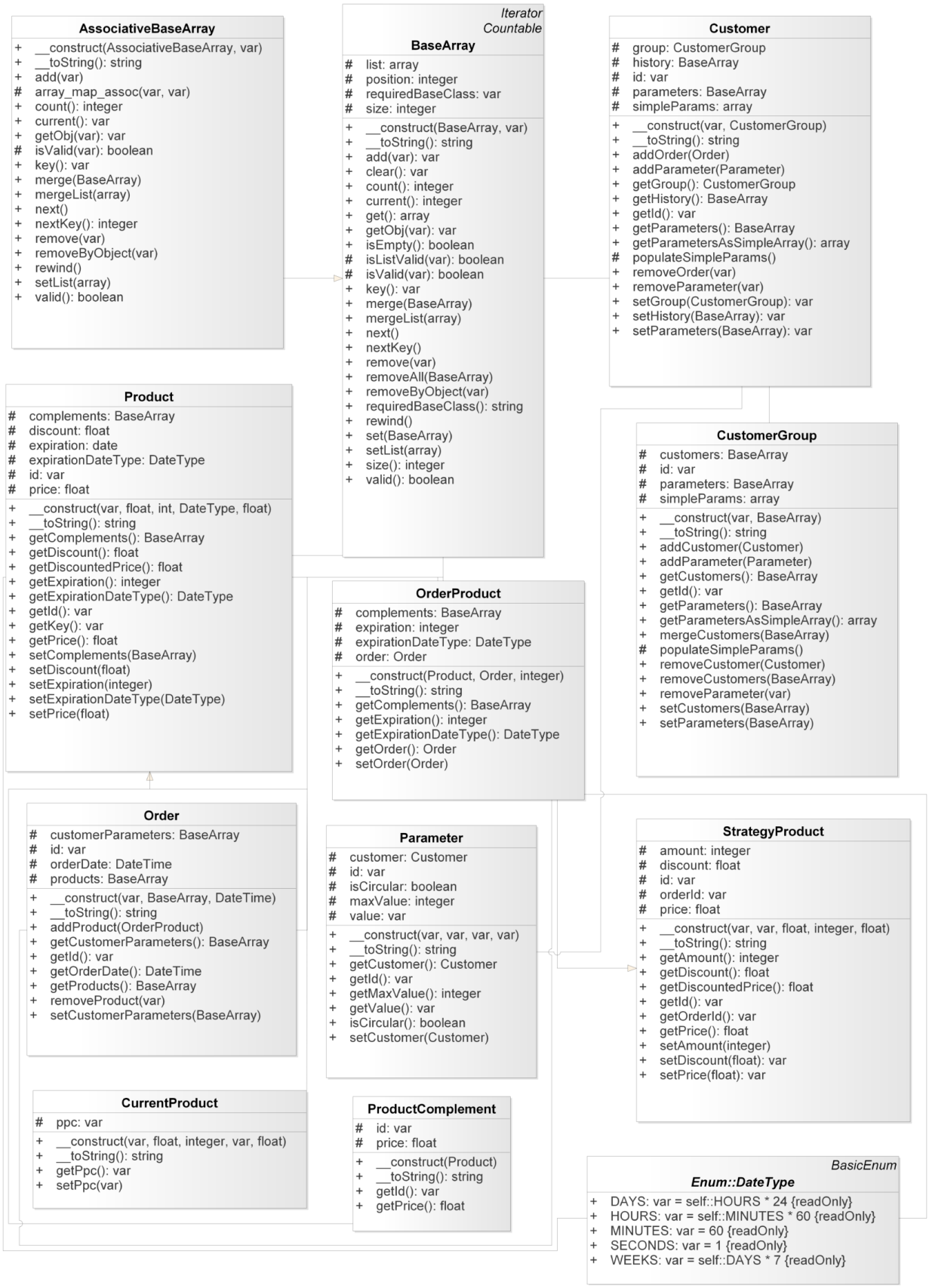
Testy byly pomocí PHPUnit vytvořeny pro většinu funkcí knihovny. Jejich hlavním úkolem je asistence při přetěžování a změně logiky knihovny. Je také vhodné zmínit, že algoritmus k-means++ není deterministický, tudíž se pro stejná vstupní data může v rozdílných instancích výsledek lišit, a proto je možné vytvořit pouze omezenou sadu testů. V nich dochází v zásadě pouze k ověření správné funkčnosti algoritmů s náhodně zvoleným souborem dat.



Obrázek 15: UML – controllery A

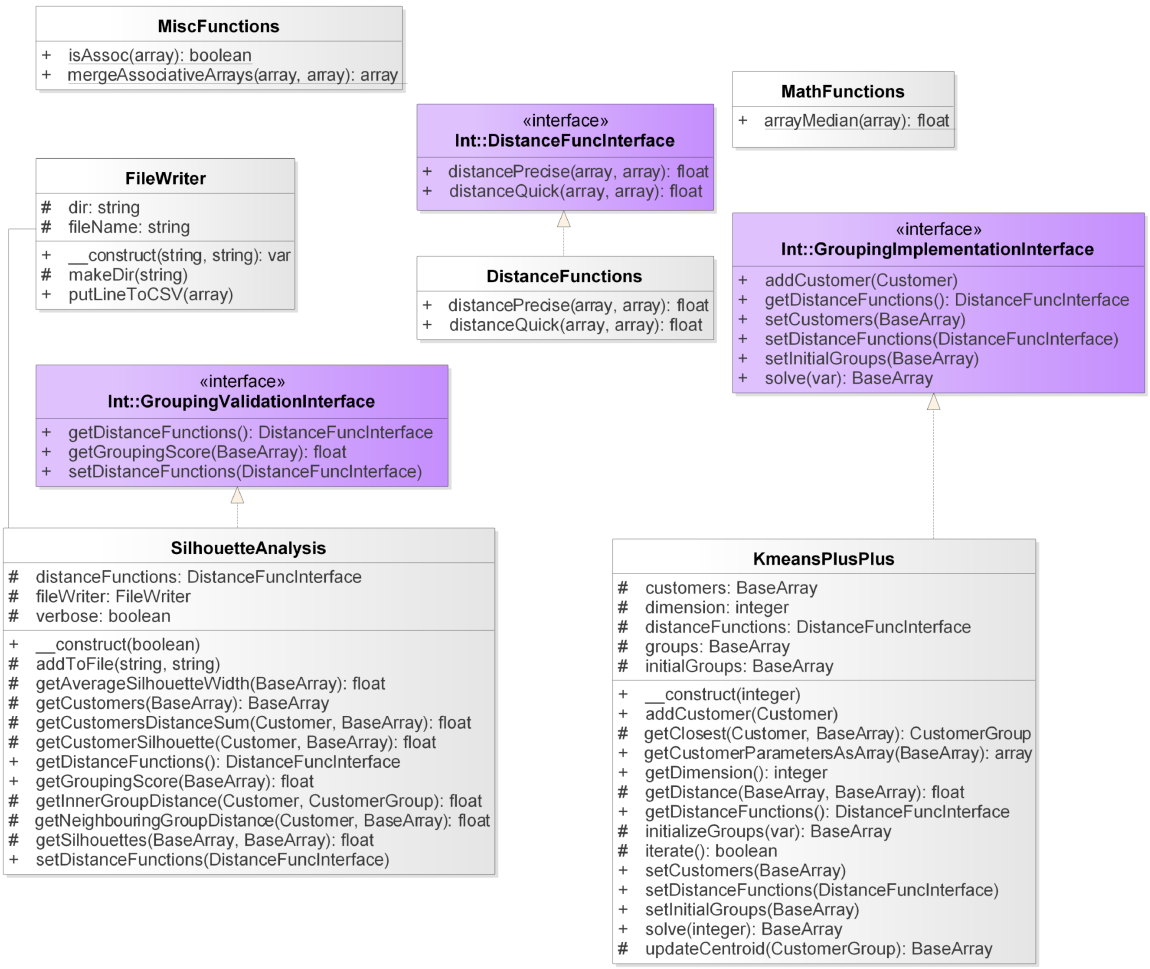
Obrázek 16: UML – controllery B

Jak je zřejmé z diagramu, každý controller implementuje příslušné rozhraní. To je nutné kvůli modulárnosti knihovny a využití dependency injection. Dependency injection umožňuje v PHP jednoduše nahradit stávající třídu vlastní implementací. Přestože dependency injection znamená složitější vývoj, je v případě jakékoliv aplikace, u které je předpokladem využití v různorodých situacích, téměř nutností. Psaní aplikace pro všechny možné situace je extrémně nepraktické, protože budou vždy existovat tak exotické požadavky, že by jejich zahrnutí nebylo v rozumném časovém období možné. Dependency injection tento problém elegantně řeší, protože pokud uživatel není se stávající logikou programu spokojen, tak pouze implementuje příslušné rozhraní, popřípadě přetíží danou třídu a funkcionalitu upraví.



Obrázek 17: UML – modely

U modelů není dependency injection umožněna. Hlavním důvodem je velmi těsná integrace do knihovny. Modely jsou také navrženy co možná nejobecněji, takže by jejich přetěžování nemělo být nutné. Samozřejmě pokud by byla nějaká změna nutná, licence programu to umožňuje.



Obrázek 18: UML – obecné funkce

U obecných funkcí se naopak předpokládá poměrně značná iniciativa jejich přetížení, popřípadě nahrazení. Hlavními kandidáty jsou matematické funkce, jako implementace siluetové analýzy a k-means++ algoritmu. Existuje totiž několik dalších algoritmů, které jejich funkce ve speciálních případech splní podstatně lépe. Jako příklad by se dalo uvést shlukování zákazníků s vysokým počtem parametrů. U algoritmu k-means dochází ke značnému navýšení doby potřebné pro výpočet v případě vysoce dimenzionálních dat. Proto existují specializované algoritmy, určené pro tyto případy.

Jako dalšího kandidáta by bylo vhodné uvést vzdálenostní funkce. Zde jsou v rámci knihovny využívány dva hlavní typy vzdáleností – rychlá a precizní. V této implementaci je pro rychlou distanční funkci využita Manhattanská vzdálenost a pro precizní vzdálenost Euklidovská. Rychlá Manhattanská vzdálenost je využívána siluetovou analýzou a precizní Euklidovská algoritmem k-means++.

V původní implementaci byla i siluetovou analýzou využívána vzdálenost euklidovská, ale při testech s náhodným souborem 20 000 zákazníků docházelo k extrémnímu nárůstu časové náročnosti výpočtu. Ta byla dána jak principem funkce siluetové analýzy jako takové, kde je nutná poměrně složitá rekurze, tak špatnou podporou vícevláknových výpočtů v PHP. Záměnou Euklidovské vzdálenosti za Manhattanskou, došlo k téměř 34% zkrácení doby výpočtu, kterou siluetová analýza vyžadovala.

Z tohoto důvodu je pro výpočty, při kterých se předpokládá značná přítomnost rekurze, využita rychlá distanční funkce, která sice neposkytuje takovou přesnost jako precizní, ale je méně náročná. V případě implementace, u které se neočekává velký počet zákazníků, se kterými by shlukování probíhalo, je pravděpodobně výhodnější nahradit Manhattanskou vzdálenost rychlé distanční funkce zpět za Euklidovskou a tím výpočet siluet zpřesnit.

```
$distance = 0;
for ($n = 0; $n < $dimension; $n++) {
    $distance += abs( number: $v1[$n] - $v2[$n]);
}

return $distance;
```

Obrázek 19: Algoritmus Manhattanské vzdálenosti

Hlavním a jediným rozdílem ve výpočtu obou vzdáleností, je využití kvadrátu vzdáleností (pow) u euklidovské vzdálenosti oproti absolutní hodnotě (abs) u Manhattanské. Absolutní hodnota je výpočetně podstatně méně náročná, než kvadrát. Zároveň ale, jak již bylo zmíněno, nedosahuje ve výsledku takové přesnosti.

```php
$distance = 0;
for ($n = 0; $n < $dimension; $n++) {
    $distance += pow( base: $v1[$n] - $v2[$n], exp: 2);
}

return sqrt($distance);
```

Obrázek 20: Algoritmus Euklidovské vzdálenosti

4 ANALÝZA INTERNETOVÉHO OBCHODU

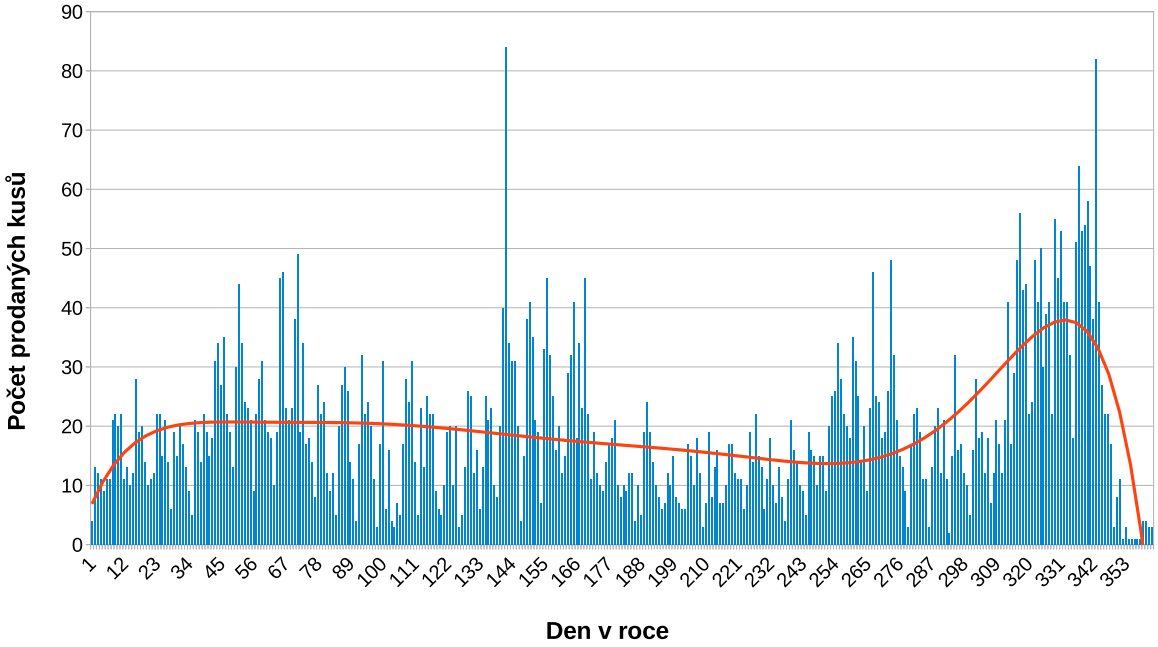
V této kapitole dojde k analýze a následnému nasazení knihovny na poměrně velký internetový obchod. Na základě dohody nebudou v této práci zmíněny žádné konkrétní údaje, přesto se bude jednat o data z reálného světa, a ne náhodně generovaná. Nadále v této kapitole bude tímto obchodem rozuměn prodejce d. Další označení budou korespondovat s klasifikací již vymezenou v kapitole 2.1 a budou využity postupy popsané v předchozích kapitolách této práce.

V době psaní bylo v tomto obchodě vytvořeno téměř čtvrt milionu objednávek, objednáno přes jeden a půl milionu produktů. Zároveň bylo vytvořeno přes čtyřicet tisíc unikátních registrací. Díky takto obrovskému souboru dat se jedná o velmi vhodného kandidáta pro ověření ustanoveného teoretického základu.

U tohoto obchodu bude také implementována knihovna vytvořená v praktické části práce. Postup implementace bude v následujících podkapitolách podrobně popsán.

4.1 Vizualizace dat

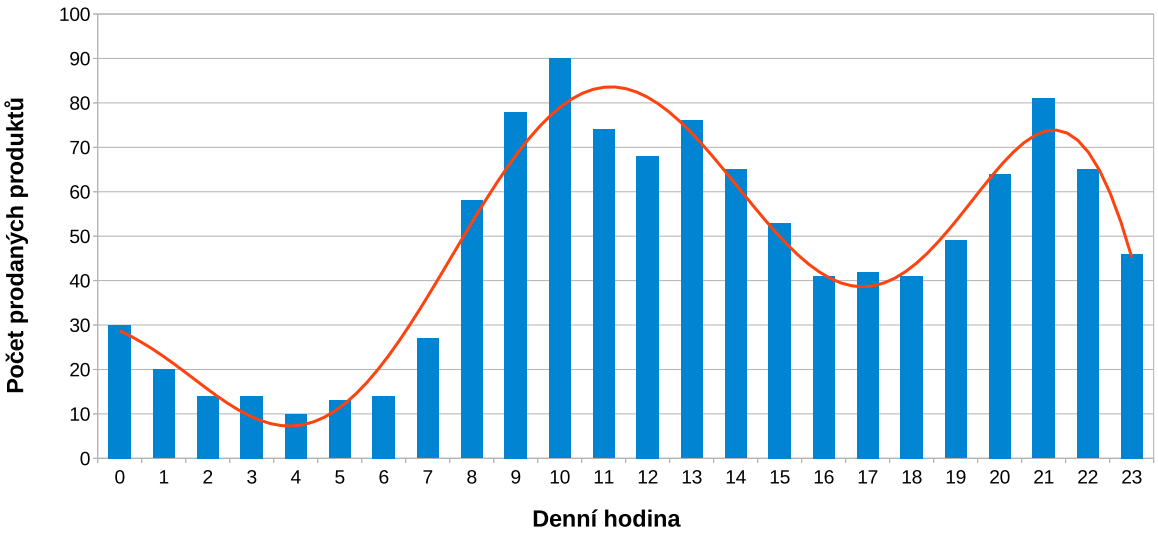
Vzhledem ke značnému množství dat bude nejdříve vhodné alespoň částečně vizualizovat chování zákazníků. Jako první bude zobrazen graf denních prodejů produktu p_{1085} za rok 2017. Tento produkt je zdaleka nejoblíbenějším výrobkem, který d prodává.



Obrázek 21: Prodeje produktu p_{1085} za rok 2017

Z grafu je jasně patrný značný nárůst prodejnosti před Vánoci. Zároveň je také vidět pokles prodejů téměř na nulu v době vánočních svátků a po novém roce postupný návrat na standardní hladinu. Tyto odchylky jsou poměrně očekávané a nejsou specifické pouze pro tento daný produkt.

Velmi viditelný extrém byl zaznamenán ve 142. den a byl následován zvýšenou prodejností. Ten byl dán seminářem zaměřujícím se na správné použití a propagaci tohoto produktu, který se uskutečnil právě před tímto datem. Další zajímavostí, specifickou pro tento produkt, je pokles jeho prodejnosti v době letních prázdnin.



Obrázek 22: Průměrný hodinový počet prodaných produtků za rok 2017

Tento graf znázorňuje průměrný počet koupených produktů v rámci všedního dne. Víkendy zahrnuty nejsou, aby nedocházelo ke zkreslování dat. Očekávaným jevem je velmi nízká prodejnost v nočních hodinách, přesto zde prodeje jsou a rozdíl oproti denním hodinám je opravdu znatelný. Velmi zajímavou položkou jsou prodeje v dopoledních hodinách. Ty se svojí četností téměř rovnají prodejům v hodinách večerních. Z toho je možné usuzovat, že v obchodě nakupují lidé, kteří nechodí do práce.

Dá se tedy odhadnout, že značnou zákaznickou skupinou budou ženy na mateřské dovolené a senioři. Dále je samozřejmě možné, že tyto nákupy uskutečňují zákazníci v práci, ale je nepravděpodobné, že by to dělali v takovém množství.

Dalším očekávaným jevem je snížení počtu objednaných produktů v době oběda. Za zmínku ale stojí lokální minimum mezi 16 a 19 hodinou. To by se dalo přisoudit hlavně přesunu zákazníků z práce domů a času večeře. Od té chvíle dochází k prudkému nárůstu, který je předvídatelný, protože dochází k přílivu objednávek od pracujících zákazníků.

Tyto objednávky pokračují téměř až do jedné hodiny ranní. Večerní nákupy podporují právě teorii o velké části zákazníků, kteří nechodí do práce. Zároveň je nepravděpodobné, že by tyto objednávky vytvářeli studenti.

Zákaznická kategorie	Počet zákazníků	Průměrný denní počet prodaných produktů na zákazníka
1	42296	0,00904586
2	1346	0,06508878
3	13	2,17693984
4	46	0,23713995
5	1	3,5
6	812	0,01364564
7	10	4,51550868

Tabulka 25: Denní průměrný počet prodaných produktů na zákazníka pro kategorii

V této tabulce jsou zobrazeny jednotlivé zákaznické kategorie v internetovém obchodě. Zároveň je k nim přiřazen počet zákazníků, kteří se v této kategorii nachází a průměrný denní počet prodaných produktů na zákazníka.

Kategorie 1 je brána jako výchozí a je do ní zařazený každý nově zaregistrovaný zákazník, proto je zdaleka nejpočetnější. Pro ostatní kategorie je nutné splnit určitá kritéria, a proto se v nich nachází pouze menšina všech zákazníků obchodu.

Přesto mají všechny nevýchozí kategorie několikanásobně vyšší podíl prodaných produktů na zákazníka, tudíž bude výhodné se zákaznickými kategoriemi počítat při výběru parametrů pro shlukovou analýzu.

4.2 Výběr parametrů

Vzhledem k vysokému počtu zákazníků bude vhodné vybrat menší množství parametrů. Je třeba brát v úvahu, že počet cirkulárních parametrů se po očištění zdvojnásobí a dimenze shluků může znatelně narůst. Sortiment v tomto internetovém obchodě není závislý na poloze zákazníka, proto nebude do parametrů zařazena.

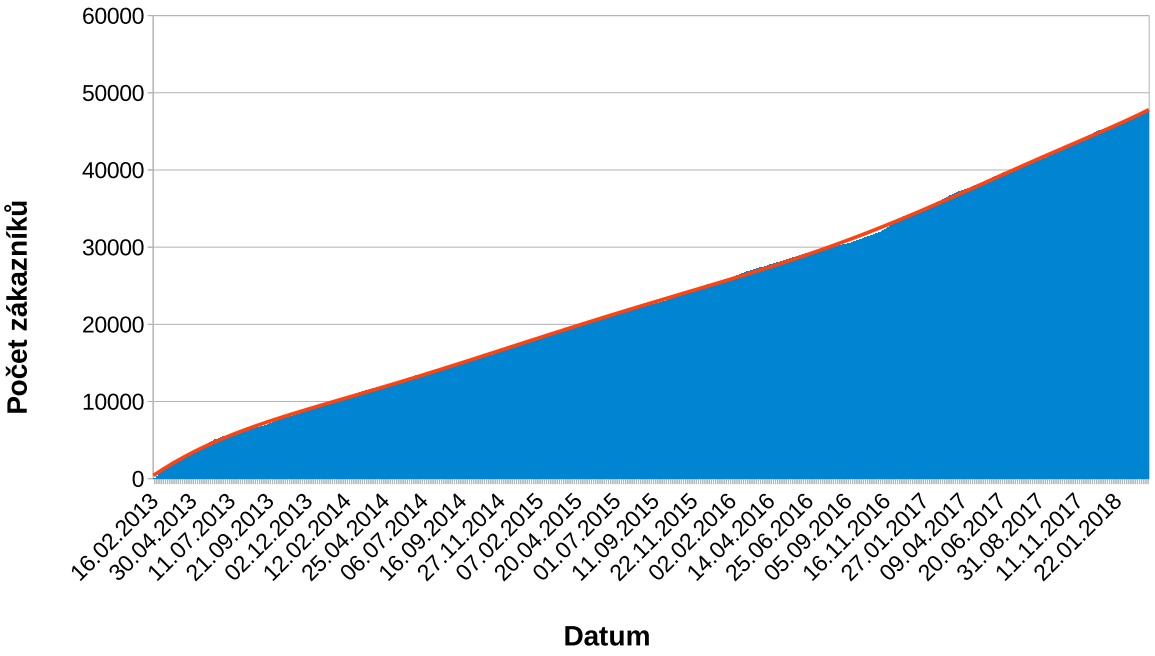
Naopak z analýzy jasně vyplývá, že prodej sortimentu je závislý na času nákupu, proto bude do parametrů zahrnuta hodina ve dni a den v roce, kdy zákazník obchod navštívil. V obchodě také dochází k rozdělení zákazníků na skupiny. Jedná se o maloobchodní, velkoobchodní, zahraniční a zaměstnanecké zákaznické kategorie. Každá z těchto skupin má určité výhody, které se projevují hlavně na cenách produktů, proto bude výhodné do shlukování tyto kategorie zařadit. Pro potřeby této implementace bude naprosto dostačující zákazníky rozdělit na ty, kteří mají výchozí kategorii a na ty, kteří ji nemají.

Zákazník	Den v roce x	Den v roce y	Hodina ve dni x	Hodina ve dni y	Výchozí zákaznická kategorie
c_{186872}	-0,421	-0,424	-0,402	0,858	1
c_{186873}	-0,762	0,647	-0,604	0,787	1
c_{186874}	-0,757	0,654	-0,5	0,866	1
c_{186875}	-0,757	0,654	-0,5	0,866	1
c_{186876}	-0,757	0,654	-0,5	0,866	0
c_{186877}	-0,017	0,673	-0,604	0,787	1
c_{186878}	-0,757	0,654	-0,259	0,966	1
c_{186880}	-0,61	0,77	0,129	0,983	1
c_{186881}	-0,745	0,667	0,983	0,129	1
c_{186882}	-0,745	0,667	0,966	-0,259	1
c_{186884}	-0,18	-0,128	0,104	-0,787	1
c_{186885}	0,034	0,289	0,038	-0,574	1
c_{186887}	-0,017	0,21	1,038	-0,39	1
c_{186889}	-0,745	0,667	0	-1	1

Tabulka 26: Ukázková očištěných parametrů zákazníků

V této tabulce je ukázka několika zákazníků internetového obchodu s již očištěnými parametry připravenými pro shlukování. Za pozornost stojí mezery v posloupnosti jejich identifikačních čísel. To je dáno tím, že pro shlukovou analýzu byli vybráni pouze ti zákazníci, kteří se zaregistrovali. V tomto případě není nutné nezaregistrované zákazníky do shlukování započítávat, protože i tak je jejich počet více než dostatečný. Nezaregistrovaní zákazníci mají také historii skládající se pouze z jednoho nákupu, tudíž nejsou tolik přínosní.

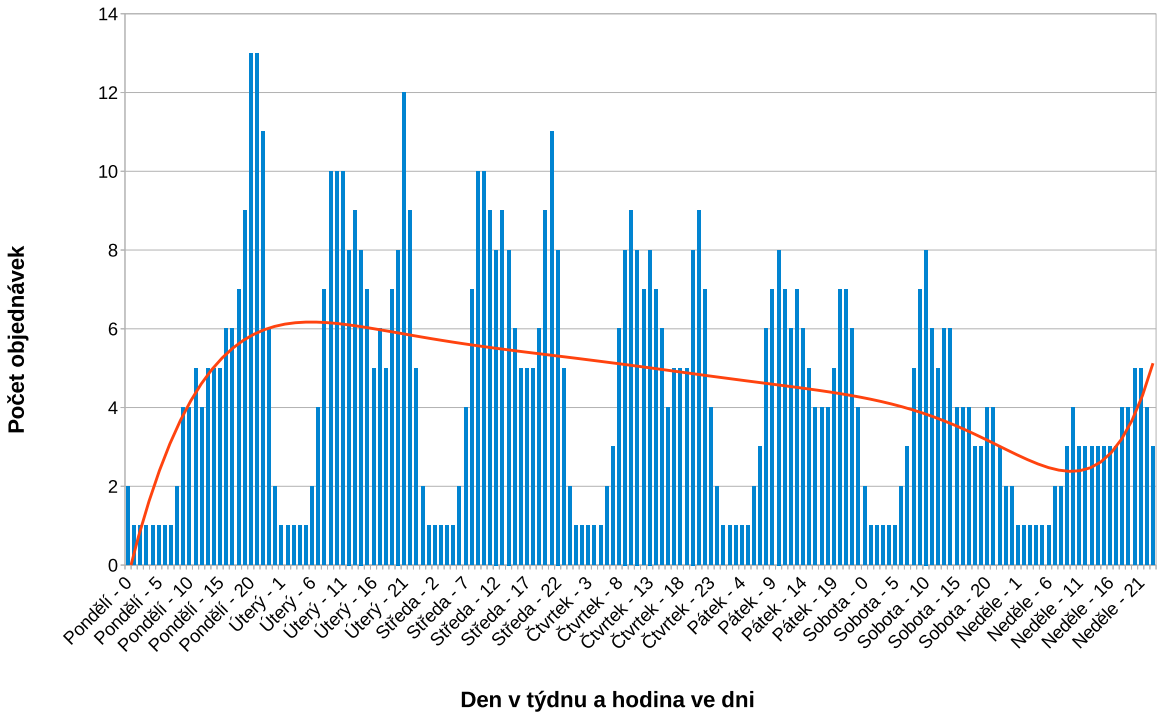
4.3 Shlukování zákazníků



Obrázek 23: Denní přírůstek zákazníků

Jak již bylo zmíněno v předchozích kapitolách, bude při implementaci nutné určit, jakým způsobem bude docházet k převytváření zákaznických shluků. K lepší představě velmi dobře poslouží graf přírůstku zákazníků od založení obchodu. Z něho je jasné, že přírůstek je téměř pouze lineárního charakteru. Na grafu nejsou patrné žádné extrémní výkyvy.

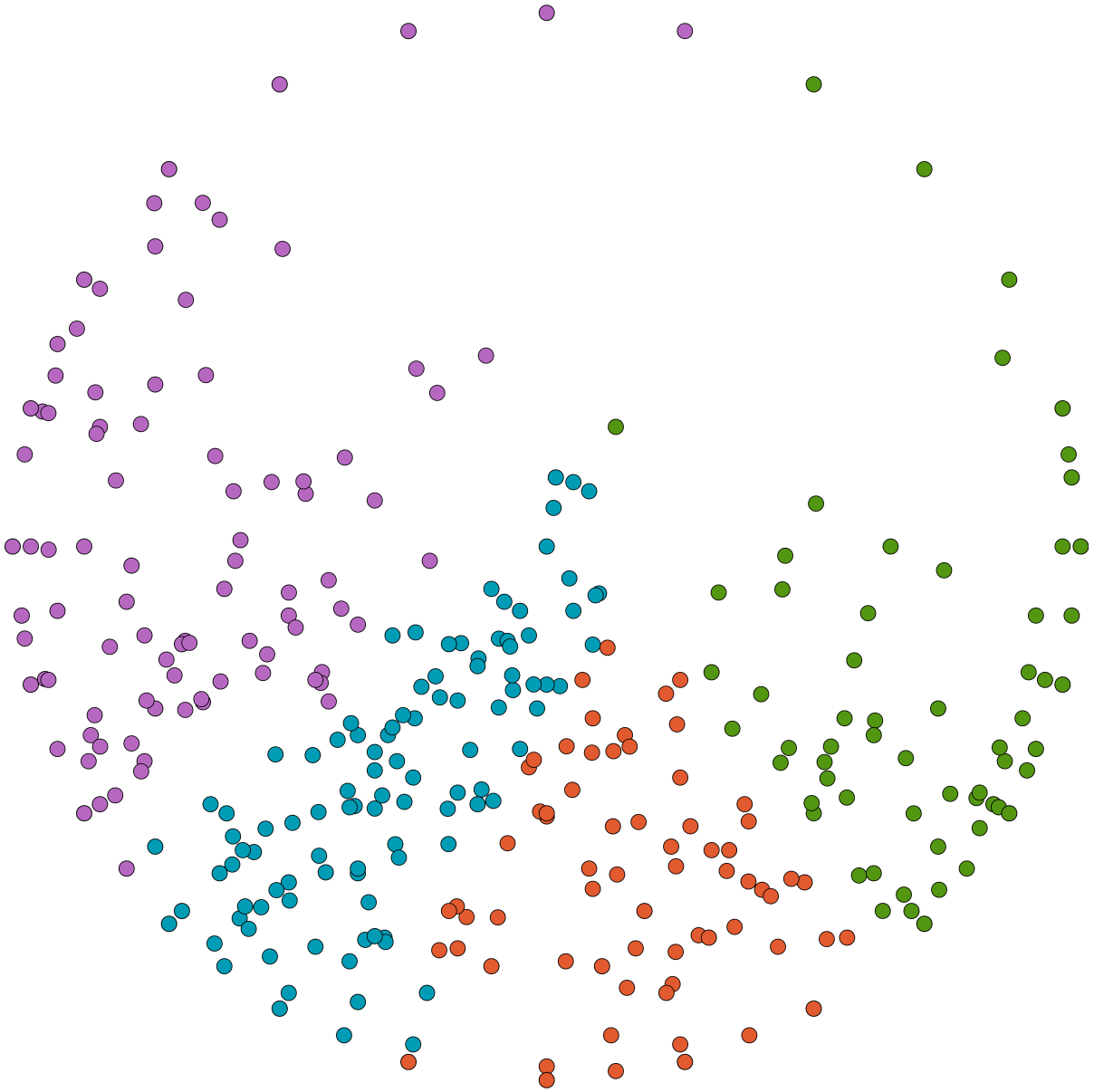
Lze předpokládat, že se na tomto trendu nic nezmění, proto nebude v tomto případě nutné vytvářet jakékoliv funkce pro předčasné převytvoření shluků. Naprosto plně postačí shluky vytvářet jednou týdně, a proto bude vhodné najít čas, kdy je v průměru nejmenší aktivita, tedy počet objednávek.



Obrázek 24: Průměrný počet nákupů během týdne

Podle hodnot v tomto grafu je nejvhodnějším časem pro převytváření shluků v neděli mezi jednou a šestou hodinou ranní. Zároveň se neděle jeví jako den nejmenší aktivity, tudíž bude existovat prostor pro řešení případných problémů před začátkem dalšího týdne, který znamená dramatický nárůst nových objednávek.

Vzhledem k enormnímu počtu zákazníků bylo rozhodnuto nevyužívat siluetovou analýzu. Siluetová analýza by znamenala několikanásobné zvýšení času potřebného pro vytvoření shluků, které již samo o sobě při počátečním rozřazení zabralo více než hodinu. Toho rozhodnutí také značně usnadní budoucí přidávání nových zákazníků, protože vzhledem k neměnnému k bude možné využít již vytvořené shluky a nové zákazníky do nich zařadit. Jelikož do obchodu přichází velké množství neregistrovaných zákazníků, bylo po uvážení rozhodnuto zvolit $k = 100$, díky čemuž bude v každém shluku značný počet zákazníků. To by mělo zaručit dostatečně velikou historii pro každý shluk.



Obrázek 25: Shluk tisíce zákazníků podle denní hodiny nákupu

Na tomto obrázku je vizualizace shluků. Zobrazení všech zákazníků obchodu je z důvodu jejich velikého počtu nepraktické, proto je zde zobrazen pouze výřez tisíce zákazníků. Každý bod reprezentuje jednoho zákazníka. Barva bodu pak náleží shluku, do kterého je zákazník zařazen.

Vzhledem k tomu, že samotná shluková analýza pracuje s pěti dimenzemi, byla pro vizualizaci vybrána pouze dvourozměrná hodina dne, ve které zákazník nakoupil. Pro účely této ukázky bylo vybráno $k = 4$.

Za pozornost stojí kruhové uspořádání bodů. To je dáno převody při čištění parametrů na základě transformace popsané v teoretické části práce. Na tomto obrázku je také velmi

dobře viditelné rozložení počtu nákupů, kde jsou v horní části zobrazeny večerní hodiny, a v dolní části dopolední a odpolední.

4.4 Strategie

Při implementaci bylo rozhodnuto, že bude prozatím experimentálně nasazena pouze strategie pasivního typu. Vzhledem k tomu, že d naprostou většinu produktů, které prodává, také sám vyrábí, bude PPC rovno pouze čistému výdělku na produkt. Cena výroby každého produktu je téměř ve všech případech dána stejnou procentuální hodnotou z celkové ceny produktu.

Tato procentuální hodnota je ale striktně důvěrnou informací, proto bude PPC rovno koncové ceně produktu. Zároveň nebude nutné pracovat se skladovými zásobami, protože pokud by byl nějaký produkt vyprodán, lze ho poměrně rychle vyrobit.

Produkty budou brány jako spotřební zboží. Pro zjednodušení konečných výsledků bude v této práci každému produktu přiřazena jednotná expirace 30 dní od data prodeje. V praxi bude samozřejmě nastavena každému produktu zvlášť.

Produkt	Cena	V prodeji	PPC
p_{6480}	1163,78	Ano	1164
p_{6481}	803,76	Ano	804
p_{6488}	425,07	Ano	425
p_{6489}	850,15	Ano	850
p_{6490}	1445,24	Ano	1445
p_{6491}	26,98	Ano	27
p_{6504}	10,09	Ne	10
p_{6505}	836,63	Ano	837
p_{6508}	1321,21	Ano	1321
p_{6509}	2246,06	Ano	2246

Tabulka 27: Ukázka produktů

Z této tabulky je zřejmé, že d disponuje cenově velmi rozmanitým portfoliem.

Pořadí	Produkt	PPC
1	p_{3571}	44
2	p_{6459}	2010
3	p_{3576}	1005
4	p_{3566}	639
5	p_{3564}	81
6	p_{3561}	589
7	p_{3560}	381
8	p_{3558}	50
9	p_{3556}	328
10	p_{3554}	214

Tabulka 28: Pasivní strategie zákazníka c_{194873}

Z předchozí tabulky je patrné, že zákazník a jeho skupina nakupují produkty, které byly přidány těsně po sobě. To je možné odvodit z faktu, že pro identifikátor p je využit identifikátor produktu v databázi, tudíž jsou produkty označeny vzestupně podle pořadí, v jakém byly přidány. Jedinou výjimkou z tohoto pravidla je produkt p_{6459}, který je podstatně novější.

Pořadí	Produkt	Navrhovaná sleva	PPC
1	p_{5507}	0%	402
2	p_{3102}	0%	219
3	p_{5509}	53%	303
4	p_{3104}	0%	351
5	p_{4429}	0%	156
6	p_{5139}	0%	84
7	p_{3063}	0%	62
8	p_{4604}	0%	262
9	p_{4498}	0%	259
10	p_{5045}	0%	231

Tabulka 29: Konzervativní strategie zákazníka c_{194873}

Z tabulky obsahující navrhovanou konzervativní strategii je patrné, že ve většině případů není možné aplikovat takovou slevu, aby uspokojila prodejce a zároveň změnila strategii zákazníka. Pouze u produktu p_{5509} byla sleva navržena.

Tato strategie se také bude podstatně lišit od strategie pasivní, protože v ní nedochází ke kombinaci strategie zákazníka a strategie prodejce. Jak již bylo popsáno, strategie prodejce je zohledněna již při hledání aplikovatelných slev.

Vzhledem k tomu, že při implementaci bude nejpravděpodobněji použita strategie pasivní, nebude vůbec docházet k testovaní agresivní strategie. Tak jak již bylo zmíněno, bude mít uplatnění jen velmi zřídka, a tudíž v naprosté většině případů postačí pravě pasivní, nebo konzervativní typ strategie.

ZÁVĚR

Náplní této práce bylo vytvořit aplikaci teorie her a jejích principů v oblasti e-commerce a na tomto základu poté vytvořit PHP knihovnu, která by po implementaci na internetový obchod sloužila k návrhu nejvhodnější strategie pro konkrétního zákazníka s ohledem na potřeby obchodníka.

Na rozdíl od bakalářské práce, kde jsem se při zpracování rešerše potýkal s nedostatkem příhodných zdrojů, bylo k dispozici rozsáhlé množství publikací zabývajících se tématem teorie her. Bohužel aplikací teorie her na oblast e-commerce je v současnosti velmi málo a pokud již nějaké implementace existují, tak jsou dobře střeženy jejich vlastníkem.

Z tohoto důvodu bylo nutno všechny potřebné postupy nejdříve navrhnout. Naštěstí je naprostá většina již velmi dobře zpracována v matematické statistice, proto bylo nejobtížnější vybrat vhodné postupy. Samotné vzorce pak již bylo možné najít v nesčetně publikacích. Samozřejmě v některých případech bylo nutné specializované matematické vzorce přímo sestavit, ale i u nich bylo možné se opřít o předešlé práce.

Tvorba samotné knihovny byla i přes svoji rozsáhlost podstatně přímočařejší. Zde byla nejdůležitější korektní implementace algoritmů zpracovaných v teoretické části práce. Jak již bylo zmíněno, k řešení jednoho problému existovalo mnoho různých postupů, kde některé byly v určitých situacích výrazně vhodnější než jiné. Přestože v aplikaci byly využity postupy co nejobecnější, je pravděpodobné, že v některých případech nebudou uživateli vyhovovat, proto bylo nutné zajistit co nejvyšší modulárnost knihovny. Díky tomu bylo nutné vytvořit sadu testů, které budou nápomocné při odhalování chyb, pokud by se uživatel rozhodl aplikaci přizpůsobit svým potřebám.

POUŽITÁ LITERATURA

[1] Theodore L. Turocy, Bernhard von Stengel *Game Theory* [Článek]. London School of Economics & Political Science: 2001 [cit. 2017-11-17].

[2] Miroslav Maňas *Teorie her a optimální rozhodování* [Kniha]. SNTL: 2001 [cit. 2017-11-17]. ISBN 80-7226-385-4.

[3] Thomas S. Ferguson *Game Theory, Second Edition* [Kniha]. University of California at Los Angeles: 2014 [cit. 2017-11-19].

[4] Osborne, Martin J., and Ariel Rubinstein *A Course in Game Theory* [Kniha]. Cambridge, MA: MIT: 1994 [cit. 2017-11-19]. ISBN 0-262-65040-1.

[5] Feryal Erhun, Pınar Keskinocak *Game Theory in Business Applications* [Článek]. Stanford University, Stanford, Georgia Institute of Technology, Atlanta: 2003 [cit. 2017-12-04].

[6] Encyclopædia Britannica *Dr. John Von Neumann* [Fotografie]. 1946 [cit. 2017-11-17].

[7] William Poundstone, John von Neumann (Based On Work by) *Prisoner's Dilemma* [Kniha]. Anchor: 1993 [cit. 2017-11-18]. ISBN 978-0307763785.

[8] Levent Koçkesen, Efe A. Ok *An Introduction to Game Theory* [Článek]. Koç University,New York University: 2007 [cit. 2017-11-25].

[9] Fiona Carmichael *A Guide to Game Theory* [Kniha]. Pearson Education: 2005 [cit. 2017-11-25]. ISBN 0-273-68496-5.

[10] Columbia University *Cluster Analysis Using K-Means* [online]. Columbia [cit. 2017-11-26]. Dostupné z: `https://www.mailman.columbia.edu/sites/default/files/png/Screen-Shot-2015-06-17-at-4.30.52-PM.png`

[11] Vipin Kumar *An Introduction to Cluster Analysis for Data Mining* [Kniha]. Addison Wesley; Auflage: 2005 [cit. 2017-11-26]. ISBN 978-0321321367

[12] Nicholas J. Cox *Speaking Stata: In praise of trigonometric predictors* [Článek]. StataCorp LP: 2006 [cit. 2017-11-29]. ISBN 978-0321321367

[13] Songwon Seo *A Review and Comparison of Methods for Detecting Outliersin Univariate Data Sets* [Článek]. BS, Kyunghee University: 2002 [cit. 2017-12-01].

[14] IBM Knowledge Center *Modified z score* [online]. IBM [cit. 2017-12-16]. Dostupné z: `https://www.ibm.com/support/knowledgecenter/en/SSWLVY_1.0.0/com.ibm.spss.analyticcatalyst.help/analytic_catalyst/modified_z.html`

[15] Ken A. Aho *Foundational and Applied Statistics for Biologists Using R* [Kniha]. CRC Press: 2007 [cit. 2017-01-13]. ISBN 978-1439873397

[16] David Arthur, Sergei Vassilvitskii *k-means++: The Advantages of Careful Seeding* [Článek]. Stanford University: 2016 [cit. 2017-01-16].

[17] P. J. Rousseeuw *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis* [Článek]. University of Fribourg: 1986 [cit. 2017-01-16].

[18] Mingjin Yan *Methods of Determining the Number of Clusters in a Data Setand a New Clustering Criterion* [Článek]. Virginia Tech: 2005 [cit. 2017-01-17].

[19] Marek Dvořák *Metody shlukové analýzy a jejich aplikace v marketingu* [Diplomová práce]. Univerzita Karlova v Praze: 2008 [cit. 2017-01-17].

[20] Herv e Abdi *Normalizing Data* [Článek]. The University of Texas at Dallas: 2010 [cit. 2017-01-17].

[21] P.-Y. Chen, C.-H. Chen, a H. Wang *A Neural Network: Family Competition Genetic Algorithm and Its Application in Electromagnetic Optimization* [Obrázek]. National Chiao Tung University: 2009 [cit. 2017-01-18].

[22] Robert Hecht-Nielsen *Theory of the Backpropagation Neural Network* [Kniha]. Harcourt Brace & Co. Orlando: 1992 [cit. 2017-01-18]. ISBN 0-12-741252-2

[23] Paul M. Jones *PSR-4: Autoloader* [online]. [cit. 2018-02-06]. Dostupné z: `https://www.php-fig.org/psr/psr-4/`

[24] Chris Pitt *Pro PHP MVC* [Kniha]. Apress: 2012 [cit. 2018-02-08]. ISBN 978-1-4302-4165-2

[25] GitHub *Git repository hosting service* [online]. [cit. 2018-02-12]. Dostupné z: `https://github.com/`

[26] Open Source Initiative *The MIT License* [online]. California [cit. 2018-02-12]. Dostupné z: `https://opensource.org/licenses/MIT/`

SEZNAM PŘÍLOH

PŘÍLOHA A

Obsah přiloženého CD

Na přiloženém CD jsou adresáře strukturovány následovně:

/Dokumentace/ – Obsahuje elektronickou verzi dokumentace knihovny.

/ECGM/ – Obsahuje zdrojové kódy knihovny.

/UML/ – Obsahuje UML diagramy knihovny.

/Prace/ – Obsahuje elektronickou verzi této práce ve formátu PDF.

PŘÍLOHA B

Dokumentace ke knihovně ECGM

ECGM

Dokumentace

OBSAH

ÚVOD

ECGM je nástroj, který má za úkol zpracovat strukturovaná statistická data a na základě nich doporučí produkty/doplňky k produktům, které by bylo nejvhodnější danému zákazníkovi v danou dobu zobrazit. Dále je možné specifikovat i automatické navrhování slev a to buď agresivním, nebo konzervativním způsobem, tak aby byl s co největší pravděpodobností prodán produkt, který je výhodný pro prodejce.

Knihovna a její kompletní zdrojové kódy jsou pod MIT licencí dostupné na github `https://github.com/okomarek/ECGM` , popřípadě na `https://packagist.org/packages/okomarek/ecgm`.

1 ZÁKLADNÍ POJMY

- PPC – PPC neboli Product Payoff Coefficient je numerická hodnota, která vyjadřuje jakou má produkt v daném stavu (sleva, výše skladových zásob, doba před expirací) pro prodejce hodnotu. Čím vyšší PPC je tím vyšší je hodnota tohoto produktu pro prodejce.
- Shluk – Shluk je skupina zákazníků, která byla vytvořena an základě shlukové analýzy. Shluky jsou určeny pouze pro interní použití v knihovně.
- Parametry zákazníka – Pro správnou funkci shlukování je nutné každému zákazníkovi přidat parametry. Ty reprezentují hodnoty, podle kterých dochází ke shlukování. Je nutné, aby měl každý zákazník stejný počet parametrů.
- *@param type $parameter – Požadovaný parametr.
- @param type $parameter = value – Volitelný parametr s výchozí hodnotou.
- @return type – Návratová hodnota funkce, pokud není specifikována, je vraceno void.
- @throws exception – Specifikace výjimky, kterou může funkce vyvolat.
- Pasivní strategie (StrategyType::PASSIVE) – U pasivní strategie dojde pouze k navžení pořadí produktů v jakém bude pro prodejce nejvýhodnější je zákazníkovi prodat.
- Konzervativní strategie (StrategyType::CONSERVATIVE) – U konzervativní strategie bude při řazení zároveň docházet k navržení minimální slevy, tak aby došlo k prioritizování daného produktu zákazníkem.
- Agresivní strategie (StrategyType::AGGRESIVE) – U agresivní strategie bude při řazení zároveň docházek k navržení maximální slevy, tak aby došlo k prioritizování daného produktu zákazníkem, ale zároveň nedošlo k poklesu daného produktu ve strategii prodejce.

2 INSTALACE

2.1 Požadavky

- PHP verze 5.6 a vyšší
- phpunit verze 5 pro vývoj

2.2 Instalace

- Pomocí composer: require okomarek/ecgm

2.3 Po instalaci

Po instalaci je nutné implementovat rozhraní:

\ECGM\MainInterface

- **getCustomers** – Má vrátit všechny zákazníky, kteří budou zahrnuti do strategie. Musí vracet BaseArray s \ECGM\Model\Customer jako requiredClass.
  - @return BaseArray
- **getUngroupedCustomers** – Má vrátit všechny zákazníky, kteří budou zahrnuti do strategie a nemají přiřazený žádný shluk. Musí vracet BaseArray s \ECGM\Model\Customer jako requiredClass.
  - @return BaseArray
- **getCustomerGroups** – Má vrátit všechny již vytvořené shluky zákazníků. Musí vracet BaseArray s \ECGM\Model\CustomerGroup jako requiredClass.
  - @return BaseArray
- **getProducts** – Má vrátit všechny produkty, které jsou momentálně v prodeji a mají být zahrnuty do strategie. Musí vracet AssociativeBaseArray s \ECGM\Model\CurrentProduct jako requiredClass.
  - @return AssociativeBaseArray
- **setProductPPC** – Nastavý požadované PPC (Product Payoff Coefficient) podle stavu vkládaného produktu.
  - *@param CurrentProduct $product
  - @return CurrentProduct

Toto rozhraní je nutné implementovat na základě daného e-shopu. Poté již stačí volat třídu \ECGM\ECGM, která zapouzdřuje všechny funkce knihovny.

2.4 \ECGM\ECGM

Pokud by Vám nevyhovovala jakákoliv funkce, je pro většinu tříd umožněno DependencyInjection. Jediným požadavkem je, aby bylo implementováno příslušné rozhraní. V tomto případě je silně doporučeno dědit již existující třídy a pouze přetěžovat potřebné funkce.

- **___construct**
  - *@param \ECGM\MainInterface $mainInterface
  - *@param int $strategyMultiplierCoefficient– Hodnota této proměnné je využita pro zvýšení váhy nákupů zákazníka oproti nákupům jeho skupiny a ke zvýšení váhy doplňků. Čím vyšší, tím více budou tyto hodnoty .
  - *@param int $dimension – Určuje dimenzi parametrů zákazníka. Je nutné počítat s tím, že po očištění cirkulární hodnoty se její dimenze zdvojnásobí. Například pro hodnoty [Skupina(necirkulární), DevVRoce(cirkulární)] by $dimension = 3.
  - *@param int $initialClusterNumber – Tento parametr určuje minimální množství shluků, se kterými bude shluková analýza pracovat. Pokud bude $autoClusterNumberAdjustment nastaveno na false, nebude toto číslo dále měněno. Správný počáteční odhad počtu shluků bude mít výrazný vliv na časovou náročnost shlukování.
  - @param bool $autoClusterNumberAdjustment = true – Pokud je nastaven na true, bude počet shluků automaticky upraven na základě siluetové analýzy. Siluetová analýza je značně výpočetně náročná, proto je u velkého počtu zákazníků doporučeno tuto funkci vypnout.
  - @param int $maxProductsInStrategy = 40 – Určuje maximální počet produktů, které budou zařazeny do výpočtu strategie. Tyto produkty jsou vybrány na základě zákazníkovi strategie. V základě bude do výpočtu strategie zařazeno maximálně 40 produktů, které si zákazník s největší pravděpodobností koupí. Zvláště u Konzervativní a Agresivní strategie bude vysoký počet produktů způsobovat zpomalení výpočtu.

  - @throws \ECGM\Exceptions\InvalidArgumentException
  - @throws \ECGM\Exceptions\UndefinedException
- **getStrategyController**
  - @return \ECGM\Int\StrategyInterface
- **setStrategyController**
  - *@param \ECGM\Int\StrategyInterface $strategyController
- **getGroupingController**
  - @return \ECGM\Int\CustomerGroupingInterface
- **setGroupingController**
  - *@param \ECGM\Int\CustomerGroupingInterface $groupingController
- **getParameterCleaningController**
  - @return \ECGM\Int\CustomerParametersCleaningInterface
- **setParameterCleaningController**
  - *@param \ECGM\Int\CustomerParametersCleaningInterface $parameterCleaningController
- **groupCustomers**
  - @return \ECGM\Model\BaseArray
- **getStrategy**
  - *@param \ECGM\Model\Customer $customer – Zákazník pro kterého má dojít k výpočtu strategie.
  - @param \ECGM\Model\Order|null $currentOrder = null – Aktuální objednávka zákazníka, pokud má již nějaké produkty v košíku.
  - *@param \ECGM\Enum\StrategyType $strategyType = \ECGM\Enum\StrategyType::CONSERVATIVE
  - @return \ECGM\Model\AssociativeBaseArray – Produkty jsou vraceny sestupně od nejlepšího po nejhorší, společně s navrženými slevami.

3 MODELY

3.1 \ECGM\Model\BaseArray

Implementuje \Iterator, \Countable

Jedná je o třídu obalující jednoduché PHP pole (array).

- **___construct**
  - @param BaseArray|null $baseArray = null – Pokud je hodnota dojde k počátečnímu naplnění pole obsahem tohoto BaseArray.
  - @param string|null $requiredBaseClass = null – Pokud je hodnota naplněna neumožní vložení jiné hodnoty, než té, která je nebo má jako předka danou třídu.
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **requiredBaseClass**
  - @return string – Vrací aktuální hodnotu požadované třídy.
- **add** – Vloží hodnotu nakonec pole.
  - *@param mixed $obj
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **setList** – Přepíše všechny aktuální hodnoty v pole obsahem proměnné $list.
  - *@param $list
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **mergeList** – Obsah proměnné $listy přidá na konec aktuálního pole.
  - *@param $list
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **mergeList** – Přepíše všechny aktuální hodnoty obsahem pole v $baseArray.
  - *@param BaseArray $baseArray
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **mergeList** – Obsah pole v $baseArray přidá na konec aktuálního pole.
  - *@param BaseArray $baseArray
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **clear** – Smaže všechny aktuální hodnoty.
- **size** – Vrací aktuální velikost pole.

  - @return int
- **removeAll** – Smaže všechny hodnoty, které se nachází v průniku aktuálního pole a pole které obsahuje parametr $baseArray.
  - *@param BaseArray $baseArray
  - @throws @throws \ECGM\Exceptions\InvalidArgumentException
- **remove** – Smaže objekt, na pozici parametru $key.
  - *@param int $key
- **removeByObject** – Smaže objekt, který je roven parametru $obj, pokud se v poli nachází.
  - *@param mixed $obj
- **isEmpty**
  - @return bool
- **getObj** – Vrátí objekt, které se nachází na pozici dané hodnotou $key. Vrací null, pokud daná pozice neexistuje, nebo je parametr $key nesprávně zadaný.
  - *@param int $key
  - @return mixed|null

3.2 \ECGM\Model\AssociativeBaseArray

Rozšiřuje \ECGM\Model\BaseArray

Jedná se o rozšíření třídy \ECGM\Model\AssociativeBaseArray, které místo indexového pole využívá pole asociativní. Všechny vkládané hodnoty musí implementovat \ECGM\Int\KeyeableValue. Dále budou zmíněny pouze přetížené funkce.

- **___construct**
  - @param AssociativeBaseArray|null $baseArray = null – Pokud je hodnota dojde k počátečnímu naplnění pole obsahem tohoto BaseArray.
  - @param string|null $requiredBaseClass = null – Pokud je hodnota naplněna neumožní vložení jiné hodnoty, než té, která je nebo má jako předka danou třídu.
  - @throws InvalidArgumentException
- **requiredBaseClass**
  - @return string – Vrací aktuální hodnotu požadované třídy.
- **add** – Vloží hodnotu na pozici určenou jejím klíčem.

  - *@param \ECGM\Int\KeyeableValue $obj
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **setList** – Přepíše všechny aktuální hodnoty v pole obsahem proměnné $list.
  - *@param \ECGM\Int\KeyeableValue[] $list
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **mergeList** – Obsah proměnné $listy vloží do aktuálního pole. Pokud by hodnota v poli a hodnota v proměnné $list sdílely klíč, přepisuje hodnota v $list aktuální hodnotu.
  - *@param \ECGM\Int\KeyeableValue[] $list
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **remove** – Smaže objekt, na pozici parametru $key.
  - *@param mixed $key
- **removeByObject** – Smaže objekt, který je roven parametru $obj, pokud se v poli nachází.
  - *@param \ECGM\Int\KeyeableValue $obj
- **getObj** – Vrátí objekt, které se nachází na pozici dané hodnotou $key. Vrací null, pokud daná pozice neexistuje.
  - *@param mixed $key
  - @return mixed|null

3.3 \ECGM\Model\Product

Implementuje \ECGM\Int\KeyeableValue

- **___construct**
  - *@param mixed $id – Unikátní identifikátor produktu.
  - *@param float $price – Cena produktu.
  - *@param int $expiration – Určuje dobu, za jakou by po nákupu daného produktu, mohl tento produkt znovu chtít koupit (například u spotřebního zboží bude expirace zpravidla v řádech desítek dní).
  - @param \ECGM\Enum\DateType $expirationDateType = \ECGM\Enum\DateType::DAYS
  - @param float = 0.0 $discount – Aktuální sleva aplikovaná na produkt.
  - @throws \ECGM\Exceptions\InvalidArgumentException

  - @throws \ECGM\Exceptions\ReflectionException
- **getComplements**
  - @return \ECGM\Model\BaseArray
- **setComplements** – Produkty, které jsou doplňkem tohoto produktu (například židle ke stolu apod.).
  - *@param \ECGM\Model\BaseArray $complements
  - @throws \ECGM\Exceptions\InvalidArgumentException $complements
- **getExpirationDateType**
  - @return \ECGM\Enum\DateType
- **setExpirationDateType**
  - *@param \ECGM\Enum\DateType $expirationDateType
- **getId**
  - @return mixed
- **getPrice**
  - @return float
- **setPrice**
  - *@param float $price
- **getDiscountedPrice**
  - @return float|int
- **getDiscount**
  - @return float
- **setDiscount**
  - *@param float $discount
- **getExpiration**
  - @return int
- **setExpiration**
  - *@param int $expiration

3.4 \ECGM\Model\ProductComplement

Určuje doplněk produktu. Tedy produkt, který nějakým způsobem s daným produktem souvisí a zákazník by ho mohl chtít koupit po nákupu produktu, který tento doplněk má. Například židle ke stolu.

- **___construct**
  - *@param \ECGM\Model\Product

  getId – Vrací id produktu.
  - @return mixed

  getPrice
  - @return float

3.5 \ECGM\Model\CurrentProduct

Rozšiřuje \ECGM\Model\Product

- **___construct**
  - *@param $id
  - *@param float $price
  - *@param integer $expiration
  - *@param mixed $ppc – Aktuální Product Payoff Coefficient produktu.
  - @param float $discount=0.0
  - @throws \ECGM\Exceptions\InvalidArgumentException
  - @throws \ReflectionException

3.6 \ECGM\Model\Parameter

Jeden zákazníkův parametr.

- **___construct**
  - *@param mixed $id
  - *@param float $value
  - @param boolean $isCircular=false – Specifikuje, že je daný parametr kruhový. Příklad může být hodina ve dni (1-24), nebo měsíc v roce (1-12).
  - @param float $maxValue=0 – Pokud je $isCircular nastaveno na true, je nutné zadat maximální velikost, jakou může daný parametr nabývat. U hodiny ve dni by se jednalo o hodnotu 24.
- **getId**
  - @return mixed
- **isCircular**

  - @return bool
- **getMaxValue**
  - @return float
- **getCustomer**
  - @return \ECGM\Model\Customer
- **setCustomer**
  - *@param \ECGM\Model\Customer $customer
- **getValue**
  - @return mixed

3.7 \ECGM\Model\Customer

- **___construct**
  - *@param mixed $id
  - @param CustomerGroup|null $group = null
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **addParameter** – Je nutné aby parametry byly u všech zákazníků ve stejném pořadí.
  - *@param \ECGM\Model\Parameter $parameter
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **removeParameter**
  - *@param int $parameterId
- **removeParameter** – Vrací historii nákupů (\ECGM\Model\Order) daného zákazníka.
  - @return \ECGM\Model\BaseArray
- **setHistory**
  - *@param int \ECGM\Model\BaseArray $history
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **addOrder** – Vloží jeden nákup do historie zákazníka.
  - *@param int \ECGM\Model\Order $order
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **removeOrder**
  - *@param int $orderId

- **getParameters**
  - @return \ECGM\Model\BaseArray
- **getParameters**
  - *@param \ECGM\Model\BaseArray $parameters
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getGroup**
  - @return \ECGM\Model\CustomerGroup
- **setGroup**
  - *@param \ECGM\Model\CustomerGroup $group
- **getId**
  - @return mixed
- **getParametersAsSimpleArray** – Vrátí hodnoty parametrů zákazníka jako jednoduché indexované pole.
  - @return array

3.8 \ECGM\Model\CustomerGroup

- **___construct**
  - *@param mixed $id
  - @param \ECGM\Model\BaseArray $parameters = null
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **mergeCustomers**
  - *@param \ECGM\Model\BaseArray $customers
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **mergeCustomers**
  - *@param \ECGM\Model\Customer $customer
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **removeCustomer**
  - *@param \ECGM\Model\Customer $customer
- **removeCustomers**
  - *@param \ECGM\Model\BaseArray $customers
- **addParameter**
  - *@param \ECGM\Model\Parameter $parameter

  - @throws \ECGM\Exceptions\InvalidArgumentException
- **removeParameter**
  - *@param int $parameterId
- **getId**
  - @return \ECGM\Model\BaseArray
- **getParameters**
  - @return \ECGM\Model\BaseArray
- **setParameters**
  - *@param \ECGM\Model\BaseArray $parameters
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getParametersAsSimpleArray**
  - @return array
- **setCustomers**
  - *@param \ECGM\Model\BaseArray $customers
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getCustomers**
  - @return \ECGM\Model\BaseArray

3.9 \ECGM\Model\Order

- **___construct**
  - *@param mixed $id
  - *@param BaseArray $parameters – Parametry zákazníka v době objednávky.
  - *@param \DateTime $orderDate
  - @throws \\ECGM\Exceptions\InvalidArgumentException
- **getId**
  - @return mixed
- **addProduct**
  - *@param \ECGM\Model\OrderProduct $product
  - @throws \\ECGM\Exceptions\InvalidArgumentException
- **removeProduct**
  - *@param int $productId
- **getCustomerParameters**

  - @return \ECGM\Model\BaseArray
- **addProduct**
  - *@param \ECGM\Model\BaseArray $customerParameters
  - @throws \\ECGM\Exceptions\InvalidArgumentException
- **getProducts**
  - @return \ECGM\Model\BaseArray

3.10 \ECGM\Model\StrategyProduct

Zjednodušená reprezentace produktu pro použití při výpočtu strategie zákazníka.

- **___construct**
  - *@param mixed $id
  - *@param mixed $orderId
  - *@param float $price
  - *@param int $amount
  - *@param float $discount
- **getId**
  - @return mixed
- **getOrderId**
  - @return mixed
- **getAmount**
  - @return int
- **setAmount**
  - *@param int $amount
- **getPrice**
  - @return float
- **setPrice**
  - *@param float $price
- **getDiscountedPrice**
  - @return float|int
- **getDiscount**
  - @return float
- **setDiscount**

- *@param float $discount

3.11 \ECGM\Model\OrderProduct

Rozšiřuje \ECGM\Model\StrategyProduct. Pro využití v \ECGM\Model\Order.

- **___construct**
  - *@param \ECGM\Model\Product $product
  - *@param \ECGM\Model\Order $order
  - *@param int $amount
- **getExpirationDateType**
  - @return \ECGM\Enum\DateType
- **getComplements**
  - @return \ECGM\Model\BaseArray
- **getExpiration**
  - @return int
- **getOrder**
  - @return \ECGM\Model\Order
- **setOrder**
  - *@param \ECGM\Model\Order $order

4 CONTROLLERY

4.1 \ECGM\Controller\ CustomerParametersMergeController

Implementuje \ECGM\Int\CustomerParametersMergeInterface

Pomocí této třídy dochází k vytvoření váženého průměru parametrů napříč zákaznícko-vou historii. U extrémních hodnot dochází ke snížení vah, ale ne k jejich odříznutí.

- **mergeCustomerHistory**
  - *@param \ECGM\Model\BaseArray $customerHistory
  - @return \ECGM\Model\BaseArray
  - @throws \ECGM\Exceptions\InvalidArgumentException
  - @throws \ECGM\Exceptions\UndefinedException

4.2 \ECGM\Controller\ CustomerParametersCleaningController

Implementuje \ECGM\Int\CustomerParametersCleaningInterface

Pomocí této třídy dochází k očištění parametrů zákazníka a k vytvoření jejich průměrů. Primárně při očišťování dochází k transformaci kruhových hodnot na body ve 2D prostoru.

- **getCustomerParametersMergeController**
  - @return \ECGM\Int\CustomerParametersMergeInterface
- **setCustomerParametersMergeController**
  - *@param \ECGM\Int\CustomerParametersMergeInterface $customerParame-tersMergeController
- **cleanCustomerGroups**
  - *@param \ECGM\Model\BaseArray $customerGroups
  - @return \ECGM\Model\BaseArray
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **cleanCustomerGroup**
  - *@param \ECGM\Model\CustomerGroup $customerGroup

  - @return \ECGM\Model\CustomerGroup
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **cleanCustomers**
  - *@param \ECGM\Model\BaseArray $customers
  - @return \ECGM\Model\BaseArray
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **cleanCustomer**
  - *@param \ECGM\Model\Customer $customer
  - @return \ECGM\Model\Customer
  - @throws \ECGM\Exceptions\InvalidArgumentException

4.3 \ECGM\Controller\CustomerGroupingController

Implementuje \ECGM\Int\CustomerGroupingInterface

Zapouzdřuje shlukovací algoritmy a pomocí nich shlukuje zákazníky. Volaní by mělo probíhat pouze na zákazníky s již očištěnými parametry. V opačném případě může dojít k nesprávnému zařazení.

- **___construct**
  - *@param \ECGM\MainInterface $mainInterface
  - *@param int $dimension
  - *@param int $initK – Počáteční počet shluků. Pokud je $autoKAdjustment nastaveno na false, nebude se toto číslo měnit.
  - @param bool $autoKAdjustment = true – Pokud je nastaveno na true, bude docházet k automatickému upravovaní počtu shluků. Jedná se o výpočetně náročný proces, který není doporučen pokud je počet zákazníků větší než 5000.
  - @param bool $verbose = false – Pokud bude nastaveno na true, bude docházet k zalogování aktivity shlukovacího algoritmu.
  - @throws \ECGM\Exceptions\InvalidArgumentException
  - @throws \ECGM\Exceptions\UndefinedException
- **getValidationClass**
  - @return \ECGM\Int\GroupingValidationInterface
- **setValidationClass**

  - *@param \ECGM\Int\GroupingValidationInterface $validationClass
- **getGroupingClass**
  - @return \ECGM\Int\GroupingImplementationInterface
- **setGroupingClass**
  - *@param \ECGM\Int\GroupingImplementationInterface $groupingClass
- **getDistanceFunctions**
  - @return \ECGM\Int\DistanceFuncInterface
- **setDistanceFunctions**
  - *@param \ECGM\Int\DistanceFuncInterface $distanceFunctions
- **groupCustomers**
  - *@param \ECGM\Model\BaseArray $customers
  - @param \ECGM\Model\BaseArray $initialGroups = null – Pokud dojde k naplnění této proměnné, nebudou se vytvářen počáteční shluky, ale dojde k přidání zákazníků do těchto shluků.
  - @return \ECGM\Model\BaseArray|mixed|null
  - @throws \ECGM\Exceptions\InvalidArgumentException
  - @throws \ECGM\Exceptions\LogicalException
- **assignToGroup**
  - *@param \ECGM\Model\Customer $customer
  - *@param \ECGM\Model\BaseArray $groups
  - @return \ECGM\Model\Customer
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getDimension**
  - @return int
- **getK**
  - @return int

4.4 \ECGM\Controller\DealerStrategyController

Implementuje \ECGM\Int\DealerStrategyInterface.

- **getDealerStrategy**
  - *@param \ECGM\Model\BaseArray $products
  - @return array

- @throws \ECGM\Exceptions\InvalidArgumentException

4.5 \ECGM\Controller\CustomerStrategyController

Implementuje \ECGM\Int\CustomerStrategyInterface.

- **___construct**
  - *@param \ECGM\MainInterface $mainInterface
  - *@param float $coefficient – Hodnota této proměnné je využita pro zvýšení váhy nákupů zákazníka oproti nákupům jeho skupiny a ke zvýšení váhy doplňků. Čím vyšší, tím více budou tyto hodnoty upřednostňovány.
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getCustomerStrategy**
  - *@param \ECGM\Model\Customer $customer
  - *@param \ECGM\Model\BaseArray $currentProducts
  - @param \ECGM\Model\Order|null $currentOrder = null – Hodnota reprezentující aktuální nákup zákazníka, pokud již má nějaké produkty v košíku.
  - @return array
  - @throws \ECGM\Exceptions\InvalidArgumentException

4.6 \ECGM\Controller\StrategyController

Implementuje \ECGM\Int\StrategyInterface.

- **___construct**
  - *@param float $coefficient – Hodnota této proměnné je využita pro zvýšení váhy nákupů zákazníka oproti nákupům jeho skupiny a ke zvýšení váhy doplňků. Čím vyšší, tím více budou tyto hodnoty upřednostňovány.
  - *@param \ECGM\MainInterface $mainInterface
  - @param int $maxProductsInStrategy = 40 – Počet produktů, které budou zahrnuty do výpočtu strategie. Pokud bude tato proměnná nastavena na 40, bude do strategie zahrnuto prvních 40 produktů, které jsou ve strategii zákazníka nejvýše.
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getPassiveStrategyController**

  - @return \ECGM\Int\StrategyTypeInterface
- **setPassiveStrategyController**
  - *@param \ECGM\Int\StrategyTypeInterface $passiveStrategyController
- **getConservativeStrategyController**
  - @return \ECGM\Int\StrategyTypeInterface
- **setConservativeStrategyController**
  - *@param \ECGM\Int\StrategyTypeInterface $conservativeStrategyController
- **getAggressiveStrategyController**
  - @return \ECGM\Int\StrategyTypeInterface
- **setAggressiveStrategyController**
  - *@param \ECGM\Int\StrategyTypeInterface $aggressiveStrategyController
- **getStrategy**
  - *@param \ECGM\Model\Customer $customer
  - *@param \ECGM\Model\AssociativeBaseArray $currentProducts – Produkty, které jsou aktuálně v prodeji.
  - @param \ECGM\Model\Order|null $currentOrder = null
  - *@param \ECGM\Enum\StrategyType $strategyType = \ECGM\Enum\StrategyType::CONSERVATIVE
  - @return \ECGM\Model\AssociativeBaseArray
  - @throws \ECGM\Exceptions\InvalidArgumentException
  - @throws \ECGM\Exceptions\LogicalException
  - @throws \ReflectionException

4.7 \ECGM\Controller\PassiveStrategyTypeController

Implementuje \ECGM\Int\StrategyTypeInterface.

Tento typ strategie určí pořadí produktů v jakém je nejvýhodnější zákazníkovi prodat, tak aby byla zároveň co nejlépe zachována předpovězená strategie daného zákazníka. Nedochází ke změně slev produktů.

- **___construct**
  - *@param float $coefficient – Hodnota této proměnné je využita pro zvýšení váhy nákupů zákazníka oproti nákupům jeho skupiny a ke zvýšení váhy doplňků. Čím vyšší, tím více budou tyto hodnoty upřednostňovány.

  - *@param \ECGM\MainInterface $mainInterface
  - @param int $maxProductsInStrategy = 40 – Počet produktů, které budou zahrnuty do výpočtu strategie. Pokud bude tato proměnná nastavena na 40, bude do strategie zahrnuto prvních 40 produktů, které jsou ve strategii zákazníka nejvýše.
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getCustomerStrategyController**
  - @return \ECGM\Int\CustomerStrategyInterface
- **setCustomerStrategyController**
  - *@param \ECGM\Int\CustomerStrategyInterface $customerStrategyController
- **getDealerStrategyController**
  - @return \ECGM\Int\DealerStrategyInterface
- **setDealerStrategyController**
  - *@param \ECGM\Int\DealerStrategyInterface $dealerStrategyController
- **getIdealStrategy**
  - *@param \ECGM\Model\Customer $customer
  - *@param \ECGM\Model\AssociativeBaseArray $currentProducts – Produkty, které jsou aktuálně v prodeji.
  - @param \ECGM\Model\Order|null $currentOrder = null
  - @return \ECGM\Model\AssociativeBaseArray – Produkty jsou vraceny sestupně od nejlepšího po nejhorší.
  - @throws \ECGM\Exceptions\InvalidArgumentException

4.8 \ECGM\Controller\ConservativeStrategyTypeContr

Implementuje \ECGM\Int\StrategyTypeInterface.

Tento typ strategie navrhne minimální možnou slevu takovou, aby došlo k upřednostnění produktu ve strategii zákazníka, ale zároveň nedošlo k jeho poklesu ve strategii prodejce. Slevy jsou přiřazovány pouze produktům, u kterých mají smysl, to znamená, že pokud by nebyla sleva pro prodejce výhodná, není aplikována.

- **___construct**

  - *@param float $coefficient – Hodnota této proměnné je využita pro zvýšení váhy nákupů zákazníka oproti nákupům jeho skupiny a ke zvýšení váhy doplňků. Čím vyšší, tím více budou tyto hodnoty upřednostňovány.
  - *@param \ECGM\MainInterface $mainInterface
  - @param int $maxProductsInStrategy = 40 – Počet produktů, které budou zahrnuty do výpočtu strategie. Pokud bude tato proměnná nastavena na 40, bude do strategie zahrnuto prvních 40 produktů, které jsou ve strategii zákazníka nejvýše.
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getCustomerStrategyController**
  - @return \ECGM\Int\CustomerStrategyInterface
- **setCustomerStrategyController**
  - *@param \ECGM\Int\CustomerStrategyInterface $customerStrategyController
- **getDealerStrategyController**
  - @return \ECGM\Int\DealerStrategyInterface
- **setDealerStrategyController**
  - *@param \ECGM\Int\DealerStrategyInterface $dealerStrategyController
- **getIdealStrategy**
  - *@param \ECGM\Model\Customer $customer
  - *@param \ECGM\Model\AssociativeBaseArray $currentProducts – Produkty, které jsou aktuálně v prodeji.
  - @param \ECGM\Model\Order|null $currentOrder = null
  - @return \ECGM\Model\AssociativeBaseArray – Produkty jsou vraceny sestupně od nejlepšího po nejhorší, společně s navrženými slevami.
  - @throws \ECGM\Exceptions\InvalidArgumentException

4.9 \ECGM\Controller\AggressiveStrategyTypeControll

Implementuje \ECGM\Int\StrategyTypeInterface.

Tento typ strategie navrhne maximální možnou slevu takovou, aby došlo k upřednostnění produktu ve strategii zákazníka, ale zároveň nedošlo k jeho poklesu ve strategii prodejce. Slevy jsou přiřazovány pouze produktům, u kterých mají smysl, to znamená, že pokud by nebyla sleva pro prodejce výhodná, není aplikována.

- **___construct**
  - *@param float $coefficient – Hodnota této proměnné je využita pro zvýšení váhy nákupů zákazníka oproti nákupům jeho skupiny a ke zvýšení váhy doplňků. Čím vyšší, tím více budou tyto hodnoty upřednostňovány.
  - *@param \ECGM\MainInterface $mainInterface
  - @param int $maxProductsInStrategy = 40 – Počet produktů, které budou zahrnuty do výpočtu strategie. Pokud bude tato proměnná nastavena na 40, bude do strategie zahrnuto prvních 40 produktů, které jsou ve strategii zákazníka nejvýše.
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getCustomerStrategyController**
  - @return \ECGM\Int\CustomerStrategyInterface
- **setCustomerStrategyController**
  - *@param \ECGM\Int\CustomerStrategyInterface $customerStrategyController
- **getDealerStrategyController**
  - @return \ECGM\Int\DealerStrategyInterface
- **setDealerStrategyController**
  - *@param \ECGM\Int\DealerStrategyInterface $dealerStrategyController
- **getIdealStrategy**
  - *@param \ECGM\Model\Customer $customer
  - *@param \ECGM\Model\AssociativeBaseArray $currentProducts – Produkty, které jsou aktuálně v prodeji.
  - @param \ECGM\Model\Order|null $currentOrder = null
  - @return \ECGM\Model\AssociativeBaseArray – Produkty jsou vraceny sestupně od nejlepšího po nejhorší, společně s navrženými slevami.
  - @throws \ECGM\Exceptions\InvalidArgumentException

5 POMOCNÉ TŘÍDY

5.1 \ECGM\Util\DistanceFunctions

Implementuje \ECGM\Int\DistanceFuncInterface.

- **distanceQuick** – Rychlejší, ale méně přesná implementace distančního algoritmu. V tomto případě se jedná o Manhattanskou vzdálenost.
  - *@param array $v1
  - *@param array $v2
  - @return float|int
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **distancePrecise** – Pomalejší, ale přesnější implementace distančního algoritmu. V tomto případě se jedná o Euklidovskou vzdálenost.
  - *@param array $v1
  - *@param array $v2
  - @return float|int
  - @throws \ECGM\Exceptions\InvalidArgumentException

5.2 \ECGM\Util\MathFunctions

- **arrayMedian**
  - *@param array $array
  - @return float|int
  - @throws \ECGM\Exceptions\UndefinedException

5.3 \ECGM\Util\MiscFunctions

- **mergeAssociativeArrays** – Sloučí asociativní pole.
  - *@param array $arr1
  - *@param array $arr2
  - @return array
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **isAssoc**

- *@param array $arr
- @return bool

5.4 \ECGM\Util\KmeansPlusPlus

Implementuje \ECGM\Int\GroupingImplementationInterface.

Implementace shlukovacího algoritmu k-means++.

- **___construct**
  - *@param int $dimension
  - @throws \ECGM\Exceptions\InvalidArgumentException
  - @throws \ECGM\Exceptions\UndefinedException
- **getDistanceFunctions**
  - @return \ECGM\Int\DistanceFuncInterface
- **setDistanceFunctions**
  - *@param \ECGM\Int\DistanceFuncInterface $distanceFunctions
- **setInitialGroups**
  - *@param \ECGM\Model\BaseArray $initialGroups
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **addCustomer**
  - *@param \ECGM\Model\Customer $customer
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **setCustomers**
  - *@param \ECGM\Model\BaseArray $customers
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getDimension**
  - @return int
- **solve**
  - *@param int $nbGroups
  - @return \ECGM\Model\BaseArray|mixed|null
  - @throws \ECGM\Exceptions\InvalidArgumentException
- **getCustomerParametersAsArray**
  - *@param \ECGM\Model\BaseArray $parameters
  - @return array

  - @throws \ECGM\Exceptions\InvalidArgumentException

5.5 \ECGM\Util\SilhouetteAnalysis

Implementuje \ECGM\Int\GroupingValidationInterface.

Implementace siluetové analýzy, pro měření správnosti zařazení objektu do shluku.

- **___construct**
  - @param bool $verbose = false – Pokud nastaveno na true, dojde k logování výsledků.
- **getDistanceFunctions**
  - @return \ECGM\Int\DistanceFuncInterface
- **setDistanceFunctions**
  - *@param \ECGM\Int\DistanceFuncInterface $distanceFunctions
- **getDistanceFunctions**
  - *@param \ECGM\Model\DistanceFuncInterface $groups
  - @return float|int
  - @throws \ECGM\Exceptions\InvalidArgumentException
  - @throws \ECGM\Exceptions\LogicalException