ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

FAKULTA
STROJNÍ

DIPLOMOVÁ PRÁCE

2021

JAN
SLABÝ

České vysoké učení technické v Praze
Fakulta strojní

Ústav přístrojové a řídicí techniky

Aplikace metod prediktivní údržby a plánování výroby

Predictive Maintenance and Production Planning Methods Application

DIPLOMOVÁ PRÁCE

Vypracoval: Bc. Jan Slabý
Vedoucí práce: Ing. Cyril Oswald, Ph.D.
Rok: 2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Slabý** Jméno: **Jan** Osobní číslo: **457548**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídicí techniky**

Studijní program: **Strojní inženýrství**

Studijní obor: **Přístrojová a řídicí technika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Aplikace metod prediktivní údržby a plánování výroby

Název diplomové práce anglicky:

Predictive Maintenance and Production Plannig Methods Application

Pokyny pro vypracování:

1. Navrhněte systém pro monitorování průmyslové výroby s cílem aplikace prediktivní údržby výrobních linek.
2. Proveďte rešerši aktuálních metod prediktivní údržby a plánování výroby využívající algoritmy strojového učení a z nich vytipujte vhodné metody pro aplikaci.
3. Navrhněte SW nástroj implementující vytipované metody, jehož vstupní data budou ve Vámi navrženém unifikovaném formátu a výstupní data ve vhodném formátu pro jejich vizualizaci použitím aktuálně rozšířených SW nástrojů (např. Power BI).
4. Navržený SW nástroj realizujte a aplikujte na vhodných datech z výrobního procesu.

Seznam doporučené literatury:

[1] An introduction to predictive maintenance [elektronický zdroj] / R. Keith Mobley.https://www.sciencedirect.com/book/9780750675314/an-introduction-to-predictive-maintenanc
[2] Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challengeshttps://www.sciencedirect.com/science/article/pii/S0166361520305327
[3] Fault Class Prediction in Unsupervised Learning Using Model-Based Clustering Approach https://ieeexplore.ieee.org/document/8356831

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Cyril Oswald, Ph.D., U12110.3

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **30.10.2020** Termín odevzdání diplomové práce: **20.01.2021**

Platnost zadání diplomové práce: _____

Ing. Cyril Oswald, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Michael Valášek, DrSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

Datum převzetí zadání _____ Podpis studenta _____

Čestné prohlášení

Prohlašuji, že jsem tuto práci vypracoval samostatně s použitím literárních zdrojů a informací, které cituji a uvádím v seznamu použité literatury a zdrojů.

Datum: . . . .

Podpis

Poděkování

Tímto bych rád poděkoval panu Ing. Cyrilu Oswaldovi, Ph.D. za ochotné odborné vedení mé diplomové práce, veškerý jeho čas, trpělivost a užitečné rady, které mi pomohly při zpracování této práce. V neposlední řadě bych chtěl poděkovat své rodině za podporu během celého studia.

Anotační list

Jméno autora:	Jan Slabý
Název DP:	Aplikace metod prediktivní údržby a plánování výroby
Anglický název:	Predictive Maintenance and Production Planning Methods Application
Rok:	2021
Ústav:	Ústav přístrojové a řídicí techniky
Vedoucí práce:	Ing. Cyril Oswald, Ph.D.
Bibliografické údaje:	Počet stran 60
	Počet obrázků 37
	Počet tabulek 2
	Počet příloh 2
Klíčová slova:	prediktivní údržba, průmyslová zařízení, Python, Dash, MS SQL, PowerBI, prediktivní algoritmy
Keywords:	Predictive maintenance, industrial equipment, Python, Dash, MS SQL, PowerBI, predictive algorithms

Anotace

Tato diplomová práce se zabývá vývojem prediktivní aplikace pro podporu údržby výrobních linek v oblasti potravinářského průmyslu. V první části práce je provedena rešerše jednotlivých přístupů údržby průmyslových zařízení včetně běžně používaných ukazatelů pro měření efektivity výroby a údržby. Rešeršní část je dále zaměřena na prohloubení znalostí v oblasti analýzy dat, a to především za využití prediktivních algoritmů. Dalšími tématy obsaženými v rešeršní části práce jsou ukládání dat a jejich vizualizace jakožto podstatné oblasti nutné pro vývoj prediktivních aplikací. V praktické části je popsán návrh a vývoj aplikace sloužící k predikci frekvence výskytu krátkých zastávek na výrobní lince. Dále je v práci využit software *PowerBI*, který byl využit jako reportovací nástroj doplňující vytvořenou prediktivní aplikaci.

Abstract

This diploma thesis deals with the development of a predictive application intended for maintenance support of production lines in the food industry. The first part of the thesis is dedicated to summarizing the individual approaches to maintenance of industrial machinery, including commonly used indicators for the measurement of production and maintenance effectiveness. This part further focuses on deepening knowledge in the area of data analysis, primarily through use of predictive algorithms. Other themes included in this part are data storage and visualization, which represent a key area needed for the development of predictive applications. The practical part then describes the design and development of an application with the purpose of predicting frequency of short stop occurrence on the production line. The thesis further includes the use of the *PowerBI* software, which was used as a reporting tool to complement this developed predictive application.

Obsah

Seznam zkratek

TPM:	Total Productive Maintenance	**CRUD:**	Create, Read, Update, Delete
KPI:	Key Performance Indicator	**MLP:**	Multilayer Perceptron
OEE:	Overall Equipment Effectiveness	**MSE:**	Mean Squared Error
MTBF:	Mean Time Between Failures	**SVM:**	Support Vector Machine
MTTR:	Mean Time to Repair	**KNN:**	K-Nearest Neighbor
UPDT:	Unplanned Downtime	**IoT:**	Internet of Things
PDT:	Planned Downtime	**DBMS:**	Database Management System
BI:	Business Inteligence	**SQL:**	Structured Query Language
ETL:	Extract, Transform, Load	**HDFS:**	Hadoop Distributed File System
SSIS:	SQL Server Integration Services	**GFS:**	Google File System
MES:	Manufacturing Execution System	**AWS:**	Amazon Web Services
PLM:	Product Lifecycle Management	**RFC:**	Random Forest Classifier
YTD:	Year to Date	**SVC:**	Support Vector Classifier

1 Úvod

Mou motivací k vypracování diplomové práce obsahující téma prediktivní údržby je můj dlouhodobý zájem o toto téma a zároveň záměr věnovat se oblasti zlepšování výrobních procesů v průmyslových provozech v praxi po ukončení studia. Průmyslové podniky postupně zaměřují stále více energie a finančních prostředků do oblasti analýzy dat z výrobních linek, což je jeden z pozorovatelných jevů čtvrté průmyslové revoluce. Upouští se tedy od metod zvyšujících produktivitu výrobních linek primárně skrze investice do fyzických částí stroje a postupně je kladen větší důraz na zvyšování produktivity zařízení skrze odhalování slabých míst za využití analýz velkého množství generovaných dat. Předpokladem k tomuto posunu je dostupnost a nízké pořizovací náklady chytrých senzorů a snímačů, datových úložišť a zároveň rychlým tempem rostoucí výpočetní výkon jak standardních osobních počítačů, tak především serverů dedikovaných k analýzám dat. Analýza výrobních dat a její automatizace za využití metod strojového učení je jistě perspektivním oborem, který má potenciál zvýšit produktivitu průmyslových podniků.

V rámci této diplomové práce je popsána tvorba webové aplikace, jejímž cílem je predikování frekvence krátkých zastávek na výrobní lince v oblasti potravinářského průmyslu. Ta by měla sloužit oddělení údržby a procesním inženýrům k včasnému odhalení potenciálního výskytu krátkých zastávek a umožnit jejich předejití nebo minimalizaci. Tuto predikční funkci aplikace jsem dále doplnil o funkci vizualizační, která umožňuje analýzu dat historických i predikovaných. Následně je v práci popsáno využití grafických přehledů vytvořených v softwaru *PowerBI*, které by měly sloužit širší skupině uživatelů.

V první části diplomové práce se zaměřuji na prohloubení znalostí v oblastech, jejichž znalost je nutným základem pro tvorbu prediktivních aplikací. Kapitola 2 je věnována různým přístupům v oblasti průmyslové údržby a jedna její část je vyhrazena pro popis konceptu totálně produktivní údržby, který je v různých obdobách hojně využíván v průmyslové praxi. V kapitole 3 se věnuji základním ukazatelům výkonosti, jejichž nastavení a pravidelné měření je jedním ze základních předpokladů pro úspěšné a udržitelné zavedení prediktivní údržby. Kapitola 4 následně poskytuje přehled existujících nástrojů pro analýzu dat. Dalšími body této kapitoly jsou algoritmy strojového učení využívané v oblasti prediktivní údržby. Kapitola 5 je zaměřena na metody ukládání velkého množství dat a slouží jako základ pro rozhodování o volbě úložiště dat, které jsem v rámci tvorby aplikace využil. Teoretická část je zakončena souhrnem existujících řešení pro vizualizaci dat, kde je zároveň diskutován přínos těchto řešení společně s finančními náklady s nimi spojenými.

V rámci praktické části jsem před samotným programováním aplikace musel zajistit několik základních kroků. Prvním z těchto kroků je získaní dat, která jsou pro predikce v rámci aplikace využita.

Tato data bylo nutné vytěžit z produkčních systémů výrobní linky a provést jejich anonymizaci pro zajištění jejich zabezpečení. Těmto krokům a výběru vhodného úložiště dat jsou věnovány kapitoly 8 a 9. Dále je v práci popsán postup analýzy dat, kterou jsem musel provést k výběru vhodné oblasti pro nasazení prediktivních algoritmů a pro proces čištění těchto dat. Kapitola 11 je věnována přípravě a výběru vhodných prediktivních algoritmů, programování webové aplikace v Pythonu a popisu jejího uživatelského rozhraní. V kapitole 12 se věnuji přípravě *PowerBI* reportů, které slouží k rozšíření vytvořené aplikace a prezentaci jejích výsledků širší populaci zaměstnanců. Kapitola 14 obsahuje přehled funkcionalit, o které by bylo vhodné aplikaci dále rozšířit v případě jejího úspěšného otestování v reálném provozu.

1.1 Cíle diplomové práce

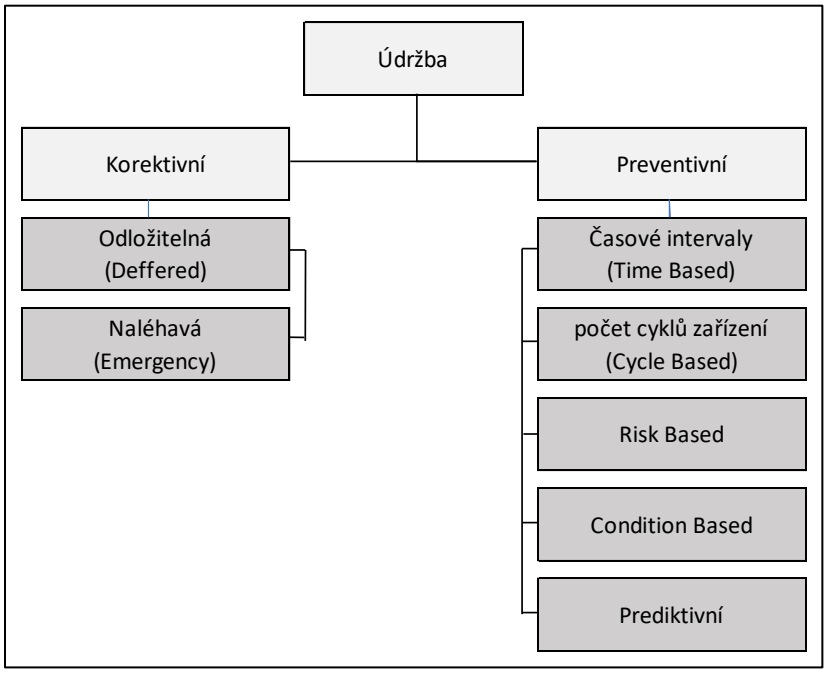
Za cíl této práce si kladu vytvořit prediktivní webovou aplikaci, která bude sloužit jako podpůrný nástroj pro oddělení údržby. Na základě rešerše zaměřené na oblast průmyslové údržby a následné analýzy dat ze zvolené výrobní linky vyberu vhodnou oblast pro nasazení prediktivních algoritmů. Následně využitím vhodného programovacího jazyka vytvořím aplikaci, jež by měla poskytovat predikce, na základě kterých bude možno provádět zásahy vedoucí ke zvýšení výkonů dané výrobní linky. Dále k vytvořené aplikaci připravím vizualizaci historických i predikovaných dat v jednom z rozšířených vizualizačních nástrojů. Konkrétní nástroj bude vybrán na základě provedené rešerše. Pro účely praxe bude celý tento koncept v budoucnu otestován a v případě, že bude zhodnocen jako vhodné řešení, dojde k přidání dalších funkcionalit, které budou diskutovány již v této práci.

2 Údržba v průmyslové výrobě

Údržba zařízení v průmyslové výrobě je jedním z klíčových pilířů pro zajištění dlouhodobé konkurenceschopnosti a stabilní výroby. Zároveň náklady s údržbou spojené představují významnou část z celkových nákladů společnosti. Zdroj [1] vyčísluje náklady spojené s údržbou na 15 až 60 procent z celkových výrobních nákladů. Na spodní hranici 15 procent se pohybují potravinářské podniky, na straně druhé stojí těžký průmysl s až 60 procenty. Část těchto nákladů má původ ve špatně provedené, popřípadě neefektivně plánované údržbě. Vícenáklady spojené s neefektivitou prováděné údržby dosahují dle [1, 2] až třetiny z celkových nákladů na údržbu. Jen ve Spojených státech neefektivita údržby představuje až 60 miliard dolarů ročně. Oddělení údržby přímo i nepřímo ovlivňuje fungování ostatních oddělení, jako jsou oddělení interní logistiky, plánování nebo oddělení výroby samotné. Nevhodně navržený systém údržby může vést k vysokým skladovým zásobám, nutnosti vyššího počtu zaměstnanců, nebo až k nedostatku zboží na trhu a oslabení pozice společnosti v dnešním vysoce konkurenčním prostředí. Investice do vývoje v oblasti průmyslové údržby společně se správně

nastaveným programem údržby jsou cestou ke snížení nákladů, zvýšení produkce a mohou napomoci celkovému zeštíhlení výrobního procesu [2].

V průmyslové praxi existuje celá řada metod a přístupů pro průmyslovou údržbu. Tyto metody je možné rozdělit na dvě skupiny podle užitého přístupu. První skupinou jsou takzvané korektivní metody. Korektivní údržba (v angličtině Corrective Maintenance nebo Run to Failure Maintenance) připouští poruchu stroje a kroky údržby jsou prováděny až po takovém incidentu. Druhou skupinou je preventivní údržba, jež cílí na včasné provádění údržby a tím se snaží poruchám předcházet [3]. Přehled a rozdělení v této práci zmíněných kategorií údržby je vizualizován na obrázku 1.



Obrázek 1: Kategorie průmyslové údržby.

2.1 Korektivní údržba

Princip korektivní nebo reaktivní údržby spočívá ve využití celé životnosti zařízení nebo jejich částí či dílů. Údržba je tedy prováděna až v době poruchy zařízení, což v případě kritických zařízení může vést k vysokým finančním ztrátám. Tento typ údržby je využíván pro nekritická zařízení, jejichž příkladem může být osvětlení nebo jiná přidružená zařízení. V případě výrobní linky se může jednat o zařízení, bez nichž může být samotná výrobní linka po čas jejich prostoje stále v provozu. Dalším faktorem, který ovlivňuje možnost použití korektivní údržby, je bezpečnost, a to jak bezpečnost personálu, tak bezpečnost s ohledem na životní prostředí. U určitých zařízení je využití korektivní údržby legislativně znemožněno povinností zařízení pravidelně servisovat. Korektivní údržbu je dále možné rozdělit do dvou kategorií. První kategorií je údržba odložitelná. Jedná se tedy o údržbu, u které

je možné naplánovat její provedení, a tím pádem i snížit náklady s ní spojené. Toto není možné u druhého typu, kterým je údržba naléhavá. V tomto případě je nutné úkony údržby provést v co nejkratším čase, což může mít za následek celkovou změnu plánu pro oddělení údržby [3, 4].

Odložitelná údržba: Deferred Maintenance
Naléhavá údržba: Emergency Maintenance

Zdroj [1] uvádí až třikrát vyšší finanční náročnost Korektivní údržby proti údržbě preventivní. Důvody těchto vyšších nákladů jsou například nutnost vyšších skladových zásob náhradních dílů, nutnost neustálé přítomnosti zaměstnanců z oddělení údržby nebo snížení dostupnosti (Availability) zařízení.

2.2 Preventivní údržba

Cílem preventivní údržby, jak je z názvu zřejmé, je prevence poruch na zařízení. Toho je docíleno například prováděním údržby v pravidelných časových intervalech (Time-Based Maintenance) nebo po předem definovaném počtu cyklů (Cycle-Based Maintenance). V průběhu plánované údržby je provedena inspekce zařízení a jsou provedeny předem naplánované výměny dílů. Společným znakem všech metod preventivní údržby je plánování. Vzhledem k existenci plánu je možné plánování nákupu náhradních dílů, příprava těchto dílů a alokace týmu k provedení údržby. Stroj je tedy odstaven až ve chvíli, kdy je vše připraveno, a tím pádem nevzniká žádná ztráta způsobená čekáním na personál či náhradní díly. Při nevhodně nastaveném plánu může docházet k výměnám dílů, které ještě nejsou na hraně životnosti a jejich výměnou vzniká finanční ztráta jak na nákupu nového dílu, tak na produkci během odstavení stroje i na efektivitě využití personálu. Opačným problémem nevhodně zvoleného plánu může být vznik poruchy před plánovanou údržbou, kdy se údržba dostává do oblasti metod korektivní údržby [1]. Existují však postupy pro správné stanovení intervalů údržby i jejího rámce. Údržba může být plánována na základě postupů nazvaných „Risk Assesment Methodology“ nebo „Condition Based Methodology“, které uvažují různé výrobní podmínky a na jejich základě je určen vhodný plán údržby. Údržbu je možno plánovat na bezpečné straně, kdy hrozí neúplná utilizace náhradních dílů, nebo naopak na straně rizika, což může přispět ke zvýšení neplánovaných zastávek zařízení [5].

Kromě zmíněných metod, které pracují na principu časových intervalů nebo cyklů zařízení, můžeme do oblasti preventivní údržby zařadit i další metody. Příkladem mohou být Risk Based Maintenance nebo Condition Based Maintenance. Condition Based Maintenance, na rozdíl od klasické preventivní údržby, plánuje údržbu na základě změny sledovaných ukazatelů zařízení. Projevy poruchy jsou předpokládány ještě před samotnou poruchou. Může se například jednat o výstup vibrační analýzy, kdy se vibrace zařízení postupně mění až do stavu poruchy. Pokud je změna zaznamenána včas, preventivní údržba může být provedena a finanční ztráta spojená s poruchou snížena. Změny

v parametrech jsou v Conditional Based Maintenance porovnávány s předem stanovenými podmínkami, přičemž dosažení těchto podmínek signalizuje blížící se poruchu [3].

2.2.1 Prediktivní údržba

V některých zdrojích, jako je [1], je prediktivní údržba vedena jako samostatná kategorie, v jiných spadá do kategorie preventivní údržby. Prediktivní údržba má stejný cíl jako údržba preventivní, a to nastavit plán údržby tak, aby se poruchám předcházelo. Rozdíly existují v přístupu k plánování údržby. Cílem prediktivní údržby není jen předejít neplánovaným zastávkám způsobeným poruchou zařízení, ale zároveň i zvýšení produktivity, kvality a dalších ukazatelů, které jsou popsány ve třetí kapitole. K plánování se v prediktivní údržbě nepoužívají průměry životnosti zařízení, ale využívá se monitorování mechanického stavu stroje, výkonu stroje, fyzických parametrů komponentů vyráběného produktu a jiných ovlivňujících parametrů. Analýzou těchto informací je určen aktuální, nikoli průměrný čas životnosti nebo čas mezi poruchami [1, 4]. Cílem prediktivní údržby tedy není plánovat údržbu zařízení v daných intervalech, ale co nejvíce tyto intervaly prodloužit a zároveň předcházet neplánovaným zastávkám. Toho je možné dosáhnout především v důsledku rozmachu chytrých senzorů, které mohou poskytovat velké množství dat o pozorovaném zařízení. Důkladným zpracováním těchto dat je možné metody prediktivní údržby implementovat. Tuto datovou analýzu je nutné provádět kontinuálně a vzhledem k množství dat je nutné tento proces automatizovat.

2.3 Totálně produktivní údržba

V případě většího počtu zařízení, jako tomu je například ve výrobním podniku, je nutné zkombinovat více údržbových metod do funkčního celku. Přestože by implementace prediktivní údržby na všechny zařízení vedla k vyšší produktivitě a zaručila by snížení neplánovaných zastávek a poruch, finanční náklady takové implementace by převýšily tyto benefity. Kromě finančních nákladů je nutné volit metodu údržby podle kritérií jako jsou bezpečnost, legislativní požadavky a kritičnost zařízení.

Jedním ze způsobů, jak zajistit správné fungování a udržitelnost údržby, je koncept Totálně produktivní údržby (TPM-Total Productive Maintenance), s nímž přišel Edwards Deming v padesátých letech [1]. TPM je součástí Six Sigma metody[1] a je zásadním principem pro Lean[2] výrobu [6]. Jedná se o koncept, který definuje povinnosti v rámci programu údržby, a to nejen v rámci týmu údržby, ale napříč celou organizací. Dále definuje šest oblastí ztrát a metriky pro jejich měření. Tyto metriky jsou

[1] Metoda, která poskytuje organizacím nástroje ke zlepšení schopností v jejich podnikovém procesu. Zvýšení výkonu a snížení variability procesu pomáhají redukovat defekty a přináší zvýšení zisku a kvality výrobků a služeb.

[2] Přístup k managementu zaměřený na snižování ztrát při současném zajištění kvality. Může být aplikován na všechny složky podniku od vývoje po výrobu a distribuci.

využívány pro měření účinnosti údržby a na jejich základě je model údržby upravován. Skrze postupně implementované pilíře [7] rozšiřuje povinnosti a buduje znalost údržby přes celou organizaci. Důraz je kladen na zvyšování technických znalostí operátorů pro možnost provádění základní údržby. Jedním z cílů TPM je právě přechod z korektivní údržby na preventivní, popřípadě prediktivní.

3 Ukazatele výkonosti

Pro zajištění dlouhodobé udržitelnosti a stabilního progresu programu údržby je nutné zavést ukazatele výkonosti, klíčové metriky (anglicky KPIs – Key Performance Indicators). Na základě progresu ukazatelů výkonosti je program údržby hodnocen a případně modifikován. Tyto metriky neslouží pouze oddělení údržby, ale umožňují vhled do fungování celého procesu výroby, jehož je údržba klíčovou součástí. Hlavním ukazatelem pro TPM je metrika OEE (Overall Equipment Effectivness neboli celková efektivnost zařízení) [1, 6]. Skrze OEE je možné sledovat vývoj v šesti oblastech ztrát, které TPM definuje [7].

3.1 OEE – celková efektivnost zařízení

OEE je primárním ukazatelem TPM a je rozdělen na tři části: kvalita, výkon a dostupnost.

$$OEE\ [\%] = \text{Kvalita*Výkon*Dostupnost*}100 \tag{1}$$

Kvalita je určena podílem shodných produktů a všech vyrobených produktů. Složka výkonu zaznamenává ztráty způsobené nižší rychlostí stroje a může být vypočtena jako podíl průměrné rychlosti a normované rychlosti. Normovaná rychlost je maximální možná rychlost zařízení. Ukazatel dostupnosti vyjadřuje ztráty způsobené prostoji, opravami a poruchami strojů. Jedná se tedy o podíl skutečného času výroby a celkového plánovaného času výroby [8]. Jelikož se jedná o standardní KPI používané velkým množstvím společností, vznikají uznávané mezinárodní soutěže porovnávající OEE. Předpokladem pro vstup do takové soutěže je OEE ve výši minimálně 85 %. Dle informací uvedených v [1] se však většina společností v USA pohybuje na hranici 50% OEE.

OEE ukazatel je hlavním ukazatelem TPM a slouží k monitorování úspěšnosti výroby jako celku. Pro možnost lépe zacílit dané problémy spojené s údržbou je nutné tento ukazatel dále rozložit a sledovat ukazatele podrobnější a více zaměřené na danou problematiku.

3.2 MTBF – průměrný čas mezi zastávkami

Průměrný čas mezi zastávkami, anglicky Mean Time between Failures, je využíván pro monitorování neplánovaných zastávek. Tento ukazatel do svého výpočtu nezahrnuje plánované

zastávky, a tím pádem umožňuje sledovat, jaký je trend v oblasti odstraňování nechtěných zastávek či poruch.

$$MTBF = \frac{\text{Skutečný výrobní čas}}{\text{Počet zastávek}} \tag{2}$$

Kde *skutečný výrobní čas* vyjadřuje čas, ve kterém byl stroj v produkčním módu, tedy plánovaný čas výroby a od něj odečtena doba trvání jak plánovaných, tak neplánovaných zastávek. Tento ukazatel lze využít pro monitorování přehledu přes všechny neplánované zastávky stroje i pro monitorování konkrétní poruchy [9].

3.3 MTTR – průměrný čas opravy

Průměrný čas opravy, anglicky Mean Time to Repair, je ukazatel měřící trvání úkonů spojených s nápravou neplánovaných zastávek či poruch. Tento ukazatel umožňuje například sledování trendu při přechodu mezi korektivní a preventivní údržbou, kdy by měl nastat znatelný progres v trvání oprav.

$$MTTR = \frac{\text{Celkový čas trvání poruch}}{\text{počet zatávek}} \tag{3}$$

Tento ukazatel umožňuje lepší vhled do problematiky oprav a na jeho základě je následně možné definovat slabá místa v procesu údržby, na které je následně možné adekvátně reagovat [9].

3.4 Procento plánovaných zastávek

Anglicky Planned Downtime (PDT). Tento ukazatel umožňuje sledovat nároky na plánovanou údržbu v poměru k výrobnímu času stroje.

$$PDT\ [\%] = \frac{\text{Doba trvání plánovaných zastávek}}{\text{Celková doba trvání výroby}} * 100 \tag{4}$$

Ukazatel PDT je pro tým údržby důležitý z důvodu schopnosti správně nastavit plánované intervaly údržby. Zvyšováním hodnoty tohoto ukazatele je možné dosáhnout lepších výsledků OEE, avšak jen do určité míry. Při dalším zvyšování začne ukazatel OEE klesat. Cíl pro tento ukazatel je tedy nutné nastavit správně tak, aby byl přínos měřený buď vyrobeným množstvím produkce nebo finančními ukazateli co nejvyšší.

3.5 Procento neplánovaných zastávek

Anglicky Unplanned Downtime (UPDT). Výpočet tohoto ukazatele je obdobný jako pro PDT, pouze se zaměřuje na neplánované zastávky. Cílem je hodnoty tohoto ukazatele co nejvíce snížit postupným odstraňováním neplánovaných zastávek či poruch. Toho je možné dosáhnout právě přechodem z korektivní údržby na preventivní, popřípadě prediktivní, a zároveň neustálým vývojem v oblasti nastavování a modifikace zařízení. Tento ukazatel lze využít jak pro celkový přehled za zařízení nebo skupinu zařízení, tak například pro jednu konkrétní poruchu.

3.6 Další ukazatele

Mezi další podpůrné ukazatele je možné zařadit například počet poruch za den, počet poruch na vyrobené množství, spolehlivost procesu nebo například procento neshodné produkce a další.

Výběr ukazatelů pro monitorování stavu údržby je důležitým krokem pro zajištění udržitelnosti programu údržby, ať se jedná o preventivní nebo prediktivní údržbu. Správný výběr a sledování ukazatelů umožňuje lepší zaměření na kořenové příčiny ztrát či poruch. Dalším pozitivem správně nastavených a používaných ukazatelů může být včasné zastavení projektů, které nepřinášejí požadovaný efekt, a které by v delším časovém horizontu mohly znamenat finanční ztrátu.

4 Datová analýza

Datová analýza je jedním ze základů pro zavedení prediktivní údržby. Čtvrtá průmyslová revoluce, tedy koncept Průmysl 4.0, s sebou přináší změnu přístupu k celému výrobnímu procesu. Tato změna spočívá například i ve změně strategie v oblasti údržby, a to například odklonem od rozhodování na základě zkušenosti k rozhodnutím vedeným daty. Výrobní linky je možné osadit množstvím chytrých senzorů a snímačů a data z nich využít pro modelování výrobního procesu a pro podrobnější vhled do slabých míst výrobní linky. Tato data je dále možné využít pro program preventivní i prediktivní údržby. Velké množství on-line i off-line sbíraných dat skýtá obrovskou příležitost pro zvýšení produktivity, snížení množství poruch, optimalizaci plánu údržby i výroby nebo lepší využití lidských zdrojů. Zároveň však přináší i výzvy, které je nutné překonat pro možnost plné výtěžnosti zmíněných benefitů. Tyto výzvy zahrnují například ukládání těchto dat, jejich interpretaci, ale především jejich analýzu [10].

Rapidní nárust objemu ukládaných dat s sebou přináší i potřebu automatizovat proces jejich analýzy a interpretace. Pro extrakci informací ze surových dat jsou využívány statistické metody, ale tento proces je časově náročný a předpokládá hluboké matematické znalosti. Pro automatizaci těchto procesů byly vyvinuty metody strojového učení, které je možné k zpracování dat využít, a to především k predikci budoucích událostí [11]. Toto tvrzení nedeklaruje náhradu statistické analýzy strojovým

učením, ale spíše uvažuje strojové učení jako nadstavbu statistické analýzy. Oba přístupy, jak statistická analýza, tak strojové učení, mají své místo v oblasti prediktivní údržby a navzájem se doplňují.

4.1 Nástroje pro analýzu dat

Existuje široká škála nástrojů umožňujících datovou analýzu, od sešitu v aplikaci MS Excel, přes nástroje, jakými jsou *Python*, *R* nebo *Matlab*, až po komerční řešení zaměřená již na konkrétní aplikace, jako je SAP Analytics. Dále je možné využít benefity platforem, jakými jsou *Azure Machine Learning* a její obdoby, například řešení *AWS* od společnosti Amazon.

Microsoft Excel je nejpoužívanější tabulkový procesor, který lze využít jak pro datovou analýzu, tak pro následnou vizualizaci dat. Umožňuje připojit širokou škálu datových zdrojů od CSV souborů přes SQL servery až po datové kostky. Doplněk Power Pivot umožňuje načtení většího množství dat do datového modelu a data dále upravovat pomocí jazyka DAX [12]. Tento jazyk mimo jiné obsahuje Time-Intelligence funkce vhodné pro práci s daty obsahujícími časové údaje. Dalším užitečným rozšířením je Power Query. Jedná se o technologii datového připojení umožňující kombinovat zdroje dat a následnou modifikaci těchto dat pomocí jazyka M [13].

Python je víceúčelový open source programovací jazyk umožňující mimo jiné datovou analýzu. Existuje celá řada knihoven, frameworků a modulů, které usnadňují tvorbu algoritmů strojového učení. Kombinací s některým z frameworků pro web development, jejichž příkladem může být *Flask*, je možné analýzy ze strojového učení převést do webové aplikace a následně toto řešení sdílet s celou organizací. Příkladem knihoven pro strojové učení může být knihovna *Scikit-learn* nebo *TensorFlow* [14].

Software R, se stejnojmenným programovacím jazykem, nabízí programovací prostředí vyvinuté pro účely statistické analýzy a jiného zpracování dat. Jedná se o open source software s velkou uživatelskou základnou, což zajišťuje stejně jako v případě Pythonu neustálý vývoj nových knihoven. Pro R existuje velké množství knihoven vhodných pro využití v oblasti strojového učení. [15]

Azure Machine Learning je cloudová platforma vyvinutá společností Microsoft. Tato platforma usnadňuje tvorbu algoritmů strojového učení pomocí předem připravených řešení. Pomocí těchto řešení je možné vytvořit aplikace pro prediktivní údržbu bez psaní kódu pouze pomocí spojování existujících modulů. Je možné připojit data z široké škály datových zdrojů i mimo produkty Microsoftu. Výpočet probíhá na cloudu s možností vysokého výpočetního výkonu, čímž dokáže zajistit i zpracovávání dat a predikci v reálném čase [16, 17]. Podporovány jsou i výše zmíněné open source jazyky Python a R, a tím pádem i možnost tvorby vlastních řešení. Nejzásadnější výhodou je právě poskytovaný výpočetní výkon, který je nutný pro online analýzy a predikci v reálném čase. Výpočetní

výkon je však pro verzi zdarma omezen, stejně jako datové úložiště. Cena standardní verze se odvíjí od počtu licencí, zakoupené výpočetní kapacity a času na této kapacitě stráveném [18].

4.2 Strojové učení

Anglicky Machine Learning. Jedná se o metodu, jež představuje jedno z odvětví umělé inteligence a zabývá se analýzou dat a její automatizací [19]. Koncept strojového učení je postaven na myšlence možnosti vytvoření algoritmu, který je schopen učit se ze vstupujících dat. Cílem strojového učení je mimo jiné využití existujících dat k vytvoření modelu, který je možné využít pro predikci budoucích událostí [20, s. 142]. Nasazením metod strojového učení v rámci prediktivní údržby je možné dosáhnout lepších výkonů v oblasti údržby i výroby. Předpokladem pro nasazení strojového učení je vysoká kvalita dat, ze kterých jsou podpůrné modely tvořeny.

Strojové učení je možné rozdělit do tří kategorií: řízené učení (Supervised Learning), neřízené učení (Unsupervised Learning) a zpětnovazebné učení (Reinforcement Learning) [21 s. 2].

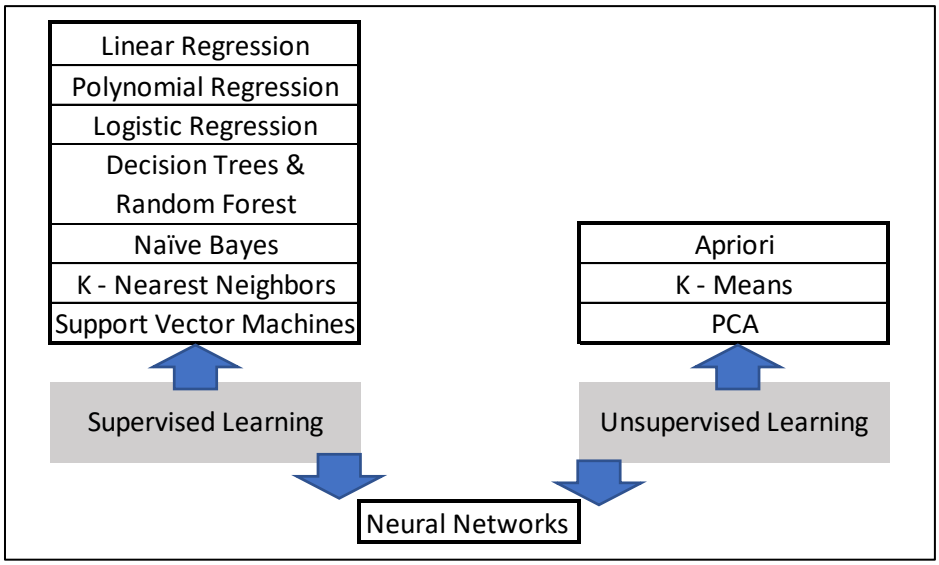
Řízené učení je typ strojového učení s cílem vytvoření modelu z označených historických dat, tedy z dat, u kterých je znám výstup. Trénováním modelu je myšleno řešení pro f v rovnici

$$Y = f(X) \tag{5}$$

kde Y je známý (řízený) výstup a X je známý vstup. Tento typ strojového učení je možné použít jak pro klasifikaci diskrétních úloh, tak pro úlohy spojité, tedy regresní. Podkategorií typu řízeného učení je metoda Ensemble, tedy sestavování. Jedná se o přístup, kdy je použito více algoritmů, jejichž výsledky jsou kombinovány a získaný výsledek je přesnější než výsledky jednotlivých algoritmů [21, s.199].

Neřízené učení je typ strojového učení, který se využívá pro sdružování nebo shlukování (associating nebo clustering). Rozdílem proti učení řízenému je neznalost výstupní proměnné. Data pro trénování modelu nejsou označena nebo mají neznámou strukturu. Cílem tohoto typu strojového učení je extrahovat obecná pravidla nebo strukturu datového modelu a rozdělit data do skupin podle podobnostních ukazatelů [20, s. 142]. Další oblastí pro použití tohoto typu strojového učení je redukce proměnných, což umožňuje zredukovat obsah datového modelu na úroveň obsahující pouze užitečné proměnné, a tím pádem zajistí snížení požadavků na výpočetní výkon a velikost použitých dat. Tento postup spadá do kategorie redukce dimenze a je podrobněji popsán v [21, s. 311].

Dalším typem strojového učení je zpětnovazebné učení. Tento typ se zaměřuje na postupné učení. Proces učení je podobný typu řízeného učení, ale série vstupů a výstupů jsou doplňovány proměnnou značící úspěch dané kombinace vstupů a výstupů. Při další učící iteraci je kombinace vstupních proměnných upravena v návaznosti na úspěch předchozích iterací [22].

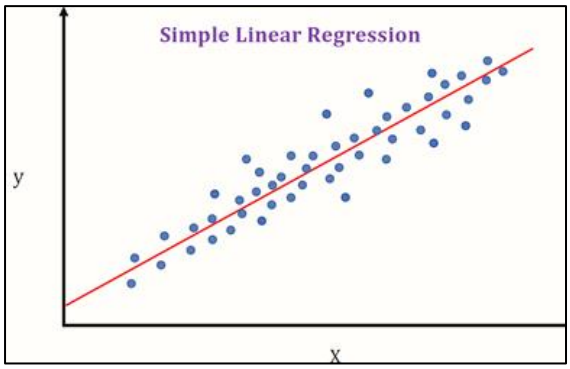


Obrázek 2: Seznam základních algoritmů strojového učení a jejich rozdělení.

Na obrázku 2 je seznam algoritmů strojového učení včetně jejich základního rozdělení. Obsáhlejší seznamy je možné dohledat ve zdrojích [22, 23, 24, 25].

4.2.1 Lineární regrese

Model lineární regrese popisuje vztah mezi závislou proměnou y a jednou nebo více nezávislými proměnnými x, které představují vstup. Cílem lineární regrese je extrakce vztahu mezi těmito proměnnými. Střední kvadratická chyba (MSE – mean squared error) je nejčastěji využívána jako účelová funkce (loss function). Model je trénován minimalizací účelové funkce za využití analytické metody nejmenších čtverců, nebo častěji numericky pomocí numerické optimalizace založené na gradientu [26]. Postup psaný v Python kódu je možné dohledat v [20, s. 93]. Dva možné problémy, které mohou nastat při využití gradientní metody, jsou nalezení lokálního, nikoli globálního, minima a zastavení iterací v rovinné oblasti funkce. Existují i další řešení, která se snaží tyto problémy eliminovat, jsou však náročnější na výpočetní výkon a vhodnějším přístupem může být opakovaný výpočet s různými počátečními podmínkami. Pro výpočet modelu obsahující velké množství dat je vhodnější využít metodu stochastického gradientního sestupu [27].



Obrázek 3: Lineární regrese, Zdroj: [28]

Popisovaný model lineární regrese je citlivý na výskyt anomálií v datech. Jedním ze způsobů, jak tento problém řešit, je očištění dat od těchto bodů. Jiným řešením může být použití regularizačních modelů, jako například *Ridge regression* nebo *LASSO regression* [29, s. 13], [21, s. 278].

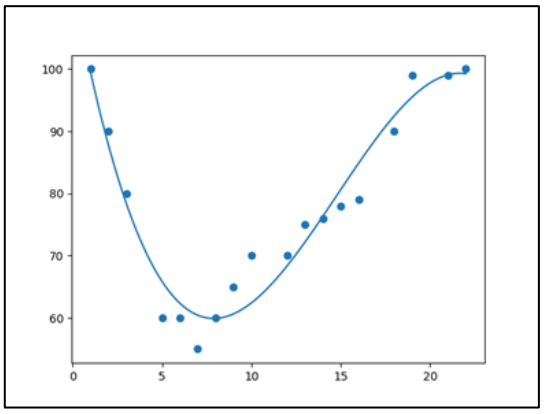
Pole využití lineární regrese je široké. Jedním z příkladů může být predikce výkonu výrobní linky nebo jejích částí v závislosti na znalosti plánovaných zastávek a parametrů výroby, kterými mohou být použité materiály pro výrobu finálního produktu a nastavení stroje. Příklad využití lineární regrese pro plánování prediktivní údržby v oblasti výroby polovodičů je uveden v článku [30].

4.2.2 Polynomická regrese

Polynomická či polynomiální regrese umožňuje modelování nelineárních vztahů mezi nezávislými a závislými proměnnými (vstupy a výstupy). Polynomická regrese je považována za speciální případ regrese lineární, a to vzhledem k linearitě koeficientů *b*.

$$y = a_0 + b_1 x + b_2 x^2 \dots \dots b_n x^n \tag{6}$$

Vzhledem k linearitě koeficientů *b* může být využit stejný výpočetní postup jako pro lineární regresi popsaný v bodě 4.2. Střední kvadratickou chybu je možné využít pro zavedení účelové funkce, která je optimalizována pomocí gradientní metody [21, s. 298].

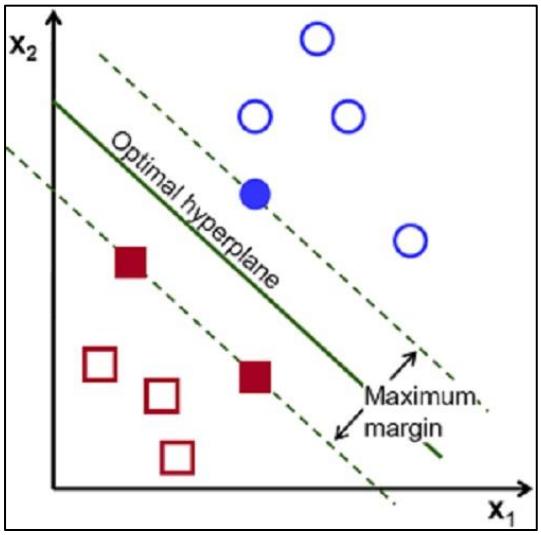


Obrázek 4: Polynomická regrese. Zdroj: [31]

Polynomická regrese je využívána v případech, kdy lineární regrese nedosahuje požadované přesnosti z důvodu její linearity a kde není možné data pro analýzu linearizovat. Příklad využití polynomické regrese a porovnání přesnosti predikce s dalšími metodami je možné nalézt v článku zabývajícím se predikcí poruch v potrubí vodovodní distribuční sítě [32].

4.2.3 Support Vector Machine – SVM

Support Vector Machines, neboli metoda podpůrných vektorů, je typ strojového učení spadající do kategorie řízeného učení a je možné jej využít jak pro klasifikační úlohy, tak pro regresní. Hlavním cílem této metody je najít optimální nadrovinu (hyperplane), která rozdělí datové body do tříd. Body, které leží nejblíže u rozdělující nadroviny, jsou nazvány podpůrné vektory (support vectors), z nichž je odvozen název této metody. Cílem algoritmu je najít takovou nadrovinu, jejíž vzdálenost od podpůrných vektorů je co největší, viz obrázek 5. Jádrová transformace (kernel transformation) umožňuje převod lineárně neseparovatelné úlohy na úlohu lineárně separovatelnou, a to transformací do vyšší dimenze [33, s. 144].



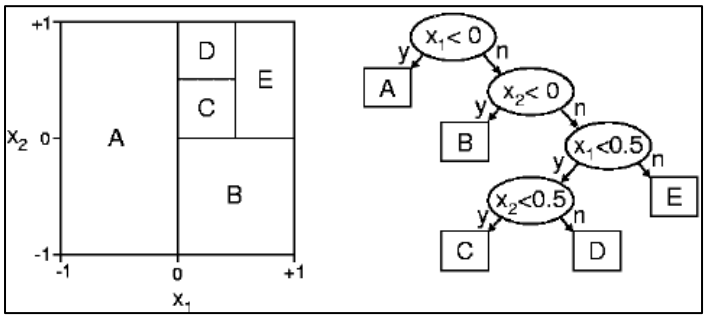
Obrázek 5: SVM optimalizace. Zdroj: [34]

Příklad a evaluaci využití SVM algoritmu pro predikci času mezi zastávkami (MTBF) je možné nalézt v článku [35]. Ze závěru tohoto výzkumu vyplývá silný potenciál pro využití v oblasti prediktivní údržby. Dalším z příkladů využití SVM v oblasti podpory výrobních ukazatelů může být klasifikace budoucí výroby do kategorií podle výskytu neshodných produktů. Na základě historických dat obsahujících výstupy neshodných produktů společně s parametry produkce může být trénován model schopný identifikovat potenciální budoucí riziko vyšší úrovně neshodných materiálů.

4.2.4 Rozhodovací strom

Anglicky Decision Tree. Algoritmy rozhodovacích stromů umožňují modelování nelineárních úloh. Tato metoda spadá do kategorie řízeného učení a lze ji využít jak pro regresní, tak pro klasifikační problémy. Příklady algoritmů pracujících na principu rozhodovacího stromu jsou například: ID3, C4.5 a CART stromy [33, s. 179].

Během procesu učení jsou tréninková data postupně rozdělována do po sobě jdoucích podmnožin. Rozdělování začíná v kořenovém uzlu a v každém dalším uzlu jsou data rozdělena do konečného počtu skupin. Každý z těchto uzlů představuje podmínku, na jejímž základě je postaveno rozdělení dat. Na sérii uzlů navazuje list stromu, v němž jsou již data přiřazena konkrétní skupině, tedy jsou klasifikována. Volba vhodných rozhodovacích podmínek v uzlech stromu je postavena na informačním zisku porovnáváním entropií [21, s. 305].



Obrázek 6: Rozhodovací strom.

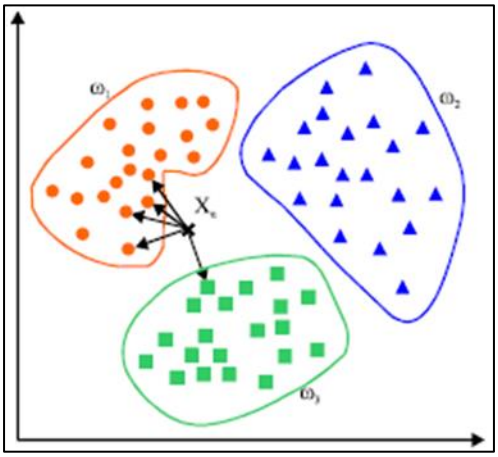
Příklad rozhodovacího stromu je naznačen na obrázku 6. Pomocí čtyř uzlů jsou data rozdělena do pěti listů A až E. Jedním z benefitů stromových algoritmů je jejich snadná vizualizace a tím pádem i možná interpretace rozhodovacího procesu. Je tedy možné určit relativní důležitost jednotlivých parametrů a nedůležité parametry odebrat pro zvýšení výkonu a snížení možnosti přeučení modelu. Další výhodou těchto algoritmů je jejich schopnost pracovat s neúplnými sadami dat [36]. Použitím Ensemble učící metody, popsané v kapitole 4.1, vzniká algoritmus nazvaný Náhodný les (Random Forest). Náhodný les je zkonstruován z více rozhodovacích stromů, které jsou trénovány odděleně a trénovací data jsou mezi ně náhodně rozdělena pro zajištění jejich diverzity. Finální rozhodnutí je založeno na hlasování jednotlivých rozhodovacích stromů. Příklad algoritmu pro Náhodný les a jeho detailní popis je možné dohledat v [29, s. 18].

Studie v [37] se zaměřuje na využití algoritmů rozhodovacích stromů a náhodného lesa pro plánování údržby železničních výhybek. Studie se mimo jiné zaměřuje na interpretaci výstupu modelů, a to například pomocí relativní důležitosti jednotlivých charakteristik. Dalším příkladem použití může být predikce krátkých neplánovaných zastávek na výrobní lince. Tyto krátké zastávky mohou být

poměrně časté a jejich četnost se liší na základě různých charakteristik od základního nastavení výrobní linky, přes charakteristiky produktu, až po intervaly čištění. Predikce vyšší četnosti určité krátké zastávky by umožnila zásah údržby v předstihu a tím pádem snížení jejího dopadu na celkový výkon výrobní linky.

4.2.5 K-Nearest Neighbors – K-NN

Tento algoritmus strojového učení patří do kategorie řízeného učení a může být použit jak pro klasifikační, tak regresní problémy. Písmeno K v názvu značí počet nejbližších sousedů, kterých je využito pro klasifikaci nového datového bodu. Pro klasifikaci nového bodu jsou určeny vzdálenosti všech bodů od nově klasifikovaného bodu. Podle příslušnosti k-nejbližších sousedů je nový bod klasifikován. Namísto počtu k-bodů může být klasifikace provedena na základě předem určeného rádiusu, kdy je nový bod klasifikován na základě tříd bodů v daném rádiusu. Pokud je k-NN algoritmus využit pro regresi, pak je výstup určen na základě průměru hodnot k-nejbližších sousedů. V určitých případech je vhodné do modelu zavést váhy, kdy body ležící blíže klasifikovanému bodu mají pro určení třídy vyšší vliv než body více vzdálené [21].



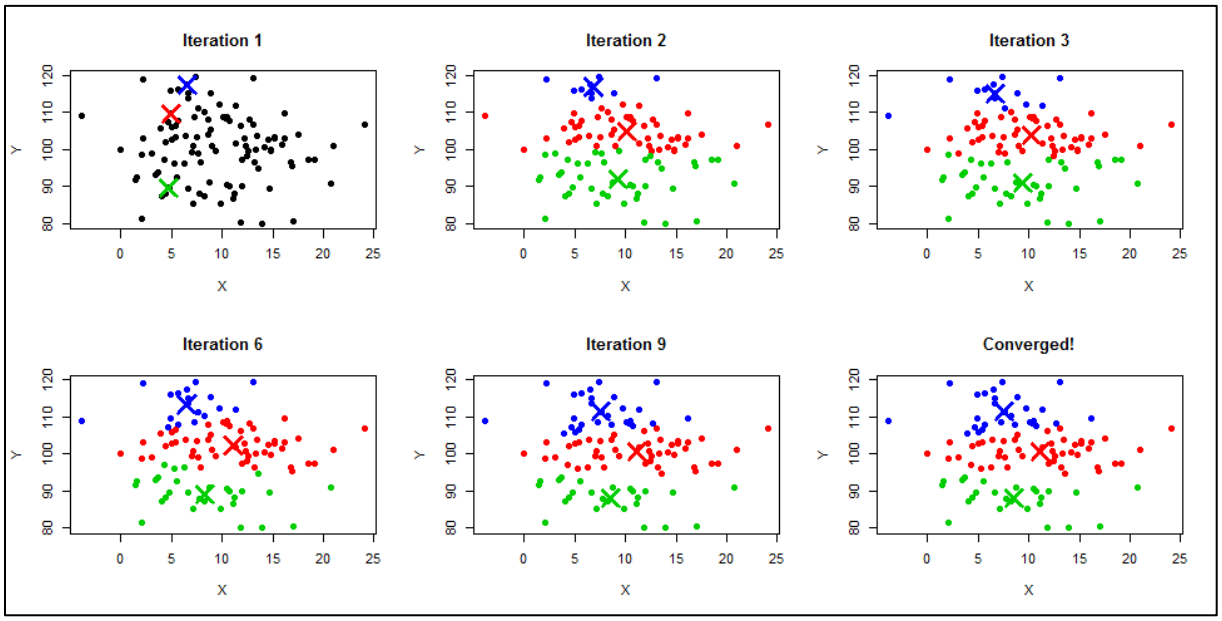
Obrázek 7: Určení nejbližších sousedů k-NN algoritmu. Zdroj: [38]

Využití k-NN algoritmu pro prediktivní údržbu ložisek je popsáno v článku [39]. Algoritmus k-NN je zde porovnán s výše zmíněnými algoritmy SVM, rozhodovací strom nebo náhodný les. Ze závěrů této studie vyplývá silný potenciál k-NN algoritmu pro využití v oblasti prediktivní údržby.

4.2.6 K-Means

Tento algoritmus spadá do kategorie neřízeného učení a je využíván ke shlukové analýze. Úkolem tohoto algoritmu je roztřídit data do klastrů podle podobnosti. Písmeno K v názvu určuje počet klastrů, do kterých algoritmus data roztřídí. Tento parametr je nutné určit manuálně. Některé z metod vhodných pro určení parametru K, jako je například metoda nazvaná elbowing, je možné dohledat

v článku [40]. Algoritmus K-průměrů shlukuje datové body do klastrů iterativním postupem. Prvním krokem je náhodné umístění k-centroidů a na základě vzdálenosti (například Euklidovské) jsou datové body přiřazeny k nejblíže položeným centroidům. V dalším kroku jsou centroidy přesunuty do středu datových bodů, které byly v prvním kroku k centroidům přiřazeny. Po této změně polohy centroidů jsou datové body znovu přiřazeny k centroidům na základě vzdálenosti a celý proces se iterativně opakuje [21, s. 318]. Postup pro řazení bodů do klastrů je naznačen na obrázku 8.



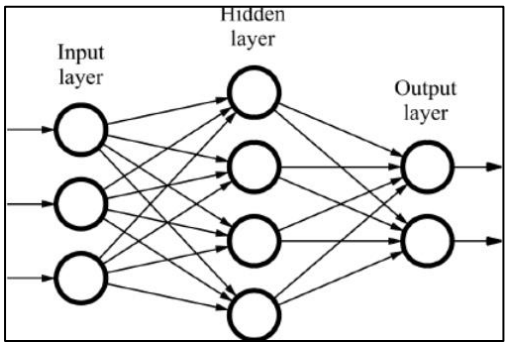
Obrázek 8: Iterace metody k-průměrů. Zdroj: [41]

Využití tohoto algoritmu pro plánování preventivní údržby je uvedeno v článku [42]. Algoritmus k-průměrů je zde využit pro zařazení strojů do klastrů na základě frekvence jejich údržby a jejich polohy. Údržba je následně plánována po těchto klastrech, čímž je minimalizován čas a náklady s údržbou spojené. Algoritmus k-průměrů je dále možné využít v kombinaci s některým z klasifikačních algoritmů, kdy metoda k-průměrů rozřadí data do klastrů a ty jsou následně využity pro predikci některým z klasifikačních algoritmů. Případné další využití je možné nalézt v určení parametrů s největším vlivem, popřípadě parametrů, které výstup neovlivňují a je možné je z datového setu odebrat pro snížení nároků na výpočetní výkon.

4.2.7 Neuronové sítě

Algoritmy postavené na principu neuronových sítí lze zařadit do kategorií řízeného i neřízeného strojového učení. Neuronové sítě jsou inspirovány fungováním lidského mozku. Jedná se tedy o simulaci fungování neuronu pomocí zjednodušeného matematického modelu. Existuje mnoho typů neuronových sítí lišících se topologií neuronů i směrem pohybu signálu mezi neurony. Zde je princip neuronových sítích popsán na dopředné neuronové síti. Neuronové sítě jsou sestaveny z vrstev

neuronů, jak je naznačeno na obrázku 9. Vstupní vrstva obsahuje tolik neuronů, kolik je vstupováno proměnných. Počet neuronů ve výstupní vrstvě se rovná počtu výstupních proměnných. Skryté neurony mohou být rozděleny do jedné vrstvy, jak je tomu na obrázku 9, nebo do více vrstev. Signály v podobě reálných čísel se neuronovou sítí šíří zleva doprava. Tyto signály jsou v neuronu procesovány aktivační funkcí a poslány do následujících neuronů. Mezi neurony jsou signály upravovány vahami a hodnotami nazývanými bias pro dosažení požadovaného výstupu. Aktivační funkce jsou voleny při tvorbě neuronové sítě a během procesu učení zůstávají neměnné. Během procesu učení se přepočítávají pouze váhy a bias. Jednou z metod pro trénování neuronové sítě je metoda zpětného šíření (anglicky back propagation). Na počátku jsou pro váhy a bias zvoleny náhodné hodnoty. Do neuronové sítě jsou vstupovány vstupní proměnné a výstup je porovnán s požadovanými hodnotami. Chyba mezi získaným a požadovaným výstupem je popsána účelovou funkcí, která je minimalizována gradientní metodou s ohledem na všechny váhy a bias s využitím řetězového pravidla (anglicky chain-rule). Náhodně zvolené váhy a bias jsou přepočítány a celý proces se iterativně opakuje až do dosažení ukončujících podmínek [20, s.213], [43].



Obrázek 9: Struktura neuronové sítě. Zdroj: [44]

Využití různých druhů neuronových sítí pro prediktivní údržbu je popsáno a výsledky jsou zhodnoceny v článku [45]. Závěry tohoto článku poukazují na silný potenciál neuronových sítí v oblasti prediktivní údržby, avšak při splnění určitých podmínek, které mohou být finančně náročné a je tedy nutné před nasazením neuronových sítí zhodnotit jejich případnou finanční návratnost.

4.3 Strojové učení pro prediktivní údržbu

Metody strojového učení postupně nachází cestu do oblasti prediktivní údržby, což dokládají články [10, 46], které poskytují systematický přehled literatury zabývající se tímto tématem. Z těchto článků vyplývá rapidní nárust počtu studií zabývajících se strojovým učením v oblasti prediktivní údržby obecně a prediktivní údržby výrobních linek od roku 2015 do současnosti. Nejčastějšími oblastmi pro nasazení strojového učení jsou předpověď poruchy, predikce zbývající životnosti zařízení a predikce výskytu neshodných produktů. Nejčastěji využívaným algoritmem jsou základní neuronové sítě

popsané v kapitole 4.2.7 a dále metody SVM, k-NN a Náhodný les. Nasazení řízeného učení je v porovnání s učením neřízeným častější. Z metod neřízeného učení je nejčastěji využíván algoritmus k-průměrů popsaný v kapitole 4.2.6.

5 Systémy pro ukládání dat

Jak již bylo zmíněno v kapitole 4, jednou z velkých výzev v oblasti implementace prediktivní údržby je ukládání dat. Chytré senzory, IoT a další přínosy Průmyslu 4.0 generují obrovské množství dat, která jsou nepostradatelná pro systémy prediktivní údržby. Při výběru datového úložiště pro systém prediktivní údržby je nutné uvažovat i fakt, že objem ukládaných dat v budoucnu stále poroste a požadavky na prediktivní údržbu se budou měnit a rozšiřovat. Kapacita úložiště není jediným faktorem, je nutné se zaměřit i na možnosti a obtížnost údržby datového úložiště, na výkon, finanční náklady a škálovatelnost. V neposlední řadě je nutné zaměřit se na zabezpečení dat, jelikož se často může jednat o citlivá data nebo o data strategická, která by v případě úniku mohla společnosti uškodit v konkurenčním prostředí [1, s. 343], [47]. Výběr správného úložiště dat je tedy jednou z hlavních podmínek pro úspěšnou, a hlavně dlouhodobě udržitelnou implementaci prediktivní údržby.

5.1 Ukládání do souborů

Jednou z možností, jak ukládat data, která budou využita pro prediktivní údržbu, je využití možnosti ukládání souborů s daty buď přímo na osobním počítači, nebo na serveru, na kterém budou zpracovávána. Dále mohou být tyto soubory uloženy na sdíleném disku, popřípadě na cloudovém úložišti. Soubory je možné rozdělit na soubory s plochou strukturou a na soubory s vnitřní hierarchií [48]. Příkladem souborů s plochou strukturou jsou například soubory s koncovkou .txt nebo .csv. Příklady souborů s vnitřní hierarchií jsou XML nebo JSON soubory.

Toto řešení může být vhodné pro menší projekty spojené s prediktivní údržbou, popřípadě pro první implementaci a otestování projektu. Data mohou být ukládána do složek ve formě souborů z různých zdrojů, tento proces může být automatizován a uživatel se procesu sběru dat nemusí dále věnovat. Aplikace podporující prediktivní údržbu vytvořená například v Pythonu s těmito soubory může pracovat a výsledky ukládat do dalších souborů, popřípadě upravovat tyto soubory. Výhodou tohoto řešení je jeho jednoduchost a nízké náklady na zavedení a údržbu. U větších projektů a v případě, kdy jsou předpokládány další požadavky, mohou nastávat následující problémy:

V případě požadavku na rozšíření aplikace o další funkce mohou být uložená data nedostatečná a bude nutné přidání dalších souborů obsahujících část stejných dat. Data jsou do souborů ukládána a strukturována tak, aby splňovala požadavky pro konkrétní aplikaci. V případě nového požadavku je

data nutné znovu uložit s novou strukturou. Tím vzniká duplikace dat a větší nároky na úložnou kapacitu. To může vést i k nekonzistentnosti dat.

Jak již bylo zmíněno, ukládání dat do souborů je možné automatizovat. Toto řešení však na rozdíl například od databázového řešení nemá dostatečné možnosti v oblasti validace dat, což může vést k chybným záznamům nebo i k chybějícím datům. Tato skutečnost může ohrozit fungování celé aplikace, popřípadě vést k nesprávným predikcím.

Zabezpečení takového řešení může představovat další problém. Přístup k datům by měl být umožněn jen oprávněným osobám a dále by měl být upraven do kategorií jako je čtení, zápis nebo úprava. Toho je sice možné dosáhnout, avšak udržovat takové řešení může být časově náročné, a to především v případě vyššího počtu uživatelů.

Při sestavování aplikací z takto ukládaných dat je vyžadována podrobná znalost struktury dat, což přináší problém v případě nových zaměstnanců, ale i při udržování takového způsobu ukládání dat. Z tohoto důvodu je pro komplexnější řešení vhodné zvolit jiný způsob, jakým může být například databázový systém.

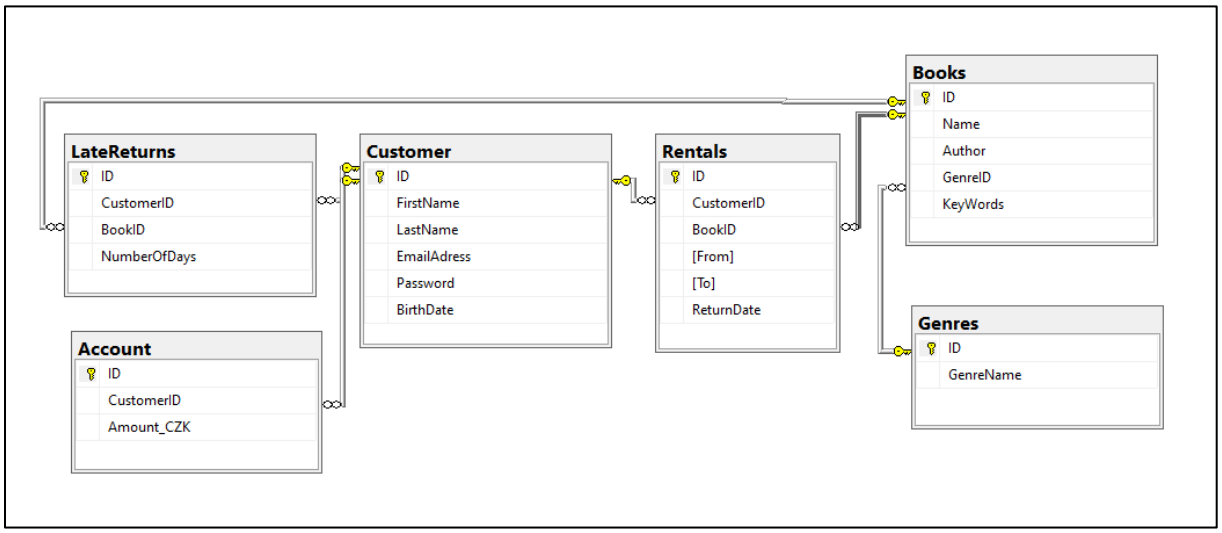
5.2 Databázové systémy

Databázový systém je celek obsahující data a systém pro správu databáze (DBMS – database mamagement system). Databázových systémů existuje několik typů, například: Hierarchická databáze, Síťová databáze, Objektová databáze a nejpoužívanější Relační databáze [49]. DBMS je software, který umožňuje práci s daty, jako je ukládaní dat, čtení dat či jiná manipulace s daty. DBMS poskytuje mnoho dalších funkcí od zabezpečení dat, přes zajištění konzistence dat, až po automatizaci procesů v databázi. Příklady populárních DBMS softwarů jsou MySQL, Oracle nebo Microsoft SQL server.

5.2.1 Relační databáze

Relační databáze využívá relační datový model, který je nejpoužívanějším mezi databázovými systémy [50, s. 8]. Je tvořena sadou tabulek a vztahy, tedy relacemi mezi nimi. Vztahy mezi tabulkami nebo jednotlivými záznamy jsou zajištěny pomocí klíčů, které jsou rozděleny na klíče primární a cizí. Sloupce obsahují atributy a jejich datové typy jsou pevně definovány pro zajištění konzistence dat. V řádcích jsou uchovávány informace k daným atributům. Informace uložené v různých tabulkách je možné propojit na základě kombinace primárního a cizího klíče a tím získat informace ze dvou či více tabulek najednou. Tento postup ukládání dat umožňuje získání lepšího přehledu o vztazích mezi daty. Zároveň umožňuje ukládat data bez jejich duplikace a tím snížit nároky na kapacitu úložiště. Pro práci s daty v relačních databází se využívá jazyka SQL (Structured Query Language). Tento jazyk slouží nejen

pro čtení databáze, ale zároveň i pro ukládání dat a další manipulaci s nimi. Pomocí tohoto jazyka je možná i tvorba nových tabulek nebo úprava stávajících. DBMS obsahuje funkce, které usnadňují tvorbu aplikací například ukládáním dotazovacích procedur, které je možné replikovat ve více aplikacích [51]. Na obrázku 10 je znázorněna možná struktura tabulek v relační databázi, včetně vztahů určených klíči.



Obrázek 10: Příklad organizace relační databáze.

Pro zajištění konzistentnosti dat pracují relační databáze na principu transakcí. Každý úkon v databázi prováděný je transakcí, která může být buď zcela dokončena, nebo nesmí mít žádný efekt. Příkladem může být situace přesunu materiálu mezi sklady, kdy je z jedné tabulky (představující první sklad) přesouván materiál do druhé tabulky (představující druhý skald). V případě chyby během transakce k žádné změně nedojde a tím pádem nemůže nastat situace, kdy se z první tabulky materiál odečte, ale do druhé tabulky se již nepřičte. Princip transakčního řízení je blíže popsán v [50] na straně 20. Další výhodou relačních databází je jejich snadné zálohování a zrcadlení (mirroring), což zajišťuje stabilitu celého řešení. Poslední zmíněnou výhodou je zabezpečení a snadné přiřazování přístupů k jednotlivým částem databáze [52].

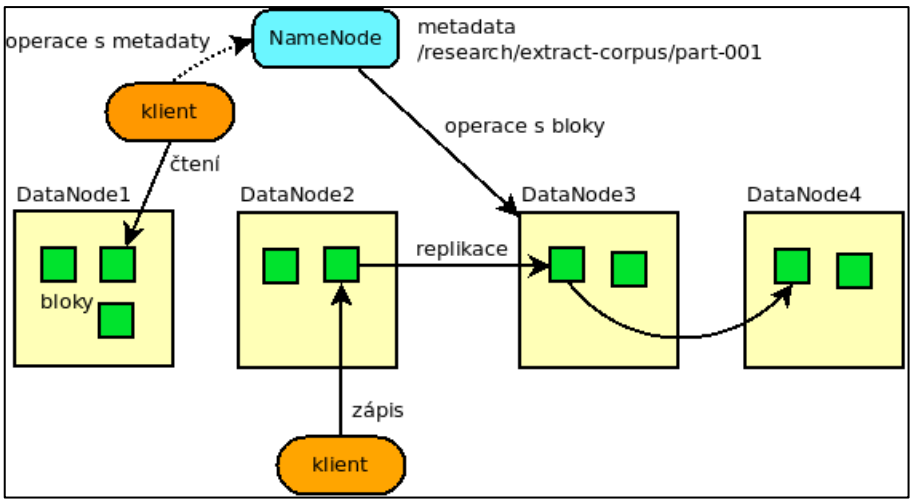
Databázové systémy, ať už se jedná o relační databáze nebo jiný typ, jsou pro ukládání dat pro prediktivní údržbu lepší variantou než ukládání do souborů popsané v kapitole 5.1. Jejich použití je ale nutné dále zhodnotit na základě struktury ukládaných dat. Relační databáze nejsou vhodné pro ukládání nestrukturovaných dat. Dále pak na parametrech, kterými jsou objem dat a rychlost ukládání a čtení dat [50, s. 468]. Pro aplikace, které pracují s množstvím dat, které by běžné relační databáze nezvládaly nebo by celou aplikaci zpomalovaly, popřípadě pro aplikace, které využívá velké množství uživatelů, jsou vyvinuta řešení umožňující ukládat data po částech do databázových uzlů. Příklady takových řešení jsou Systém distribuovaných souborů (Distributed File System), Horizontální dělení dat

(Sharding), Key-value databáze (Key Value Storage Systems) a Paralelní a distribuované databáze (Parallel and Distributed Databases) [50, s. 472].

5.3 Systém distribuovaných souborů

Systém distribuovaných souborů ukládá data na různé počítače a využívá jejich výkonu při výpočtech. Data jsou rozdělena do bloků a tyto bloky jsou uloženy na více počítačů, čímž vzniká duplicita dat, ale zároveň je zajištěna jejich dostupnost při poruše jednoho z počítačů. Takto distribuovaný systém umožňuje ukládání velkých množství dat a zároveň umožňuje souběžný přístup mnoha klientům. Zpracování dat probíhá paralelně na více uzlech, v důsledku čehož je možné zpracovávat větší množství dat souběžně. Hlavní výhodou jsou však nižší finanční náklady na stejné množství uložených dat v porovnání s relačními databázemi. Dále na rozdíl od relačních databází umožňuje ukládání nestrukturovaných dat [50, s. 474]. Příklady systému distribuovaných systémů jsou Google File Systém (GFS) a na jeho základech postavený Hadoop File Systém (HDFS). Uvedené výhody jsou postaveny na fungování HDFS společně s dalšími jeho moduly.

Nevýhodou HDFS je v první řadě nižší počet uživatelů, což může působit komplikace v případě přijímání nových zaměstnanců, kteří by tento systém měli spravovat. Jako nevýhoda může být spatřována i skutečnost, že se jedná o open source řešení s nižší podporou v porovnání s komerčními relačními databázovými systémy. Zároveň implementace a správa takového open source systému může být nákladná a neudržitelná. Tento problém řeší komerční projekty, jakými jsou například Amazon Elastic MapReduce, IBM Open Platform nebo jiná řešení například od Microsoftu nebo různých startupů [53]. Tato komerční řešení odstraňují i problémy týkající se zabezpečení dat a autorizace přístupu.



Obrázek 11: Architektura Hadoop file systému. Zdroj: [54]

Na obrázku 11 je znázorněna architektura HDFS a naznačeny jednotlivé operace jako čtení, zápis a replikace dat. HDFS funguje na principu master a slave, kde master-NameNode udržuje seznam souborů a ty poskytuje klientovi, který poté komunikuje již s určitým datovým uzlem (slave). Replikace probíhá automaticky po zápisu, a to na základě předem definovaných parametrů.

5.4 Horizontální dělení dat

Anglicky Sharding. Horizontální dělení dat je přístup, při kterém jsou data rozdělena do několika databází. Příkladem může být ukládání dat z výrobních linek, kdy by bylo dosaženo maximální kapacity úložiště. V tomto případě je možné ukládat data z poloviny výrobních linek do jedné databáze a data druhé poloviny linek do databáze druhé. Struktura by zůstala stejná, jen je potřeba přidat atribut, na jehož základě je možné určit, v jaké databázi se daná výrobní linka nachází. S tím je nutné pracovat v aplikacích, které k databázím přistupují. V aplikacích je nutné sledovat, v jaké databázi se stroj nachází a dotazovat tu správnou. Již zmíněnou výhodou je možnost ukládání většího množství dat. Další výhodou je prohledávání menšího objemu dat při dotazování konkrétní databáze a v důsledku toho i rychlejší zpracování dat. V případě výpadku databáze je ovlivněna jen část dat. Zároveň je možné dotazy do databází provádět paralelně, což umožňuje rychlejší zpracování dat. Nevýhodou takového řešení je komplikovanější úprava dat. Není možné manipulovat data přes více shardů jedním dotazem [50, s. 475]. Horizontálně dělené databáze je možné vytvářet i na cloudových platformách, jakými jsou například Microsoft Azure nebo na datových centrech od Amazonu. Finanční náklady na takovéto řešení mohou být nižší než při škálování vertikálním, avšak až od určitého objemu dat. Mimo finančních nákladů spojených s nákupem zařízení je nutné uvažovat i finanční náklady spojené s údržbou těchto zařízení (netýká se jen cloudových řešení), ale také s komplikacemi spojenými s tvorbou aplikací přes více databází.

5.5 NoSQL databáze

Key-value databáze zmíněné v kapitole 5.2.1 jsou jedním z dalších přístupů pro ukládání velkého množství dat. Jiný název pro Key-value databáze je NoSQL databáze, kde No může značit skutečnost, že pro ně není využíván jazyk SQL, nebo v novějším podání (Not only – Ne pouze) značící fakt, že je možné využít i jiný jazyk pro dotazování. V současné době existují NoSQL databáze, u kterých je možné jazyk SQL využít. Tento vývoj byl způsoben obecným požadavkem vyplývajícím ze standardu jazyka SQL v oblasti databázových systémů. Systém distribuovaných souborů popsaný v kapitole 5.3 je vhodný pro ukládání větších záznamů. V případě potřeby ukládat velké množství relativně malých záznamů (v řádech kilobajtů) není využití systému distribuovaných souborů vhodné. Pro takovýto účel je možné využít právě NoSQL databáze. Data jsou ukládána s klíčem, na jehož základě jsou data dohledatelná. Paralelní NoSQL databáze umožňují ukládání dat přes více počítačů, což zajišťuje

replikaci dat při zajištění jejich konzistentnosti. Zátěž mezi jednotlivými počítači je automaticky balancována [50, s. 476]. Nejznámější NoSQL databází je MongoDB. Dalšími příklady NoSQL databází jsou BigTable od Googlu a jím motivovaná NoSQL databáze Cassandra od Facebooku. Všechny tyto databáze jsou open source. Jako zástupce komerčního řešení může být uvedena Amazon DynamoDB NoSQL databáze. Jednou z nevýhod NoSQL databází může být konzistence dat, která nemusí být zaručena v každém okamžiku, jako tomu je u databází relačních. Proto je nutné zvážit její nasazení podle typu aplikace. Výhodou je naopak schopnost ukládat nestrukturovaná data [55]. Key-value databáze jsou pouze jedním typem NoSQL databází. Další příklady NoSQL databází jsou dokumentové databáze, sloupcové databáze nebo grafové databáze.

Využití NoSQL databází je nutné zvážit na základě konkrétní aplikace. Jsou vhodné pro případy, kdy je nutné ukládat nestrukturovaná data, u kterých je nutné často přidávat nové atributy nebo v případě požadavku na horizontální škálovatelnost.

5.6 Cloudová řešení

Všechna výše zmíněná řešení pro ukládání dat je možné implementovat jak na lokálních počítačích nebo serverech (on-premises), tak i na cloudu. Které z těchto dvou řešení je výhodnější a vhodnější, záleží na konkrétní aplikaci. Následuje seznam bodů, které je nutné uvažovat při rozhodování [56]:

U cloudového řešení není nutná vysoká počáteční investice pro nákup serverů a tím pádem bude méně finančně nákladným řešením v případě tvorby nového řešení datového úložiště. V případě již existující on-premises infrastruktury tomu tak ale nemusí být.

S on-premises řešením jsou spojeny další náklady v podobě údržby serverů a na nich běžících aplikacích, což je v podobě cloudového řešení spojeno s pravidelnými poplatky. Tyto poplatky lze však optimalizovat a platit pouze za využitou kapacitu.

Údržba On-premises řešení vyžaduje kvalifikovaný personál. V případě kritických aplikací podmiňujících chod výrobních linek je nutné zajistit nonstop přítomnost tohoto personálu. U cloudových řešení tento problém alespoň částečně odpadá. Nevýhodou cloudového řešení je možnost výpadku služby. V tomto případě záleží na kvalitě poskytovaného řešení.

Pravděpodobně nejdůležitějším bodem je zabezpečení dat. Data uložená lokálně má společnost plně pod kontrolou a nehrozí žádné ztráty či úniky dat. V případě cloudových řešení jsou data uložena u třetí strany, zabezpečení dat je tedy na straně poskytovatele služby. Záleží tedy na kvalitě daného řešení.

Škálovatelnost úložiště je v případě využití cloudového řešení mnohem flexibilnější a provedení je možné téměř okamžitě. V případě lokálního řešení je nutný nákup nového serveru, jeho zprovoznění a instalace, což je časově i finančně náročné.

Nejlépe hodnocené produkty v oblasti cloudových řešení jsou Microsoft Azure, Amazon Web Services nebo Google Cloud. Při volbě vhodného řešení je nutné uvažovat nejen body uvedené výše, ale i kompatibilitu s jinými využívanými produkty. Příkladem mohou být Business Inteligence produkty pro vizualizaci dat.

6 Vizualizační nástroje

Jak již bylo zmíněno v předchozích kapitolách, v oblasti prediktivní údržby je generováno velké množství dat. Tato data je nutné interpretovat přehledným způsobem, který umožní jejich snadné pochopení a zároveň podpoří rychlé a efektivní rozhodování. Vzhledem k rychlosti, jakou jsou nová data generována, je jedním z požadavků proces interpretace dat plně nebo alespoň částečně automatizovat. K tomu je možné využít celou řadu vizualizačních nástrojů spadajících do kategorie business inteligence (BI) [57]. Nároky na výstup vizualizačních nástrojů nekončí u prezentace dat v podobě statických grafů, ale je vyžadována možnost interakce s daty pro možnost podrobnější analýzy. Uživatel takového nástroje by měl mít možnost s daty dále pracovat, analyzovat je a transformovat. Dalším z nároků na takovéto nástroje je možnost snadné prezentace dat na více úrovních. Odůvodněnost tohoto nároku může být demonstrována na vizualizaci dat z výrobních linek, kde management továrny vyžaduje agregovaný pohled přes všechny výrobní linky, manažer oddělení potřebuje přehled dat po jednotlivých výrobních linkách a tým z oddělení údržby zajímá podrobný přehled rozdělený po částech výrobní linky.

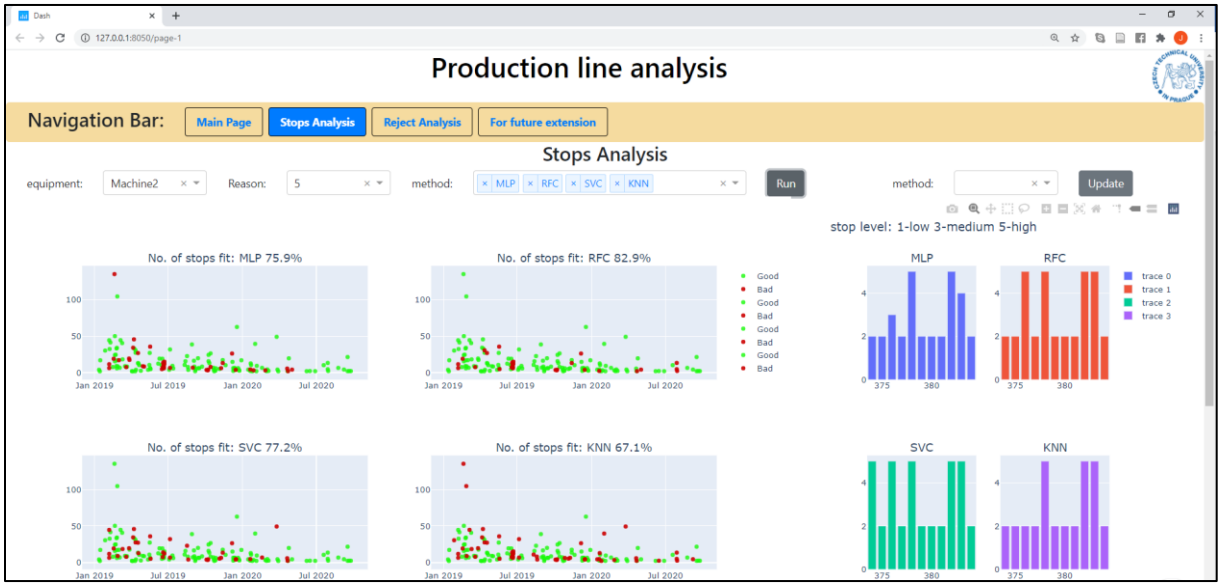
Pro implementaci vizualizačních nástrojů do firemního prostředí je možné využít dvou přístupů. Prvním přístupem je tvorba vlastního softwaru pro vizualizaci dat například využitím programovacího jazyka Python s využitím předpřipravených knihoven a frameworků. Druhým možným přístupem je využití některé z komerčních platforem. Cena za služby těchto platforem je však poměrně vysoká.

6.1 Vizualizace dat v Pythonu

Tvorba vlastního BI řešení v Pythonu je narozdíl od využití komerčních platforem zcela zdarma, avšak pouze z hlediska využitého softwaru. Náklady takového řešení jsou spojené s vývojem samotné aplikace pro vizualizaci dat a dále s její údržbou. Pro vývoj takového řešení je nutná znalost programovacího jazyka. Úprava těchto vizualizací nebo tvorba nových je závislá na konkrétním člověku nebo malé skupině lidí a tím pádem je rozvoj tohoto řešení pomalý. Údržba takových řešení je

v porovnání s komerčními produkty časově náročná a může se objevit i větší množství chyb. Toto řešení může být vhodné pro menší podniky, kde je třeba vizualizovat menší množství dat a finanční náklady na pořízení komerčního řešení převyšují jeho přínos.

Jedním z vhodných řešení pro vývoj webové aplikace pro vizualizaci dat je využití frameworku Flask v kombinaci s knihovnami pro vizualizaci dat, jakými jsou například Matplotlib, Plotly, Seaborn nebo ggplot. Jiným řešením může být využití frameworku Dash, který je postaven na Flasku v kombinaci s knihovnou Plotly [58, 59].



Obrázek 12: Aplikace vytvořená v rámci této diplomové práce za využití frameworku Dash

Takto vytvořená aplikace může být spuštěna na lokálním serveru nebo může být využito cloudového hostingu.

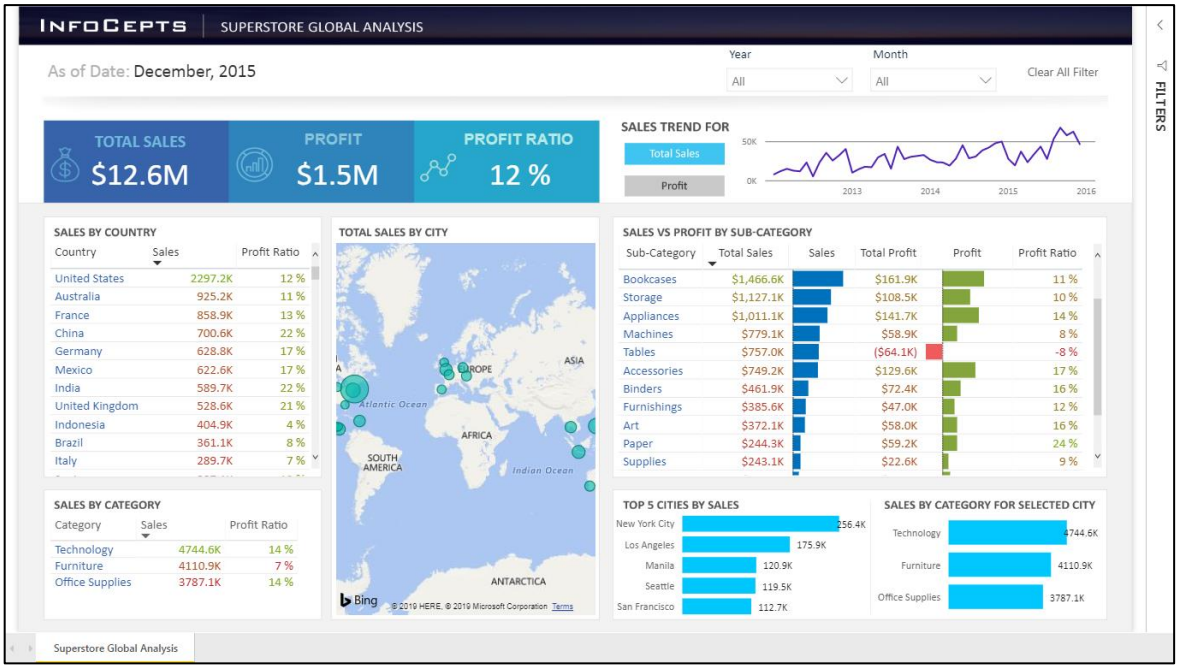
6.2 Komerční BI nástroje

Komerční platformy pro vizualizaci dat umožňují snadnou tvorbu interaktivních vizualizací dat i pro uživatele, kteří neovládají žádný programovací jazyk. Tyto platformy umožňují připojení široké škály datových zdrojů. Na rozdíl od vývoje vlastního řešení je využití komerční platformy finančně nákladné. Poskytují však jednoduché, funkční a bezúdržbové řešení, které umožňuje rychlou vizualizaci velkého množství dat. Měsíční náklady spojené s využíváním komerčních BI řešení závisí na využívaných funkcích platforem. Cena služeb jedné z nejpoužívanějších platforem PowerBI od Microsoftu začínají na 10$ za měsíc za uživatele ve verzi PRO. Verze Premium poskytuje širší množství služeb a její cena se neodvíjí od počtu uživatelů, ale od zakoupené kapacity a výpočetního výkonu, jejíž cena začíná na 5000$ za měsíc [60].

Využití komerčních platforem je vhodné v případě, kdy je nutná tvorba velkého množství reportů v uživatelsky jednoduchém prostředí.

6.2.1 Power BI

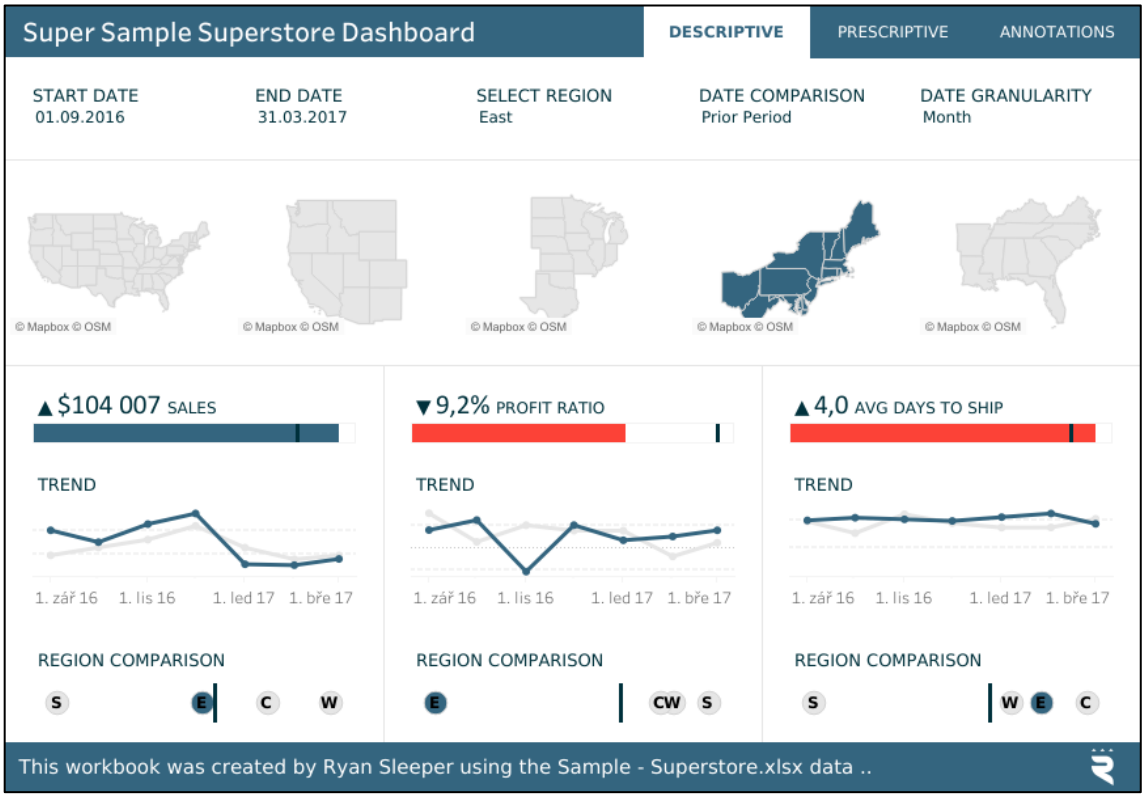
Jedním z nejpoužívanějších BI nástrojů je platforma Power BI od společnosti Microsoft. Jedná se o cloudovou platformu, která umožňuje zpracovat data z různých datových zdrojů do jednoho datového setu. Data je dále možné upravovat pomocí jazyků M a DAX. Celý koncept Power BI je rozdělen do tří platforem, Power BI Desktop sloužící pro tvorbu reportů, Power BI Service sloužící pro publikaci reportů a Power BI Mobile podporující využití reportů na mobilních zařízeních. Pro vizualizaci dat je možné využít širokou škálu již předpřipravených vizuálů, zároveň je možná i tvorba vlastních, například v Pythonu nebo R. Dále je možné využít nadstandartních funkcí, jakými jsou například NLP (zpracování přirozeného jazyka) pro vyhledávání dat v podobě pokládání otázek nebo zabudovaných funkcí umělé inteligence pro podrobnější analýzy. Nabízí rovněž několik strategií pro aktualizaci dat, od aktualizace dat v živém čase po aktualizaci v intervalech dnů či týdnů [61]. Na Obrázku 13 je návrh reportu vytvořeného v Power BI. Power BI je využíváno mnoha společnostmi, příkladem může být Nokia, HP nebo Rolls Royce.



Obrázek 13: Power BI report. Zdroj: [62]

6.2.2 Tableau

Tableau je považováno za jedno z nejlepších řešení pro vizualizaci dat. Jednou z výhod tohoto řešení je snadná organizace, správa a vizualizace dat s minimálními technickými požadavky na uživatele. Je možné připojit širokou škálu datových zdrojů pomocí Tableau Desktop, což je software sloužící pro přípravu vizualizací dat. Vytvořený dashboard obsahující vizualizace je možné sdílet s uživateli několika různými způsoby, které zajišťují různou úroveň zabezpečení dat. Jednotlivé produkty, přes které je k dashboardům možné přistupovat, jsou Tableau Public, Server, Online a Reader. Data je možné zobrazovat i na mobilních zařízeních. Tableau stejně jako PowerBI umožňuje propojení s Pythonem a v souvislosti s tím využití knihoven pro strojové učení. Dalšími speciálními funkcemi jsou například možnost využití již zabudovaných predikčních algoritmů nebo využití přirozeného jazyka pro vytěžování dat z dashboardů [63]. Společnosti využívající Tableau pro vizualizaci dat jsou například Honeywell, Lenovo nebo Verizon. Cenová politika Tableau se liší od Power BI premium verze v nutnosti zakoupení licence pro každého uživatele. Jelikož je funkcionalita obou platforem podobná, hraje výraznou roli při volbě vizualizačního nástroje právě cena. V případě nižšího počtu uživatelů se tedy může Tableau jevit jako vhodnější varianta, kdežto při velkém počtu uživatelů může být výhodnější volbou Power BI.



Obrázek 14: Tableau dashboard. Zdroj: [64]

6.2.3 Další komerční nástroje

Další platformy, které poskytují podobné funkce jako výše popsané Tableau a Power BI, jsou Qlink, SAP Lumira, ThoughtSpot, SAS Business Inteligence, případně pak nástroje od společností IBM či SAP. Všechny tyto platformy poskytují podobný standard pro vizualizaci dat a pro potřeby této diplomové práce by bylo možné využít jakoukoli z těchto platforem. Rozdíly se mohou objevovat až u pokročilých funkcí, popřípadě v oblasti distribuce reportů.

7 Zavedení produktivní údržby

Kapitoly 2 až 6 poskytují soubor informací, jež jsou základem pro implementaci systému prediktivní údržby v oblasti výrobních linek a které slouží jako vstupní body pro tvorbu nástroje popsaného v následujících kapitolách. Neexistuje přitom pouhé jedno univerzální řešení pro zavedení systému prediktivní údržby. Tento systém je nutné pro každý konkrétní případ vytvořit individuálně na základě jeho potřeb a zároveň dostupných zdrojů. Vždy je nutné uvažovat finanční náklady a návratnost daného řešení. Není tedy možné implementovat všechny techniky prediktivní údržby, ale je nutné zvolit ty, které jsou dostupné [1].

Cílem následujících kapitol je strukturovaně popsat tvorbu prvního ze sady nástrojů pro podporu systému prediktivní údržby výrobních linek v oblasti potravinářského průmyslu. Tento nástroj by měl ověřit aplikovatelnost prediktivních systémů na výrobní linky a obdobná průmyslová zařízení. Kapitola 8 je věnována výběru dat a jejich vytěžování. V kapitole 9 je popsán výběr úložiště těchto dat společně s návrhem procesu jejich ukládání. Kapitola 10 se věnuje analýze těchto dat, na jejímž základě je vybrána oblast pro aplikaci prediktivních algoritmů. Další kapitoly popisují tvorbu webové aplikace, která slouží k predikování počtu krátkých zastávek výrobní linky, umožňuje následné analýzy a řídí ukládání predikovaných dat pro možnost dalších výstupů v podobě PowerBI reportů, jejichž tvorba je popsána v kapitole 12. Výstupem této práce je zhodnocení využitelnosti navrženého systému pro identifikaci výskytu krátkých zastávek na výrobní lince, popis možného postupu pro jeho efektivní využití a návrh dalších oblastí, o něž by mohl být tento nástroj rozšířen.

8 Analyzovaná data

Data využitá v této práci byla poskytnuta výrobní firmou z oblasti potravinářského průmyslu. Název společnosti, která pro účely této práce data poskytla, není v této práci zmíněn na základě dohody o poskytnutí dat. Tento krok společně s anonymizací dat samotných zajišťuje ochranu dat před jejich

využitím k jiným účelům, což byla jedna z podmínek pro jejich poskytnutí k možnosti vypracování diplomové práce.

Analyzovaná výrobní linka se skládá z šesti na sebe navazujících strojů, které jsou propojeny dopravníky s přidruženými zásobníky. Zásobníky slouží k zajištění kontinuální produkce i v případě krátké poruchy jednoho ze strojů. V případě zastavení jednoho stroje tedy není nutné zastavit celou výrobní linku. Tato skutečnost byla uvažována při procesu vytěžování dat, kdy byla data označena na základě příslušnosti k danému stroji a slabá místa bylo možné lépe adresovat. Pro potřeby této práce byl poskytnut přístup k výrobním systémům vybrané výrobní linky, v nichž se bylo nejprve nutné zorientovat a následně vytěžit potenciálně vhodná data pro predikci v oblasti údržby. Jedná se o tři různé výrobní systémy. Prvním z nich je Manufacturing Execution System (MES), tedy výrobní informační systém zajišťující řízení výroby, plánování výroby, sběr dat, poskytování informací o produkci a další. Druhým z využitých systémů je Product Lifecycle Management (PLM), neboli řízení cyklu výrobku. V tomto systému je mimo jiné možno nalézt informace o materiálech použitých pro výrobu finálního produktu. Třetím využitým zdrojem dat je datový sklad na MS SQL serveru, kam jsou přes OPC server ukládána data z PLC. Jedná se o data, jakými jsou například rychlosti strojů, informace o stavu stroje, informace o produkci nebo neshodných produktech.

Po identifikaci vhodných zdrojů dat jsem vytvořil obecný přehled dostupných dat, která by bylo možné využít pro další analýzy a predikce. Tento seznam byl představen a konzultován s týmem údržby a procesními inženýry dané výrobní linky. Na základě poznatků těchto expertů byla vybrána vhodná data z oblastí, které mají přímý vliv na OEE parametry popsané v kapitole 3. Dále byla z prvotního seznamu odfiltrována data, která byla nerelevantní nebo mohla obsahovat nepřesné informace a celý proces predikce zkreslovat. Týmem údržby bylo doporučeno použití dat, která nejsou starší dvou let, a to s ohledem na změny na výrobní lince provedené. Starší data by mohla ovlivnit výsledky predikce negativním způsobem z důvodu jiné skladby výrobní linky. Výsledný výběr obsahuje data, která bylo možné shlukovat do čtyř tabulek. Pro zajištění anonymity dat jsem provedl zakódování těchto dat. Textové údaje byly převedeny na numerické hodnoty a informace o produkci či parametry stroje byly pozměněny pomocí koeficientů, které nemají vliv na přesnost predikcí. První skupinou dat jsou informace o plánované výrobě. Tato tabulka obsahuje tři sloupce: Datum, Rozlišení části dne podle směn a číslo výrobní zakázky.

Tabulka 1: Seznam atributů uložených v tabulce dbo.FutureProd

Date	DatePart	Order
44098	2	374
44098	2	374
44099	1	376
44099	2	376
44100	1	375
44100	2	375

Tuto tabulku je nutné aktualizovat v pravidelných intervalech, které je nutné určit na základě frekvence úprav prováděných oddělením plánování. Tuto aktualizaci je nutno provádět automaticky. Toho je možné dosáhnout například vytvořením procedur na SQL serveru a jejich pravidelným spouštěním, které lze nastavit v SQL server agentovi. Pro účely této diplomové práce byla data poskytnuta ve formě CSV souborů. K MES serveru, ze kterého je data nutné čerpat, není z bezpečnostních důvodů možný přístup z vnější sítě, v důsledku toho jsem tento krok nerealizoval a využívám pouze poskytnutých statických dat.

Druhou skupinou dat je přehled událostí na výrobní lince za dvacet měsíců výroby. Data jsou uložena v tabulce o devatenácti sloupcích obsahující více než dva miliony řádků. Sloupce jsou popsány v tabulce 1. Na obrázku 15 je znázorněna struktura dat.

Tabulka 2: Seznam atributů uložených v tabulce dbo.ProdData

Sloupec	**Popis**
Date	datum události
Eq	číslo stroje v rámci výrobní linky
Order	číslo výrobní zakázky
Start	čas začátku události
End	čas konce události
Time	doba trvání události
TotalTime	celková doba trvání události
Stop	identifikace zastavení stroje
Cat	kategorie události
Sub	podkategorie události
RootEq	číslo stroje, který událost způsobil
ReasonCode	identifikační číslo důvodu zastavení
Daypart	identifikace části dne podle směny
TS	rychlost stroje maximální
AS	rychlost stroje průměrná
TP	vyrobené množství
RP	neshodné množství
Origin	prvotní příčina události
Cat2	druhotná kategorie události

	Date	Eq	Order	Start	End	Time	TotalTime	Stop	Cat	Sub	RootEq	ReasonCode	Daypart	TS	AS	TP	RP	Origin	Cat2
1	43816	4	253	43816.730230000001	43816,73069	40	NULL	0	1	1	4	7	1	84	82,04	536	4	NULL	1
2	43816	4	253	43816.729879999999	43816,73023	30	NULL	0	1	2	4	7	1	84	64,85	400	0	NULL	1
3	43816	4	253	43816.729800000001	43816,72988	7	7	1	2	3	4	7	1	84	26,33	4	0	3	5
4	43816	4	253	43816.729530000004	43816,7298	24	24	0	1	2	4	12	1	84	66,96	300	4	NULL	1
5	43816	4	253	43816.728479999998	43816,72953	90	90	1	2	3	4	12	1	84	12,35	16	4	5	5

Obrázek 15: Přehled uložených dat v tabulce dbo.ProdData

Tato data byla vytěžena z MS SQL datového skladu. Tento datový sklad jsem zvolil jako ideální datové úložiště, na které je vyvíjená aplikace připojena. Z tohoto důvodu není tato data nutné nikam přesunovat ani jiným způsobem manipulovat. Pro zajištění rychlého chodu aplikace by bylo vhodné uchovávat v této tabulce jen určité množství dat a pro historická data navrhnout systém automatického odmazávání nebo zálohování na jiné místo. V rámci této práce to však z důvodu využití pouze poskytnutých dat není nutné a pro zajištění rychlejší odezvy aplikace byla vytvořena pouze indexace zmíněné tabulky. Návrh architektury aplikace je popsán v následující kapitole.

Třetí skupinou dat jsou informace o materiálech využívaných k výrobě finálního produktu. Stejně jako v případě již zmíněných dat není z bezpečnostních důvodů možné využití jejich zdroje, v tomto případě PLM, ale byla poskytnuta v podobě CSV souboru. Využití dat o materiálech bylo doporučeno procesními techniky, a to z důvodu vysoké variance používaných materiálů na analyzované výrobní lince. Konkrétnější výběr sledovaných materiálů byl konzultován s inženýrem kvality, který pomohl identifikovat materiály a jejich fyzikální parametry, jež mají potenciál ovlivňovat kontinuitu produkce nebo procento neshodných produktů. Výsledný výběr obsahuje 72 sloupců s informacemi o typu materiálů nebo jejich fyzikálních parametrech. Struktura dat je znázorněna na obrázku 16. Názvy sloupců představují anonymizované názvy materiálů nebo jejich parametrů, na řádcích jsou uvedeny číselné hodnoty představující jednotlivé kategorie pro sloupcové atributy.

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	aa
1	1	1	1_1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	2	2_2	2	1	2	2	2	1	1	1	1	1	1	1	1	2	2	2	2	1	1	1	2	1	1	2
3	3	3	3_3	3	1	3	3	3	2	1	1	2	1	1	1	1	3	3	3	3	1	1	1	3	1	1	3

Obrázek 16: Tabulka dbo.Materials. ID sloupec a obsahuje informaci o zakázce, sloupce b až bt obsahují informace o materiálech.

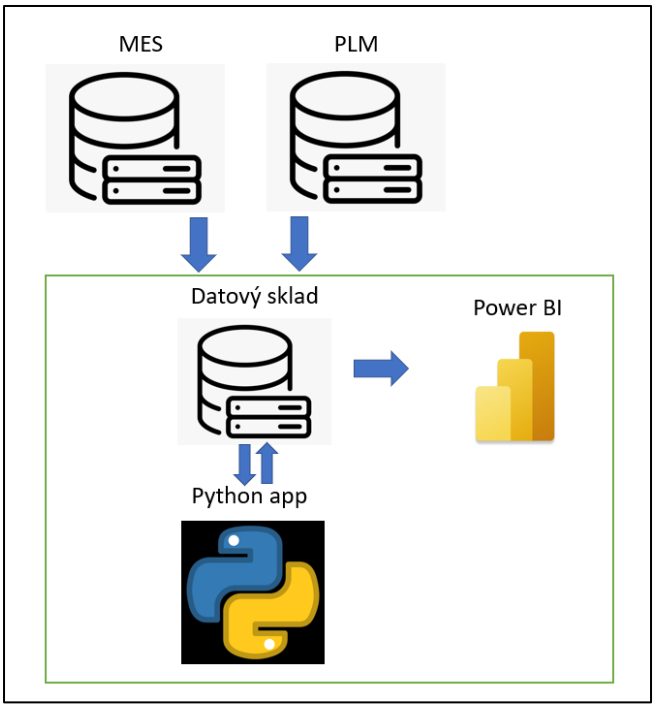
V případě dat uložených v tabulce *dbo.Materials* je navržený postup pro reálné využití stejný jako u dat čerpaných z MES databáze, tedy vytvoření automatického procesu přesunu potřebných dat do datového skladu.

Poslední skupina zvolených dat obsahuje informace o počtu neshodných produktů vyřazených z výrobní linky a důvodu tohoto vyřazení. Tato data jsou agregována po hodinových intervalech. Tato

data byla ve vyvíjené aplikaci využita pouze pro výpočet parametru kvality a možnost jejich využití v rámci prediktivní části aplikace je diskutována v kapitole 14. Všechny zmíněné tabulky obsahují ID výrobní zakázky pro možnost jejich kombinace. Tyto čtyři tabulky byly uloženy do CSV souborů a poskytnuty k vypracování této diplomové práce.

9 Datové úložiště

Ideálním způsobem ukládání dat sloužících pro vytvářenou prediktivní aplikaci by bylo využití již existujícího datového skladu na MS SQL serveru, ze kterého byla vytěžena data historických událostí. Extrakci, transformaci a nahrávání dat (ETL) z MES a PLM serverů by bylo možné automatizovat využitím Microsoft SQL Server Integration Services (SSIS), který umožňuje tvorbu balíčků automatizující databázové procesy nebo již v předchozí kapitole zmíněnými procedurami na SQL serveru. Druhou možností by bylo aplikaci připojit ke všem zmíněným datovým zdrojům, což ale není ideálním řešením, jelikož v případě MES a PLM databází se jedná o kritické produkční systémy. Vzhledem ke skutečnosti, že poskytnutá data byla uložena do CSV souborů a v rámci této diplomové práce není možné využít přímo zmíněný datový sklad, byla zvolena alternativa, kterou je simulace tohoto datového skladu využitím MS SQL serveru Express edice. Na obrázku 17 je zobrazena architektura provedeného řešení práce s daty. Rámečkem je označena část, která je součástí této práce.

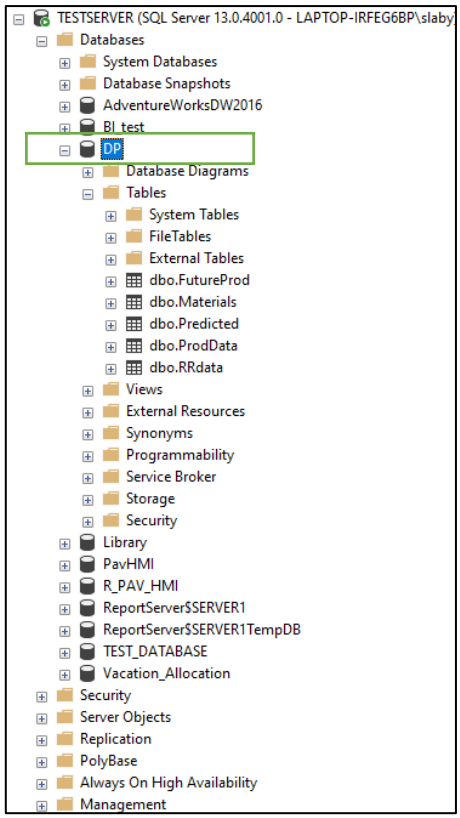


Obrázek 17: Architektura vyvíjeného systému. Zeleným rámečkem je označena část zahrnutá v diplomové práci

Python aplikace čte data z datového skladu, provádí predikce a jejich výsledky ukládá zpět do datového skladu. Datový sklad je dále využívám BI nástrojem *Power BI*, který slouží pro vizualizaci výsledků.

Řešení v podobě relační databáze jsem zvolil z důvodu možné snadné implementace do firemního IT ekosystému v případě prokázání funkčnosti a užitečnosti této aplikace. Alternativní řešení pro ukládání dat jsou uvedena v kapitole 5. Jedním z nich je ukládání dat do souborů, které by v tomto případě bylo také vhodné. Případné rozšíření aplikace o další výrobní linky, popřípadě o další funkcionality, by ale mohlo vést k duplikování dat a k nárustu počtu souborů, což by postupně vedlo k nepřehlednosti celého řešení. Další alternativou je Systém distribuovaných souborů HDFS, jehož realizace na jednom počítači postrádá smysl. Využití NoSQL databází by v tomto případě bylo možné, avšak struktuře využitých dat, která je znázorněna na obrázcích 15 a 16, spíše odpovídá relační řešení. Pro zvolené řešení je možné data předpřipravit k budoucímu horizontálnímu škálování pro případ, že by bylo nutné nahrávat větší množství dat, případně data z velkého počtu výrobních linek.

Pro správu MS SQL serveru a přístupu k datům na něm uloženým jsem využil programu MS SQL Server Management Studio. Pomocí tohoto programu byl založen server a na něm databáze s názvem DP. V této databázi byly vytvořeny čtyři tabulky pro analyzovaná data a jedna tabulka pro ukládání predikcí. Obrázek 18 znázorňuje organizaci databáze.



Obrázek 18: Relační databáze DP pro ukládání dat využitých vyvíjenou aplikací

K uložení poskytnutých dat do databáze bylo využito Open Source knihoven *pandas*, *pyodbc* a *fast_to_sql* pro Python. Tyto knihovny jsou dále využity pro veškerou komunikaci Python aplikace s MS

SQL Serverem. Část kódu připravená pro nahrání dat z CSV souborů do tabulky *dbo.Materials* v databázi *DP:*

```
01    import pandas as pd
02    import pyodbc
03    from fast_to_sql import fast_to_sql as fts
04
05
06    def conn_string(server, database):
07        conn = pyodbc.connect(
08            "Driver={SQL Server Native Client 11.0};"
09            "Server="+server+";"
10            "Database="+database+";"
11            "Trusted_Connection=yes;"
12        )
13        return conn
14
15
16    conn = conn_string("TESTSERVER", "DP")
17    df = pd.read_csv(r"C:\Users\slaby\Desktop\data\Materials.csv")
18
19    create_statement = fts.fast_to_sql(df, "Materials",
20                                       conn,
21                                       if_exists="replace",
22                                       temp=False,custom={"a":"INT PRIMARY KEY"})
23    conn.commit()
24    conn.close()
```

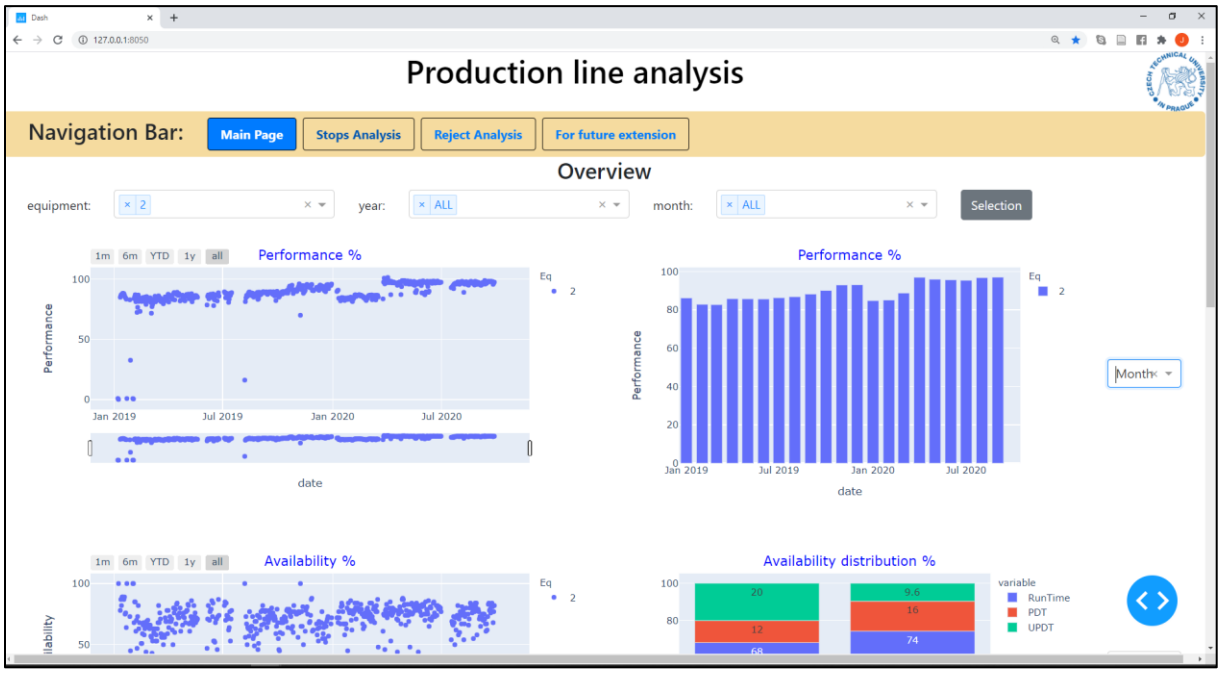
10 Výběr oblasti pro predikci

V rámci konceptu TPM popsaného v kapitole 2.3 je hlavním sledovaným ukazatelem OEE, tedy celková efektivnost zařízení. Tento ukazatel je možné využít pro hodnocení efektivity údržby a z tohoto důvodu byl zvolen jako základ pro identifikaci oblastí, které způsobují největší ztráty a tím pádem představují největší potenciál pro aplikaci prediktivních metod. Výsledný ukazatel OEE je součinem ukazatelů Performance, Availability a Quality. Tyto ukazatele je možné dále rozšířit o podpůrné ukazatele, jakými jsou například MTBF nebo procentuální vyjádření doby trvání neplánovaných zastávek.

Pro výpočet výše zmíněných ukazatelů byla použita data uložená v tabulce *ProdData* v databázi *DP*. Jedná se o data jednotlivých událostí na výrobní lince. Vzhledem ke kritičnosti zmíněných ukazatelů a potřebě jejich průběžného monitoringu byla jejich vizualizace zakomponována do vytvářené predikční aplikace, a to v podobě interaktivních grafů.

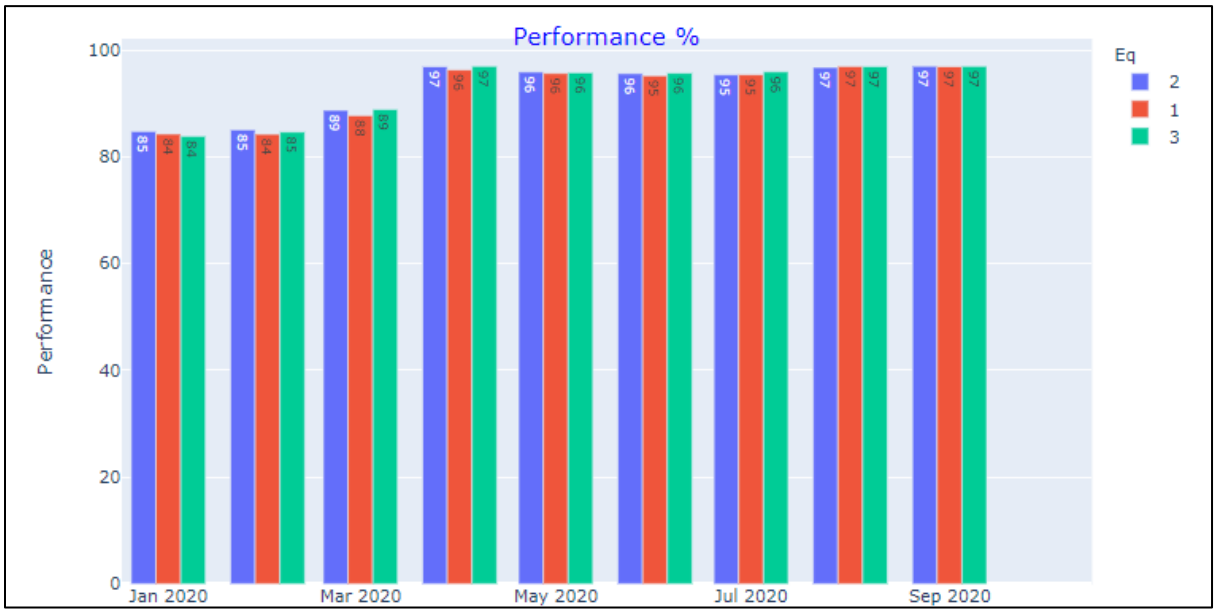
Pro tvorbu této aplikace byly vytipovány dva vhodné frameworky. Framework Dash [58] a Bokeh [65]. Po prostudování dokumentací k těmto frameworkům byl jako vhodnější řešení zvolen Dash, a to především z důvodu přehlednosti jeho dokumentace a velkého počtu kvalitně zpracovaných návodů pro práci s jednotlivými komponenty.

Vizualizaci vybraných parametrů pro měření efektivity výroby a údržby výrobní linky jsem vyhradil úvodní stránku aplikace. Jelikož k výběru oblasti pro nasazení prediktivních algoritmů bylo nutné provést analýzu poskytnutých dat a v rámci této analýzy data vizualizovat, byla tato úvodní stránka aplikace vytvořena na začátku celého procesu. Obrázek 19 obsahuje náhled úvodní stránky vyvíjené aplikace.



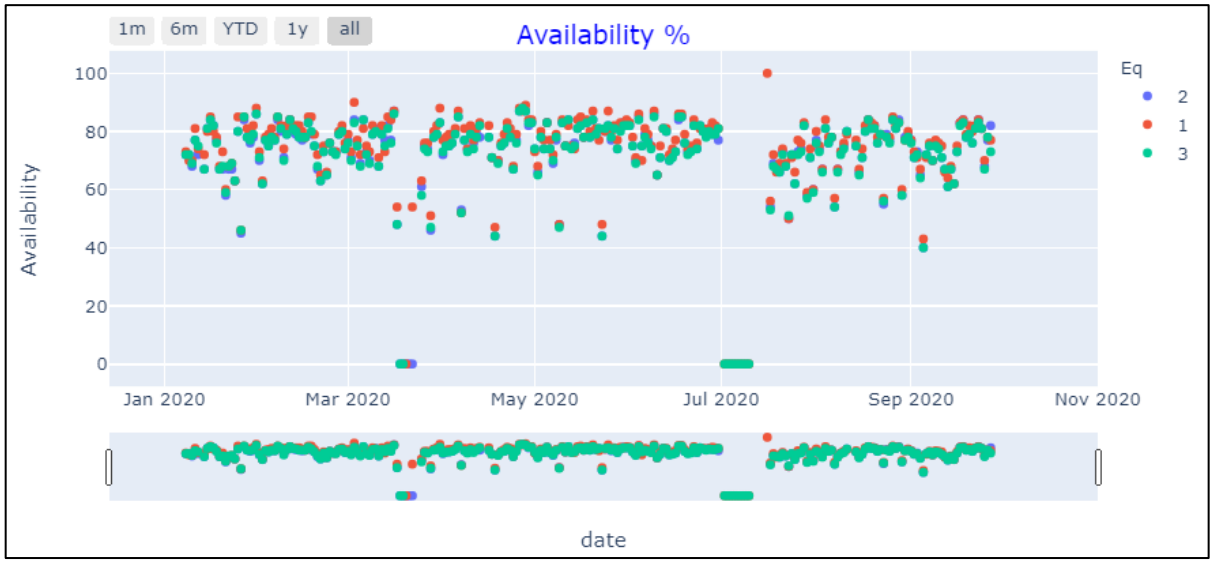
Obrázek 19: Vizualizace kritických ukazatelů na úvodní stránce vyvíjené aplikace

Prvním krokem při výběru vhodné oblasti pro nasazení prediktivních algoritmů bylo porovnání ukazatelů Performance, Availability a Quality. Z analýzy všech tří ukazatelů vyplývá jejich rostoucí trend, což musí být následně uvažováno při tvorbě predikčních algoritmů pro zajištění kvality jejich predikcí. Ukazatel Performance, tedy výkon, vykazuje stabilní hodnoty přes 90 %. Výsledky za devět měsíců pro první tři stroje v rámci výrobní linky jsou prezentovány na obrázku 20.



Obrázek 20: Ukazatel výkonu po měsících pro první tři zařízení výrobní linky

Ukazatel Availability, tedy dostupnost zařízení, značící procento celkového času běhu stroje, je pro první tři stroje za první tři čtvrtletí na úrovni 70 %. Na obrázku 21 je přehled ukazatele dostupnosti po dnech pro první tři zařízení výrobní linky.

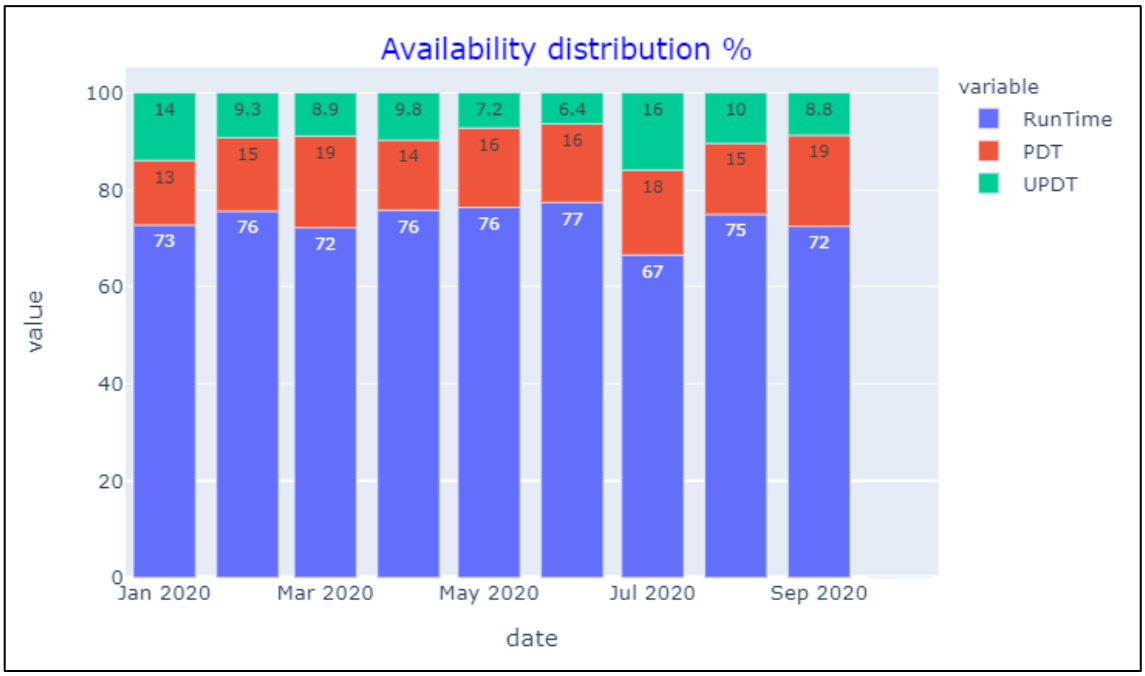


Obrázek 21: Ukazatel dostupnosti

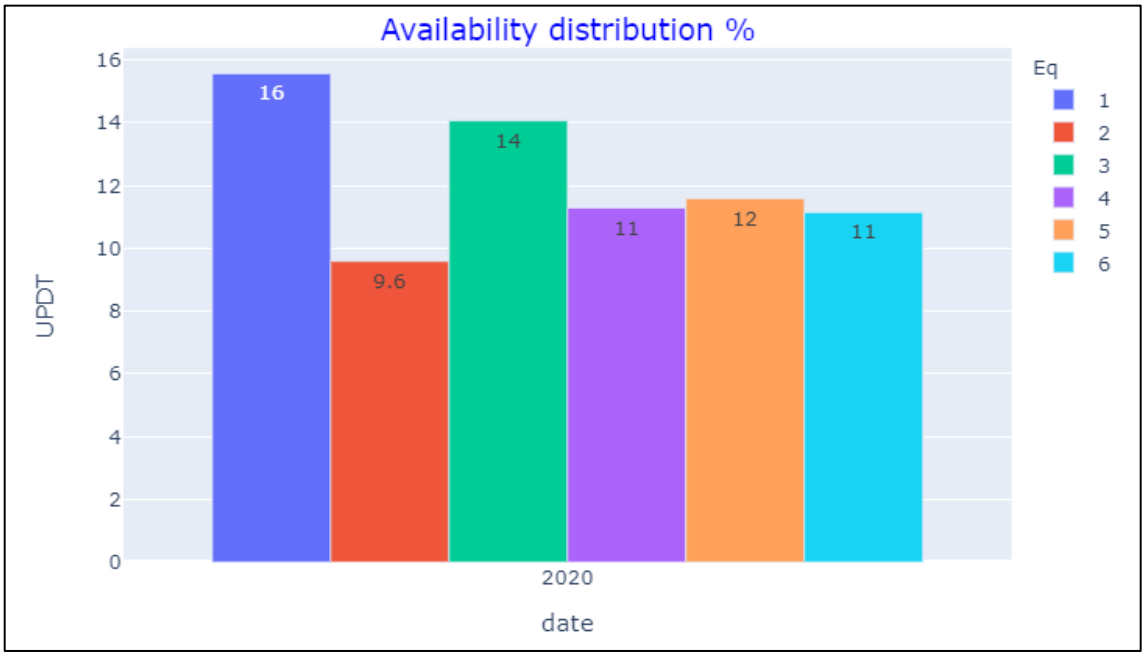
Ukazatel kvality dosahuje stabilně hodnot přes 99 % u všech zařízení výrobní linky. Narozdíl od předchozích dvou ukazatelů je nutné ztráty kvality sečíst přes všechny stroje. I přes tento fakt je celková ztráta v oblasti kvality nižší než ztráta u ostatních ukazatelů.

Z výše uvedeného přehledu vyplývá, že největší procento ztrát, a tím pádem nejvyšší vliv na celkový ukazatel OEE, je spojen s ukazatelem dostupnosti zařízení. Ztráty na dostupnosti zařízení je možné rozdělit na ztráty plánované a neplánované. Plánované ztráty jsou spojeny s plánovanými

událostmi, jakými jsou preventivní údržba, čištění zařízení nebo čas nutný na přenastavení výrobní linky při změně typu produkce. Neplánované ztráty jsou navázány na neplánované zastávky, jakými jsou poruchy, krátké procesní zastávky nebo neplánované čekání způsobené například nedostatkem vstupních materiálů. Poměr plánovaných a neplánovaných zastávek pro druhý stroj výrobní linky je zobrazen na obrázku 22. Porovnání procent neplánovaných zastávek je možné získat z obrázku 23.

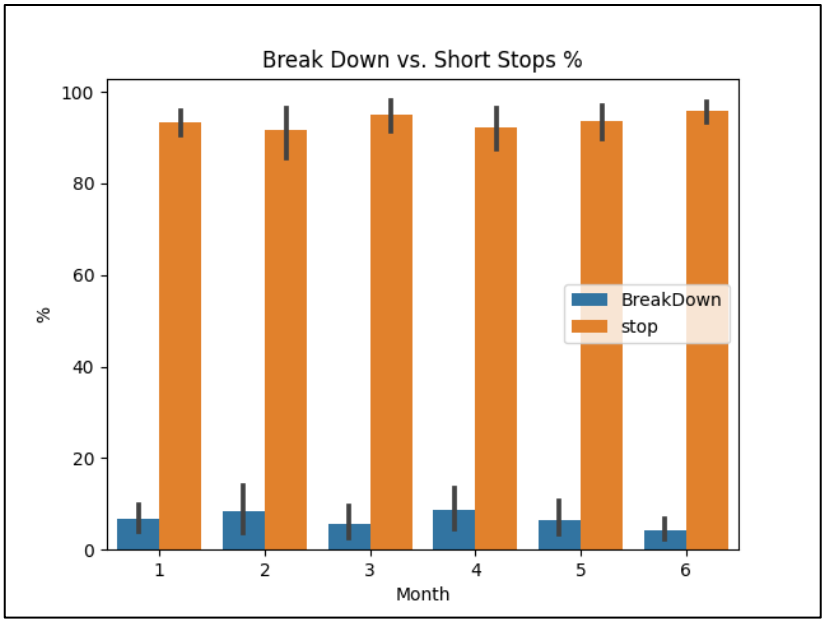


Obrázek 22: Distribuce plánovaných a neplánovaných zastávek pro stroj č. 2 analyzované výrobní linky



Obrázek 23: Procenta doby trvání neplánovaných zastávek pro stroje výrobní linky

Touto postupnou analýzou OEE ukazatele byl získán seznam ztrát a jejich procentuální vyjádření. Celková hodnota ukazatele OEE za první tři čtvrtletí roku 2020 je 68.5 %. Identifikované ztráty a jejich procentuální dopad: Ztráty výkonu - 6.5 %, Ztráty dostupnosti - 24.5 %, Ztráty kvality - 0.5 %. Ztráty spojené s dostupností je možné rozdělit na ztráty: plánované - 15.5 % a neplánované - 9 %. Vzhledem k pozitivnímu vývoji ukazatele výkonu v druhé části analyzovaného období je tato ztráta v současné době nižší než uvedený průměr. Na základě těchto dat bylo rozhodnuto o zaměření vývoje aplikace na ztrátu v oblasti dostupnosti. Přestože je doba trvání plánovaných zastávek vyšší než neplánovaných, potenciál pro snížení ztrát je spíše na straně zastávek neplánovaných. Jak již bylo zmíněno, do plánovaných zastávek spadají úkony preventivní údržby, čištění zařízení a úpravy stroje při změnách produkce. Tyto úkony je zcela jistě možné zefektivnit a tím zkrátit čas těchto plánovaných zastávek, ale není možné je zcela eliminovat. Z tohoto důvodu je vytvářená prediktivní aplikace zaměřena na neplánované zastávky. Tyto je dále možné rozdělit na poruchy a krátké procesní zastávky. Poměr těchto dvou kategorií včetně vymezení odchylek pro jednotlivé stroje je zobrazen na obrázku 24.



Obrázek 24: Porovnání výskytu krátkých zastávek a poruch

V rámci popisovaného procesu výběru vhodné oblasti pro nasazení prediktivních algoritmů jsem pro možnost analýzy musel připravit více přehledů než obsahuje úvodní stránka aplikace. K tomu jsem využil knihovny *Seaborn* pro Python. Následuje ukázka části kódu pro tvorbu grafických přehledů využitím dat uložených na SQL serveru:

```
01    import pandas as pd
02    import matplotlib.pyplot as plt
03    import seaborn as sns
04    import pyodbc
05
06    def conn_string(server, database):
07        conn = pyodbc.connect(
08            "Driver={SQL Server Native Client 11.0};"
09            "Server=" + server + ";"
10            "Database=" + database + ";"
11            "Trusted_Connection=yes;"
12        )
13        return conn
14    conn = conn_string("TESTSERVER", "DP")
15    sql_data = """
…
47    """
48
49    df = pd.read_sql(sql_data, conn)
50
51    g =sns.barplot(data=df, x= "Eq", y = "StopPercentage", hue="stoptype",
ci=None)\
52    .set_title("Break Down vs. Short Stops %")
53    plt.xlabel("Month")
54    plt.ylabel("%")
55    plt.legend(loc='center right')
56    plt.show()
```

Proměnná sql_data na řádku 15 obsahuje dotaz do databáze, který je přiložen v příloze 1. Z dat zobrazených na obrázku 24 lze vypozorovat přibližně 90% podíl krátkých procesních zastávek na celkových neplánovaných zastávkách. Delší poruchy jsou zastoupeny přibližně 10 %.

Vzhledem k výše zmíněnému bude druhá část aplikace vytvořená v rámci této diplomové práce zaměřena na predikci krátkých procesních zastávek. Tuto kategorii ztrát jsem identifikoval jako nejkritičtější a s největším potenciálem pro zvýšení OEE ukazatele. V kapitole 14 je diskutován potenciál rozšíření této aplikace o ostatní zmíněné ztráty.

11 Prediktivní aplikace

Na základě poskytnutých dat, doporučení od odborníků z praxe a provedených analýz byla identifikována ztráta na OEE ukazateli způsobená krátkými zastávkami výrobní linky jako oblast s nejvyšším potenciálem pro zvýšení OEE ukazatele. Aplikace, která je vyvíjena v rámci této diplomové práce, by měla za využití prediktivních algoritmů sloužit týmu údržby, ale také týmu obsluhujícímu výrobní linku jako podpůrný nástroj predikující četnost výskytu budoucích krátkých zastávek. Cílem této aplikace je poskytovat taková data, která umožní zaměření na budoucí zastávky v dostatečném předstihu, aby mohly být provedeny kroky vedoucí k minimalizaci jejich výskytu. Dalším požadavkem na tuto aplikaci je vizualizace predikovaných dat a zároveň možnost pro uživatele data dále analyzovat.

Během návrhu aplikace jsem se zaměřil na to, aby zajišťovala tři základní funkce. První funkcí je vizualizace dat, umožňující sledovat vývoj kritických ukazatelů. Tato část by měla zajistit například možnost sledování dopadu provedených úprav nebo jiných zásahů na výrobní lince provedených. Druhou funkcionalitou aplikace je predikce výskytu krátkých zastávek, a to pro jednotlivá zařízení výrobní linky. Predikovat je možné jak celkový počet zastávek, tak počet zastavení pro konkrétní typy zastávek. Vzhledem k vysokému počtu typů zastávek není možné vytvořit unikátní prediktivní algoritmus pro každý z nich. Proto je do aplikace zakomponováno více algoritmů, které uživatel může využít a na základě představených dat vybrat ten nejvhodnější, který uloží pro další analýzy. Uživatel by měl mít zároveň možnost využít predikovaných dat pro další analýzy. K tomu slouží třetí funkcionalita aplikace, která uživatelům umožňuje tvorbu „dočasných" grafů, které by měly sloužit podrobnějším analýzám individuálních oblastí.

Pro tvorbu aplikace byl zvolen programovací jazyk Python a integrované vývojové prostředí PyCharm Community Edition, a to především z důvodu předchozích zkušeností s tímto programovacím jazykem a prostředím. Další možnou variantou bylo využití jazyka R. Tento je popsán v kapitole 4.1. Pro možnost nasazení vyvíjené aplikace do reálného provozu bylo nutné uvažovat způsob její distribuce mezi uživatele. Jednou z možností by byla distribuce samotného kódu aplikace, popřípadě vytvoření .exe souboru pro aplikaci pomocí PyInstaleru nebo obdobného softwaru. Toto řešení by však v případě dalšího vývoje aplikace vedlo k existenci různých verzí a obecně by bylo uživatelsky nekomfortní, a to především pro uživatele s nižšími dovednostmi v oblasti IT. Vhodnější variantou je tvorba webové aplikace, kterou je možné spustit na lokálním serveru nebo zvolit některé z mnoha dostupných cloudových řešení. Tento způsob umožňuje lepší možnosti zabezpečení aplikace a zároveň je uživatelsky pohodlnější. Takovéto řešení je zároveň schopné zajistit rychlejší chod aplikace vzhledem k vyšším výpočetním výkonům serverů proti běžným osobním počítačům.

Vzhledem k výše zmíněným důvodům jsem se rozhodl vytvořit webovou aplikaci, která bude spuštěna na lokálním serveru a přístup k ní by měl být zajištěn pro všechny uživatele na lokální síti. Pro tvorbu aplikace bylo využito frameworku Dash, který jsem zvolil z důvodu jeho zaměření na tvorbu aplikací poskytujících kvalitní vizualizaci dat.

11.1 Predikce krátkých zastávek

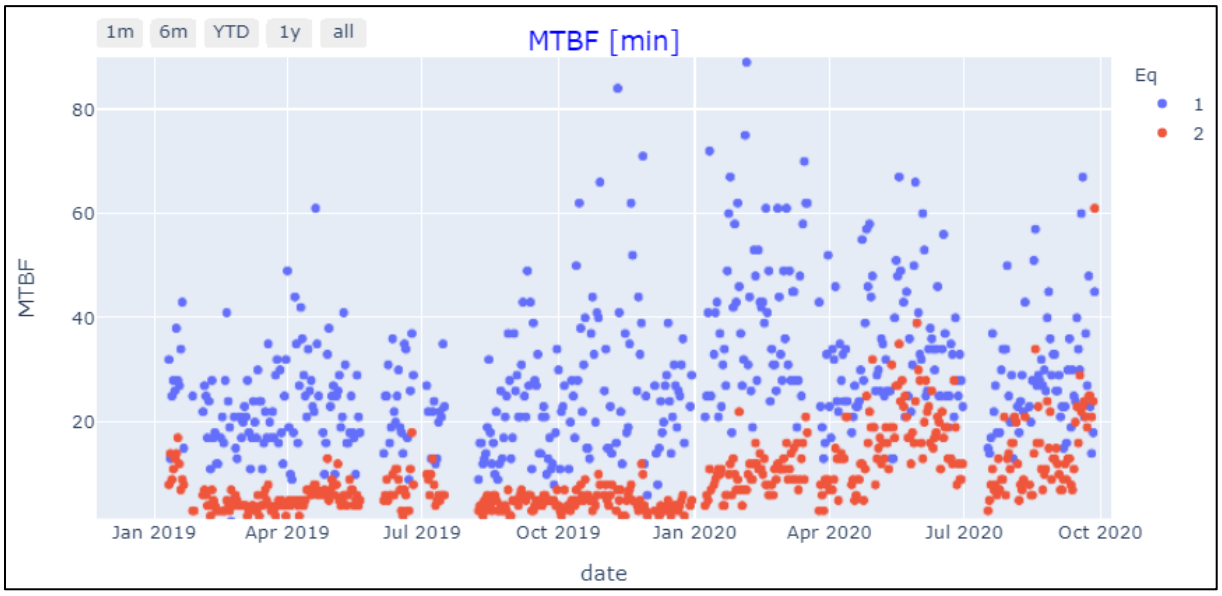
Hlavní funkcionalitou vyvíjené aplikace je predikování krátkých zastávek, což by mělo umožnit jejich předejití nebo alespoň jejich minimalizaci. Z poskytnutých dat byla zjištěna existence 310 různých typů těchto zastávek, které způsobují zastavení jednotlivých zařízení výrobní linky. Na základě diskusí s procesními inženýry byla zjištěna spojitost těchto zastávek s typem vyráběného produktu. Analyzovaná výrobní linka je schopna vyrábět produkty různých rozměrů, počtu kusů v balení a

k výrobě produktů jsou používány různé kombinace materiálů. Důvodem těchto změn je produkce výrobků různých značek, popřípadě pro různé zahraniční trhy, které mají specifické požadavky na složení produktu, velikost produktu nebo jeho design. Výhodou takovéto univerzálnosti zařízení je úspora na finančních nákladech, které by byly vyšší v případě pořízení většího počtu zařízení jednoúčelových. Tato univerzálnost s sebou ale přináší komplikace v podobě nutnosti neustálého nastavování zařízení podle typu produkce. Kombinace používaných materiálů se zároveň průběžně mění. Důvodem takovýchto změn může být například požadavek trhu nebo změna dodavatele vstupního materiálu. Z tohoto důvodu byla pro prediktivní algoritmy zvolena i data fyzikálních parametrů daných materiálů. Příkladem sledovaných parametrů materiálů je například hmotnost obalového materiálu, savost podkladového materiálu nebo hustota náplně. Použitím těchto dat je zajištěna porovnatelnost produkce i v případě změny dodavatele obalového materiálu, pokud jeho hmotnost a další parametry zůstanou stejné.

Dalším vstupem pro prediktivní algoritmy jsou data uložená v tabulce *dbo.ProdData.* Tato tabulka obsahuje přes dva miliony řádků, ve kterých je zaznamenáno 21 měsíců produkce této výrobní linky. Pro účely predikce bylo nutné tato data agregovat. Agregace byla provedena na základě datumu, směny, výrobní zakázky a stroje. Touto agregací bylo získáno 1727 záznamů pro každý typ krátké zastávky. Jelikož tato agregace nezajišťuje rozložení na stejně dlouhé výrobní úseky, bylo nutné ukazatel počtu krátkých zastávek převést na normovaný ukazatel počtu krátkých zastávek za den. K tomu byl použit následující výpočet:

$$\text{počet zastávek za den} = \frac{\text{počet zastávek}}{\frac{\text{Doba trvání produkce [s]}}{(24*60*60)}} \tag{7}$$

Tento ukazatel udává, kolik by bylo generováno krátkých zastávek v případě produkce trvající 24 hodin. Cílem využitých algoritmů tedy bude predikce tohoto ukazatele pro budoucí zakázky na výrobní lince se vstupními daty parametrů použitých materiálů. Před implementací prediktivních algoritmů bylo nutné provést analýzu historických dat. Analýzou byla zjištěna přítomnost outlierů, tedy bodů, které se vymykají standardním hodnotám daného ukazatele. Dále byla zjištěna přítomnost nulových hodnot. Tyto se vyskytují v neprodukčních intervalech, kterými jsou například preventivní údržby. Tyto body bylo nutné z dat odfiltrovat, jelikož by jejich přítomnost mohla negativně ovlivňovat výsledky predikcí. Dále byl pro analýzu využit ukazatel MTBF popsaný v kapitole 3.2. Na základě tohoto ukazatele byla odhalena přítomnost trendu v analyzovaných datech. To je možné sledovat na obrázku 25 pro první dva stroje výrobní linky.

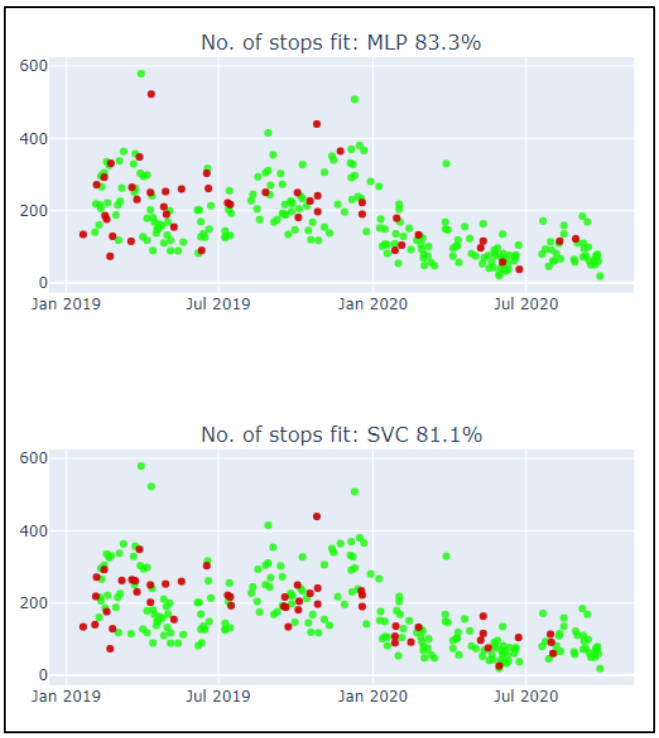


Obrázek 25: Ukazatel MTBF vykazující rostoucí trend pro stroj 1 a 2

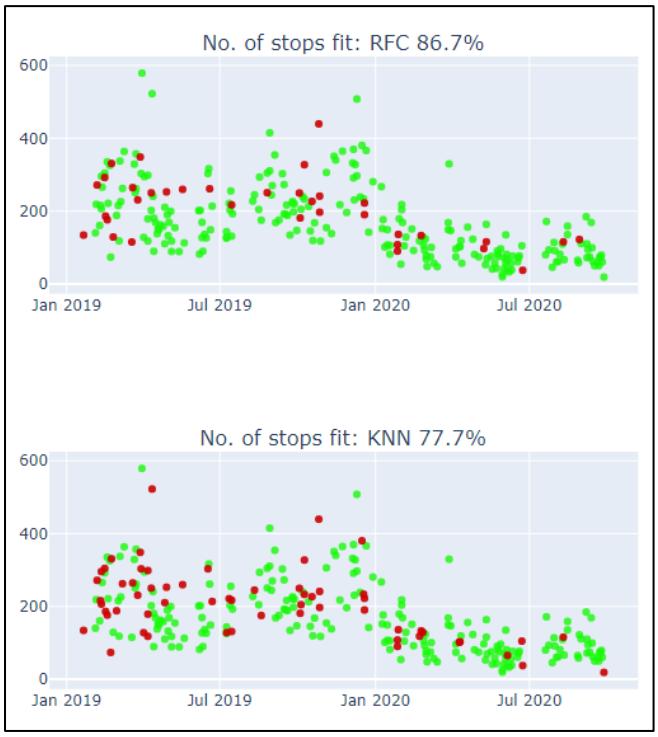
Z grafu na obrázku 25 je patrný rostoucí trend tohoto ukazatele, který značí průběžný pokles krátkých zastávek za den od začátku roku 2019 do konce září roku 2020. Výpočet MTBF na obrázku 25 byl proveden pro celkový počet krátkých zastávek, obdobné výsledky však byly získány i v případě MTBF pro jednotlivé typy krátkých zastávek. Tento trend by mohl mít negativní vliv na přesnost predikovaných dat, proto jej bylo nutné z dat odstranit. Jelikož rostoucí trend ukazatele MTBF vykazovaly všechny typy krátkých zastávek, byl trend odstraněn pomocí kategorizace ukazatele počet zastávek za den. Toho bylo docíleno vytvořením tříměsíčních průměrů a na jejich základě rozdělení dat do pěti kategorií označených 1 až 5, kde 1 značí extrémně nízký počet zastávek, 2 značí nižší počet zastávek, 3 označuje průměrný počet zastávek, 4 zvýšený počet zastávek a 5 značí extrémně vysoký počet zastávek. Tento přístup rozdělení počtu krátkých zastávek do kategorií může být přínosný i pro uživatele aplikace, kteří obsluhují více výrobních linek. Příkladem mohou být členové týmu údržby, pro které by konkrétní počet zastávek pro jeden typ zastávky nemusel být vypovídající z důvodu vysokého počtu typů těchto událostí a rozdílných hodnot na různých výrobních linkách. Zároveň pro eskalaci provedení akce na stroji postačuje informace o predikci vyššího počtu zastávek a není nutné znát jejich konkrétní počet. Úpravou dat byl snížen počet záznamů pro každý typ zastávky na 1334 a sloupec udávající počet stopů za den byl nahrazen sloupcem kategorických hodnot rozdělujících výskyt zastávek do pěti skupin. Tato data a na ně nalinkovaná data z tabulky materiálů slouží jako vstup pro prediktivní algoritmy.

Otestováno bylo celkem šest klasifikačních algoritmů popsaných v kapitole 4. Ty byly naprogramovány s využitím Python knihovny *scikit-learn*. Jedná se o algoritmy *Multi Layer Perceptron* (MLP), *Random Forest*, *AdaBoost Classiefier*, *SVM*, *KNN* a *Ridge Classifier*. Ladění parametrů probíhalo

na základě informací dostupných v dokumentaci knihovny scikit-learn [66] a pomocí informací čerpaných z literatury [33, 21, 29]. Postupným laděním parametrů jednotlivých algoritmů se podařilo dosáhnout predikční přesnosti pohybující se okolo 85 % pro algoritmy MLP, Random Forest, SVM. U algoritmu KNN se podařilo dosáhnout výsledků přesahujících 75% predikční přesnost a pro algoritmy AdaBoost a Ridge Classifier bylo dosaženo přesnosti 50 %. Uvedených hodnot bylo dosaženo pro predikci celkového počtu zastávek jednotlivých strojů výrobní linky. Pro predikci jednotlivých typů zastávek bylo dosaženo stejných výsledů pouze pro 50 nejfrekventovanějších typů. Kvalita predikce koreluje s frekvencí zastávek. Tato skutečnost může být způsobena tím, že tyto typy krátkých zastávek nejsou ovlivněny použitými materiály, a tím pádem je jejich predikce na základě těchto materiálů nevhodná. Diskuse nad způsobem predikce těchto zastávek je obsažena v kapitole 14. Pro účely vyvíjené aplikace byly vybrány čtyři algoritmy, které během testování dosáhly nejvyšší přesnosti. Postup implementace těchto algoritmů je popsán v kapitole 11. Výsledky predikce pro celkový počet zastávek druhého stroje výrobní linky jsou uvedeny na obrázcích 26 a 27. Grafy na těchto obrázcích jsou součástí vyvíjené aplikace a slouží uživatelům pro rozhodování při volbě vhodného algoritmu. Zeleně jsou označeny body, které byly predikovány správně, červeně jsou poté označeny body určené chybně. Rozdělení dat mezi data trénovací a testovací bylo v poměru 80/20.



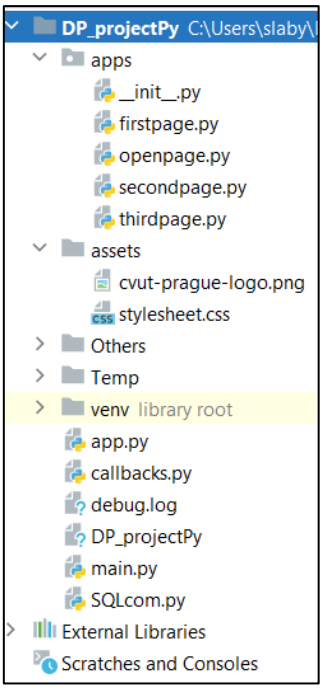
Obrázek 26: Úspěšnost predikce MLP a SVC algoritmů pro počet krátkých zastávek druhé stroje výrobní linky. Zeleně – správně predikované hodnoty, Červeně – chybně predikované hodnoty



Obrázek 27: Úspěšnost predikce RFC a KNN algoritmů pro počet krátkých zastávek druhé stroje výrobní linky. Zeleně – správně predikované hodnoty, Červeně – chybně predikované hodnoty

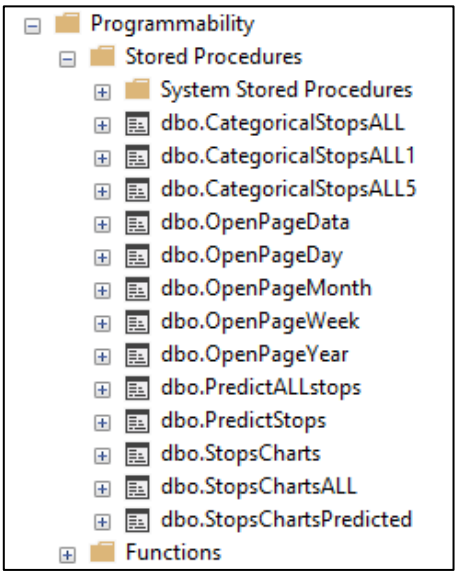
11.2 Tvorba prediktivní aplikace

Prvním krokem při tvorbě této aplikace bylo seznámení se s dokumentacemi a základními funkcionalitami využitých komponent, kterými jsou MS SQL server a jeho Management Studio, vývojové prostředí Pycharm, framework Dash a knihovna strojového učení scikit-learn. V dalším kroku jsem musel navrhnout rozložení Python projektu tak, aby bylo umožněno snadné rozšiřování aplikace o další funkce a aby byl celý projekt přehledný i pro případné další uživatele. Architektura Python projektu s názvem DP_projectPy je zobrazena na obrázku 28. V projektu jsem založil celkem osm .py souborů, pro jednotlivé oblasti, které je nutné vytvořit v rámci této aplikace. Soubor main.py je hlavním souborem, který obsahuje layout celé aplikace dále rozšiřovaný o jednotlivé stránky a zajišťuje inicializaci a chod lokálního serveru. Soubor app.py obsahuje pouze konfiguraci Dash aplikace. Py soubory obsažené v balíčku apps slouží pro návrh jednotlivých stránek aplikace, které jsou volány ze souboru main.Py. Soubor callbacks.py obsahuje všechna zpětná volání, která ovládají jednotlivé interaktivní komponenty aplikace, spouští predikce a přepínají mezi jednotlivými URL adresami aplikace. Posledním souborem je SQLcom.py, který zajišťuje komunikaci s SQL serverem a jsou v něm definovány jednotlivé dotazy do databáze.



Obrázek 28: Architektura projektu prediktivní aplikace v programu Pycharm

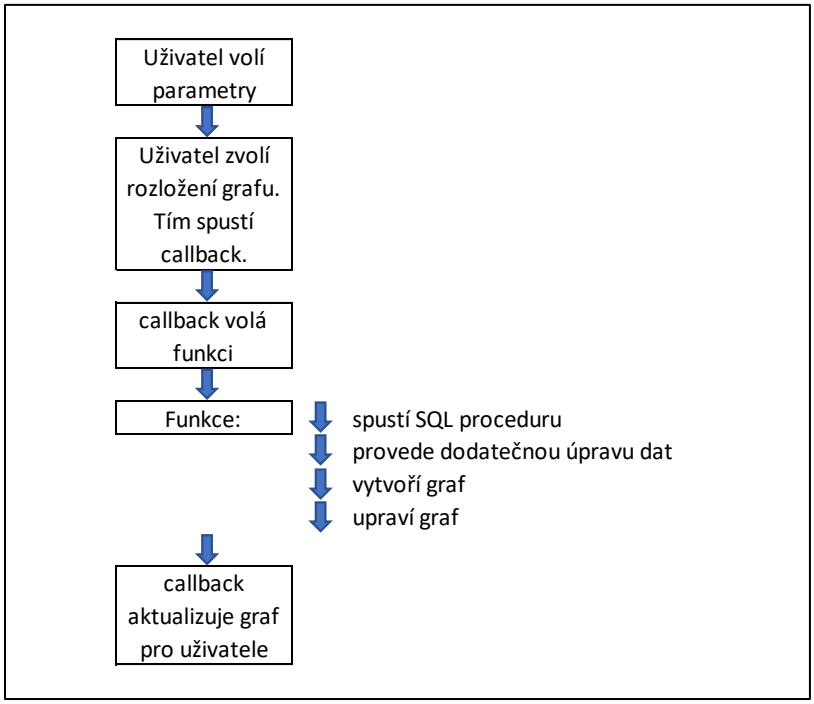
Přístup k datům uloženým v databázi DP, která je uložena na MS SQL serveru, je zajištěn pomocí v databázi uložené sady procedur. Tyto procedury jsou v případě požadavku na získání dat aplikací volány a vrací data filtrovaná uživatelem volenými parametry. Tyto procedury jsem navrhl tak, aby nevznikala zbytečná potřeba pro další transformaci dat v Pythonu. Data jsou tedy filtrována, transformována a agregována na straně SQL serveru a tím je sníženo množství zasílaných dat mezi SQL serverem a Python aplikací na minimum. Seznam vytvořených procedur je zobrazen na obrázku 29.



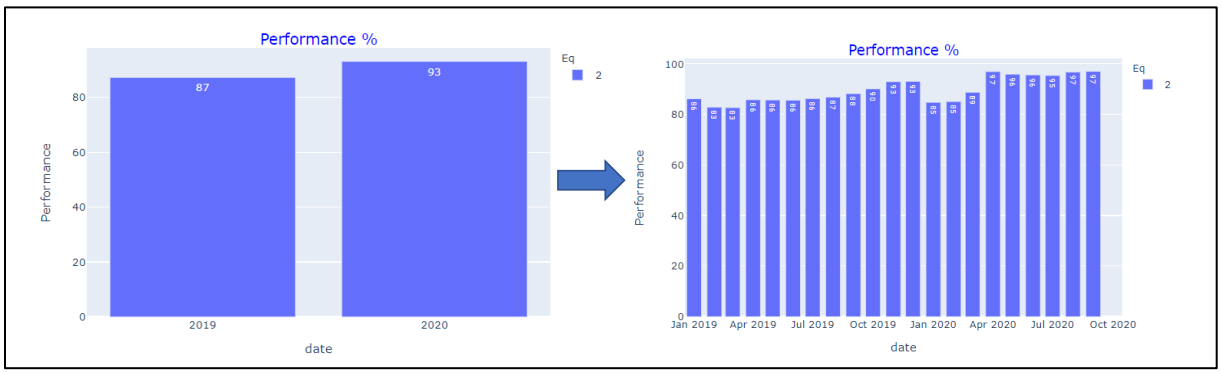
Obrázek 29: Seznam procedur volaných python aplikací

Proceduru *dbo.PredictStops* je možné nalézt v příloze 2. Tato procedura vrací data filtrovaná parametry v aplikaci vstupovanými uživatelem. Tato data jsou následně využita k provedení predikcí, které jsou následně prezentovány uživateli aplikace. Všechny procedury jsou přiloženy k této diplomové práci na CD ve formátu Create. Soubory ve formátu Create slouží k vytvoření těchto procedur na novém serveru.

Nejpodstatnější částí aplikace jsou zpětná volání (anglicky callbacks), která v aplikaci zajišťují přechod mezi jednotlivými stránkami aplikace, transformaci grafů, výpočty predikcí a tvorbu nových komponent. Tato volání jsou spouštěna splněním podmínek, jakými může být například stisknutí tlačítka uživatelem. V diagramu na obrázku 30 je znázorněna funkce zpětného volání na příkladu změny rozložení grafu *Performance %* na první stránce aplikace. Výsledek tohoto zpětného volání je zobrazen na obrázku 31.



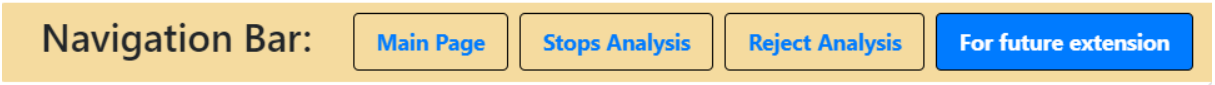
Obrázek 30: Diagram procesu zpětného volání



Obrázek 31: Změna rozložení grafu po spuštění zpětného volání výběrem položky Month v dropdown listu

Pro tuto aplikaci jsem vytvořil celkem 11 těchto zpětných volání, kterými je celá aplikace řízena. Pro tvorbu zpětných volání jsem zvolil několik strategií, a to na základě funkcí, které spouští. Pro vizualizační stránku aplikace, která slouží pouze pro vizualizaci historických dat a obsahuje celkem osm grafů, byla vytvořena jen dvě zpětná volaní. Každé z nich ovládá čtyři grafy. První ze zpětných volání dále při prvním spuštění aplikace zajistí předvyplnění dropdown listů, které uživatelům slouží k ovládání grafů. Jedno zpětné volání zajišťuje přepínání jednotlivých stránek aplikace. Zbylá zpětná volání jsou dedikována pro druhou stránku aplikace, která slouží k provádění predikcí a jejich následného analyzování.

Jeden z důvodů výběru frameworku Dash je možnost tvorby vícestránkových aplikací. Této funkcionality jsem využil pro rozdělení aplikace na části podle jejich účelu. Vytvořil jsem celkem čtyři stránky aplikace, z čehož dvě jsou plně funkční a dvě slouží jako příprava pro její další rozvoj. Kostra aplikace je definována v Python souboru *main.py*. Tato kostra obsahuje nadpis, logo a navigační lištu pro přepínání jednotlivých stránek aplikace. Stiskem tlačítek navigační lišty je spouštěno zpětné volání, které zajistí přepnutí stránky aplikace. Jednotlivé stránky aplikace jsou definovány v samostatných python souborech uložených v balíčku s názvem *apps*. To je znázorněno na obrázku 28. Jednotlivé stránky jsou vytvořeny pomocí knihovny Dash Bootstrap Components, která je postavena na CSS Frameworku Bootstrap. Tato knihovna obsahuje řadu předpřipravených šablon pro komponenty, jako jsou formuláře, tlačítka dropdown listy, nebo formuláře a komponenty rozložení stránky, jako jsou kontejner, řádky a sloupce. Jak již bylo zmíněno, vytvořil jsem celkem čtyři stránky aplikace. Při stisknutí tlačítka *Stops Analysis* na navigační liště je spuštěno zpětné volání, které spouští funkci, jejímž výstupem je soubor obsahující rozložení stránky *Stops Analysis*.

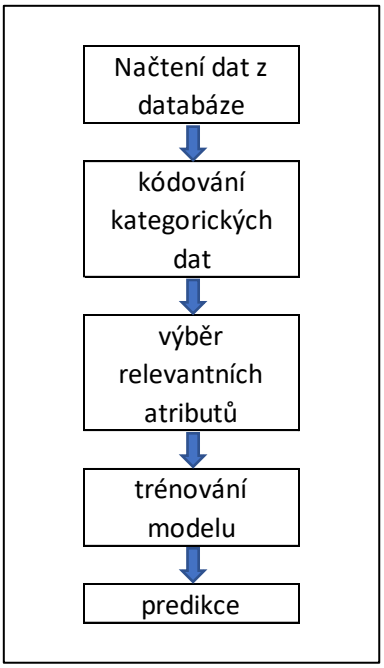


Obrázek 32: Navigační lišta aplikace

Rozložení aplikace je definováno v souboru *main.py*, kde se nachází komponenta kontejner obsahující komponenty nadpisu, logo aplikace a navigační lištu. Dále se v tomto kontejneru nachází další kontejner, který je plněn při volání funkce sloužící k přepínání jednotlivých stránek aplikace. Tato funkce do kontejneru vloží obsah ze zvoleného souboru pro konkrétní stránku aplikace. Pokud uživatel stiskne tlačítko *Stops Analysis* umístěné na navigační liště (obrázek 32), je provedeno zpětné volání, které spustí funkci, jejímž výstupem je požadovaná stránka, v tomto případě je tedy do kontejneru vložen obsah souboru *firstpage.py*. Vzhled stránky je dále upravován CSS souborem s názvem *stylesheet.css*, který jsem vytvořil pro úpravu stylu určitých komponent aplikace. Rozložení projektu s názvy jednotlivých souborů je zobrazeno na obrázku 28.

Nejpodstatnější funkcionalitou aplikace je predikce frekvence krátkých zastávek pro budoucí výrobní zakázky. K tomu jsem za využití knihovny *scikit-learn* vytvořil čtyři prediktivní algoritmy, jejichž výběr je popsán v kapitole 11.1. V této kapitole je dále popsán postup rozdělení historických dat do tříd. Vytvořené algoritmy predikují třídu frekvence výskytu krátkých zastávek. Ty jsou rozděleny do pěti tříd 1 až 5 podle četnosti jejich výskytu. Komunikace aplikace s MS SQL databází je zajištěna pomocí knihoven *pandas*, *pyodbc* a *fast_to_sql*, které zajišťují jak čtení dat, tak i následné ukládání dat predikovaných. Jak již bylo zmíněno, predikce krátkých zastávek je postavena na typu finálního produktu výrobní linky, tedy na použitých materiálech a jejich parametrech. Je zcela jasné, že použité materiály a nejsou jediným faktorem ovlivňující frekvenci krátkých zastávek, ale z výsledků testů popsaných v kapitole 11.1 vyplývá schopnost predikovat krátké zastávky pro nejčastějších 50 typů zastávek s přesností 85 %. Dalšími faktory, které ovlivňují frekvenci krátkých zastávek, mohou být instalace přídavných zařízení, kvalita vstupních materiálů nebo lidský faktor v podobě kvality prováděné pravidelného čištění.

Proces predikce dat zahrnuje několik kroků, jež jsem do aplikace zakomponoval. Tyto kroky jsou znázorněny na obrázku 33.



Obrázek 33: Proces predikce krátkých zastávek

Prvním krokem je načtení dat z databáze. Pro tento krok jsem připravil několik SQL procedur, které zajišťují selekci požadovaných dat v databázi na základě uživatelem definovaných filtrů. Dalším krokem je kódování kategorických dat. Se všemi daty využívanými k predikci jsem zacházel jako s kategorickými, a to i s fyzikálními parametry materiálů, jakými jsou hmotnost nebo hustota materiálů. K tomuto kroku jsem se rozhodl po analýze, ze které sice vyplynul vliv těchto parametrů na počet

zastávek, ale tento vliv nevykazoval korelaci. Pro kódování dat jsem využil funkce *get_dummies* z knihovny pandas. Alternativou k tomuto řešení může být OneHotEncoder z knihovny scikit-learn, který jsem také otestoval. Zakódováním použitých dat je 72 využitých materiálových atributů převedeno na více než 1000 booleovských sloupců. Vzhledem k tomu, že materiál, který do výrobní linky vstupuje až v rámci třetího stroje, nemůže ovlivnit frekvenci krátkých zastávek na stroji prvním, bylo nutné zajistit výběr vhodných sloupců pro každý typ zastávky. To by však při počtu 310 zastávek bylo časově náročné.

První uvažovanou variantou bylo umožnit výběr materiálů pro predikci uživateli aplikace. Ten by ze 72 vstupních materiálů před každou analýzou vybral ty, které mají potenciál počet zastávek ovlivnit. To by ale zkomplikovalo celý proces predikce a bylo uživatelsky nevhodným řešením. Druhou uvažovanou variantou bylo rozdělit materiálové atributy do skupin podle toho, na jakém místě do výrobní linky vstupují. Problém tohoto řešení ale spatřuji například v tom, že materiály vstupované do výrobní linky na třetím stroji nemusí ovlivňovat typy všechny typy zastávek, které se vyskytují na tomto zařízení. Důvodem je to, že materiál do stroje nemusí vstupovat na jeho začátku, ale může být do stroje zaváděn v jeho jiných částech. Tento postup by dále vyžadoval další konzultace s procesními inženýry. Třetím a zvoleným řešením je využití třídy *SelectKBest* z knihovny *scikit-learn*. Ta zajistí ohodnocení jednotlivých atributů podle jejich vlivu na predikovanou proměnnou na základě zvolené hodnotící funkce *chi2*. Pro parametr *K,* kterým je volem počet nejvhodnějších parametrů, jsem zvolil hodnotu 500 jako konstantní. Tuto hodnotu jsem zvolil metodou postupného navyšovaní parametrů, při kterém jsem pozoroval nárust predikční přesnosti. Při dalším navyšování počtu parametrů zůstávala přesnost predikcí konstantní. Mezi další benefity využití třídy *SelectKBest* se řadí například snížení pravděpodobnosti přeučení prediktivního modelu nebo snížení trénovacího času prediktivních algoritmů. Po vyřešení načítání dat, jejich kódování a výběru vhodných parametrů byly do aplikace zakomponovány vybrané prediktivní algoritmy z kapitoly 11.1.

Aplikace obsahuje dvě prediktivní funkcionality. První funkcionalita nabízí uživateli možnost hodnocení prediktivních algoritmů na základě jejich prediktivní přesnosti. Uživatelem jsou zadány vstupní parametry, kterými jsou zvolené zařízení a typ zastávky. Aplikace provede rozdělení historických dat v poměru 80/20 na trénovací sadu a testovací sadu. Trénovací sada je využita k natrénování všech čtyřech prediktivních modelů (MLP, RFC, SVC a KNN) a následně jsou modely otestovány na sadě trénovací. Uživateli aplikace jsou následně prezentovány výsledky ve formě grafů i vypočtené přesnosti modelu. Zároveň je provedena predikce budoucích výrobních zakázek, jejíž výsledky jsou prezentovány v další sadě grafů. Uživatel má následně možnost porovnat výsledky jednotlivých predikčních algoritmů a zvolit ten nejvhodnější. Zároveň je tímto porovnáním výsledků všech algoritmů možno odhadnout relevanci predikce. Druhou funkcionalitou je výběr metody pro

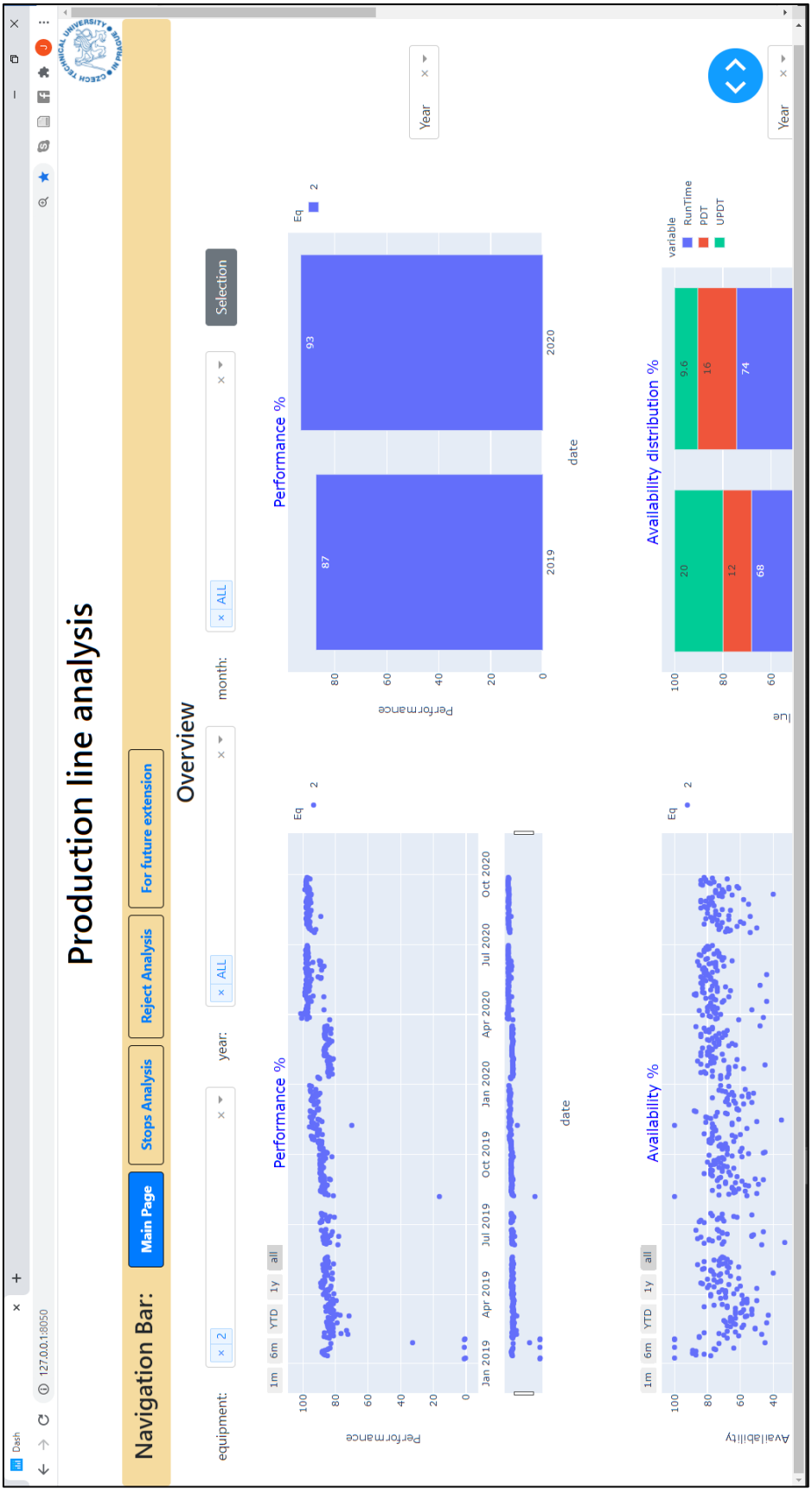
uložení predikovaných dat do databáze pro jejich další využití, jakým může být například analýza materiálů ovlivňující frekvenci krátkých zastávek. Uživatel aplikace zvolí konkrétní predikční metodu, která je využita pro predikci krátkých zastávek budoucí výroby a tlačítkem update ji uloží do databáze. Pro tuto finální predikci již nejsou data rozdělována na trénovací a testovací sady, ale je využito všech historických dat pro trénování algoritmu, což by mělo vést k lepším predikčním výsledkům. Prvním z využitých algoritmů je *Multi Layer Percoptron* se třemi skrytými vrstvami. Obsahuje celkem 90 neuronů. Hyperbolický tangens byl zvolen jako aktivační funkce a pro výpočet byl zvolen optimalizační algoritmus L-BFGS. Pro klasifikační algoritmus Náhodný les byla zvolena jeho maximální hloubka 30 a s 200 zvolenými stromy. Pro algoritmus *Support Vector Machine* byl jako nejvhodnější zvolen lineární kernel. Pro metodu *K-Nearest Neighbor* byl zvolen počet nejbližších sousedů rovný čtyřem. Toto jsou hlavní parametry algoritmů, které byly postupně laděny a testovány na základě informací získaných z dokumentace knihovny *scikit-learn* [66] a literatury [21, 29, 33].

Poslední částí, kterou bylo nutné naprogramovat, byla část umožňující tvorbu grafů pro vizualizaci predikcí, což by mělo uživatelům aplikace umožnit další analýzy predikovaných dat. Vzhledem k existenci velkého množství různých kombinací dat, které by uživatel mohl potenciálně požadovat po aplikaci, bylo nutné vymyslet způsob, který by umožnil jejich prezentaci uživateli ve formě, kterou požaduje. Z tohoto důvodu jsem tuto část aplikace naprogramoval tak, aby si uživatel do aplikace mohl přidat libovolný počet grafů, které by prezentovaly jím zvolená data v požadovaném pořadí. Do aplikace bylo přidáno tlačítko, které spouští zpětné volání. Toto zpětné volání následně do rozložení stránky přidá dva nové grafy včetně čtyř vstupních polí. Ta slouží pro výběr dat pro dané grafy. Tímto způsobem si uživatel může vytvořit sadu grafů, které následně využije pro analýzu konkrétního typu zastávky.

Přehled jednotlivých stránek aplikace bude popsán v následující kapitole. Celá aplikace je přiložena k této diplomové práci na CD včetně vytvořených SQL procedur a SQL databáze. Dále je na CD přiložen .txt soubor, který obsahuje seznam kroků nutných pro spuštění aplikace.

11.3 Uživatelské rozhraní

Aplikace, včetně všech jejích funkcionalit, byla otestována ve třech různých prohlížečích a také na mobilním telefonu s dotykovým displejem. Během tohoto testování jsem neodhalil žádné problémy. Jak již bylo zmíněno, aplikace obsahuje dvě funkční stránky a dvě stránky připraveny pro její další rozvoj, který je diskutován v kapitole 14. První stránka obsahuje sadu osmi interaktivních grafů, které jsou uživatelem ovládány pomocí filtrů v podobě dropdown listů. Tyto grafy vizualizují tři části OEE ukazatelem, tedy výkon, dostupnost a kvalitu. Dále byl vytvořen graf pro sledování ukazatele MTBF. Obrázek 34 prezentuje vrchní část této úvodní stránky aplikace.

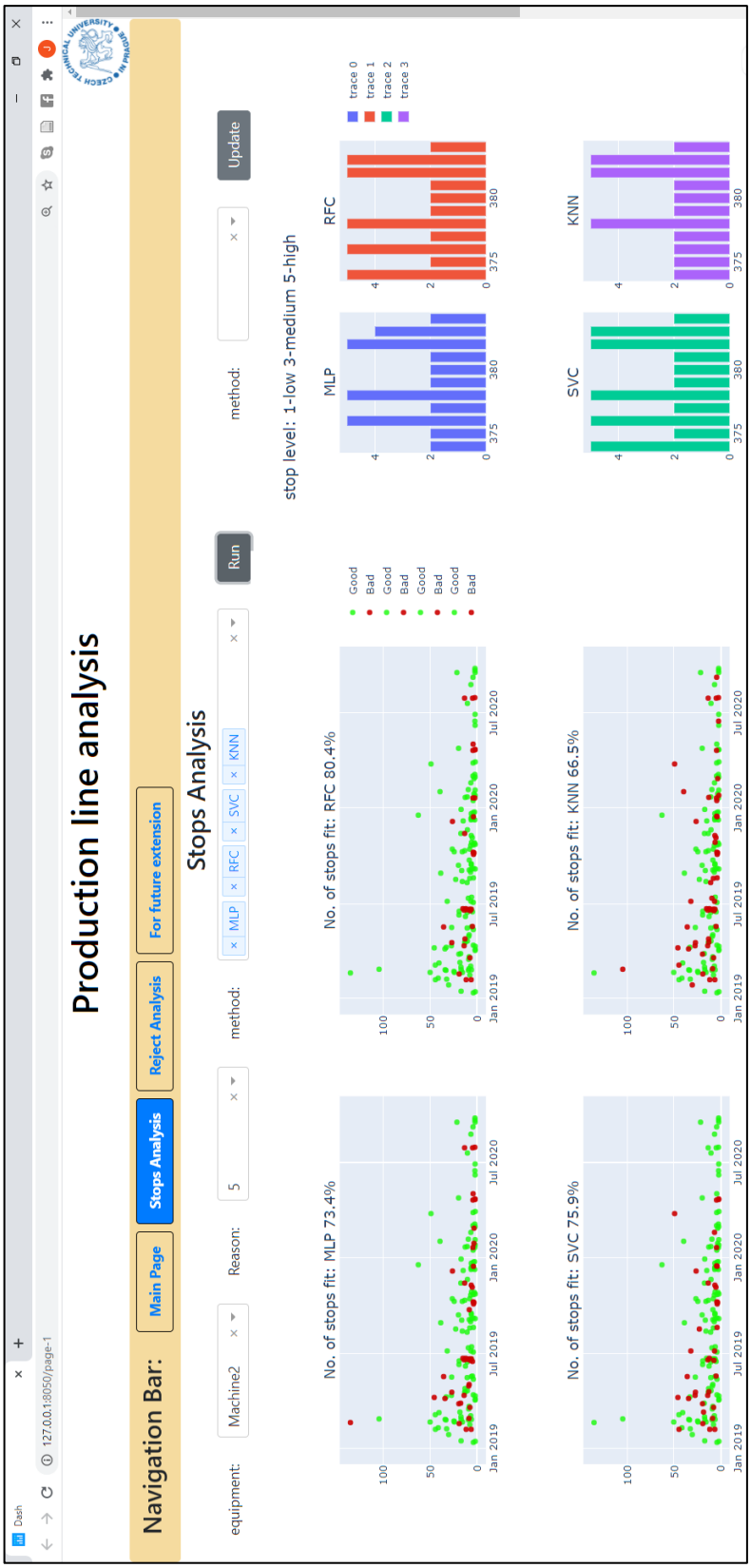


Obrázek 34: Ukázka vrchní části úvodní stránky aplikace

Úvodní stránka aplikace obsahuje nadpis aplikace, logo a navigační lištu, které jsou společné pro všechny stránky aplikace. Navigační lišta byla nastavena tak, aby byla na obrazovce vždy přítomna, při posunutí obrazovky směrem dolů zůstane lišta kvůli uživatelské pohodlnosti připnutá k horní části obrazovky. Pod touto lištou se nachází sada tří filtrů, kterými jsou ovládány grafy pod nimi. Pomocí těchto filtrů je možné nastavit stroj, popřípadě výběr více strojů z výrobní linky, rok, pro který chce uživatel data vizualizovat a dále je možné vybrat konkrétní měsíce. Na obrázku 34 je filtrován druhý stroj výrobní linky pro všechny dostupné roky a měsíce. Stisknutím tlačítka Selection se provede přenastavení grafů podle nastavených filtrů. Pod těmito grafy se nachází sada osmi grafů – Grafy v levé části slouží pro vizualizaci výsledků jednotlivých směn. Obsahuje tedy dva výsledky pro každý den, kdy byla výrobní linka v provozu. Tyto grafy jsou dále opatřeny sadou filtrů pro výběr konkrétnějšího období. Je možné volit poslední měsíc, půlrok, hodnoty YTD, tedy hodnoty od začátku roku k dnešnímu dni, a hodnoty za poslední rok. Další funkcí, kterou jsou tyto grafy vybaveny, je posuvná lišta, kterou lze vybrat přesné časové rozmezí. Ta se nachází přímo pod grafem. Sada čtyř grafů v pravé části aplikace slouží pro doplnění grafů nalevo. Z obrázku 34 je patrné, že první dva grafy vizualizují stejný ukazatel, kterým je výkon stroje. Pravý graf ukazuje agregované výsledky za vybrané období, které je možné vybrat pomocí filtru položeného vpravo od něj. Lze vybírat agregaci po letech, měsících, týdnech a dnech. Graf se automaticky přenastaví při změně tohoto filtru. Obdobně jsou řešené i další grafy, které vizualizují ukazatele dostupnosti, kvality a průměrný čas mezi zastávkami vybraného stroje. Graf pro ukazatel dostupnosti byl dále doplněn o možnost zobrazovat data jednotlivých ztrát na dostupnosti, tedy plánované zastávky a neplánované zastávky. Je tedy možné pomocí filtrů zvolit více strojů výrobní linky a porovnávat ztráty na dostupnosti u jednotlivých zařízení. Z toho je možné vyvodit například zjištění, který stroj vykazuje největší ztrátu a snižuje tím OEE ukazatel celé výrobní linky. Obdobně je možné porovnat i ukazatel kvality, kde lze porovnávat ztrátu na neshodných výrobcích po jednotlivých zařízeních a pomocí toho určit priority pro další údržbu.

Tato úvodní stránka aplikace by měla sloužit jako nástroj pro sledování trendů jednotlivých ukazatelů, a to nejen týmu údržby a procesním inženýrům, ale také širšímu okruhu pracovníků, kteří zajišťují chod této výrobní linky, popřípadě pracovníkům na vedoucích pozicích, kteří potřebují tyto trendy také sledovat. Dále může tato stránka sloužit jako vhodný nástroj pro validaci prováděných zásahů na výrobní lince. Po provedení zásahu je možné na denní bázi sledovat další vývoj a validovat relevantnost provedeného zásahu, kterým může být například výměna náhradního dílu nebo například změna nastavení této výrobní linky.

Na obrázku 35 je zobrazena první část druhé strany této aplikace. Ta je zaměřena na provádění predikcí pro jednotlivé typy zastávek a jejich následné ukládání do databáze.



Obrázek 35: Ukázka druhé stránky aplikace, zaměřená na predikci krátký zastávek

Z obrázku 35 je patrné, že při přepnutí stránky je v navigačním listu aktivní stránka podbarvena modře. Na obrázku je zobrazena pouze první část stránky, která zajišťuje interakci s prediktivními algoritmy. V levé části obrazovky pod navigační lištou je sada tří filtrů, které umožňují výběr stroje výrobní linky, typ zastávky, který chce uživatel predikovat a filtr sloužící pro výběr prediktivních algoritmů, jež chce k predikci použít. Tlačítkem Run se spouští predikční algoritmy a navrácené výsledky přenastaví všech osm grafů. Čtyři grafy v levé části obrazovky zobrazují výsledky predikce prováděné na testových datech, která byla vyčleněna z historických dat v poměru 80/20. 80 % dat bylo využito pro trénování algoritmů a 20 % na následné testování. Výsledky tohoto testování jsou zobrazeny na těchto čtyřech grafech v jejichž nadpisu je uvedena predikční přesnost. Na ose x těchto grafů je datum, na ose y je počet denních stopů pro vybraný typ krátké zastávky. Tento typ zastávky je označen číslem 5 z důvodu již zmíněné nutnosti data anonymizovat. Zeleně jsou označeny správně určené body, červené znázorňují body predikčním algoritmem určené chybně. Z grafů vyplývá, že nejvyšší predikční přesnost vykazuje RFC algoritmus s úspěšností predikce 80.4 %. Ostatní metody vykazují predikční přesnost nižší. Tato predikční přesnost se u jednotlivých typů zastávek liší. Právě to je důvodem využití více algoritmů a možnosti jejich porovnání. Čtyři grafy, které tvoří pravou část této stránky aplikace, zobrazují predikce pro 11 budoucích výrobních zakázek, které jsou na tuto výrobní linku zaplánované. Predikce je provedena na stejných predikčních modelech, které byly použity pro grafy v levé části aplikace, tedy pouze na 80 % historických dat. Tyto predikce ohodnotily jednotlivé výrobní zakázky do pěti skupin podle predikované frekvence zastávek, kde stupeň jedna značí extrémně nízký počet zastávek a stupeň 5 extrémně vysoký stupeň zastávek. Stupeň 3 označuje průměr a stupně 2 a 4 lehký výkyv oproti průměru. Z obrázku je patrné, že ne všechny zakázky byly klasifikovány stejně všemi algoritmy. Například první zakázka s číslem 374 je dvakrát klasifikována do stupně 5 a dvakrát do stupně 2. V tomto případě není jistota predikce příliš přesvědčivá a je na uživateli, pro jaký algoritmus se rozhodne. Naopak ostatní zakázky jsou všemi algoritmy kromě KNN hodnoceny stejně. Jediným dalším rozdílem je zakázka číslo 383, která byla MLP algoritmem klasifikována do stupně 4, tedy vyšší počet zastávek, než je průměr a ostatními algoritmy do stupně 5. Po takovéto analýze uživatel zvolí prediktivní metodu, kterou chce použít a zadá ji do filtru vedle tlačítka Update. Tímto tlačítkem následně potvrdí uložení výsledků pro daný algoritmus do databáze. Tím se spustí další zpětné volání, které provede trénování modelu znovu, nyní již na celém vzorku historických dat a provede novou predikci počtu krátkých zastávek pro vybraný stroj a typ zastávky. Tyto výsledky jsou následně uloženy do databáze. V případě již existujících predikcí pro tuto kombinaci vstupních parametrů jsou tato data pouze upravena. Funkce na ukládání dat je ošetřena proti vzniku duplicit.

Na obrázku 36 je zobrazena druhá část této stránky aplikace, která obsahuje možnost tvorby neomezeného množství grafů pro další analýzy.

Obrázek 36: Ukázka druhé strany aplikace s funkcionalitou přidávání grafů

Na obrázku z předchozí strany je znázorněna část druhé strany aplikace, která je uzpůsobena analýze predikovaných dat v závislosti na materiálech používaných při výrobě finálního produktu. V levé horní části je tlačítko *Add Charts* umožňující uživateli vytvořit novou dvojici grafů a filtrů. Pomocí filtrů uživatel zvolí požadované parametry, které chce analyzovat. V tomto případě je zvolen materiál m, typ zastávky 19, zakázka číslo 382 a stroj číslo 2. Z důvodu nutnosti anonymizace dat byly použity tyto kódy. Pro představu funkce je tento případ možné popsat následovně. Zvoleným parametrem materiálu je hmotnost čtverečního metru obalu produktu a typem zastávky je zastávka s názvem *Ucpání při předávce produktu mezi druhým a třetím strojem*. V nadpisu levého grafu je uvedeno vybrané číslo zakázky 382 a vybraný kód materiálu 4. Kódy materiálu by v případě neanonymizovaných dat mohly vypadat následovně: 1-150g/m^2, 2-155 g/m^2, 3-160 g/m^2 a zvolený kód 4-165 g/m^2. V grafu jsou data vyfiltrovaných krátkých zastávek typu 19 vizualizována pomocí krabicového grafu. V druhém grafu je přehled provedené predikce pro jednotlivé plánované výrobní zakázky. Z grafů vyplývá, že vybranou zakázku číslo 382 byl predikován extrémně vysoký počet zastávek typu 19. Dále vyplývá, že je v rámci této zakázky bude použit obalový materiál (kategorie m) s hmotností 165 g/m^2, který při historických produkcích vykazoval v průměru vyšší počet zastávek typu 19. Se znalostí této informace je možné na výrobní lince provést zásah, který by tomuto zvýšenému množství zastávek mohl předejít. *Uvedené názvy materiálu, typu zastávky a kategorie materiálu jsou pouze pro ilustrativní účely a neshodují se se skutečnými parametry.

Tato část aplikace by měla sloužit především procesním inženýrům a týmu údržby, kteří si mohou vytvořit neomezené množství grafů a provádět další analýzy, na základě kterých naplánují případné zásahy na výrobní lince, jimiž mohou snížit počet krátkých zastávek.

12 PowerBI report

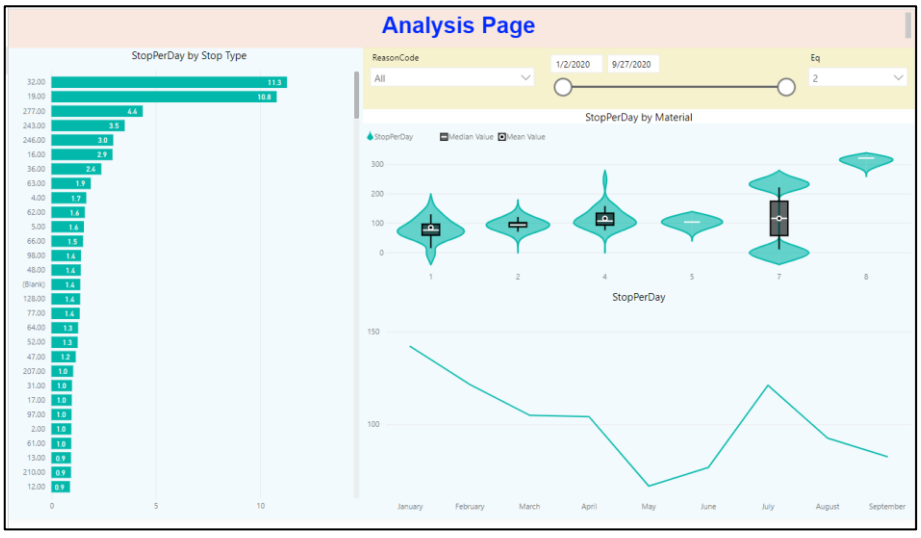
Druhým zvoleným způsobem pro prezentaci dat koncovým uživatelům je využití BI nástroje PowerBI od společnosti Microsoft. Tento nástroj společně s jeho obdobami od jiných společností je popsán v kapitole 6. Před finálním výběrem PowerBI jsem dále otestoval BI nástroj Tableau, který sice nabízí stejné funkce, ale jeho ovládací funkce nejsou tak přímočaré jako u PowerBI a zorientovat se v tomto prostředí mi zabralo více času. Další výhodou PowerBI je přehlednější dokumentace a silná uživatelská základna, která zajišťuje mnoho dalších kvalitních materiálů.

Reporty, které jsem v PowerBI vytvořil, slouží pro vizualizaci jednotlivých sledovaných výrobních ukazatelů a zároveň pro vizualizaci predikcí tvořených aplikací, která je popsaná v předchozích kapitolách. Narozdíl od přehledů, které poskytuje vytvořená aplikace, jsou tyto předem nadefinované a manipulovat s nimi nelze tak rozsáhle, jako v případě vytvořené aplikace. Reporty

vytvořené v *PowerBI* by měly poskytovat přehledové informace zaměstnancům, kteří nepotřebují tak detailní data jako například procesní inženýři. Těmito zaměstnanci mohou být lidé z podpůrných oddělení, jakými jsou oddělení plánování nebo například vedoucí pracovníci, kteří nejsou zaměřeni na takovou míru detailu.

K tvorbě těchto repotů jsem vytvořil dotazy do databáze *DP*, pomocí kterých jsou z databáze vytěžena data pro tvorbu přehledů. V případě *PowerBI* nebyl kladen takový důraz na minimalizaci počtu přijímaných dat jako v případě vytvářené aplikace. Důvodem je možnost v *PowerBI* zvolit mezi několika způsoby připojení dat z *MS SQL* databáze. Základní způsoby jsou přímé dotazy. V tomto případě je databáze dotazována při každé interakci s reportem, obdobně jako v případě vytvořené aplikace. Druhým způsobem je import dat s možností nastavení pravidelné aktualizace. Vzhledem k tomu, že pro tyto reporty není potřeba nejaktuálnějších dat, byl zvolen právě způsob importu dat s aktualizací po 24 hodinách. V případě, že by aktualizace trvala příliš dlouho z důvodu příliš velkého množství dat, existuje i možnost inkrementální aktualizace, kdy jsou aktualizovány pouze nové záznamy. Takto lze nastavit například aktualizaci dat pouze za posledních 24 hodin a starší data načítána nejsou. Tento způsob je však možný pouze ve verzi Premium, která je poměrně nákladnou záležitostí.

V *PowerBI* jsem vytvořil celkem dva reporty. První obsahuje přehledy kritických ukazatelů pro možnost sledování vývoje OEE a jeho podsložek. Zároveň report obsahuje filtry pro výběr konkrétních pohledů. Pro každý graf je dále možné upravovat pohled podle požadované granularity dat. Druhým reportem jsou přehledy počtu stopů za den, které je možné získat po strojích, typech zastávek a je možné je dále filtrovat po materiálových skupinách. Na obrázku 37 je ukázka vytvořeného *PowerBI* reportu. Tento report je zároveň přiložen na CD k této diplomové práci.



Obrázek 37: Ukázka jednoho z vytvořených reportů v PowerBI

Náročnost práce v softwaru *PowerBI* je mnohem nižší než v případě tvorby aplikace využitím Pythonu a pro základní funkce nevyžaduje znalost žádného z programovacích jazyků. Zároveň je tento nástroj pravidelně aktualizován o další funkce, čímž roste i jeho flexibilita při vytváření reportů. Další výhodou oproti tvorbě vlastních aplikací je robustnost systému, který není nutné nijak udržovat. To vše však za cenu měsíčních poplatků, které se v případě Premium verze pohybují v řádech tisíců dolarů.

13 Práce s aplikací

Aplikace by měla sloužit jak pro jednorázové analýzy kritických ukazatelů nebo vlivu materiálů na frekvenci krátkých zastávek, tak pro pravidelnou analýzu a predikci budoucích výrobních zakázek. Pro jednorázové analýzy není nutné vytvářet žádná doporučení, jediným bodem, který je nutné zajistit, je proškolení zaměstnanců, kteří k aplikaci budou mít přístup. To je nutné k zajištění jejího efektivního využívání. Pro pravidelné analýzy je však nutné vytvořit alespoň částečné doporučení pro její používání, což povede ke standardizaci procesu a zajistí její správné využití. Tento proces bude pravděpodobně nutné dále upravit podle potřeb a komentářů týmu údržby a procesních techniků, ale základním návrhem je využívání aplikace ve dvou paralelních módech. Prvním je týdenní důkladná analýza. Při této analýze by v prvním kroku měla proběhnout analýza kritických ukazatelů za předchozí týden. Ta by měla přinést první vhled do vývoje kondice jednotlivých strojů výrobní linky. Druhým krokem je provedení predikcí pro nejčastější zastávky každého stroje výrobní linky. Následovat by měla nejnáročnější část, kterou je příprava akčního plánu pro výrobní zakázky, pro něž byly predikovány vysoké frekvence krátkých zastávek. Pro každý typ takto predikované zastávky by měla být provedena analýza materiálů, která odhalí materiál nebo skupinu materiálů, které tento stav způsobují. To je v aplikaci umožněno tvorbou neomezeného počtu grafů, které porovnávají četnost zastávek podle nastavených filtrů. Po identifikaci těchto kritických materiálů musí vzniknout plán pro jejich odstranění, který musí být aplikován během přípravy stroje na tuto konkrétní výrobní zakázku. Denní analýza by následně měla sloužit k validaci provedených nápravných akcí a v případě nepříznivého vývoje by měla spustit přípravu nových akcí.

Tento proces by měl být podpořen využitím připravených powerBI reportů, které budou dále doplňovány o další informace na základě požadavků týmu údržby a procesních inženýrů.

14 Další vývoj

Jak bylo zmíněno na začátku praktické části této diplomové práce, jedná se o první verzi aplikace, která by měla ověřit možnost využití prediktivních algoritmů na výrobní linky v potravinářském průmyslu. Podle dosažených prediktivních výsledků popsaných v kapitole 11 je velice pravděpodobné, že aplikace, kterou jsem vytvořil, užitečnost nasazení prediktivních algoritmů prokáže

a její vývoj bude dále pokračovat. K tomu je však nutné danou aplikaci otestovat v reálném provozu s daty, která nejsou anonymizovaná.

V případě úspěšné validace vytvořené aplikace je naplánováno několik oblastí, které do aplikace budou zakomponovány. První oblastí je predikce méně frekventovaných krátkých zastávek. Pro ty bude nutné vyvinout jiný postup predikce. Vzhledem ke skutečnosti, že se tyto zastávky vyskytují jen výjimečně, bude nutné upravit algoritmus predikce na jejich rozdělení pouze do dvou tříd namísto nyní využívaných pěti. Algoritmus by tedy predikoval pouze výskyt tohoto typu zastávky nebo jeho negaci. Další oblastí, o kterou bude nutné tuto aplikaci rozšířit, je predikce úrovně neshodných produktů. V této oblasti předpokládám vliv použitých materiálů mnohem výraznější než u predikovaných zastávek, a z tohoto důvodu by predikce mohla vykazovat přesnost kolem 90 %. To však bude nutné dále otestovat. Jednou z funkcionalit, kterou by bylo vhodné do aplikace přidat, je zakomponování CRUD (create, read, update, delete) tabulky, která by sloužila pro ukládání úkolů a poznámek během přípravy analýz. Uživatel by během analýzy mohl vytvořit vlastní seznam úkolů, který by byl aplikací uložen do databáze a následně skrze *PowerBI* report distribuován cílovým uživatelům, kterými by v tomto případě mohli být například mechanici nebo operátoři výrobní linky. Tímto způsobem by bylo možné vytvořit seznam akcí, které by následně tyto pověřené osoby splnily. Další vývoj bude probíhat v oblasti P*owerBI* reportů, které budou rozšiřovány společně s novými funkcionalitami aplikace.

15 Závěr

Cílem této diplomové práce bylo navrhnout systém pro monitorování průmyslové výroby s cílem aplikace prediktivní údržby výrobních linek. Prvním krokem ke splnění tohoto cíle bylo provedení rešerše oblasti prediktivní údržby průmyslových zařízení. V rámci provedené rešerše jsem zpracoval několik témat souvisejících s návrhem systému prediktivní údržby. V první části rešerše jsem se zabýval různými přístupy údržby průmyslových zařízení. Následně jsem podrobněji popsal koncept *totálně produktivní údržby* včetně běžně využívaných výkonnostních ukazatelů, které slouží k validaci a monitoringu výsledků prediktivní údržby. V následující části rešerše jsem se zabýval datovou analýzou, a to především vhodnými nástroji pro její aplikaci, algoritmy strojového učení a možností jejich využití v oblasti prediktivní údržby. Možnost využití strojového učení v oblasti údržby jsem zkoumal převážně na základě odborných článků popisujících využití strojového učení v oblasti údržby průmyslových zařízení. Nedílnou součástí prediktivních systémů jsou datová úložiště, kterými jsem se v teoretické části práce také zabýval. Poslední částí provedené rešerše bylo zhodnocení aktuálně rozšířených nástrojů sloužících k vizualizaci dat. Provedením této rešerše jsem získal kvalitní teoretický

základ informací pro tvorbu prediktivního systému a zároveň tím splnil druhý bod zadání této diplomové práce.

Po provedení rešerše jsem se věnoval návrhu prediktivního nástroje, který by měl sloužit týmu údržby k včasnému odhalení problémů na výrobní lince, a to za využití prediktivních algoritmů. Na poskytnutých datech z výrobní linky jsem provedl analýzu, kterou byla identifikována oblast krátkých zastávek jako oblast s nejvyšším potenciálem pro zvýšení efektivity výroby. Na základě provedené analýzy, rešerše a rozhovorů s týmem údržby dané výrobní linky jsem se rozhodl nástroj na predikci krátkých zastávek realizovat ve formě webové aplikace napsané v Pythonu. Jako nejvhodnější řešení pro ukládání dat jsem zvolil MS SQL server. Pro implementaci prediktivních algoritmů jsem využil Python knihovnu scikit-learn, pomocí které jsem vytvořil šest různých prediktivních algoritmů, z nichž čtyři dosahovaly predikční přesnosti přes 80 %. Tyto jsem zvolil pro další implementaci do webové aplikace. Poslední částí návrhu byl výběr vhodného vizualizačního softwaru, kterým byl zvolen nástroj PowerBI. Ten v navrhovaném systému slouží pro vizualizaci historických i predikovaných dat a jeho výstupy by měly být poskytnuty širšímu publiku v rámci firmy. Tímto provedením návrhu systému jsem splnil třetí bod zadání této práce.

K realizaci prediktivního nástroje jsem využil Python framework Dash pomocí kterého jsem vytvořil vícestránkovou webovou aplikaci, která splňuje jak funkci predikční, tak umožňuje provádění analýzy dat. Aplikace využívá čtyř prediktivních algoritmů pro predikci frekvence krátkých zastávek u budoucích výrobních zakázek. Výsledky těchto predikcí jsou v aplikaci prezentovány uživateli, který následně zvolí vhodný algoritmus. Predikovaná data jsou ukládána do MS SQL databáze, odkud jsou dále využívána pro jejich vizualizaci. V aplikaci je dále možné provádět podrobné analýzy jednotlivých typů zastávek výrobní linky a také sledovat přehledy základních ukazatelů dané výrobní linky. Tato aplikace je navržena pro tvorbu predikcí a jejich následnou podrobnou analýzu a měli by ji využívat převážně procesní inženýři a tým údržby. Pro širší okruh uživatelů jsem vytvořil přehledové reporty v softwaru PowerBI. Tyto reporty poskytují přehledy o historických vývojích výrobní linky a zároveň jsou v nich prezentovány predikce, které byly vytvořeny skrze webovou aplikaci. Aplikace byla realizována na anonymizovaných datech z výrobní linky v potravinářském průmyslu. Tvorbou prediktivní webové aplikace a dalším doplněním tohoto řešení o přehledové reporty v softwaru PowerBI jsem splnil čtvrtý bod zadání této diplomové práce, čímž byly naplněny všechny body zadání.

Po nasazení této aplikace do praxe bude její funkce a užitečnost validována. V případě kladných výsledků bude dále rozšířena o další prediktivní funkcionality, a to především v oblasti kvality výroby.

Seznam zdrojů

[1] R. K. Mobley, *An Introduction to Predictive Maintenance*. Oxford: Elsevier Science & Technology, 2002.

[2] R. Peimbert-García, J. Limón-Robles, M. Beruvides, a D. García-Hernández, *An Economic Framework for total productive maintenance (TPM)*. 2012, s. 2769.

[3] „Types of Maintenance: The 9 Different Strategies Explained", *ROAD to RELIABILITY™*. https://www.roadtoreliability.com/types-of-maintenance/ (viděno lis. 01, 2020).

[4] „What Are The 4 Types of Maintenance Strategies?", *Fiix*, úno. 01, 2019. https://www.fiixsoftware.com/blog/evaluating-maintenance-strategies-select-model-asset-management/ (viděno lis. 01, 2020).

[5] Y. Wang, G. Liang, a N. Wang, „Optimal Preventive Maintenance Strategy Based on Risk Assessment", in *2011 International Conference on Computer and Management (CAMAN)*, kvě. 2011, s. 1–4, doi: 10.1109/CAMAN.2011.5778894.

[6] „TPM, Total Productive Maintenance and its role in OEE, MTBF, and MTTR", *Six-Sigma-Material.com*. https://www.six-sigma-material.com/TPM.html (viděno lis. 02, 2020).

[7] „TPM Improves Equipment Effectiveness | Lean Production". https://www.leanproduction.com/tpm.html (viděno pro. 20, 2020).

[8] „OEE = celková efektivnost zařízení a výroby | Automatizace.HW.cz". https://automatizace.hw.cz/oee-celkova-efektivnost-zarizeni-a-vyroby.html (viděno pro. 20, 2020).

[9] „MTBF (Mean Time Between Failure)", *Six-Sigma-Material.com*. https://www.six-sigma-material.com/MTBF.html (viděno pro. 20, 2020).

[10] J. Dalzochio *et al.*, „Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges", *Comput. Ind.*, roč. 123, s. 103298, pro. 2020, doi: 10.1016/j.compind.2020.103298.

[11] S. Cunningham, „Machine Learning and Statistics: A matter of perspective", kvě. 1995.

[12] „Power Pivot: Powerful data analysis and data modeling in Excel". https://support.microsoft.com/en-us/office/power-pivot-powerful-data-analysis-and-data-modeling-in-excel-a9c2c6e2-cc49-4976-a7d7-40896795d045 (viděno pro. 20, 2020).

[13] „Power Query - Overview and Learning". https://support.microsoft.com/en-us/office/power-query-overview-and-learning-ed614c81-4b00-4291-bd3a-55d80767f81d (viděno pro. 20, 2020).

[14] „Welcome to Python.org", *Python.org*. https://www.python.org/ (viděno pro. 20, 2020).

[15] A. Choudhury, „Top 12 R Packages For Machine Learning In 2020", *Analytics India Magazine*, čer. 09, 2020. https://analyticsindiamag.com/top-12-r-packages-for-machine-learning-in-2020/ (viděno pro. 20, 2020).

[16] „3 Benefits of Machine Learning for Microsoft Azure", *Sherweb*, úno. 17, 2018. https://www.sherweb.com/blog/cloud-server/benefits-machine-learning-microsoft-azure/ (viděno pro. 20, 2020).

[17] „Azure Machine Learning | Microsoft Azure". https://azure.microsoft.com/en-us/services/machine-learning/ (viděno pro. 20, 2020).

[18] „Pricing - Machine Learning | Microsoft Azure". https://azure.microsoft.com/en-us/pricing/details/machine-learning/ (viděno pro. 20, 2020).

[19] „Machine Learning: What it is and why it matters". https://www.sas.com/en_us/insights/analytics/machine-learning.html (viděno pro. 20, 2020).

[20] J. Grus, *Data science from scratch: first principles with Python*, Second. Beijing;Sebastopol;Tokyo;Farnham;Boston; O'Reilly, 2019.

[21] S. Raschka a V. Mirjalili, *Python Machine Learning*, 2. vyd. Packt Publishing, 2017.

[22] R. Shaw, „The Top 10 Machine Learning Algorithms for ML Beginners", *Dataquest*, čer. 26, 2019. https://www.dataquest.io/blog/top-10-machine-learning-algorithms-for-beginners/ (viděno lis. 06, 2020).

[23] „13+ List of Machine Learning Algorithms with Details [2018 Updated]", *New Tech Dojo*, bře. 06, 2018. https://www.newtechdojo.com/list-machine-learning-algorithms/ (viděno pro. 20, 2020).

[24] „Unsupervised learning", *Wikipedia*. pro. 19, 2020, Viděno: pro. 20, 2020. [Online]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Unsupervised_learning&oldid=995165446.

[25] „Supervised learning", *Wikipedia*. pro. 17, 2020, Viděno: pro. 20, 2020. [Online]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Supervised_learning&oldid=994773800.

[26] F. Chollet, *Deep learning with Python*. Shelter Island, NY: Manning, 2018.

[27] „Optimization: Ordinary Least Squares Vs. Gradient Descent — from scratch | by Chayan Kathuria | Towards Data Science". https://towardsdatascience.com/https-medium-com-chayankathuria-optimization-ordinary-least-squares-gradient-descent-from-scratch-8b48151ba756 (viděno pro. 21, 2020).

[28] B. O. T. Ph.D, „Linear Regression Basics for Absolute Beginners", *Medium*, kvě. 29, 2020. https://medium.com/towards-artificial-intelligence/linear-regression-basics-for-absolute-beginners-68ed9ff980ae (viděno led. 18, 2021).

[29] P. Joshi, *Python Machine Learning Cookbook*, 1. vyd. Packt Publishing, 2016.

[30] T. Le, M. Luo, J. Zhou, a H. L. Chan, „Predictive maintenance decision using statistical linear regression and kernel methods", in *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, zář. 2014, s. 1–6, doi: 10.1109/ETFA.2014.7005357.

[31] „Python Machine Learning Polynomial Regression". https://www.w3schools.com/python/python_ml_polynomial_regression.asp (viděno led. 18, 2021).

[32] R. Farmani, K. Kakoudakis, K. Behzadian, a D. Butler, „Pipe Failure Prediction in Water Distribution Systems Considering Static and Dynamic Factors", *Procedia Eng.*, roč. 186, s. 117–126, led. 2017, doi: 10.1016/j.proeng.2017.03.217.

[33] Y. Liu, *Python machine learning by example: implement machine learning algorithms and techniques to build intelligent systems*, Second. Birmingham;Mumbai; Packt, 2019.

[34] J. Desmond, „Support Vector Machines (SVM) for AI Self-Driving Cars", *AI Trends*, led. 19, 2018. https://www.aitrends.com/ai-insider/support-vector-machines-svm-ai-self-driving-cars/ (viděno led. 18, 2021).

[35] M. das C. Moura, E. Zio, I. D. Lins, a E. Droguett, „Failure and reliability prediction by support vector machines regression of time series data", *Reliab. Eng. Syst. Saf.*, roč. 96, č. 11, s. 1527–1534, lis. 2011, doi: 10.1016/j.ress.2011.06.006.

[36] „ui-02-dectrees.pdf". Viděno: pro. 23, 2020. [Online]. Dostupné z: https://cw.fel.cvut.cz/old/_media/courses/a3m33ui/prednasky/ui-02-dectrees.pdf.

[37] Z. Allah Bukhsh, A. Saeed, I. Stipanovic, a A. G. Doree, „Predictive maintenance using tree-based classification techniques: A case of railway switches", *Transp. Res. Part C Emerg. Technol.*, roč. 101, s. 35–54, dub. 2019, doi: 10.1016/j.trc.2019.02.001.

[38] „kNN classifier". https://www.mathworks.com/matlabcentral/fileexchange/63621-knn-classifier (viděno led. 18, 2021).

[39] M. Cakir, M. A. Guvenc, a S. Mistikoglu, „The experimental application of popular machine learning algorithms on predictive maintenance and the design of IIoT based condition monitoring system", *Comput. Ind. Eng.*, s. 106948, říj. 2020, doi: 10.1016/j.cie.2020.106948.

[40] „Determining The Optimal Number Of Clusters: 3 Must Know Methods", *Datanovia*. https://www.datanovia.com/en/lessons/determining-the-optimal-number-of-clusters-3-must-know-methods/ (viděno pro. 08, 2020).

[41] „K-Means Clustering – What it is and How it Works – Learn by Marketing". http://www.learnbymarketing.com/methods/k-means-clustering/ (viděno led. 18, 2021).

[42] A. Abdelhadi, „HEURISTIC APPROACH TO SCHEDULE PREVENTIVE MAINTENANCE OPERATIONS USING K-MEANS METHODOLOGY", 2017. /paper/HEURISTIC-APPROACH-TO-SCHEDULE-PREVENTIVE-USING-Abdelhadi/54a146525f1e18134124574f81dfefd94848a0a3 (viděno pro. 27, 2020).

[43] „Explained: Neural networks", *MIT News | Massachusetts Institute of Technology*. https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414 (viděno pro. 27, 2020).

[44] „What is a Neural Network?", *Databricks*. https://databricks.com/glossary/neural-network (viděno led. 18, 2021).

[45] J. Křenek, K. Kuca, P. Blazek, O. Krejcar, a D. Jun, „Application of Artificial Neural Networks in Condition Based Predictive Maintenance", 2016, s. 75–86.

[46] Z. Kang, C. Catal, a B. Tekinerdogan, „Machine Learning Applications in Production Lines: A Systematic Literature Review", *Comput. Ind. Eng.*, roč. 149, s. 106773, srp. 2020, doi: 10.1016/j.cie.2020.106773.

[47] R. Agrawal a C. Nyamful, „Challenges of big data storage and management", *Glob. J. Inf. Technol.*, roč. 6, bře. 2016, doi: 10.18844/gjit.v6i1.383.

[48] MQ, „Lekce 2 - Úvod do práce se soubory v Pythonu". https://www.itnetwork.cz/uvod-do-prace-se-soubory-v-pythonu (viděno pro. 27, 2020).

[49] „What is DBMS? Application,Types,Example,Advantages". https://www.guru99.com/what-is-dbms.html (viděno pro. 27, 2020).

[50] A. Silberschatz, H. F. Korth, a S. Sudarshan, *Database system concepts*, Seventh edition. New York, NY: McGraw-Hill, 2020.

[51] „relational-databases", čer. 18, 2020. https://www.ibm.com/cloud/learn/relational-databases (viděno pro. 27, 2020).

[52] „Hadoop Vs Relational Databases", *Acheron Analytics*. http://www.acheronanalytics.com/2/post/2019/07/hadoop-vs-relational-databases.html (viděno pro. 27, 2020).

[53] „Top 6 Hadoop Vendors providing Big Data Solutions in Open Data Platform". https://www.dezyre.com/article/top-6-hadoop-vendors-providing-big-data-solutions-in-open-data-platform/93 (viděno pro. 27, 2020).

[54] „Hadoop (2) - základní součásti, souborový systém", *Blog Seznam.cz*, led. 23, 2012. https://blog.seznam.cz/2012/01/hadoop-2-zakladni-soucasti-souborovy-system/ (viděno led. 18, 2021).

[55] „SQL vs NoSQL Exact Difference (Know When To Use NoSQL and SQL)". https://www.softwaretestinghelp.com/sql-vs-nosql/ (viděno pro. 28, 2020).

[56] „On-Premise vs Cloud: Which is Right for Your Business?" https://phoenixnap.com/blog/on-premise-vs-cloud (viděno pro. 28, 2020).

[57] J. Pour, Z. Šedivá, M. Maryška, a I. Stanovská, *Self Service Business Intelligence: Jak si vytvořit vlastní analytické, plánovací a reportingové aplikace*, 1. elektronické vydání. Praha: Grada, 2018.

[58] „Dash Documentation & User Guide | Plotly". https://dash.plotly.com/ (viděno pro. 28, 2020).

[59] „Welcome to Flask — Flask Documentation (1.1.x)". https://flask.palletsprojects.com/en/1.1.x/ (viděno pro. 28, 2020).

[60] „Pricing & Product Comparison | Microsoft Power BI". https://powerbi.microsoft.com/en-us/pricing/ (viděno pro. 28, 2020).

[61] „Microsoft Power BI Features - Reasons Why Power BI is a Leader in its Field!", *DataFlair*, srp. 17, 2018. https://data-flair.training/blogs/power-bi-features/ (viděno pro. 28, 2020).

[62] „Themes Gallery". https://community.powerbi.com/t5/Themes-Gallery/bd-p/ThemesGallery (viděno led. 18, 2021).

[63] „10 Features of Tableau to Smoothen your Data Visualization Tasks", *DataFlair*, čer. 11, 2018. https://data-flair.training/blogs/tableau-features/ (viděno pro. 28, 2020).

[64] „6 real-world examples of business intelligence dashboards", *Tableau*. https://www.tableau.com/learn/articles/business-intelligence-dashboards-examples (viděno led. 18, 2021).

[65] „Bokeh 2.2.3 Documentation". https://docs.bokeh.org/en/latest/ (viděno led. 13, 2021).

[66] „scikit-learn: machine learning in Python — scikit-learn 0.24.0 documentation". https://scikit-learn.org/stable/index.html (viděno led. 14, 2021).

Seznam obrázků

Seznam tabulek

Příloha A: Dotaz do DB pro graf k obrázku 24

```
with cte as (
select
 month(DATEADD(D,Date,'1900-01-01')) as month
,[Eq]
,sum([TotalTime]) as TotalTime

FROM [DP].[dbo].[ProdData]
where year(DATEADD(D,Date,'1900-01-01')) > 2019 and TotalTime is not null and Cat in
(2,3,5,6)
group by [Eq],  month(DATEADD(D,Date,'1900-01-01'))
)
,cte1 as
(
SELECT
 month(DATEADD(D,Date,'1900-01-01')) as month
,[Eq]
,sum([TotalTime]) as StopTime
,case when [Cat] = 6 then 'BreakDown' else 'stop' end as stoptype

 FROM [DP].[dbo].[ProdData]
where year(DATEADD(D,Date,'1900-01-01')) > 2019 and TotalTime is not null and Cat in
(2,3,5,6)
group by [Eq],  month(DATEADD(D,Date,'1900-01-01')), case when [Cat] = 6 then
'BreakDown' else 'stop' end
)
select
 a.Eq
,a.month
,a.stoptype
,a.StopTime/b.TotalTime*100 as StopPercentage

from cte1 a
left join cte b on a.Eq=b.Eq and a.month = b.month
order by stoptype
```

Příloha B: Procedura dbo.PredictStops uložená v databázi DP na MS SQL serveru

```
 @Machine int
,@Reason int

with cte as
(
SELECT [Date]
      ,[Eq]
      ,[Order]
      ,[Daypart]
         ,case when SUM(case when Cat not in (7,8,4) then Time else 0 end)=0 then 0 else
          SUM(case when Cat in (2,3,5,6) and Stop = 1 and ReasonCode = @Reason
          then 1 else 0 end)/(SUM(case when Cat not in (7,8,4) then Time else 0
end)*1.0/(24*60*60))
          end as StopsDay
      ,case when SUM(TP) = 0 then 0 else SUM(RP)/SUM(TP)*100 end  as  RR
         ,SUM(case when Cat not in (7,8,4) then Time else 0 end)/60 as PlannedRunTime

FROM [DP].[dbo].[ProdData]
where Eq = @Machine and [order] is not null
group by [Date]
        ,[Eq]
        ,[Order]
        ,[Daypart]
)
,aa as
(
select distinct
 year(DATEADD(D,Date,'1900-01-01')) as Year
,case when month(DATEADD(D,Date,'1900-01-01')) <7 then 'H1' else 'H2' end as H
,PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY StopsDay)OVER(PARTITION BY
 year(DATEADD(D,Date,'1900-01-01')),case when month(DATEADD(D,Date,'1900-01-01')) <7
 then 'H1' else 'H2'end) AS StopsDayAVG

from cte
where StopsDay >1 and StopsDay <60
)
,cteOrder as
(
SELECT  distinct
[Order]
,PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY StopsDay)
 OVER(PARTITION BY [order]) AS StopsOrder
from cte where StopsDay >1 and StopsDay <60
)
,cte5 as
(
select
convert(date, (DATEADD(D,a.Date,'1900-01-01'))) as Date
,a.StopsDay
,case when d.StopsOrder is null then '4' when d.StopsOrder > StopsDayAVG*1.25 then '4'
when d.StopsOrder > StopsDayAVG*1.07 then '3' when d.StopsOrder < StopsDayAVG*0.75
then 0 when d.StopsOrder < StopsDayAVG*0.93 then 1 else 2 end as level
,a
,Future = 0
,case when LAG(d.StopsOrder,1) OVER (order by Date, Daypart) is null then '4'
when LAG(d.StopsOrder,1) OVER (order by Date, Daypart) > StopsDayAVG*1.25 then '4'
when LAG(d.StopsOrder,1) OVER (order by Date, Daypart) > StopsDayAVG*1.07 then '3'
```

```
when LAG(d.StopsOrder,1) OVER (order by Date, Daypart) < StopsDayAVG*0.75 then 0
when LAG(d.StopsOrder,1) OVER (order by Date, Daypart) < StopsDayAVG*0.93 then 1 else
2 end
as level1
,b,c,d,e,f,g,b.h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,aa,ab
,ac,ad,ae,af,ag,ah,ai,aj,ak,al,am,an,ao,ap,aq,ar,[as],[at],au,av
,aw,ax,ay,az,ba,bb,bc,bd,be,bf,bg,bh,bi,bj,bk,bl,bm,bn,bo,bp
,bq,br,bs,bt

from cte a
left join dbo.Materials b on a.[Order] = b.a
left join aa c on year(DATEADD(D,a.Date,'1900-01-01')) = c.Year and case when
month(DATEADD(D,a.Date,'1900-01-01')) <7 then 'H1' else 'H2' end = c.H
left join cteOrder d on a.[order] = d.[order]
WHERE a.StopsDay >1 and StopsDay <600
)
select * from cte5

Union all

select
 convert(date, (DATEADD(D,a.Date,'1900-01-01'))) as Date
,StopsDay = null
,level = null
,a
,Future = 1
,level1 = (select top(1) level from cte5 order by date desc)

,b,c,d,e,f,g,b.h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,aa,ab
,ac,ad,ae,af,ag,ah,ai,aj,ak,al,am,an,ao,ap,aq,ar,[as],[at],au,av
,aw,ax,ay,az,ba,bb,bc,bd,be,bf,bg,bh,bi,bj,bk,bl,bm,bn,bo,bp
,bq,br,bs,bt


from FutureProd a
left join dbo.Materials b on a.[Order] = b.[a]

order by date
```